LAMPIRAN I
ALAT DAN BAHAN

Solar Panel

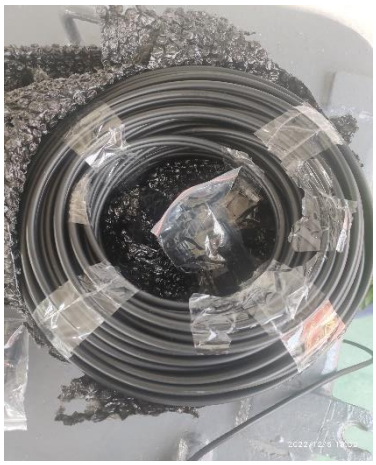
Baterai

Inverter

Box Panel

MCB DC, MCB AC, SPPD

MC4 Connector 2 in 1



Kabel PV

Tang Ampere

Multimeter

Solar power Meter

Mesin Bor

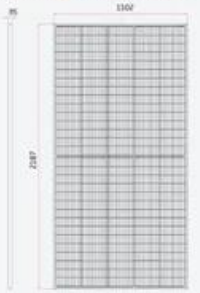
Gerinda

LAMPIRAN II
SPESIFIKASI SOLAR PANEL

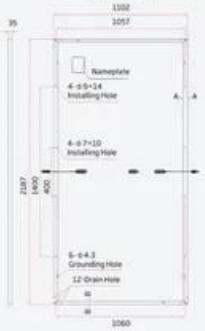
Vertex

BACKSHEET MONOCRYSTALLINE MODULE

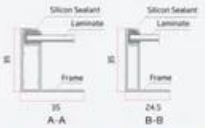
DIMENSIONS OF PV MODULE(mm)



Front View

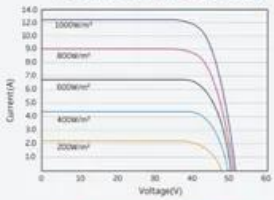


Back View

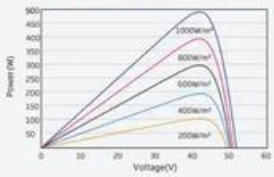


A-A B-B

I-V CURVES OF PV MODULE(495 W)



P-V CURVES OF PV MODULE(495W)



ELECTRICAL DATA (STC)

Peak Power Watts-P_{MAX} (Wp)*	485	490	495	500	505	510
Power Tolerance-P_{MAX} (W)	0 ~ +5					
Maximum Power Voltage-V_{MPP} (V)	42.2	42.4	42.6	42.8	43.0	43.2
Maximum Power Current-I_{MPP} (A)	11.49	11.56	11.63	11.69	11.75	11.81
Open Circuit Voltage-V_{OC} (V)	51.1	51.3	51.5	51.7	51.9	52.1
Short Circuit Current-I_{SC} (A)	12.07	12.14	12.21	12.28	12.35	12.42
Module Efficiency η_m (%)	20.1	20.3	20.5	20.7	21.0	21.2

STC: Irradiance 1000W/m², Cell Temperature 25°C, Air Mass AM1.5.
*Measuring tolerance: ±3%.

ELECTRICAL DATA (NOCT)

Maximum Power-P_{MAX} (Wp)	365	369	373	377	381	385
Maximum Power Voltage-V_{MPP} (V)	39.9	40.0	40.2	40.4	40.6	40.5
Maximum Power Current-I_{MPP} (A)	9.17	9.22	9.28	9.33	9.38	9.50
Open Circuit Voltage-V_{OC} (V)	48.1	48.2	48.4	48.6	48.8	49.0
Short Circuit Current-I_{SC} (A)	9.73	9.78	9.84	9.90	9.95	10.01

NOCT: Irradiance at 800W/m², Ambient Temperature 20°C, Wind Speed 1m/s.

MECHANICAL DATA

Solar Cells	Monocrystalline
Cell Orientation	150 cells
Module Dimensions	2187×1102×35 mm (86.10×43.39×1.38 inches)
Weight	26.5 kg (58.4 lb)
Glass	3.2 mm (0.13 inches), High Transmission, AR Coated Heat Strengthened Glass
Encapsulant Material	EVA
Backsheet	White
Frame	35 mm (1.38 inches) Anodized Aluminium Alloy
J-Box	IP 68 rated
Cables	Photovoltaic Technology Cable 4.0mm² (0.006 inches²), Portrait: N 280mm/P 280mm(11.02/11.02inches) Landscape: N 1400 mm /P 1400 mm (55.12/55.12 inches)
Connector	TS4

TEMPERATURE RATINGS

NOCT (Nominal Operating Cell Temperature)	43°C (±2°C)
Temperature Coefficient of P_{MAX}	- 0.34%/ C
Temperature Coefficient of V_{OC}	- 0.25%/ C
Temperature Coefficient of I_{SC}	0.04%/ C

MAXIMUMRATINGS

Operational Temperature	-40~+85 C
Maximum System Voltage	1500V DC (IEC)
Max Series Fuse Rating	20A

(Do not connect Fuse in Combiner Box with two or more strings in parallel connection)

WARRANTY

12 year Product Workmanship Warranty
25 year Power Warranty
2% first year degradation
0.55% Annual Power Attenuation

(Please refer to product warranty for details)

PACKAGING CONFIGUREATION

Modules per box: 31 pieces
Modules per 40' container: 620 pieces

Trinasolar

CAUTION: READ SAFETY AND INSTALLATION INSTRUCTIONS BEFORE USING THE PRODUCT.

Version number: TSM_EN_2021_APAC_A www.trinasolar.com

LAMPIRAN III
DOKUMENTASI PEMASANGAN SOLAR PANEL

Persiapan Pemasangan

Pemotongan Rangka

Pengeboran Solar Panel

Pemasangan Solar Panel Ke Rangka

Pemasangan Box Panel dan Inverter

Penyambungan baterai dengan inverter

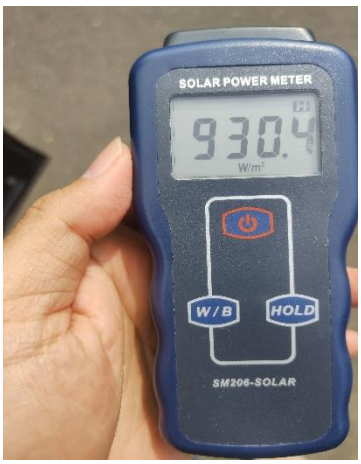
LAMPIRAN IV
PENGUKURAN *IRRADIACE*

Pengukuran Tgl 08. 01.2023

Pengukuran Tgl 08. 01.2023

Pengukuran Tgl 09. 01.2023

Pengukuran Tgl 09. 01.2023



Pengukuran Tgl 10. 01.2023

Pengukuran Tgl 09. 01.2023

LAMPIRAN V
CODING IOT

```
#include <EmonLib.h>
#include <SoftwareSerial.h>

int rx = 10; //Serial Port RX values.
int tx = 8;  //Ports are linked to the RX line.
bool debug = true;
String data;

EnergyMonitor emon1;
SoftwareSerial mySerial = SoftwareSerial(rx, tx);

void primeSerialPort()
{
  digitalWrite(rx, LOW);//Read on web that this is a good idea.
  digitalWrite(tx, LOW);
  delay(50);
  digitalWrite(rx, HIGH);//Read on web that this is a good idea.
  digitalWrite(tx, HIGH);//Read on web that this is a good idea.
  mySerial.listen(); //Activate the serial port.
  mySerial.flush(); //Clear the Serial port.
}

void setup() {
  emon1.current(A1, 55.5);
  emon1.voltage(A0, 377, 1.8);
  Serial.begin(9600);
  mySerial.begin(9600);
  Serial.println("Port RX/TX Ready to Use");
  primeSerialPort();
}

void loop() {
  if(mySerial.available()) {
    emon1.calcVI(20, 2000);
    int vrms = emon1.Vrms;
    float irms = emon1.Irms;
    float watt = emon1.realPower;
    float va = emon1.apparentPower;
    float pf = emon1.powerFactor;
    data = "";
    data = String(vrms);
```

```
    data += ",";
    data += String(irms);
    data += ",";
    data += String(watt);
    data += ",";
    data += String(va);
    data += ",";
    data += String(pf);

    if(debug == true){
      Serial.print(">>> Received: ");
      Serial.println(mySerial.read());
      Serial.println("Now Sending >>>");
      Serial.print("Data : ");
      Serial.println(data);
      Serial.println("-----------------------------");
    }
    //delay(5000);
    mySerial.print(data);
    primeSerialPort();
  }
}
//This code is tested and running using WeMOS D1 R1 Board
#define BLYNK_TEMPLATE_ID "fill with your template ID"
#define BLYNK_DEVICE_NAME "fill with your device name"
#define BLYNK_PRINT Serial
#define MAX485_DE D13 //for EPEVER Modbus
#define MAX485_RE D12 //for EPEVER Modbus
#define rx D11 //for communication between arduino
#define tx D10 //for communication between arduino
#include <ModbusMaster.h>
#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>
#include <SoftwareSerial.h>

boolean debug = true; //for troubleshooting
boolean enableTRXData = true; //if you dont want to communicate with other arduino board
float vrms, irms, watt, va, pf;

SoftwareSerial mySerial(rx, tx);

ModbusMaster epever;
BlynkTimer timer;
```

```
void primeSerialPort() { //for communicate between arduino board
  digitalWrite(rx, HIGH); //Read on web that this is a good idea.
  digitalWrite(tx, HIGH); //Read on web that this is a good idea.
  mySerial.listen(); //Activate the serial port.
  mySerial.flush(); //Clear the Serial port.
}

void preTransmission() { //for EPEVER Modbus
  digitalWrite(MAX485_RE, 1);
  digitalWrite(MAX485_DE, 1);
}

void postTransmission() { //for EPEVER Modbus
  digitalWrite(MAX485_RE, 0);
  digitalWrite(MAX485_DE, 0);
}

void setup() {
  pinMode(MAX485_RE, OUTPUT);
  pinMode(MAX485_DE, OUTPUT);
  digitalWrite(MAX485_RE, 0);
  digitalWrite(MAX485_DE, 0);
  digitalWrite(rx, 0);
  digitalWrite(tx, 0);
  Serial.begin(115200);
  mySerial.begin(9600);
  epever.begin(1, Serial);

  epever.preTransmission(preTransmission);
  epever.postTransmission(postTransmission);
  Blynk.begin("fill with your auth", "fill with your wifi ssid", "fill with your wifi
password");
  timer.setInterval(10000, GetData);
}

void GetData() {
  uint8_t energyMeterResult, tempResult, generateEnergyResult, statusLoadResult;

  if (Blynk.connected()) {

    epever.clearResponseBuffer();
    energyMeterResult = epever.readInputRegisters(0x3100, 7);

    if (energyMeterResult == epever.ku8MBSuccess) {
      Blynk.virtualWrite(V0, epever.getResponseBuffer(0x00) / 100.0f); //pv voltage
```

```
    Blynk.virtualWrite(V1, epever.getResponseBuffer(0x01) / 100.0f); //pv current
    Blynk.virtualWrite(V2, epever.getResponseBuffer(0x04) / 100.0f); //bat voltage
    Blynk.virtualWrite(V3, epever.getResponseBuffer(0x05) / 100.0f);//bat current
    Blynk.virtualWrite(V4, (epever.getResponseBuffer(0x04) / 100.0f) *
(epever.getResponseBuffer(0x05) / 100.0f)); //ba watt
    Blynk.virtualWrite(V5, ((epever.getResponseBuffer(0x00) / 100.0f) *
(epever.getResponseBuffer(0x01) / 100.0f)) / ((epever.getResponseBuffer(0x04) /
100.0f) * (epever.getResponseBuffer(0x05) / 100.0f)) * 100.0f); //mppt efeciency
  }
  else {
    Serial.print("Miss read energyMeterResult, ret val:");
    Serial.println(energyMeterResult);
  }

  delay(500);
  epever.clearResponseBuffer();
  energyMeterResult = epever.readInputRegisters(0x3110, 3);

  if (tempResult == epever.ku8MBSuccess) {
    Blynk.virtualWrite(V13, epever.getResponseBuffer(0x00) / 100.0f); //batt temp
    Blynk.virtualWrite(V11, epever.getResponseBuffer(0x01) / 100.0f); //mppt temp
  }
  else {
    Serial.print("Miss read tempResult, ret val:");
    Serial.println(tempResult);
  }

  delay(500);
  epever.clearResponseBuffer();
  generateEnergyResult = epever.readInputRegisters(0x3312, 1);

  if (generateEnergyResult == epever.ku8MBSuccess) {
    Blynk.virtualWrite(V8, epever.getResponseBuffer(0x00) / 100.0f);
  }
  else {
    Serial.print("Miss read generateEnergyResult, ret val:");
    Serial.println(generateEnergyResult);
  }

  delay(500);
  epever.clearResponseBuffer();
  statusLoadResult = epever.readInputRegisters(0x310C, 1);

  if (statusLoadResult == epever.ku8MBSuccess) {
    if ( (epever.getResponseBuffer(0x00) / 100.0f) > 0) {
```

```
      Blynk.virtualWrite(V10, "Running");
     }
     else {
      Blynk.virtualWrite(V10, "Shutdown");
     }
    }
    else {
     Serial.print("Miss read statusLoadResult, ret val:");
     Serial.println(statusLoadResult);
    }
    epever.clearResponseBuffer();
    if(enableTRXData == true) {
     primeSerialPort();
     GetDataArduinoNeighbour();
     Blynk.virtualWrite(V14, vrms); //inverter voltage
     Blynk.virtualWrite(V15, irms); //inverter current
     Blynk.virtualWrite(V16, watt); //inverter watt
     Blynk.virtualWrite(V17, pf); //inverter power factor
    }
   }
}

void GetDataArduinoNeighbour() {
  mySerial.print(1);
  if(debug == true) Serial.println("Request to send status from outside");
  delay(50);

  if(mySerial.available()) {
    String dataReceived = mySerial.readString();
    int ind1 = dataReceived.indexOf(',');
    String vrmsString = dataReceived.substring(0, ind1);
    vrms = vrmsString.toFloat();
    int ind2 = dataReceived.indexOf(',', ind1+1 );
    String irmsString = dataReceived.substring(ind1+1, ind2);
    irms = irmsString.toFloat();
    int ind3 = dataReceived.indexOf(',', ind2+1 );
    String wattString = dataReceived.substring(ind2+1, ind3);
    watt = wattString.toFloat();
    int ind4 = dataReceived.indexOf(',', ind3+1 );
    String vaString = dataReceived.substring(ind3+1, ind4);
    va = vaString.toFloat();
    int ind5 = dataReceived.indexOf(',', ind4+1 );
    String pfString = dataReceived.substring(ind4+1, ind5);
    pf = pfString.toFloat();
```

```
    if(debug == true) {
      Serial.println("----------------");
      Serial.println("Now Receiving");
      Serial.print("VRMS : ");
      Serial.println(vrms);
      Serial.print("IRMS : ");
      Serial.println(irms);
      Serial.print("WATT : ");
      Serial.println(watt);
      Serial.print("VA   : ");
      Serial.println(va);
      Serial.print("PF   : ");
      Serial.println(pf);
    }
  }

  if(debug == true) {
    Serial.println("");
    Serial.println("#### FINISH ####");
    Serial.println("");
  }
  primeSerialPort();
}

void loop() {
  Blynk.run();
  timer.run();
}
```