ARDUINO PROJECT

아두이노를 이용해서 심령 장비 만들기

주제 선정

윤 시원이라는 고스트헌터를 한 매체를 통해 접했고 개인으로 활동하며 그가 여러 가지 과학 장비를 이용해서 시각적/청각적 심령 현상을 감지해 실시간으로 이러한 현상을 납득이 가능하게 보여주기 때문에 고스트 헌팅이라는 세계에 관심을 가지게 되었고 이 장비를 한번 만들어 보고 싶다는 생각에 이번 과제의 주제로 선정했다.

그가 사용하는 장비중 전자기장을 측정하여 5단계의 소리와 조명으로 나타내는 복합적인 장비이다. 하지만 전자기장을 측정하고 이를 증폭시키는 것 보다 온도 제어에 중점을 두고 다른 센서를 추가해 새로운 장비를 만들어 보려고 한다.

센서

INPUT 센서

온도센서

사용할 온도센서는 온도에 따라 전압이 변화하고 이 전압을 이용해서 우리는 온도를 알 수 있다.

초음파 센서

trig 와 echo로 초음파를 측정하는데 trig로 초음파를 쏘고 나서 echo로 되돌아 오면 소리가 전압으로 바뀌면서 돌아왔다고 알려주며 몇 초안에 돌아오는지에 따라 거리를 알 수 있다.

버튼

단순한 장치로 전압이 높은 곳에서 낮은 곳으로 흐르는 것을 이용해서 전압을 흘려 보내주는 장치이다.

센서

OUTPUT 센서

RGB LED

함께 전기 에너지를 빛 에너지로 변환시켜 전류가 흐를 때 빛을 내는 성질을 가진 소자입니다.

PIEZO BUZZER

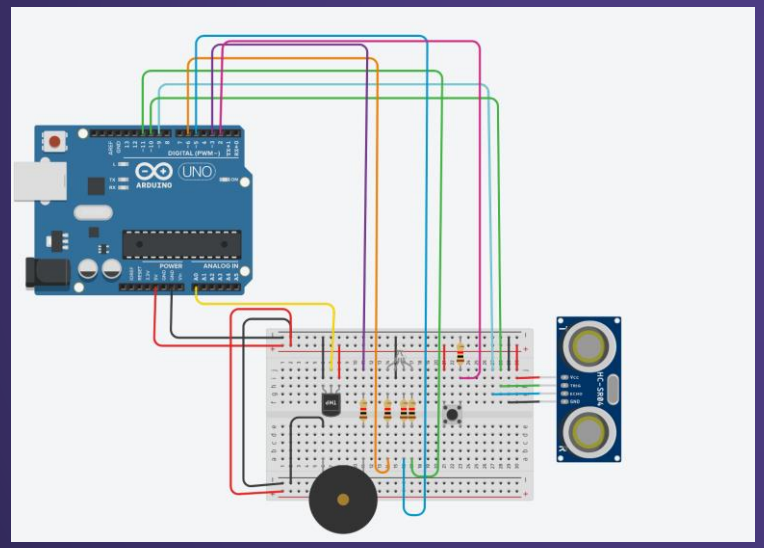
피에조 효과를 이용하여 소리를 내는 작은 스피커입니다.

(피에조 효과란 수정이나 세라믹 같은 결정체에 압력을 주게 되면 변형이 일어나면서 표면에 전압이 발생하고, 반대로 전압을 걸어주면 응축,신장을 하는 현상을 말하며 압전효과라고도 합니다.)

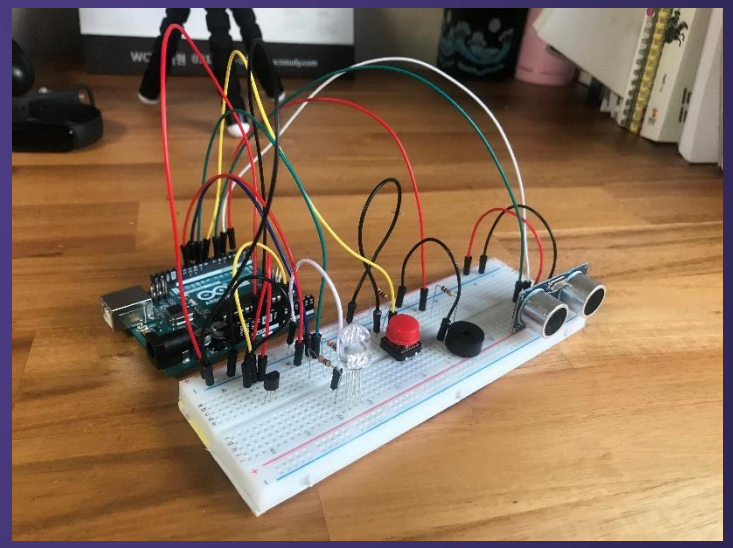
작업과정

하드웨어 구축

우선 시뮬레이터를 이용해서 가상으로 구현을 하고 이를 참고하여 실제 모델을 만들었습니다.



시뮬레이터



실제 모델

작업과정

소프트웨어 코드

마스터 모드

```
boolean debounse(boolean last)
  {
    boolean current=digitalRead(Button);
    if(last!=current)
    {
      delay(5);
      current=digitalRead(Button);
    }
    return current;
}
```

버튼의 스위치바운싱 현상을 막아주는 코드입니다.

```
currentButton=debounse(lastButton);

if(lastButton==LOW&&currentButton==HIGH)
{
  if(changemode==false){
    changemode=true;
  }
  else{
    changemode=false;
  }
  Serial.println("----------------change the mode----------------");
  analogWrite(GLED,255);
  analogWrite(BLED,255);
  analogWrite(RLED,255);
}
lastButton=currentButton;
```

그리고 위의 코드를 이용해서 모드를 바꾸어 주는 코드입니다.

슬레이브 모드

```
def debounse(last):
    current=board.digitalRead(Button)
    if last!=current:
        time.sleep(0.005)
        current=board.digitalRead(Button)
    return current
```

```
currentButton = debounse(lastButton)

if lastButton==0 and currentButton==1:
    if changeMode==False:
        changeMode=True
        board.analogWrite(GLED,255)
        board.analogWrite(BLED,255)
        board.analogWrite(RLED,255)
    else:
        changeMode=False
    print("------------------CHANGE MODE------------------")
lastButton=currentButton
```

위의 두 코드를 이용해서 버튼을 이용해서 온도를 측정하는 모드와 초음파를 측정하는 모드를 서로 분리해서 사용할 수 있도록 했습니다.

작업과정

소프트웨어 코드

마스터 모드 코드

```
if(changemode==false)
{
  delay(200);
  startTmp=analogRead(TMP);
  changeTMP=(startTmp)/9.28;
  Serial.print("start");
  Serial.print(temperature);
  Serial.print("change");
  Serial.println(changeTMP);

  if((changeTMP)<temperature-1.00&&(changeTMP)>temperature-3)
  {
    analogWrite(GLED,0);
    analogWrite(BLED,255);
    analogWrite(RLED,255);
    Serial.println("level 1");
  }
  else if((changeTMP)<temperature-3&&(changeTMP)>temperature-4)
  {
    analogWrite(BLED,0);
    analogWrite(GLED,255);
    analogWrite(RLED,255);
    Serial.println("level 2");
  }
  else if((changeTMP)<temperature-4)
  {
    analogWrite(RLED,0);
    analogWrite(GLED,255);
    analogWrite(BLED,255);
    Serial.println("level 3");
  }
  else
  {
    analogWrite(GLED,255);
    analogWrite(BLED,255);
    analogWrite(RLED,255);
  }
}
```

온도 제어 모드에서 사용하는 코드입니다

간단하게 처음 시작할 때 미리 받아놓고 나중의 온도와 비교해서 달라진 온도만큼 단계를 나눠 LED를 킵니다.

슬레이브 모드 코드

```
if changeMode==False:        #check temperature
    time.sleep(0.02)
    startTmp=board.analogRead(TMP);
    changeTMP=(startTmp)/2
    print "start temperature",temperature, "₩tchange temperature",changeTMP

    if changeTMP<temperature-1.0 and changeTMP>temperature-3.0:
        board.analogWrite(GLED,0) #GLED ON
        board.analogWrite(BLED,255)
        board.analogWrite(RLED,255)
        print("level 1")
    elif changeTMP<temperature-3.0 and changeTMP>temperature-5.0:
        board.analogWrite(BLED,0) #BLED ON
        board.analogWrite(GLED,255)
        board.analogWrite(RLED,255)
        print("level 2")
    elif changeTMP<temperature-5.0:
        board.analogWrite(RLED,0) #RLED ON
        board.analogWrite(GLED,255)
        board.analogWrite(BLED,255)
        print("level 3")
    else:
        board.analogWrite(RLED,255) #LED OFF
        board.analogWrite(GLED,255)
        board.analogWrite(BLED,255)
```

스케치에서 온도센서 값과 파이썬에서 온도센서 값이 다르게 나와서
마스터 모드에서는 changeTMP=(startTmp)/9.28로 현재 온도를 구하고
슬레이브 모드에서는changeTMP=(startTmp)/2로 현재 온도를 구합니다.
또한 잘 작동하는지 보기 위해 온도 차를 매우 작게 설정 했습니다.

작업과정

소프트웨어 코드

마스터 모드 코드

```
else if(changemode==true)
{
  digitalWrite(trigPin, LOW);
  digitalWrite(echoPin, LOW);
  delay(10);
  digitalWrite(trigPin, HIGH);
  delay(10);
  digitalWrite(trigPin, LOW);

  duration= pulseIn(echoPin,HIGH);
  distance = ((float)(340*duration)/10000)/2;

  Serial.print(distance);
  Serial.println("cm");
  delay(500);
  if(distance<30){
    Serial.println("ghost!");
    tone(piezo,500,1000);
  }

}
```

초음파 모드의 코드입니다

마스터 모드에서는 초음파를 쐈을 때 echo에서 자동으로 받아왔지만

슬레이브 모드에서는 echo에서 자동으로 받아 오지를 못해서 다른 방법을 이용해서 구현했습니다.

슬레이브 모드 코드

```
#echo를 받아오지 못함
#board.digitalWrite(trigPin,"HIGH")  #wave ON


#pulse_start = time.time()
#print("pulse_start", pulse_start)

#time.sleep(0.01)
# board.digitalWrite(trigPin,"LOW")   #wave OFF

#pulse_end = time.time()

#a = board.digitalRead(echoPin)
#b = board.digitalRead(trigPin)
#print('a:', a,"echo:",b)
#while board.digitalRead(echoPin)==0: #start time
#    pulse_end=time.time()
#    print("in while")
#print("out")

#board.digitalWrite(trigPin,"LOW")   #wave OFF
#print("pulse_end", pulse_end)

# while board.digitalRead(echoPin)==1:#end time
#     pulse_end=time.time()

#duration = pulse_end-pulse_start #duration time
#distance = duration*17000
#distance = round(distance,2)

#print(distance,"cm")

#if distance<100:    #in 100cm
#    board.digitalWrite(piezo,"HIGH")#piezo ON
#    time.sleep(0.5)
#    board.digitalWrite(piezo,"LOW")#piezo OFF
```

스케치에서는 pulseIn()이라는 함수를 이용해서 몇 초만에 초음파가 돌아왔는지 알 수 있었지만 파이썬에는 pulseIn()이라는 함수가 없어 while 문을 이용해서 구현을 했습니다.
하지만 마스터모드에서는 초음파가 돌아와 echo의 전압이 상승하는 것을 볼 수 있었지만 슬레이브 모드에서는 echo의 전압이 상승하지 않아 돌아오지 않는 현상이 발생했습니다.
그에 따라 다른 방법으로 이를 해결했습니다.

작업과정

소프트웨어 코드

마스터 모드 코드(prototype 타입)

```
#include <SoftwareSerial.h>
#include <Wire.h>
#include <Servo.h>
#include <EEPROM.h>
#include <Ultrasonic.h>
```

Ultrasonic.h을 임포트

```
void ultrasonicDistance(String data) {
   String sdata[2];
   split(sdata, 2, data, '%');
   int trigPin = sdata[0].toInt();
   int echoPin = sdata[1].toInt();

   Ultrasonic ultrasonic(trigPin,echoPin);
   Serial.println(ultrasonic.distanceRead(30));
}
```

위의 코드를 이용해서 시리얼 통신

슬레이브 모드 코드

```
from Arduino import Arduino
from Arduino.arduino import build_cmd_str
import time

class Ultrasonic(object):
    def __init__(self,board,trigPin=6,echoPin=5):
        self.board=board
        self.sr=board.sr
        self.trigPin=trigPin
        self.echoPin=echoPin

    def distanceMeasure(self):
        # return values are in centimeters
        cmd_str = build_cmd_str("us", (self.trigPin, self.echoPin))
        self.sr.write(cmd_str)
        self.sr.flush()
        rd = self.sr.readline().replace("\r\n", "")

        if rd.isdigit():
            return int(rd)
```

받아온 정보를 파이썬으로 읽음 및 출력 이때 build_cmd_str는 따로 임포트 받아서 사용(스트링을 만들어주는 함수)

```
elif changeMode==True:      #check wave

    us1=Ultrasonic(board,6,5)    #6, trigPin; 5, echoPin

    dis=us1.distanceMeasure()
    print "물체의 거리",dis, 'cm'

    if dis<10:
        board.digitalWrite(piezo,"HIGH")#piezo ON
        print "warning GHOST"
        time.sleep(0.02)
        board.digitalWrite(piezo,"LOW")#piezo OFF
    time.sleep(0.5)
```

이 받아온 정보를 토대로 거리에 따라 부저가 울리도록 설계

파이썬에서 초음파 쎈서가 작동하지 않아서 이를 해결하기 위해 초음파 센서를 아두이노 스캐치에서 작동시키고 시리얼 통신으로 보내 파이썬으로 읽어서 이를 해결했습니다.
그리고 잘 작동하는 지 보기위해 작동 범위를 짧게 설정했습니다.

결과

결과 영상 및 후기

결과 영상
  별도 첨부

후기
  처음에 기획한 방향으로 순탄하게 진행 되지않았다 특히 초음파 센서를 작동 시킬 때 마스터 모드에서는 수월하게 되었지만 슬레이브 모드를 진행 할때는 echo를 받아 오지 못해서 매우 진행이 더뎠고 결국 해결하지 못해 인터넷에서 찾은 방법으로 해결한 것이 너무 아쉬웠다. 하지만 그 이외의 것들은 수업에서 배운 걸 이용하고 스스로 방법을 찾아서 수월하게 진행했다.