|  |  |  |
| --- | --- | --- |
| 컴퓨터시스템입문 프로젝트 결과물 보고서  : 알람 로봇(Alam-Bot) 만들기 | | |
|  | | |
| 2조 | 201600517 | 김성민 |
| 201600765 | 김주원 |
| 201600903 | 김태준 |
| 201601185 | 박기웅 |
| 201603234 | 조규민 |
| 201603727 | 허성일 |

<목차>

1. 주제………………………………………………………………………………………………………………………………………...2
2. 동기………………………………………………………………………………………………………………………………………...3
3. 결과물 제조 과정…………………………………………………………………………………………………………………...3

-사용제품………………………………………………………………………………………………………………………………...4

-회로도 구성…………………………………………………………………………………………………………………………...5

-코드 작성시 발생한 문제점과 해결과정………………………………………………………………………………5

-완성된 결과물………………………………………………………………………………………………………………………..6

1. 소스코드 …………………………………………………………………………………………………………………………………6
2. 결론 & 조원들 소감……………………………………………………………………………………………………………..15
3. 추가 참고한 자료 사이트 및 참고한 자료…………………………………………….…………………………....17

**컴퓨터시스템입문 프로젝트 결과물 보고서 : 알람 로봇(Alam-Bot) 만들기**

1. 주제: 타이머 값(1~30분)을 설정하여 멜로디를 재생하는 아두이노 로봇을 만들어 보았다.
2. 동기: ‘아두이노 기말 프로젝트 과제’를 제출하기 위해 일차적으로 시험기간 전 5월 28일 아이디어 회의를 진행했다. 회의 진행과정에서 ‘엘리베이터 만들기’, ‘RC자동차’, ‘드론제작’, ‘OLEDX서보모터 로봇’등 다양한 의견이 나왔다. 그 중에서 ‘OLEDX서보모터 로봇’을 선정하였다.

선정한 이유는 다음과 같다.

1. **새로운 기기 활용**- 기말 프로젝트 과제를 시작하기 전 조의 다수의 학생들이 기본 키트 외의 제품을 사용하고 싶어했다. 따라서 조원들 협의 하에 수업 이외의 추가 연구 자료와 수업 때 배운 내용을 결합하여 결과물을 만들자는 목표를 설정하였다. OLEDX서보모터 로봇은 ‘OLED LCD 128X64 모듈’, ‘서보 마운트’ 등 수업 때 배우지 않은 기기들을 활용했기에 이러한 목표에 적합하고 추가 연구하기에도 알맞은 난이도라고 판단하기 때문에 선정하게 되었다.
2. **강의시간에 배웠던 실습 활용 가능성**- 다른 주제 ‘RC자동차’나 ‘드론 제작’과 같은 주제는 기존 인터넷 자료를 연구해본 결과 조원들이 가지고 있는 역량으로는 수업 때 배웠던 내용을 추가하기에 어려움이 있다고 판단하였다. 반면 OLEDX서보모터는 가변저항, 거리센서를 이용하여 움직임을 제한하거나 서보 모터 움직임을 변화 시키는 등의 코딩을 할 수 있었기에 강의실습시간때 배웠던 내용을 효과적으로 적용할 수 있을 거라고 판단했다. 추가로 LCD화면을 여러 내용을 출력하면 기능을 추가시킬 수 있을 거라고 생각해 이 주제를 선정하게 되었다.
3. 결과물 제조 과정

결과물을 만들기 참고한 자료는 단순이 서보 모터가 위아래로 움직이고 화면이 출력되는 형태의 단순한 모델이었다. 따라서 우리는 이 로봇을 활용하여 새로운 기능, 타이머 기능을 추가하기로 결정하였다.

초반에는 if문과 3~4개에서 스위치를 이용하여 10분, 20분, 30분 등의 시간으로 타이머를 설정하려고 했으나 구현할 수 있는 타이머 시간대가 적은 문제점을 발견하였다.

따라서 가변저항을 활용함으로써 다양한 시간대를 구현할 수 있었고 서보 모터의 움직임을 변화를 주어 로봇의 움직임을 바뀌었다. 또한 버저와 tonemelody예제, 그리고 캐논 변주곡을 이용하여 알람 음을 만들었다.

그 결과 1~30분 까지의 타이머 기능을 가지고있는 알람 음을 내는 앙증맞은 알람봇(Alarm-Bot)을 만들게 되었다.

-사용제품

|  |
| --- |
| 품목 |
| [SG90 서보모터 (Arduino SG-90 Mini Servo Motor)](http://mechasolution.com/shop/goods/goods_view.php?goodsno=71795) 2개  아두이노 우노 R3 호환보드 (Arduino uno R3) 1개  SG90 서보모터 / 카메라 팬틸트 프레임 거치대 1개  (Servo Controlled Camera Pan-tilt unit SG90S MG90S Compatible)  브레드 보드 1개,  아두이노 키트 기본 전선  아두이노 보드 연결 잭 1개  [아두이노 12864 OLED LCD 모듈 6핀 1개](http://mechasolution.com/shop/goods/goods_view.php?goodsno=540459)  [(0.96" inch SPI Communication 12864 OLED LCD Module)](http://mechasolution.com/shop/goods/goods_view.php?goodsno=540459)  [40핀 커넥터 F-M 20cm (Female - Male), 점퍼케이블 / 점퍼선](http://mechasolution.com/shop/goods/goods_view.php?goodsno=514)  [40핀 커넥터 M-M 10cm (Male - Male), 점퍼케이블 / 점퍼선](http://mechasolution.com/shop/goods/goods_view.php?goodsno=9681)  가변 저항 1개  스위치(switch) 1개  버저(Buzzer) 1개  하드보드지 1장 |

|  |
| --- |
| 사진 |
|  |

-회로도 구성

|  |
| --- |
| 사진 |
| C:\Users\김주원\AppData\Local\Microsoft\Windows\INetCacheContent.Word\20170621_185519.jpg |

-코드 작성시 생긴 문제점과 해결과정

1)헤더파일

시간을 측정하는 함수(swRTC.h), I2C를 연결을 할 수 있도록 만드는 헤더파일(wire.h), OLED LCD 그래픽을 구현할 수 있도록 하는 헤더 파일(Adafruit\_GFX.h), OLED LCD 128X64 헤더 파일(Adafruit\_SSD1306.h), 등이 깔려 있지 않아 초반에 코드가 돌아가지 않았다. 따라서

아두이노 ‘스케치’ 창에서 라이브러리 포함하기->라이브러리 관리->라이브러리 매니저 를 통해 헤더파일을 다운받음으로 문제를 해결하였다.

2) 풀 업 저항의 이용

초반에 스위치를 이용하여 알람을 키고 끌 수 있도록 하였는데, 스위치 클릭 이후 계속 화면이 원하지 않는 결과가 출력되고 있었다. 따라서 문제를 해결하기 위해서 풀 업 저항을 이용하였다. HIGH일 때에는(스위치 Off)일 때에는 일반적인 로봇의 모습이 출력될 수 있도록 하고, LOW일 때에는 (스위치 On)일 때에는 타이머가 작동할 수 있도록 만들었다.

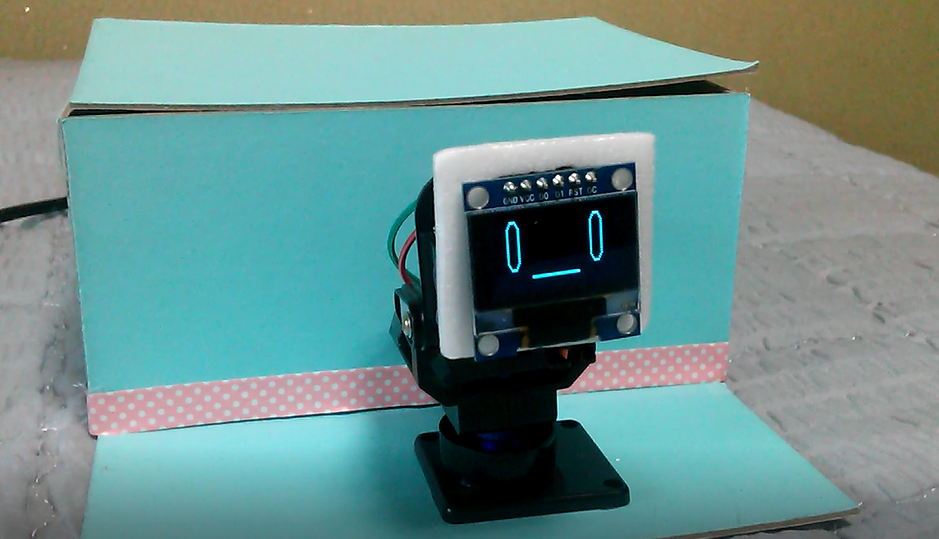
1. 클래스 선언 이용

아두이노 소스 코드가 C++혹은 C언어의 일부 문법을 포함하고 있다는 사실을 알고 있었는데 클래스를 선언할 수 있다는 사실을 알지 못하였고, 웹 상의 아두이노 관련 소스코드를 분석하여 이번과제에 쓰이는 소스코드를 이해하고 추가적으로 설계해 나갈 수 있었다.

1. 가변저항 이용

‘가변저항 값=사용자가 설정할 수 있는 타이머 시간’로 표현하려고 하였다. 그러나 가변저항의 값은 1024칸인 아날로그 형태로 값을 출력해내기 때문에 원하는 값(Maximum 30분)을 만들기가 쉽지 않았다. 따라서 1024/34=30.11…임으로 int형 정수 변수로 LCD의 값을 출력할 경우 0~30까지 구현 할 수 있으므로 analog값을 34로 나누어주어서 0~30분을 구현해냈다.

-완성된 결과물(사진)



1. 소스코드

소스코드 설명은 주석을 통해 서술했음을 밝힌다.

|  |
| --- |
| Alarmbot소스코드 |
| #include <gfxfont.h>  #include <Servo.h>//서보 모터 헤더 파일  #include <swRTC.h>//시간 측정 헤더 파일  #include <Wire.h>//I2C device를 이용할 수 있도록 만든 함수  #include <Adafruit\_GFX.h>//OLED LCD 그래픽 헤더 파일  #include <Adafruit\_SSD1306.h>//OLED LCD 128X64 헤더 파일  #include "pitches.h"//음계 헤더 파일  int Canonmelody[] = {  NOTE\_G7, 0, NOTE\_E7, NOTE\_F7, NOTE\_G7, 0, NOTE\_E7, NOTE\_F7, NOTE\_G7, NOTE\_B6, NOTE\_A6, NOTE\_B6, NOTE\_C7, NOTE\_D7, NOTE\_E7, NOTE\_F7, NOTE\_E7, 0, NOTE\_C7, NOTE\_D7, NOTE\_E7,0, NOTE\_E6, NOTE\_F6, NOTE\_G6, NOTE\_A6, NOTE\_G6  , NOTE\_F6, NOTE\_G6, NOTE\_C7, NOTE\_B6, NOTE\_C7, NOTE\_A6, 0, NOTE\_C7, NOTE\_B6, NOTE\_A6, 0, NOTE\_G6, NOTE\_F6, NOTE\_G6, NOTE\_F6, NOTE\_E6, NOTE\_F6, NOTE\_G6, NOTE\_A6, NOTE\_B6, NOTE\_C7, NOTE\_A6, 0, NOTE\_C7, NOTE\_B6, NOTE\_C7, 0,  NOTE\_B6, NOTE\_C7, NOTE\_B6, NOTE\_A6, NOTE\_B6, NOTE\_C7, NOTE\_D7, NOTE\_E7, NOTE\_F7, NOTE\_G7,  };//캐논 음계 함수  int noteDurations[] = {  6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6  };//캐논 박자 함수  int buzzer = 8;//버저 핀 변수선언  int button = 2;//스위치 핀 변수 선언  class AlarmBot //알람봇 클래스 선언  {  public:  Adafruit\_SSD1306 \*display; //Adafruit에서 제공하는 클래스의 포인터 객체를 선언해준다  Servo servo1; //서보 모터1  Servo servo2; //서보 모터2  int servo1pin, servo2pin; //서보 모터 1, 2 가 연결된 핀  AlarmBot(Adafruit\_SSD1306 \*\_display, int \_servo1pin, int \_servo2pin)  //클래스의 생성자에서 멤버 변수들을 초기화해준다  {  display = \_display;  servo1pin = \_servo1pin;  servo2pin = \_servo2pin;  };  void init() //화면에 알람 로봇의 기본 얼굴이 나오게 하는 함수  {  display->begin(SSD1306\_SWITCHCAPVCC);  display->clearDisplay(); //화면을 클리어함  display->drawLine(display->width() \* 2 / 6, display->height() \* 4 / 5, display->width() \* 4 / 6, display->height() \* 4 / 5, WHITE);  //얼굴을 그리기 위해 drawLine함수를 불러와 해당 함수에 인자들을 넣어줌으로써 선을 그린다  eyeClose(); //미리 정의한 눈을 감는 모션을 화면에 출력하는 함수를 호출  servo1.attach(servo1pin); //서보모터1 연결  servo2.attach(servo2pin); //서보모터2 연결  servo1.write(90); //서보 모터가 90도를 유지하도록한다  servo2.write(90);  \_x = 0;  \_y = 0;  delay(1000); //1초 딜레이  servo1.detach();  servo2.detach();  }  void eyeClose() //눈을 감는 모션을 화면에 출력하는 함수  {  for (int i = 0; i < 10; i+=2) //총 5번을 도는 반복문. 천천히 눈이 감기도록 하기 위해 써주었다.  {  display->drawRoundRect(display->width() \* 1 / 5 - 5, display->height() / 3 - 10 + i , 11, 22 - i \* 2, 5 - i/2, WHITE);  //drawRoundRect함수를 이용하여 for문 한바퀴 돌 때마다 다른 인자를 넣어주어 천천히 감기는 눈을 구현  display->drawRoundRect(display->width() \* 4 / 5 - 5, display->height() / 3 - 10 + i , 11, 22 - i \* 2, 5 - i/2, WHITE);  display->display(); //display()함수를 써주어야만 화면에 출력된다  delay(30);  display->drawRoundRect(display->width() \* 1 / 5 - 5, display->height() / 3 - 10 + i , 11, 22 - i \* 2, 5 - i/2, BLACK);  //하얀 색으로 칠한 부분을 출력 후 검은색으로 덮어씌워 점점 감기는 눈을 구현하였다.  display->drawRoundRect(display->width() \* 4 / 5 - 5, display->height() / 3 - 10 + i , 11, 22 - i \* 2, 5 - i/2, BLACK);  }  display->drawLine(display->width() \* 4 / 5 - 5, display->height() / 3, display->width() \* 4 / 5 + 5, display->height() / 3, WHITE);  //눈이 모두 감긴 이후에는 -\_- 이 모양이 되므로 이 때는 눈을 선으로 나타낸다.  display->drawLine(display->width() \* 1 / 5 - 5, display->height() / 3, display->width() \* 1 / 5 + 5, display->height() / 3, WHITE);  display->display();  display->drawLine(display->width() \* 4 / 5 - 5, display->height() / 3, display->width() \* 4 / 5 + 5, display->height() / 3, BLACK);  //눈을 감은 얼굴을 출력한 뒤 다음 출력에서 다시 검은색으로 덮어씌워 다른 출력이 보일 수 있도록 한다.  display->drawLine(display->width() \* 1 / 5 - 5, display->height() / 3, display->width() \* 1 / 5 + 5, display->height() / 3, BLACK);  }  void eyeOpen() //눈을 뜨는 모션을 화면에 출력하는 멤버 함수.  {  for (int i = 10; i >= 0; i-=2) //천천히 눈을 뜨는 모션을 보여주기 위해 for문을 이용하였다.  {  display->drawRoundRect(display->width() \* 1 / 5 - 5, display->height() / 3 - 10 + i , 11, 22 - i \* 2, 5 - i/2, WHITE);  display->drawRoundRect(display->width() \* 4 / 5 - 5, display->height() / 3 - 10 + i , 11, 22 - i \* 2, 5 - i/2, WHITE); //drawRoundRect함수에 인자를 for문을 한 번 돌 때마다 다른 인자를 넣어줌으로서 천천히 눈이 뜨는 모션을 보여줌  display->display();  delay(30);  display->drawRoundRect(display->width() \* 1 / 5 - 5, display->height() / 3 - 10 + i , 11, 22 - i \* 2, 5 - i/2, BLACK); //스탑 모션으로 눈이 떠지는 걸 구현하기 위해 흰색으로 출력한 부분에 검은색으로 덮어씌워  display->drawRoundRect(display->width() \* 4 / 5 - 5, display->height() / 3 - 10 + i , 11, 22 - i \* 2, 5 - i/2, BLACK); //다음 스탑모션에서 더 큰 RoundRectangle을 그려 눈이 점점 떠지는 것 처럼 구현하였다.  }  }  void Blink() //눈이 깜박이는 모션을 구현한 함수  {  eyeClose(); //우선 눈을 감고  delay(300); //0.3초 딜레이 후  eyeOpen(); //눈을 다시 뜬다  }  void See(int x, int y) //서보 모터를 움직여 x, y에 따라 쳐다보는 시선이 달라지도록 하는 함수  {  servo1.attach(servo1pin); //각 서보모터를 연결한다  servo2.attach(servo2pin);  while (\_x != x || \_y != y)  //init에서 초기화한 \_x, \_y를 바탕으로, x와 \_x가 다르거나, y와 \_y가 다를 경우  //while문 조건을 만족하여 반복문이 돌게 된다.  {  servo1.write(90 + (x != \_x ? (x < 0 ? \_x-- : \_x++) : \_x));  //while문 안에서 \_x 와 \_y값이 바뀌고, 이후 다시 while문 조건을 비교하는 식으로  servo2.write(90 + (y != \_y ? (y < 0 ? \_y-- : \_y++) : \_y));  //\_x와 x, \_y와 y사이의 값의 조정이 일어난다  delay(100/((x>\_x?x-\_x:\_x-x)+(y>\_y?y-\_y:\_y-y)+1));  }  servo1.detach();  servo2.detach();  }  private: //클래스의 private변수 \_x, \_y  int \_x = 0;  int \_y = 0;  };  #define OLED\_MOSI 9  #define OLED\_CLK 10  #define OLED\_DC 11  #define OLED\_CS 12  #define OLED\_RESET 13  Adafruit\_SSD1306 display(OLED\_MOSI, OLED\_CLK, OLED\_DC, OLED\_RESET, OLED\_CS);//SSD 1306 OLED LCD 전원 공급을 위한 함수 실정  swRTC rtc;//시간함수 시작  AlarmBot AlarmBot(&display, 6, 7);  void setup() //Set-up 함수  {  Serial.begin(9600);//전력량 공급  display.begin();//OLED LCD 사용 설정  display.clearDisplay();  display.setTextSize(2);//텍스트 크기 설정  display.setTextColor(WHITE);//LED 색 하얀색으로 설정  pinMode(buzzer, OUTPUT);//버저 출력 인자로 설정  pinMode(button, INPUT\_PULLUP);//스위치: 풀업 저항 설정(Swith on: LOW, Switch off: HIGH)  AlarmBot.init();  delay(1000);  AlarmBot.eyeOpen();//초기설정: 알람봇 display 상태(깨어있음)  delay(1000);  }  void Set\_lowThanTen(int time) {//흘러간 시간 측정함수  display.setTextSize(2);//텍스트 크기 설정-2  if(time < 10) {  display.print("0");  display.display();  display.print(time%10);  display.display();  }  else{  display.print(time);  display.display();}  }  void loop()  {  rtc.stopRTC(); //정지  rtc.setTime(0,0,0); //시간, 분, 초 초기화  rtc.startRTC(); //시간 측정 시작  int analogInput = analogRead(A0);//가변 저항 값을 analogRead 함수를 이용하여 analogInput 변수에 저장  int minute;// minute 변수 선언-가변 저항을 통해 설정한 타이머 시간을 받아준다    if((analogInput/34) > 0){//analogInput값을 34로 나눠 0~30까지 값을 만들수있도록 조정하였다.  display.clearDisplay();  while(1){  analogInput = analogRead(A0);  //가변 저항 값을 analogRead 함수를 이용하여 analogInput 변수에 저장  //-while문을 이용해 가변저항이 변할때 마다 값을 출력할 수 있도록 설계  display.setCursor(58,5);//출력 위치 설정  display.setTextSize(2);//텍스트 크기 설정-2  display.println(analogInput / 34);//analogInput을 34로 나눈 값을 출력  display.setCursor(45,20);//출력 위치 설정  display.setTextSize(1);//텍스트 크기 설정-1  display.print("minutes");//minute 출력  display.display();  if(digitalRead(button)==LOW){//Switch-On이 되면 타이머 작동: while루프 break  display.clearDisplay();  minute = analogInput/34;//minute변수에 설정한 타이머 값 저장  break;  }  display.clearDisplay();  }  AlarmBot.eyeClose();//알람봇 상태: 수면  AlarmBot.See(0, 30);  rtc.setTime(0,0,0);  while(1){  display.setCursor(17,8);//1마이크로초 마다 현재시간을 LCD에 출력(위치 17,8)  Set\_lowThanTen(rtc.getHours());//흘러간 '시간' 출력  display.print(":");  display.display();  Set\_lowThanTen(rtc.getMinutes());//흘러간 '분' 출력  display.print(":");  display.display();  Set\_lowThanTen(rtc.getSeconds());//흘러간 '초' 출력  display.clearDisplay();  if(minute == rtc.getMinutes()){//타이머 음악 출력-'캐논 변주곡'  for (int Note = 0; Note < 64; Note++) {  int noteDuration = 1000 / noteDurations[thisNote];  tone(8, melody[thisNote], noteDuration);  int pauseBetweenNotes = noteDuration \* 1.30;  delay(pauseBetweenNotes);  noTone(8);  }  delay(1000);  break;  }  }  AlarmBot.init();  AlarmBot.See(60, -50);  AlarmBot.Blink();  delay(1000);  AlarmBot.See(-60, -50);  delay(500);  //깬 알람봇 좌우로 움직인다.  AlarmBot.See(0, 0);  }  //깨어있는동안은 눈을 깜빡인다.  AlarmBot.Blink();  delay(4000);  } |
| Pitches.h 소스코드 |
| #define NOTE\_B0 31  #define NOTE\_C1 33  #define NOTE\_CS1 35  #define NOTE\_D1 37  #define NOTE\_DS1 39  #define NOTE\_E1 41  #define NOTE\_F1 44  #define NOTE\_FS1 46  #define NOTE\_G1 49  #define NOTE\_GS1 52  #define NOTE\_A1 55  #define NOTE\_AS1 58  #define NOTE\_B1 62  #define NOTE\_C2 65  #define NOTE\_CS2 69  #define NOTE\_D2 73  #define NOTE\_DS2 78  #define NOTE\_E2 82  #define NOTE\_F2 87  #define NOTE\_FS2 93  #define NOTE\_G2 98  #define NOTE\_GS2 104  #define NOTE\_A2 110  #define NOTE\_AS2 117  #define NOTE\_B2 123  #define NOTE\_C3 131  #define NOTE\_CS3 139  #define NOTE\_D3 147  #define NOTE\_DS3 156  #define NOTE\_E3 165  #define NOTE\_F3 175  #define NOTE\_FS3 185  #define NOTE\_G3 196  #define NOTE\_GS3 208  #define NOTE\_A3 220  #define NOTE\_AS3 233  #define NOTE\_B3 247  #define NOTE\_C4 262  #define NOTE\_CS4 277  #define NOTE\_D4 294  #define NOTE\_DS4 311  #define NOTE\_E4 330  #define NOTE\_F4 349  #define NOTE\_FS4 370  #define NOTE\_G4 392  #define NOTE\_GS4 415  #define NOTE\_A4 440  #define NOTE\_AS4 466  #define NOTE\_B4 494  #define NOTE\_C5 523  #define NOTE\_CS5 554  #define NOTE\_D5 587  #define NOTE\_DS5 622  #define NOTE\_E5 659  #define NOTE\_F5 698  #define NOTE\_FS5 740  #define NOTE\_G5 784  #define NOTE\_GS5 831  #define NOTE\_A5 880  #define NOTE\_AS5 932  #define NOTE\_B5 988  #define NOTE\_C6 1047  #define NOTE\_CS6 1109  #define NOTE\_D6 1175  #define NOTE\_DS6 1245  #define NOTE\_E6 1319  #define NOTE\_F6 1397  #define NOTE\_FS6 1480  #define NOTE\_G6 1568  #define NOTE\_GS6 1661  #define NOTE\_A6 1760  #define NOTE\_AS6 1865  #define NOTE\_B6 1976  #define NOTE\_C7 2093  #define NOTE\_CS7 2217  #define NOTE\_D7 2349  #define NOTE\_DS7 2489  #define NOTE\_E7 2637  #define NOTE\_F7 2794  #define NOTE\_FS7 2960  #define NOTE\_G7 3136  #define NOTE\_GS7 3322  #define NOTE\_A7 3520  #define NOTE\_AS7 3729  #define NOTE\_B7 3951  #define NOTE\_C8 4186  #define NOTE\_CS8 4435  #define NOTE\_D8 4699  #define NOTE\_DS8 4978 |

1. 결론 : 조원들 소감

김주원: 과제를 하면서 마이크로폰 앰프를 납땜하는 등 소리인식 기능을 추가하려고 하였지만 추가해보지 못해 아쉬움이 남았다. 0~30분말고 1024로 나누어 0~1024분까지 표현할 수 있었음에 아쉬움이 남는다. 추가로 연구했던 자료를 완벽히 이해하지 못한 부분이 있어 아쉬움이 남는다. 그러나 하나의 완성된 작품을 조원들끼리 만들어 보람을 느꼈고 수업때 들었던 내용을 활용할 수 있었던 의미있는 코딩이었던 것 같다. 아두이노에대해 더 알게 되어 의미있는 과제였던 것 같다.

조규민: 이번 과제를 수업시간동안 배웠던 것들과 인터넷을 활용하여 제작하였는데 생각보다 많이 어렵고 이렇게 작품을 만들어 낸다는게 쉬운일이 아니라는것을 깨달았다.

김성민: 그동안 로봇이라고 하면 나와 거리가 멀다고 생각했는데, 이번 아두이노 과제를 통해서 물론 단순한 기능을 가진 로봇이지만 로봇이라는 대상과 한걸음 더 친근하게 느껴질 수 있었던 경험이 되었다. 코딩을 하는 과정에서 살짝 어려움을 느꼈지만 팀원들이 많이 도와줌으로써 기능을 설정할 수 있었던 것 같다!

박기웅: 처음에 생각했던거보다 코드를 구현하는 과정이 어려웠고 원하는 만큼의 모니터를 구현하기가 힘들었다 팀원 모두가 노력하고 여러번의 수정을 거쳐 완성하였는데 다들 몇일 밤새서 고생하였다 다음에 기회가 되면 좀 더 보완된 타이머를 만들어 보고싶다

김태준: 이번과제를하면서 모니터를 만드는 기업에서 클래스를 따로 제공하기도 한다는 것이 놀라웠다. 또 여러가지 키트들을 이용하여 직접코딩하여 쓸모있는 물건을 만들어낼 수 있다는 사실을 깨달았다



<마이크로폰 앰프를 납땜하는 모습>

1. 추가 참고한 자료 사이트 및 참고한 자료

\*참고한 사이트:

<http://blog.naver.com/roboholic84>

<http://deneb21.tistory.com/429->예제연습을 위해 참고한 사이트

<http://hellcross.tistory.com/entry/ARDUINO-%EC%95%8C%EB%9E%8C%EC%8B%9C%EA%B3%84-%EC%86%8C%EC%8A%A4%EC%BD%94%EB%94%A9->타이머를 만들기 위해 참고한 사이트

\*추가자료: 코드 3-1의 CanonMelody 함수와 noteDuration함수는

캐논 변주곡 악보를 이용하여 음계를 배치하여 직접 멜로디를 만들었습니다.

화음으로 되어있는 부분은 화음 부분의 높은 음들을 이용하여 코드를 구성하였습니다.

