```
float watt_average;
int measuredvalue = 0;
float measuredvalue1 = 0;
float measuredvalue2 = 0;
float digstepV = 0.00080566;
int voltage = 23;  // 0.1 of the the net voltage if we have to divide by 10 and the multiply bij 230 we can do it by one
multiplication of 23

int AnalogPin = 20; //  = a5 the pin that is being used on the Olimexino-STM32  a0 = 15 a1 = 16 see the latest schematic
// https://www.olimex.com/Products/Duino/STM32/OLIMEXINO-STM32/resources/OLIMEXINO-
STM32_Rev_D.pdf


void setup()
{
  // Declare the sensorPin as INPUT_ANALOG:
   pinMode(AnalogPin, INPUT_ANALOG);

}

void measure_power()
{
 int n = 0;
 float totalsum = 0;
 watt_average = 0;

 while(n < 1000) // preform 1000 measurements and look if somebody or something is interested
  {
     // measure AC current in watts
     // The ACS712 chip 20A has a linear curve of 100 mv per ampere between -20 till 20 ampere from 0,5 volt till 4,5
volt by a supply-voltage of 5 volt
     // The Olimexino_stm32 works only with 3.3 volts. With a resistor divider (potentiometer between 5V and ground
trim the output till 3.3V and you have the correct divider)
     // on the output of the ACS712 is fed into the ADC of the Olimexino-stm32, as long there is no current flowing the
division is nearly linear so take at least 4,7K as value.
     // the Olimexino stm32 has a 12 bit DAC digital analog converter. 2^12 is 4096 the zero-crossing is at 2048
     // If the measured value is less than 2048 the relevant voltage = 2048 min the measured value (example 2048 - 490
= 1558)
     // If the measured value is more than 2048 the relevant voltage = the measured value min 2048 ( example 3567 -
2048 = 1519)
     // The result is only positive values. 3.3 Volts divided by 4096 = 0.00080566 volt per bit step.
     // multiplication of the measured value with 0.00080566 gives the voltage of the output from the ACS712 every 1
mV = 0.001V = 0.01 amp
     // so if we measure as value of 1346.5 Millivolt = 1.3465 Volt it's 13,465 ampere (volts times 10 or millivolts
divided by 100)
     // According to the Law of Ohm P=I*V The voltage of the net in Europe is 230Volt when we combine the previous
step volts times 10 with this one we get 23 as multiplication factor
     // By preforming many measurements in rapid succession(1000)and store every measurement we get a clear picture
of the real power that is being used.
     // after 1000 measurements we calculate the mean value and look is there is any interest in the value. If not we start
a new cycle

     measuredvalue = analogRead(AnalogPin);

     if(measuredvalue < 2048){
       measuredvalue = 2048 - measuredvalue;
       }
     else if(measuredvalue > 2048){
       measuredvalue = measuredvalue - 2048;
       }
     else if( measuredvalue == 2048){
       measuredvalue = 0;
     }
```

```
        measuredvalue1 = measuredvalue * digstepV; // multiply the measurement with the constant of the bit-step voltage
        measuredvalue2 = measuredvalue1 * voltage; // P=I*V law of Ohm NB the voltage is divided by 10 to spare a
multiplication
        totalsum += measuredvalue2;    // sum all the values together so we can calculate the mean value after 1000
measurements
        n++;                  // Add one to the counter
  }

  watt_average = totalsum / n;  // The mean wattage = sum of measurements divided by number of measurements
}

void loop()
{
  char var;
  measure_power();
  if (SerialUSB.available() > 0){
     var = SerialUSB.read();
     // if there is some request on the serial port just print the result of the last measurement and start a new one.
     // if there is no interest start a new measurement as well.
     SerialUSB.println(watt_average);
  }
  delay(100);
}
```