

Mașină de scris - Preoteasa Mircea-Costin 331CA

Introducere

Prezentarea pe scurt a proiectului vostru:

- ce face
- care este scopul lui
- care a fost ideea de la care ați pornit
- de ce credeți că este util pentru alții și pentru voi

Proiectul propus presupune simularea unei mașini de scris al cărei element central este un cap de scriere ce execută anumite mișcări pentru a desena caractere pe o bandă de hartie. Utilizatorul interacționează cu mașina prin intermediul unor butoane, introducând un caracter într-o anumită codificare. Capul de scriere coboară până la nivelul de contact cu banda de scriere, realizează mișcările necesare caracterului selectat, iar apoi realizează o mișcare la dreapta iar procesul se continuă. Scopul proiectului este ca mișcările realizate de capul de scriere să fie suficient de precise astfel încât caracterele scrise pe banda de hartie să fie inteligibile, iar utilizatorul să poată folosi cu ușurință butoanele pentru a introduce un text.

Proiectul a pornit de la o idee mai complexă, anume o mașină Turing ce ar putea efectua și operații de citire pe bandă, astfel fiind posibilă executia unui program, însă fiind dat timpul relativ scurt realizării acestui proiect a fost redus la o variantă simplificată.

În principiu, proiectul este un proof-of-concept, întrucât tehnologia modernă permite metode mult mai eficiente de a transpune pe hartie anumite informații. Totuși, proiectul poate fi folosit pentru a traduce în "cleartext" anumite codificări într-un mod mai apropiat de modul în care au fost concepute (de exemplu, codul Morse ar putea fi reprezentat de apăsări lungi și scurte de buton, față de reprezentarea abstractă prin linii și puncte)

Descriere generală

O schemă bloc cu toate modulele proiectului vostru, atât software cât și hardware însoțită de o descriere a acestora precum și a modului în care interacționează.

Exemplu de schemă bloc: <http://www.robs-projects.com/mp3proj/newplayer.html>



Hardware Design

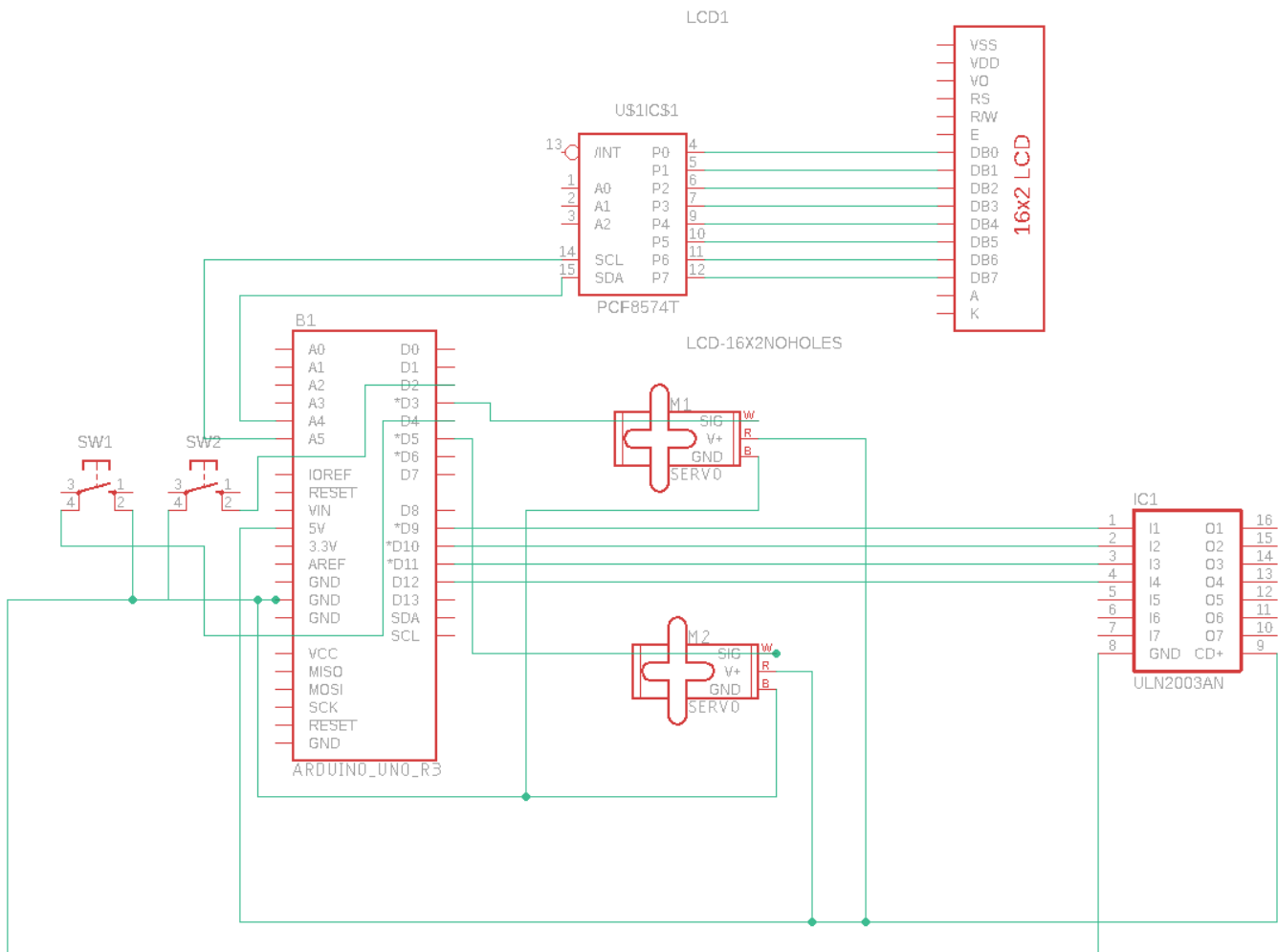
Aici puneți tot ce ține de hardware design:

- listă de piese
- scheme electrice (se pot lua și de pe Internet și din datasheet-uri, e.g. <http://www.captain.at/electronic-atmega16-mmc-schematic.png>)
- diagrame de semnal
- rezultatele simulării

Piesele folosite în cadrul proiectului sunt:

- Arduino Uno
- Breadboard
- 2 butoane
- 2 servomotoare
- Bandă de hârtie
- Pix
- Un motor pas cu pas (stepper)
- Roți dințate
- O curea închisă
- Un ecran LCD
- 3 piese suport printate 3D

Schemă electrică:



Pentru a putea monta piesele de bază am printat 3D mai multe piese “suport”. Banda de hartie se va misca in jurul unui suport ce are la un capat montat motorul stepper si o roata dintata, si o alta roata dintata la celalalt capat, intre cele doua fiind trasa o curea inchisa. Astfel, prin actionarea servomotorului se va misca cureaua si deci si banda de hartie de deasupra ei.

La baza suportului se va plasa un servomotor. De aceasta va fi legat un suport ce va contine un alt servomotor, de acesta fiind legat suportul de pix. Prin actionarea primului servomotor, se va putea muta pixul pana fie in pozitie ridicata (aproximativ 50 de grade), respectiv la nivelul foii (aproximativ 80 de grade). Prin actionarea celui de-al doilea servomotor se va putea controla pozitia verticala a pixului pe foaie. Astfel, prin actionarea celor 3 motoare vom putea cu usurinta ridica/cobori pixul pe foaie si muta stanga/dreapta.

Software Design

Descrierea codului aplicației (firmware):

- mediu de dezvoltare (if any) (e.g. AVR Studio, CodeVisionAVR)
- librării și surse 3rd-party (e.g. Procyon AVRlib)
- algoritmi și structuri pe care plănuieți să le implementați
- (etapa 3) surse și funcții implementate

Pentru dezvoltare am folosit Arduino IDE:

Biblioteci utilizate

- Servo - pentru controlul servomotoarelor
- LiquidCrystal-I2C - afisare pe LCD

Vom avea doua butoane, unul roșu și unul negru. Prin apăsarea scurtă a butonului negru (trecerea de pe HIGH pe LOW a pinului asociat butonului) vom adăuga în codul curent un punct, iar la apăsarea lungă vom adăuga o linie. Codul curent va fi afișat și pe LCD. La apăsarea scurtă a butonului roșu se va șterge ultimul caracter din codul curent iar la apăsarea lungă a acestui se va confirma codul. Dacă acest cod reprezintă o decodificare validă a unei litere atunci aceasta va fi scrisă pe foaie (se vor executa secvențele corespunzătoare ale celor 3 motoare pentru a scrie caracterul respectiv), se va acționa stepperul pentru a muta foaia mai la dreapta iar apoi se va curăța LCD-ul. În cazul în care codul Morse scris nu corespunde unui caracter valid se va afișa un mesaj de eroare pentru un anumit timp iar apoi se va curăța LCD-ul.

```
#include <LiquidCrystal_I2C.h>
#include <Servo.h>
#include <Wire.h>

#define BUTTON 2
#define RED_BUTTON 4
#define STEPPER_IN1 9
#define STEPPER_IN2 10
#define STEPPER_IN3 11
#define STEPPER_IN4 12
#define NUM_STEPPER_PINS 4
#define SERVO_PIN 3
#define SERVO_PIN2 5

#define MAX_CHARACTERS 10
#define ALPHABET_LEN 26

int step = 0;
int buttonState = HIGH;
int pressTime = 0;
int releaseTime = 0;

int redButtonState = HIGH;
int redButtonPressTime = 0;
int redButtonReleaseTime = 0;

Servo servo;
Servo servo2;
LiquidCrystal_I2C lcd(0x27, 16, 2);

char characters[] = {'A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K',
```

```

'L', 'M', 'N', 'O', 'P', 'Q', 'R', 'S', 'T', 'U',
    'V', 'W', 'X', 'Y', 'Z'};

char *morsecode[] = {".-", "-...", "-.-.", "-. .", ". .", ".-.-.", "-.-.", "-... .", ". . .",
    "-.-.", "-.-.", "-... .", "-.-.", "-. .", "-.-.", "-.-.", "-.-.",
    ". . .", ". . .", "-.", "-.-.", ". . .", ".-.-.", "-.-.", "-.-.",
    "-.-.", "-.-.", "-.-.", "-.-.", "-.-.", "-.-.", "-.-.", "-.-.",
    "-.-.", "-.-.", "-.-.", "-.-.", "-.-.", "-.-.", "-.-.", "-.-."};

char morseCode[MAX_CHARACTERS + 1];
int readCursor = 0;

int secondServoAngle = 160;
int firstServoAngle = 0;

void calibrate() {
    firstServoAngle = 0;
    secondServoAngle = 180;
}

void fullStep(int direction = 1) {
    for (int i = 0; i < NUM_STEPPER_PINS; i++) {
        if (i == step) {
            digitalWrite(STEPPER_IN1 + i, HIGH);
        } else {
            digitalWrite(STEPPER_IN1 + i, LOW);
        }
    }
    step += direction;
    if (step >= NUM_STEPPER_PINS) {
        step = 0;
    } else if (step < 0) {
        step = NUM_STEPPER_PINS - 1;
    }
    delay(2);
}

void executeStepper(int steps, int direction = 1) {
    for (int i = 0; i < steps; i++) {
        fullStep(direction);
    }
    delay(100);
}

void setup() {
    // put your setup code here, to run once:
    Wire.begin();
}

```

```
Serial.begin(9600);
lcd.begin();

pinMode(BUTTON, INPUT_PULLUP);
pinMode(RED_BUTTON, INPUT_PULLUP);
for (int i = 0; i < NUM_STEPPER_PINS; i++) {
    pinMode(STEPPER_IN1 + i, OUTPUT);
}

servo.attach(SERVO_PIN);
servo.write(0);

servo2.attach(SERVO_PIN2);
servo2.write(180);

lcd.clear();
lcd.blink();
}

// Rotates specified servo to a new angle
void rotateServo(Servo *servo, int *servoAngle, int endAngle) {
    int direction = 1;
    if (endAngle < *servoAngle) {
        direction = -1;
    }

    while (direction * (*servoAngle) < direction * endAngle) {
        *servoAngle += direction;
        servo->write(*servoAngle);
        delay(10);
    }
}

// Offsets the servo angle by specified value
void rotateServoOffset(Servo *servo, int *servoAngle, int offset) {
    rotateServo(servo, servoAngle, (*servoAngle) + offset);
    delay(200);
}

void penDown() {
    rotateServo(&servo, &firstServoAngle, 80);
    delay(2000);
}

void penUp() {
    rotateServo(&servo, &firstServoAngle, 60);
    delay(2000);
}
```

```
void putPoint() {
    penDown();
    delay(100);
    executeStepper(75, 1);
    penUp();
}

void putLine() {
    penDown();
    delay(100);
    for (int i = 0; i < 400; i++) {
        fullStep();
    }
    penUp();
}

void putA() {
    penDown();

    executeStepper(250, 1);
    rotateServoOffset(&servo2, &secondServoAngle, -10);

    penUp();
    executeStepper(250, -1);

    penDown();
    rotateServoOffset(&servo2, &secondServoAngle, 5);

    executeStepper(250, 1);

    penUp();

    executeStepper(250, -1);

    penDown();
    rotateServoOffset(&servo2, &secondServoAngle, 5);

    penUp();

    executeStepper(250, 1);
}

void putE() {
    penDown();

    executeStepper(250, 1);

    penUp();
```

```
executeStepper(250, -1);

penDown();
rotateServoOffset(&servo2, &secondServoAngle, -5);

executeStepper(250, 1);

penUp();

executeStepper(250, -1);

penDown();

rotateServoOffset(&servo2, &secondServoAngle, -5);

penDown();

executeStepper(250, 1);

penUp();
rotateServoOffset(&servo2, &secondServoAngle, 10);

}

void putL() {
    penDown();

    rotateServo(&servo2, &secondServoAngle, secondServoAngle - 10);
    delay(200);
    executeStepper(250, 1);
    penUp();
    rotateServo(&servo2, &secondServoAngle, secondServoAngle + 10);
    delay(200);
}

void put0() {
    penDown();

    executeStepper(250, 1);
    rotateServo(&servo2, &secondServoAngle, secondServoAngle - 10);
    delay(200);

    executeStepper(250, -1);
```



```
    rotateServo(&servo2, &secondServoAngle, secondServoAngle + 10);
    delay(200);

    penUp();

    executeStepper(250, 1);
}

void putC() {
    penDown();
    executeStepper(250, 1);
    penUp();
    rotateServo(&servo2, &secondServoAngle, secondServoAngle - 10);
    delay(200);

    penDown();

    executeStepper(250, -1);

    rotateServo(&servo2, &secondServoAngle, secondServoAngle + 10);
    delay(200);

    penUp();

    executeStepper(250, 1);
}

void putU() {
    penDown();
    rotateServo(&servo2, &secondServoAngle, secondServoAngle - 10);
    delay(200);

    executeStepper(250, 1);

    rotateServo(&servo2, &secondServoAngle, secondServoAngle + 10);
    delay(200);

    penUp();
}

void putP() {
    rotateServo(&servo2, &secondServoAngle, secondServoAngle - 10);
    delay(200);
```

```
penDown();

rotateServo(&servo2, &secondServoAngle, secondServoAngle + 10);
delay(200);

executeStepper(125, 1);

rotateServo(&servo2, &secondServoAngle, secondServoAngle - 5);
delay(200);

executeStepper(125, -1);

penUp();

rotateServo(&servo2, &secondServoAngle, secondServoAngle + 5);
executeStepper(250, 1);
}

void putN() {
    penDown();
    rotateServoOffset(&servo2, &secondServoAngle, -10);
    penUp();
    rotateServoOffset(&servo2, &secondServoAngle, 10);

    penDown();
    executeStepper(125, 1);
    rotateServoOffset(&servo2, &secondServoAngle, -10);
    executeStepper(125, 1);

    rotateServoOffset(&servo2, &secondServoAngle, +10);

    penUp();
}

void putT() {
    penDown();

    executeStepper(125, 1);
    rotateServoOffset(&servo2, &secondServoAngle, -10);

    penUp();
```

```
    rotateServoOffset(&servo2, &secondServoAngle, 10);

    penDown();

    executeStepper(125, 1);

    penUp();

}

void putH() {
    penDown();

    rotateServoOffset(&servo2, &secondServoAngle, -5);
    executeStepper(250, 1);

    penUp();
    executeStepper(250, -1);

    penDown();
    rotateServoOffset(&servo2, &secondServoAngle, -5);

    penUp();

    executeStepper(250, 1);
    penDown();

    rotateServoOffset(&servo2, &secondServoAngle, 10);

    penUp();
}

void putI() {
    executeStepper(125, 1);

    penDown();
    rotateServo(&servo2, &secondServoAngle, secondServoAngle - 10);
    delay(200);

    penUp();

    rotateServo(&servo2, &secondServoAngle, secondServoAngle + 10);
    delay(200);

    executeStepper(125, 1);
}

void putM() {
```

```
penDown();
executeStepper(83, 1);
rotateServoOffset(&servo2, &secondServoAngle, -5);
executeStepper(83, 1);
rotateServoOffset(&servo2, &secondServoAngle, 5);
executeStepper(83, 1);
rotateServoOffset(&servo2, &secondServoAngle, -10);

penUp();

executeStepper(249, -1);

penDown();
rotateServoOffset(&servo2, &secondServoAngle, 10);

penUp();

executeStepper(249, 1);
}

// Pointers to the put functions
void (*putters[]) (void) = {
    putA, // A
    NULL, // B
    putC, // C
    NULL, // D
    putE, // E
    NULL, // F
    NULL, // G
    putH, // H
    putI, // I
    NULL, // J
    NULL, // K
    putL, // L
    putM, // M
    NULL, // N
    putO, // O
    putP, // P
    NULL, // Q
    NULL, // R
    NULL, // S
    putT, // T
    putU, // U
    NULL, // V
    NULL, // W
    NULL, // X
    NULL, // Y
    NULL // Z
};
```

```
void nextLetter() {
    executeStepper(100, 1);
}

char decodeMorseCode(char *code) {
    for (int i = 0; i < ALPHABET_LEN; i++) {
        if (!strcmp(code, morsecode[i])) {
            return characters[i];
        }
    }

    return 0;
}

void readMorseCode() {
    int newButtonState = digitalRead(BUTTON);
    int newRedButtonState = digitalRead(RED_BUTTON);
    if (newButtonState == LOW && buttonState == HIGH) {
        pressTime = millis();
    } else if (newButtonState == HIGH && buttonState == LOW) {
        releaseTime = millis();

        Serial.println(readCursor);

        if (readCursor < MAX_CHARACTERS) {
            // long press
            if (releaseTime - pressTime >= 1000) {
                morseCode[readCursor++] = '-';
                lcd.print("-");
                //lcd.flush();
            } else {
                // short press
                morseCode[readCursor++] = '.';
                lcd.print(".");
                //lcd.flush();
            }
        }
    }

    if (newRedButtonState == LOW && redButtonState == HIGH) {
        redButtonPresTime = millis();
    } else if (newRedButtonState == HIGH && redButtonState == LOW) {
        redButtonReleaseTime = millis();
        // long press
        if (redButtonReleaseTime - redButtonPresTime >= 1000) {
            morseCode[readCursor++] = '\0';
        }
    }
}
```

```
char decoded = decodeMorseCode(morseCode);

lcd.clear();
if (decoded != 0 && putters[decoded - 'A'] != NULL) {
    lcd.print(decoded);
    putters[decoded - 'A']();
} else {
    lcd.print("Unknown!");
    delay(5000);
}
readCursor = 0;
lcd.clear();
} else {
    // short press

    if (readCursor > 0) {
        readCursor--;
        lcd.setCursor(readCursor, 0);
        lcd.print(" ");
        lcd.setCursor(readCursor, 0);
    }
}

buttonState = newButtonState;
redButtonState = newRedButtonState;
}

void loop() {
    readMorseCode();

    servo.write(firstServoAngle);
    servo2.write(secondServoAngle);
}
```

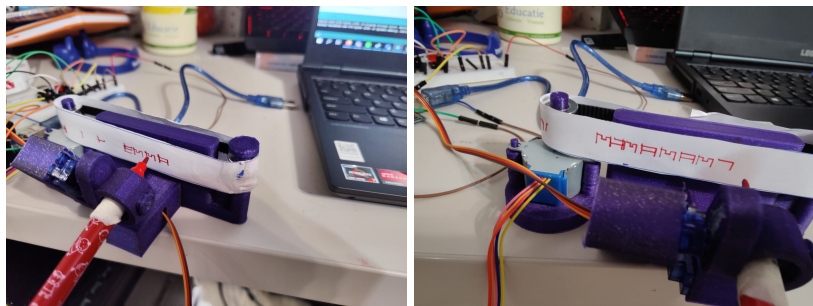
Rezultate Obținute

Care au fost rezultatele obținute în urma realizării proiectului vostru.

Din cauza unor dificultati tehnice (defectarea unor piese printate 3d si nevoia de a le reprinta), au putut sa fie implementate in timp doar desenarea urmatoarelor litere:

Versiunea 1 (puncte si linii): <https://youtu.be/CugPcJ30PQg>

Versiunea 2 (litere):



Concluzii

Proiectul este funcțional, în stadiul de proof-of-concept. Din cauza unor dificultăți tehnice (defectarea unor piese printate 3d și nevoia de a le reprinta), au putut să fie implementate în timp doar desenarea unui număr limitat de litere, o continuare imediată fiind reprezentată de implementarea tuturor literelor alfabetului latin. Pe termen mai lung, s-ar putea adăuga un sistem mai ușor de alimentare cu benzi de hârtie și sau un sistem de ștergere al conținutului benzii

Download

O arhivă (sau mai multe dacă este cazul) cu fișierele obținute în urma realizării proiectului: surse, scheme, etc. Un fișier README, un ChangeLog, un script de compilare și copiere automată pe uC crează întotdeauna o impresie bună .

Fișierele se încarcă pe wiki folosind facilitatea **Add Images or other files**. Namespace-ul în care se încarcă fișierele este de tipul **:pm:prj20??:c?** sau **:pm:prj20??:c?:nume_student** (dacă este cazul).
Exemplu: Dumitru Alin, 331CC → **:pm:prj2009:cc:dumitru_alin**.

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** și **Resurse Hardware**.

Datasheets:

<https://www.mouser.com/datasheet/2/758/stepd-01-data-sheet-1143075.pdf> - motor pas cu pas

<https://datasheetspdf.com/pdf-file/866417/LONGTECHOPTICS/LCM1602A/1> - LCD

<https://www.alldatasheet.com/datasheet-pdf/pdf/424340/PHILIPS/PCF8574T.html> - modul I2C pentru LCD

<https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf> - Arduino UNO R3

http://www.wecl.com.hk/distribution/PDF/Robotics_IoT/58-01-9024.pdf - Servomotor

Useful videos and tutorials:

<https://www.youtube.com/watch?v=avrdDZD7qEQ>

<https://www.instructables.com/Arduino-Servo-Motors/>

<https://www.youtube.com/watch?v=EAEuxjtkumM>

[Export to PDF](#)

From:

<http://ocw.cs.pub.ro/courses/> - **CS Open CourseWare**

Permanent link:

<http://ocw.cs.pub.ro/courses/pm/prj2022/cristip/typewriter>



Last update: **2022/06/02 08:57**