CAP110

DJS-1 CAPTEUR DE CONDUCTIVITE ET QUALITE DE L EAU COMPATIBLE ARDUINO ET STM32

Note: This

The difference between this product and TDS products is: large range: 1ms/cm--20ms/cm. T D S is suitable for short-range measurement of household water quality

Product Introduction

This analog conductivity meter has the characteristics of simple connection, convenience and practicality. After connecting according to the schematic diagram, and then through the program control, the conductivity of the solution can be measured very conveniently.

The most important thing is that we open source all designs and codes. Users can easily collect conductivity data through Ar duino for further control and research. Fans are welcome to study, share, and provide valuable opinions.

Conductivity is the ability of a substance to carry current, and it is the reciprocal of resistivity. In liquids, the reciprocal of electrical resistance is often used to measure its electrical conductivity. The conductivity of water is a very important indicator to measure water quality, and it can reflect the degree of electrolytes present in the water. Depending on the concentration of the electrolyte in the aqueous solution, the degree of conductivity of the solution is also different. In the International System of Units, the unit of conductivity is called Siemens/meter (S/m), and other units are: S/m, mS/cm, μS/cm.

Scope of application

- Water quality testing
- Mixed agriculture
- aquaculture
- Watch the aquarium

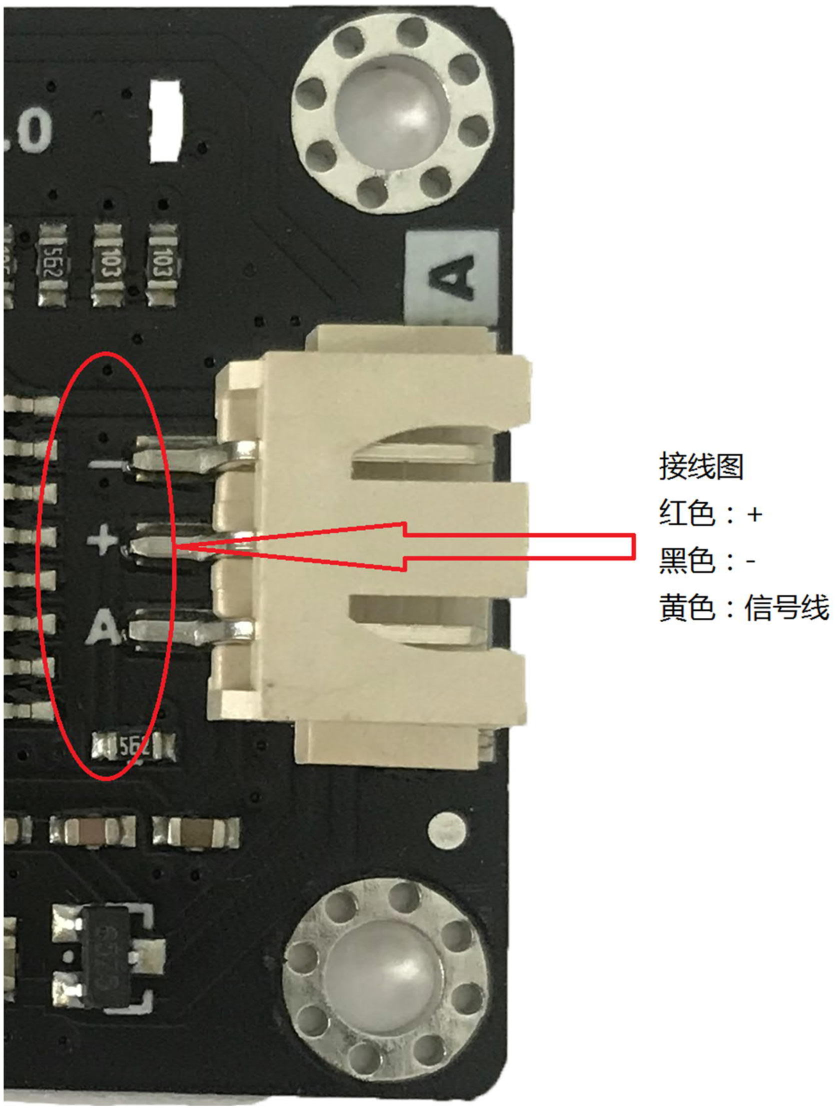
technical parameter

- Working voltage: +5.00V
- PCB size: 45mm×32mm
- Measuring range: 1ms/cm--20ms/cm
- Applicable temperature: 5-40℃
- Accuracy: <±10%FS (The specific accuracy depends on your calibration accuracy)
- XH2.54 interface (3-pin patch)
- BNC interface type conductivity electrode (electrode constant is 1)
- Conductivity electrode cable length: about 60 cm
- Power Indicator

Wiring diagram

Special note: Since the earlier analog connection lines have different colors in different batches,

please follow the silk-screen logo-+ A wiring on the sensor circuit board.



EC Meter ---- Ar duino

V ---- 5.0V;

G ---- GND;

A ---- Analog IO (corresponding to the source code)

Invoice

- Conductivity electrode with BNC interface 1 pc

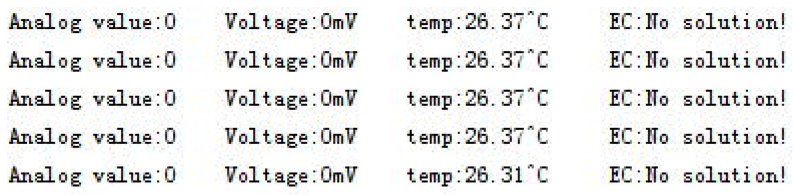
- EC Meter circuit board 1
- Adapter cable 1

Steps for usage

Notice:

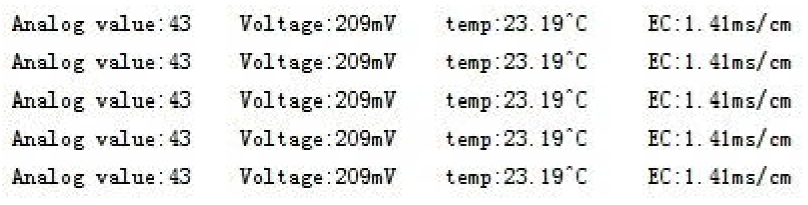
- Please use an external power supply to make the supply voltage of EC Meter close to +5.00V. The more accurate the voltage, the higher the accuracy.
- Before measuring a different solution, clean the conductivity electrode and temperature sensor with clean water to prevent inaccurate reading and contamination of the solution. It is recommended to use deionized water for cleaning.
- When measuring the conductivity of the solution, make sure that the temperature electrode is inserted into the solution to be measured, and the solution is stirred with the conductivity electrode so that the conductive part of the conductivity electrode fully contacts the solution. After the temperature value and the conductivity value are stable, you can read the required value. Affected by the polarization of the solution, when measuring high-conductivity solutions, the conductivity indication will appear to jitter within a certain range. The higher the conductivity, the greater the jitter.

- (1) Connect each device as shown in the figure, that is, connect the conductivity electrode to the BNC interface of the EC Meter circuit board, and then use the analog cable to connect the EC Meter circuit board to the analog port 1 of the Arduino main controller. Then connect the waterproof DS18B20 temperature sensor to the terminal of the pluggable sensor adapter, and then use the digital cable to connect to the digital port 2 of the Arduino main controller. After powering on the Arduino main controller, you can see that the blue indicator light of the EC Meter circuit board turns on.
  (2) Write sample code to Arduino main controller.
  (3) Open the serial monitor of Arduino IDE, at this time, it can output some parameters, such as voltage value, temperature value, and prompt that there is no solution.



```
Analog value:0    Voltage:0mV    temp:26.37^C    EC:No solution!
Analog value:0    Voltage:0mV    temp:26.37^C    EC:No solution!
Analog value:0    Voltage:0mV    temp:26.37^C    EC:No solution!
Analog value:0    Voltage:0mV    temp:26.37^C    EC:No solution!
Analog value:0    Voltage:0mV    temp:26.31^C    EC:No solution!
```

(4) Insert the conductivity electrode and temperature sensor into the calibration solution to measure the conductivity value of the solution. Stir the solution and wait until the number is stable. If the reading is close to the value marked on the standard solution bottle, it can be put into use. Take the test conductivity solution with a conductivity of 1413us/cm as an example:

- 

```
Analog value:43    Voltage:209mV    temp:23.19^C    EC:1.41ms/cm
Analog value:43    Voltage:209mV    temp:23.19^C    EC:1.41ms/cm
Analog value:43    Voltage:209mV    temp:23.19^C    EC:1.41ms/cm
Analog value:43    Voltage:209mV    temp:23.19^C    EC:1.41ms/cm
Analog value:43    Voltage:209mV    temp:23.19^C    EC:1.41ms/cm
```

Ar duino source code:

```
#include <OneWire.h>

#define StartConvert 0
#define ReadTemperature 1

const byte numReadings = 20; //the number of sample times
byte ECsensorPin = A0; //EC Meter analog output,pin on analog 1
byte DS18B20_Pin = 2; //DS18B20 signal, pin on digital 2
unsigned int AnalogSampleInterval=25,printInterval =700,tempSampleInterval=850; //analog
sample interval;serial print interval;temperature sample interval
unsigned int readings[numReadings]; // the readings from the analog input
byte index = 0; // the index of the current reading
unsigned long AnalogValueTotal = 0; // the running total
unsigned int AnalogAverage = 0,averageVoltage=0; // the average
unsigned long AnalogSampleTime,printTime,tempSampleTime;
float temperature,ECcurrent;

//Temperature chip i/o
OneWire ds(DS18B20_Pin); // on digital pin 2

void setup() {
 // initialize serial communication with computer:
  Serial.begin(115200);
  // initialize all the readings to 0:
  for (byte thisReading = 0; thisReading <numReadings; thisReading++)
    readings[thisReading] = 0;
  TempProcess (StartConvert); //let the DS18B20 start the convert
  AnalogSampleTime=millis();
  printTime=millis();
  tempSampleTime=millis();
}

void loop() {
  /*
   Every once in a while,sample the analog value and calculate the average.
  */
  if(millis()-AnalogSampleTime>=AnalogSampleInterval)
  {
    AnalogSampleTime=millis();
     // subtract the last reading:
    AnalogValueTotal = AnalogValueTotal-readings[index];
    // read from the sensor:
    readings[index] = analogRead(ECsensorPin);
    // add the reading to the total:
    AnalogValueTotal = AnalogValueTotal + readings[index];
    // advance to the next position in the array:
    index = index + 1;
    // if we're at the end of the array...
    if (index >= numReadings)
    // ...wrap around to the beginning:
    index = 0;
    // calculate the average:
    AnalogAverage = AnalogValueTotal / numReadings;
  }
  /*
   Every once in a while, MCU read the temperature from the DS18B20 and then let the DS18B20
start the convert.
   Attention: The interval between start the convert and read the temperature should be greater
than 750 millisecond, or the temperature is not accurate!
  */
   if(millis()-tempSampleTime>=tempSampleInterval)
  {
    tempSampleTime=millis();
    temperature = 25; // read the current temperature from the DS18B20
    TempProcess(StartConvert); //after the reading,start the convert for next reading
  }
   /*
   Every once in a while,print the information on the serial monitor.
  */
  if(millis()-printTime>=printInterval)
  {
    printTime=millis();
    averageVoltage=AnalogAverage*(float)5000/1024;
    Serial.print(" Analog value:");
    Serial.print(AnalogAverage); //analog average,from 0 to 1023
    Serial.print(" Voltage:");
    Serial.print(averageVoltage); //millivolt average,from 0mv to 4995mV
    Serial. print("mV ");
    Serial.print("temp:");
    Serial.print(temperature); //current temperature
    Serial.print("^C EC:");

    float TempCoefficient=1.0+0.0185*(25-25.0); //temperature compensation formula:
```

```
fFinalResult(25^C) = fFinalResult(current)/(1.0+0.0185*(fTP-25.0));
   float CoefficientVolatge=(float)averageVoltage /TempCoefficient;
   if(CoefficientVolatge<150)Serial.println("No solution!"); //25^C 1413us/cm<-->about 216mv if
the voltage(compensate)<150,that is <1ms/cm, out of the range
   else if(CoefficientVolatge>3300)Serial.println("Out of the range!"); //>20ms/cm,out of the
range
   else
   {
     if(CoefficientVolatge<=448)ECcurrent=6.84*CoefficientVolatge- 64.32;
//1ms/cm<EC<=3ms/cm
     else if(CoefficientVolatge<=1457)ECcurrent=6.98*CoefficientVolatge-127;
//3ms/cm<EC<=10ms/cm
     else ECcurrent=5.3*CoefficientVolatge+2278; //10ms/cm<EC<20ms/cm
     ECcurrent/=1000; //convert us/cm to ms/cm
     Serial.print(ECcurrent,2); //two decimal
     Serial. println("ms/cm");
   }
  }

}
/*
ch=0,let the DS18B20 start the convert;ch=1,MCU read the current temperature from the
DS18B20.
*/
float TempProcess(bool ch)
{
  //returns the temperature from one DS18B20 in DEG Celsius
  static byte data [12];
  static byte addr[8];
  static float TemperatureSum;
  if(!ch){
          if (!ds.search(addr)) {
              Serial.println("no more sensors on chain, reset search!");
              ds .reset_search();
              return 0;
          }
          if (OneWire::crc8( addr, 7) != addr[7]) {
              Serial.println("CRC is not valid!");
              return 0;
          }
          if (addr[0] != 0x10 && addr[0] != 0x28) {
              Serial.print("Device is not recognized!");
              return 0;
          }
          ds.reset();
          ds.select(addr);
          ds .write(0x44,1); // start conversion, with parasite power on at the end
  }
  else{
          byte present = ds.reset();
          ds.select(addr);
          ds.write(0xBE); // Read Scratchpad
          for (int i = 0; i <9; i++) {// we need 9 bytes
            data[i] = ds.read();
          }
```

```
        ds.reset_search();
        byte MSB = data[1];
        byte LSB = data [0];
        float tempRead = ((MSB << 8) LSB); //using two's compliment
        TemperatureSum = tempRead / 16;
    }
        return TemperatureSum;
}
```