## LAMPIRAN

```
#include <SPI.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#include "EEPROM.h"
#include <esp_now.h>

//#include "EmonLib.h"
#if defined(ARDUINO) && ARDUINO >= 100

#include "Arduino.h"

#else

#include "WProgram.h"

#endif

// define theoretical vref calibration constant for use in
readvcc()
// 1100mV*1024 ADC steps
http://openenergymonitor.org/emon/node/1186
// override in your code with value for your specific AVR chip
// determined by procedure described under "Calibrating the
internal reference voltage" at
// http://openenergymonitor.org/emon/buildingblocks/calibration
#ifndef READVCC_CALIBRATION_CONST
#define READVCC_CALIBRATION_CONST 1126400L
#endif

// to enable 12-bit ADC resolution on Arduino Due,
// include the following line in main sketch inside setup()
function:
//  analogReadResolution(ADC_BITS);
// otherwise will default to 10 bits, as in regular Arduino-based
boards.
#if defined(__arm__)
#define ADC_BITS    12
#else
#define ADC_BITS    10
#endif

#define ADC_COUNTS  (1<<ADC_BITS)

#define BLYNK_PRINT Serial
#include <WiFi.h>
#include <WiFiClient.h>
#include <BlynkSimpleEsp32.h>

// You should get Auth Token in the Blynk App.
// Go to the Project Settings (nut icon).
char auth[] = "5a7c8acbe0fe4295a632f870247dcdca";
```

```
// Your WiFi credentials.
// Set password to "" for open networks.
char ssid[] = "Wifi Rumah";
char pass[] = "bigpond4g";

class EnergyMonitor
{
  public:

    void voltage(unsigned int _inPinV, double _VCAL, double
_PHASECAL);
    void current(unsigned int _inPinI, double _ICAL);

    void voltageTX(double _VCAL, double _PHASECAL);
    void currentTX(unsigned int _channel, double _ICAL);

    void calcVI(unsigned int crossings, unsigned int timeout);
    double calcIrms(unsigned int NUMBER_OF_SAMPLES);
    void serialprint();

    long readVcc();
    //Useful value variables
    double realPower,
      apparentPower,
      powerFactor,
      Vrms,
      Irms;

  private:

    //Set Voltage and current input pins
    unsigned int inPinV;
    unsigned int inPinI;
    //Calibration coefficients
    //These need to be set in order to obtain accurate results
    double VCAL;
    double ICAL;
    double PHASECAL;

    //--------------------------------------------------------------
-------------------------
    // Variable declaration for emon_calc procedure
    //--------------------------------------------------------------
-------------------------
    int sampleV;                        //sample_ holds the raw
analog read value
    int sampleI;

    double lastFilteredV,filteredV;          //Filtered_ is the
raw analog value minus the DC offset
    double filteredI;
    double offsetV;                          //Low-pass filter
output
    double offsetI;                          //Low-pass filter
output
```

```
    double phaseShiftedV;                                //Holds the
calibrated phase shifted voltage.

    double sqV,sumV,sqI,sumI,instP,sumP;                 //sq =
squared, sum = Sum, inst = instantaneous

    int
startV;                                             //Instantaneous
voltage at start of sample window.

    boolean lastVCross, checkVCross;                     //Used to
measure number of times threshold is crossed.


};


EnergyMonitor emon1;

//lcd
#define OLED_RESET 4
Adafruit_SSD1306 display(OLED_RESET);

//i2c
#define I2C_SDA_PIN 22
#define I2C_SCL_PIN 23
#define AC_pin 25

//tombol
#define sw4 5
#define sw3 17
#define sw2 16
#define sw1 4
#define EEPROM_SIZE 64

//esp now
#define NUMSLAVES 20
esp_now_peer_info_t slaves[NUMSLAVES] = {};
int SlaveCnt = 0;

#define CHANNEL_MASTER 3
#define CHANNEL_SLAVE 1
#define PRINTSCANRESULTS 0
#define DATASIZE 48
#define LED_PIN 2

//Power Monitor
//uint8_t ledR = 25;
int whaddress = 0,dimadress = 7;
int intr;
int value = 0;
int nilteg,nilarus,voltageA,niljarak,niljarak1,jarak,jarak1;
double arus,watthour,wht;
unsigned int baseAddr = 0;

float Irms=0,supplyVoltage=0,powerFActor,apparentPower,realPower;
```

```
volatile int i=0;
volatile boolean zero_cross=0;
int dim = 0, benda;      //192              // Dimming level (0-
100)  0 = on, 100 = 0ff
volatile boolean ben=0,ben1=0;
int setdim=75,setdim1=75;
int inc=1;                       // counting up or down, 1=up, -
1=down
//int freqStep = 75;
float energiter[5];
long energyall;
int node1;
int range=50;
int range1=10;

char databuff[DATASIZE],wktubuff[32];
//void interupt
void IRAM_ATTR zero_crosss_int()
{
 zero_cross = true;               // set the boolean to true to
tell our dimming function that a zero cross has occured
  i=0;

  digitalWrite(AC_pin, LOW);
}

//void intteruptimmer
volatile int Counter;
int detik,menit,jam,dtk,bendt,mnt;
volatile boolean det=0,jm=0;
volatile int timzero;
int totalInterruptCounter;
hw_timer_t * timer = NULL;
portMUX_TYPE timerMux = portMUX_INITIALIZER_UNLOCKED;
void IRAM_ATTR onTimer() {
  portENTER_CRITICAL_ISR(&timerMux);
  Counter++;

  if(Counter>=10000){
    Counter=0;detik++;dtk++;bendt++;mnt++;
    if(detik>=60){menit++;detik=0;}
    if(menit>=60){jam++;menit=0;}

    det=true;
    }
  if(zero_cross == true) {
    if(i>=dim) {
                digitalWrite(AC_pin, HIGH); // turn on
light
                i=0;  // reset time step
counter
                zero_cross = false; //reset zero cross detection
                }
    else {
          i++; // increment time step counter
```

```
          }
      }
  portEXIT_CRITICAL_ISR(&timerMux);
}




void setup() {

Serial.begin(115200);



Wire.begin( I2C_SDA_PIN, I2C_SCL_PIN );
display.begin(SSD1306_SWITCHCAPVCC, 0x3C);
analogReadResolution(10);
//init timer
timer = timerBegin(0, 80, true);
timerAttachInterrupt(timer, &onTimer, true);
timerAlarmWrite(timer, 100, true);
timerAlarmEnable(timer);


//init display
display.clearDisplay();
display.setTextSize(1);
display.setTextColor(WHITE);
display.setCursor(0,0);

display.println("Hello, Faizal A A");
display.println("MTE.178");
display.println("Nice To Meet You");
display.display();
delay(500);

//
if (!EEPROM.begin(64))
  {
    display.clearDisplay();
    display.setCursor(0,0);
    display.println("Failed to initialise EEPROM");
    display.println("Restarting...");
    display.display();
    delay(1000);
    //ESP.restart();
  }

//ledcAttachPin(ledR, 1);
//ledcSetup(1, 10000, 8);
pinMode(sw1,INPUT_PULLUP);
pinMode(sw2,INPUT_PULLUP);
pinMode(sw3,INPUT_PULLUP);
pinMode(sw4,INPUT_PULLUP);
```

```
pinMode(AC_pin, OUTPUT);
attachInterrupt(18, zero_crosss_int, RISING);

emon1.voltage(33, 257.8, 1.7);  // Voltage: input pin,
calibration, phase_shift 263.8

emon1.current(32, 25.38);//25.38 //21.24
//emon1.current(33, 0.35);
display.clearDisplay();
zero_cross = false;
det=false;


 pinMode(LED_PIN, OUTPUT);

  //Set device in STA mode to begin with
  display.setCursor(0,0);
  WiFi.mode(WIFI_MODE_APSTA);
   Blynk.begin(auth, ssid, pass);

  display.println("ESPNow/Multi-Slave/Master Example");
  display.display();

  // configure device AP mode
  configDeviceAP();
  // This is the mac address of the Master in Station Mode
  display.print("STA MAC: "); display.println(WiFi.macAddress());
  display.print("AP MAC: ");
display.println(WiFi.softAPmacAddress());
    display.display();
    //delay(500);
  // Init ESPNow with a fallback logic

  InitESPNow();
  // Once ESPNow is successfully Init, we will register for Send
CB to
  // get the status of Trasnmitted packet
  esp_now_register_send_cb(OnDataSent);
  esp_now_register_recv_cb(OnDataRecv);
  display.clearDisplay();

  wht = EEPROM.readDouble(whaddress);
  //setdim = EEPROM.readInt(dimadress);
   display.setCursor(0,0);
   display.print(wht);display.println(" wh");
   display.print(setdim);display.println(" dim");
   display.display();
   dim=0;
   delay(500);
   display.clearDisplay();
   watthour=wht;
   setdim1=setdim;
```

```
}

void loop() {
  // put your main code here, to run repeatedly:

emonlb();

logikajarak();
if(C()){
        display.clearDisplay();
        watthour=0;
        EEPROM.writeDouble(whaddress,0);
        EEPROM.commit();
        display.setCursor(0,0);
        display.print(watthour);display.println(" wh");
        display.display();
        delay(500);
        display.clearDisplay();


        }
if (U()){
        display.clearDisplay();
        setdim1=setdim1+5;
        if(setdim1>100){setdim1=0;}
        if(setdim1<0){setdim1=100;}
        display.setCursor(0,0);
        display.print(setdim1);display.println(" ");
        display.display();
        delay(300);
        display.clearDisplay();
        setdim=setdim1;


}
else if (D()){
        display.clearDisplay();
        setdim1=setdim1-5;
        if(setdim1>100){setdim1=0;}
        if(setdim1<0){setdim1=100;}
        display.setCursor(0,0);
        display.print(setdim1);display.println(" ");
        display.display();
        delay(300);
        display.clearDisplay();
        setdim=setdim1;


}



  display.clearDisplay();
  display.setCursor(0,0);
  display.print(Irms);display.print(" A ");
  display.print(supplyVoltage);display.println(" V ");
```

```
  display.print(jarak);display.print(" cm
");display.print(jarak1);display.print(" cm
");display.print(benda);display.println(" B");
  display.print(realPower);display.print(" W ");
  display.print(wht);display.println(" wH");
  display.print(detik);display.print("s
");display.print(menit);display.print("m
");display.print(jam);display.print("j ");
  //display.print((char *)databuff);
  display.display();

wh();




}

void readadc(){

  niljarak1=analogRead(34);
  niljarak=analogRead(35);
  //niljarak= map(niljarak1,0,4095,0,1023);

  jarak=28250/(niljarak-229.5);
  //jarak1=4800/(niljarak1-20);
  jarak1=2076/(niljarak1 - 11);

}

void logikajarak(){
  readadc();
  if
(jarak1>=range1&&jarak>=range&&bendt>1){dim=setdim1;ben1=true;}
  else if (jarak<range){dim=0;bendt=0;ben=true;}
  else if(jarak1<range1){dim=0;bendt=0;ben=true;}


  if(ben==true&&ben1==true&&det==true){benda++;ben=ben1=false;}
}


void wh() {
  if(det==true){
  watthour+=realPower/3600;
  wht=watthour;

  EEPROM.writeDouble( whaddress, wht);
  EEPROM.commit();


  if (dtk>=10){espnowgetdata();dtk=0;}
```

```
  det=false;
  }
   Blynk.run();
   sprintf(wktubuff,"%d:%d:%d",jam,menit,detik);
   Blynk.virtualWrite(V3,wktubuff);
  inet_blynk();

}

void inet_blynk(){
   if (mnt>11){



  energiter[1]=wht;
  energyall=energiter[1]+energiter[2]+energiter[3]+energiter[4];
 Blynk.virtualWrite(V0,energyall);
 Blynk.virtualWrite(V1,supplyVoltage);
 Blynk.virtualWrite(V2,Irms);
   mnt=0;
   }
}
bool U(){
 if(digitalRead(sw1)==LOW){return true;}
 else {return false;}
}
bool D(){
 if(digitalRead(sw2)==LOW){return true;}
 else {return false;}
}
bool C(){
 if(digitalRead(sw3)==LOW){return true;}
 else {return false;}
}
bool O(){
 if(digitalRead(sw4)==LOW){return true;}
 else {return false;}
}


void emonlb(){
  emon1.calcVI(20,2000);         // Calculate all. No.of half
wavelengths (crossings), time-out
  emon1.serialprint();

  realPower       = abs(emon1.realPower);        //extract Real
Power into variable
  apparentPower   = abs(emon1.apparentPower);    //extract
Apparent Power into variable
  powerFActor     = emon1.powerFactor;      //extract Power Factor
into Variable
   //supplyVoltage   =1.1387*emon1.Vrms-
30.936;             //extract Vrms into Variable
```

```
   //Irms            = 15.379*log(emon1.Irms)+0.052;
   Irms            = emon1.Irms;
   supplyVoltage   =emon1.Vrms;
}

void EnergyMonitor::voltage(unsigned int _inPinV, double _VCAL,
double _PHASECAL)
{
  inPinV = _inPinV;
  VCAL = _VCAL;
  PHASECAL = _PHASECAL;
  offsetV = ADC_COUNTS>>1;
}

void EnergyMonitor::current(unsigned int _inPinI, double _ICAL)
{
  inPinI = _inPinI;
  ICAL = _ICAL;
  offsetI = ADC_COUNTS>>1;
}

//-----------------------------------------------------------------
----------------------
// Sets the pins to be used for voltage and current sensors based
on emontx pin map
//-----------------------------------------------------------------
----------------------
void EnergyMonitor::voltageTX(double _VCAL, double _PHASECAL)
{
  inPinV = 2;
  VCAL = _VCAL;
  PHASECAL = _PHASECAL;
  offsetV = ADC_COUNTS>>1;
}

void EnergyMonitor::currentTX(unsigned int _channel, double _ICAL)
{
  if (_channel == 1) inPinI = 3;
  if (_channel == 2) inPinI = 0;
  if (_channel == 3) inPinI = 1;
  ICAL = _ICAL;
  offsetI = ADC_COUNTS>>1;
}

//-----------------------------------------------------------------
----------------------
// emon_calc procedure
// Calculates realPower,apparentPower,powerFactor,Vrms,Irms,kWh
increment
// From a sample window of the mains AC voltage and current.
// The Sample window length is defined by the number of half
wavelengths or crossings we choose to measure.
//-----------------------------------------------------------------
----------------------
void EnergyMonitor::calcVI(unsigned int crossings, unsigned int
timeout)
```

```
{
  #if defined emonTxV3
  int SupplyVoltage=3300;
  #else
  int SupplyVoltage = readVcc();
  #endif

  unsigned int crossCount = 0;                                //Used
to measure number of times threshold is crossed.
  unsigned int numberOfSamples = 0;                           //This
is now incremented

  //-------------------------------------------------------------------
-----------------------------------------------------------
  // 1) Waits for the waveform to be close to 'zero' (mid-scale
adc) part in sin curve.
  //-------------------------------------------------------------------
-----------------------------------------------------------
  boolean st=false;                                       //an
indicator to exit the while loop

  unsigned long start = millis();    //millis()-start makes sure
it doesnt get stuck in the loop if there is an error.

  while(st==false)                                        //the while
loop...
  {
    startV = analogRead(inPinV);                     //using the
voltage waveform
    if ((startV < (ADC_COUNTS*0.55)) && (startV >
(ADC_COUNTS*0.45))) st=true;  //check its within range
    if ((millis()-start)>timeout) st = true;
    logikajarak();
    wh();
  }

  //-------------------------------------------------------------------
-----------------------------------------------------------
  // 2) Main measurement loop
  //-------------------------------------------------------------------
-----------------------------------------------------------
  start = millis();

  while ((crossCount < crossings) && ((millis()-start)<timeout))
  {
    logikajarak();

    wh();
    numberOfSamples++;                          //Count number of
times looped.
    lastFilteredV = filteredV;                  //Used for
delay/phase compensation

    //-------------------------------------------------------------------
-----------------
    // A) Read in raw voltage and current samples
```

```
    //-------------------------------------------------------------
-----------------
    sampleV = analogRead(inPinV);                 //Read in raw
voltage signal
    sampleI = analogRead(inPinI);                 //Read in raw
current signal

    //-------------------------------------------------------------
-----------------
    // B) Apply digital low pass filters to extract the 2.5 V or
1.65 V dc offset,
    //     then subtract this - signal is now centred on 0 counts.
    //-------------------------------------------------------------
-----------------
    offsetV = offsetV + ((sampleV-offsetV)/1024);
    filteredV = sampleV - offsetV;
    offsetI = offsetI + ((sampleI-offsetI)/1024);
    filteredI = sampleI - offsetI;

    //-------------------------------------------------------------
-----------------
    // C) Root-mean-square method voltage
    //-------------------------------------------------------------
-----------------
    sqV= filteredV * filteredV;                 //1) square
voltage values
    sumV += sqV;                                //2) sum

    //-------------------------------------------------------------
-----------------
    // D) Root-mean-square method current
    //-------------------------------------------------------------
-----------------
    sqI = filteredI * filteredI;                //1) square
current values
    sumI += sqI;                                //2) sum

    //-------------------------------------------------------------
-----------------
    // E) Phase calibration
    //-------------------------------------------------------------
-----------------
    phaseShiftedV = lastFilteredV + PHASECAL * (filteredV -
lastFilteredV);

    //-------------------------------------------------------------
-----------------
    // F) Instantaneous power calc
    //-------------------------------------------------------------
-----------------
    instP = phaseShiftedV * filteredI;          //Instantaneous
Power
    sumP +=instP;                               //Sum

    //-------------------------------------------------------------
-----------------
```

```
    // G) Find the number of times the voltage has crossed the
initial voltage
    //    - every 2 crosses we will have sampled 1 wavelength
    //    - so this method allows us to sample an integer number
of half wavelengths which increases accuracy
    //-------------------------------------------------------------
-----------------
    lastVCross = checkVCross;
    if (sampleV > startV) checkVCross = true;
                    else checkVCross = false;
    if (numberOfSamples==1) lastVCross = checkVCross;

    if (lastVCross != checkVCross) crossCount++;
  }

  //-------------------------------------------------------------
-------------------------------------------------------------
  // 3) Post loop calculations
  //-------------------------------------------------------------
-------------------------------------------------------------
  //Calculation of the root of the mean of the voltage and current
squared (rms)
  //Calibration coefficients applied.

  double V_RATIO = VCAL *((SupplyVoltage/1000.0) / (ADC_COUNTS));
  Vrms = V_RATIO * sqrt(sumV / numberOfSamples);

  double I_RATIO = ICAL *((SupplyVoltage/1000.0) / (ADC_COUNTS));
  Irms = I_RATIO * sqrt(sumI / numberOfSamples);

  //Calculation power values
  realPower = V_RATIO * I_RATIO * sumP / numberOfSamples;
  apparentPower = Vrms * Irms;
  powerFactor=realPower / apparentPower;

  //Reset accumulators
  sumV = 0;
  sumI = 0;
  sumP = 0;

//-------------------------------------------------------------
----------------------
}

//-------------------------------------------------------------
----------------------
double EnergyMonitor::calcIrms(unsigned int Number_of_Samples)
{

  #if defined emonTxV3
    int SupplyVoltage=3300;
  #else
    int SupplyVoltage = readVcc();
  #endif
```

```
  for (unsigned int n = 0; n < Number_of_Samples; n++)
  {
    sampleI = analogRead(inPinI);

    // Digital low pass filter extracts the 2.5 V or 1.65 V dc
offset,
    //  then subtract this - signal is now centered on 0 counts.
    offsetI = (offsetI + (sampleI-offsetI)/1024);
    filteredI = sampleI - offsetI;

    // Root-mean-square method current
    // 1) square current values
    sqI = filteredI * filteredI;
    // 2) sum
    sumI += sqI;
  }

  double I_RATIO = ICAL *((SupplyVoltage/1000.0) / (ADC_COUNTS));
  Irms = I_RATIO * sqrt(sumI / Number_of_Samples);

  //Reset accumulators
  sumI = 0;
  //--------------------------------------------------------------------
-------------------------

  return Irms;
}

void EnergyMonitor::serialprint()
{
  Serial.print(realPower);
  Serial.print(' ');
  Serial.print(apparentPower);
  Serial.print(' ');
  Serial.print(Vrms);
  Serial.print(' ');
  Serial.print(Irms);
  Serial.print(' ');
  Serial.print(powerFactor);
  Serial.println(' ');

}

//thanks to http://hacking.majenko.co.uk/making-accurate-adc-
readings-on-arduino
//and Jérôme who alerted us to
http://provideyourown.com/2012/secret-arduino-voltmeter-measure-
battery-voltage/

long EnergyMonitor::readVcc() {
  long result;

  //not used on emonTx V3 - as Vcc is always 3.3V - eliminates
bandgap error and need for calibration
http://harizanov.com/2013/09/thoughts-on-avr-adc-accuracy/
```

```
  #if defined(__AVR_ATmega168__) || defined(__AVR_ATmega328__) ||
defined (__AVR_ATmega328P__)
  ADMUX = _BV(REFS0) | _BV(MUX3) | _BV(MUX2) | _BV(MUX1);
  #elif defined(__AVR_ATmega644__) || defined(__AVR_ATmega644P__)
|| defined(__AVR_ATmega1284__) || defined(__AVR_ATmega1284P__)
  ADMUX = _BV(REFS0) | _BV(MUX4) | _BV(MUX3) | _BV(MUX2) |
_BV(MUX1);
  #elif defined(__AVR_ATmega32U4__) || defined(__AVR_ATmega1280__)
|| defined(__AVR_ATmega2560__) || defined(__AVR_AT90USB1286__)
  ADMUX = _BV(REFS0) | _BV(MUX4) | _BV(MUX3) | _BV(MUX2) |
_BV(MUX1);
  ADCSRB &= ~_BV(MUX5);   // Without this the function always
returns -1 on the ATmega2560
http://openenergymonitor.org/emon/node/2253#comment-11432
  #elif defined (__AVR_ATtiny24__) || defined(__AVR_ATtiny44__) ||
defined(__AVR_ATtiny84__)
  ADMUX = _BV(MUX5) | _BV(MUX0);
  #elif defined (__AVR_ATtiny25__) || defined(__AVR_ATtiny45__) ||
defined(__AVR_ATtiny85__)
  ADMUX = _BV(MUX3) | _BV(MUX2);

  #endif


  #if defined(__AVR__)
  delay(2);                                        // Wait for
Vref to settle
  ADCSRA |= _BV(ADSC);                             // Convert
  while (bit_is_set(ADCSRA,ADSC));
  result = ADCL;
  result |= ADCH<<8;
  result = READVCC_CALIBRATION_CONST / result;  //1100mV*1024 ADC
steps http://openenergymonitor.org/emon/node/1186
  return result;
  #elif defined(__arm__)
  return (3300);                                  //Arduino Due
  #else
  return (3300);                                  //Guess that
other un-supported architectures will be running a 3.3V!
  #endif
}

void InitESPNow() {
  if (esp_now_init() == ESP_OK) {
    Serial.println("ESPNow Init Success");
  }
  else {
    Serial.println("ESPNow Init Failed");
    // Retry InitESPNow, add a counte and then restart?
    // InitESPNow();
    // or Simply Restart
    ESP.restart();
  }
}

// Scan for slaves in AP mode
```

```
void ScanForSlave() {

  int8_t scanResults = WiFi.scanNetworks();



  logikajarak();
  //reset slaves
  memset(slaves, 0, sizeof(slaves));
  SlaveCnt = 0;
  Serial.println("");
  if (scanResults == 0) {
    Serial.println("No WiFi devices in AP Mode found");
  } else {
    Serial.print("Found "); Serial.print(scanResults);
Serial.println(" devices ");
    for (int i = 0; i < scanResults; ++i) {
      // Print SSID and RSSI for each device found
      String SSID = WiFi.SSID(i);
      int32_t RSSI = WiFi.RSSI(i);
      String BSSIDstr = WiFi.BSSIDstr(i);

      if (PRINTSCANRESULTS) {
        Serial.print(i + 1); Serial.print(": ");
Serial.print(SSID); Serial.print(" ["); Serial.print(BSSIDstr);
Serial.print("]"); Serial.print(" ("); Serial.print(RSSI);
Serial.print(")"); Serial.println("");
      }
      //delay(10);
      // Check if the current device starts with `Slave`
      if (SSID.indexOf("ESPNOW") == 0) {
        // SSID of interest
        Serial.print(i + 1); Serial.print(": ");
Serial.print(SSID); Serial.print(" ["); Serial.print(BSSIDstr);
Serial.print("]"); Serial.print(" ("); Serial.print(RSSI);
Serial.print(")"); Serial.println("");
        // Get BSSID => Mac Address of the Slave
        int mac[6];

        if ( 6 == sscanf(BSSIDstr.c_str(),
"%02x:%02x:%02x:%02x:%02x:%02x",  &mac[0], &mac[1], &mac[2],
&mac[3], &mac[4], &mac[5] ) ) {
          for (int ii = 0; ii < 6; ++ii ) {
            slaves[SlaveCnt].peer_addr[ii] = (uint8_t) mac[ii];
          }
        }
        slaves[SlaveCnt].channel = CHANNEL_MASTER; // pick a
channel
        slaves[SlaveCnt].encrypt = 0; // no encryption
        SlaveCnt++;
      }
    }
  }

  if (SlaveCnt > 0) {
```

```
    Serial.print(SlaveCnt); Serial.println(" Slave(s) found,
processing..");
  } else {
    Serial.println("No Slave Found, trying again.");
  }

  // clean up ram
  WiFi.scanDelete();
}

// Check if the slave is already paired with the master.
// If not, pair the slave with master
void manageSlave() {
  if (SlaveCnt > 0) {
    for (int i = 0; i < SlaveCnt; i++) {
      const esp_now_peer_info_t *peer = &slaves[i];
      const uint8_t *peer_addr = slaves[i].peer_addr;
      Serial.print("Processing: ");
      for (int ii = 0; ii < 6; ++ii ) {
        Serial.print((uint8_t) slaves[i].peer_addr[ii], HEX);
        if (ii != 5) Serial.print(":");
      }
      Serial.print(" Status: ");
      // check if the peer exists
      bool exists = esp_now_is_peer_exist(peer_addr);
      if (exists) {
        // Slave already paired.
        Serial.println("Already Paired");
      } else {
        // Slave not paired, attempt pair
        esp_err_t addStatus = esp_now_add_peer(peer);
        if (addStatus == ESP_OK) {
          // Pair success
          Serial.println("Pair success");
        } else if (addStatus == ESP_ERR_ESPNOW_NOT_INIT) {
          // How did we get so far!!
          Serial.println("ESPNOW Not Init");
        } else if (addStatus == ESP_ERR_ESPNOW_ARG) {
          Serial.println("Add Peer - Invalid Argument");
        } else if (addStatus == ESP_ERR_ESPNOW_FULL) {
          Serial.println("Peer list full");
        } else if (addStatus == ESP_ERR_ESPNOW_NO_MEM) {
          Serial.println("Out of memory");
        } else if (addStatus == ESP_ERR_ESPNOW_EXIST) {
          Serial.println("Peer Exists");
        } else {
          Serial.println("Not sure what happened");
        }

      }
    }
  } else {
    // No slave found to process
    Serial.println("No Slave found to process");
  }
}
```

```
uint8_t data[DATASIZE];
uint64_t pos=0;

// send data
void sendData() {

  sprintf((char *)data,"1 %0.2f",wht);
  for (int i = 0; i < SlaveCnt; i++) {
    const uint8_t *peer_addr = slaves[i].peer_addr;
    if (i == 0) { // print only for first slave
      Serial.print("Sending: ");
      Serial.println((char *)data);
    }
    esp_err_t result = esp_now_send(peer_addr, data, DATASIZE);
    Serial.print("Send Status: ");
    if (result == ESP_OK) {
      Serial.println("Success");
    } else if (result == ESP_ERR_ESPNOW_NOT_INIT) {
      // How did we get so far!!
      Serial.println("ESPNOW not Init.");
    } else if (result == ESP_ERR_ESPNOW_ARG) {
      Serial.println("Invalid Argument");
    } else if (result == ESP_ERR_ESPNOW_INTERNAL) {
      Serial.println("Internal Error");
    } else if (result == ESP_ERR_ESPNOW_NO_MEM) {
      Serial.println("ESP_ERR_ESPNOW_NO_MEM");
    } else if (result == ESP_ERR_ESPNOW_NOT_FOUND) {
      Serial.println("Peer not found.");
    } else {
      Serial.println("Not sure what happened");
    }
    //delay(100);
  }
}

// callback when data is sent from Master to Slave
void OnDataSent(const uint8_t *mac_addr, esp_now_send_status_t
status) {
  char macStr[18];
  snprintf(macStr, sizeof(macStr),
"%02x:%02x:%02x:%02x:%02x:%02x",
           mac_addr[0], mac_addr[1], mac_addr[2], mac_addr[3],
mac_addr[4], mac_addr[5]);
  Serial.print("Last Packet Sent to: "); Serial.println(macStr);
  Serial.print("Last Packet Send Status: "); Serial.println(status
== ESP_NOW_SEND_SUCCESS ? "Delivery Success" : "Delivery Fail");
  logikajarak();
}

// callback when data is recv from Master
void OnDataRecv(const uint8_t *mac_addr, const uint8_t *data, int
data_len) {
  char macStr[18];
  char node[1];
```

```
  digitalWrite(LED_PIN, 1);
  snprintf(macStr, sizeof(macStr),
"%02x:%02x:%02x:%02x:%02x:%02x",
           mac_addr[0], mac_addr[1], mac_addr[2], mac_addr[3],
mac_addr[4], mac_addr[5]);
  Serial.print("\t\tLast Packet Recv from: ");
Serial.println(macStr);
  Serial.print("\t\tLast Packet Recv Data: ");
Serial.println((char *)data);
  Serial.println("");


     //node[0]=data[0];


  for (int i = 0; i < DATASIZE-1; i++){
     databuff[i]=data[i];
  }
  node1=atoi(&databuff[0]);
  energiter[node1]=atof(&databuff[2]);
  logikajarak();

}


// config AP SSID
void configDeviceAP() {
  String Prefix = "ESPNOW:";
  String Mac = WiFi.macAddress();
  String SSID = Prefix + Mac;
  String Password = "123456789";
  bool result = WiFi.softAP(SSID.c_str(), Password.c_str(),
CHANNEL_SLAVE, 0);
  if (!result) {
    Serial.println("AP Config failed.");
  } else {
    Serial.println("AP Config Success. Broadcasting with AP: " +
String(SSID));
  }
}



void espnowgetdata(){
   display.clearDisplay();
  display.setCursor(0,0);
  //display.print(wht);display.println("wH thisNode 3");
  for (node1 = 1; node1 < 5; node1++){
    display.print(node1);display.print(" node
");display.print(energiter[node1]);display.println(" wH");
  }

  display.display();

  ScanForSlave();
```

```
  logikajarak();
  // If Slave is found, it would be populate in `slave` variable
  // We will check if `slave` is defined and then we proceed
further
  if (SlaveCnt > 0) { // check if slave channel is defined
    // `slave` is defined
    // Add slave as peer if it has not been added already
    manageSlave();
    // pair success or already paired
    // Send data to device
    sendData();
    logikajarak();
  } else {
   logikajarak();
  }

}
```

YAYASAN BADAN WAKAF SULTAN AGUNG
UNIVERSITAS ISLAM SULTAN AGUNG
**PROGRAM STUDI MAGISTER (S2) TEKNIK ELEKTRO**
Jl. Raya Kaligawe Km. 4 PO BOX. 1054 Telp. (024) 6583584 Ext. 349 Semarang 50012
Email: mte@unissula.ac.id, website: http://mte.unissula.ac.id

# LEMBAR REVISI & TUGAS
## SEMINAR PROPOSAL TESIS

**Berdasarkan Rapat Tim Penilai SEMINAR PROPOSAL TESIS :**

| | | |
|---|---|---|
| **Hari** | : | **Jumat** |
| **Tanggal** | : | **3 Agustus 2018** |

**Memutuskan bahwa mahasiswa :**

| | | |
|---|---|---|
| **Nama** | : | **Faizal Aminuddin Aziz** |
| **NIM** | : | **MTE.16.17.0178** |
| **Judul Proposal Tesis** | : | **Implementasi Jaringan WSN (Wireless Sensor Network) Pada Penerangan Jalan Umum Dengan Sensor Jarak Sebagai Kontrol Untuk Penghematan Energi Listrik** |

**wajib melakukan perbaikan seperti tercantum dibawah ini:**

| NO | REVISI | BATAS REVISI |
|---|---|---|
| 1 | Judul | |
| 2 | Tujuan Penelitian | |
| 3 | Metodologi | |

| NO | TUGAS |
|---|---|
| | |

**Semarang, 3 Agustus 2018**

**Tim Penilai Seminar Proposal Tesis**

**Dr. Ir. H. Muhamad Haddin, MT**

YAYASAN BADAN WAKAF SULTAN AGUNG
UNIVERSITAS ISLAM SULTAN AGUNG
**PROGRAM STUDI MAGISTER (S2) TEKNIK ELEKTRO**
Jl. Raya Kaligawe Km. 4 PO BOX. 1054 Telp. (024) 6583584 Ext. 349 Semarang 50012
Email: mte@unissula.ac.id, website: http://mte.unissula.ac.id

## LEMBAR REVISI & TUGAS
### SEMINAR PROPOSAL TESIS

**Berdasarkan Rapat Tim Penilai SEMINAR PROPOSAL TESIS :**

Hari : Jumat
Tanggal : 3 Agustus 2018

**Memutuskan bahwa mahasiswa :**

Nama : Faizal Aminuddin Aziz
NIM : MTE.16.17.0178
Judul Proposal Tesis : Implementasi Jaringan WSN (Wireless Sensor Network) Pada Penerangan Jalan Umum Dengan Sensor Jarak Sebagai Kontrol Untuk Penghematan Energi Listrik

**wajib melakukan perbaikan seperti tercantum dibawah ini:**

| NO | REVISI | BATAS REVISI |
|---|---|---|
| 1. | Judul dirubah | Imuyan |
| 2. | Tujuan → protokol & routing<br>data → dirancang | |
| 3. | kesimpulan Awal yg membedakan<br>penelitian Anda dgn penelitian sebelum yg ! | |
| 4. | komunikasi mesh → Forward & decode. | |

06/08/2018

| NO | TUGAS |
|---|---|
| | |

Semarang, 3 Agustus 2018

Tim Penilai Seminar Proposal Tesis

Arief Marwanto, ST., M.Eng.,PhD

YAYASAN BADAN WAKAF SULTAN AGUNG
UNIVERSITAS ISLAM SULTAN AGUNG
**PROGRAM STUDI MAGISTER (S2) TEKNIK ELEKTRO**
Jl. Raya Kaligawe Km. 4 PO BOX. 1054 Telp. (024) 6583584 Ext. 349 Semarang 50012
Email: mte@unissula.ac.id, website: http://mte.unissula.ac.id

# LEMBAR REVISI & TUGAS
## SEMINAR PROPOSAL TESIS

**Berdasarkan Rapat Tim Penilai SEMINAR PROPOSAL TESIS :**

| | | |
|---|---|---|
| **Hari** | : | **Jumat** |
| **Tanggal** | : | **3 Agustus 2018** |

**Memutuskan bahwa mahasiswa :**

| | | |
|---|---|---|
| **Nama** | : | **Faizal Aminuddin Aziz** |
| **NIM** | : | **MTE.16.17.0178** |
| **Judul Proposal Tesis** | : | **Implementasi Jaringan WSN (Wireless Sensor Network) Pada Penerangan Jalan Umum Dengan Sensor Jarak Sebagai Kontrol Untuk Penghematan Energi Listrik** |

**wajib melakukan perbaikan seperti tercantum dibawah ini:**

| NO | REVISI | BATAS REVISI |
|---|---|---|
| | | |

| NO | TUGAS |
|---|---|
| . | Mekanisme kegagalan sensor node. 6/8/2018 |

**Semarang, 3 Agustus 2018**

**Tim Penilai Seminar Proposal Tesis**

**Ir. Suryani Alifah, MT, PhD**

YAYASAN BADAN WAKAF SULTAN AGUNG
UNIVERSITAS ISLAM SULTAN AGUNG
**PROGRAM STUDI MAGISTER (S2) TEKNIK ELEKTRO**
Jl. Raya Kaligawe Km. 4 PO BOX. 1054 Telp. (024) 6583584 Ext. 349 Semarang 50012
Email: mte@unissula.ac.id, website: http://mte.unissula.ac.id

# LEMBAR REVISI & TUGAS
## SEMINAR TESIS

**Berdasarkan Rapat Tim Penilai Seminar Tesis :**

**Hari** : **Jumat**
**Tanggal** : **14 September 2018**

**Memutuskan bahwa mahasiswa :**

**Nama** : **Faizal Aminuddin Aziz**
**NIM** : **MTE.16.17.0178**
**Judul Tesis** : **Efisiensi Energi PJU Menggunakan SMART Dimmer Berbasis Wireless Sensor Network**

***wajib melakukan perbaikan seperti tercantum dibawah ini:***

| NO | REVISI | BATAS REVISI |
|---|---|---|
| | — | |

| NO | TUGAS |
|---|---|
| | — |

Semarang, 14 September 2018

Tim Penilai Seminar

Imam Much Ibnu S, ST., M.Sc, Ph.D

YAYASAN BADAN WAKAF SULTAN AGUNG
UNIVERSITAS ISLAM SULTAN AGUNG
**PROGRAM STUDI MAGISTER (S2) TEKNIK ELEKTRO**
Jl. Raya Kaligawe Km. 4 PO BOX. 1054 Telp. (024) 6583584 Ext. 349 Semarang 50012
Email: mte@unissula.ac.id, website: http://mte.unissula.ac.id

# LEMBAR REVISI & TUGAS
## SEMINAR TESIS

**Berdasarkan Rapat Tim Penilai Seminar Tesis :**

**Hari** : **Jumat**
**Tanggal** : **14 September 2018**

**Memutuskan bahwa mahasiswa :**

**Nama** : **Faizal Aminuddin Aziz**
**NIM** : **MTE.16.17.0178**
**Judul Tesis** : **Efisiensi Energi PJU Menggunakan SMART Dimmer Berbasis Wireless Sensor Network**

***wajib melakukan perbaikan seperti tercantum dibawah ini:***

| NO | REVISI | BATAS REVISI |
|---|---|---|
| 1. | Listing Program dilampirkan saja | |
| 2. | Flow chart diperbaiki. | |
| 3. | Kesimpulan diperbaiki | |
| 4. | Publikasi ? | |

| NO | TUGAS |
|---|---|
| | |

Semarang, 14 September 2018

Dr. Ir. H. Muhamad Haddin, MT

YAYASAN BADAN WAKAF SULTAN AGUNG
UNIVERSITAS ISLAM SULTAN AGUNG
**PROGRAM STUDI MAGISTER (S2) TEKNIK ELEKTRO**
Jl. Raya Kaligawe Km. 4 PO BOX. 1054 Telp. (024) 6583584 Ext. 349 Semarang 50012
Email: mte@unissula.ac.id, website: http://mte.unissula.ac.id

# LEMBAR REVISI & TUGAS
## SEMINAR TESIS

**Berdasarkan Rapat Tim Penilai Seminar Tesis :**

**Hari** : **Jumat**
**Tanggal** : **14 September 2018**

**Memutuskan bahwa mahasiswa :**

**Nama** : **Faizal Aminuddin Aziz**
**NIM** : **MTE.16.17.0178**
**Judul Tesis** : **Efisiensi Energi PJU Menggunakan SMART Dimmer Berbasis Wireless Sensor Network**

**wajib melakukan perbaikan seperti tercantum dibawah ini:**

| NO | REVISI | BATAS REVISI |
|---|---|---|
| 1. | Tambahkan waktu total lama energi di Blynk<br>2. gambarkan aliran protokol | Acc 10/9/2018 |

| NO | TUGAS |
|---|---|
| | |

Semarang, 14 September 2018

Tim Penilai Seminar

Ir. Suryani Alifah, MT, Ph.D

YAYASAN BADAN WAKAF SULTAN AGUNG
UNIVERSITAS ISLAM SULTAN AGUNG
**PROGRAM STUDI MAGISTER (S2) TEKNIK ELEKTRO**
Jl. Raya Kaligawe Km. 4 PO BOX. 1054 Telp. (024) 6583584 Ext. 349 Semarang 50012
Email: mte@unissula.ac.id, website: http://mte.unissula.ac.id

# LEMBAR REVISI & TUGAS
## SEMINAR TESIS

**Berdasarkan Rapat Tim Penilai Seminar Tesis :**

**Hari** : **Jumat**
**Tanggal** : **14 September 2018**

**Memutuskan bahwa mahasiswa :**

**Nama** : **Faizal Aminuddin Aziz**
**NIM** : **MTE.16.17.0178**
**Judul Tesis** : **Efisiensi Energi PJU Menggunakan SMART Dimmer Berbasis Wireless Sensor Network**

***wajib melakukan perbaikan seperti tercantum dibawah ini:***

| NO | REVISI | BATAS REVISI |
|---|---|---|
| | | |

| NO | TUGAS |
|---|---|
| | |

**Semarang, 14 September 2018**

**Tim Penilai Seminar**

**Arief Marwanto, ST., M.Eng.,Ph.D**

YAYASAN BADAN WAKAF SULTAN AGUNG
UNIVERSITAS ISLAM SULTAN AGUNG
**PROGRAM STUDI MAGISTER (S2) TEKNIK ELEKTRO**
Jl. Raya Kaligawe Km. 4 PO BOX. 1054 Telp. (024) 6583584 Ext. 349 Semarang 50012
Email: mte@unissula.ac.id, website: http://mte.unissula.ac.id

# LEMBAR REVISI & TUGAS
## SEMINAR TESIS

**Berdasarkan Rapat Tim Penilai Seminar Tesis :**

**Hari** : **Jumat**
**Tanggal** : **14 September 2018**

**Memutuskan bahwa mahasiswa :**

**Nama** : **Faizal Aminuddin Aziz**
**NIM** : **MTE.16.17.0178**
**Judul Tesis** : **Efisiensi Energi PJU Menggunakan SMART Dimmer Berbasis Wireless Sensor Network**

***wajib melakukan perbaikan seperti tercantum dibawah ini:***

| NO | REVISI | BATAS REVISI |
|---|---|---|
| 1 | Abstrak di sempurnakan. | 2 hari saja. |
| 2. | listing program diletakkan di lampiran. | Acc. |
| 3. | Hal 63 berikan ulasan kenapa terjadi | 18/9/18 |

| NO | TUGAS | |
|---|---|---|
| | PWM & manfaat kegunaan & rumus (ada logika atau Rumus/Teorema yg mungkin bisa diterapkan ?) | Acc 18/9/18 |
| 4. | Kesimpulan di no 4 apakah perlu ? | |

Semarang, 14 September 2018

Dr. Hj. Sri Arttini Dwi P, M.Si

Acc
12/9/2018

# EFISIENSI ENERGI PJU MENGGUNAKAN SMART DIMMER BERBASIS WIRELESS SENSOR NETWORK

*by* Faizal Aminuddin Aziz

**Submission date:** 11-Sep-2018 08:06PM (UTC+0800)
**Submission ID:** 1000066662
**File name:** Laporan_Thesis.pdf (1.37M)
**Word count:** 12808
**Character count:** 84916

## EFISIENSI ENERGI PJU MENGGUNAKAN SMART DIMMER BERBASIS WIRELESS SENSOR NETWORK

ORIGINALITY REPORT

| 8% | 8% | 0% | 7% |
|---|---|---|---|
| SIMILARITY INDEX | INTERNET SOURCES | PUBLICATIONS | STUDENT PAPERS |

PRIMARY SOURCES

| | | |
|---|---|---|
| 1 | Submitted to Queensland University of Technology<br>Student Paper | 3% |
| 2 | lechacal.com<br>Internet Source | 3% |
| 3 | desire.giesecke.tk<br>Internet Source | 2% |

| | | | |
|---|---|---|---|
| Exclude quotes | On | Exclude matches | < 2% |
| Exclude bibliography | On | | |

Acc 19/8/2019.

turnitin

Acc

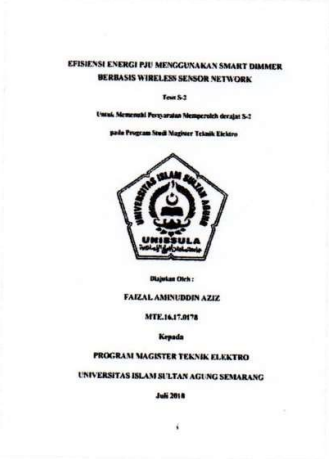
12/9/2018

# Digital Receipt

This receipt acknowledges that Turnitin received your paper. Below you will find the receipt information regarding your submission.

The first page of your submissions is displayed below.

| | |
|---|---|
| Submission author: | Faizal Aminuddin Aziz |
| Assignment title: | REPORT THESIS |
| Submission title: | EFISIENSI ENERGI PJU MENGGUN... |
| File name: | Laporan_Thesis.pdf |
| File size: | 1.37M |
| Page count: | 89 |
| Word count: | 12,808 |
| Character count: | 84,916 |
| Submission date: | 11-Sep-2018 08:06PM (UTC+0800) |
| Submission ID: | 1000066662 |



EFISIENSI ENERGI PJU MENGGUNAKAN SMART DIMMER BERBASIS WIRELESS SENSOR NETWORK

Tesis S-2

Untuk Memenuhi Persyaratan Memperoleh derajat S-2

pada Program Studi Magister Teknik Elektro

UNISSULA

Disjukan Oleh :

FAIZAL AMINUDDIN AZIZ

MTE.16.17.0178

Kepada

PROGRAM MAGISTER TEKNIK ELEKTRO

UNIVERSITAS ISLAM SULTAN AGUNG SEMARANG

Juli 2018