|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **결 과 보 고 서** | | | | |
| 학 과 | 학 년 | 학 번 | 조 | 성 명 |
| 전자공학과 | 3 | 12191505 | - | 윤수연 |
| 실험 제목 | MINDLESS FARM | | | |
| 1. **실험 과정**    1. **회로구성**     그림 1. 회로 구성도  위의 회로 구성도는 예비 보고서의 계획과 다소 다르다. 먼저 각 센서로 들어오는 신호를 확인하기 위해 오실로스코프를 통해 파형을 관찰한다. 그리고 센서들에 공통적으로 Vcc와 GND가 연결되어야 하기 때문에 이를 모두 이어줬다.   * + 1. **button**   플레이어가 조작하는 부분으로, 두 개의 button이 해당된다. 여러 소자들의 입력 값을 상황에 따라 관찰하기 위해 상황을 나누는 용도로 사용했다. 지금까지 입력을 위해 가장 많이 사용된 button소자를 사용했다.    그림 2 buttons   * + 1. **LCD**   플레이어가 보는 화면으로, 2line을 사용하는 LCD를 사용했다. 우선 두 개의 라인만 사용하기 때문에 담을 수 있는 정보량이 매우 작다. 센서를 통해 읽어 들이는 정보는 센서 당 하나씩 3개나 되는데 한 line에 한 시간대에 표시할 수 있는 글자 수도 너무나 제한적이다.  이에 스크린을 여러 부분으로 나누어 button으로 나눈 상황에 맞게 출력이 이루어지도록 했다.    그림 3. LCD   * + 1. **오실로스코프**     그림 4. 오실로스코프 결과  오실로스코프를 센서들의 출력단에 각각 네 신호를 연결했다. 모든 센서들은 실제로는 아날로그 읽어 들이기 때문에 A/D 컨버터가 필요하지만 지금은 하드웨어 가상환경 시뮬레이터인 proteus 위에서 실험을 진행하고 있기 때문에 이미 양자화된 신호가 들어올 것이다.   * + 1. **sensors**     그림 5. 센서들 실행 결과  각 센서들을 모두 연결한 모습이다. 구글링을 통해 필요한 소자들을 간단하게 저항과 커패시터 두 개를 연결했다. 그런데 실행 결과를 보면 아래 위 버튼이 있다. 하드웨어 가상환경 시뮬레이터인 proteus 상에서 실험을 하는 것이기 때문에 실제 읽을 값이 없어서 사용자가 설정해주어야 하는 모습이다. 각 센서마다 달려있는 아래 위 화살표가 그려진 빨간 버튼을 누르면 값을 변경할 수 있다.   * 1. **동작 확인**  1. **오실로스코프**   처음 실행 버튼을 누르면 가장 먼저 오실로크코프 화면이 뜬다. 하지만 센서에 실제 측정할 입력이 없기 때문에 유의미한 결과는 찾을 수 없다.    그림 6. 오실로스코프 첫 실행 화면   1. **초기 화면**   첫 줄에는 MINDLESS FARM이라고 이번 프로젝트 이름을 제시하고 있다. 오실로스코프를 닫은 초기 화면이며 아무 버튼이나 누르면 다음으로 넘어간다.    그림 7. 초기 화면   1. **관찰 센서 선택 화면**   초기화면에서 아무 버튼이나 누르면 나타나는 관찰 센서 선택화면 모습이다. 총 growth temp hum 선택지가 있으며 growth는 적외선 거리측정 센서로, temp는 온습도 센서 중 온도 측정 방법으로, hum는 온습도 센서 중 습도 측정 방법으로 값을 읽을 수 있다. 이후 선택으로 센서를 관찰하고 나서 아무 버튼이나 누르면 다시 이 화면으로 돌아온다.    그림 8. 관찰 센서 선택 화면   1. **센서 선택**   왼쪽 첫번째 버튼을 눌러 각 센서에 대해 관찰할 대상을 선택할 수 있다. 여러 번 눌러 선택지를 바꿀 수 있는데 > 표시를 통해 바꿀 수 있었다.    그림 9, 10. Temp(좌)와 Hum(우) 선택 사진   1. **Growth 선택**   Growth를 선택한 경우 500ms동안 선택을 확인하는 창을 띄운 후 측정된 값을 cm단위로 출력한다.    그림 11, 12. growth 선택 사진(좌: 실행 즉시, 우: 500ms 후)  .   1. **Temp 선택**   temp를 선택한 경우 500ms동안 선택을 확인하는 창을 띄운 후 측정된 값을 섭씨단위로 출력한다.    그림 13, 14. Temp 선택 사진(좌: 실행 즉시, 우: 500ms 후)   1. **Hum 선택**   Hum를 선택한 경우 500ms동안 선택을 확인하는 창을 띄운 후 측정된 값을 %단위로 출력한다.    그림 15, 16. Hum 선택 사진(좌: 실행 즉시, 우: 500ms 후)   1. **출력 소자 추가**   실제 동작에서는 입력을 받아 LCD화면에 출력하는 것보다 출력소자를 통해 구동하는 것이 중요할 것이다. 따라서 motor와 buzzer를 달았다. 만약 growth를 출력하는 적외선 거리 측정 센서의 값이 짧아지면, 즉 식물이 높게 생장하면 수확할 때가 되었다는 뜻이므로 수확을 하라는 의미로 buzzer를 울리고, 온도센서를 통해 온도가 너무 올라갔다면 motor를 이용한 팬을 가동하여 온도를 낮춘다. 또한 습도가 너무 낮아지면 워터펌프를 이용해 물을 끌어 올릴 수 있는데 이 또한 motor를 이용한다. 아래 회로는 이를 반영하여 출력소자를 추가한 회로도다.    그림 17. 출력소자를 추가한 회로도   1. **실험 결과**    1. 코드  |  | | --- | | #define F\_CPU 16000000  #include <avr/io.h>  #include <avr/interrupt.h>  #include <util/delay.h>  #include <stdio.h>  //INTRODUCTION FOR PIN  //A, B = LCD control, data  //D = BUTTON  //E = SENSROS  // Normal Mode  volatile unsigned int sec, min, hour=0;  volatile unsigned int count=0;  volatile unsigned int m\_case=0;  volatile unsigned int growth, temp, hum =0;  //initialize  volatile char str1[1][16] = {{"MINDLESS FARM"}};  volatile char str2[1][16] = {{"press any button"}};  //fuction declaration about LCD  void cmd(int command);  void data(char str);  void display1(char\*str);  //fuctions about interrupt  ISR(TIMER2\_OVF\_vect) {  }  // switch  ISR(INT0\_vect) {  TIMSK = 0b01000000;  if(m\_case == 0){  m\_case = 1;  display1(">growth ");  display2(" Temp Hum ");  }  else if(m\_case == 1){  m\_case = 2;  display1(" growth ");  display2(" >Temp Hum ");  }  else if(m\_case == 2){  m\_case = 3;  display1(" growth ");  display2(" Temp >Hum ");  }  else if(m\_case > 2){  m\_case = 1;  display1(">growth ");  display2(" Temp Hum ");  }  }  // select  ISR(INT1\_vect) {  TIMSK = 0x00;  if(m\_case == 0){  m\_case = 1;  display1(">growth ");  display2(" Temp Hum ");  }  else if(m\_case == 1){  m\_case = 4;  display1("you chose growth ");  display2(" ");  \_delay\_ms(50);  display1("Growth is ");  display2(" %d ", growth);  }  else if(m\_case == 2){  m\_case = 5;  display1("you chose Temp ");  display2(" ");  \_delay\_ms(50);  display1("Temperature is ");  display2("%d ",temp);  }  else if(m\_case == 3){  m\_case = 6;  display1("you chose Hum ");  display2(" ");  \_delay\_ms(50);  display1("Humidity is ");  display2(" %d ", hum);  }  else if(m\_case > 2){  m\_case = 1;  display1(">growth ");  display2(" Temp Hum ");  }  }  int main()  {  //initial setting for Timer/Counter  TCCR2 = 0b00000101; //prescaler 1024  TCNT2 = 0; //init\_0  //TIMSK = 0b01000000;  //initial setting for LCD  DDRA = 0xff;  DDRB = 0xff;    cmd(0b00111100); // set 8bit 2line 4\*10dot  cmd(0x01); // display clear  cmd(0x80); // set DDRAM address or cursor position on display  cmd(0x0c); // display on cursor off  cmd(0x06); // entry mode    //setting for button interrupt  DDRD = 0x00; // input  PORTD = 0b00000111; // pull up on  EICRA = 0b00101010;  EIMSK = 0b00000111;  SREG = 0x80; //allow global interrupt  display1(str1[0]);  int porte\_ = 0;    while(1) {  int porte\_ = PORTE;  growth = 1;  temp = 1;  hum = 1;  }  }  void cmd(int command) {  PORTB = command;  PORTA = 0x04;  \_delay\_ms(1);  PORTA = 0x00;  }  void data(char str) {  PORTB = str;  PORTA = 0x05;  \_delay\_ms(1);  PORTA = 0x01;  }  // LCD 1st line  void display1(char \*str) {  cmd(0x80);  int i = 0;  while(str[i] != '\0') {  data(str[i++]);  }  }  // LCD 2nd line  void display2(char \*str) {  cmd(0b11000000);  int i = 0;  while(str[i] != '\0') {  data(str[i++]);  }  } |  * 1. 아두이노 코드   하지만 proteus의 실험에서는 아쉬운 점이 센서들이 실물과 너무 다르다는 점이다. 이에, 보다 현실감 있는 실험을 위해 여러 방법을 찾던 중, ATmega128보다 간단하고 학습용으로 널리 쓰이는 아두이노를 이용하여 다시 실험을 진행해 보았다. 이 때 예비보고서 작성을 위해 사용한 tinkercad는 회로 연결을 할 수 없어서 사용하지 않았다.  이번 실험에서 사용한 출력 소자 buzzer, lcd, motor 외에도 워터펌프, fan, 식물 생장 led 및 실제 물 보관함 등이 추가되었다고 가정하고 작성한 코드다.  https://youtu.be/bVc3W8sSx1U   |  | | --- | | #include <Wire.h>  #include <hd44780.h>  #include <hd44780ioClass/hd44780\_I2Cexp.h>  #include <DHT.h>  #define DHTPIN A1  #define DHTTYPW DHT11  //functions  void SoilWater(int SoilWaterValue);  void DistanceSensor(int distance);  //init button  const int buttonPin = 2; // the number of the pushbutton pin  int buttonState = 0; // variable for reading the pushbutton status  //init lcd  hd44780\_I2Cexp lcd;  int btn = 8;  int page\_num = 0;  //init led  const int ledPin = 13; // the number of the LED pin  //init dht  DHT dht(DHTPIN, DHTTYPE);  //init SoilWaterSensor  int soilPin = A1;  //init\_pin  int AA = 10;  int AB = 6;  //init data  int distance = 0;  int d\_volt = 0; //temporary value to check distance  int SoilWaterValue = 0;  const char\* a[3][20] = { "1. Water", "2. Distance", "3. Temperature" };  //Humidity is replaced by soil water.  void setup() {  //lcd  lcd.begin(16,2);  pinMode(btn, INPUT);  pinMode(buttonPin, INPUT);  pinMode(ledPin, OUTPUT);  pinMode(9,OUTPUT);  pinMode(8,OUTPUT);  //lcd.noBacklight  lcd.init();  lcd.backlight();  //waterpump  pinMode(AA,OUTPUT);  pinMode(AB,OUTPUT);  }  void loop() {  buttonState = digitalRead(buttonPin);    //lcd init  lcd.clear();  lcd.setCursor(0,0);  lcd.print("MINDLESS FARM");  lcd.setCursor(0,1);  lcd.print("press to start");    //INTERRUPT  //check soil water  SoilWaterValue = analogRead(soilPin);  delay(100);    //abouts soil water  if(SoilWaterValue > 600){  lcd.clear();  lcd.setCursor(0,0);  lcd.print("Lack of Water");  lcd.setCursor(0,1);  lcd.print("Droping... please wait");  //waterpump  digitalWrite(AA, HIGH);  digitalWrite(AB,HIGH );  delay(5000);    digitalWrite(AA, LOW);  digitalWrite(AB, HIGH);  delay(5000);  }  //check temp  int t = dht.readTemperature();  //about temp    if (t>10){    lcd.clear();  lcd.setCursor(0,0);  lcd.print("Too hot");  lcd.setCursor(0,1);  lcd.print("FAN... please wait");  digitalWrite(9,HIGH);  digitalWrite(8,LOW);//high -> CCW  }  delay(500);  //check distance  d\_volt = map(analogRead(A0), 0, 1023, 0, 5000);  distance = (27.61 / (d\_volt - 0.1696)) \* 1000;  //about distance  if(distance >10){  lcd.clear();  lcd.setCursor(0,0);  lcd.print("Plants are all grown.");  lcd.setCursor(0,1);  lcd.print("Harvest please");  while(buttonState == LOW){  digitalWrite(ledPin, HIGH);  delay(500);  digitalWrite(ledPin, HIGH);  delay(500);  }  }  ///MAIN ALGORITHM  if (buttonState == HIGH && page\_num==0) {  SoilWater(SoilWaterValue);  } else if(buttonState == HIGH && page\_num==1){  DistanceSensor(distance);  }  else {  //Temp please  }  }  void DistanceSensor(int distance){  int volt = map(analogRead(A0), 0, 1023, 0, 5000);  distance = (27.61 / (volt - 0.1696)) \* 1000;  Serial.print(distance);  Serial.print(" cm");  Serial.println("");  delay(1000);  }  void SoilWater(int SoilWaterValue){  lcd.clear();  lcd.setCursor(0,0);  lcd.print("Lack of Water");  lcd.setCursor(0,1);  lcd.print("Drop the Water" );  delay(1000);  //about soil water  if(SoilWaterValue > 600){  lcd.clear();  lcd.setCursor(0,0);  lcd.print("Lack of Water");  lcd.setCursor(0,1);  lcd.print("Drop the Water");  //waterpump  digitalWrite(AA, HIGH);  digitalWrite(AB,HIGH );  delay(5000);    digitalWrite(AA, LOW);  digitalWrite(AB, HIGH);  delay(5000);    }  else{  lcd.clear();  lcd.setCursor(0,0);  lcd.print("Enough Water");  lcd.setCursor(0,1);  lcd.print("good");  //We dont need to know value.  //that's not intuitive value.  }  } |  1. **고찰**   이번 주차는 전자공학실험2의 마지막 주차로 이제까지 배운 내용을 바탕으로 자율 프로젝트를 진행했다. 한 학기 동안 배운 내용이 꽤 많아서 어떤 프로젝트를 진행할지 고민이 많았다. 배운 내용을 점검하며 자연스럽게 한 학기를 돌아보게 되었는데, 일주일에 한 주제만 간단하게 배운 실험 과목이 16주가 쌓이니 굉장히 많은 것을 배울 수 있었다. 특히 첫 주차에는 proteus 설치만 해도 어려웠고 MCU가 무엇인지 이해가 가지 않아서 전자공학실험2의 1주차 이론 영상을 몇 번씩 돌려보며 필기를 반복해서 보았던 기억이 난다.  이번 마지막 주차는 꽤나 성공적이기도 했고, 다소 흔할 수 있지만 그만큼 직접 코드를 짜볼 수 있는 경험이 유익했다. 특히 아두이노로도 구현해 보니 두 MCU 차이점, 장단점이 잘 드러나 더욱 유익했다. 아무 생각 없이 보드와 소자가 있길래 진행해 봤는데 좋은 결과를 얻게 되어 기쁘다.  코드 작성에서 크게 어려운 것은 없었지만 아두이노 코드는 거의 처음 사용하는 것이라 조금 오래 걸렸다. 하지만 기본적으로 c를 바탕으로 하는 ATmega128을 사용하다가 친절한 Arduino IDE에서 python기반으로 코딩을 하니 쉽다고 느꼈다.  알고리즘 적으로는 display하나에 여러 정보를 넣기 위해 버튼들을 사용했는데 생각보다 예비보고서에서 계획했던 대로 하니까 어렵지 않았다. 다만 lcd의 표시부는 덮어씌워지는 형식이기 때문에 문자열에 각각 스페이스바를 넣어 해결했다.  이것 외에도 0.5초가 실제 0.5초보다 느렸다. 아무래도 코드가 무거워서 더 그랬나보다. lcd는 16x2화면이 LCD 주소값 순서대로 update가 되기 때문에 주르륵 나타나는 것이 아날로그 감성이라 굉장히 마음에 들었다. 실제로 LCD구현을 했다면 더 귀여웠을 것 같아 아쉬웠다.  활용방안도 생각해봤는데, 모든 식물이 높게 자라지 않는 점, 온도와 습도로 간단히 조절할 수 있다는 점에서 콩나물이나 상추 등을 심어보면 좋을 것 같다. | | | | |
|  | | | | |