SD Card Buddy

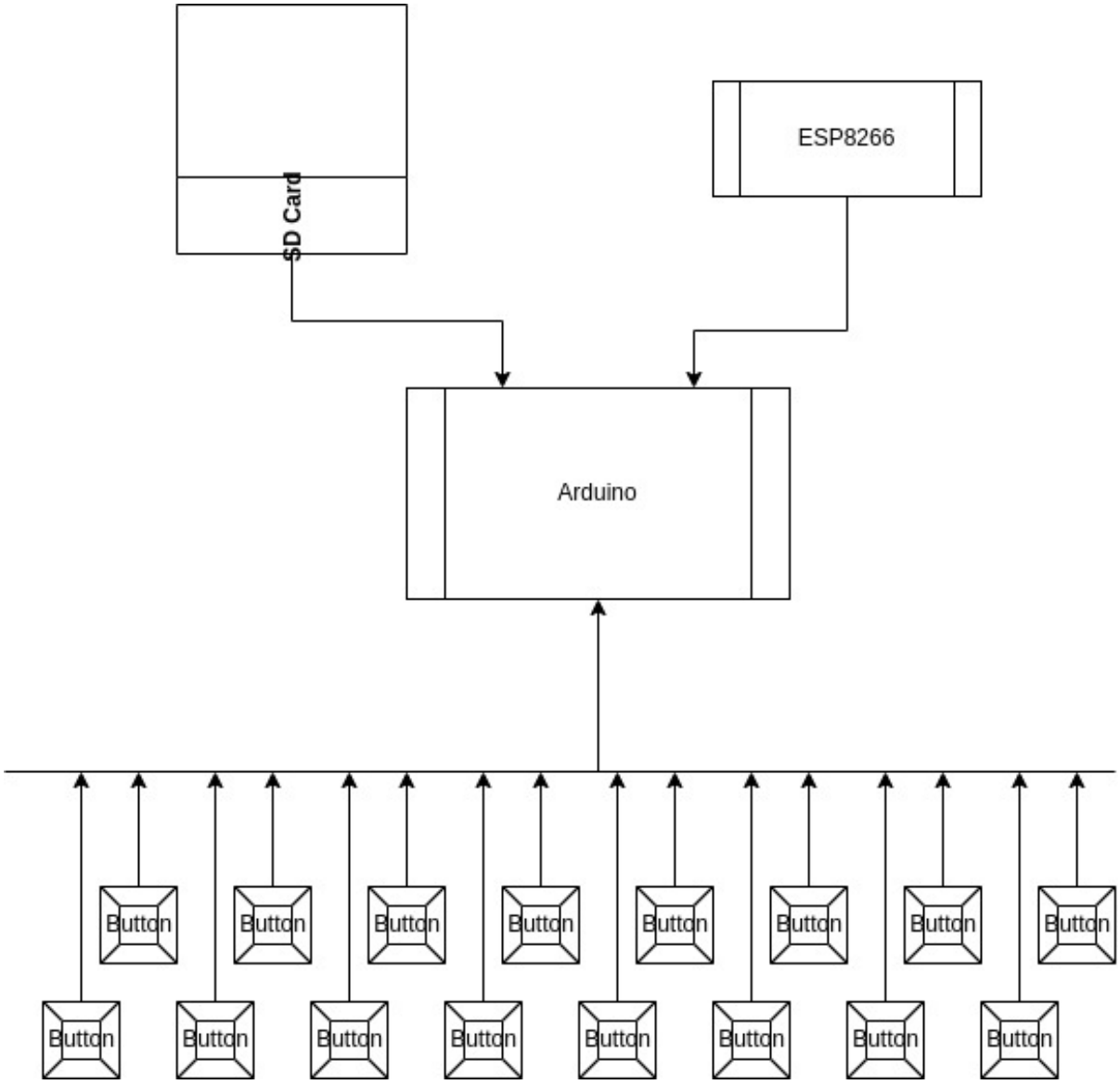
Sergiu Moga 332CB

Introducere

SD Card Buddy este un proiect Arduino care are ca scop expunerea unui shell prin care un utilizator poate interactiona cu sistemul de fisiere de pe un SD Card. Ideea proiectului este de a avea ceva micut si la indemana cu care poti modifica continutul de pe un SD Card, ceea ce il poate face util in anumite cazuri.

Descriere generală

Folosind o placuta compatibila Arduino, prin comunicare SPI cu 2 sclavi (un display si un modul de SD Card) o sa expun printr-un display un shell interactiv cu care utilizatorul poate naviga/citi/scrie intr-un sistem de fisiere de pe SD Card. Pentru interactionare se va folosi o tastatura improvizata (15-16 butoane legate la acelasi pin prin rezistente diferite pentru a le putea distinge cu ajutorul analogRead).

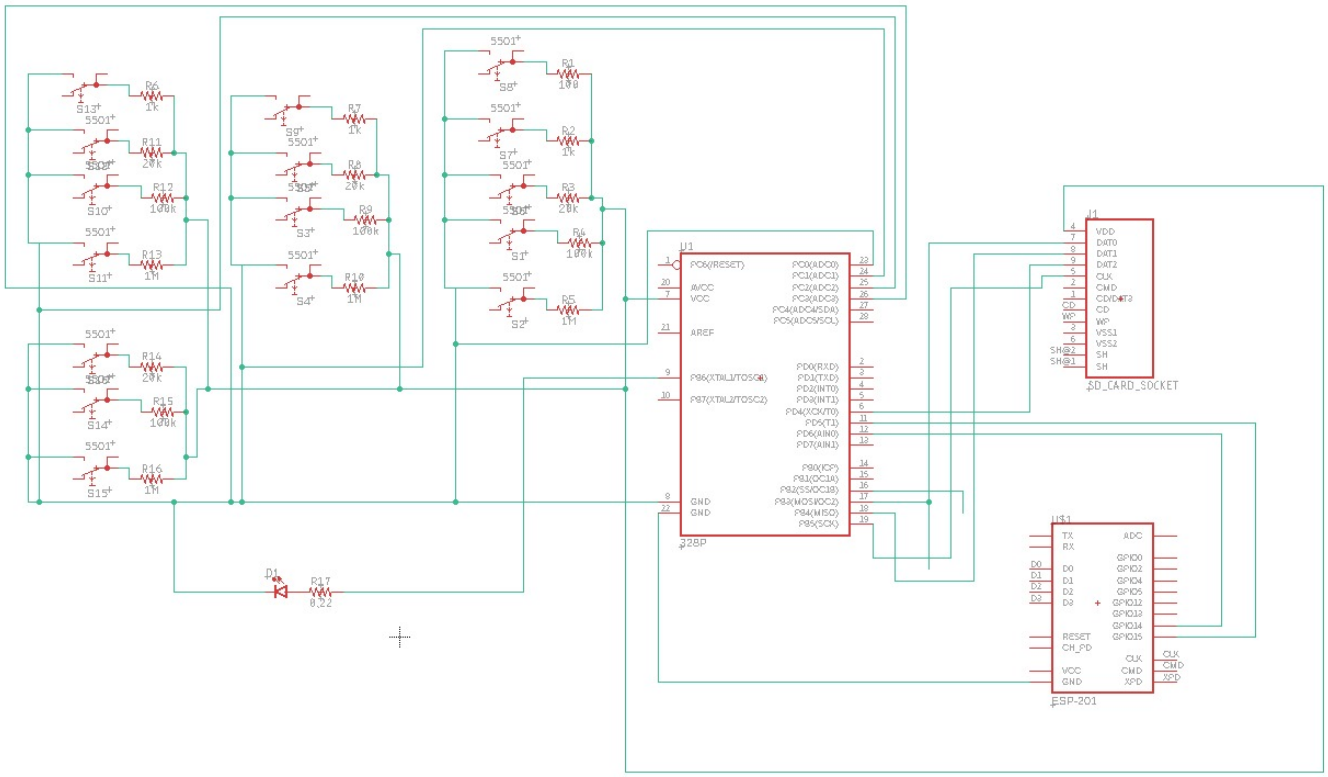


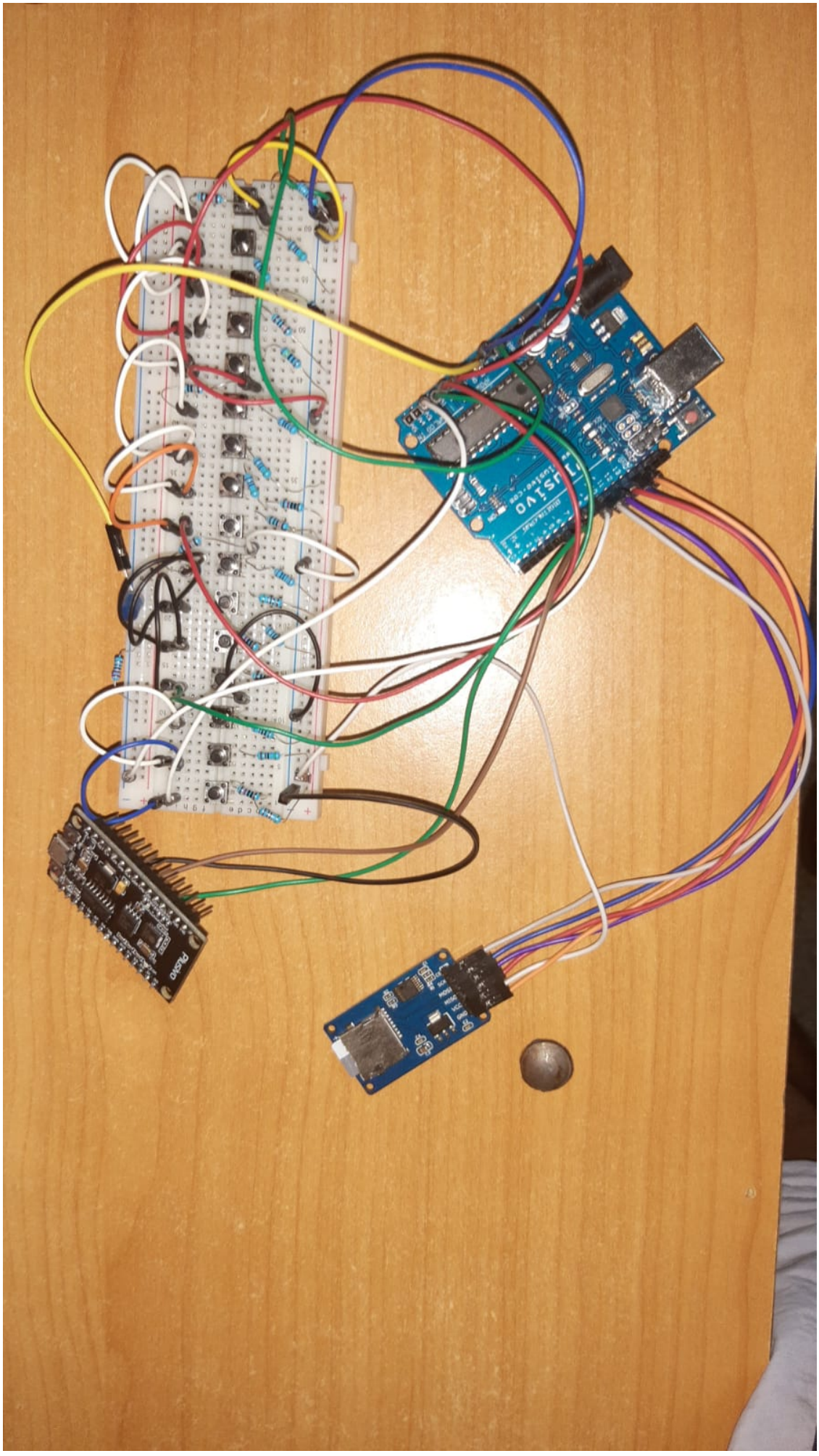
Hardware Design

Componente:

- Placuta compatibila Arduino UNO
- display cu interfatare SPI
- modul de SD Card cu interfatare SPI
- 15-16 push-buttons
- niste fire + cablaj testare

Schematic:


U1
328P
J1
SD_CARD_SOCKET
U$1
ESP-201
D1
R17
0.22



Software Design

Software utilizat:

- Arduino IDE
- draw.io
- Eagle

Biblioteci utilizate:

- **ESP8266WebServer** pentru a putea obtine functionalitatea de Server Web pe ESP8266
- **SoftwareSerial** pentru a putea realiza usor comunicarea intre cele doua placute compatibile Arduino
- **SPI** si **SD** pentru a putea realiza comunicarea cu cardul microSD

Structuri de date utilizate:

- o matrice **kbd_buttons** care contine asocierea litere ↔ butoane tastatura
- un array **kbd_button_index** care retine indecsii din matricea **kbd_buttons** corespunzatoare valorii citite prin **analogRead**
- **second_half** este folosit ca index al primei dimensiuni a matricii ca sa poata fi accesata a doua jumatate a tastaturii (ca un **Caps Lock**)

```
int second_half = 0;
// ranges = {{1000, 1050}, {950, 1000}, {850, 900}, {450, 550}, {50, 150}}
int kbd_button_index[] = {4, 4, -1, -1, 3, 3, -1, -1, 2, 1, 0};
char kbd_buttons[2][4][5] = { { {'A', 'B', 'C', 'D', 'E'},
                                {'F', 'G', 'H', 'I'},
                                {'J', 'K', 'L', 'M'},
                                {' ', '0', '\n'}  },
                              { {'N', 'O', 'P', 'Q', 'R'},
                                {'S', 'T', 'U', 'V'},
                                {'W', 'X', 'Y', 'Z'},
                                {'/', '0', '\n'}  } };
```

- un buffer care retine comanda de interpretat si unul pentru a retine output-ul ce trebuie afisat pe serverul web
- indecsii pentru a retine pozitia in cele doua buffere

```
char cmd[50];
char output[150];
int cmd_index, output_index;
```

Functii/Algoritmi:

Setup:

- ESP8266: incepe seriala intre Uno si ESP8266, realizeaza conexiunea Wi-Fi, incepe server-ul HTTP

```
void setupServer()
{
  server.on("/", htmlIndex);

  server.begin();
}

void htmlIndex()
{
  int replyCode = 200;

  String contentType = "text/html";

  memcpy(markdown + sizeof(head) - 1, message, sizeof(message));

  server.send(replyCode, contentType, markdown);
}

void connectToWiFi()
{
  WiFi.mode(WIFI_STA);

  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED)
    delay(500);
}

void setup()
{
  mySerial.begin(4800);

  delay(1000);

  connectToWiFi();

  setupServer();
}
```

- Uno: Incepe seriala intre Uno si ESP8266 si conexiunea cu modulul SDCard. Daca conexiunea din urma esueaza, agata executia

```
void setup() {
  mySerial.begin(4800);

  if (!SD.begin(10)) {
    while (1);
  }
}
```

Loop:

- ESP8266: citeste din conexiunea seriala si asteapta ca clientul sa faca un HTTP Request. Cand clientul face un request, se va executa **htmlIndex**

```
void htmlIndex()
{
  int replyCode = 200;

  String contentType = "text/html";

  memcpy(markdown + sizeof(head) - 1, message, sizeof(message));

  server.send(replyCode, contentType, markdown);
}

void loop()
{
  mySerial.readBytes(message, sizeof(message));

  server.handleClient();
}
```

- Uno: parcurge pinii ADC si decide ce cheie a fost apasata si stocheaz-o in buffer-ul de comenzi. Daca s-a apasat **Enter**, inseamna ca se doreste executia de comenzi si functia **cmd_exec** este apelata. Aceasta functie executa operatia de I/O dorita cu SD Card-ul. De exemplu, daca se doreste afisarea continutului SD Card-ului, parcurge recursiv continutul directoarelor apeland functia **list**

```
void list(File dir, int level) {
  int len;
  while (true) {
    File entry =  dir.openNextFile();

    if (!entry) {
      return;
    }

    for (uint8_t i = 0; i < level; i++) {
      output[output_index++] = '-';
      output[output_index++] = '-';
    }

    len = strlen(entry.name());
    memcpy(output + output_index, entry.name(), len);
    output_index += len;

    output[output_index++] = '<';
    output[output_index++] = 'b';
    output[output_index++] = 'r';
    output[output_index++] = '>';

    if (entry.isDirectory()) {
      output[output_index++] = '<';
```

```
      output[output_index++] = 'b';
      output[output_index++] = 'r';
      output[output_index++] = '>';

      list(entry, level + 1);

    }

    entry.close();
  }
}

void cmd_exec() {
  char *p = strtok(cmd, " ");
  File file, dir;
  int offset, bytes_count;

  if (p) {
    if (*p == 'L') {
      p = strtok(NULL, " ");

      dir = SD.open(p);

      list(dir, 0);

      output[output_index] = 0;
      mySerial.write(output, sizeof(output));
      output_index = 0;

      return;

    } else if (*p == 'R') { // read
      p = strtok(NULL, " ");
      if (!SD.exists(p)) {
        memcpy(output + output_index, "No such file or directory!", sizeof(
"No such file or directory!"));
        output_index += sizeof("No such file or directory!");

        output[output_index] = 0;
        mySerial.write(output, sizeof(output));
        output_index = 0;

        return;
      }

      Serial.println(p);
      file = SD.open(p, FILE_READ);

      char c = 0;
      c = file.read();
      while (c != -1) {
```

```
        output[output_index++] = c;
        c = file.read();
      }

      file.close();

      output[output_index] = 0;
      mySerial.write(output, sizeof(output));
      output_index = 0;

      return;

    } else if (*p == 'W') { // write
      p = strtok(NULL, " ");

      Serial.println(p);
      file = SD.open(p, FILE_WRITE);

      p = strtok(NULL, " ");

      Serial.println(p);
      file.write(p, strlen(p));

      file.close();

      return;

    } else {
      Serial.println("lool");
      memcpy(output + output_index, "No such command!\n", sizeof("No such
command!<br>"));
      output_index += sizeof("No such command!<br>");

      output[output_index] = 0;
      mySerial.write(output, sizeof(output));
      output_index = 0;

      return;
    }
  }
}

void loop() {
  char c;

  A0_value = analogRead(A0);
  if (A0_value > THRESHOLD) {
    c = kbd_buttons[second_half][0][kbd_button_index[A0_value / 100]];

    cmd[cmd_index++] = c;
```

```
    output[output_index++] = c;
    output[output_index] = 0;
    mySerial.write(output, sizeof(output));
  }

  A1_value = analogRead(A1);
  if (A1_value > THRESHOLD) {
    c = kbd_buttons[second_half][1][kbd_button_index[A1_value / 100]];

    cmd[cmd_index++] = c;

    output[output_index++] = c;
    output[output_index] = 0;
    mySerial.write(output, sizeof(output));
  }

  A2_value = analogRead(A2);
  if (A2_value > THRESHOLD) {
    c = kbd_buttons[second_half][2][kbd_button_index[A2_value / 100]];

    cmd[cmd_index++] = c;

    output[output_index++] = c;
    output[output_index] = 0;
    mySerial.write(output, sizeof(output));
  }

  A3_value = analogRead(A3);
  if (A3_value > THRESHOLD) {
    c = kbd_buttons[second_half][3][kbd_button_index[A3_value / 100]];

    switch (c) {
    case '/':
    case ' ':
      cmd[cmd_index++] = c;

      output[output_index++] = c;
      output[output_index] = 0;
      mySerial.write(output, sizeof(output));

      break;

    case '\n':
      output[output_index++] = '<';
      output[output_index++] = 'b';
      output[output_index++] = 'r';
      output[output_index++] = '>';
      output[output_index] = 0;

      cmd_exec();
      memset(cmd, 0, sizeof(cmd));
```

```
      cmd_index = 0;
      output_index = 0;
      memset(output, 0, sizeof(output));

      break;

    case '0':
      second_half ^= 1;

      analogWrite(9, second_half);

      break;
    }
  }

  delay(150);
}
```

Rezultate Obţinute

Proiectul functioneaza cum a fost intentionat. Astfel, interactionand cu tastatura din cei 16 push-buttons, un utilizator poate face I/O pe un SDCard. Desi am avut probleme cu LCD-ul si a trebuit sa il returnez, am reusit sa improvizez cu ajutorul unui server de web realizat cu ajutorul ESP8266. Utilizatorul isi poate vedea rezultatele actiunilor sale urmarind shell-ul expus in server-ul web care se actualizeaza la fiecare secunda.

Concluzii

Desi este un proiect relativ simplu, acesta poate veni la indemana cand, din motive oarecare, un utilizator nu are la indemana un laptop sau calculator sau pur si simplu se intampla sa aiba nevoia de a stoca ceva pe un mediu de stocare nonvolatil.

O aplicatie practica a acestui proiect ar putea fi un sistem integrat de Journaling.

Download

sd_card_buddy.zip

Jurnal

- 31.05.2022 - Discutia cu asistentul legata de alegerea temei proiectului
- 08.05.2022 - Crearea paginii wiki: Introducere, Diagrama Block, Lista componente
- 12.05.2022 - Depistarea unor probleme cu LCD-ul.
- 14.05.2022 - Proiectarea temei folosind Serial Monitor pana cand imi dau seama ce sa fac cu LCD-ul.
- 22.05.2022 - Gasirea unui workaround pentru problema cu LCD: expunerea shell-ului se va face prin ESP8266 Web Server, interconectat prin seriala cu Arduino Uno. Asistentul a confirmat ca este OK cu aceasta modificare.
- 27.05.2022 - Terminarea proiectului din punct de vedere Software
- 01.06.2022 - Terminarea paginii de wiki.

Bibliografie/Resurse

Resurse Software:

- https://www.arduino.cc/reference/en/libraries/sd/
- https://randomnerdtutorials.com/esp8266-web-server/

Resurse Hardware:

- https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf
- https://components101.com/sites/default/files/component_datasheet/Micro-SD-Card-Module-Datasheet.pdf
- https://components101.com/sites/default/files/component_datasheet/ESP8266-NodeMCU-Datasheet.pdf
- https://www.instructables.com/How-to-Multiple-Buttons-on-1-Analog-Pin-Arduino-Tu/

Export to PDF