Laboratorieøving 5 Datateknikk

Pragaash Mohan

April 14, 2019

Beskrivelse

Målet med prosjektet var å utfordre kreativiteten min, samtidig som å lage noe nytt og spennende. Til forarbeidet bestemte jeg meg hovedsaklig for å utnytte en SSD1306 OLED, en lyssensor og en buzzer. Jeg ville lære få frem flere forskjellige bilder på OLED basert på data avlest fra lyssesensoren og lage korte melodier fra buzzeren.

Den endelige avgjørelsen var å lage en animasjon av den beryktede Death Star, basert på Star Wars universet. Brukeren skal få en prompt gjennom konsollen, hvor en så starter animasjonen. Den massive stasjonen skulle loope seg gjennom noen enkle animasjoner, før den så basert på lyssensoren ville starte en ny animasjon. En rød LED vil blinke samtidig som en alarmlyd spilles fra buzzeren. Den nye animasjonen viser frem stasjonen lade opp superlaseren, etterfulgt av lyden av at våpenet lades spilles av buzzeren. Kort etter vil vi se den nærmeste planeten sprenges, etterfulgt av at stasjonen reiser videre. Animasjonen avsluttes med et endelig bilde på skjermen og en kort melodi fra buzzeren, før brukeren så promptes om programmet skal kjøres om igjen.

Animasjonen

Photoshop ble benyttet for å opprette bildene. De ble så konvertert til bitmapdata gjennom LCDAssistant, før de så ble benyttet i kodebiten. Følgende bilder ble opprettet:

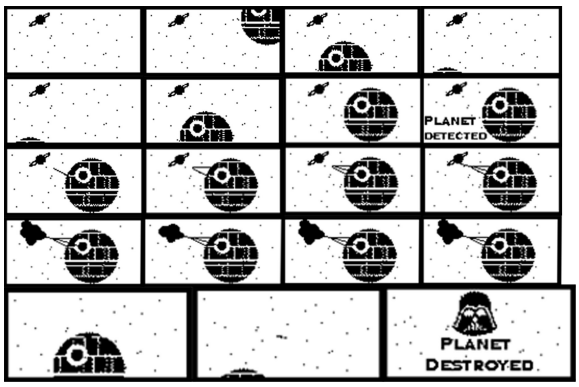

Figure 1: Bildesekvens

Melodi og photoresistor

Målet for piezo-buzzeren var å spille av en bit fra den kjente theme-sangen fra Star Wars, i dette tilfellet "Imperial March". Dette endte opp med å være overraskende mer krevende enn først tenkelig. En midi variant av sangen ble benyttet og kuttet ned gjennom Audacity. De uønskelige instrumentene ble fjernet, før så sangen ble konvertert til arduino-støttet format gjennom en nettside. Dette ble ikke like bra som ønskelig. Endelige løsningen var å gjøre dette manuelt. Fant de nødvendige tonefrekvensene over nettet, og brukte så et enkelt noteark til å kunne spille av ønsket bit.

Lyssensoren var programmert relativt simpelt. Gjennomsnittet av tre målinger ble utnyttet for å sikre god data.

Skjemategning og kodebit

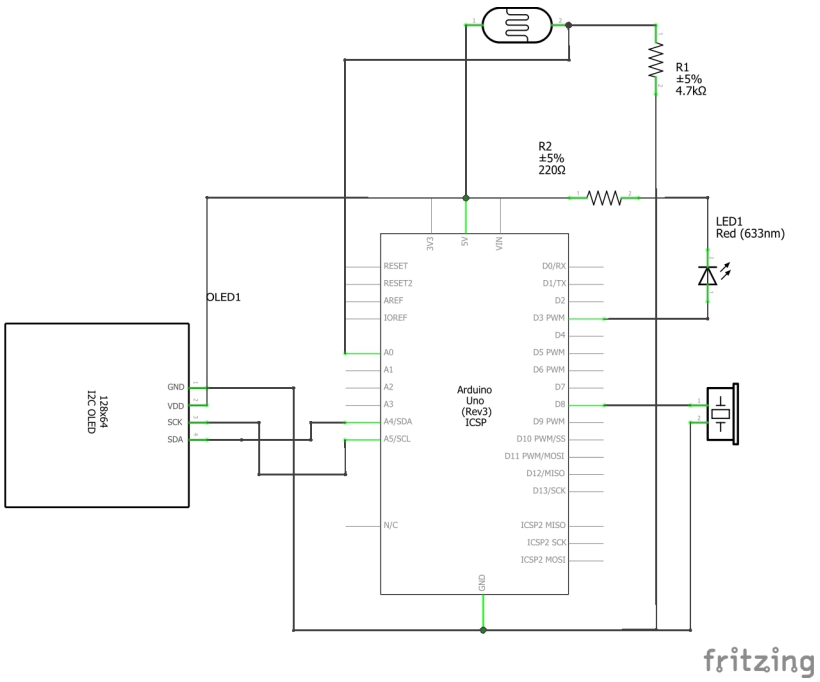

Figure 2: Skjemategning

Bitmap-data er ikke inkludert for å unngå overflod.

```
int redLed = 4;
boolean isRead = false;
const int sensorPin = A0;
int levelVal;

//Definerer n dvendige toner og hastighet
#define  LA3 220.00
#define  F3  174.61
#define  C4  261.63

#define BPM 120
#define H 2*Q
#define Q 60000/BPM
#define E Q/2
#define S Q/4

#include "U8glib.h"
U8GLIB_SSD1306_128X64 u8g(U8G_I2C_OPT_NONE | ...
    U8G_I2C_OPT_DEV_0);    // I2C / TWI



const uint8_t n1[] PROGMEM = {
//Bitmat—data
}

const uint8_t n3[] PROGMEM = {
}
const uint8_t n5[] PROGMEM = {
}
const uint8_t n6[] PROGMEM = {
}
const uint8_t n8[] PROGMEM = {
}
const uint8_t b1[] PROGMEM = {
}
const uint8_t b2[] PROGMEM = {
}
const uint8_t b3[] PROGMEM = {
}
const uint8_t b4[] PROGMEM = {
}
const uint8_t b5[] PROGMEM = {
}
const uint8_t b6[] PROGMEM = {
}
const uint8_t b7[] PROGMEM = {
}
const uint8_t b8[] PROGMEM = {
}
```

```
const uint8_t b[] PROGMEM = {
}
const uint8_t bf[] PROGMEM = {
}


void setup() {
  pinMode(8, OUTPUT);
  Serial.begin(9600);

  Serial.println("Welcome Commander.");
  Serial.println("Help the Imperial hunt the rebel scum. Use any ...
      lightsource you have to detect the");
  Serial.println("rebels hiding in one the planets. Press 's' to ...
      start.");
  Serial.println("Do not dissapoint.");
  clearOLED();
}



void loop(void) {


  if (Serial.available() > 0) {
    int readS = Serial.read();
    if (readS == 's' || readS == 'r') {
      isRead = true;
      Serial.println("Move the Death Star!");
      march();
      while (readAverage() < 700) {
        driveBy();
      }
      alert();
      shootEm();
      clearOLED();
      if (isRead == true) {
        Serial.println("Excellent work Commander. Press 'r' to ...
            repeat mission");
        isRead = false;
      }
    }
  }
}




void charge() {
  for (int i = 20; i < 500; i++) {
    tone(8, i);
    delay(10);
  }
}
```

```
1  void alert() {
2    for (int i = 0; i < 10; i++) {
3      digitalWrite(redLed, HIGH);
4      tone(8, 800);
5      delay(100);
6      digitalWrite(redLed, LOW);
7      delay(100);
8      noTone(8);
9    }
10
11 }
12
13 int readAverage() {
14
15   int count = 3;  // vi stiller inn l kka til    g  5 ganger.
16   int sensorVal = 0;  // nullstiller sensorvariabelen.
17
18   while (count > 0) {
19     // l per 3 ganger og gir summen av 3 sensoravlesninger.
20     sensorVal = sensorVal + analogRead(sensorPin);
21     count = count - 1;
22   }
23
24   sensorVal = sensorVal / 3; // beregner gjennomsnittet.
25   return sensorVal;     // returnerer gjennomsnittet.
26 }
27
28 int blink1() {
29
30 }
31 void draw(void) {
32   // graphic commands to redraw the complete screen should be ...
         placed here
33   u8g.drawBitmapP( 0, 0, 16, 64, n1);
34 }
35
36 void draw2(void) {
37   u8g.drawBitmapP( 0, 0, 16, 64, n3);
38 }
39
40 void draw3(void) {
41   u8g.drawBitmapP( 0, 0, 16, 64, n5);
42 }
43
44 void draw4(void) {
45   u8g.drawBitmapP( 0, 0, 16, 64, n6);
46 }
47
48 void draw5(void) {
49   u8g.drawBitmapP( 0, 0, 16, 64, n8);
50 }
51
52 void boom(void) {
53   u8g.drawBitmapP( 0, 0, 16, 64, b);
54 }
```

```
void boom1(void) {
  u8g.drawBitmapP( 0, 0, 16, 64, b1);
}

void boom2(void) {
  u8g.drawBitmapP( 0, 0, 16, 64, b2);
}

void boom3(void) {
  u8g.drawBitmapP( 0, 0, 16, 64, b3);
}

void boom4(void) {
  u8g.drawBitmapP( 0, 0, 16, 64, b4);
}

void boom5(void) {
  u8g.drawBitmapP( 0, 0, 16, 64, b5);
}

void boom6(void) {
  u8g.drawBitmapP( 0, 0, 16, 64, b6);
}

void boom7(void) {
  u8g.drawBitmapP( 0, 0, 16, 64, b7);
}
void boom8(void) {
  u8g.drawBitmapP( 0, 0, 16, 64, b8);
}

void boomf(void) {
  u8g.drawBitmapP( 0, 0, 16, 64, bf);
}
```

```
void driveBy(void) {
  u8g.firstPage();
  do {
    draw();
  } while ( u8g.nextPage() );
  delay(500);
  u8g.firstPage();
  do {
    draw2();
  } while (u8g.nextPage());
  delay(500);
  u8g.firstPage();
  do {
    draw3();
  } while (u8g.nextPage());
  delay(500);
  u8g.firstPage();
  do {
    draw4();
  } while (u8g.nextPage());
  delay(500);
  u8g.firstPage();
  do {
    draw5();
  } while (u8g.nextPage());
  delay(500);

}

void shootEm(void) {
  u8g.firstPage();
  do {
    draw4();
  } while ( u8g.nextPage() );
  delay(500);
  u8g.firstPage();
  do {
    draw3();
  } while ( u8g.nextPage() );
  delay(500);
  u8g.firstPage();
  do {
    boom();
  } while ( u8g.nextPage() );
  delay(1000);
  u8g.firstPage();
  do {
    draw3();
  } while ( u8g.nextPage() );
  delay(500);
```

```
  u8g.firstPage();
  do {
    boom();
  } while ( u8g.nextPage() );
  delay(1000);
  u8g.firstPage();
  do {
    boom1();
  } while ( u8g.nextPage() );
  delay(500);
  u8g.firstPage();
  do {
    boom2();
  } while ( u8g.nextPage() );
  delay(500);
  u8g.firstPage();
  do {
    boom3();
  } while ( u8g.nextPage() );
  charge();
  noTone(8);
  delay(500);
  u8g.firstPage();
  do {
    boom4();
  } while ( u8g.nextPage() );
  delay(500);
  u8g.firstPage();
  do {
    boom5();
  } while ( u8g.nextPage() );
  delay(300);
  u8g.firstPage();
  do {
    boom6();
  } while ( u8g.nextPage() );
  delay(300);
  u8g.firstPage();
  do {
    boom5();
  } while ( u8g.nextPage() );
  delay(300);
  u8g.firstPage();
  do {
    boom6();
  } while ( u8g.nextPage() );
  delay(300);
```

```
  u8g.firstPage();
  do {
    boom7();
  } while ( u8g.nextPage() );
  delay(500);
  u8g.firstPage();
  do {
    boom8();
  } while ( u8g.nextPage() );
  delay(500);
  u8g.firstPage();
  do {
    boomf();
  } while ( u8g.nextPage() );
  march();
  delay(200);

}

void march() {
  //tone(pin, note, duration)
  tone(8, LA3, Q);
  delay(1 + Q); //delay duration should always be 1 ms more than ...
      the note in order to separate them.
  tone(8, LA3, Q);
  delay(1 + Q);
  tone(8, LA3, Q);
  delay(1 + Q);
  tone(8, F3, E + S);
  delay(1 + E + S);
  tone(8, C4, S);
  delay(1 + S);

  tone(8, LA3, Q);
  delay(1 + Q);
  tone(8, F3, E + S);
  delay(1 + E + S);
  tone(8, C4, S);
  delay(1 + S);
  tone(8, LA3, H);
  delay(1 + H);


}

void clearOLED() { \\Empty screen
  u8g.firstPage();
  do {
  } while ( u8g.nextPage() );
}
```

Utføring av programmet

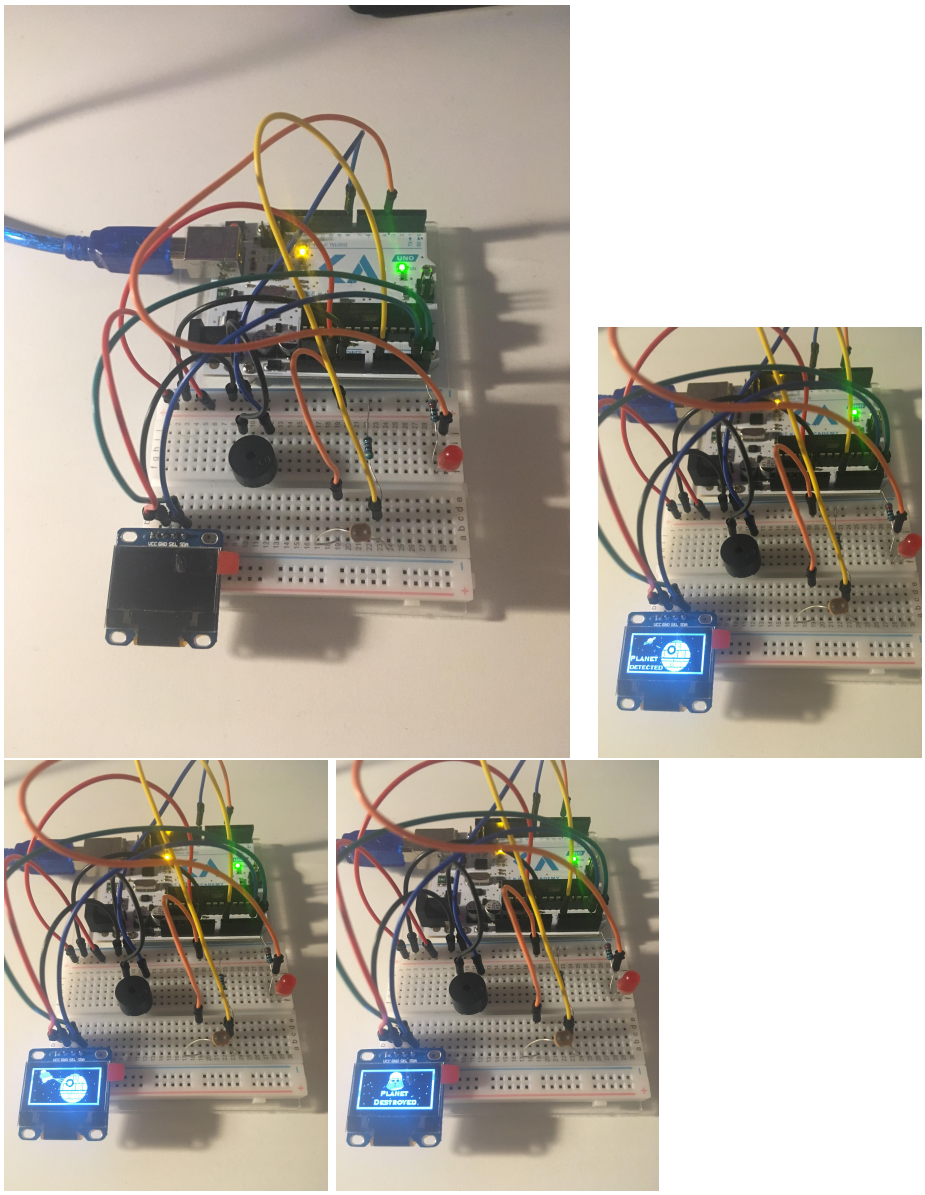

Figure 3: Oppkobling og utføring

Programmet kjørte som ønskelig, samtidig som de satte målene ble oppfylt. I helhet er jeg selv fornøyd med arbeidet.