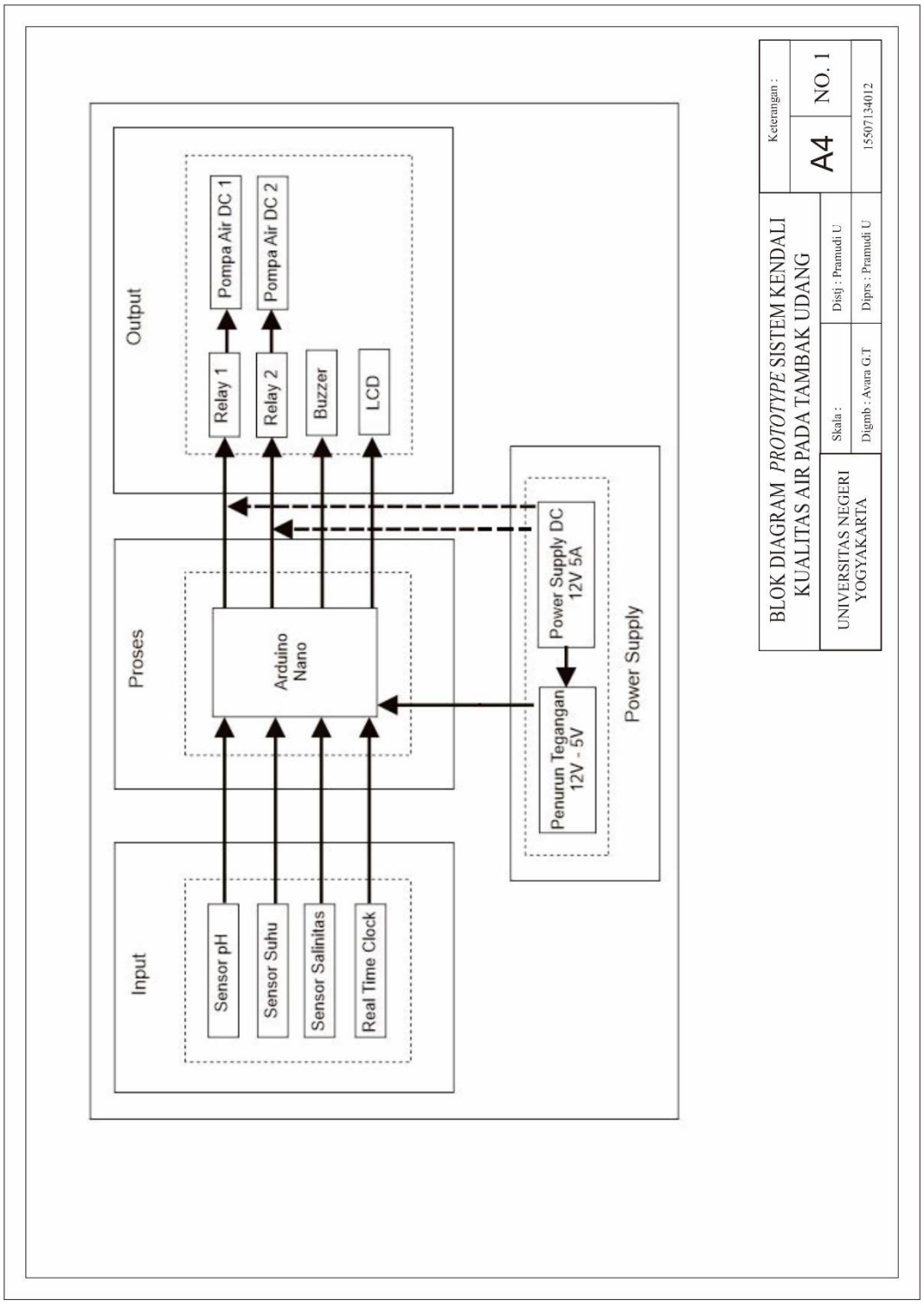
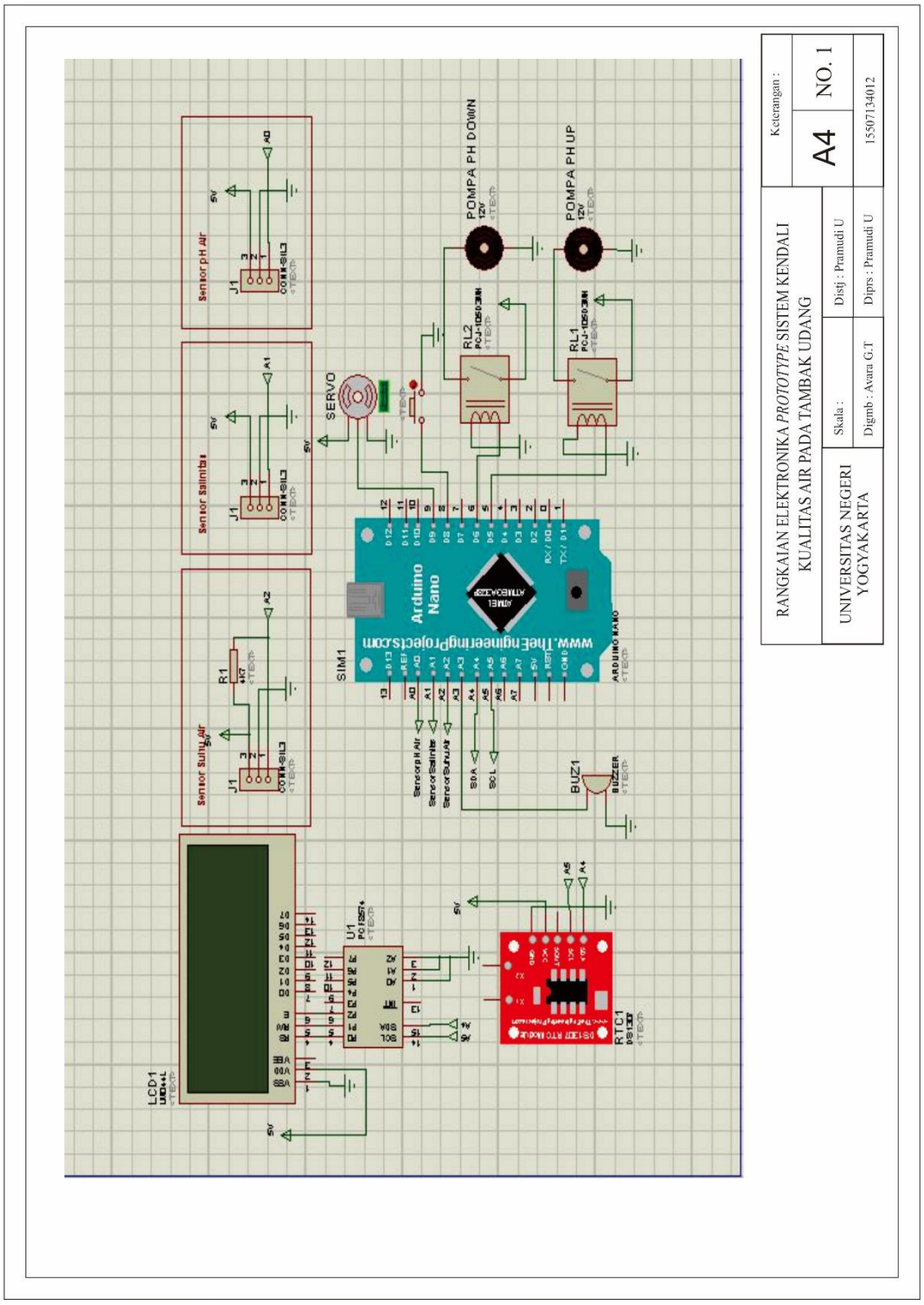
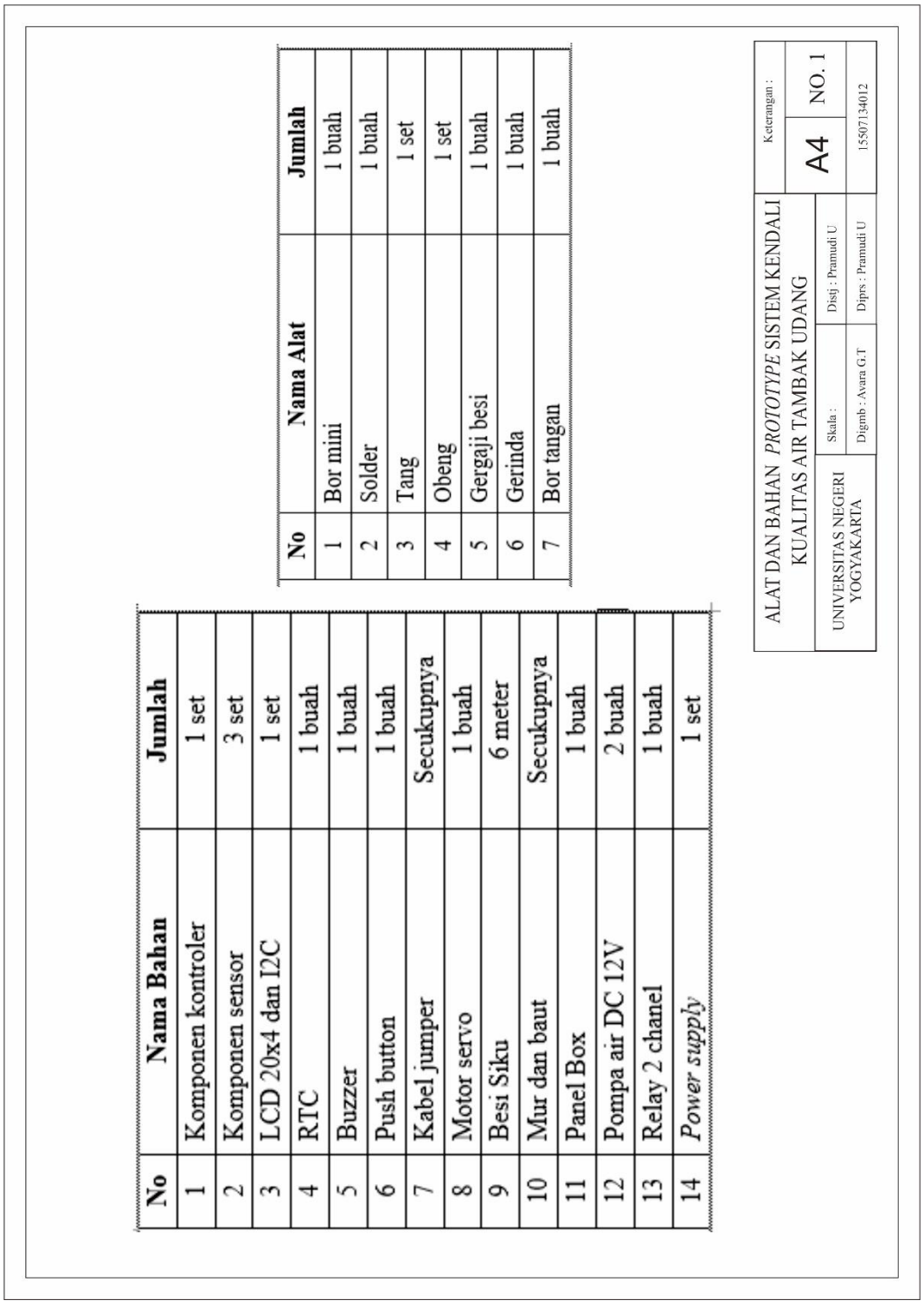
Lampiran 1. Blok Diagram



Lampiran 2. Rangkaian Elektronik



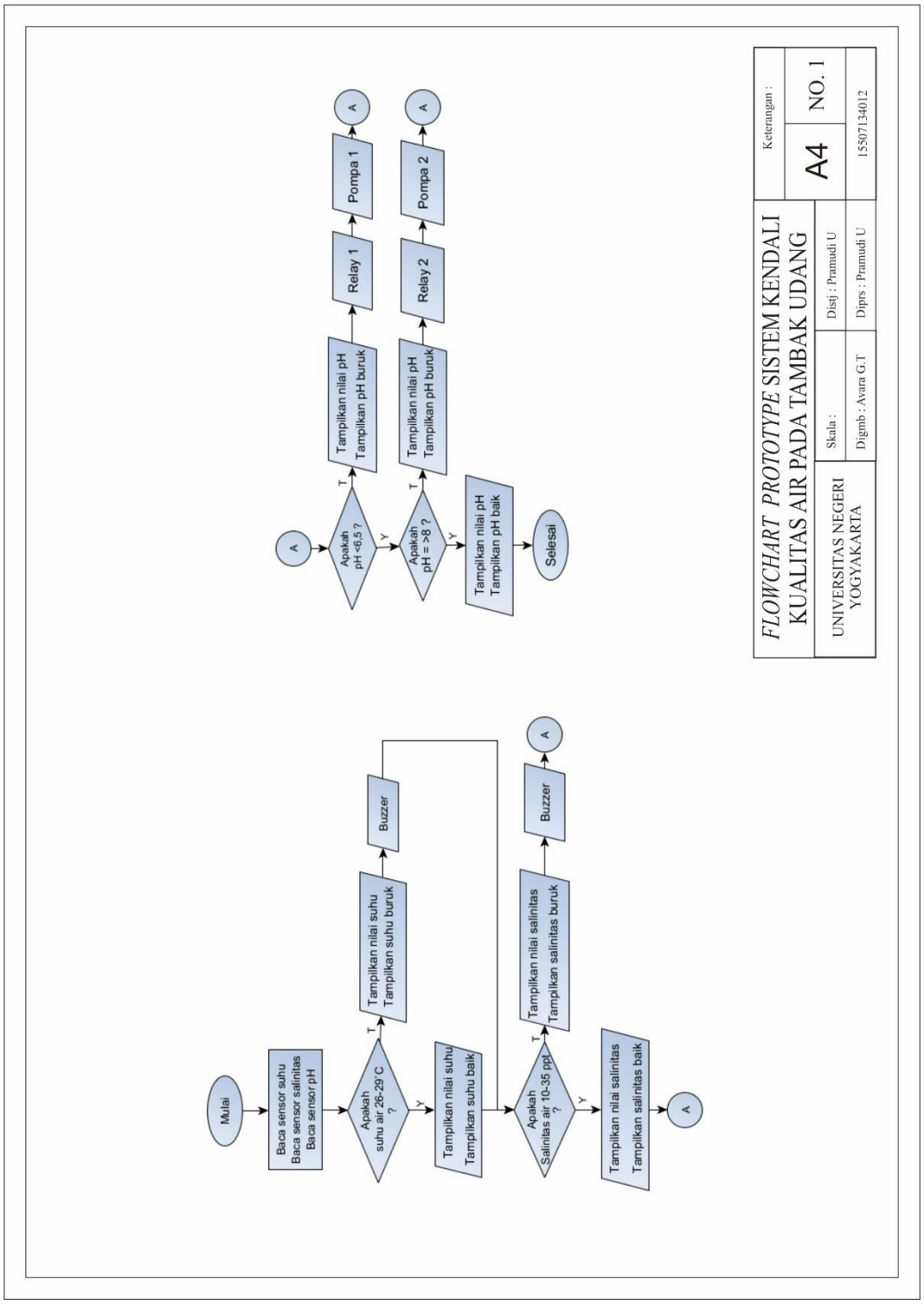
Lampiran 3. Alat dan bahan



No	Nama Bahan	Jumlah
1	Komponen kontroler	1 set
2	Komponen sensor	3 set
3	LCD 20x4 dan I2C	1 set
4	RTC	1 buah
5	Buzzer	1 buah
6	Push button	1 buah
7	Kabel jumper	Secukupnya
8	Motor servo	1 buah
9	Besi Siku	6 meter
10	Mur dan baut	Secukupnya
11	Panel Box	1 buah
12	Pompa air DC 12V	2 buah
13	Relay 2 chanel	1 buah
14	*Power supply*	1 set

No	Nama Alat	Jumlah
1	Bor mini	1 buah
2	Solder	1 buah
3	Tang	1 set
4	Obeng	1 set
5	Gergaji besi	1 buah
6	Gerinda	1 buah
7	Bor tangan	1 buah

Lampiran 4. Flowchart



Lampiran 5. Program

```
#include <Wire.h> //lcd
#include <LiquidCrystal_I2C.h>
#include <OneWire.h>  //suhu
#include <DallasTemperature.h>
#include <EEPROM.h>
#define analogInPin A1 //salinitas
#define ONE_WIRE_BUS A2
#define relay1 2
#define relay2 3
#define button 8
#define BUZZER A3
#include <Servo.h>
#include "RTClib.h"
 int adc;
 float teg;
 float hasil;
 const int numReadings1 = 10;
 float readings1[numReadings1];
 int readIndex1 = 0;
 float total1 = 0;
 float average1 = 0;
 float Salin;

 RTC_DS1307 rtc;
 char daysOfTheWeek[7][12] = {"Ming",
"Sen", "Sel", "Rab", "Kam", "Jum",
"Sab"};
LiquidCrystal_I2C lcd(0x3F, 2, 1, 0, 4, 5,
6, 7, 3, POSITIVE);  // Atur LCD I2C
address

Servo myservo;
int pos = 0;
void setup() {
//Serial.begin(9600);
lcd.begin(20,4);
lcd.setCursor(1,1);
lcd.print("TEKNIK ELEKTRONIKA");
lcd.setCursor(7,2);
lcd.print("(2015)");
lcd.setCursor(0,3);
delay(100);
lcd.print("_");
delay(100);
lcd.print("_");
delay(100);

 pinMode(relay1,OUTPUT);
 pinMode(relay2,OUTPUT);
 pinMode(BUZZER, OUTPUT);
 pinMode(button, INPUT_PULLUP);

lcd.print("_");
delay(100);
lcd.print("_");
delay(100);
lcd.print("_");
delay(100);
lcd.print("_");
delay(100);
lcd.print("_");
delay(100);
lcd.print("_");
delay(100);
lcd.print("_");
delay(100);

 digitalWrite(relay1,HIGH);
 digitalWrite(relay2,HIGH);
```

```
  digitalWrite(BUZZER, LOW);
  digitalWrite(BUZZER, HIGH);
  delay(300);
  digitalWrite(BUZZER, LOW);
  delay(200);
  digitalWrite(BUZZER, HIGH);
  delay(300);
  digitalWrite(BUZZER, LOW);

 lcd.print("_");
 delay(100);
 lcd.print("_");
 delay(100);
 lcd.print("_");
 delay(100);
 lcd.print("_");
 delay(100);
 lcd.print("_");
 delay(100);
 lcd.print("_");
 delay(100);
 lcd.print("_");
 delay(100);
 lcd.print("_");
 delay(100);
 lcd.print("_");
 delay(100);


  myservo.attach(9);
  for (pos = 0; pos <= 90; pos += 1) { //
goes from 0 degrees to 180 degrees
   // in steps of 1 degree
   myservo.write(pos);          // tell
servo to go to position in variable 'pos'
   delay(25);                   // waits 15ms
for the servo to reach the position
  }
  delay (1000);


 lcd.print("_");
 delay(100);
 lcd.print("_");
 delay(100);
 lcd.print("_");
 delay(400);
 lcd.print("_");
 delay(400);


 DateTime now = rtc.now();


 Serial.begin(9600);
 if (! rtc.begin()) {
  Serial.println("Couldn't find RTC");
  while (1);
 }


 if (! rtc.isrunning()) {
  Serial.println("RTC is NOT running!");
  //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));
 }


 lcd.clear();
 lcd.setCursor(1,1);
 lcd.print("AVARA GHAMALIEL TY");
```

```
 lcd.setCursor(3,2);
 lcd.print("(15507134012)");
 delay(5000);
 lcd.clear();
 lcd.setCursor(5,1);
 lcd.print("LOADING");
 delay(500);
 lcd.print(".");
 delay(500);
 lcd.print(".");
 delay(500);
 lcd.print(".");
 delay(500);
 proses();

 digitalWrite(BUZZER, HIGH);
 delay(500);
 digitalWrite(BUZZER, LOW);

 EEPROM.write(5,now.minute());
}

void loop(){
DateTime now = rtc.now();

 OneWire oneWire(ONE_WIRE_BUS);
 DallasTemperature
sensors(&oneWire);
 float suhu;
 sensors.requestTemperatures();
 suhu = sensors.getTempCByIndex(0);
 EEPROM.write(2, suhu);

 Serial.print("suhu: ");
 Serial.println(suhu);

 lcd.setCursor(1,0);
 lcd.print("MONITORING  TAMBAK");
 lcd.setCursor(0,1);
 lcd.print("_______UDANG________");
 lcd.setCursor(0,2);
 lcd.print("Suhu:");
 lcd.setCursor(5,2);
 lcd.print(suhu,0);
 lcd.print(" C");
 lcd.setCursor(0,3);
 lcd.print("Sal :");
 lcd.setCursor(5,3);
 lcd.print(Salin,0);
 lcd.print(" PPT");
 lcd.setCursor(11,2);
 lcd.print("|");
 lcd.setCursor(11,3);
 lcd.print("|");
 lcd.setCursor(12,2);
 lcd.print("pH:");
 lcd.setCursor(16,2);
 float ph=EEPROM.read(1);
 ph=ph/10;
 lcd.print(ph);
 lcd.setCursor(13,3);
 lcd.print(now.hour(), DEC);
 lcd.print(":");
 lcd.print(now.minute(), DEC);
 lcd.print(":");
 lcd.print(now.second(), DEC);
 delay(1000);
```

```
 timer();
}
void pompa(){
float ph=EEPROM.read(1);
ph=ph/10;
int i=0;
int j=0;
  if(ph<6){
    lcd.clear();
    lcd.setCursor(1,1);
    lcd.print("PH TERLALU RENDAH");
    delay(1000);
    lcd.setCursor(5,2);
    lcd.print("LOADING");
    delay(500);
    lcd.print(".");
    delay(500);
    lcd.print(".");
    delay(500);
    lcd.print(".");
    delay(500);
    i=0;
    do{
     lcd.clear();
     lcd.setCursor(0,1);
     lcd.print("Tambah PH Up");
     delay(500);
     lcd.print(".");
     delay(500);
     lcd.print(".");
     delay(500);
     lcd.print(".");
     delay(500);
     digitalWrite(relay1,LOW);
     delay(3000);
     digitalWrite(relay1,HIGH);
     delay(5000);
     j=0;
     do{
     lcd.setCursor(5,3);
     lcd.print("PH : ");
     lcd.print(sensorPo());
      delay(200);
      j=j+1;
     }while(j<50);
     ph=EEPROM.read(1);
     ph=ph/10;
     if(ph<6){
      i=i+1;
     }else{
      i=5;
     }
     lcd.clear();
     lcd.setCursor(7,1);
     lcd.print("PH : ");
     lcd.print(ph);
     delay(2000);
    }while(i<5);
  }else if(ph>8){
    lcd.clear();
    lcd.setCursor(1,1);
    lcd.print("PH TERLALU TINGGI");
    delay(1000);
    lcd.setCursor(5,2);
    lcd.print("LOADING");
    delay(500);
    lcd.print(".");
```

```
delay(500);
lcd.print(".");
delay(500);
lcd.print(".");
delay(500);
i=0;
do{
 lcd.clear();
 lcd.setCursor(0,1);
 lcd.print("Tambah PH Down");
 delay(500);
 lcd.print(".");
 delay(500);
 lcd.print(".");
 delay(500);
 lcd.print(".");
 delay(500);

 digitalWrite(relay2,LOW);
 delay(3000);
 digitalWrite(relay2,HIGH);
 delay(5000);
 j=0;
  do{
  lcd.setCursor(5,3);
  lcd.print("PH : ");
  lcd.print(sensorPo());
    delay(200);
    j=j+1;
  }while(j<50);
 ph=EEPROM.read(1);
 ph=ph/10;
 if(ph>8){
  i=i+1;
       }else{
        i=5;
       }

       lcd.clear();
       lcd.setCursor(5,1);
       lcd.print("PH : ");
       lcd.print(ph);
       delay(2000);
      }while(i<5);
    }
}
void proses(){
  int pos = 0;
  int i=5;
  int j=0;

  lcd.clear();
  lcd.setCursor(5,1);
  lcd.print("MONITORING");

    for (pos = 90; pos >= 0; pos -= 1) { //
goes from 180 degrees to 0 degrees
      myservo.write(pos);           // tell
servo to go to position in variable 'pos'
      delay(25);                    // waits 15ms
for the servo to reach the position
    }
    delay(300);

    lcd.clear();
    lcd.setCursor(2,1);
    lcd.print("UJI SALINITAS");
    delay(500);
    j=0;
```

```
  do{
   Salin += sensorSalinitas();
   for(int k=15;k<19;k++){
     delay(200);
     lcd.setCursor(k,1);
     lcd.print(".");
     lcd.setCursor(1,3);
     lcd.print("Salinitas: ");
     lcd.print(sensorSalinitas(),0);
     lcd.print(" PPT");
   }
   lcd.setCursor(15,1);
   lcd.print("    ");
   j=j+1;
  }while(j<50);
  Salin=Salin/50;
  lcd.clear();
  lcd.setCursor(1,1);
  lcd.print("SALINITAS: ");
  lcd.print(Salin,0);
  lcd.print(" PPT");

  for (pos = 0; pos <= 90; pos += 1) { //
goes from 0 degrees to 180 degrees
  // in steps of 1 degree
    myservo.write(pos);          // tell
servo to go to position in variable 'pos'
    delay(25);                   // waits 15ms
for the servo to reach the position
  }
  delay(300);
  for (pos = 90; pos <= 180; pos += 1) {
// goes from 0 degrees to 180 degrees
    // in steps of 1 degree
    myservo.write(pos);          // tell
servo to go to position in variable 'pos'
    delay(25);                   // waits 15ms
for the servo to reach the position
  }

  lcd.clear();
  lcd.setCursor(5,1);
  lcd.print("UJI PH");
  delay(500);
  j=0;
  do{
   for(int k=11;k<15;k++){
     delay(200);
     lcd.setCursor(k,1);
     lcd.print(".");
     lcd.setCursor(5,3);
     lcd.print("PH: ");
     lcd.print(sensorPo());
   }
   lcd.setCursor(11,1);
   lcd.print("    ");
   j=j+1;
  }while(j<50);
  pompa();
  delay(300);

  float Po = EEPROM.read(1);
  Po=Po/10;
  lcd.clear();
  lcd.setCursor(5,1);
  lcd.print("PH : ");
  lcd.print(Po);

  for (pos = 180; pos >= 75; pos -= 1) { //
goes from 180 degrees to 0 degrees
```

```
    myservo.write(pos);              // tell
servo to go to position in variable 'pos'
    delay(25);                       // waits 15ms
for the servo to reach the position
  }

  lcd.clear();
  lcd.setCursor(4,1);
  lcd.print("UJI SUHU");

  delay(500);
  lcd.print(".");
  delay(500);
  lcd.print(".");
  delay(500);
  lcd.print(".");
  delay(2000);

  sensorSuhu();

  lcd.clear();
  int Suhu = EEPROM.read(2);
  lcd.setCursor(5,1);
  lcd.print("SUHU : ");
  lcd.print(Suhu);
  lcd.print(" C");

  lcd.clear();

  lcd.setCursor(3,1);
  lcd.print("PERHITUNGAN");
  delay(500);
  lcd.print(".");
  delay(500);
  lcd.print(".");
  delay(500);
  lcd.print(".");
  delay(2000);
  lcd.clear();

  i=0;
  do{
    if(Suhu>30 && Salin>33){
      lcd.setCursor(0,0);
      lcd.print("SUHU TERLALU
TINGGI!");
      lcd.setCursor(0,1);
      lcd.print("SUHU : ");
      lcd.print(Suhu,0);
      lcd.print(" C");
      lcd.setCursor(0,2);
      lcd.print("SALINITAS TINGGI!");
      lcd.setCursor(0,3);
      lcd.print("SALINITAS : ");
      lcd.print(Salin,0);
      lcd.print(" PPT");
      digitalWrite(BUZZER, HIGH);
      i=0;
    }else if(Suhu>30 && Salin<10){
      lcd.setCursor(0,0);
      lcd.print("SUHU TERLALU
TINGGI!");
      lcd.setCursor(0,1);
      lcd.print("SUHU : ");
      lcd.print(Suhu,0);
      lcd.print(" C");
      lcd.setCursor(0,2);
      lcd.print("SALINITAS RENDAH!");
      lcd.setCursor(0,3);
      lcd.print("SALINITAS : ");
```

```
      lcd.print(Salin,0);
      lcd.print(" PPT");
      digitalWrite(BUZZER, HIGH);
      i=0;
    }else if(Suhu<25 && Salin>33){
      lcd.setCursor(0,0);
      lcd.print("SUHU TERLALU
RENDAH!");
      lcd.setCursor(0,1);
      lcd.print("SUHU : ");
      lcd.print(Suhu,0);
      lcd.print(" C");
      lcd.setCursor(0,2);
      lcd.print("SALINITAS TINGGI!");
      lcd.setCursor(0,3);
      lcd.print("SALINITAS : ");
      lcd.print(Salin,0);
      lcd.print(" PPT");
      digitalWrite(BUZZER, HIGH);
      i=0;
    }else if(Suhu<25 && Salin<10){
      lcd.setCursor(0,0);
      lcd.print("SUHU TERLALU
RENDAH!");
      lcd.setCursor(0,1);
      lcd.print("SUHU : ");
      lcd.print(Suhu,0);
      lcd.print(" C");
      lcd.setCursor(0,2);
      lcd.print("SALINITAS RENDAH!");
      lcd.setCursor(0,3);
      lcd.print("SALINITAS : ");
      lcd.print(Salin,0);
      lcd.print(" PPT");
      digitalWrite(BUZZER, HIGH);
      i=0;
    }else if(Suhu>30){
      lcd.setCursor(0,1);
      lcd.print("SUHU TERLALU
TINGGI!");
      lcd.setCursor(0,2);
      lcd.print("SUHU : ");
      lcd.print(Suhu,0);
      lcd.print(" C");
      digitalWrite(BUZZER, HIGH);
      i=0;
    }else if(Suhu<25){
      lcd.setCursor(0,1);
      lcd.print("SUHU TERLALU
RENDAH!");
      lcd.setCursor(0,2);
      lcd.print("SUHU : ");
      lcd.print(Suhu,0);
      lcd.print(" C");
      digitalWrite(BUZZER, HIGH);
      i=0;
    }else if(Salin>33){
      lcd.setCursor(0,1);
      lcd.print("SALINITAS TINGGI!");
      lcd.setCursor(0,2);
      lcd.print("SALINITAS : ");
      lcd.print(Salin,0);
      lcd.print(" PPT");
      digitalWrite(BUZZER, HIGH);
      i=0;
    }else if(Salin<10){
      lcd.setCursor(0,1);
      lcd.print("SALINITAS RENDAH!");
      lcd.setCursor(0,2);
      lcd.print("SALINITAS : ");
```

```
      lcd.print(Salin,0);
      lcd.print(" PPT");
      digitalWrite(BUZZER, HIGH);
      i=0;
    }else{
      lcd.clear();
      lcd.setCursor(7,1);
      lcd.print("NORMAL");
      i=3;
    }

    int buttonState;
    buttonState = digitalRead(button);

    if(buttonState == LOW){
      lcd.clear();
      lcd.setCursor(3,1);
      lcd.print("ALARM MATI !!!");
      digitalWrite(BUZZER, LOW);
      delay(3000);
      i=3;
    }

   }while(i<2);
}
float sensorPo(){
 float Value;
 Value = analogRead(A0);
 Value = (Value*4.85)/1023;
 float pH;
 pH = (-5.45546*Value)+24.20507;
 float tes=pH*10;
 Serial.print("ph: ");
 Serial.println(pH);
 EEPROM.write(1, tes);
 return (pH);
}

float sensorSalinitas(){
 adc = analogRead(A1);
 teg = (adc*4.85)/1023;
 hasil = (teg*14.21)-13.611;
 if (hasil<0){hasil=0;}
 total1 = total1 - readings1[readIndex1];
 readings1[readIndex1] = hasil;
 total1 = total1 +
readings1[readIndex1];
 readIndex1 = readIndex1 + 1;
 if (readIndex1 >= numReadings1)
{readIndex1 = 0;}
 average1 = total1 / numReadings1;
 return(average1);
}

void sensorSuhu(){
 OneWire oneWire(ONE_WIRE_BUS);
 DallasTemperature
sensors(&oneWire);
 float suhu;
 sensors.requestTemperatures();
 suhu = sensors.getTempCByIndex(0);
 EEPROM.write(2, suhu);
}
void ph(){
float pH=EEPROM.read(1);
pH=pH/10;
lcd.setCursor(0,0);
lcd.print("ph:");
lcd.print(pH);
```

```
}

void suhu(){
int suhu=EEPROM.read(2);
lcd.setCursor(8,0);
lcd.print("suhu:");
lcd.print(suhu);
}

void salinitas ()
{
float valSalinitas=sensorSalinitas();
lcd.setCursor(0,1);
lcd.print("Salinitas:");
lcd.print(valSalinitas);

Serial.print("sal=");
Serial.println(valSalinitas);
}
void timer(){
 DateTime now = rtc.now();
 int i=5;
 int waktuawal=EEPROM.read(i);
 int waktukini=now.minute();
 int waktucek=waktuawal+5;//menit

 if(waktucek>60){
  waktucek=waktucek-60;
 }
 if(waktukini==waktucek){
  proses();
  EEPROM.write(i,waktukini);
 }
}
```

Lampiran 6. Gambar Alat

Tampak Depan

Tampak Atas

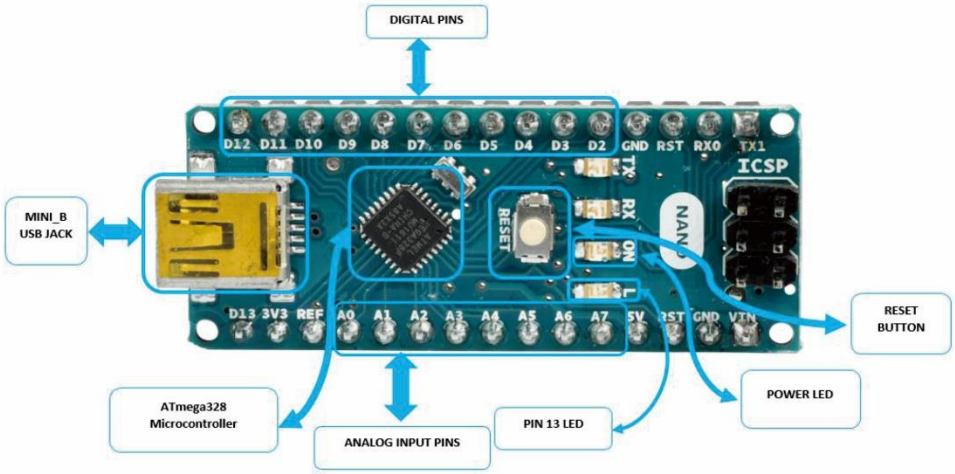
Box Komponen

Gambar Alat			Keterangan :	
UNIVERSITAS NEGERI YOGYAKARTA	Skala :	Distj : Pramudi U	A4	NO. 1
	Digmb : Avara G.T	Diprs : Pramudi U	15507134018	

Lampiran 7. Pengoprasian dan Spesifikasi Alat

Lampiran 8. Datasheet Arduino Nano

ARDUINO NANO



INTRODUCTION

Arduino nano differ from other Arduino as it very small so it suitable for small sized projects and it supports breadboards so it can be plugged with other components in only one breadboard.

ARDUINO NANO PHYSICAL COMPONENTS

Microcontroller

In Arduino Nano 2.x version, still used ATmega168 microcontroller while the Arduino Nano 3.x version already used ATmega328 microcontroller.

1

- SCK (Serial Clock) - The clock pulses which synchronize data transmission generated by the master and one-line specific for every device:
- SS (Slave Select) - the pin on each device that the master can use to enable and disable specific devices.

When a device's Slave Select pin is low, it communicates with the master. When it's high, it ignores the master. This allows you to have multiple SPI devices sharing the same MISO, MOSI, and CLK lines.

Arduino Nano Specifications

Microcontroller	Atmel ATmega168 or ATmega328
Operating Voltage (logic level)	5 V
Input Voltage (recommended)	7-12 V
Input Voltage (limits)	6-20 V
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	8
DC Current per I/O Pin	40 mA
Flash Memory	16 KB (ATmega168) or 32 KB (ATmega328) of which 2 KB used by bootloader
SRAM	1 KB (ATmega168) or 2 KB (ATmega328)
EEPROM	512 bytes (ATmega168) or 1 KB (ATmega328)
Clock Speed	16 MHz
Dimensions	0.73" x 1.70"
Length	45 mm
Width	18 mm
Weigth	5 g

Lampiran 9. Datasheet DS18b20

PRELIMINARY

DS18B20
Programmable Resolution 1-Wire® Digital Thermometer

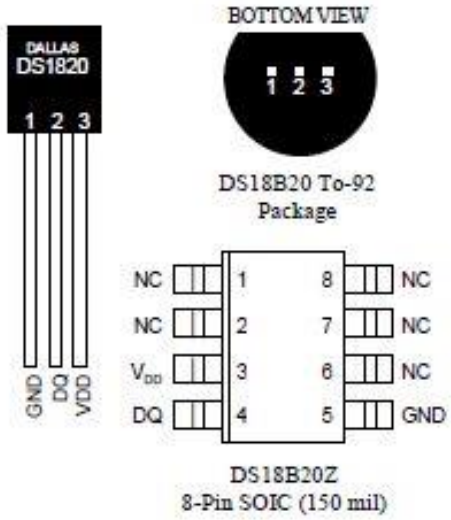
DALLAS SEMICONDUCTOR

www.dalsemi.com

FEATURES

- Unique 1-Wire interface requires only one port pin for communication
- Multidrop capability simplifies distributed temperature sensing applications
- Requires no external components
- Can be powered from data line. Power supply range is 3.0V to 5.5V
- Zero standby power required
- Measures temperatures from -55°C to +125°C. Fahrenheit equivalent is -67°F to +257°F
- ±0.5°C accuracy from -10°C to +85°C
- Thermometer resolution is programmable from 9 to 12 bits
- Converts 12-bit temperature to digital word in 750 ms (max.)
- User-definable, nonvolatile temperature alarm settings
- Alarm search command identifies and addresses devices whose temperature is outside of programmed limits (temperature alarm condition)
- Applications include thermostatic controls, industrial systems, consumer products, thermometers, or any thermally sensitive system

PIN ASSIGNMENT



PIN DESCRIPTION

GND - Ground
DQ - Data In/Out
V_{DD} - Power Supply Voltage
NC - No Connect

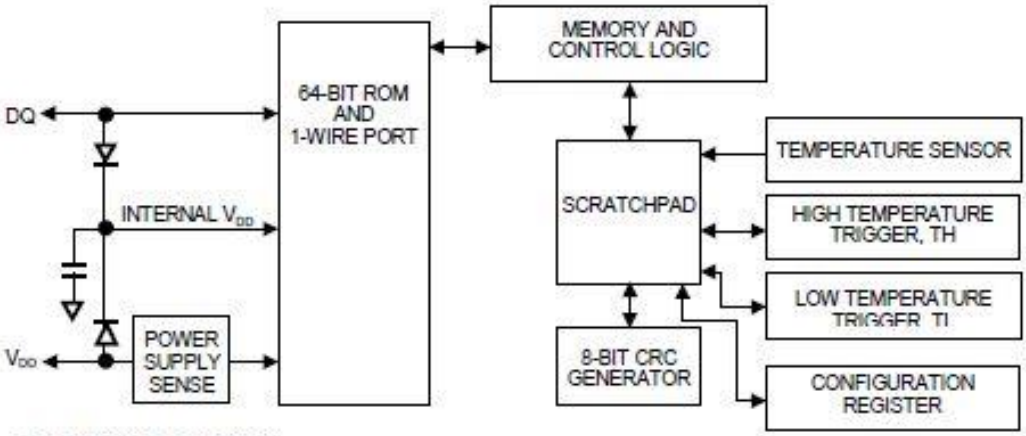
DESCRIPTION

The DS18B20 Digital Thermometer provides 9 to 12-bit (configurable) temperature readings which indicate the temperature of the device.

Information is sent to/from the DS18B20 over a 1-Wire interface, so that only one wire (and ground) needs to be connected from a central microprocessor to a DS18B20. Power for reading, writing, and performing temperature conversions can be derived from the data line itself with no need for an external power source.

Because each DS18B20 contains a unique silicon serial number, multiple DS18B20s can exist on the same 1-Wire bus. This allows for placing temperature sensors in many different places. Applications where this feature is useful include HVAC environmental controls, sensing temperatures inside buildings, equipment or machinery, and process monitoring and control.

DS18B20 BLOCK DIAGRAM Figure 1



PARASITE POWER

The block diagram (Figure 1) shows the parasite-powered circuitry. This circuitry "steals" power whenever the DQ or V_{DD} pins are high. DQ will provide sufficient power as long as the specified timing and voltage requirements are met (see the section titled "1-Wire Bus System"). The advantages of parasite power are twofold: 1) by parasiting off this pin, no local power source is needed for remote sensing of temperature, and 2) the ROM may be read in absence of normal power.

In order for the DS18B20 to be able to perform accurate temperature conversions, sufficient power must be provided over the DQ line when a temperature conversion is taking place. Since the operating current of the DS18B20 is up to 1.5 mA, the DQ line will not have sufficient drive due to the 5k pullup resistor. This problem is particularly acute if several DS18B20s are on the same DQ and attempting to convert simultaneously.

There are two ways to assure that the DS18B20 has sufficient supply current during its active conversion cycle. The first is to provide a strong pullup on the DQ line whenever temperature conversions or copies to the E^2 memory are taking place. This may be accomplished by using a MOSFET to pull the DQ line directly to the power supply as shown in Figure 2. The DQ line must be switched over to the strong pull-up within 10 µs maximum after issuing any protocol that involves copying to the E^2 memory or initiates temperature conversions. When using the parasite power mode, the V_{DD} pin must be tied to ground.

Another method of supplying current to the DS18B20 is through the use of an external power supply tied to the V_{DD} pin, as shown in Figure 3. The advantage to this is that the strong pullup is not required on the DQ line, and the bus master need not be tied up holding that line high during temperature conversions. This allows other data traffic on the 1-Wire bus during the conversion time. In addition, any number of DS18B20s may be placed on the 1-Wire bus, and if they all use external power, they may all simultaneously perform temperature conversions by issuing the Skip ROM command and then issuing the Convert T command. Note that as long as the external power supply is active, the GND pin may not be floating.

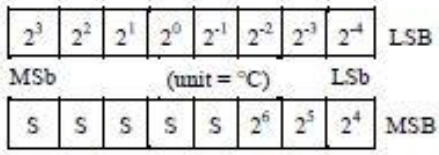
The use of parasite power is not recommended above 100°C, since it may not be able to sustain communications given the higher leakage currents the DS18B20 exhibits at these temperatures. For applications in which such temperatures are likely, it is strongly recommended that V_{DD} be applied to the DS18B20.

OPERATION - MEASURING TEMPERATURE

The core functionality of the DS18B20 is its direct-to-digital temperature sensor. The resolution of the DS18B20 is configurable (9, 10, 11, or 12 bits), with 12-bit readings the factory default state. This equates to a temperature resolution of 0.5°C, 0.25°C, 0.125°C, or 0.0625°C. Following the issuance of the Convert T [44h] command, a temperature conversion is performed and the thermal data is stored in the scratchpad memory in a 16-bit, sign-extended two's complement format. The temperature information can be retrieved over the 1-Wire interface by issuing a Read Scratchpad [BEh] command once the conversion has been performed. The data is transferred over the 1-Wire bus, LSB first. The MSB of the temperature register contains the "sign" (S) bit, denoting whether the temperature is positive or negative.

Table 2 describes the exact relationship of output data to measured temperature. The table assumes 12-bit resolution. If the DS18B20 is configured for a lower resolution, insignificant bits will contain zeros. For Fahrenheit usage, a lookup table or conversion routine must be used.

Temperature/Data Relationships Table 2



TEMPERATURE	DIGITAL OUTPUT (Binary)	DIGITAL OUTPUT (Hex)
+125°C	0000 0111 1101 0000	07D0h
+85°C	0000 0101 0101 0000	0550h*
+25.0625°C	0000 0001 1001 0001	0191h
+10.125°C	0000 0000 1010 0010	00A2h
+0.5°C	0000 0000 0000 1000	0008h
0°C	0000 0000 0000 0000	0000h
-0.5°C	1111 1111 1111 1000	FFF8h
-10.125°C	1111 1111 0101 1110	FF5Eh
-25.0625°C	1111 1110 0110 1111	FF6Fh
-55°C	1111 1100 1001 0000	FC90h

*The power on reset register value is +85°C.

OPERATION - ALARM SIGNALING

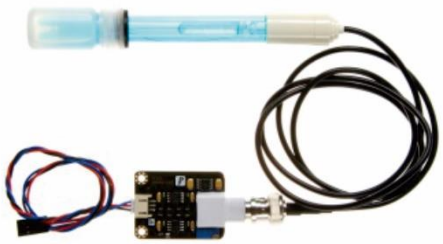
After the DS18B20 has performed a temperature conversion, the temperature value is compared to the trigger values stored in TH and TL. Since these registers are 8-bit only, bits 9-12 are ignored for comparison. The most significant bit of TH or TL directly corresponds to the sign bit of the 16-bit temperature register. If the result of a temperature measurement is higher than TH or lower than TL, an alarm flag inside the device is set. This flag is updated with every temperature measurement. As long as the alarm flag is set, the DS18B20 will respond to the alarm search command. This allows many DS18B20s to be connected in parallel doing simultaneous temperature measurements. If somewhere the temperature exceeds the limits, the alarming device(s) can be identified and read immediately without having to read non-alarming devices.

Lampiran 10. Datasheet Sensor pH

PH meter(SKU: SEN0161)

From Robot Wiki

Contents



Analog pH Meter Kit

Introduction

Need to measure water quality and other parameters but haven't got any low cost pH meter? Find it difficult to use with Arduino?
Here comes an analog pH meter, specially designed for Arduino controllers and has built-in simple, convenient and practical connection and features. It has an LED which works as the Power Indicator, a BNC connector and PH2.0 sensor interface. To use it, just connect the pH sensor with BNC connector, and plug the PH2.0 interface into the analog input port of any Arduino controller. If pre-programmed, you will get the pH value easily. Comes in compact plastic box with foams for better mobile storage.
Attention:In order to ensure the accuracy of the pH probe, you need to use the standard solution to calibrate it regularly.Generally, the period is about half a year. If you meaure the dirty aqueous solution, you need to increase the frequency of calibration.

Applications

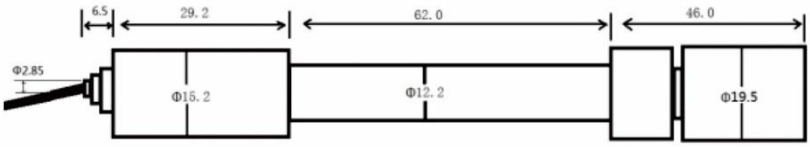
- Water quality testing
- Aquaculture

Specification

- Module Power : 5.00V
- Module Size : 43mm×32mm
- Measuring Range:0-14PH
- Measuring Temperature :0-60 ℃

- Accuracy : ± 0.1pH (25 ℃)
- Response Time : ≤ 1min
- pH Sensor with BNC Connector
- PH2.0 Interface (3 foot patch)
- Gain Adjustment Potentiometer
- Power Indicator LED
- Cable Length from sensor to BNC connector:660mm

pH Electrode Size



pH Electrode Characteristics

The output of pH electrode is Millivolts,and the pH value of the relationship is shown as follows (25 ℃):

VOLTAGE （mV）	pH value	VOLTAGE （mV）	pH value
414.12	0.00	-414.12	14.00
354.96	1.00	-354.96	13.00
295.80	2.00	-295.80	12.00
236.64	3.00	-236.64	11.00
177.48	4.00	-177.48	10.00
118.32	5.00	-118.32	9.00
59.16	6.00	-59.16	8.00
0.00	7.00	0.00	7.00

Use the pH Meter

Connecting Diagram

Lampiran 11. Datasheet Sensor Salinitas

www.depoinovasi.com

DATASHEET

SENSOR KONDUKTIVITAS / TDS / KADAR GARAM

Tipe Aplikasi :

- Sensor konduktivitas (conductivity sensor)
- Sensor TDS (total dissolve solid)
- Sensor kadar garam (salinity sensor)

Spesifikasi :

- Bekerja pada tegangan DC 5 Volt
- Support arduino dan mikrokontroller lainnya
- Koefisien linearitas data konduktivitas sebesar 0.9639
- Koefisien linearitas data TDS sebesar 0.983
- Memiliki sensitivitas pada bahan yang bersifat konduktif
- Kedalaman cairan pada saat pengukuran sebesar 5.5 cm dari ujung sensor
- Rumus persamaan umum konversi data konduktivitas **y = 0.2142x + 494.93, dimana : x = nilai ADC, dan y=konduktivitas**
- Rumus persamaan umum konversi data TDS **y= 0.3417x + 281.08, dimana : x = nilai ADC, dan y=TDS**

DESKRIPSI

SENSOR KONDUKTIVITAS / TDS / KADAR GARAM

Pin	Deskripsi
5	5V arduino
Gnd	GND arduino
Output	Output ke pin A0 arduino

Tabel 1. Pin Sensor

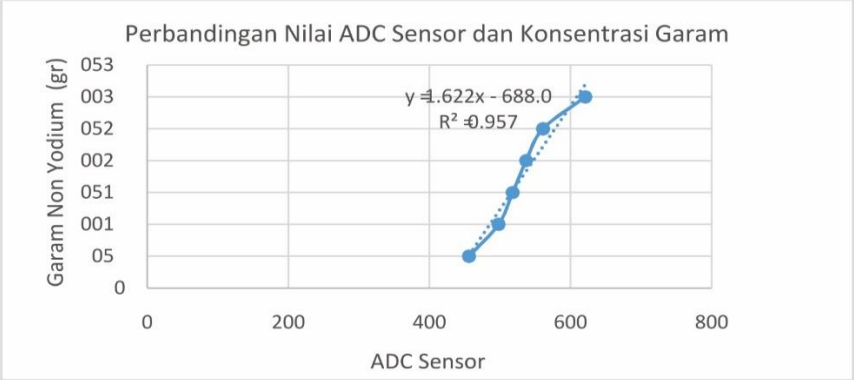
Sensor Konduktivitas / TDS / Kadar Garam memiliki desain yang kompak. Probe sensornya berbahan stik stainless yang berfungsi sebagai penerima data dari bahan yang diuji. Sensor ini dapat langsung disambungkan dengan pin analog arduino maupun pin analog mikrokontroller lainnya, tanpa harus memakai modul penguat tambahan.

KARAKTERISTIK

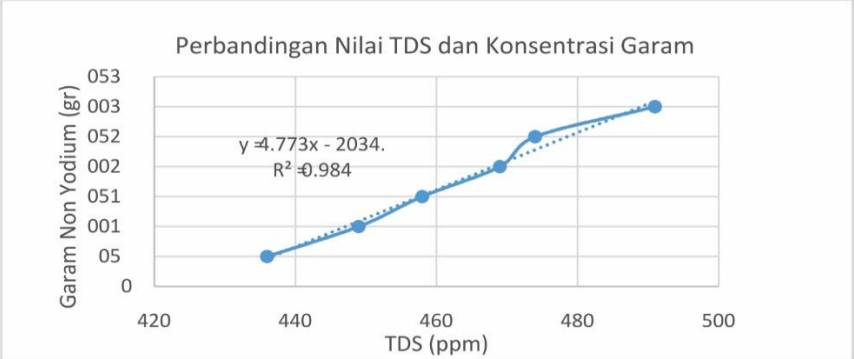
SENSOR KONDUKTIVITAS / TDS / KADAR GARAM

Parameter	Simbol	Min	Max	Units
Tegangan masukan	Vcc		5.0	V
Tegangan operasional	Vcc	3.0	4.7	V
Tegangan keluaran	ADC	0	1023	ADC
Respon waktu	t	0.1	0.3	s
Sensitivitas	Vcc	0.1	0.5	V

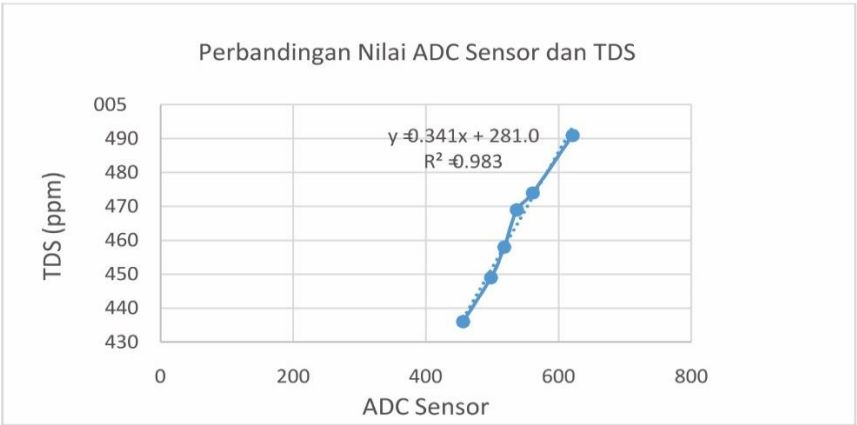
Tabel 2. Karakteristik Sensor



Grafik 6. Karakteristik nilai ADC sensor terhadap konsentrasi garam (uji TDS)



Grafik 7. Karakteristik TDS sensor kalibrator TDS meter terhadap konsentrasi garam (uji TDS)



Grafik 8. Karakteristik nilai ADC sensor terhadap nilai TDS sensor kalibrator TDS meter (uji TDS)