Report of the Course Project Done for the Assignment of ECT 203 Logic Circuit Design of Third semester Bachelor of Technology in Electronics and Communication Engineering

Low Power DC Energy Meter

PROJECT REPORT

Done by

Noel Joyce Varghese ,AJC22EC088

Under the guidance of

Ms. Praseeda B Nair

Asst. Professor, Dept of ECE

Bachelor of Technology In

Electronics & Communication Engineering

Department of Electronics & Communication Engineering

Amal Jyothi College of Engineering

(Affiliated to APJ Abdul Kalam Technological University, Thiruvananthapuram)

Kanjirappally – 686518

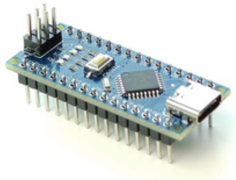
DECEMBER 2023

TABLE OF CONTENTS

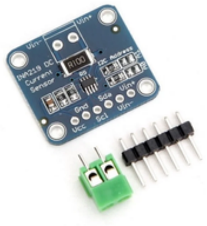
OBJECTIVE

To develop a low-power energy meter/logger capable of measuring and logging 5V DC power consumption and to implement an efficient power monitoring solution.

COMPONENTS REQUIRED

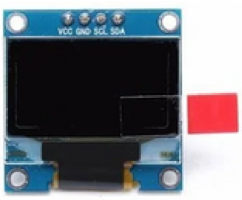


Arduino Nano (ATmega328P) CH340 C Type

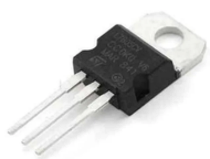


INA219 Power Monitor IC

Micro SD Card Reader Module



4pin OLED Display Module



7805 Voltage Regulator

9V Battery

3V LED Chip Warm White

NB: to be tested on a breadboard and then soldered on a PCB

WORKING PRINCIPLE

Here we are utilising an Arduino Nano and INA219 for precise current and voltage measurements.process the data and the overall control of the system .For the real-time display of power metrics , the output is to be printed on an OLED display module.

Inorder to enable long-term data storage of the recorded values we aim to use Micro SD Card reader module.We are using 9V battery regulated by a 7805 regulator for the power supply.

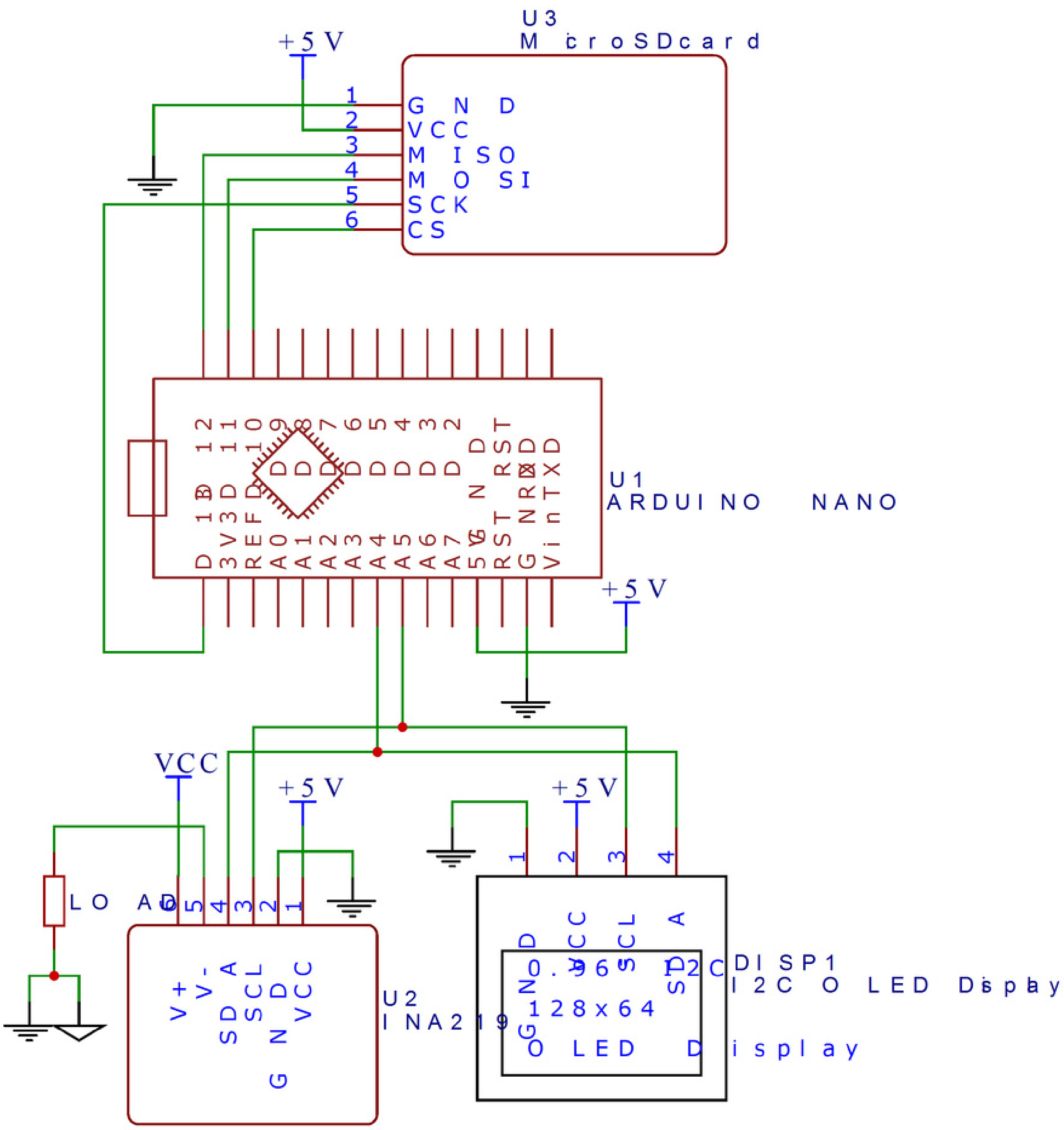
The Arduino Nano (ATmega328P) serves as the central control unit, managing the operation of the energy meter. Firmware on the Arduino initializes and controls the various components, including the INA219, OLED display, and Micro SD Card module.

The INA219 Power Monitor IC is connected in-line with the load (3V LED in this case) to measure current flowing through it. It also measures the voltage across the load, providing both current (I) and voltage (V) readings.

- Raw sensor data is processed by the Arduino to calculate real-time power consumption using the formula: Power (P) = Voltage (V) * Current (I).
- Algorithms are implemented to calculate energy consumption over time, providing cumulative power data.

The Arduino sends processed data to the OLED display module for real-time visualization. The OLED display shows information such as current, voltage, power, and potentially accumulated energy consumption.
The 3V LED serves as a load to simulate a real-world scenario. The energy meter captures and displays the power consumption of the LED.

CIRCUIT DIAGRAM



[NB:includes a 9v Battery conntected using 7805 regulator IC for 5V supply and a 3V led as load]

CODE

```
#include <Adafruit_INA219.h>
#include <SSD1306AsciiAvrI2c.h>
#include <SdFat.h>

//declare timer trigger flag and counter value
volatile boolean triggered = false;

//declare SSD1306 OLED display variables
#define OLED_RESET 4
SSD1306AsciiAvrI2c display;

//declare INA219 variables
Adafruit_INA219 ina219;
float current_mA = 0.0, oldcurr = 0.0;
float loadvoltage = 0.0, oldvolt = 0.0;
float power_mW = 0.0, oldpow = 0.0;
float energy_mWh = 0.0, oldegy = 0.0;
unsigned long elapsed = 0;

//declare microSD variables
#define CHIPSELECT 10
#define ENABLE_DEDICATED_SPI 1
#define SD_CONFIG SdSpiConfig(SD_CS_PIN, DEDICATED_SPI, SPI_CLOCK)
#define SPI_DRIVER_SELECT 0
uint8_t cycles = 0;
SdFat32 sd;
File32 measurFile;

void setup() {
  // Disable ADC
  ADCSRA = 0;
  ACSR = 0x80;

  //setup the INA219
  ina219.begin();
```


[compiled and uploaded through Arduino IDE]

```
//setup the SDcard reader
 sd.begin(CHIPSELECT);
 measurFile.open("MEAS.csv", O_WRITE | O_CREAT | O_TRUNC);
 measurFile.print("Time,Voltage,Current\n");
 measurFile.sync();
//setup the display
  display.begin(&Adafruit128x64, 0x3C, OLED_RESET);
  display.setFont(System5x7);
  display.clear();

  // stop interrupts
  cli();

  // TIMER 1 for interrupt frequency 1 Hz:

  //initialise the CCR register and the counter
  TCCR1A = 0;
  TCCR1B = 0;
  TCNT1  = 0;

  // set compare match register for 10 Hz increments
  OCR1A = 12499; // = 8000000 / (64 * 10) - 1 (must be <65536)

  // turn on CTC mode
  TCCR1B |= (1 << WGM12);

  // Set CS12, CS11 and CS10 bits for 64 prescaler
  TCCR1B |= (0 << CS12) | (1 << CS11) | (1 << CS10);

  // enable timer compare interrupt
  TIMSK1 |= (1 << OCIE1A);

  // allow interrupts
  sei();
}
void loop() {
 //if timer has been reached
 if (triggered)
  {
 //get the values measured by the INA219
 ina219values();
```

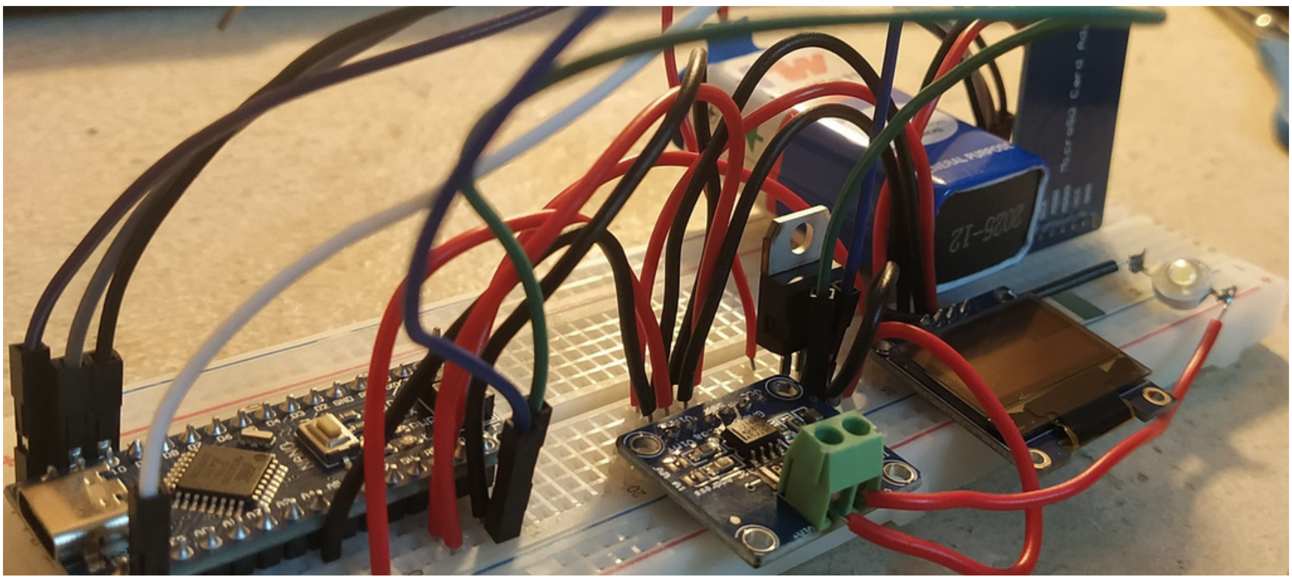
```
//write the data at the end of MEAS.csv
writeFile();
//
// Display update procedure in main loop to avoid
//  wasting clock time in function call
//
  //update the voltage line on the SSD1306 display
if(loadvoltage != oldvolt){
 displayline(loadvoltage, 0, " V");
 oldvolt = loadvoltage;
}

  //update the current line on the SSD1306 display
if(current_mA != oldcurr){
 displayline(current_mA, 2, " mA");
 oldcurr = current_mA;
}

  //update the power line on the SSD1306 display
if(power_mW != oldpow){
 displayline(power_mW, 4, " mW");
 oldpow = power_mW;
}
//update the energy line on the SSD1306 display
if(energy_mWh != oldegy){
displayline(energy_mWh, 6, " mWh");
oldegy = energy_mWh;
}

//reset the flag
triggered = false;
}
}

ISR(TIMER1_COMPA_vect){
triggered = true;
}
```

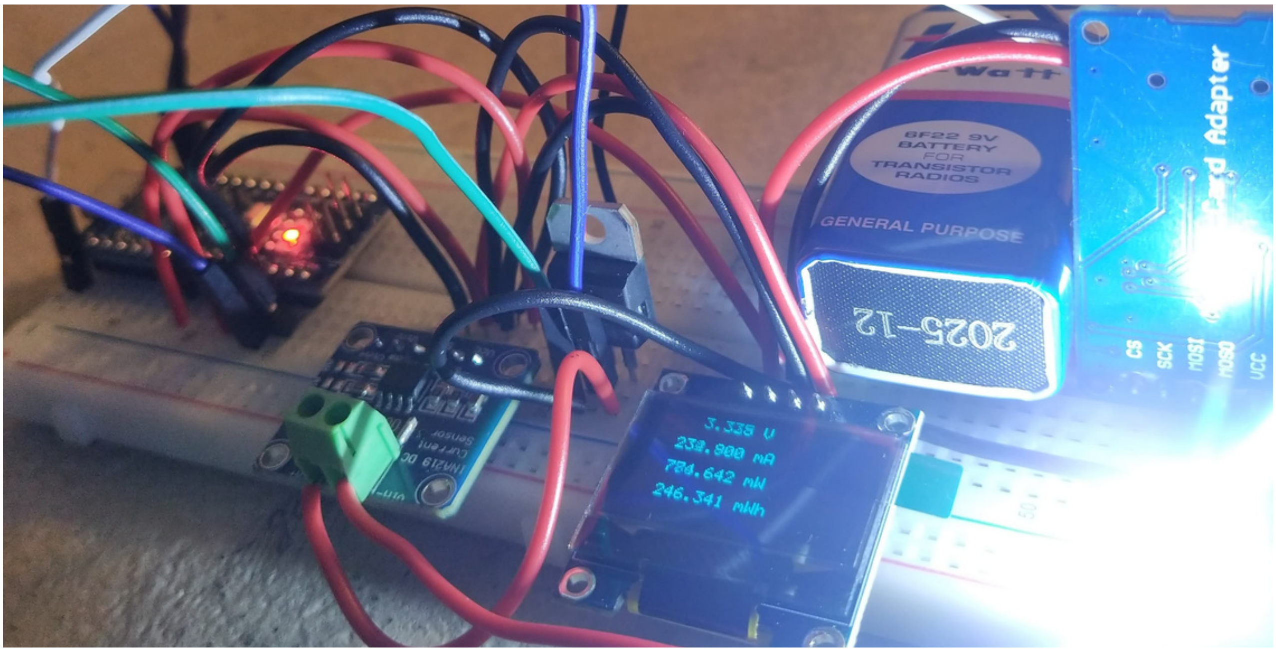
```
void displayline(const float measurment, const uint8_t line_num, const char
line_end[]) {
  char floatbuf[16]={0};

  //format the line ([-]xxxxx.xxx [unit])
  dtostrf(measurment, 10, 3, floatbuf);
  strcat(floatbuf, line_end);

  //place the cursor and write the line
  display.setCursor(0, line_num);
  display.print(floatbuf);
}
void ina219values() {
 float shuntvoltage = 0.0;
 float busvoltage = 0.0;

 //turn the INA219 on
 ina219.powerSave(false);

 //get the shunt voltage, bus voltage, current and power consumed from the INA219
 shuntvoltage = ina219.getShuntVoltage_mV();
 busvoltage = ina219.getBusVoltage_V();
 current_mA = ina219.getCurrent_mA();
 elapsed = millis();

 //turn the INA219 off
 ina219.powerSave(true);

 //compute the load voltage
 loadvoltage = busvoltage + (shuntvoltage / 1000.0);

 //compute the power consumed
 power_mW = loadvoltage*current_mA;

 //compute the energy consumed (t = elapsed[ms] / 3600[s/h] * 1000[ms/s])
 energy_mWh += power_mW * ( elapsed / 3600000.0);
}
```

```
void writeFile() {
  char buf[32], voltbuf[16]={0}, curbuf[16]={0};

  //prepare buffers with the voltage and current values in strings
  dtostrf(loadvoltage, 10, 3, voltbuf);
  dtostrf(current_mA, 10, 3, curbuf);


//format a csv line : time,voltage,current\n
  sprintf(buf, "%ld,%s,%s\n", elapsed, voltbuf, curbuf);

  //write the line in the file
  measurFile.write(buf);

  //after 9 cycles (1 sec.), apply SD buffer changes to file in SD
  if(cycles >=9)
   measurFile.sync();

  //increment cycles count + reset to 0 after 10 cycles
  cycles++;
  cycles %= 10;
}
```

RESULT ANALYSIS

Collected data is analyzed and discussed.The accuracy and reliability of the energy meter are evaluated based on the test results.





CONCLUSION

- The successful implementation of the Low Power DC 5V Energy Meter is summarized.
- Key findings and the overall performance of the system are highlighted

REFERENCE

Instructables website
Arduino forum
Github
@Greatscottlab