Priza Smart

Introducere

Ai patit vreodata sa te trezesti pe mail cu o factura la curent de 400 de lei venita de nicaieri? S-a intamplat vreodata sa te uiti la contorul de electricitate sa citesti 343kWh ca mai apoi, dupa o zi sa vezi 393kWh tu avand bagat in priza doar un amarat de televizor? Esti curios sa vezi cat consuma calculatorul tau de gaming atunci cand te joci Hogwarts Legacy?

Toate aceste probleme pot fi usor rezolvate cu ajutorul unei prize smart, care va arata pe un display consumul de curent pe care il aveti. Aceasta idee mi s-a nazarit atunci cand mi-a venit curiozitatea de a masura cat curent consuma diferite electronice de la mine de acasa. Sunt de parere ca aceasta priza smart este un bun foarte folositor atunci cand incercam sa reducem consumul electric din casa.

Descriere generală

Priza smart este un prelungitor cu o singura priza care inregistreaza informatii despre consumul de curent pe care dispozitivul bagat in priza il foloseste. Aceasta priza stocheaza datele inregistrate care pot fi folosite mai tarziu pentru a trasa grafice de consum. Priza foloseste de asemenea un display pentru a afisa atat consumul instantaneu, cat si consumul total de cand a inceput masuratoarea. Exista si un buton de reset pentru a reincepe masuratoarea

Hardware Design

Voi folosi urmatoarele componente:

- Placa de Dezvoltare Compatibila cu Arduino UNO R3 (ATmega328p + ATmega16u2)

- Senzor de curent Hall ACS712 (30A)
- Display OLED 0.96''
- Modul Slot Card Compatibil cu MicroSD
- Baterie 12V
- Buton pentru reset
- Cabluri
- Suport priza tata si mama

Software Design

Mediu De Dezvoltare

Arduino IDE

Biblioteci folosite

- Waveshare OLED_0_96_Display
- SdFat by Bill Greiman

Pentru a reduce consumul de memorie, a trebuit sa fac cateva schimbari in biblioteca celor de la Waveshare

Cod Sursa

Functie de Setup

Functia de setup are ca rol initializarea diferitilor pini si difertelor componenta care creeaza proiectul

```
void setup() {
  sei(); // Permite intreruperi

  configure2msTimer(); // Configurare

  /* Initializare pin pentru card SD */
  pinMode(sdCardPin, OUTPUT);
  digitalWrite(sdCardPin, LOW);

  /* Initializare biblioteca si imagine al display-ului */
  System_Init();
  OLED_0in96_Init();
  Driver_Delay_ms(500);
  OLED_0in96_clear();
  UWORD Imagesize = ((OLED_0in96_WIDTH%8==0)? (OLED_0in96_WIDTH/8): (
```

```
OLED_0in96_WIDTH/8+1)) * OLED_0in96_HEIGHT;
  if((BlackImage = (UBYTE *)malloc(Imagesize)) == NULL) {
      return -1;
  }
  Paint_NewImage(BlackImage, 64, 128, 90, BLACK);
  Paint_SelectImage(BlackImage);
  Paint_Clear(BLACK);

  pinMode(A0, INPUT); // Initializare pin pentru citire curent

  /* Initializare pin pentru Reset*/
  pinMode(resetPin, INPUT_PULLUP);
  attachInterrupt(digitalPinToInterrupt(resetPin), resetTotal, FALLING);
}
```

Functie de Resetare

Functia de resetare are ca scop resetarea variabilei care tine energia totala consumata de priza

```
void resetTotal() {
  totalPower = 0.0;
  displayPower(oldAvgPower, totalPower);
}
```

Functia de citire curent

Functia de citire a curentului este folosita pentru a converti valoarea citita analogic in curentul real citit de senzor. Pentru a avea niste valori mai apropiate de realitate, se va face media a 500 de valori citite la fiecare secunda

```
void readCurrent() {
  int currentSensorValue = analogRead(A0);

  float sensedVoltage = (Vcc / analogMax) * currentSensorValue;

  float sensedCurrent = (sensedVoltage - ACS712_Ref) / ACS712_Sensitivity;

  if (sensedCurrent < 0) {
    sensedCurrent *= -1;
  }

  avgSensedCurrentTotal += sensedCurrent;

  measures++;
}
```

Functia de afisare pe display

Functia de afisare pe display este folosita pentru a crea si afisa pe ecran string-urile care redau informatia

```
void displayPower(float avgPower, float totalPower) {
  char avgPowerStr[24] = "Consmpt.: ";
  char totalPowerStr[24] = "Total: ";

  char avgPowerF[10];
  char totalPowerF[10];

  floatToStr(avgPowerF, avgPower);
  floatToStr(totalPowerF, totalPower);

  strcat(avgPowerStr, avgPowerF);
  strcat(totalPowerStr, totalPowerF);
  strcat(avgPowerStr, "W");
  strcat(totalPowerStr, "Wh");

  Paint_Clear(BLACK);
  Paint_DrawString_EN(10, 0, avgPowerStr, &Font8, WHITE, WHITE);
  Paint_DrawString_EN(10, 17, totalPowerStr, &Font8, WHITE, WHITE);
  // Show image
  OLED_0in96_display(BlackImage);
}
```

Functie pentru scriere pe card SD

Aceasta functie este folosita pentru a scrie intr-un fisier de pe cardul SD informatiile necesare pentru trasarii graficului, adica o pereche (x,y) unde x este secunda la care a facut efectuata masura si y energia totala consumata pana la acel punct.

```
void writeToSDCard(float x, float y) {
  digitalWrite(displayPin, LOW);
  digitalWrite(sdCardPin, HIGH);

  if (sd.begin(sdCardPin, SPI_HALF_SPEED)) {
    bool res = graph.open("graph.txt", O_WRONLY | O_APPEND | O_CREAT);

    char xStr[10], yStr[0];

    floatToStr(xStr, x);
    floatToStr(yStr, y);

    char result[16] = "(";
```

```
    strcat(result, xStr);
    strcat(result, ", ");
    strcat(result, yStr);
    strcat(result, ")\n");

    Serial.println(result);

    graph.print(result);
    graph.close();
  } else {
    Serial.println("Could not initialize SD card");
  }

  digitalWrite(sdCardPin, LOW);
  digitalWrite(displayPin, HIGH);
}
```

Intreruperea

Intreruperea timer-ului are loc la fiecare 2ms. Este folosita pentru a citi atat pentru a citi senzorul (la fiecare 2ms) si pentru a afisa pe ecran valorile updatate (la fiecare secunda). Valoarea puterii este calculata folosind media ultimelor 500 de valori citite

```
ISR(TIMER1_COMPA_vect) {
  readCurrent();
  entries++;

  if (entries == 500) {
    entries = 0;
    float avgPower = (avgSensedCurrentTotal / measures) * 230;

    avgSensedCurrentTotal = 0.0;

    measures = 0;

    totalPower += oldAvgPower / 3600;

    if (avgPower < 0) {
      avgPower = 0.0;
    }

    oldAvgPower = avgPower;

    displayPower(avgPower, totalPower);
    writeToSDCard(counter, avgPower);
    counter++;
  }
}
```

Functii ajutatoare

Functia `floatToStr` este folosita pentru a transforma un float intr-un char *

```
void floatToStr(char *result, float num) {
  long a = (long) (num * 100);

  long copy;
  copy = a;
  short int len = 0;

  while (copy > 0) {
    copy /= 10;
    len++;
  }

  if (len <= 2) {
    len = 3;
  }

  result[len + 1] = 0;

  result[len--] = (a % 10) + 48;
  a /= 10;
  result[len--] = (a % 10) + 48;
  a /= 10;
  result[len--] = '.';

  while (len >= 0) {
    result[len--] = (a % 10) + 48;
    a /= 10;
  }
}
```

Din motive de memorie, modulul de card SD si cel de display nu pot functiona simultan pe un Arduino Uno care are doar 2kb de memorie SRAM. Am adaugat si verificat modulele individual, dar pot functiona impreuna doar cu un microcontroller cu mai multa memorie

Rezultate Obţinute

Concluzii

Download

priza_smart.zip

Jurnal

Puteți avea și o secțiune de jurnal în care să poată urmări asistentul de proiect progresul proiectului.

Bibliografie/Resurse

Listă cu documente, datasheet-uri, resurse Internet folosite, eventual grupate pe **Resurse Software** şi **Resurse Hardware**.

Export to PDF