```
#include <OneWire.h>
OneWire  ds(A12);  // on pin 10 (a 4.7K
resistor is necessary)
#define Buzzer 14
#define Led_Hijau 17
#define Led_Kuning 16
#define Led_Merah 15
#define V_R A6
#define V_S A7
#define V_T A8
#define I_R A9
#define I_S A10
#define I_T A11
#define T_Trafo A12
#define T_Genset A13
#define T_Kubikel A14
int tampil;
 float getIPPr;
 float resultir;
 int readValueir;           //value read from
the sensor
float getIPPs;
 float resultis;
 int readValueis;           //value read from
the sensor
float getIPPt;
float resultit;
int readValueit;           //value read from
the sensor
 float getVPPr;
 float resultr;
 int readValuer;           //value read from
the sensor
 float getVPPs;
  //
 float results;
 int readValues;           //value read from
the sensor
 //
float getVPPt;
float resultt;
int readValuet;           //value read from
the sensor
int ulang;
int ds1,ds2,ds3;
const int sensorInr = A6;
double mVperAmpr = 2.42; // use 100 for
20A Module and 66 for 30A Module
double Voltager = 0;
double VRMSr = 0;
double AmpsRMSr = 0;
const int sensorIns = A7;
double mVperAmps = 2.61; // use 100 for
20A Module and 66 for 30A Module
double Voltages = 0;
double VRMSs = 0;
double AmpsRMSs = 0;
const int sensorInt = A8;
double mVperAmpt = 2.57; // use 100 for
20A Module and 66 for 30A Module
double Voltaget = 0;
double VRMSt = 0;
double AmpsRMSt = 0;
const int sensorInir = A9;
double mIperAmpir = 1; // use 100 for
20A Module and 66 for 30A Module
double Currentir = 0;
double IRMSr = 0;
double ImpsRMSr = 0;
const int sensorInis = A10;
double mIperAmpis = 1; // use 100 for
20A Module and 66 for 30A Module
double Currentis = 0;
double IRMSs = 0;
double ImpsRMSs = 0;
const int sensorInit = A11;
double mIperAmpit = 1; // use 100 for
20A Module and 66 for 30A Module
double Currentit= 0;
double IRMSt = 0;
double ImpsRMSt = 0;
/*
int data_vr;
int data_vs;
int data_vt;
*/
int data_ir;
int data_is;
int data_it;
int data_ttrafo;
```

```
int data_tgenset;
int data_tkubikel;
/*
int h_vr;
int h_vs;
int h_vt;
*/
int h_ir;
int h_is;
int h_it;
unsigned int j_ttrafo;
unsigned int j_tgenset;
unsigned int j_tkubikel;
float temptrafo,tempgenset,tempkubikel;
float arus_r;
float arus_s;
float arus_t;
float suhu_trafo;
float suhu_genset;
float suhu_kubikel;
int jumlah;
#include <LiquidCrystal.h>
LiquidCrystal lcd(53,52,51,50,49,48);
void setup() {
pinMode( Buzzer,OUTPUT);
pinMode(  Led_Hijau ,OUTPUT);
pinMode(  Led_Kuning ,OUTPUT);
pinMode(  Led_Merah ,OUTPUT);
digitalWrite (Buzzer,HIGH);
digitalWrite (  Led_Hijau ,HIGH);
digitalWrite (  Led_Kuning ,HIGH);
digitalWrite(  Led_Merah ,HIGH);
 lcd.begin(20,4);
// put your setup code here, to run once:
pinMode(Buzzer,OUTPUT);
Serial.begin(9600);
lcd.setCursor(4,0);
lcd.write(" Politeknik ");
lcd.setCursor(6,1);
lcd.write(" Negeri ");
lcd.setCursor(4,2);
lcd.write(" Sriwijaya ");
lcd.setCursor(1,3);
lcd.write(" www.polsri.ac.id ");
delay(1000);
digitalWrite (Buzzer,LOW);
digitalWrite (  Led_Hijau ,LOW);
digitalWrite (  Led_Kuning ,LOW);
digitalWrite(  Led_Merah ,LOW);
lcd.clear();
}
void loop()
// put your main code here, to run
repeatedly:
{
byte i;
byte present = 0;
byte type_s;
byte data[12];
byte addr[8];
float celsius, fahrenheit;
if ( !ds.search(addr)) {
jumlah=0;
// Serial.println("No more addresses.");
// Serial.println();
ds.reset_search();
delay(250);
return;
}
// Serial.print("ROM =");
for( i = 0; i < 8; i++) {
// Serial.write(' ');
// Serial.print(addr[i], HEX);
}
if (OneWire::crc8(addr, 7) != addr[7]) {
// Serial.println("CRC is not valid!");
return;
}
 // Serial.println();
// the first ROM byte indicates which chip
switch (addr[0]) {
case 0x10:
// Serial.println("  Chip = DS18S20");  //
or old DS1820
type_s = 1;
break;
case 0x28:
// Serial.println("  Chip = DS18B20");
type_s = 0;
break;
case 0x22:
// Serial.println("  Chip = DS1822");
```

```
type_s = 0;
break;
default:
      //Serial.println("Device is not a
DS18x20 family device.");
      return;
  }

  ds.reset();
  ds.select(addr);
  ds.write(0x44, 1);        // start conversion,
with parasite power on at the end

  delay(1000);     // maybe 750ms is
enough, maybe not
  // we might do a ds.depower() here, but
the reset will take care of it.

  present = ds.reset();
  ds.select(addr);
  ds.write(0xBE);         // Read Scratchpad
jumlah++;
  //Serial.print("  Data = ");
  //Serial.print(present, HEX);
  //Serial.print(" ");
  for ( i = 0; i < 9; i++) {           // we need
9 bytes
    data[i] = ds.read();
   // Serial.print(data[i], HEX);
   // Serial.print(" ");
  }
//Serial.print(" CRC=");
  //Serial.print(OneWire::crc8(data, 8),
HEX);
  //Serial.println();
  // Convert the data to actual temperature
  // because the result is a 16 bit signed
integer, it should
  // be stored to an "int16_t" type, which is
always 16 bits
  // even when compiled on a 32 bit
processor.
  int16_t raw = (data[1] << 8) | data[0];
  if (type_s) {
    raw = raw << 3; // 9 bit resolution
default
    if (data[7] == 0x10) {
      // "count remain" gives full 12 bit
resolution
      raw = (raw & 0xFFF0) + 12 - data[6];
    }
  } else {
    byte cfg = (data[4] & 0x60);
    // at lower res, the low bits are
undefined, so let's zero them
    if (cfg == 0x00) raw = raw & ~7;  // 9
bit resolution, 93.75 ms
    else if (cfg == 0x20) raw = raw & ~3; //
10 bit res, 187.5 ms
    else if (cfg == 0x40) raw = raw & ~1; //
11 bit res, 375 ms
    //// default is 12 bit resolution, 750 ms
conversion time
  }
  celsius = (float)raw / 15.14;
  fahrenheit = celsius * 1.8 + 32.0
 // Serial.print(" jumlah ");
 // Serial.println(jumlah);
 if(jumlah==1){  temptrafo=celsius;}
if(jumlah==2){  tempgenset=celsius;}
if(jumlah==3){  tempkubikel=celsius;}
//tampil
lcd.setCursor(12,0);
lcd.print("r:");
lcd.print(temptrafo);
lcd.setCursor(12,1);
lcd.print("s:");
lcd.print(tempgenset);
lcd.setCursor(12,2);
lcd.print("t:");
lcd.print(tempkubikel);
dogetVPP();
 Voltager = resultr;
 VRMSr = (Voltager) *1;
 AmpsRMSr = (VRMSr)/mVperAmpr;
 int maxValuer = 0;          // store max
value here
 int minValuer = 1024;
 Voltages =  results;
 VRMSs = (Voltages) *1;
 AmpsRMSs = (VRMSs)/mVperAmps;
```

```
  int maxValues = 0;          // store max
value here
 int minValues = 1024;
 Voltaget =  resultt;
 VRMSt = (Voltaget) *1;
 AmpsRMSt = (VRMSt)/mVperAmpt;
  int maxValuet = 0;          // store max
value here
 int minValuet = 1024;
//tampil
lcd.setCursor(0,0);
lcd.print("R:");
lcd.print(AmpsRMSr);
lcd.print("");
lcd.setCursor(0,1);
lcd.print("S:");
lcd.print(AmpsRMSs);
lcd.print("");
lcd.setCursor(0,2);
lcd.print("T:");
lcd.print(AmpsRMSt);
lcd.print("");
lcd.setCursor(6,0);
lcd.print("IR:");
lcd.setCursor(6,1);
lcd.print("IS:");
lcd.setCursor(6,2);
lcd.print("IT:");
 dogetIPP();
 Currentir = resultir;
 IRMSr = (Currentir) *1;
 ImpsRMSr = (IRMSr )/mIperAmpir;
  int maxValueir = 0;          // store max
value here
 int minValueir = 1024;
 Currentis = resultis;
 IRMSs = (Currentis) *1;
 ImpsRMSs = (IRMSs )/mIperAmpis;
  int maxValueis = 0;          // store max
value here
 int minValueis = 1024;
 Currentit = resultit;
 IRMSt = (Currentit) *1;
 ImpsRMSt = (IRMSt )/mIperAmpit;
  int maxValueit = 0;          // store max
value here
 int minValueit = 1024;
 //tampil
    lcd.setCursor(6,0);
lcd.print("IR:");
lcd.print(ImpsRMSr);
lcd.print("  ");
lcd.setCursor(6,1);
lcd.print("IS:");
lcd.print(ImpsRMSs);
lcd.print("  ");
lcd.setCursor(6,2);
lcd.print("IT:");
lcd.print(ImpsRMSt);
lcd.print("  ");
  //--------------------
 /*
digitalWrite (Buzzer,HIGH);
digitalWrite (  Led_Hijau ,HIGH);
digitalWrite (  Led_Kuning ,HIGH);
pinMode(  Led_Merah ,HIGH);
  */
if (AmpsRMSr>=0  &&
AmpsRMSr<=475){digitalWrite(
Led_Hijau ,HIGH); digitalWrite(
Led_Kuning ,LOW);
digitalWrite(Led_Merah
,LOW);digitalWrite (Buzzer,LOW);}
if (AmpsRMSs>=0  &&
AmpsRMSs<=475){digitalWrite(
Led_Hijau ,HIGH); digitalWrite(
Led_Kuning ,LOW);
digitalWrite(Led_Merah
,LOW);digitalWrite (Buzzer,LOW);}
if (AmpsRMSt>=0  &&
AmpsRMSt<=475){digitalWrite(
Led_Hijau ,HIGH); digitalWrite(
Led_Kuning ,LOW);
digitalWrite(Led_Merah
,LOW);digitalWrite (Buzzer,LOW);}
if (ImpsRMSr>=200  &&
ImpsRMSr<=800){digitalWrite(
Led_Hijau ,HIGH); digitalWrite(
Led_Kuning ,LOW);
digitalWrite(Led_Merah
,LOW);digitalWrite (Buzzer,LOW);}
```

```
if (ImpsRMSs>=200  &&
ImpsRMSs<=800){digitalWrite(
Led_Hijau ,HIGH); digitalWrite(
Led_Kuning ,LOW);
digitalWrite(Led_Merah
,LOW);digitalWrite (Buzzer,LOW);}
if (ImpsRMSt>=200  &&
ImpsRMSt<=800){digitalWrite(
Led_Hijau ,HIGH); digitalWrite(
Led_Kuning ,LOW);
digitalWrite(Led_Merah
,LOW);digitalWrite (Buzzer,LOW);}
if (temptrafo>=30  &&
temptrafo<=100){digitalWrite(
Led_Hijau ,HIGH); digitalWrite(
Led_Kuning ,LOW);
digitalWrite(Led_Merah
,LOW);digitalWrite (Buzzer,LOW);}
if (tempgenset>=30  &&
tempgenset<=100){digitalWrite(
Led_Hijau ,HIGH); digitalWrite(
Led_Kuning ,LOW);
digitalWrite(Led_Merah
,LOW);digitalWrite (Buzzer,LOW);}
if (tempkubikel>=30  &&
tempkubikel<=100){digitalWrite(
Led_Hijau ,HIGH); digitalWrite(
Led_Kuning ,LOW);
digitalWrite(Led_Merah
,LOW);digitalWrite (Buzzer,LOW);}
//Set point LED KUNING
if (AmpsRMSr>476  &&
AmpsRMSr<=480){digitalWrite(
Led_Hijau ,LOW); digitalWrite(
Led_Kuning ,HIGH);
digitalWrite(Led_Merah
,LOW);digitalWrite (Buzzer,LOW);}
if (AmpsRMSs>476  &&
AmpsRMSs<=480){digitalWrite(
Led_Hijau ,LOW); digitalWrite(
Led_Kuning ,HIGH);
digitalWrite(Led_Merah
,LOW);digitalWrite (Buzzer,LOW);}
if (AmpsRMSt>476  &&
AmpsRMSt<=480){digitalWrite(
Led_Hijau ,LOW); digitalWrite(
Led_Kuning ,HIGH);
digitalWrite(Led_Merah
,LOW);digitalWrite (Buzzer,LOW);}
if (ImpsRMSr>801  &&
ImpsRMSr<=900){digitalWrite(
Led_Hijau ,LOW); digitalWrite(
Led_Kuning ,HIGH);
digitalWrite(Led_Merah
,LOW);digitalWrite (Buzzer,LOW);}
if (ImpsRMSs>801  &&
ImpsRMSs<=900){digitalWrite(
Led_Hijau ,LOW); digitalWrite(
Led_Kuning ,HIGH);
digitalWrite(Led_Merah
,LOW);digitalWrite (Buzzer,LOW);}
if (ImpsRMSt>801  &&
ImpsRMSt<=900){digitalWrite(
Led_Hijau ,LOW); digitalWrite(
Led_Kuning ,HIGH);
digitalWrite(Led_Merah
,LOW);digitalWrite (Buzzer,LOW);}
if (temptrafo>100  &&
temptrafo<=115){digitalWrite(
Led_Hijau ,LOW); digitalWrite(
Led_Kuning ,HIGH);
digitalWrite(Led_Merah
,LOW);digitalWrite (Buzzer,LOW);}
if (tempkubikel>100 &&
tempkubikel<=115){digitalWrite(
Led_Hijau ,LOW); digitalWrite(
Led_Kuning ,HIGH);
digitalWrite(Led_Merah
,LOW);digitalWrite (Buzzer,LOW);}
if (tempgenset>100  &&
tempgenset<=115){digitalWrite(
Led_Hijau ,LOW); digitalWrite(
Led_Kuning ,HIGH);
digitalWrite(Led_Merah
,LOW);digitalWrite (Buzzer,LOW);}
//Set point LED MERAH BUZZER
if (AmpsRMSr>490){digitalWrite(
Led_Hijau ,LOW); digitalWrite(
Led_Kuning ,LOW);
digitalWrite(Led_Merah
,HIGH);digitalWrite (Buzzer,HIGH);}
```

```
if (AmpsRMSs>490){digitalWrite(
Led_Hijau ,LOW); digitalWrite(
Led_Kuning ,LOW);
digitalWrite(Led_Merah
,HIGH);digitalWrite (Buzzer,HIGH);}
if (AmpsRMSt>490){digitalWrite(
Led_Hijau ,LOW); digitalWrite(
Led_Kuning ,LOW);
digitalWrite(Led_Merah
,HIGH);digitalWrite (Buzzer,HIGH);}
if (ImpsRMSr>901){digitalWrite(
Led_Hijau ,LOW); digitalWrite(
Led_Kuning ,LOW);
digitalWrite(Led_Merah
,HIGH);digitalWrite (Buzzer,HIGH);}
if (ImpsRMSs>901){digitalWrite(
Led_Hijau ,LOW); digitalWrite(
Led_Kuning ,LOW);
digitalWrite(Led_Merah
,HIGH);digitalWrite (Buzzer,HIGH);}
if (ImpsRMSt>901){digitalWrite(
Led_Hijau ,LOW); digitalWrite(
Led_Kuning ,LOW);
digitalWrite(Led_Merah
,HIGH);digitalWrite (Buzzer,HIGH);}
if (temptrafo>116){digitalWrite(
Led_Hijau ,LOW); digitalWrite(
Led_Kuning ,LOW);
digitalWrite(Led_Merah
,HIGH);digitalWrite (Buzzer,HIGH);}
if (tempgenset>116){digitalWrite(
Led_Hijau ,LOW); digitalWrite(
Led_Kuning ,LOW);
digitalWrite(Led_Merah
,HIGH);digitalWrite (Buzzer,HIGH);}
if (tempkubikel>116){digitalWrite(
Led_Hijau ,LOW); digitalWrite(
Led_Kuning ,LOW);
digitalWrite(Led_Merah
,HIGH);digitalWrite (Buzzer,HIGH);}
   lcd.setCursor(6,0);
lcd.print("IR:");
lcd.print(ImpsRMSr);
lcd.print("  ");
lcd.setCursor(6,1);
lcd.print("IS:");
lcd.print(ImpsRMSs);
lcd.print("  ");
lcd.setCursor(6,2);
lcd.print("IT:");
lcd.print(ImpsRMSt);
lcd.print("  ");
 lcd.setCursor(12,0);
lcd.print("r:");
lcd.print(temptrafo);
lcd.setCursor(12,1);
lcd.print("s:");
lcd.print(tempgenset);
lcd.setCursor(12,2);
lcd.print("t:");
lcd.print(tempkubikel);
Serial.print("a");
Serial.println(AmpsRMSr);
Serial.print("b");
Serial.println(AmpsRMSs);
Serial.print("c");
Serial.println(AmpsRMSt);
Serial.print("d");
Serial.println(ImpsRMSr);
Serial.print("e");
Serial.println(ImpsRMSs);
Serial.print("f");
Serial.println(ImpsRMSt);
Serial.print("g");
Serial.println(temptrafo);
Serial.print("h");
Serial.println(tempgenset);
Serial.print("i");
Serial.println(tempkubikel);
 lcd.setCursor(0,3);
 lcd.print("  Gardu Distribusi  ");
 }
void dogetVPP()

{
  int maxValuer= 0;          // store max
value here
  int minValuer = 1024;          // store min
value here
  int maxValues= 0;          // store max
value here
```

```
  int minValues= 1024;          // store min
value here
   int maxValuet= 0;          // store max
value here
  int minValuet = 1024;          // store min
value here
  uint32_t start_time = millis();
   while((millis()-start_time) < 1000)
//sample for 1 Sec
   {
    readValuer = analogRead(sensorInr);
    readValues = analogRead(sensorIns);
    readValuet = analogRead(sensorInt);
     // see if you have a new maxValue
     if (readValuer > maxValuer)
     {
        /*record the maximum sensor
value*/
        maxValuer = readValuer;
     }
     if (readValuer < minValuer)
     {
        /*record the maximum sensor
value*/
        minValuer = readValuer;
     }
//"'
     if (readValues > maxValues)
     {
        /*record the maximum sensor
value*/
        maxValues = readValues;
     }
     if (readValues < minValues)
     {
        /*record the maximum sensor
value*/
        minValues = readValues;
     }
//""'
 if (readValuet > maxValuet)
     {
        /*record the maximum sensor
value*/
        maxValuet = readValuet;
     }
     if (readValuet < minValuet)
     {
        /*record the maximum sensor
value*/
        minValuet = readValuet;
     }
 //
   }
    // Subtract min from max
   resultr = maxValuer/0.554;
   results = maxValues/0.505;
   resultt = maxValuet/0.52;
// see if you have a new maxValue
void dogetIPP()
 {
 int maxValueir= 0;          // store max
value here
  int minValueir = 1024;          // store min
value here
   int maxValueis= 0;          // store max
value here
  int minValueis = 1024;          // store min
value here
  int maxValueit= 0;          // store max
value here
  int minValueit = 1024;          // store min
value here
   uint32_t start_time = millis();
   while((millis()-start_time) < 1000)
//sample for 1 Sec
   {
      readValueir =
analogRead(sensorInir);
   readValueir = readValueir-43;
      if( readValueir <= 0){readValueir =
0;}
      // see if you have a new maxValue
      if (readValueir > maxValueir)
      {
         /*record the maximum sensor
value*/
         maxValueir = readValueir;
      }
      if (readValueir < minValueir)
      {
```

```
        /*record the maximum sensor
value*/
        minValueir = readValueir;
      }
      readValueis =
analogRead(sensorInis);
      // see if you have a new maxValue
        readValueis = readValueis-43;
      if( readValueis <= 0){readValueis =
0;}
          if (readValueis > maxValueis)
      {
        /*record the maximum sensor
value*/
        maxValueis = readValueis;
      }
      if (readValueis < minValueis)
      {
        /*record the maximum sensor
value*/
        minValueis = readValueis;
      }
 readValueit = analogRead(sensorInit);
      // see if you have a new maxValue
        readValueit = readValueit-43;
      if( readValueit <= 0){readValueit =
0;}

      if (readValueit > maxValueit)
      {
        /*record the maximum sensor
value*/
        maxValueit = readValueit;
      }
      if (readValueit < minValueit)
      {
        /*record the maximum sensor
value*/
        minValueit = readValueit;
      }
}
  // Subtract min from max
  resultir = maxValueir/0.75;
 resultis = maxValueis/1.9;
resultit = maxValueit/1.46;
}
```