REPORT

제 목 : Season project Report

과 목 명 : 임베디드 시스템
담당교수 : 장우영 교수님
소 속 : 전자전기공학부
학 번 : 32131539
이 름 : 정은우
제 출 일 : 2017년12월18일

단국대학교
Dankook University

1. 목표

바르지 않은 자세로 앉아있을 때 경고등을 표시하는 자세교정방석을 만든다.

2. 제작 이유

현대인들이 보통 하루 활동하는 시간의 반 이상 의자에 앉아 생활한다. 그러나 바른 자세로 앉아 오랫동안 유지하는 것은 힘들다. 앉아있다 보면 다리를 꼰 상태로 앉아있기도 하고 팔걸이나 등받이한쪽에 기대서 앉아있기도 한다. 이렇게 바르지 않은 자세로 오랫동안 앉아있게 되면 척추가 휘고 근육이 그대로 굳는 등 좋지 않다.

평소에 허리디스크가 있어 바른 자세로 앉으려고 노력하지만 정신차려보면 다시 구부정하게 혹은 삐딱하게 앉아있는 나 자신을 발견한다. 바른 자세로 앉아있지 않을 때 누군가 알려줬으면 하는 마음에서 만들게 되었다.

3. 기능

전원스위치를 통해 전원을 켜고 끌 수 있도록 한다. 압력센서 2개를 통해 양쪽 엉덩이의 무게를 측정하여 자세가 틀어져 있는지 인식한다. 자세가 기울어져 일정 이상 압력이 한쪽에 쏠리면 2개의 경고등을 이용하여 어느 방향으로 무게가 몰리는지 알려준다. 바른 자세로 앉아있을 경우 경고등은 작동하지 않는다. 자리에서 일어났을 경우 경고등에서 신호(무게)가 없음을 알려준다.

4. 이론

가. 압력센서(RA9P)

그림 1 압력센서

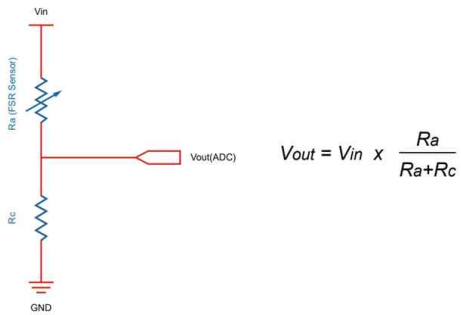

그림 2 압력센서 회로 구조도

압력센서 한쪽에 V_{in}을 가하고 다른 한쪽은 V_{out}이 나오는 구조이다. V_{out}에 저항과 GND를 연결하여 저항에 따라 전압 분배의 차이에 따라 압력의 정도를 데이터로 읽어올 수 있게 한 소자이다. V_{out}쪽에 저항의 크기값에 따라 V_{out}의결과 해상도가 달라진다. 그래서 저항 값을 소자에 사용가능한 범위 내에서 가장 큰 값을 연결했다. [그림3]을 참고하여 10Kohm을 사용했다. V_{out}을 계산하는 방법과 회로를 구성하는 방법은 [그림2]의 식과 같다.

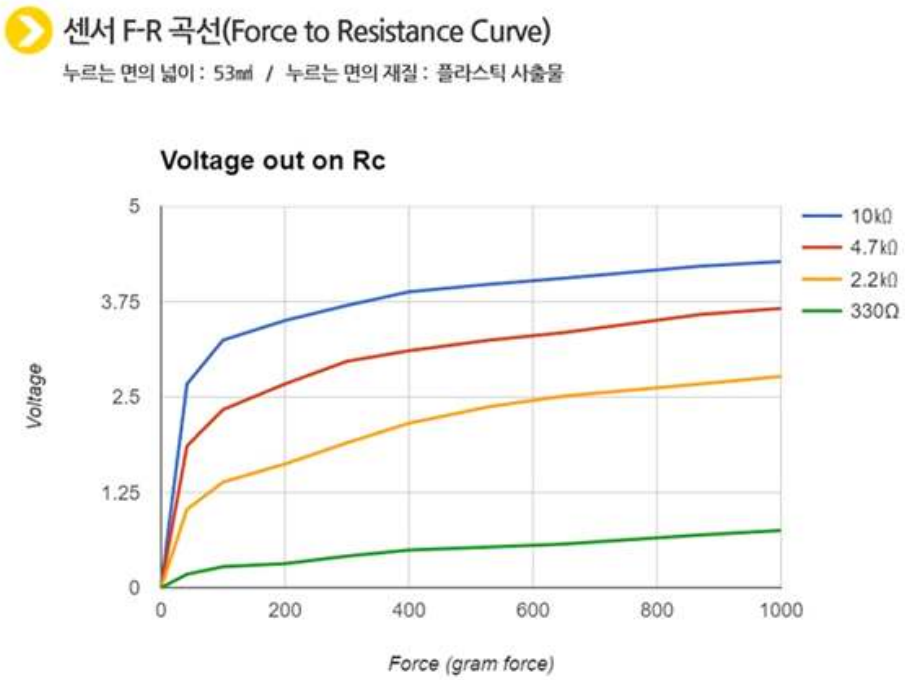

그림 3 센서 Force-Resistance 곡선

나. 회로 제작

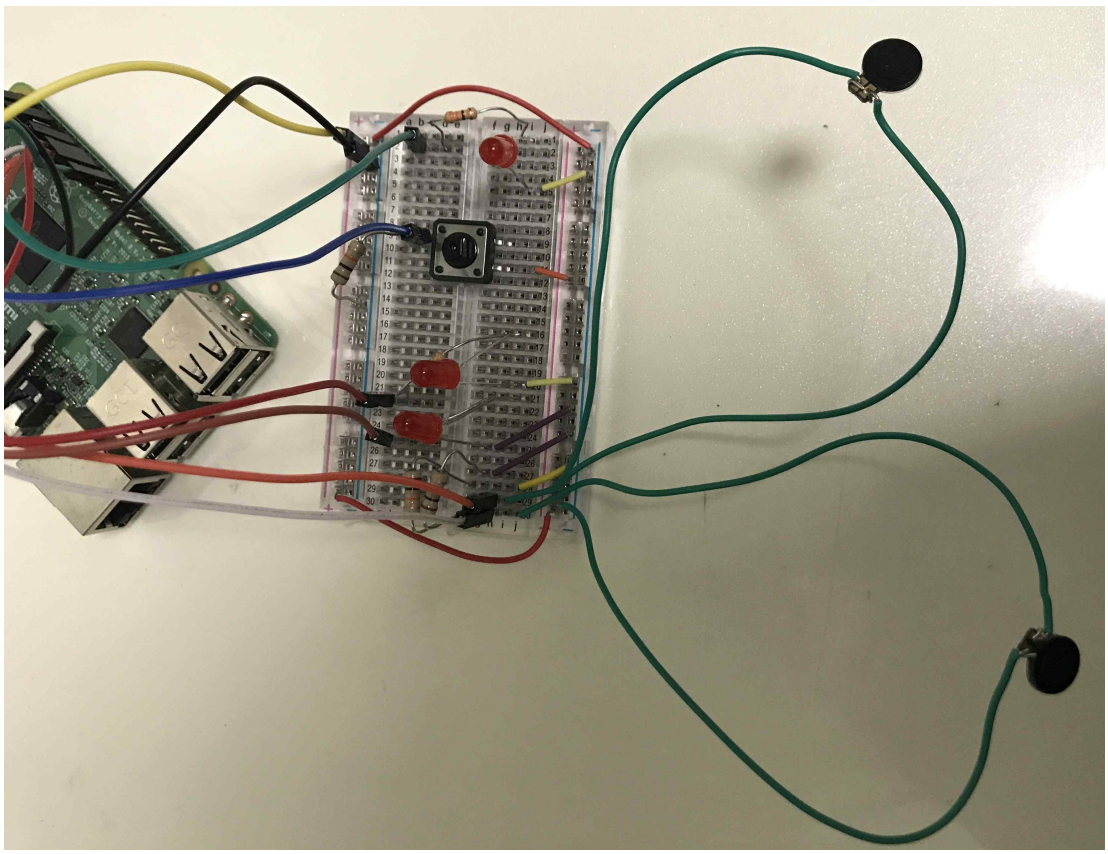

그림 4 회로사진

회로를 위와 같이 꾸며준다.

1) LED는 GPIO에서 나와 저항(330ohm)을 지나 LED를 거쳐 GND로 가게 한다.

2) 스위치는 양쪽에 핀이 나오게 두면 아래 부분2핀과 윗부분2핀이 각각 연결되어있고 위와 아래는 스위치로 연결되어있는 구조이다. 아래 핀에는 +5V를 연결했고 위 핀에는 GPIO와 저항(10Kohm)을 연결했고 저항반대편에는 GND가 연결되어있다.
3) 압력센서는 +5V(V_{in}) 을 지나 압력센서 가있고 반대편에 GPIO핀과 저항(10Kohm)이 연결되어있고 저항 반대편에는 GND가 연결되어있다.

다. GPIO 설정

1) +5V : 2번 핀을 사용
2) GND(0V) : 6번 핀 사용
3) GPIO 2 : 압력센서 왼쪽 사용
4) GPIO 3 : 압력센서 오른쪽 사용
5) GPIO 4 : LED 왼쪽 사용
6) GPIO 5 : LED 오른쪽 사용
7) GPIO 6 : 전원 LED 사용
8) GPIO 1 : 파워 스위치 사용

그림 5 홀수번 PIN 모습

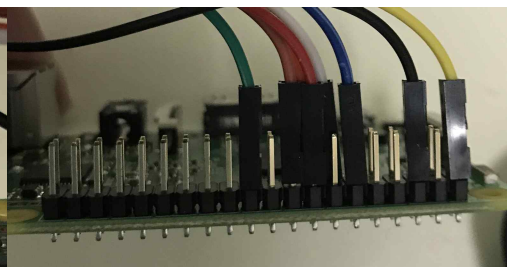

그림 6 짝수번 PIN 모습

5. 코드

가. 1차 작성(인터럽트 스위치 제외)

```
#include <stdio.h>
#include <wiringPi.h>

#define RAL 2 // BCM_GPIO 27
#define RAR 3 // BCM_GPIO 22
#define LEDL 4 // BCM_GPIO 23
#define LEDR 5 // BCM_GPIO 24

int main (void)
{
      int heavy;
  int diff;
  if(wiringPiSetup () ==-1)
       return 1;

  pinMode (LEDL, OUTPUT);
  pinMode (LEDR, OUTPUT);
  pinMode (RAL, INPUT);
  pinMode (RAR, INPUT);

  while(1)
  {
    while(digitalRead(RAL)==0 && digitalRead(RAR)==0)
    {
      digitalWrite(LEDL,1);
      digitalWrite(LEDR,1);
    }
    diff = digitalRead(RAL)-digitalRead(RAR)

    if(diff>0)
    {
      heavy=digitalRead(RAL)-digitalRead(RAR);
      digitalWrite(LEDL, heavy);
      digitalWrite(LEDR, 0);
    }
    else if(diff<0)
    {
      heavy=digitalRead(RAR)-digitalRead(RAL);
      digitalWrite(LEDR, heavy);
      digitalWrite(LEDL, 0);
    }
    else
    {
      digitalWrite(LEDL, 0);
      digitalWrite(LEDR, 0);
```

```
    }
  }
  return 0;
}
```

나. 2차 작성(인터럽트 스위치 포함)

```
#include <stdio.h>
#include <wiringPi.h>

#define RAL 2 // BCM_GPIO 27
#define RAR 3 // BCM_GPIO 22
#define LEDL 4 // BCM_GPIO 23
#define LEDR 5 // BCM_GPIO 24
#define LEDP 6 // BCM_GPIO 25
#define PW_SW 1 // BCM_GPIO 18

volatile int POWER_state = 0;
volatile int POWER_change = 0;

void powerPressed(void)
{
      if(POWER_state==0)
      POWER_state = 1;
      else
      POWER_state = 0;//POWER_state가 0이면 1을 넣고 1이면 0을 넣는다.
      POWER_change = 1; //POWER_change를1로 변경한다.
}

int main (void)
{
      int heavy;
  int diff;
  if(wiringPiSetup () ==-1)
       return 1;

  pinMode (LEDL, OUTPUT);
  pinMode (LEDR, OUTPUT);
  pinMode (RAL, INPUT);
  pinMode (RAR, INPUT)
      pinMode (LEDP, OUTPUT);
      pinMode (PW_SW, INPUT);

      wiringPiISR(PW_SW,INT_EDGE_RISING,powerPressed);

  while(1)
  {
                POWER_change=0;
```

```
        digitalWrite(LEDP, POWER_state);
        while(POWER_state==1)
        {
                digitalWrite(LEDP,1);
                while(digitalRead(RAL)==0 && digitalRead(RAR)==0)
    {
      digitalWrite(LEDL,1);
      digitalWrite(LEDR,1);
                        delay(1000);
                        digitalWrite(LEDL,0);
      digitalWrite(LEDR,0);
                        delay(1000);
    }
                while(digitalRead(RAL)==0 && digitalRead(RAR)==0)
    {
                        diff = digitalRead(RAL)-digitalRead(RAR)

                        if(diff>0)
                        {
                                heavy=digitalRead(RAL)-digitalRead(RAR);
                                digitalWrite(LEDL, heavy);
                                digitalWrite(LEDR, 0);
                        }
                        else if(diff<0)
                        {
                                heavy=digitalRead(RAR)-digitalRead(RAL);
                                digitalWrite(LEDR, heavy);
                                digitalWrite(LEDL, 0);
                        }
                        else
                        {
                                digitalWrite(LEDL, 0);
                                digitalWrite(LEDR, 0);
                        }
                }
        }
        while(POWER_state==0)
        {
                digitalWrite(LEDL, 0);
                digitalWrite(LEDR, 0);
        }
  }
  return 0;
}
```

6. 라즈베리파이 구동 결과 영상 링크

https://www.youtube.com/watch?v=HPD1qHqdIPs

7. 고찰

처음 프로젝트를 수행할 때 스위치를 만들 생각을 하지 않았다. 압력센서를 익숙해지는 것에 중점을 뒀고 LED와 센서가 잘 동작하도록 하는 것에 초점을 두었다. 처음에 digitalRead(혹은 Write)함수 대신에 analogRead(혹은 Write)함수를 쓰려고 했으나 그런 함수는 존재하지 않았다는 것을 알았다. Python에서만 구현이 되어있는 것을 알았다. 그래서 digitalRead를 사용하여 센서로 받는 입력을 1과 0으로 만들어 아쉬움이 있었으나 만약 아날로그 값으로 받았다면 무게차이를 조절하는 것을 한번더 시행착오를 격어야 했다. digital이 아닌 analotWrite로 작성하여 LED의 밝기를 조절하려고 했다. 그러나 압력센서와 LED구동이 성공적으로 이루어졌고 다음단계로 넘어갈 수 있었다.

회로를 이용자 친화적으로 바꾸려고 했다. 압력센서와 LED가 앞쪽으로 나오고 GND 점프선과 저항등이 뒤로가는 등 구조를 사용자가 편하도록 조정했다. 압력센서가 방석에 장착되는 역할을 해야 하므로 긴 점프선을 이용해 압력센서를 연결했다.

그 다음엔 인터럽트를 사용하여 전체 전원을 켜고 끄는 스위치를 제작하려 했으나 전원을 켜는 것은 구현했지만 전원이 켜져있는 것을 끄는 것은 구현에 실패했다. 전원이 켜져있는 상태에서 버튼을 다시 누르면 RESET 역할을 했다. wiringPiISR(PW_SW, INT_EDGE_RISING, powerPressed); 함수와 POWER_state, POWER_change변수를 이용해 인터럽트를 사용하려 했지만 실패했다. wiringPiISR함수에서 맨 앞은 핀번호를 쓰고, 가운데는 신호가 어떻게 반응할 때 동작할지 쓰는 곳이고 맨 마지막은 수행할 함수를 쓰는 칸이다. 그래서 잘 사용했다고 생각했는데 결국엔 실패했다.

마지막까지 코드를 수정했지만 전원을 종료하는 기능 외에 모든 동작은 다 제대로 잘 작동했다.

8. 참고자료

- http://min81love.blog.me/221135222816
- http://cafe.naver.com/gimbujang/1658
- http://cafe.naver.com/circuitsmanual/153655
- http://www.eleparts.co.kr/EPXHKW7F