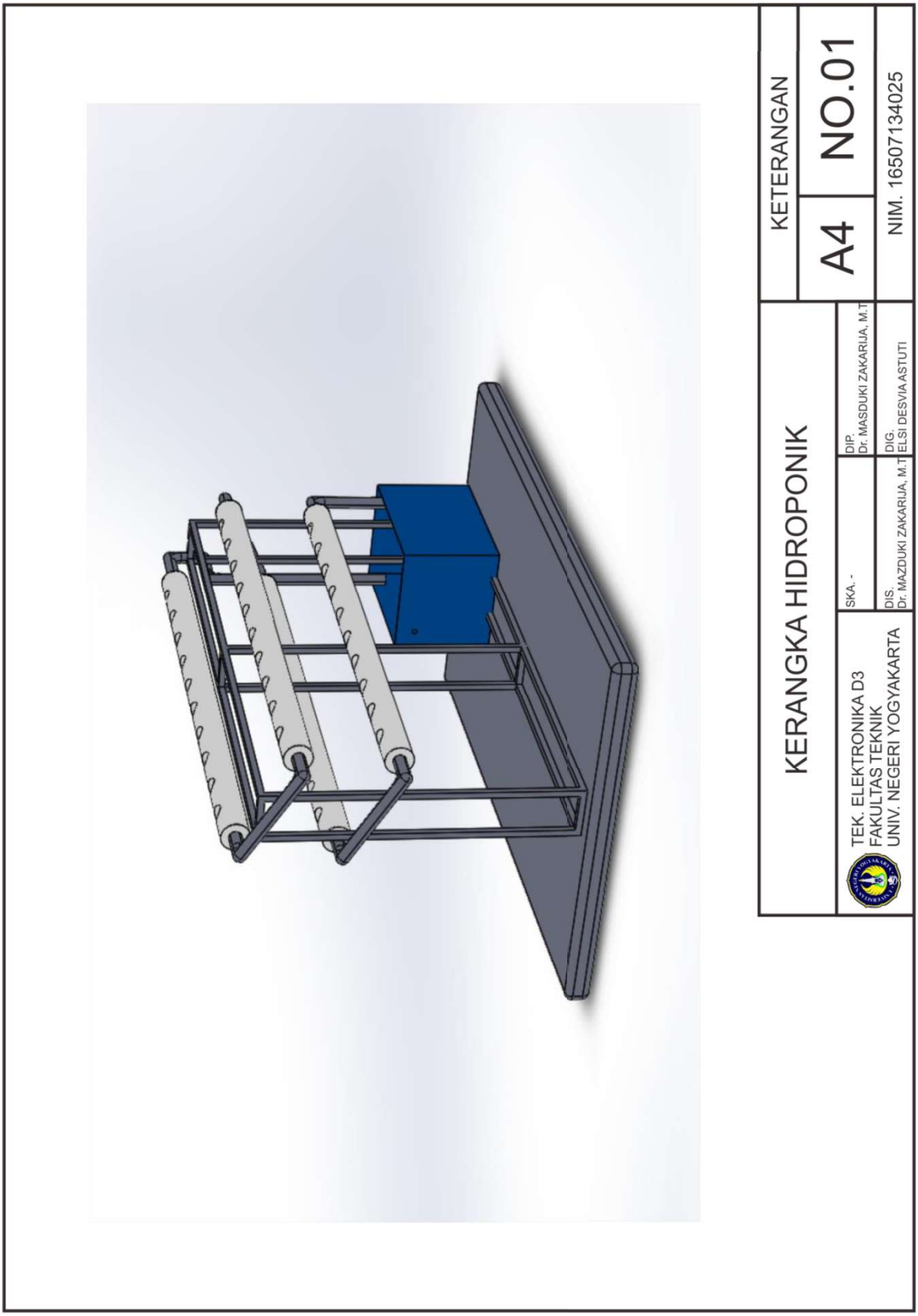
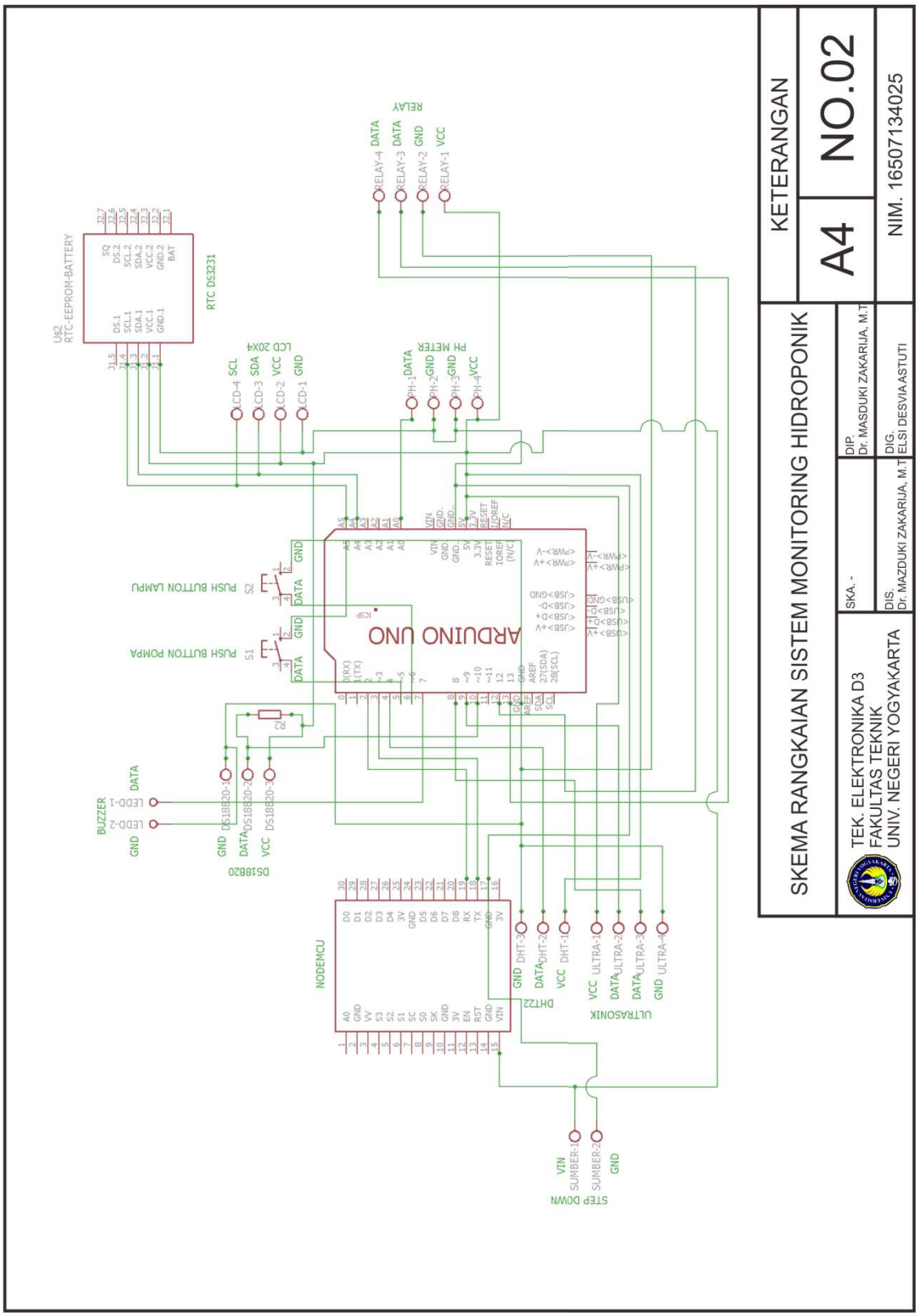
LAMPIRAN

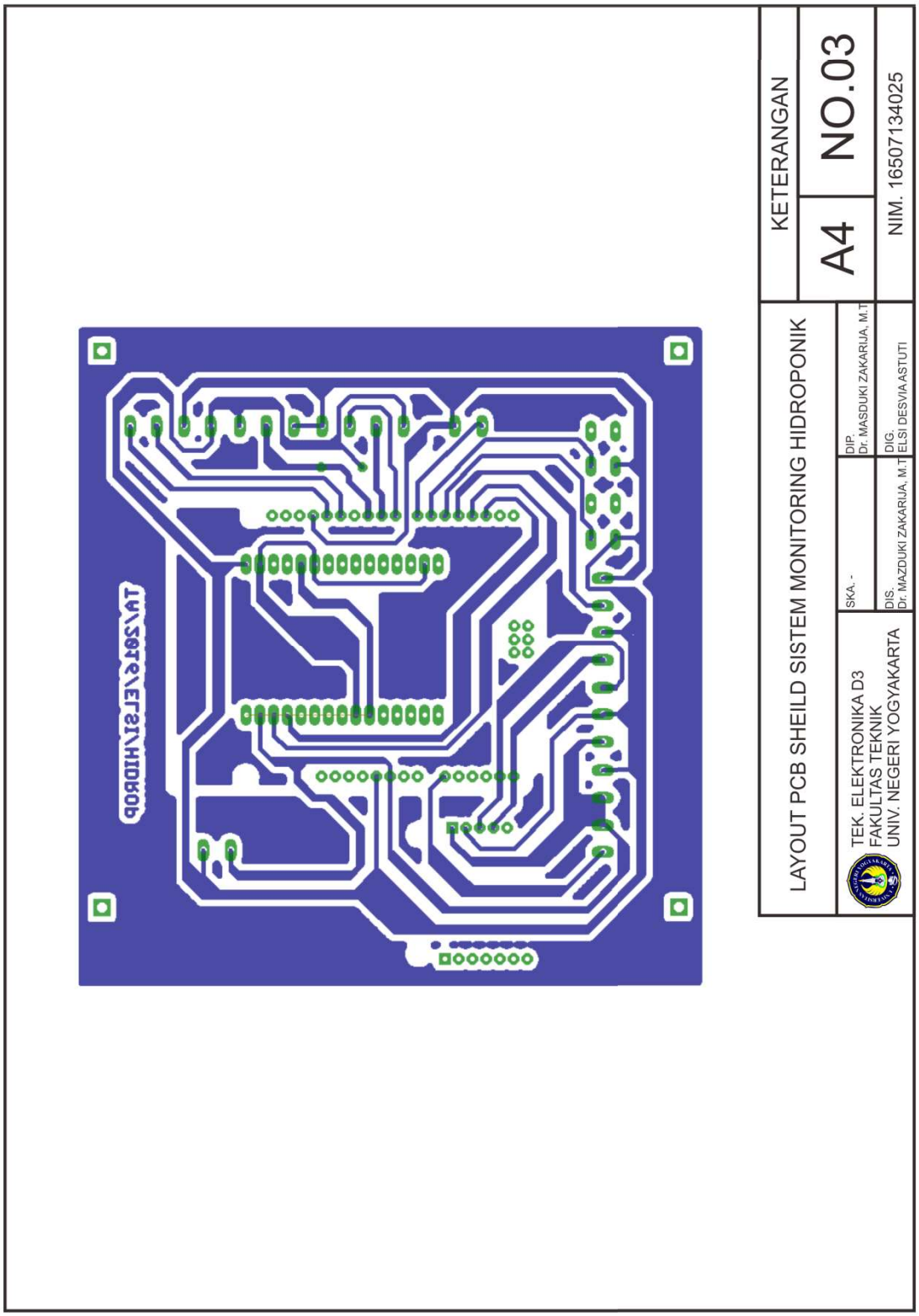
Lampiran 1. Kerangka Hidroponik Sistem *Monitoring* Hidroponik



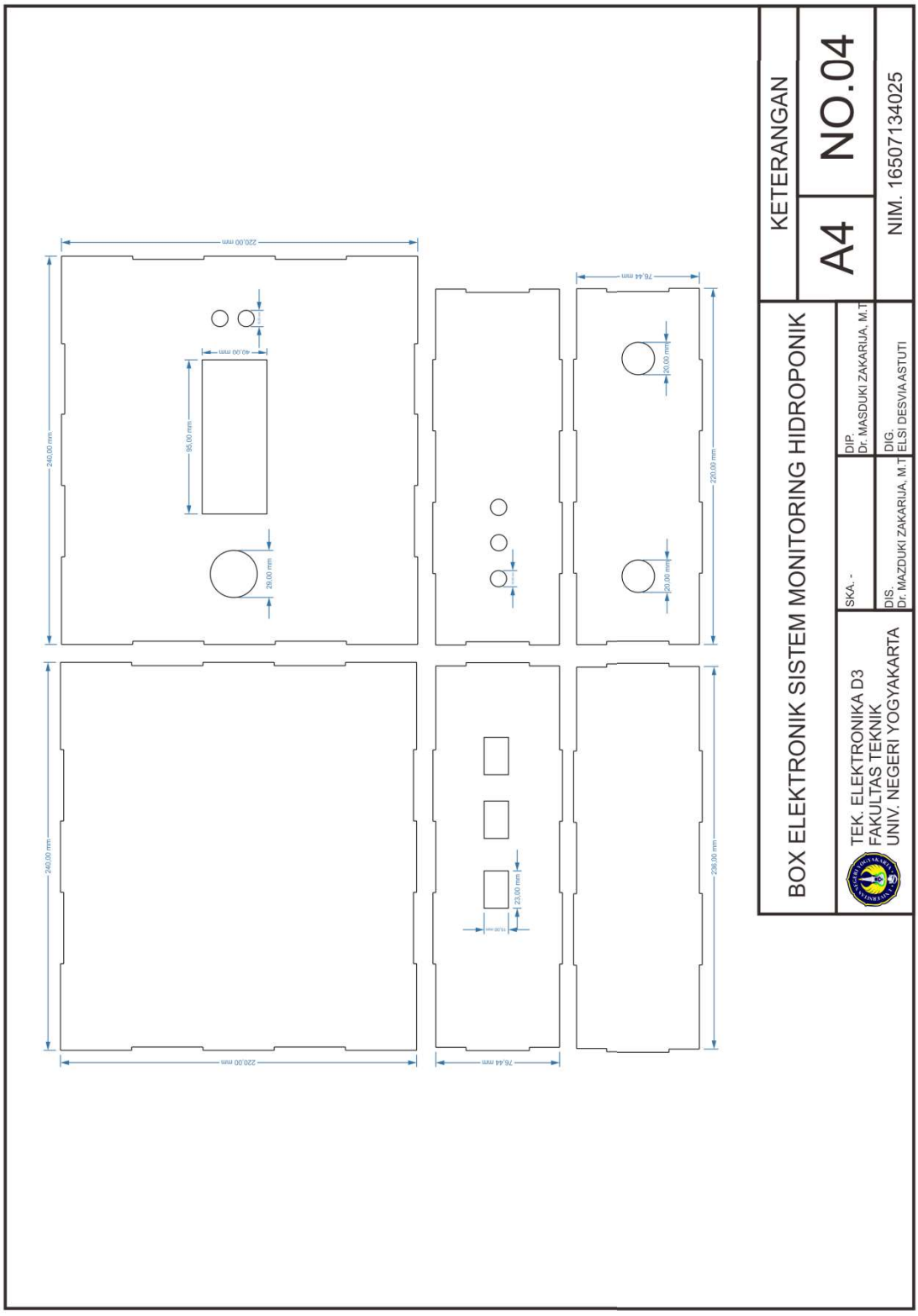
Lampiran 2. Skema Rangkaian Ssitem *Monitoring* Hidroponik



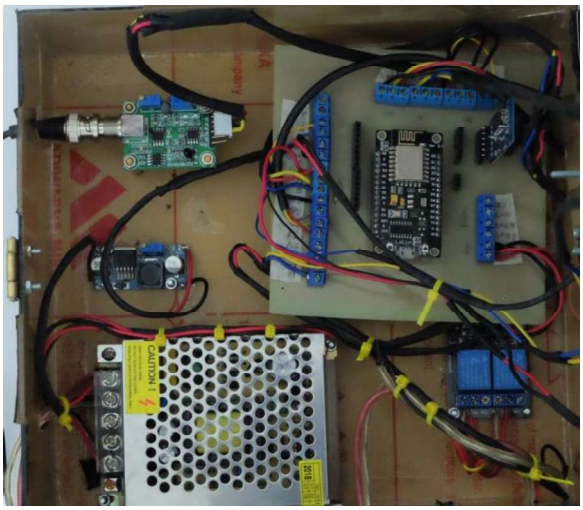
Lampiran 3. *Layout* PCB *Seild* Sistem *Monitoring* Hidroponik

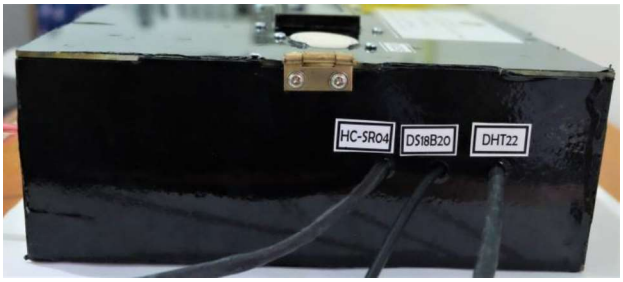


Lampiran 4. *Box* Elektronik Sistem *Monitoring* Hidroponik

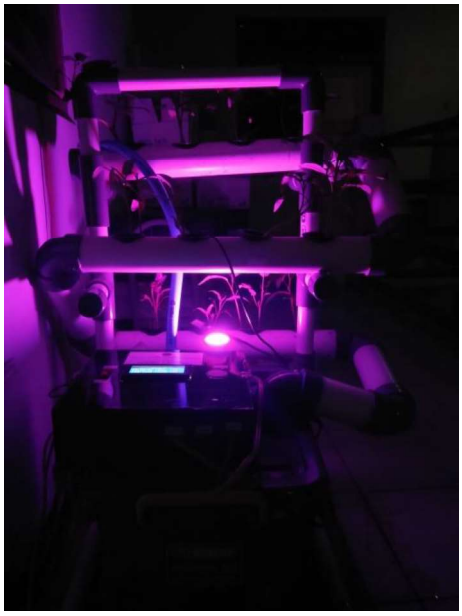


Lampiran 5. Foto *Box* Sistem *Monitoring* Hidroponik





Lampiran 6. Foto Kerangka Sistem *Monitoring* Hidroponik



Lampiran 7. Tampilan Aplikasi *Blynk* Sistem *Monitoring* Hidroponik

Lampiran 8. Daftar Komponen Sistem *Monitoring* Hidroponik

No	Daftar Komponen	Jumlah
1	LCD 20X4	1
2	Buzzer	1
3	*Push Button*	*2*
4	Sensor Ultrasonik HC-SR04	1
5	Sensor Suhu dan Kelembaban DHT22	1
6	Sensor Suhu *Waterproof* DS18B20	1
7	Sensor Ph meter Air	1
8	Arduino UNO	1
9	NodeMCU ESP8266	1
10	Modul Relay 2 *Channel*	1
11	Modul *Step Down*	1
12	Modul RTC DS3231	1
13	Pompa DC 12V	2
14	Lampu *Grow Light*	1
15	*Power Supply*	1

Lampiran 9. Listing Program Arduino

```
/*****************************************************
        Program  : PROGRAM SEMUA NON LCD
        Chip     : ARDUINO
        Author   : ELSI DESVIA ASTUTI
        TUGAS AKHIR HIDROPONIK BERBASIS IOT
*****************************************************/

/*==========Pemanggilan LIBRARY==========*/
#include <DHT.h>                //library dht22
#define DHTPIN 4                //Pin yang dijadikan pembacaan sensor DHT yaitu
pada 4
#define DHTTYPE DHT22           // sensor yang dipakai DHT22
#include <DallasTemperature.h>  //library ds18b20
#include <OneWire.h>
#include <SoftwareSerial.h>
#include "RTClib.h"             //library RTC
#include <LiquidCrystal_I2C.h>
#include <NewPing.h>

/*==========Define Serial==========*/
SoftwareSerial nodemcu(2,3); //rx tx

/*==========Define Pin==========*/
const int ONE_WIRE_BUS = 10;          //pin Dallas 8 ke Arduino
const int PIN_RELAY1 = 12;            // RELAY1 di pin 12 LAMPU
const int PIN_RELAY2 = 13;            // RELAY2 di pin 13 POMPA
const int buttonPin1 = 6; // pin yang digunakan yaitu pin 2
const int buttonPin2 = 5; // pin yang digunakan yaitu pin 3
const int pinBuzzer = 7;                //pin buzzer di pin 7 arduino
#define trigPin 8                     //pin Trig Ultrasonik di pin 8 arduino
#define echoPin 9                     //pin echo Ultrasonik di pin 9 arduino
#define PIN_PH A0                     //pin ph ke arduino A0

/*==========Variabel Global==========*/
String cdata;                       //lengkapi data, yang terdiri dari nilai sensor
String Value = "";                          //ph
int nilaiPin = 0;
int angka_jam = 0;                  //hasil konversi jam ke angka jam:menit
int pinValue1;
int pinValue2;
int buf[10], tempP;                         //ph
int f;                              //ph
```

```
float phValue;
float tegangan;
float t;                          // deklarasi temperatur dht
float h;                          // deklarasi humidity dht
float temp;                       // deklarasi temperatur ds18b20
float b;                          //ph
float val;                        //ph
double DataUs;
const int JamHIDUP = 1600;        // jam sore akan nyala jam 16:00
const int JamMATI = 0500;         // jam pagi akan mati jam 5:00
int bacaButton1; //variabel pembaca kondisi button1
int bacaButton2; //variabel pembaca kondisi button2
int lockB1=1;  //variabel bantu untuk pembacaan button1 sekali tekan sekali baca
int lockB2=1;  //variabel bantu untuk pembacaan button2 sekali tekan sekali baca
int lockPH=0;  //variabel bantu untuk lock PH, saat PH dibawah 5 motor hidup,
namun button dapat menghentikan motor, dan fungsi ini dapat digunakan kembali
jika PH sudah mengalami kenaikan diatas 5
int KonRelay1=0; //variabel untuk menyimpan kondisi relay1
int KonRelay2=0; //variabel untuk menyimpan kondisi relay2
char buff2[10];                   //ph

/*==========Define rule relay==========*/
#define RELAY_ON HIGH             //logika Hidup untuk relay
#define RELAY_OFF !RELAY_ON       //logika Mati untuk relay
#define MAX_DISTANCE 200

// ===== Konfigurasi LCD ===========
// LCD20x4 dengan Modul I2C
// Alamat I2C di 0x27
// lcd_Addr, EN, RW, RS, D4, D5, D6, D7, PIN_BACKLIGHT, Pol
LiquidCrystal_I2C lcd(0x27,20,4);

/*==========Set Sensor==========*/
DHT dht(DHTPIN, DHTTYPE);         //dht22
OneWire oneWire(ONE_WIRE_BUS);    //onewire
DallasTemperature sensors(&oneWire);  //dallas
RTC_DS3231 rtc;                   //rtcds1307
NewPing sonar(trigPin, echoPin, MAX_DISTANCE);
int numberOfDevices;              // Number of temperature devices found
DeviceAddress tempDeviceAddress;  // We'll use this variable to store
a found device address
DateTime now;

/*==========VAriabel Counter==========*/
unsigned long waktu;              //counter waktu mili second
unsigned long lastLCD=0;          //save last update data
```

```
long LCDtime=250;                       //jeda update data LCD

unsigned long lastKirim=0;                  //save last kirim data nodeMCU
long Kirimtime=1000;                        //jeda kirim data nodeMCU

unsigned long int avgValue;                         //ph
unsigned long int Tegangan;
//ph
/*==========Set RTC==========*/
#if defined(ARDUINO_ARCH_SAMD)
// for Zero, output on USB Serial console, remove line below if using
programming port to program the Zero!
  #define Serial SerialUSB
#endif

int get_angka(DateTime now){
// konversi jam dalam ratusan, menit dalam puluhan dan satuan
return (now.hour() * 100) + now.minute();
}

void setup()
{
/*===========Set INPUT==========*/
   pinMode(PIN_PH, INPUT);
   pinMode(ONE_WIRE_BUS, INPUT);
   pinMode(trigPin, INPUT);
   pinMode(echoPin, INPUT);

/*==========Set OUTPUT==========*/
   pinMode(PIN_RELAY1, OUTPUT);
   pinMode(PIN_RELAY2, OUTPUT);
   pinMode(pinBuzzer,OUTPUT);

/*==========AutoSensor==========*/
//    Wire.begin();
   dht.begin();                         //prosedure memulai pemanggilan library dht
   sensors.begin();

/*==========AutoRTC==========*/
 #ifndef ESP8266
 while (!Serial); // for Leonardo/Micro/Zero
#endif
 lcd.begin();

   if (! rtc.begin()) {
   Serial.println("Couldn't find RTC");
```

```
    while (1);
  }

  if (rtc.lostPower()) {
    Serial.println("RTC lost power, lets set the time!");
    // following line sets the RTC to the date & time this sketch was compiled
    rtc.adjust(DateTime(F(__DATE__), F(__TIME__)));
    // This line sets the RTC with an explicit date & time, for example to set
    // January 21, 2014 at 3am you would call:
    // rtc.adjust(DateTime(2014, 1, 21, 3, 0, 0));
  }
/*==========Set LCD==========*/

    lcd.begin();                        //Prosedur Memulai LCD

    lcd.setCursor(1,0);
    lcd.print("SISTEM HIDROPONIK");
    lcd.setCursor(1,1);
    lcd.print("ELSI DESVIA ASTUTI");
    lcd.setCursor(4,2);
    lcd.print("16507134025");
    lcd.setCursor(8,3);
    lcd.print("D TE");
    delay(1000);
    lcd.clear();    //Perintah membersihkan layar LCD
}

void loop(){
/*==========Baudrate komunikasi Serial dan NodeMCU==========*/
    Serial.begin(9600); //baudrate(kecepatan komunukasi) serial
    nodemcu.begin(9600);  //baudrate (kecepatan komunukasi) nodemcu

/*==========Timer/Waktu==========*/
    waktu=millis();  //counter waktu

    unsigned long currentMillis1    = millis();
    unsigned long currentMillis2    = millis();

    DateTime now = rtc.now();
    angka_jam = get_angka(now);

/*==========Sensor pH==========*/

  for(int i=0;i<10;i++)       //Get 10 sample value from the sensor for smooth the
value
  {
```

```
   buf[i]=analogRead(PIN_PH);
   delay(10);
  }
  for(int i=0;i<9;i++)        //sort the analog from small to large
  {
   for(int j=i+1;j<10;j++)
   {
    if(buf[i]>buf[j])
    {
     temp=buf[i];
     buf[i]=buf[j];
     buf[j]=tempP;
    }
   }
  }
  avgValue=0;
  for(int i=2;i<8;i++)                    //take the average value of 6 center sample
   avgValue+=buf[i];
  float phValue=(float)avgValue*5.0/1024/6; //convert the analog into millivolt
  phValue=3.5*phValue;                      //convert the millivolt into pH value

    Value =  dtostrf(phValue, 4, 2, buff2);  //4 is mininum width, 6 is precision

/*==========Sensor DHT==========*/
h = dht.readHumidity();              //pembacaan kelembaban dht
t = dht.readTemperature();            //pembacaan suhu dht


  /*==========Sensor DALAS==========*/
  sensors.requestTemperatures();
  temp = sensors.getTempCByIndex(0);

  /*==========Sensor ULTRASONIK==========*/
  unsigned int uS = sonar.ping(); // kirim ping dan simpan hasilnya di variabel uS
(satuannya mikrodetik)
  DataUs = uS / US_ROUNDTRIP_CM;


/*==========Pengaturan OUTPUT========*/
  if (DataUs >= 20){digitalWrite(pinBuzzer, HIGH);}  //jika ketinggian lebih dari
20 buzzer nyala
  else if (5 < DataUs < 20){digitalWrite(pinBuzzer, LOW);} //Jika ketinggian
antara 5-20 buzzer mati
  if (DataUs <= 5){digitalWrite(pinBuzzer, HIGH);} //jika ketinggian kurang dari
5 buzzer nyala
```

```
  if (DataUs<=5 && lockPH==1){
    KonRelay2=0;  //kondisi motor hidup
    lockPH=0;}    //membantu untuk lock kondisi, sehingga hanya fungsi hanya
dieksekusi 1 kali
   else if (DataUs>=5.1 && lockPH==0){
     KonRelay2=1;  //kondisi motor mati
     lockPH=1;}    //membantu untuk lock kondisi, sehingga hanya fungsi hanya
dieksekusi 1 kali


   if(JamHIDUP==angka_jam){  //Saat jam menunjukan jamHIDUP maka sistem
akan menghidupkan lampu&pompa dan akan terus hidup sampai ada perintal
lebih lanjut
     KonRelay1=1;
     KonRelay2=1;
     }
   else if(JamMATI==angka_jam ){  //Saat jam menunjukan jamMATI maka
sistem akan mematikan lampu&pompa dan akan terus mati sampai ada perintal
lebih lanjut
     KonRelay1=0;
     KonRelay2=0;
     }

bacaButton1= digitalRead(buttonPin1);  //membaca kondisi button 1
bacaButton2= digitalRead(buttonPin2);  //membaca kondisi button 2

  if (bacaButton1==HIGH){
    for (lockB1;lockB1>0;lockB1-=1){   //berfungsi agar walau button dipencet
lama, eksekusi perintah hanya sekali
      KonRelay1+=1;
      if (KonRelay1==2){KonRelay1=0;}
    }
  }
  else if (bacaButton1==LOW){lockB1=1;}  //mengembalikan lockButton 1
menjadi 1, mengindikasikan button sudah tidak dipencet

  if (bacaButton2==HIGH){
    for (lockB2;lockB2>0;lockB2-=1){
      KonRelay2+=1;
      if (KonRelay2==2){KonRelay2=0;}
    }
  }
  else if (bacaButton2==LOW){lockB2=1;}

/*==========OUTPUT KE RELAY========*/
```

```
  if(KonRelay1==0){digitalWrite(PIN_RELAY1,HIGH);}
  else if(KonRelay1==1){digitalWrite(PIN_RELAY1,LOW);}

  if(KonRelay2==0){digitalWrite(PIN_RELAY2,HIGH);}
  else if(KonRelay2==1){digitalWrite(PIN_RELAY2,LOW);}

/*==========Pemanggilan fungsi Output LCD========*/
OUTPUT_LCDSerial();


/*==========Pengiriman data ke nodeMCU========*/
if (nodemcu.available() > 0 ){
  if(waktu-lastKirim>=Kirimtime){
     lastKirim=waktu;
      cdata = cdata +h +","+t+","
+Value+","+temp+","+DataUs+","+pinValue1+","+pinValue2;
      Serial.println(cdata);
      nodemcu.println(cdata);
      cdata = "";
    }
  }
}

void OUTPUT_LCDSerial(){  //Anak Program untuk keluaran data pada LCD
  if(waktu-lastLCD>=LCDtime){
     lastLCD=waktu;

    DateTime now = rtc.now();
/*==========PEnampilan waktu==========*/
    lcd.setCursor(2,0);
    lcd.print(now.day(), DEC);
    lcd.print('/');
    lcd.print(now.month(), DEC);
    lcd.print('/');
    lcd.print(now.year(), DEC);
    lcd.print(" ");
    lcd.print(now.hour(), DEC);
    lcd.print(':');
    lcd.print(now.minute(), DEC);
    lcd.print(':');
    lcd.print(now.second(), DEC);

/*==========Penampilan PH==========*/
  lcd.setCursor(0,1);
  lcd.print("pH =");
  lcd.setCursor(9,1);
```

```
  lcd.print("|");
  lcd.setCursor(4,1);
  lcd.print(phValue, 2);
  Serial.print("hasil tegangan: ");
  Serial.print(Tegangan);
  Serial.print("PH: ");
  Serial.print(phValue, 2);

/*==========Penampilan Humidity==========*/
  lcd.setCursor(10,1);
  lcd.print("Hum  =");;
  lcd.setCursor(16,1);
  lcd.print((int)h);
  Serial.print("  humidity = ");
  Serial.print(h);
  Serial.print("%  ");

/*==========Penampilan Temperatur==========*/
  lcd.setCursor(10,2);
  lcd.print("TemU =");;
  lcd.setCursor(16,2);
  lcd.print((int)t);
  Serial.print("temperature = ");
  Serial.print(t);

  Serial.print("  TemA = ");
  Serial.println(temp);
  lcd.setCursor(10,3);
  lcd.print("TemA =");;
  lcd.setCursor(16,3);
  lcd.print(temp, 0);

/*==========Penampilan DataUs==========*/
  Serial.print("Ping: "); //kirim tulisan 'Ping' ke serial monitor
  Serial.print(DataUs, 0); // konversi pingtime ke jarak (cm)
  Serial.println("cm");   //kirim tulisan 'cm' ke serial monitor

  lcd.setCursor(0,2);
  lcd.print("Dis=");
  lcd.setCursor(9,2);
  lcd.print("|");
  lcd.setCursor(4,2);
  lcd.print("   ");
  lcd.setCursor(4,2);
  lcd.print(DataUs, 0);
  lcd.setCursor(6,2);
```

```
  lcd.print("cm");

/*==========Penampilan Kondisi Relay==========*/
    lcd.setCursor(0,3);
    lcd.print("L=");
    lcd.setCursor(2,3);
    if (KonRelay1==1){lcd.print("H");}
    if (KonRelay1==0){lcd.print("M");}
    lcd.setCursor(4,3);
    lcd.print("|");
    Serial.print("L: ");
    if (KonRelay1==1){Serial.println("H");}
    if (KonRelay1==0){Serial.println("M");}

    lcd.setCursor(5,3);
    lcd.print("P=");
    lcd.setCursor(7,3);
    if (KonRelay2==1){lcd.print("H");}
    if (KonRelay2==0){lcd.print("M");}
    lcd.setCursor(9,3);
    lcd.print("|");
    Serial.print("P: ");
    if (KonRelay2==1){Serial.println("H");}
    if (KonRelay2==0){Serial.println("M");}

/*==========Penampilan Kondisi Button==========*/
    Serial.print("Lbut: "); //kirim tulisan 'Ping' ke serial monitor
    if (bacaButton1==1){Serial.println("H");}
    if (bacaButton1==0){Serial.println("M");}

    Serial.print("Pbut: "); //kirim tulisan 'Ping' ke serial monitor
    if (bacaButton2==1){Serial.println("H");}
    if (bacaButton2==0){Serial.println("M");}
  }
}
```

Lampiran 10. Listing Program Nodemcu Esp8266

```
/*************************************************
        Program  : PROGRAM SEMUA
        Chip     : NODEMCU V3
        Author   : ELSI DESVIA ASTUTI
        TUGAS AKHIR HIDROPONIK BERBASIS IOT
*************************************************/
/*==========LIBRARY==========*/
#define BLYNK_PRINT Serial

#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>
#include <SoftwareSerial.h>
#include <SimpleTimer.h>   //including the library of SimpleTimer
/*==========Define Bylnk==========*/
char auth[] = "9903416abdc94784ab4e1175cd0bf630";

/*==========Set WiFi==========*/
char ssid[] = "500rb"; //nama wifi
char pass[] = "satusampenol0"; //password wifi

SimpleTimer timer;
int satuVal, duaVal, tigaVal, empatVal, limaVal;      //sensor
int pinPus;
int pinValue1;
int pinValue2;
/*Fungsi ini mengirimkan waktu Arduino setiap detik ke Virtual Pin (1).
 * Dalam aplikasi, frekuensi membaca Widget harus diatur ke PUSH. Ini berarti
 menentukan seberapa sering mengirim data ke Blynk App.*/

void sensorvalue1()
{
 int sdata = satuVal;
  // dapat mengirim nilai apa saja kapan saja.
  // jangan kirim lebih dari 10 nilai per detik.
  Blynk.virtualWrite(V5, sdata); //ph
}
void sensorvalue2()
{
int sdata = duaVal;
  // dapat mengirim nilai apa saja kapan saja.
  // jangan kirim lebih dari 10 nilai per detik.
  Blynk.virtualWrite(V6,sdata); //temU
```

```
}
void sensorvalue3()
{
int sdata = tigaVal;
  // dapat mengirim nilai apa saja kapan saja.
  // jangan kirim lebih dari 10 nilai per detik.
  Blynk.virtualWrite(V2,sdata); //kelembaban
}
void sensorvalue4()
{
 int sdata= empatVal;
  // dapat mengirim nilai apa saja kapan saja.
  // jangan kirim lebih dari 10 nilai per detik.
  Blynk.virtualWrite(V4,sdata); //temA
}
void sensorvalue5()
{
int sdata = limaVal;
  // dapat mengirim nilai apa saja kapan saja.
  // jangan kirim lebih dari 10 nilai per detik.
  Blynk.virtualWrite(V3, sdata); //ultra
}
String myString; // complete message from arduino, which consistors of snesors
data
char sdata, rdata; // received charactors
String getValue(String data, char separator, int index)
{
    int found = 0;
    int strIndex[] = { 0, -1 };
    int maxIndex = data.length() - 1;

    for (int i = 0; i <= maxIndex && found <= index; i++) {
        if (data.charAt(i) == separator || i == maxIndex) {
            found++;
            strIndex[0] = strIndex[1] + 1;
            strIndex[1] = (i == maxIndex) ? i+1 : i;
        }
    }
    return found > index ? data.substring(strIndex[0], strIndex[1]) : "";
}
BLYNK_WRITE(D6)
{
  pinValue1 = param.asInt();
}
BLYNK_WRITE(D5)
{
```

```
  pinValue2 = param.asInt();
}

void myTimerEvent()
{
  // You can send any value at any time.
  // Please don't send more that 10 values per second.
  Blynk.virtualWrite(V1, millis() / 1000);

}


void setup()
{
    Serial.begin(9600);// baud komunikasi serial monitor

    Blynk.begin(auth, ssid, pass);

    timer.setInterval(1000L,sensorvalue1); //ph
    timer.setInterval(1000L,sensorvalue2); //temperatureudara
    timer.setInterval(1000L,sensorvalue3); //humi
    timer.setInterval(1000L,sensorvalue4); //temperatureair
    timer.setInterval(1000L,sensorvalue5); //waterlevel
}


void loop()
{
   if (Serial.available() == 0 )
    {
  Blynk.run();
  timer.run(); // Initiates BlynkTimer
    }

  if (Serial.available() > 0 )
  {
    rdata = Serial.read();
    myString = myString+ rdata;
   // Serial.print(rdata);
    if( rdata == '\n')
    {
      //Serial.println(myString);
      //Serial.println("phmeter")l
      //program baru
      String l = getValue(myString, ',',0);
      String m = getValue(myString, ',',1);
```

```
    String n = getValue(myString, ',',2);
    String o = getValue(myString, ',',3);
    String p = getValue(myString, ',',4);

    satuVal = l.toInt();
    duaVal = m.toInt();
    tigaVal = n.toInt();
    empatVal = o.toInt();
    limaVal = p.toInt();


    myString = "";
    //selesai program baru
   }
  }

}
```

Lampiran 11. Spesifikasi Sistem *Monitoring* Hidroponik

No	Spesifikasi	Sistem *Monitoring* Hidroponik
1.	Ukuran *Box*	Panjang 22cm, Lebar 24cm, Tinggi 8cm.
2.	Ukuran Kerangka Hidroponik	42cm x 79cm
3.	Mikrokontroler	Arduino UNO NodeMCU ESP8266
4.	Input data	Sensor Ultrasonik HC-SR04 Sensor Suhu dan Kelembaban DHT22 Sensor Suhu *Waterproof* DS18B20 Sensor pH meter Air
5.	Output	Pompa Lampu Grow Light Buzzer LCD 20X4
6.	Sensor	Sensor Ultrasonik HC-SR04 Sensor Suhu dan Kelembaban DHT22 Sensor Suhu *Waterproof* DS18B20 Sensor pH meter Air
7.	Pompa	Pompa Air 12 DC sebagai pengaliran air dari penampung ke tanaman
8.	Software Complier	Arduino IDE dan Blynk
9.	Objek yang dideteksi	Ketinggian Air Suhu dan Kelembaban Udara Suhu Air pH air
10.	Tegangan Input	Catu daya 12 VDC Switching 5VDC
11.	USB Port	Micro USB

Lampiran 12. Datasheet Arduino UNO

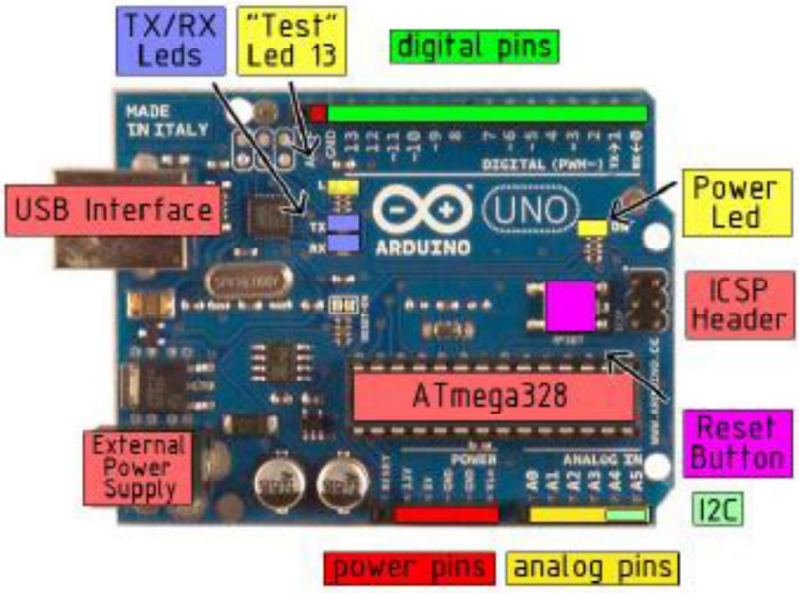
Technical Specification

EAGLE files: arduino-duemilanove-uno-design.zip Schematic: arduino-uno-schematic.pdf

Summary

Microcontroller	ATmega328
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	6
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB of which 0.5 KB used by bootloader
SRAM	2 KB
EEPROM	1 KB
Clock Speed	16 MHz

the board



Lampiran 13. Datasheet NodeMCU

Internet of Things

NodeMCU ESP8266 ESP-12E WiFi Development Board

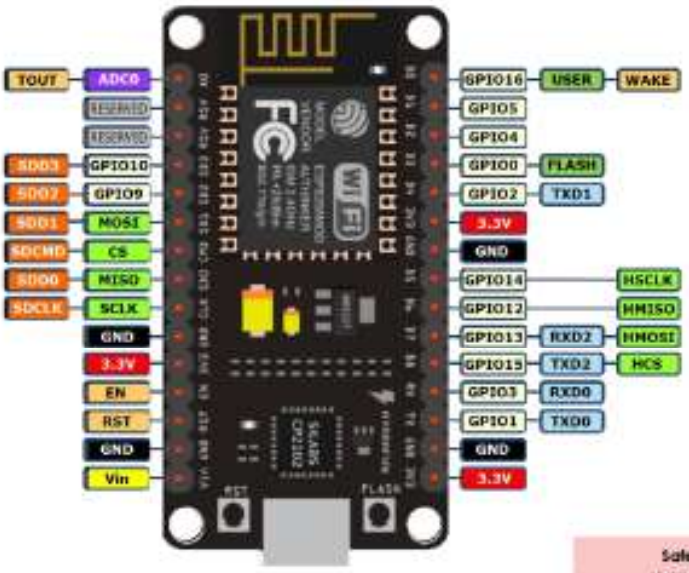
NodeMCU is an open source IoT platform. It includes firmware which runs on the ESP8266 Wi-Fi SoC from Espressif Systems, and hardware which is based on the ESP-12 module. The term 'NodeMCU' by default refers to the firmware rather than the DevKit. The firmware uses the Lua scripting language. It is based on the eLua project, and built on the Espressif Non-OS SDK for ESP8266. It uses many open source projects, such as lua-cjson, and spiffs.

Features

- Version : DevKit v1.0
- Breadboard Friendly
- Light Weight and small size.
- 3.3V operated, can be USB powered.
- Uses wireless protocol 802.11b/g/n.
- Built-in wireless connectivity capabilities.
- Built-in PCB antenna on the ESP-12E chip.
- Capable of PWM, I2C, SPI, UART, 1-wire, 1 analog pin.
- Uses CP2102 USB Serial Communication interface module.
- Arduino IDE compatible (extension board manager required).
- Supports Lua (alike node.js) and Arduino C programming language.

PINOUT DIAGRAM

NodeMCU ESP8266 v1.0



Safety Precaution

All GPIO runs at 3.3V !!

Source

https://iotbytes.wordpress.com/nodemcu-pinout/

NodeMCU ESP8266

Front View

Front View

Specifications of ESP-12E WiFi Module

Wireless Standard	IEEE 802.11 b/g/n
Frequency Range	2.412 - 2.484 GHz
Power Transmission	802.11b : +16 ± 2 dBm (at 11 Mbps) 802.11g : +14 ± 2 dBm (at 54 Mbps) 802.11n : +13 ± 2 dBM (at HT20, MCS7)
Receiving Sensitivity	802.11b : -93 dBm (at 11 Mbps, CCK) 802.11g : -85 dBm (at 54 Mbps, OFDM) 802.11n : -82 dBm (at HT20, MCS7)
Wireless Form	On-board PCB Antenna
IO Capability	UART, I2C, PWM, GPIO, 1 ADC
Electrical Characteristic	3.3 V Operated 15 mA output current per GPIO pin 12 - 200 mA working current Less than 200 uA standby current
Operating Temperature	-40 to +125 °C
Serial Transmission	110 - 921600 bps, TCP Client 5
Wireless Network Type	STA / AP / STA + AP
Security Type	WEP / WPA-PSK / WPA2-PSK
Encryption Type	WEP64 / WEP128 / TKIP / AES
Firmware Upgrade	Local Serial Port, OTA Remote Upgrade
Network Protocol	IPv4, TCP / UDP / FTP / HTTP
User Configuration	AT + Order Set, Web Android / iOS, Smart Link APP

Lampiran 14. Datasheet DHT22

Standard AM2302/DHT22

AM2302/DHT22 with big case and wires

Digital relative humidity & temperature sensor AM2302/DHT22

1. Feature & Application:

*High precision
*Capacitive type
*Full range temperature compensated
*Relative humidity and temperature measurement
*Calibrated digital signal
*Outstanding long-term stability
*Extra components not needed
*Long transmission distance, up to 100 meters
*Low power consumption
*4 pins packaged and fully interchangeable

2. Description:

AM2302 output calibrated digital signal. It applys exclusive digital-signal-collecting-technique and humidity sensing technology, assuring its reliability and stability. Its sensing elements is connected with 8-bit single-chip computer.

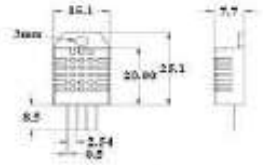
Every sensor of this model is temperature compensated and calibrated in accurate calibration chamber and the calibration-coefficient is saved in type of programme in OTP memory, when the sensor is detecting, it will cite coefficient from memory.

Small size & low consumption & long transmission distance(100m) enable AM2302 to be suited in all kinds of harsh application occasions. Single-row packaged with four pins, making the connection very convenient.

3. Technical Specification:

Model	AM2302	
Power supply	3.3-5.5V DC	
Output signal	digital signal via 1-wire bus	
Sensing element	Polymer humidity capacitor	
Operating range	humidity 0-100%RH;	temperature -40-.80Celsius
Accuracy	**humidity +-2%RH**(Max +-5%RH);	temperature +-0.5Celsius
Resolution or sensitivity	humidity 0.1%RH;	temperature 0.1Celsius
Repeatability	humidity +-1%RH;	temperature +-0.2Celsius
Humidity hysteresis	+-0.3%RH	
Long-term Stability	+-0.5%RH/year	
Interchangeability	fully interchangeable	

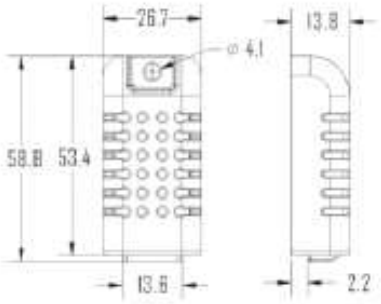
4. Dimensions: (unit—mm)



Pin sequence number: 1 2 3 4 (from left to right direction).

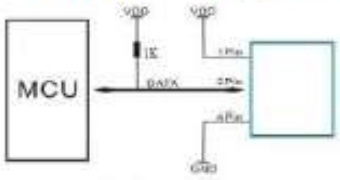
Pin	Function
1	VDD—power supply
2	DATA—signal
3	GND
4	GND

Standard AM2302's dimensions as above



Big case's dimensions as above
Red wire--power supply, Black wire--GND
Yellow wire--Data output

5. Electrical connection diagram:



6. Operating specifications:

(1) Power and Pins

Power's voltage should be 3.3-5.5V DC. When power is supplied to sensor, don't send any instruction to the sensor within one second to pass unstable status. One capacitor valued 100nF can be added between VDD and GND for wave filtering.

(2) Communication and signal

1-wire bus is used for communication between MCU and AM2302. (Our 1-wire bus is specially designed, it's different from Maxim/Dallas 1-wire bus, so it's incompatible with Dallas 1-wire bus.)

Illustration of our 1-wire bus:

7. Electrical Characteristics:

Items	Condition	Min	Typical	Max	Unit
Power supply	DC	3.3	5	6	V
Current supply	Measuring	1		1.5	mA
	Stand-by	40	Null	50	uA
Collecting period	Second		2		Second

8. Attentions of application:

(1) Operating and storage conditions

We don't recommend the applying RH-range beyond the range stated in this specification. The AM2302 sensor can recover after working in abnormal operating condition to calibrated status, but will accelerate sensors' aging.

(2) Attentions to chemical materials

Vapor from chemical materials may interfere AM2302's sensitive-elements and debase AM2302's sensitivity.

(3) Disposal when (1) & (2) happens

Step one: Keep the AM2302 sensor at condition of Temperature 50~60Celsius, humidity <10%RH for 2 hours;

Step two: After step one, keep the AM2302 sensor at condition of Temperature 20~30Celsius, humidity >70%RH for 5 hours.

(4) Attention to temperature's affection

Relative humidity strongly depend on temperature, that is why we use temperature compensation technology to ensure accurate measurement of RH. But it's still be much better to keep the sensor at same temperature when sensing.

AM2302 should be mounted at the place as far as possible from parts that may cause change to temperature.

(5) Attentions to light

Long time exposure to strong light and ultraviolet may debase AM2302's performance.

(6) Attentions to connection wires

The connection wires' quality will effect communication's quality and distance, high quality shielding-wire is recommended.

(7) Other attentions

* Welding temperature should be bellow 260Celsius.

* Avoid using the sensor under dew condition.

* Don't use this product in safety or emergency stop devices or any other occasion that failure of AM2302 may cause personal injury.

Lampiran 15. Datasheet DS18B20

Click here for production status of specific part numbers.

DS18B20

Programmable Resolution 1-Wire Digital Thermometer

General Description

The DS18B20 digital thermometer provides 9-bit to 12-bit Celsius temperature measurements and has an alarm function with nonvolatile user-programmable upper and lower trigger points. The DS18B20 communicates over a 1-Wire bus that by definition requires only one data line (and ground) for communication with a central microprocessor. In addition, the DS18B20 can derive power directly from the data line ("parasite power"), eliminating the need for an external power supply.

Each DS18B20 has a unique 64-bit serial code, which allows multiple DS18B20s to function on the same 1-Wire bus. Thus, it is simple to use one microprocessor to control many DS18B20s distributed over a large area. Applications that can benefit from this feature include HVAC environmental controls, temperature monitoring systems inside buildings, equipment, or machinery, and process monitoring and control systems.

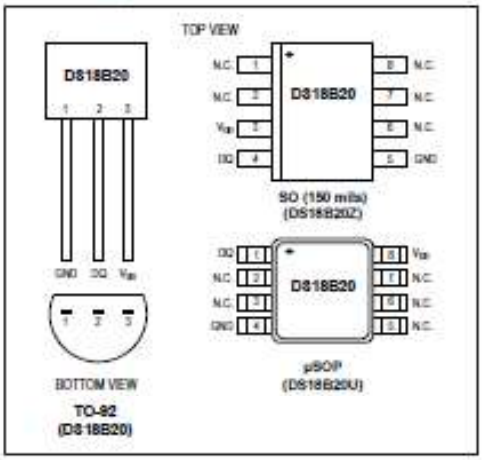
Applications

- Thermostatic Controls
- Industrial Systems
- Consumer Products
- Thermometers
- Thermally Sensitive Systems

Benefits and Features

- Unique 1-Wire® Interface Requires Only One Port Pin for Communication
- Reduce Component Count with Integrated Temperature Sensor and EEPROM
  - Measures Temperatures from -55°C to +125°C (-67°F to +257°F)
  - ±0.5°C Accuracy from -10°C to +85°C
  - Programmable Resolution from 9 Bits to 12 Bits
  - No External Components Required
- Parasitic Power Mode Requires Only 2 Pins for Operation (DQ and GND)
- Simplifies Distributed Temperature-Sensing Applications with Multidrop Capability
  - Each Device Has a Unique 64-Bit Serial Code Stored in On-Board ROM
- Flexible User-Definable Nonvolatile (NV) Alarm Settings with Alarm Search Command Identifies Devices with Temperatures Outside Programmed Limits
- Available in 8-Pin SO (150 mils), 8-Pin µSOP, and 3-Pin TO-92 Packages

Pin Configurations



Ordering Information appears at end of data sheet.

1-Wire is a registered trademark of Maxim Integrated Products, Inc.

19-7487; Rev 5; 9/18

maxim integrated.

Absolute Maximum Ratings

Voltage Range on Any Pin Relative to Ground.....-0.5V to +6.0V
Operating Temperature Range.....-55°C to +125°C
Storage Temperature Range.....-55°C to +125°C
Solder Temperature.....Refer to the IPC/JEDEC J-STD-020 Specification.

These are stress ratings only and functional operation of the device at these or any other conditions above those indicated in the operation sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods of time may affect reliability.

DC Electrical Characteristics

(-55°C to +125°C; V_{DD} = 3.0V to 5.5V)

PARAMETER	SYMBOL	CONDITIONS		MIN	TYP	MAX	UNITS
Supply Voltage	V_{DD}	Local power (Note 1)		+3.0		+5.5	V
Pullup Supply Voltage	V_{PU}	Parasite power	(Notes 1, 2)	+3.0		+5.5	V
		Local power		+3.0		V_{DD}	
Thermometer Error	t_{ERR}	-10°C to +85°C	(Note 3)			±0.5	°C
		-30°C to +100°C				±1	
		-55°C to +125°C				±2	
Input Logic-Low	V_{IL}	(Notes 1, 4, 5)		-0.3		+0.8	V
Input Logic-High	V_{IH}	Local power	(Notes 1,6)	+2.2		The lower of 5.5 or V_{DD} + 0.3	V
		Parasite power		+3.0			
Sink Current	I_L	$V_{I/O}$ = 0.4V		4.0			mA
Standby Current	I_{DDS}	(Notes 7, 8)			750	1000	nA
Active Current	I_{DD}	V_{DD} = 5V (Note 9)			1	1.5	mA
DQ Input Current	I_{DQ}	(Note 10)			5		µA
Drift		(Note 11)			±0.2		°C

Note 1: All voltages are referenced to ground.
Note 2: The Pullup Supply Voltage specification assumes that the pullup device is ideal, and therefore the high level of the pullup is equal to V_{PU}. In order to meet the V_{IH} spec of the DS18B20, the actual supply rail for the strong pullup transistor must include margin for the voltage drop across the transistor when it is turned on; thus: $V_{PU\_ACTUAL} = V_{PU\_IDEAL} + V_{TRANSISTOR}$.
Note 3: See typical performance curve in Figure 1. Thermometer Error limits are 3-sigma values.
Note 4: Logic-low voltages are specified at a sink current of 4mA.
Note 5: To guarantee a presence pulse under low voltage parasite power conditions, V_{ILMAX} may have to be reduced to as low as 0.5V.
Note 6: Logic-high voltages are specified at a source current of 1mA.
Note 7: Standby current specified up to +70°C. Standby current typically is 3µA at +125°C.
Note 8: To minimize I_{DDS}, DQ should be within the following ranges: GND ≤ DQ ≤ GND + 0.3V or V_{DD} − 0.3V ≤ DQ ≤ V_{DD}.
Note 9: Active current refers to supply current during active temperature conversions or EEPROM writes.
Note 10: DQ line is high ("high-Z" state).
Note 11: Drift data is based on a 1000-hour stress test at +125°C with V_{DD} = 5.5V.

AC Electrical Characteristics–NV Memory

(-55°C to +125°C; V_{DD} = 3.0V to 5.5V)

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
NV Write Cycle Time	t_{WR}			2	10	ms
EEPROM Writes	N_{EEWR}	-55°C to +55°C	50k			writes
EEPROM Data Retention	t_{EEDR}	-55°C to +55°C	10			years

AC Electrical Characteristics

(-55°C to +125°C; V_{DD} = 3.0V to 5.5V)

PARAMETER	SYMBOL	CONDITIONS		MIN	TYP	MAX	UNITS
Temperature Conversion Time	t_{CONV}	9-bit resolution	(Note 12)			93.75	ms
		10-bit resolution				187.5	
		11-bit resolution				375	
		12-bit resolution				750	
Time to Strong Pullup On	t_{SPON}	Start convert T command issued				10	µs
Time Slot	t_{SLOT}	(Note 12)		60		120	µs
Recovery Time	t_{REC}	(Note 12)		1			µs
Write 0 Low Time	t_{LOW0}	(Note 12)		60		120	µs
Write 1 Low Time	t_{LOW1}	(Note 12)		1		15	µs
Read Data Valid	t_{RDV}	(Note 12)				15	µs
Reset Time High	t_{RSTH}	(Note 12)		480			µs
Reset Time Low	t_{RSTL}	(Notes 12, 13)		480			µs
Presence-Detect High	t_{PDHIGH}	(Note 12)		15		60	µs
Presence-Detect Low	t_{PDLOW}	(Note 12)		60		240	µs
Capacitance	$C_{IN/OUT}$					25	pF

Note 12: See the timing diagrams in Figure 2.
Note 13: Under parasite power, if t_{RSTL} > 960µs, a power-on reset can occur.

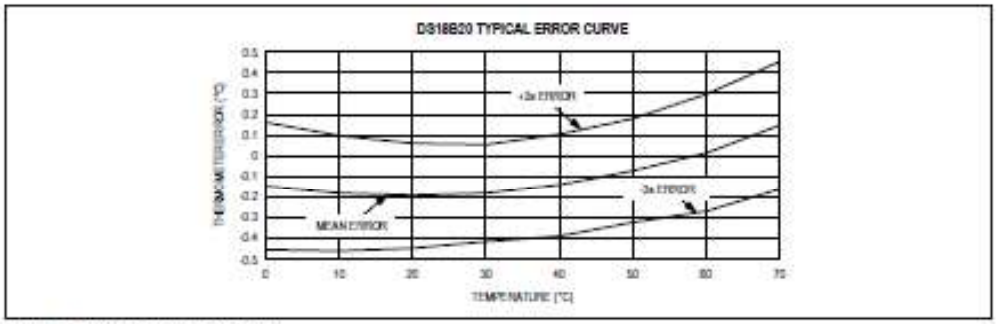


Figure 1. Typical Performance Curve

Lampiran 16. Datasheet Ultrasonik

HC-SR04 User Guide

1. Ultrasonic Distance Measurement Principles

The transmitter emits a 8 bursts of an directional 40KHz ultrasonic wave when triggered and starts a timer. Ultrasonic pulses travel outward until they encounter an object, The object causes the the wave to be reflected back towards the unit. The ultrasonic receiver would detect the reflected wave and stop the stop timer. The velocity of the ultrasonic burst is 340m/sec. in air. Based on the number of counts by the timer, the distance can be calculated between the object and transmitter The TRD Measurement formula is expressed as: D = C X T which is know as the time/rate/distance measurement formula where D is the measured distance, and R is the propagation velocity (Rate) in air (speed of sound) and T represents time. In this application T is devided by 2 as T is double the time value from transmitter to object back to receiver.

2. Product Features

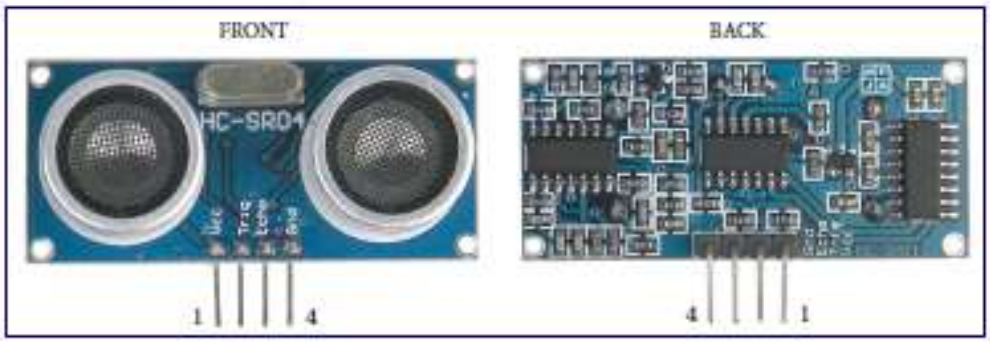
Features

- Stable performance (Xtal.)
- Accurate distance measurement
- High-density SMD Board
- Close Range (2cm)

Uses

- Robotics barrier
- Object distance measurement
- Level detection
- Security systems
- Vehicle detection/avoidance

3. Product Views



4. Module Pin Asignments

	Pin Symbol	Pin Function Description
1	VCC	5V power supply
2	Trig	Trigger Input pin
3	Echo	Receiver Output pin
4	GND	Power ground

5. Electrical Specifications

WARARNING

Do Not connect Module with Power Applied! Always apply power after connecting Connect "GND" Terminal first

Electrical Parameters	HC-SR04 Ultrasonic Module
Operating Voltage	5VDC
Operating Current	15mA
Operating Frequency	40KHz
Max. Range	4m
Nearest Range	2cm
Measuring Angle	15 Degrees
Input Trigger Signal	10us min. TTL pulse
Output Echo Signal	TTL level signal, proportional to distance
Board Dimensions	1-13/16" X 13/16" X 5/8"
Board Connections	4 X 0.1" Pitch Right Angle Header Pins

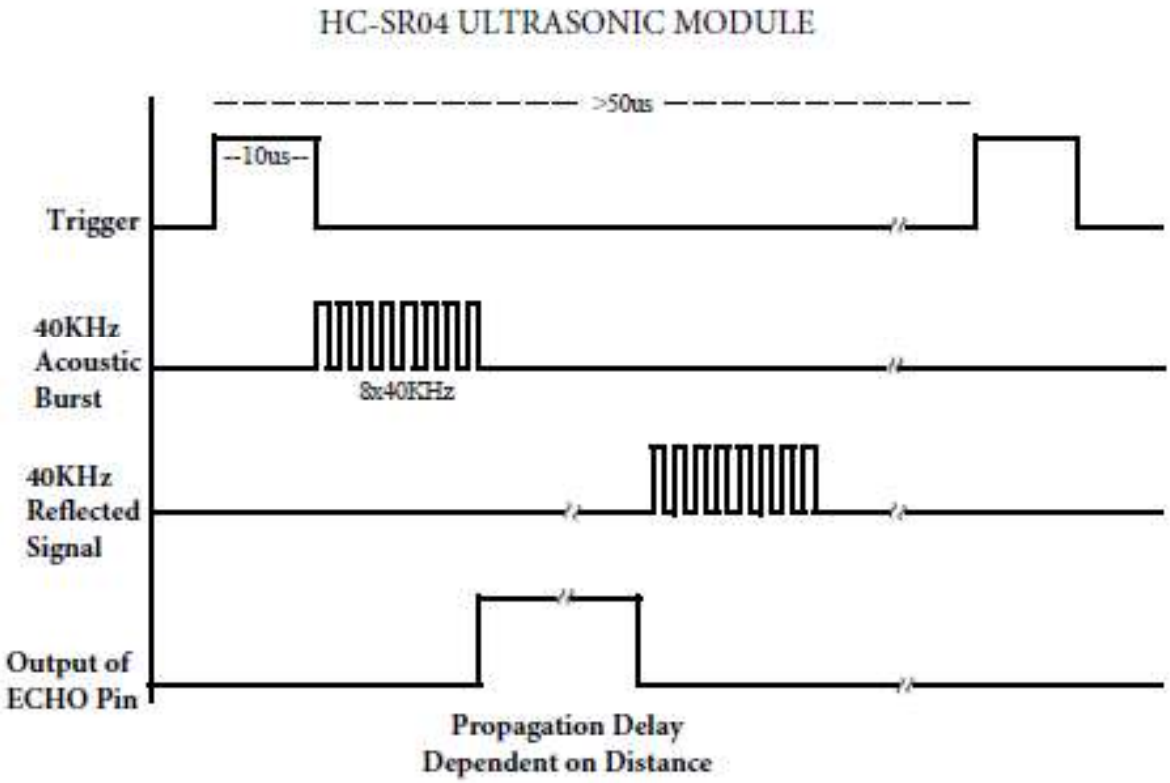
6. Module Operation

Set Trig and Echo Low to initalize module. Place a minimum 10us High level pulse to "Trigger" (module will automatically send eight 40KHz acoustic bursts). At the same time, Gate the microcontroller timer to start timing.

Wait to capture the rising edge output of ECHO port to stop the timer. Now read the time of the counter, which is the ultrasonic propagation time in the air. According to the formula: Distance = (ECHO high level time X ultrasonic velocity (Speed of Sound in air 340m/sec) / 2, you can calculate the distance to the obstacle.

For best results and maximum range, the Object should be larger than 0.5M^2 the nearer the target object, the smaller it may be

7. ModuleTiming



Trigger 10us min. start measurement from microcontroller.
Max Rep. Rate: 50us

ECHO Output pulse to microcontroller, width is the time from last of 8 40KHz bursts to detected reflected signal (microcontroller Timer gate signal)
Distance in cm = echo pulse width in uS/58
Distance in inch = echo pulse width in uS/148