DAFTAR LAMPIRAN

Lampiran- 1 Program alat

1. rangkaian pengirim (TX)

```
/*
Nano  UART-LoRa
D3    -> TXD
D4    -> RXD
NC    -> AUX
5V    -> VCC
GND   -> GND
*/
#include <SoftwareSerial.h>
SoftwareSerial loraSerial(3,4); //3->TX_Lora
#include "GravityTDS.h"

#define pHSensor A1  //pH meter Analog output
#define tbdSensor A3
#define tdsSensor A5
#define Offset 0.00  //deviation compensate
#define LED 13
#define samplingInterval 20
#define printInterval 2000 //800
#define ArrayLenth  40    //times of collection
#define VREF 5.0      // analog reference voltage(Volt) of the ADC
#define SCOUNT  30           // sum of sample point
int analogBuffer[SCOUNT];    // store the analog value in the
array, read from ADC
int analogBufferTemp[SCOUNT];
int analogBufferIndex = 0,copyIndex = 0;
int pHArray[ArrayLenth];   //Store the average value of the sensor
feedback
int pHArrayIndex=0;

float pHValue, volt, ntu, IP;

GravityTDS gravityTds;
float averageVoltage = 0,tdsValue = 0,temperature = 25;
String sPH, sTDS, sNTU, sIP;

String CNum="C1", CRX;
String dataTX= ""; //data kirim
String dataRX=""; //data terima
int counter = 0; //Lora initialization counter
unsigned int readingID = 0; //packet counter

void setup() {
  Serial.begin(9600);
  startLoRA();
  pinMode(tdsSensor,INPUT);
}
```

```
void loop() {
  Ph();
  Turbidity();
  TDS();
  indeksP();
  terimaData();
}

void Ph() {
  static unsigned long samplingTime = millis();
  static unsigned long printTime = millis();
  static float pHValue,voltage;
  if(millis()-samplingTime > samplingInterval){
      pHArray[pHArrayIndex++]=analogRead(pHSensor);
      if(pHArrayIndex==ArrayLenth)pHArrayIndex=0;
      voltage = avergearray(pHArray, ArrayLenth)*5.0/1024;
      pHValue = 3.5*voltage+Offset;
      samplingTime=millis();
  }
  if(millis() - printTime > printInterval){
     sPH = String(pHValue,1);
     sNTU = String(ntu,0);
     sTDS = String(tdsValue,1);
     sIP = String(IP,2);

     Serial.println(" pH value: "+ sPH);
     Serial.println(" ntu : "+ sNTU);
     Serial.println(" tds : "+ sTDS+ " ppm");
     Serial.println(" IP : "+ sIP);
     Serial.println();
     digitalWrite(LED,digitalRead(LED)^1); //ubah status LED
     printTime=millis();
  }
}

double avergearray(int* arr, int number){
  int i,max,min;
  double avg;
  long amount=0;
  if(number<=0){
    Serial.println("Error number for the array to avraging!/n");
    return 0;
  }
  if(number<5){   //less than 5, calculated directly statistics
    for(i=0;i<number;i++) { amount+=arr[i]; }
    avg = amount/number;
    return avg;
  }else{
    if(arr[0]<arr[1]){ min = arr[0];max=arr[1]; }
    else{ min=arr[1];max=arr[0]; }
    for(i=2;i<number;i++){
      if(arr[i]<min){
        amount+=min;  //arr<min
        min=arr[i];
      }else {
        if(arr[i]>max){
```

```
          amount+=max;    //arr>max
          max=arr[i];
        }else{
          amount+=arr[i]; //min<=arr<=max
        }
      }//if
    }//for
    avg = (double)amount/(number-2);
  }//if
  return avg;
}

void Turbidity() {
   volt = 0;
    for(int i=0; i<800; i++){
        volt += ((float)analogRead(tbdSensor)/1023)*5;
    }
    volt = volt/800;
    volt = round_to_dp(volt,2);
    if(volt < 2.5) ntu = 3000;
    else ntu = -1120.4*square(volt)+5742.3*volt-4353.8;
   // Serial.println(" ntu : "+String(ntu,2));
}

float round_to_dp( float in_value, int decimal_place ){
  float multiplier = powf( 10.0f, decimal_place );
  in_value = roundf( in_value * multiplier ) / multiplier;
  return in_value;
}

void TDS() {
    static unsigned long analogSampleTimepoint = millis();
   if(millis()-analogSampleTimepoint > 40U)     //every 40
milliseconds,read the analog value from the ADC
   {
     analogSampleTimepoint = millis();
     analogBuffer[analogBufferIndex] = analogRead(tdsSensor);
//read the analog value and store into the buffer
     analogBufferIndex++;
     if(analogBufferIndex == SCOUNT)
         analogBufferIndex = 0;
   }
   static unsigned long printTimepoint = millis();
   if(millis()-printTimepoint > 800U)
   {
      printTimepoint = millis();
      for(copyIndex=0;copyIndex<SCOUNT;copyIndex++)
        analogBufferTemp[copyIndex]= analogBuffer[copyIndex];
      averageVoltage = getMedianNum(analogBufferTemp,SCOUNT) *
(float)VREF / 1024.0; // read the analog value more stable by the
median filtering algorithm, and convert to voltage value
float compensationCoefficient=1.0+0.02*(temperature-25.0);
//temperature compensation formula: fFinalResult(25^C) =
fFinalResult(current)/(1.0+0.02*(fTP-25.0));
```

```
floatcompensationVolatge=averageVoltage/compensationCoefficient;
//temperature compensation

tdsValue=(133.42*compensationVolatge*compensationVolatge*compensat
ionVolatge - 255.86*compensationVolatge*compensationVolatge +
857.39*compensationVolatge)*0.5; //convert voltage value to tds
value
     //Serial.print("voltage:");
     //Serial.print(averageVoltage,2);
     //Serial.print("V   ");
     //Serial.println(" TDS Value  : "+String(tdsValue,3)+ "
ppm");
    delay(1000);
   }
}
int getMedianNum(int bArray[], int iFilterLen)
{
      int bTab[iFilterLen];
      for (byte i = 0; i<iFilterLen; i++)
      bTab[i] = bArray[i];
      int i, j, bTemp;
      for (j = 0; j < iFilterLen - 1; j++)
      {
      for (i = 0; i < iFilterLen - j - 1; i++)
          {
        if (bTab[i] > bTab[i + 1])
            {
        bTemp = bTab[i];
            bTab[i] = bTab[i + 1];
        bTab[i + 1] = bTemp;
         }
      }
      }
      if ((iFilterLen & 1) > 0)
    bTemp = bTab[(iFilterLen - 1) / 2];
      else
    bTemp = (bTab[iFilterLen / 2] + bTab[iFilterLen / 2 - 1]) / 2;
      return bTemp;
}

void indeksP() {
  float Ph = pHValue/7.5;
  float tds = tdsValue/1000;
  float ntuValue = ntu/3000;

  float jumlah = Ph + tds + ntuValue;
  float rerata = jumlah/3;
  float maks1 = max(Ph, tds);
  float maks2 = max(maks1, ntuValue);

  IP = sqrt((((sq(jumlah)) * maks2) + ((sq(jumlah)) * rerata))/2);
  //Serial.println(" IP : "+String(IP,4));
}

void terimaData() { //Receive from Server
```

```
  if(loraSerial.available()>0) {
     dataRX = loraSerial.readString(); // Read packet
     Serial.println("Terima dulu: "+dataRX);
    //format terima = C1~
    int pos1 = dataRX.indexOf('~');
    CRX = dataRX.substring(0, pos1);
    dataRX ="";
  }
  if(CNum==CRX) kirimData();
}

void kirimData() {
  dataTX = CNum+"~"+sPH+"#"+sNTU+"$"+sTDS+"%"+sIP+"*";
  loraSerial.print(dataTX); //Kirimkan ke receiver
  Serial.println("Baru Kirim : "+dataTX);
  Serial.println();
}

//Initialize LoRa module
void startLoRA(){
  loraSerial.begin(9600);
  while (!loraSerial && counter < 10) {
    Serial.print(".");
    counter++; delay(100);
  }
  if(counter==10) {
     readingID++;
     Serial.println("Starting LoRa failed!");
  }
  Serial.println("LoRa Initialization OK!");
  delay(3000);
}
```

2. Rangkaian penerima(RX)

```
#include <SoftwareSerial.h>
SoftwareSerial loraSerial(3,4); // RX, TX

#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27,20,4);

#include <SPI.h>
#include <SD.h>
#include <I2C_RTC.h>
static PCF8563 RTC;

const int chipSelect = 10; // Pin 10 on Arduino Uno
const int buttonRead = A0;
const int redLEDpin = 9;
String hri, tgl, bln, jam, mnt, dtk, thn;
bool statusRead=false;
String filename = "datalog.txt"; //notepad

String sRN, sPH, sTDS, sNTU, sIP;
String CNum="C1", CRX;
String dataTX= ""; //data kirim
```

```
String dataRX=""; //data terima
int counter = 0; //Lora initialization counter
int recordNum=0;
int periodaData=3000; //mdet.
unsigned int readingID = 0; //packet counter
unsigned long lastT1=0, previousT1=0, limitT1=5000;
unsigned long lastT2=0, previousT2=0, limitT2=60000;

File myFile;
void(* resetFunc) (void) = 0;

void setup() {
  Serial.begin(9600);
  /*
   RTC.setDay(25);
   RTC.setMonth(02);
   RTC.setYear(2024);
   RTC.setWeek(1);  // Always Set weekday after setting Date

   RTC.setHours(14);
   RTC.setMinutes(54);
   RTC.setSeconds(10);
   */
  lcd.init(); lcd.setBacklight(100);
  startLoRA();
  pinMode(chipSelect, OUTPUT);
  pinMode(buttonRead, INPUT_PULLUP);
  pinMode(redLEDpin,OUTPUT);
  delay(100);

  lcd.setCursor(0,0); lcd.print(F("Mencek Kartu SD "));
  delay(2000);
  if (SD.begin(chipSelect)) {
      lcd.setCursor(0,0); lcd.print(F(" Kartu SD SIAP  "));
  }
  else if (!SD.begin(chipSelect)) {
     lcd.setCursor(0,0); lcd.print(F("Kartu SD Error !"));
     while(1) {
       digitalWrite(redLEDpin, 1);delay(100);
       digitalWrite(redLEDpin, 0);delay(100);
     }
     return;
  }
  delay(2000);

  bacaRTC();
  lcd.setCursor(0,0);
  lcd.print("Mulai Mencatat : ");
  lcd.setCursor(0,1);
  lcd.print(hri+','+tgl+'-'+bln+'-'+thn);
  lcd.setCursor(0,2);
  lcd.print("Jam : "+jam+':'+mnt+':'+dtk);
  delay(5000);

  myFile = SD.open("datalog.txt", FILE_WRITE); // Create/Open file
  if(!myFile) { //if the file isn't open, pop up an error
```

```
      lcd.setCursor(0,0); lcd.print(F("  File Error !  "));
      return;
  }


  if(myFile.size() == 0) { //jika file masih kosong
     myFile.println("No.   pH    ntu    TDS   IP     Waktu ");
     myFile.close();
  }
  else {
     myFile.println("");
     myFile.println("No.   pH    ntu    TDS   IP     Waktu ");
     myFile.close();
  }

  lcd.clear();
  lcd.setCursor(0,0);
  lcd.print(tgl+'-'+bln+'-'+thn+"    "+jam+':'+mnt+':'+dtk);
  displayData();
}

void loop() {
 bacaRTC();
 if(!digitalRead(buttonRead)&&!statusRead) { statusRead=1;
readText(); }
 if(digitalRead(buttonRead)&& statusRead) {
    statusRead=0; lcd.clear();
    displayData();
 }
 if(digitalRead(buttonRead)&& !statusRead) {
     lcd.setCursor(0,0);
     lcd.print(tgl+'-'+bln+'-'+thn+"    "+jam+':'+mnt+':'+dtk);

     kirimData();
     terimaData();
     displayData();

     lastT2=millis();
     if(lastT2-previousT2>limitT2 && sPH!="--- ") {
       simpanData();
       previousT2=lastT2;
     }
   }
}

void displayData() {
   lcd.setCursor(0,1);  lcd.print(F("pH : ")); lcd.print(sPH);
   lcd.setCursor(10,1); lcd.print(F("TDS: ")); lcd.print(sTDS);
   lcd.setCursor(5,2);  lcd.print(F("NTU: ")); lcd.print(sNTU);
   lcd.setCursor(0,3);  lcd.print(F("IP : ")); lcd.print(sIP);
}

void simpanData() {
   myFile=SD.open("datalog.txt", FILE_WRITE);

   if(myFile) {
```

```
      recordNum++;
      if(recordNum<10) sRN="00"+String(recordNum);
      else if(recordNum<100) sRN="0"+String(recordNum);
      else if(recordNum<1000) sRN=String(recordNum);

      lcd.setCursor(10,3); lcd.print("JD : "); lcd.print(sRN);
      myFile.println(sRN+"   "+sPH+"   "+sNTU+"   "+sTDS+"
"+sIP+"   "+
                  tgl+'-'+bln+'-'+thn+"   "+
                  jam+':'+mnt+':'+dtk);
      myFile.close();
      Serial.println(sRN+" Data tersimpan");
   }
   else { Serial.println(F("gagal buka file"));
          myFile.close();
   }
}

void readText() {
  lcd.clear();
  lcd.setCursor(0,0); lcd.print(F("  Membaca File  "));
   // open the file for reading:
  myFile = SD.open("datalog.txt");
  if (myFile) {
     // read from the file until there's nothing else in it:
     while (myFile.available()) { Serial.write(myFile.read()); }
     myFile.close(); // close the file:
    }
  else {
     Serial.println(F("gagal buka file ! "));
  }
  lcd.setCursor(0,0);
  lcd.print(F("Hapus(Y/y), Tdk(T/t)"));
  Serial.println(F("Hapus(Y/y), Tdk(T/t)"));

  while(!Serial.available()) { }
  if (Serial.available()>0) {
      char k=Serial.read();
      if(k=='Y' || k=='y') {
        SD.remove("datalog.txt"); //hapus file
        Serial.println(F(" File Sudah dihapus "));
        lcd.setCursor(0,0); lcd.print(F(" File Sudah dihapus "));
        delay(2000);
        resetFunc();
        lcd.setCursor(0,1); lcd.print(F("Tekan Tombol Hijau !"));
        Serial.println(F("Tekan Tombol Hijau !"));
      }
      else {
        lcd.setCursor(0,0); lcd.print(F("Tekan Tombol Hijau !"));
        Serial.println(F("Tekan Tombol Hijau !"));
      }
  }
}

//Initialize LoRa module
void startLoRA() {
```

```
  loraSerial.begin(9600);
  while (!loraSerial && counter < 10) {
    loraSerial.print("."); counter++; delay(100);
  }
  if (counter >= 10) {
    Serial.println(F("Inisialisasi LoRa gagal!"));
  }
  Serial.println(F("Inisialisasi LoRa OK!"));
  delay(1000);
}

void kirimData() {
  dataTX = "C1~";
  loraSerial.print(dataTX); //Kirimkan LoRa packet ke receiver
  Serial.print(F("Kirim Dulu : ")); Serial.println(dataTX);
  delay(periodaData);
}

void terimaData() {
  if(loraSerial.available()>0) {
    dataRX = loraSerial.readStringUntil('*'); // Read packet
    //dataRX = CNum+'~'+sPH+'#'+sNTU+'$'+sTDS+'%'+sIP+'*';
    Serial.print(F("Baru Terima : ")); Serial.println(dataRX);
    previousT1=millis();
    int pos1 = dataRX.indexOf('~');
    int pos2 = dataRX.indexOf('#');
    int pos3 = dataRX.indexOf('$');
    int pos4 = dataRX.indexOf('%');
    int pos5 = dataRX.indexOf('*');
    CNum = dataRX.substring(0, pos1);
    if(CNum == "C1") {
       sPH = dataRX.substring(pos1+1, pos2);
       sNTU = dataRX.substring(pos2+1, pos3);
       sTDS = dataRX.substring(pos3+1, pos4);
       sIP = dataRX.substring(pos4+1, pos5);
    }
   loraSerial.readStringUntil(' '); //kosongkan buffer
  }
  else {
    lastT1=millis();
    if(lastT1-previousT1>limitT1) {
       sPH = "--- ";
       sNTU = "--- ";
       sTDS = "--- ";
       sIP = "--- ";
       Serial.println(F("Tidak Menerima DATA! \n"));
    }
  }
}

void bacaRTC() {
    switch (RTC.getWeek()) {
      case 1: hri = "Ahad";  break;
      case 2: hri = "Senin"; break;
      case 3: hri = "Selasa"; break;
      case 4: hri = "Rabu";  break;
```

```
      case 5: hri = "Kamis"; break;
      case 6: hri = "Jum'at"; break;
      case 7: hri = "Sabtu"; break;
    }
    tgl=String(RTC.getDay());
    if(RTC.getDay()<10) tgl="0"+String(RTC.getDay());
    bln= String(RTC.getMonth());
    if(RTC.getMonth()<10) bln="0"+String(RTC.getMonth());
    thn= String(RTC.getYear()-2000);

    jam= String(RTC.getHours());
    if(RTC.getHours()<10) jam= "0"+String(RTC.getHours());
    mnt= String(RTC.getMinutes());
    if(RTC.getMinutes()<10) mnt= "0"+String(RTC.getMinutes());
    dtk= String(RTC.getSeconds());
    if(RTC.getSeconds()<10) dtk= "0"+String(RTC.getSeconds());
}
```

Lampiran- 2 Alat penelitian















Lampiran-3 Datasheet

1. Sensor pH

PH meter(SKU: SEN0161)

From Robot Wiki



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

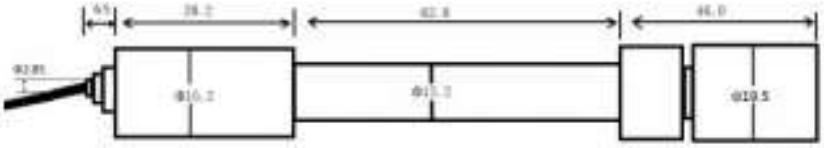
- Water quality testing
- Aquaculture

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

2. Sensor *Turbidity*

DFROBOT

Turbidity sensor SKU: SEN0189



Contents

- 1 Introduction
- 2 Specification
- 3 Connection Diagram
- 4 Examples

Introduction

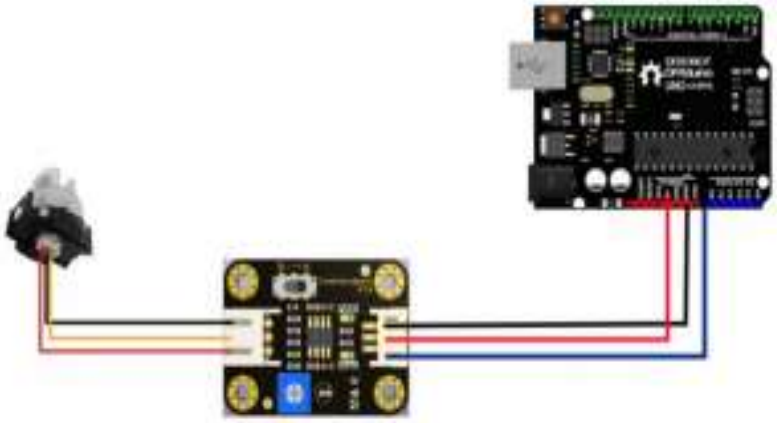
The turbidity sensor detects water quality by measuring the levels of turbidity. It uses light to detect suspended particles in water by measuring the light transmittance and scattering rate, which changes with the amount of total suspended solids (TSS) in water. As the TTS increases, the liquid turbidity level increases.
Turbidity sensors are used to measure water quality in rivers and streams, wastewater and effluent measurements, control instrumentation for settling ponds, sediment transport research and laboratory measurements.
This sensor provides analog and digital signal output modes. The threshold is adjustable when in digital signal mode. You can select the mode according to your MCU.

Note: The top of probe is not waterproof.

Specification

- Operating Voltage: 5V DC
- Operating Current: 40mA (MAX)
- Response Time : <500ms
- Insulation Resistance: 100M (Min)
- Output Method:
  Analog output: 0-4.5V
  Digital Output: High/Low level signal (you can adjust the threshold value by adjusting the potentiometer)
- Operating Temperature: 5℃~90℃
- Storage Temperature: -10℃~90℃
- Weight: 30g
- Adapter Dimensions: 38mm*28mm*10mm/1.5inches * 1.1inches*0.4inches

Connection Diagram



Interface Description:

---

1. "D/A" Output Signal Switch
1. "A": Analog Signal Output, the output value will decrease when in liquids with a high turbidity
2. "D": Digital Signal Output, high and low levels, which can be adjusted by the threshold potentiometer
2. Threshold Potentiometer: you can change the trigger condition by adjusting the threshold potentiometer in digital signal mode.

Examples

Here are two examples:
Example 1 uses Analog output mode
Example 2 uses Digital output mode

Example 1

```
void setup() {
  Serial.begin(9600); //Baud rate: 9600
}
void loop() {
  int sensorValue = analogRead(A0);// read the input on analog pin 0:
  float voltage = sensorValue * (5.0 / 1024.0); // Convert the analog read
ing (which goes from 0 - 1023) to a voltage (0 - 5V):
  Serial.println(voltage); // print out the value you read:
  delay(500);
}
```

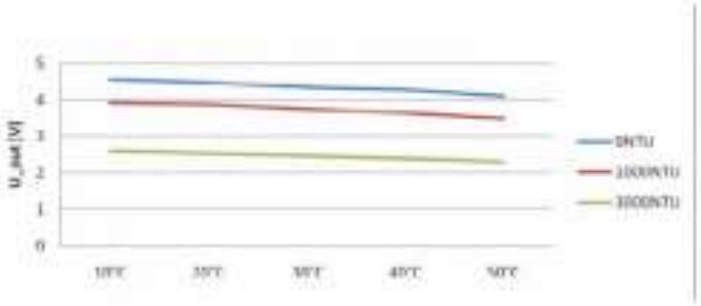
Example 2

```
int ledPin = 13;                // Connect an LED on pin 13, or use the on
board one
int sensor_in = 2;                 // Connect turbidity sensor to Digital
Pin 2
```

```
void setup(){
  pinMode(ledPin, OUTPUT);      // Set ledPin to output mode
  pinMode(sensor_in, INPUT);       //Set the turbidity sensor pin to input
mode
}

void loop(){
   if(digitalRead(sensor_in)==LOW){       //read sensor signal
        digitalWrite(ledPin, HIGH);   // if sensor is LOW, then turn on
     }else{
        digitalWrite(ledPin, LOW);    // if sensor is HIGH, then turn off
the led
     }
}
```

This is a reference chart for the mapping from the output voltage to the NTU according to different temperature. e.g. If you leave the sensor in the pure water, that is NTU < 0.5, it should output "4.1±0.3V" when temperature is 10~50℃.



characteristic curve "Voltage ----Temperature

Note: In the diagram, the unit measuring turbidity is shown as NTU, also it is known as JTU (Jackson Turbidity Unit). 1JTU = 1NTU = 1 mg/L. Refer to Turbidity wikipedia

Powered By DFRobot © 2008-2017

3. Sensor TDS

DFROBOT



Gravity: Analog TDS Sensor / Meter For Arduino

SKU: SEN0244

TDS (Total Dissolved Solids) indicates that how many milligrams of soluble solids dissolved in one liter of water. In general, the higher the TDS value, the more soluble solids dissolved in water, and the less clean the water is. Therefore, the TDS value can be used as one of the references for reflecting the cleanliness of water.

TDS pen is a widely used equipment to measure TDS value. The price is affordable, and it is easy to use, but it is not able to transmit data to the control system for online monitoring to do some water quality analysis. The professional instrument has high accuracy and can send data to the control system, but the price is expensive for the ordinary people. To this end, we have launched an analog TDS sensor kit which is compatible with Arduino, plug and play, easy to use. Matching with Arduino controller, you can build a TDS detector easily to measure the TDS value of liquid.

This product supports 3.3 ~ 5.5V wide voltage input, and 0 ~ 2.3V analog voltage output, which makes it compatible with 5V or 3.3V control system or board. The excitation source is AC signal, which can effectively prevent the probe from polarization and prolong the life of the probe, meanwhile, increase the stability of the output signal. The TDS probe is waterproof, it can be immersed in water for long time measurement.

This product can be used in water quality application, such as domestic water, hydroponics. With this product, you can easily DIY a TDS detector to reflect the cleanliness of water to protect your health.

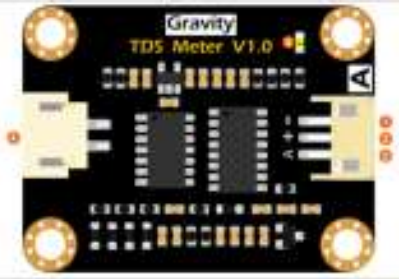
Attention:

1.The probe can not be used in water above 55 degrees centigrade.
2.The probe can not be left too close to the edge of the container, otherwise it will affect the reading.
3.The head and the cable of the probe are waterproof, but the connector and the signal transmitter board are not waterproof. Please be careful.

Specification

- **Signal Transmitter Board**
  Input Voltage: 3.3 ~ 5.5V
  Output Voltage: 0 ~ 2.3V
  Working Current: 3 ~ 6mA
  TDS Measurement Range: 0 ~ 1000ppm
  TDS Measurement Accuracy: ± 10% F.S. (25 ℃)
  Module Size: 42 * 32mm
  Module Interface: PH2.0-3P
  Electrode Interface: XH2.54-2P
- **TDS probe**
  Number of Needle: 2
  Total Length: 83cm
  Connection Interface: XH2.54-2P
  Colour: Black
  Other: Waterproof Probe

Board Overview



Analog TDS Sensor / Meter For Arduino

Num	Label	Description
1	-	Power GND(0V)
2	+	Power VCC(3.3 ~ 5.5V)
3	A	Analog Signal Output(0 ~ 2.3V)
4	TDS	TDS Probe Connector
5	LED	Power Indicator

Tutorial

This tutorial will show you how to measure the TDS value of the water. Please read this tutorial carefully, and pay attention to the steps and details.

The probe can not to be used in water above 55 degrees centigrade.
The probe can not be too close to the edge of the container, otherwise it will affect the reading.
The head and the cable of the probe are waterproof, but the connector and the signal transmitter board are not waterproof.Please pay attention to use.

Requirements

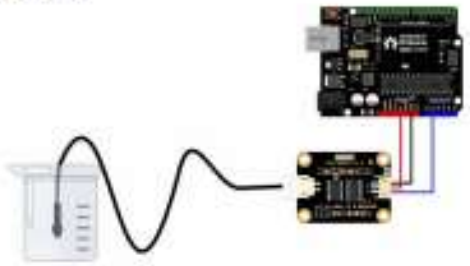
- **Hardware**
  DFRduino UNO R3 (or similar) x 1
  Analog TDS Sensor / Meter Module x 1
  TDS Probe x1
  Jumper Wires x3
  tested liquid x1

- **Software**

Arduino IDE (Version requirements: V1.0.x or V1.8.x), Click to Download Arduino IDE from Arduino®

https://www.arduino.cc/en/Main/Software%7C

Connection Diagram



4. Lora

1 Overview

1.1 Product introduction



E32-433T20D is a wireless serial port module (TTL level) based on Semtech's SX1278 radio frequency chip, transparent transmission, working in the 410-441MHz frequency band (default 433MHz), using LoRa spread spectrum technology.

SX1278 supports LoRa spread spectrum technology. LoRa direct sequence spread spectrum technology has the advantages of longer communication distance, strong anti-interference ability, and strong confidentiality. SX1278 is a milestone in the field of low-speed communication, and is favored by people in the industry. The default air rate of this series is 2.4kbps , the transmit power is 20dBm, and the industrial -grade crystal oscillator is used to ensure its stability and consistency, and the accuracy is less than 10ppm, which is generally used in the industry. At present, it has been in stable mass production, and has been widely used in the three-meter industry, Internet of Things transformation, smart furniture and other fields. The module has data encryption and compression functions. The data transmitted by the module in the air is random, and the data interception is meaningless through strict encryption and decryption algorithms. The data compression function has the possibility to reduce the transmission time, reduce the probability of being interfered, and improve reliability and transmission efficiency.

E32-433T20D strictly abides by FCC, CE, CCC and other domestic and foreign design specifications, meets various RF-related certifications, and meets export requirements.

1.2 Features

- Support advanced LoRa modulation , with the advantage of long-distance anti-interference ;
- Under ideal conditions, the communication distance can reach 3km , and the transmission distance is better than that of traditional G FSK ;
- Support fixed-point transmission, broadcast transmission, channel monitoring;
- Support air wake-up (ultra-low power consumption), suitable for battery applications;
- Support FEC forward error correction to improve communication stability;
- maximum transmit power is 100mW, and the software is multi-level adjustable;
- Support global license-free ISM 433MHz frequency band;
- Support data transmission rate of 0.3k ~ 19.2kbps;
- Support 2.3 ~ 5.5 V power supply, more than 3.3V power supply can ensure the best performance;
- Industrial-grade standard design, support long-term use at -40 ~ + 85 ℃ ;
- SMA-K interface for easy connection of coaxial cable or external antenna .

1.3 Application scenarios

- Home security alarm and remote keyless entry;
- Smart home and industrial sensors, etc.;
- Wireless alarm security system;
- Building automation solutions;
- Wireless industrial grade remote control;
- Intelligent intelligent agriculture, oilfield solutions;

Copyright © 2012–2022 , Chengdu Ebyte Electronic Technology Co., Ltd.

- healthcare products;
- Advanced Meter Reading Architecture (AMI);
- Automotive industry applications.

2 Specifications

2.1 Limit parameters

The main parameters	performance		Remark
	minimum	maximum value	
Supply voltage (V)	0	5.5	More than 5.2V will permanently burn the module
Blocking power (dBm)	-	10	The probability of burning at close range is small
Working temperature (℃)	-40	+ 85	Industrial grade

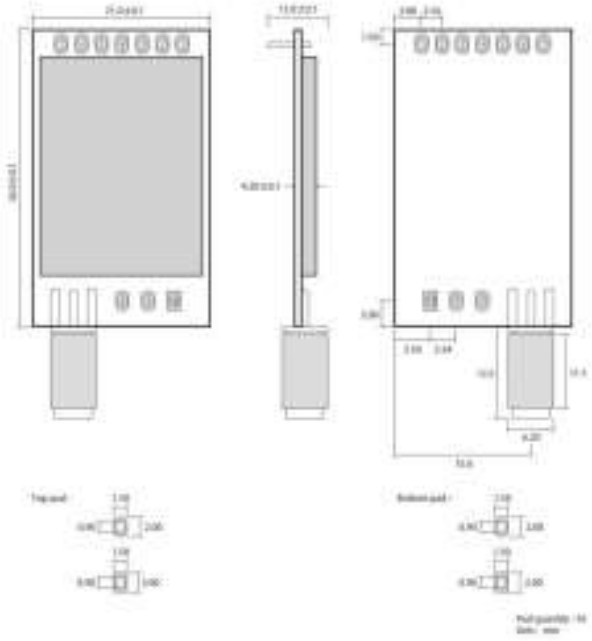
2.2 Working parameters

The main parameters		performance			Remark
		minimum	Typical value	maximum value	
Working voltage (V)		2.3	5.0	5.5	≥ 3.3 V guaranteed output power
Communication level (V)			3.3		Risk of burnout with 5V TTL
Working temperature (℃)		-40	-	+ 85	Industrial grade design
Working frequency band (MHz)		410	-	441	Support ISM frequency band, factory default 433MHz
Power consumption	Emission current (mA)		110		Instantaneous power consumption
	Receive current (mA)		14		
	Sleep current (μA)		4		software shutdown
Maximum transmit power (dBm)		19.0	-	20.0	
Receive Sensitivity (dBm)		-144	-146	-147	The air rate is 2.4kbps
Air rate (bps)		0.3k	2.4k	19.2k	User self-configured

Copyright © 2012–2022, Chengdu Ebyte Electronic Technology Co., Ltd.

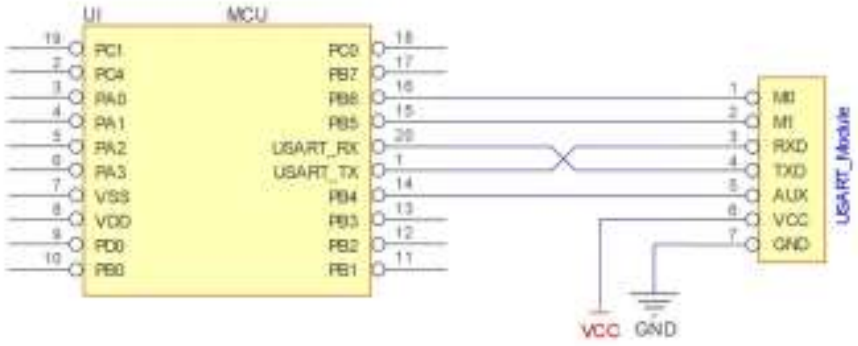
The main parameters	describe	Remark
reference distance	3000m	Clear and open, antenna gain 5dBi, antenna height 2.5m, air rate 2.4kbps
Subcontracting	5 8B tye	The maximum capacity of a single package, it will be automatically sub-packaged when it exceeds.
Cache capacity	5 12B tye	
Modulation	LoRa ™	
Communication Interface	UART serial port	TTL level
Packaging method	- line	
interface	1* 7 * 2.54mm	
Dimensions	2 1 *36mm	
Antenna interface	SMA-K	Characteristic impedance about 50 Ω

3 Mechanical Dimensions and Pin Definition



pin number	pin name	Pin Orientation	pin usage
1	M0	Input (very weak pull-up)	Cooperate with M1 to determine the 4 working modes of the module. (Cannot be left floating, can be grounded if not used)
2	M1	Input (very weak pull-up)	Cooperate with M0 to determine 4 working modes of the module. (Cannot be left floating, can be grounded if not used)
3	RXD	enter	TTL serial port input, connected to the external TXD output pin; Can be configured as open-drain or pull-up input, see parameter settings for details.
4	TXD	output	TTL serial output, connected to external RXD input pin; Can be configured as open-drain or push-pull output, see parameter settings for details.
5	AUX	output	Used to indicate the working status of the module; (can be left floating) The user wakes up the external MCU, and outputs a low level during the power-on self-test initialization; It can be configured as open-drain output or push-pull output, see parameter settings for details.
6	VCC	enter	Positive reference for module power supply, voltage range: 2.3 to 5.5 V DC
7	GND	enter	Module ground
8	fixing hole		fixing hole
9	fixing hole		fixing hole
10	fixing hole		fixing hole

4 Recommended Wiring Diagram



Copyright © 2012–2022 , Chengdu Ebyte Electronic Technology Co., Ltd.

5. LCD I2C 20x4

Handson Technology

User Guide

I2C Serial Interface 20x4 LCD Module

This is I2C interface 20x4 LCD display module, a new high-quality 4 line 20 character LCD module with on-board contrast control adjustment, backlight and I2C communication interface. For Arduino beginners, no more cumbersome and complex LCD driver circuit connection. The real significance advantages of this I2C Serial LCD module will simplify the circuit connection, save some I/O pins on Arduino board, simplified firmware development with widely available Arduino library.





SKU: DSP-1165

Brief Data:

- Compatible with Arduino Board or other controller board with I2C bus.
- Display Type: Black on yellow green backlight.
- I2C Address:0x38-0x3F (0x3F default)
- Supply voltage: 5V
- Interface: I2C to 4bits LCD data and control lines.
- Contrast Adjustment : built-in Potentiometer.
- Backlight Control: Firmware or jumper wire.
- Board Size: 98x60 mm.

Setting Up:

Hitachi's HD44780 based character LCD are very cheap and widely available, and is an essential part for any project that displays information. Using the LCD piggy-back board, desired data can be displayed on the LCD through the I2C bus. In principle, such backpacks are built around PCF8574 (from NXP) which is a general purpose bidirectional 8 bit I/O port expander that uses the I2C protocol. The PCF8574 is a silicon CMOS circuit provides general purpose remote I/O expansion (an 8-bit quasi-bidirectional) for most microcontroller families via the two-line bidirectional bus (I2C-bus). Note that most piggy-back modules are centered around PCF8574T (SO16 package of PCF8574 in DIP16 package) with a default slave address of 0x27. If your piggy-back board holds a PCF8574AT chip, then the default slave address will change to 0x3F. In short, if the piggy-back board is based on PCF8574T and the address connections (A0-A1-A2) are not bridged with solder it will have the slave address 0x27.



Address selection pads in the I2C-to-LCD piggy-back board.

Table 5. PCF8574A address map

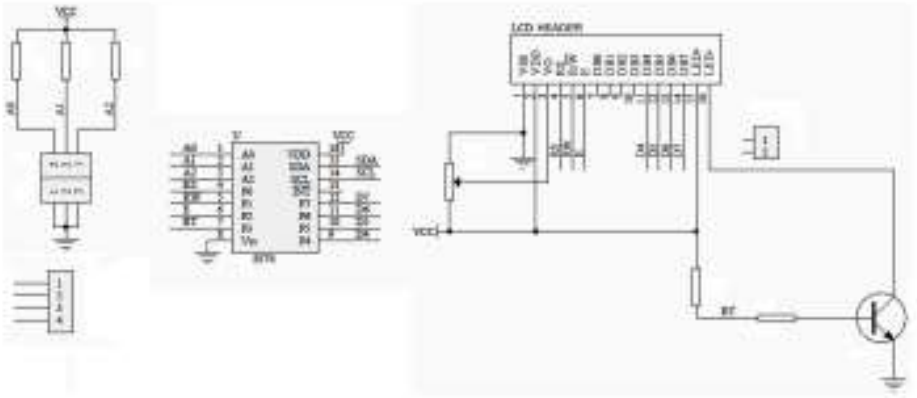
Pin connectivity			Address of PCF8574A								Address byte value		7-bit hexadecimal address without R/W
A2	A1	A0	A6	A5	A4	A3	A2	A1	A0	R/W	Write	Read	
V_{SS}	V_{SS}	V_{SS}	0	1	1	1	0	0	0	-	70h	71h	38h
V_{SS}	V_{SS}	V_{DD}	0	1	1	1	0	0	1	-	72h	73h	39h
V_{SS}	V_{DD}	V_{SS}	0	1	1	1	0	1	0	-	74h	75h	3Ah
V_{SS}	V_{DD}	V_{DD}	0	1	1	1	0	1	1	-	76h	77h	3Bh
V_{DD}	V_{SS}	V_{SS}	0	1	1	1	1	0	0	-	78h	79h	3Ch
V_{DD}	V_{SS}	V_{DD}	0	1	1	1	1	0	1	-	7Ah	7Bh	3Dh
V_{DD}	V_{DD}	V_{SS}	0	1	1	1	1	1	0	-	7Ch	7Dh	3Eh
V_{DD}	V_{DD}	V_{DD}	0	1	1	1	1	1	1	-	7Eh	7Fh	3Fh

Address Setting of PCD8574A (extract from PCF8574A data specs).

Note: When the pad A0~A2 is open, the pin is pull up to VDD. When the pin is solder shorted, it is pull down to VSS.

The default setting of this module is A0~A2 all open, so is pull up to VDD. The address is 3Fh in this case.

Reference circuit diagram of an Arduino-compatible LCD backpack is shown below. What follows next is information on how to use one of these inexpensive backpacks to interface with a microcontroller in ways it was exactly intended.



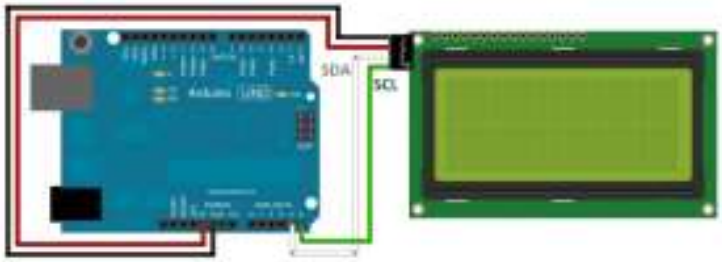
Reference circuit diagram of the I2C-to-LCD piggy-back board.

I2C LCD Display.

At first you need to solder the I2C-to-LCD piggy-back board to the 16-pins LCD module. Ensure that the I2C-to-LCD piggy-back board pins are straight and fit in the LCD module, then solder in the first pin while keeping the I2C-to-LCD piggy-back board in the same plane with the LCD module. Once you have finished the soldering work, get four jumper wires and connect the LCD module to your Arduino as per the instruction given below.

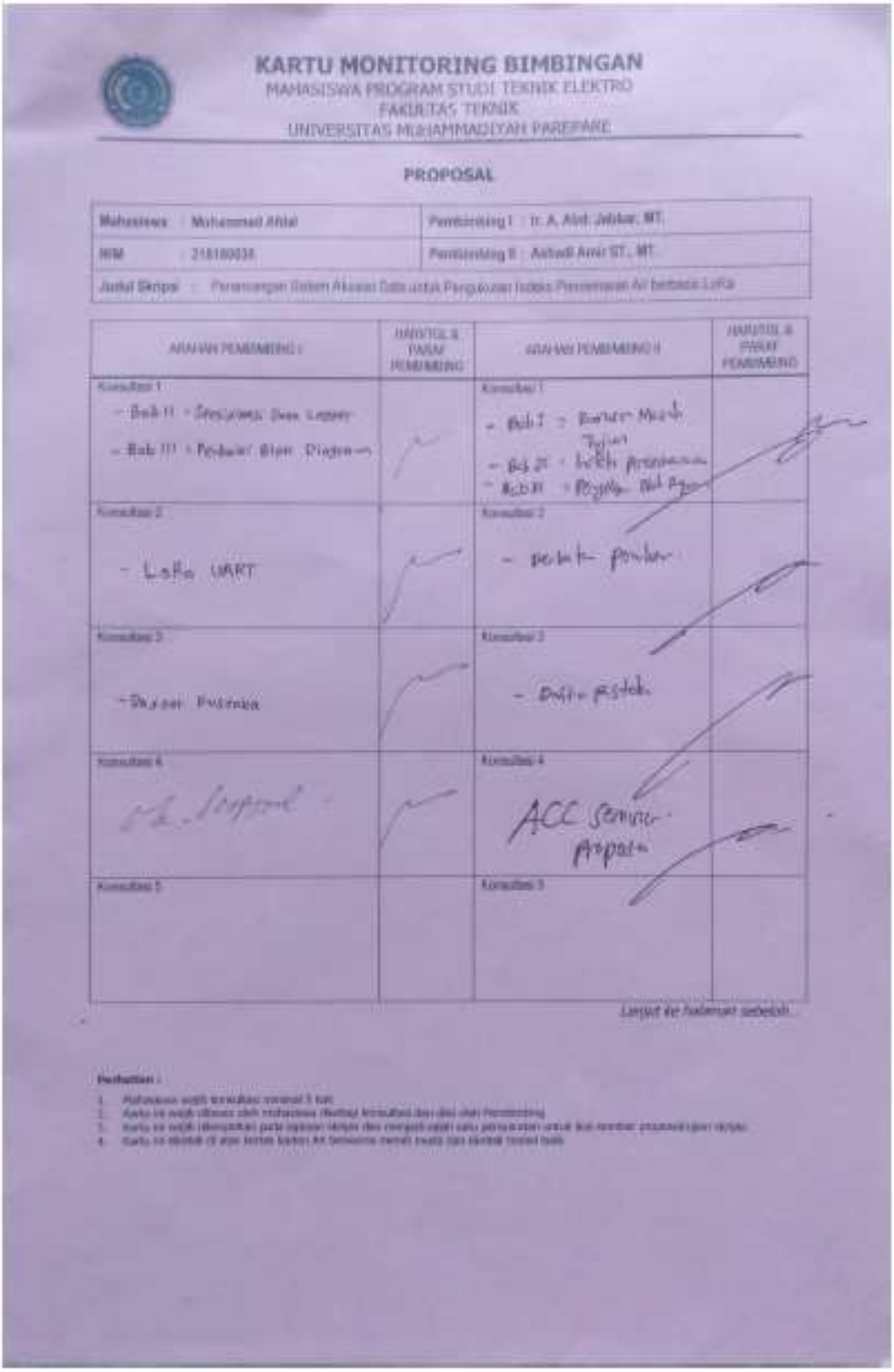


LCD display to Arduino wiring.



Lampiran-4 Kartu Monitoring Bimbingan

1. Kartu Monitoring Bimbingan Propsal



KARTU MONITORING BIMBINGAN
MAHASISWA PROGRAM STUDI TEKNIK ELEKTRO
FAKULTAS TEKNIK
UNIVERSITAS MUHAMMADIYAH PAREPARE

PROPOSAL

Mahasiswa	Muhammad Afdal	Pembimbing I	Ir. A. Abd. Jabbar, MT.
NIM	21810033	Pembimbing II	Asfadil Amir ST., MT.
Judul Skripsi	Perancangan Sistem Akuisisi Data untuk Pengukuran Indeks Pencemaran Air berbasis LoRa		

ARAHAN PEMBIMBING I	PARAF PEMBIMBING	ARAHAN PEMBIMBING II	PARAF PEMBIMBING
Konsultasi 1 - Bab II : Tinjauan Pustaka - Bab III : Perancangan Blok Diagram		Konsultasi 1 - Bab I : Latar Belakang - Bab II : Teori Pendukung - Bab III : Perancangan Blok Diagram	
Konsultasi 2 - LoRa UART		Konsultasi 2 - perbaiki penulisan	
Konsultasi 3 - Diagram Pulsanya		Konsultasi 3 - Daftar pustaka	
Konsultasi 4 ACC Proposal		Konsultasi 4 ACC Seminar Proposal	
Konsultasi 5		Konsultasi 5	

Lanjut ke halaman sebelah

Perhatian :
1. Mahasiswa wajib konsultasi minimal 5 kali
2. Kartu ini wajib diisi oleh mahasiswa (Setiap konsultasi dan diisi oleh Pembimbing)
3. Kartu ini wajib dilampirkan pada laporan hasil dan mengisi salah satu persyaratan untuk ikut ujian proposal/ujian hasil
4. Kartu ini diberikan di atas kertas A4 berwarna merah muda dan diketik rapi

Lampiran :

KONSULTASI PEMBIMBING I	HARI/TGL & PARAF PEMBIMBING	KONSULTASI PEMBIMBING II	HARI/TGL & PARAF PEMBIMBING
Konsultasi 6		Konsultasi 6	
Konsultasi 7		Konsultasi 7	
Konsultasi 8		Konsultasi 8	
Konsultasi 9		Konsultasi 9	
Konsultasi 10		Konsultasi 10	

Parepare, 11 / 7 / 2023

Mengetahui
Ketua Program Studi

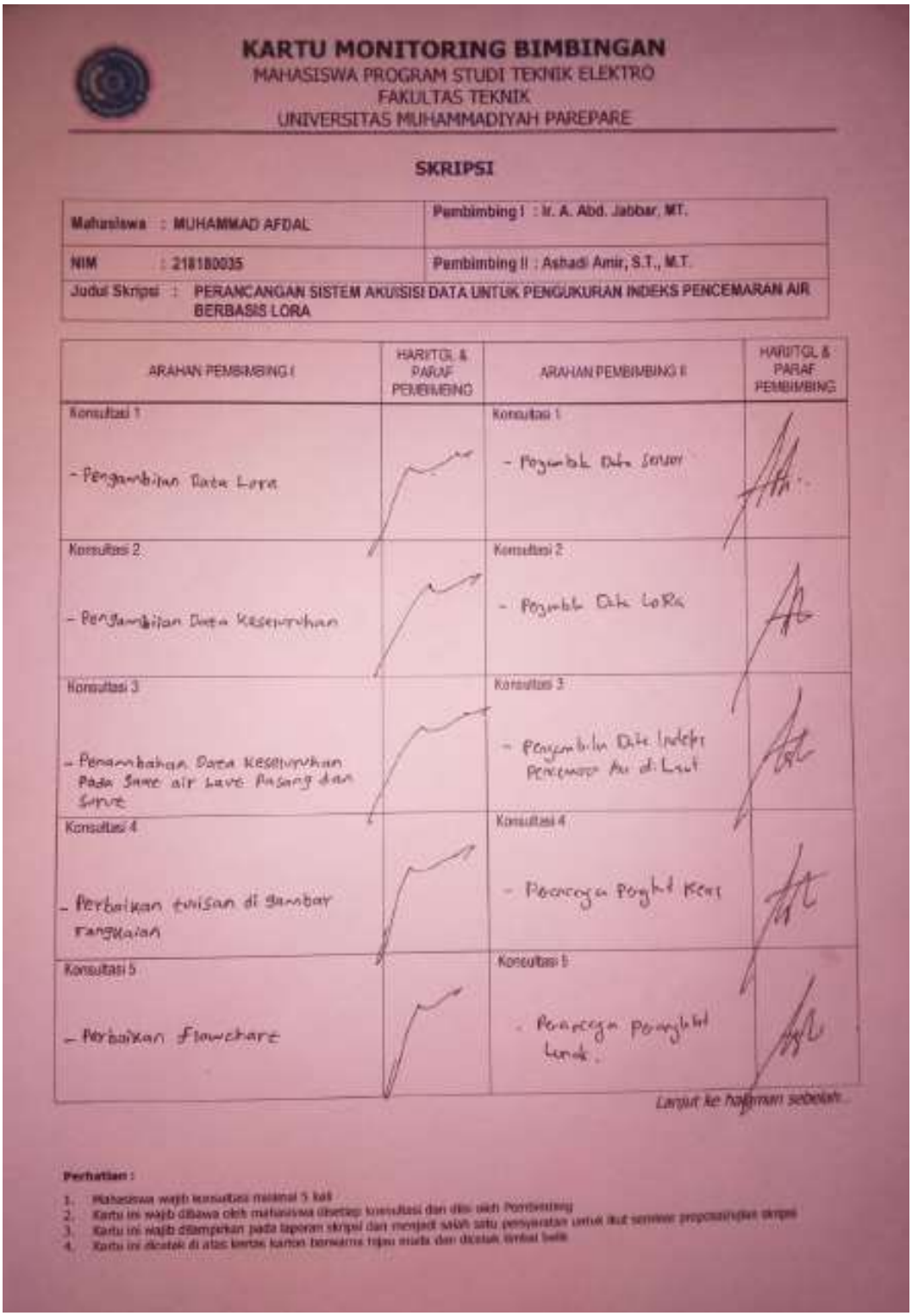
Asrul, ST., MT.
NIDN. 946 836

Mahasiswa

Muhammad Afdal
NIM. 2101180035

Perhatian :

1. Mahasiswa wajib konsultasi minimal 5 kali
2. Kartu ini wajib diisi oleh mahasiswa bimbing konsultasi dan diisi oleh pembimbing
3. Kartu ini wajib ditempelkan pada lembar disposisi dan menjadi salah satu persyaratan untuk ikut seminar proposal dan ujian
4. Kartu ini dibuat 10 atau kartu kontrol AA berwarna merah muda dan disimpan setelah ujian



KARTU MONITORING BIMBINGAN
MAHASISWA PROGRAM STUDI TEKNIK ELEKTRO
FAKULTAS TEKNIK
UNIVERSITAS MUHAMMADIYAH PAREPARE

SKRIPSI

Mahasiswa : MUHAMMAD AFDAL	Pembimbing I : Ir. A. Abd. Jabbar, MT.
NIM : 218180035	Pembimbing II : Ashadi Amir, S.T., M.T.
Judul Skripsi : PERANCANGAN SISTEM AKUISISI DATA UNTUK PENGUKURAN INDEKS PENCEMARAN AIR BERBASIS LORA	

ARAHAN PEMBIMBING I	HARI/TGL & PARAF PEMBIMBING	ARAHAN PEMBIMBING II	HARI/TGL & PARAF PEMBIMBING
Konsultasi 1 - Pengambilan Data Lora		Konsultasi 1 - Pengirim Data Server	
Konsultasi 2 - Pengambilan Data Keseluruhan		Konsultasi 2 - Pengirim Data LoRa	
Konsultasi 3 - Penambahan Data Keseluruhan Pada Sensor air Save Pasang dan Surut		Konsultasi 3 - Pengambilan Data Indeks Pencemaran Air di Laut	
Konsultasi 4 - Perbaikan tulisan di gambar rangkaian		Konsultasi 4 - Perancangan Penghubung Kerja	
Konsultasi 5 - Perbaikan Flowchart		Konsultasi 5 - Perancangan Penghubung Lanjut.	

Lanjut ke halaman sebelah

Perhatian :

1. Mahasiswa wajib konsultasi minimal 5 kali
2. Kartu ini wajib dibawa oleh mahasiswa disetiap konsultasi dan diisi oleh Pembimbing
3. Kartu ini wajib ditempelkan pada laporan skripsi dan menjadi salah satu persyaratan untuk ikut seminar proposal/hasil skripsi
4. Kartu ini dicetak di atas kertas kanton berwarna hijau muda dan dicetak timbal balik

Lanjutan ...

ARAHAN PEMBIMBING I	HARI/TGL & PARAF PEMBIMBING	ARAHAN PEMBIMBING II	HARI/TGL & PARAF PEMBIMBING
Konsultasi 6 - Pengujian Data		Konsultasi 6 - Penyajian Data	
Konsultasi 7 - Tampilan Data Logger ke Laptop		Konsultasi 7 - Analisa Data	
Konsultasi 8 - Kesimpulan		Konsultasi 8 - Revisi - Daftar Pustaka	
Konsultasi 9 Ok ac Seminar Hasil	30/4-24	Konsultasi 9 ACC Seminar Hasil	
Konsultasi 10		Konsultasi 10	

Parepare, 30 April 2024

Mengetahui
Ketua Program Studi

Asrul, ST., MT
NBM. 986 836

Mahasiswa

MUHAMMAD AFDAL
NIM. 21B180035

Perhatian :

1. Mahasiswa wajib konsultasi minimal 5 kali
2. Kartu ini wajib diberia oleh mahasiswa disetiap konsultasi dan diisi oleh Pembimbing
3. Kartu ini wajib ditampilkan pada laporan skripsi dan menjadi salah satu persyaratan untuk ikut seminar proposal/ujian skripsi
4. Kartu ini diketik di atas kertas karton berwarna hijau muda dan diketak sistvil balik