Institute of Space Techn

Electrical Engineering Department
EE 20A

Project EMB
Measuring AC Power Using Arduino Nano and ZMPT101b, ZHT103 Modules

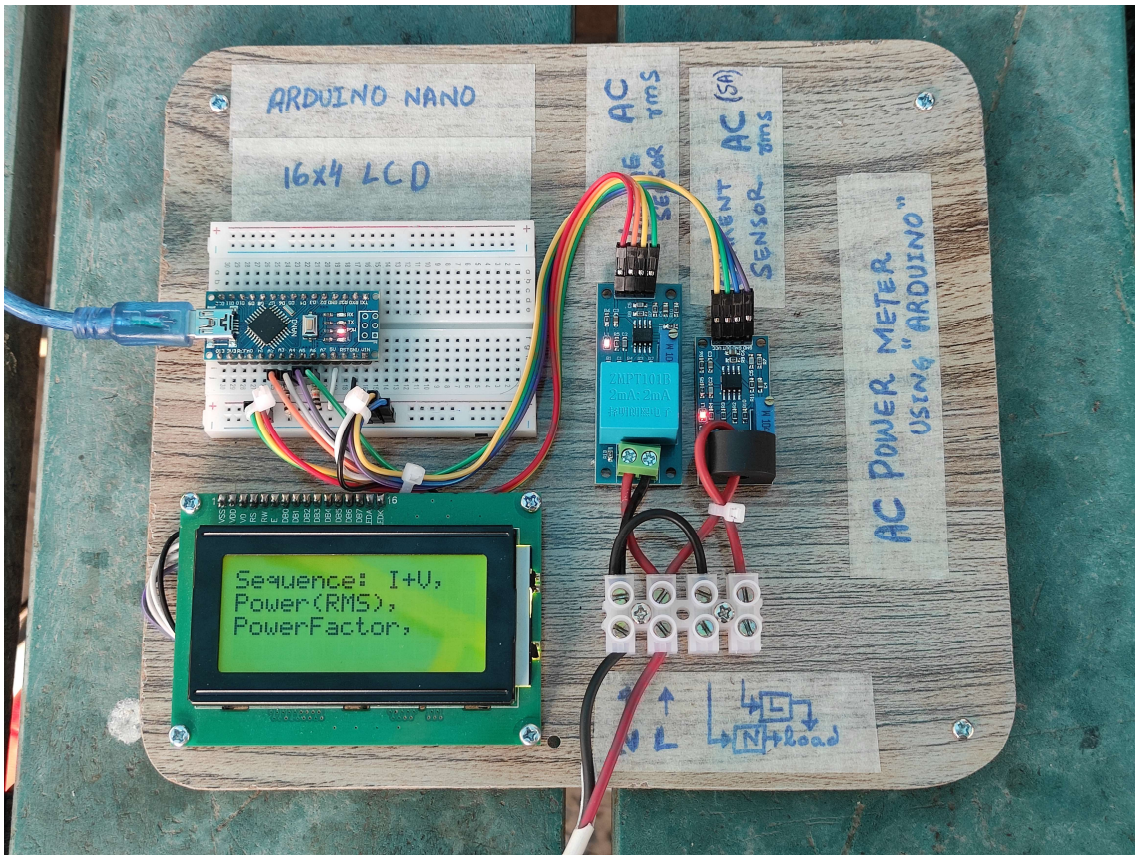
Submitted To:

Sir Mudasir

Group Members:

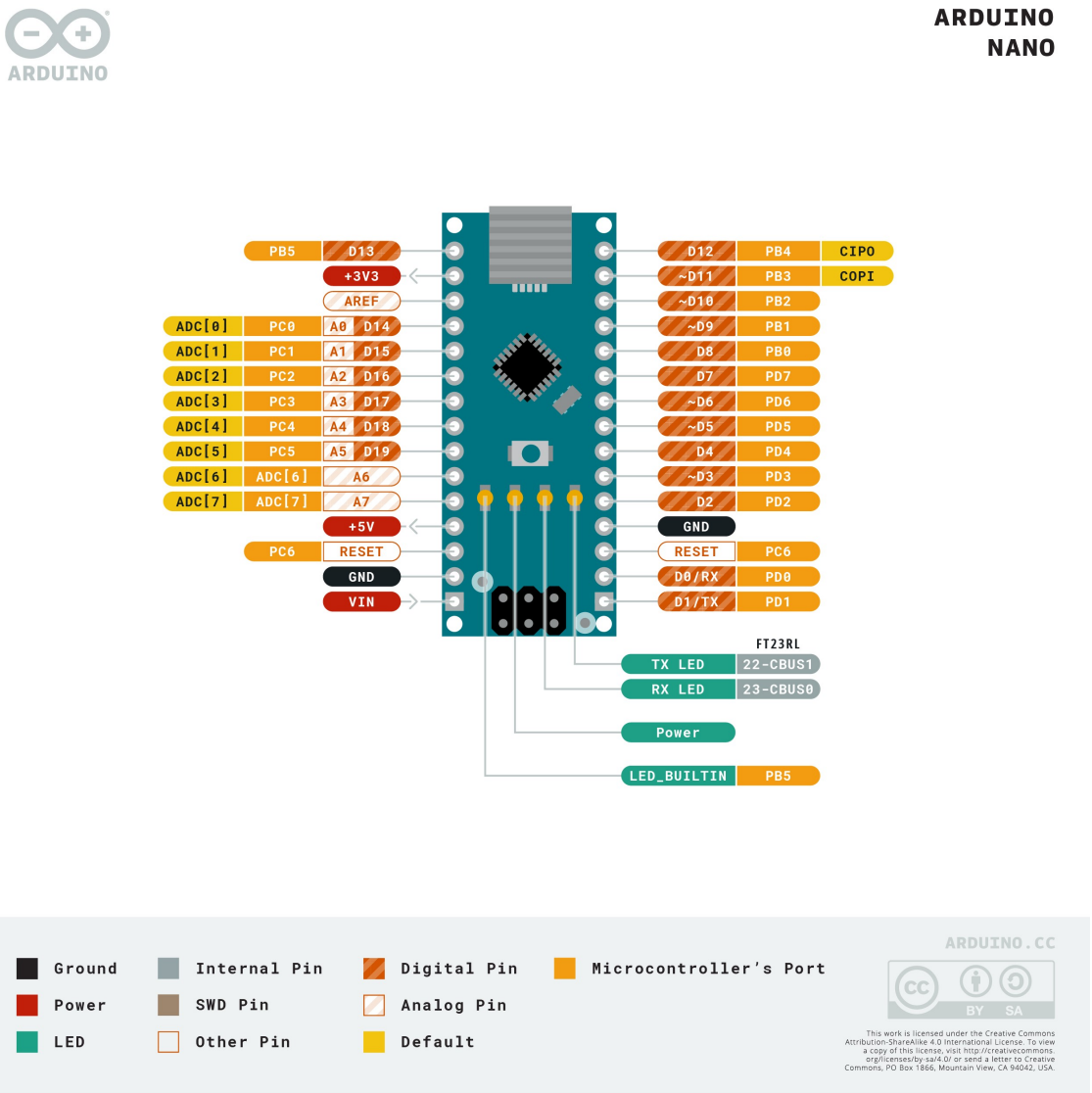
Fahad Qalbi (210401060)
Raza Hussain (210401072)
Rayan Sohail (210401074)
Raza Madni (210401034)

Table Of Content:


ARDUINO NANO
16x4 LCD
AC
AC (5A)
SENSOR
AC POWER METER
USING "ARDUINO"
Sequence: I+V,
Power(RMS),
PowerFactor,
N
L
Load

1. Introduction to Arduino

The Arduino Nano is a popular and versatile microcontroller board that has gained widespread recognition among electronics enthusiasts, students, and hobbyists. Developed by the Arduino company, this compact yet powerful board serves as the foundation for countless creative projects and prototypes. It is highly regarded for its ease of use and adaptability, making it an excellent choice for beginners and experienced electronics enthusiasts alike.



The Arduino Nano is equipped with a variety of pins and features that enable users to interact with and control various electronic components. These pins are divided into three main categories:

ATmega328 Microcontroller

- High-performance low-power 8-bit processor
- Achieve up to 16 MIPS for 16 MHz clock frequency.
- 32 kB of which 2 KB used by bootloader.
- 2 kB internal SRAM
- 1 kB EEPROM
- 32 x 8 General Purpose Working Registers
- Real Time Counter with Separate Oscillator
- Six PWM Channels
- Programmable Serial USART
- Master/Slave SPI Serial Interface

Power

- Mini-B USB connection
- 7-15V unregulated external power supply (pin 30)
- 5V regulated external power supply (pin 27)

Sleep Modes

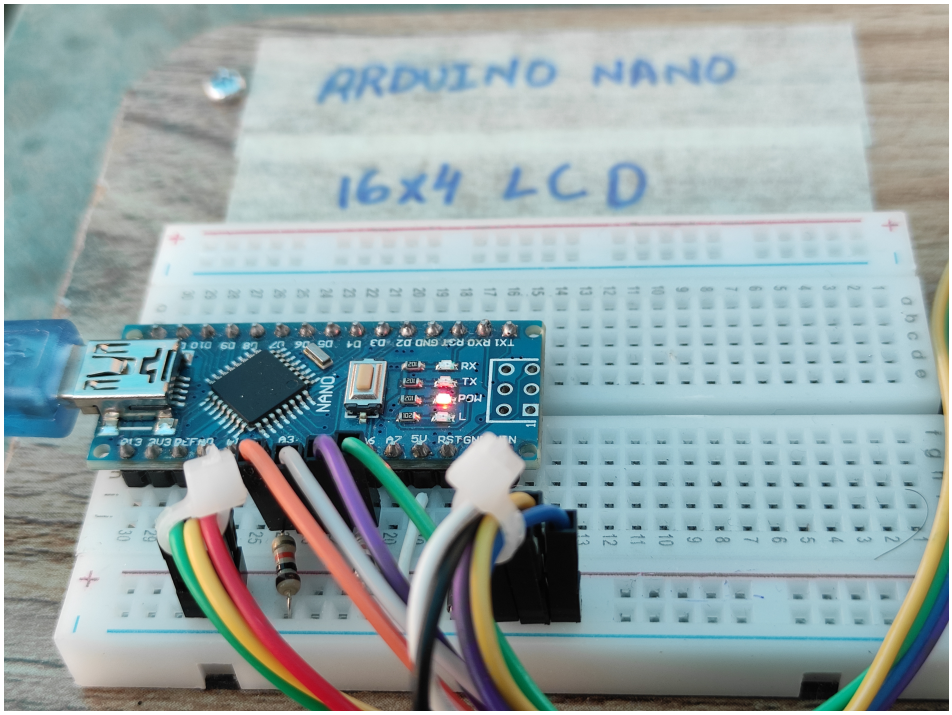
- Idle
- ADC Noise Reduction
- Power-save
- Power-down
- Standby
- Extended Standby

I/O

- 20 Digital
- 8 Analog
- 6 PWM pins

Understanding these pins and their functionalities is essential for successfully building and programming projects using the Arduino Uno. With its user-friendly interface and a wide array of libraries and resources available, the Arduino NANO opens a world of possibilities for anyone interested in electronics and programming. Whether you want to create interactive art, build home automation systems, or explore robotics, the Arduino Uno is an excellent platform to get started.

2. Equipment:



1. 16x2 LCD display:

In this project, I also used a 16x2 LCD display, which is an alphanumeric display that can show up to 32 characters in two lines (16 characters each). It is connected to the Arduino using an I2C module, a simple, reliable communication protocol, allowing the LCD to receive data from the Arduino over just two wires. The I2C bus uses two lines: a serial data line (SDA) and a serial clock line (SCL). SCL stands for Serial Clock. It's one of the two signals used in the I2C communication protocol, the other being the Serial Data (SDA) line. In I2C communication, the SCL line is used to synchronize all data transfers over the I2C bus. It's controlled by the master device, which generates the clock signal.

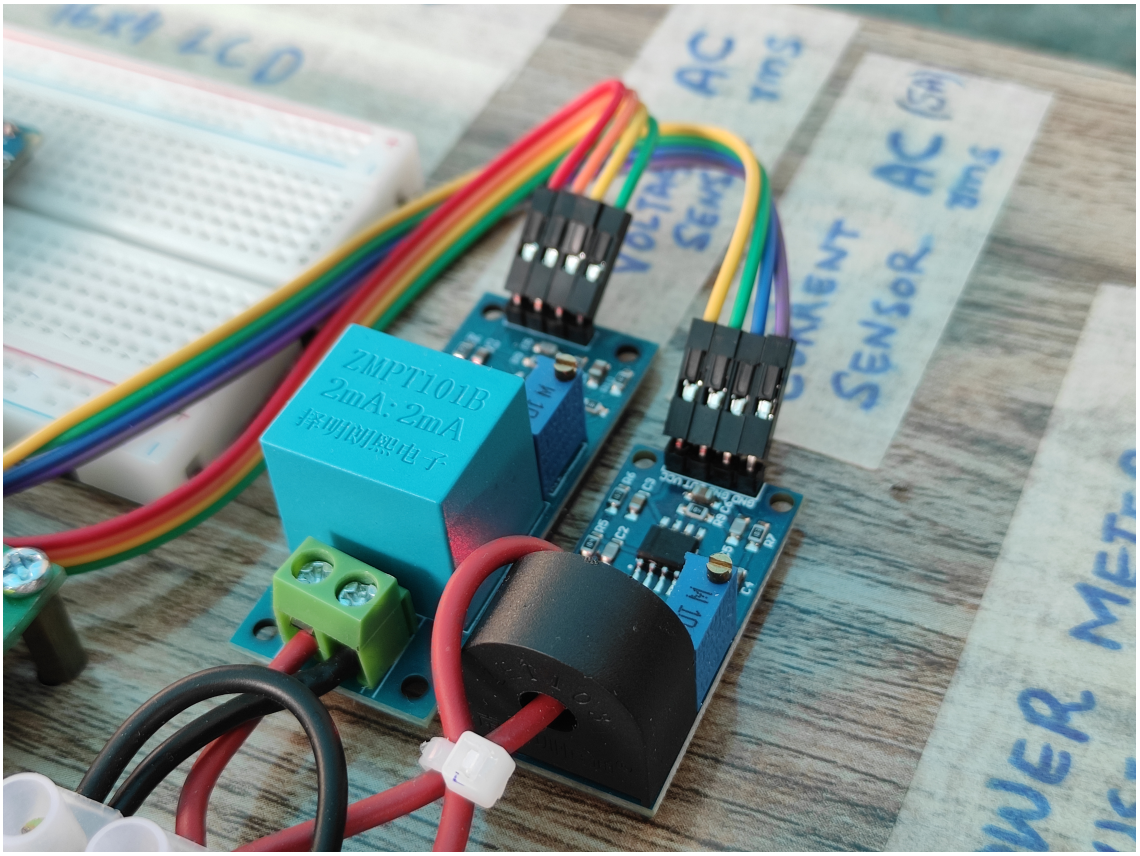
2. I2C module:

The I2C (Inter-Integrated Circuit) module, often referred to as I2C bus or I2C interface, is a widely used serial communication protocol in the field of electronics. It allows multiple devices to communicate with each other over a two-wire bus, typically consisting of a data line (SDA) and a clock line (SCL). I2C is known for its simplicity and versatility, making it a popular choice for connecting various components, such as sensors, displays, and microcontrollers in embedded systems. This protocol enables efficient data exchange between devices while using minimal pins and providing addressing capabilities for

distinguishing between multiple devices on the same bus. I2C is a fundamental tool for building complex and interconnected electronic systems.

3. ZMPT101B module (Voltage Sensor):

The ZMPT101B module is an AC voltage sensor module that is commonly utilized in various electronic applications. It is designed to provide a simple and reliable way to measure AC voltage levels in electrical circuits. The module includes a ZMPT101B voltage transformer and a signal conditioning circuit, making it suitable for interfacing with microcontrollers or other digital devices.



Key features of the ZMPT101B module:

Voltage Measurement: The ZMPT101B module is specifically designed for measuring AC voltage. It can accurately detect and provide voltage values for use in your projects.

Signal Conditioning: The module includes components for signal conditioning, which means that it can provide a more stable and reliable output signal for voltage measurement.

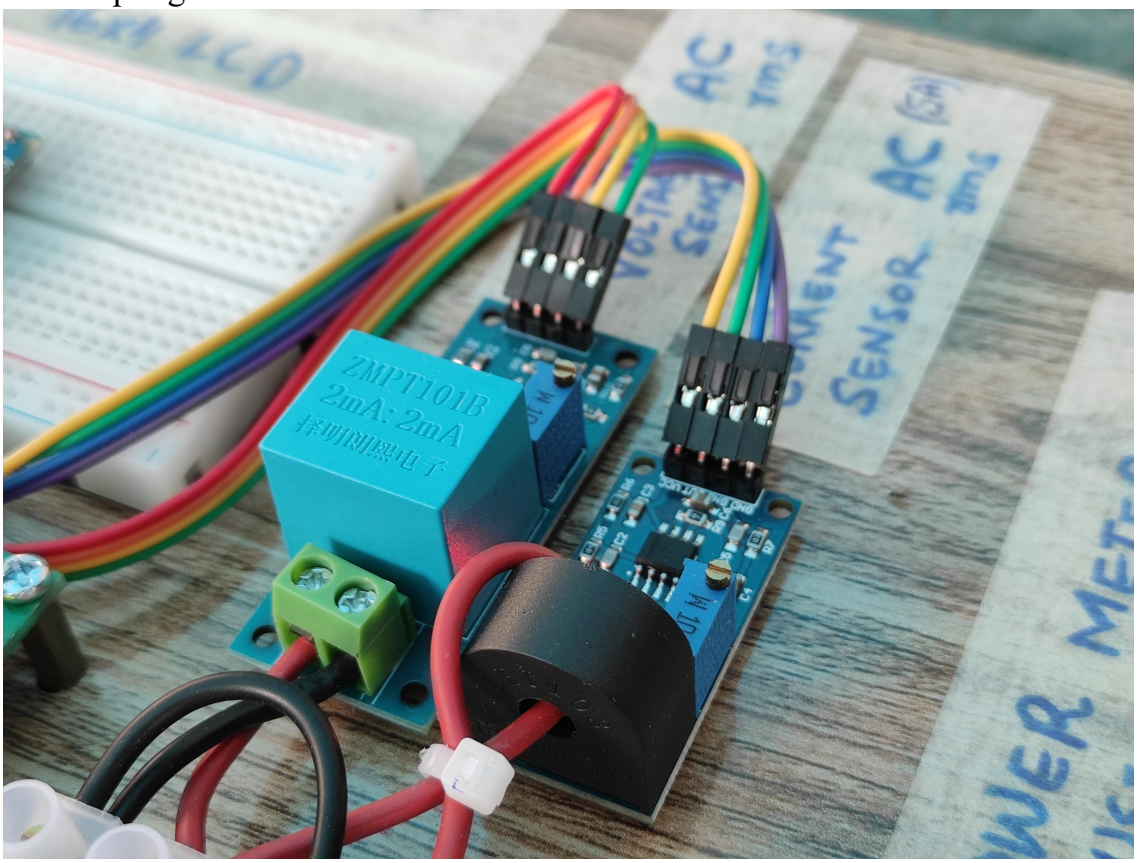
Linear Output: The output of the ZMPT101B is linearly proportional to the input AC voltage, making it easier to convert the sensor data into voltage values.

Low Power Consumption: It operates with low power consumption, making it suitable for battery-powered and low-power applications.

Easy Integration: The ZMPT101B module typically features standard pins for connection to microcontrollers, making it relatively easy to integrate into your projects.

4. ZHT103 module (Current Sensor):

The ZHT103 is a 1-phase AC current sensor. It can measure AC currents less than 5A, the corresponding analog output 5A / 5mA. It has a built-in sampling resistor and an accurate micro-current transformer.



3. Working (Abstract of Code):

1. Library and Objects Initialization:

At the top, the necessary libraries are included, and a LiquidCrystal_I2C object (lcd) and an Energy Monitor object (emon1) are initialized. The LCD has 16 columns and 2 rows, and it uses the I2C address 0x27.

1.1 Emon Library:

The Emon Library, also known as the Energy Monitoring (Emon) Library, is a software library designed for use with Arduino and other compatible microcontroller platforms. This library is primarily used for energy monitoring and measurement applications. It enables developers and hobbyists to interface with various energy monitoring hardware, making it easier to measure and analyze electricity consumption and power-related data.

Key features and functions of the Emon Library include:

1. Real-Time Energy Monitoring: The library provides tools to measure and monitor real-time energy consumption, including voltage, current, power, and power factor.

2. Support for Energy Sensors: Emon Library is compatible with a variety of energy sensors, such as current transformers (CTs) and voltage transformers (VTs), which are commonly used to measure electrical parameters in AC circuits.

3. Data Logging: It allows users to log and store energy consumption data for later analysis or reporting. This is valuable for tracking energy usage patterns and optimizing energy efficiency.

4. Calculations: The library includes functions for calculating active power (watts), apparent power (VA), and reactive power (VAR), making it easier to understand power distribution and usage.

5. Data Processing: Emon Library provides tools for processing raw sensor data and presenting it in a more user-friendly format. This can include averaging data over time or smoothing noisy measurements.

6. Visualization: It may offer functions or compatibility with visualization tools or platforms to display energy consumption data graphically, making it easier to interpret and analyze.

The Emon Library is commonly used in projects related to home energy monitoring, solar power systems, smart grid applications, and energy-efficient building management. By using this library in conjunction with appropriate hardware, developers can create solutions that help users better understand and control their energy consumption, ultimately contributing to energy savings and a more sustainable environment.

1.2 LiquidCrystal_I2C Library:

The LiquidCrystal_I2C library is used to control the LCD display via I2C communication. This allows us to write to the display and control the backlight and cursor position, among other things

2. Setup:

In the setup () function, serial communication is initiated at some baud rate. The LCD is also initialized, and a start-up message "AC Power Meter" is displayed. The `emon1.voltage()` and `emon1.current()` methods are used to set the calibration values for the voltage and current sensors.

3. Main Loop:

In the loop() function, the following sequence is used continuously:

- The string "Sequence: I+V,P" is printed on the LCD i.e. voltage and current + power.
- RMS voltage and current will be calculated using `emon1.calcVI()` for voltage and `emon1.calcIrms()` for current.
- The calculated voltage and current are multiplied to obtain the power in watts.
- The RMS voltage, current, and power will be displayed on the LCD, one at a time, with a 1-second delay between each display.
- The RMS voltage and current values are also sent to the serial monitor.

The lcd.print() and lcd.setCursor() functions will be used to control what is displayed on the LCD and where it is displayed. The Serial.print() and Serial.println() functions will be used to send data to the serial monitor. The delay() function will be used to create a pause between different readings, and lcd.clear() will be used to clear the LCD before displaying new data.

Code:

```
//8888 EmonLibrary examples openenergymonitor.org, Licence GNU GPL V3
#include <LiquidCrystal_I2C.h>
#include "EmonLib.h"                   // Include Emon Library
int lcdColumns = 32;
int lcdRows = 2;
LiquidCrystal_I2C lcd(0x27, lcdColumns, lcdRows);

EnergyMonitor emon1;                   // Create an instance

void setup() {
  lcd.init();
  lcd.backlight();
  lcd.cursor_off();
  lcd.clear();
  lcd.setCursor(0,0);
  lcd.print("AC Power Meter");
  delay(1000);
  emon1.voltage(A3,366, 3.6);
  emon1.current(A6, 1.15999);            // Current: input pin, calibration.
  lcd.clear();
}

void loop() {
  lcd.setCursor(0,0);
  lcd.print("Sequence: I+V,");
  lcd.setCursor(0,1);
  lcd.print("Power(RMS),");
  lcd.setCursor(16,0);
  lcd.print("PowerFactor,");
  delay(2000);
```

```
  emon1.calcVI(20, 4000);
  double Vrms = emon1.Vrms;
  if(Vrms<10){Vrms=0;};
  double Irms = emon1.calcIrms(4000);  // Calculate Irms only
  double Phase = emon1.powerFactor;
  if(Vrms<10 && Irms<0.009){Phase=0;};
  double apparentpower=(Irms*Vrms);

  lcd.clear();
  lcd.setCursor(0,0);
  lcd.print("Vrms:");
  lcd.setCursor(6,0);
  lcd.print(Vrms);
  lcd.setCursor(14,0);
  lcd.print("V");

  lcd.setCursor(0,1);
  lcd.print("Irms:");
  lcd.setCursor(6,1);
  lcd.print(Irms);
  lcd.setCursor(11,1);
  lcd.print("A");
  delay(2000);
  lcd.clear();


  lcd.setCursor(0,0);
  lcd.print("Power:");
  lcd.setCursor(7,0);
  lcd.print(apparentpower);
  lcd.setCursor(13, 0);
  lcd.print(" W");


  lcd.setCursor(0,1);
  lcd.print("Pfactor:");
  lcd.setCursor(9,1);
  lcd.print(Phase);
  delay(2000);
  lcd.clear();

}
```

Output: