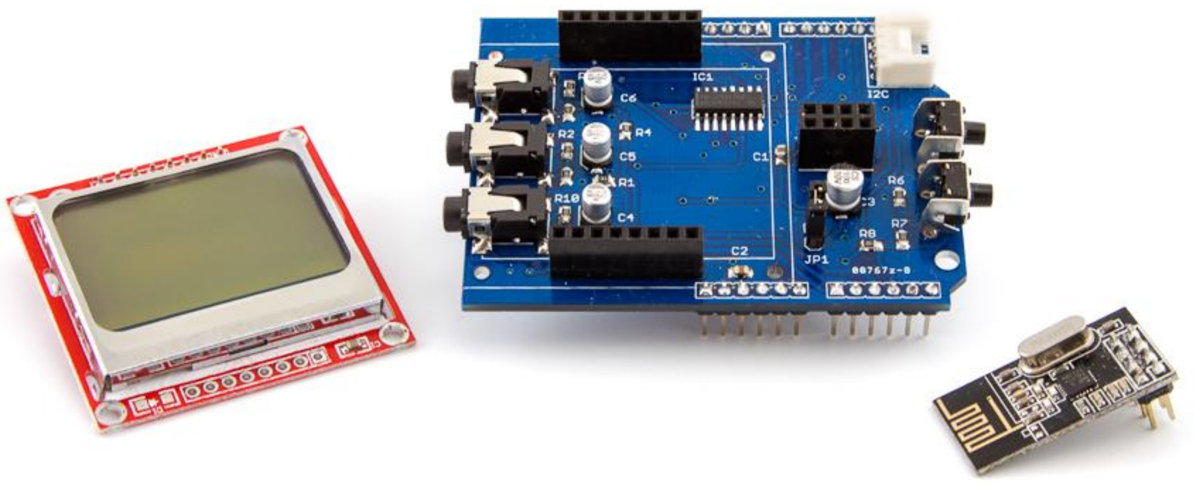
Energy Monitor Shield V0.9b

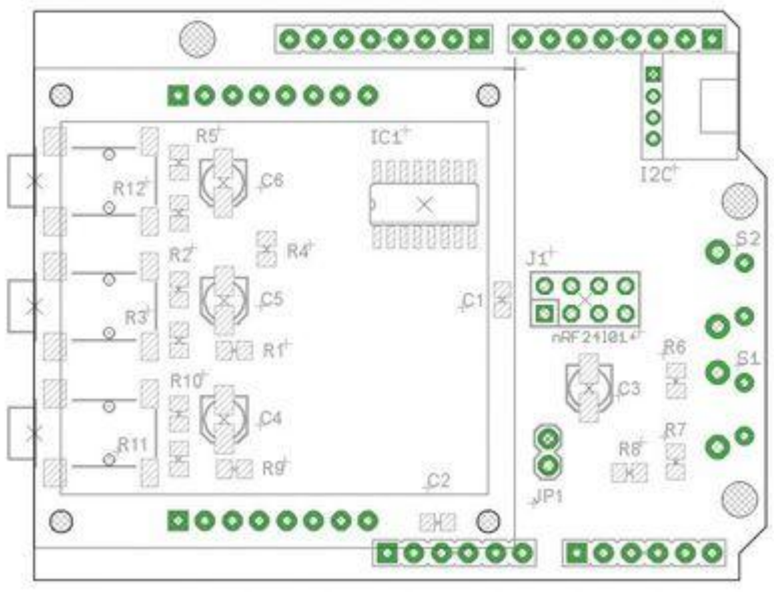
Energy Monitor Shield is an Arduino-compatible expansion card designed for building energy monitoring system with LCD screen and an interface for connecting the wireless transceiver nRF24L01 +.



Feature

- Connect up to three sensors AC (30-100A).
- Support for LCD Screen Nokia LCD5110
- Turn off the LCD backlight with a jumper
- Two buttons to control (operate one analog pin)
- Interface to connect the transceiver to 2.4G nRF24L01 +
- GROVE-compatible connector: I2C
- Fully compatible with Ethernet Shield (Wiznet 5100 + SD)

Layout and schematics



The left side of EM Shield are three connectors for current sensors, right - connector for LCD-screen.

Jumper JP1 is used to enable / disable real-backlight LCD-screen.

In the upper right corner - I2C-connector.

On the right are two buttons (labeled S1 and S2).

In the center of the board (just to the right LCD-screen) - connector for nRF24L01 +.

Schematic of the device

Basic functionality

In the basic version (without using Ethernet Shield) may organize monitoring of energy consumption in three different circuits using current sensors.

Information about the current level of consumption can be displayed on the LCD screen.

Device Management can be organized using two buttons on the Shield.

The obtained data can be transmitted by the transceiver nRF24L01 +.

Expansion Capabilities

Additionally EM Shield can connect any device using i2c Grove-compatible connector (sensors, displays, etc.). EM Shield was designed to be fully compatible with the Ethernet Shield (Wiznet 5100 + SD) - so you can use these two Shields together to create even more advanced device monitoring electricity (logging on SD-card and presenting data on a web page).

Interfaces

- A0, A1, A2 - involved for connecting sensors AC
- A4 (SDA), A5 (SCL) - displayed on the connector "I2C" (the other two pin connector - VCC and GND for sensor supply)
- Interface for connecting RF-module nRF24L01+:
    - D11 - MOSI
    - D12 - MISO
    - D13 - SCK
    - D8 - RF_CE
    - D7 - RF_CSN
    - D2 - RF_IRQ
- Interface for connecting LCD5110:
    - D11 - MOSI
    - D13 - SCK
    - D5 - LCD_D/C
    - D6 - LCD_RST
    - D3 - LCD_CS
- A3 - Buttons

Libraries

Necessary libraries

To use EM Shield requires the following libraries:

- Working with the transceiver nRF24L01+ - RF24
- Using the display LCD 51110 (supporting SPI) - LCD5110_Graph_SPI
- Work with current sensors - EmonLib

Requires the libraries that are used when working RF24 and LCD-display:

- SPI

Features using libraries

Library has used examples of them just to understand how they work.

Initialization RF-module as follows:

```
...


//RF24 radio(CE,CSN);
RF24 radio(7,8);


...
```

Initialize LCD-display is as follows:

```
...


//LCD5110 myGLCD(DC,RST,CS);
LCD5110 myGLCD(5,6,3);


...
```

Demonstration

```
#include <SPI.h>
#include <LCD5110_Graph_SPI.h>
#include "EmonLib.h"

#define WINDOW 15
#define DELTA 1500

EnergyMonitor emon1;

double sIrms;
float Pcur;
float Pmax;
float Pmin=100;

LCD5110 myGLCD(5,6,3);

extern unsigned char SmallFont[];

unsigned long measureTime;
boolean flag=false;
double delta=0;

#define DELTAMEASURE 30000
```

```
void setup(void)
{
  myGLCD.InitLCD(70);
  myGLCD.setFont(SmallFont);

  myGLCD.update();

  emon1.current(0, 111.1);             // Current: input pin, calibration.
  double Irms[WINDOW];
  // Calibrate (find offset)
  double cIrms = 0;
  flag = false;

  while (!flag) {
    myGLCD.print("calibrating", 0, 0);
    myGLCD.update();
    Irms[0] = emon1.calcIrms(1480); // the first value in the measurement obviously "crooked"
      //Serial.println("calculate delta");
      for (int i=0; i<WINDOW; i++) {
        Irms[i] = emon1.calcIrms(1480);
        cIrms = cIrms + Irms[i];
        delay(100);
      }
      delta = cIrms/WINDOW;
      flag = true;
  }

//myGLCD.print(" ready", 0, 0);
    //myGLCD.update();

}

void loop(void)
{
  // write the current value
  sIrms = emon1.calcIrms(1480) - delta;  // Calculate Irms only
  sIrms = abs(sIrms);
  Pcur = sIrms*0.220;

    // the received data is printed
    char tbuf[6];
    dtostrf(Pcur,5,5,tbuf);
    myGLCD.print(tbuf, 0, 0);
    dtostrf(analogRead(A3),4,0,tbuf);
    myGLCD.print(tbuf, 30, 15);

    myGLCD.update();

    delay(250);

}
```