Ceas cu tuburi Nixie

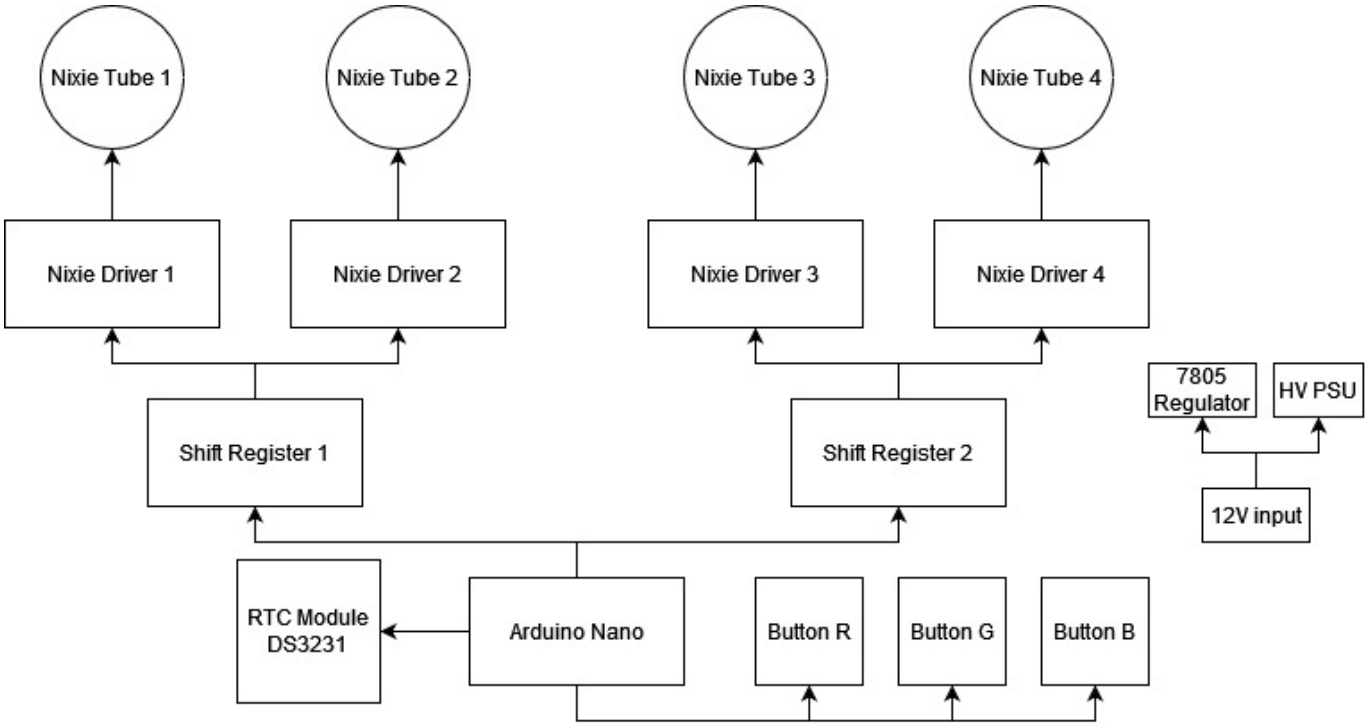
Nume: Dan Cosmin-Mihai

Grupa: 332CA

Introducere

Obiectivul acestui proiect este realizarea unui ceas cu tuburi Nixie cu mai multe functionalitati (2 moduri de afisare, mod standby) si carcasa transparenta din acrilic.

Descriere generală



Pentru a controla fiecare tub Nixie, este nevoie de cate un driver ce va primi codificat binar cifra care trebuie aprinsa. Un tub poate afisa 10 cifre, deci fiecare driver va avea 4 pini de intrare. Cei 16 pini de date rezultati vor fi controlati de 2 shift registere pentru a nu folosi prea multi pini de pe Arduino Nano.

Reglarea orei va fi realizata cu ajutorul a doua butoane (ora++, minut++). Un al treilea buton va fi folosit pentru a comuta intre cele doua moduri de afisare (ora:minute / minute:secunde) prin apasare scurta, iar prin apasare lunga va trece ceasul in modul standby (va opri tuburile nixie si sursa de tensiune).

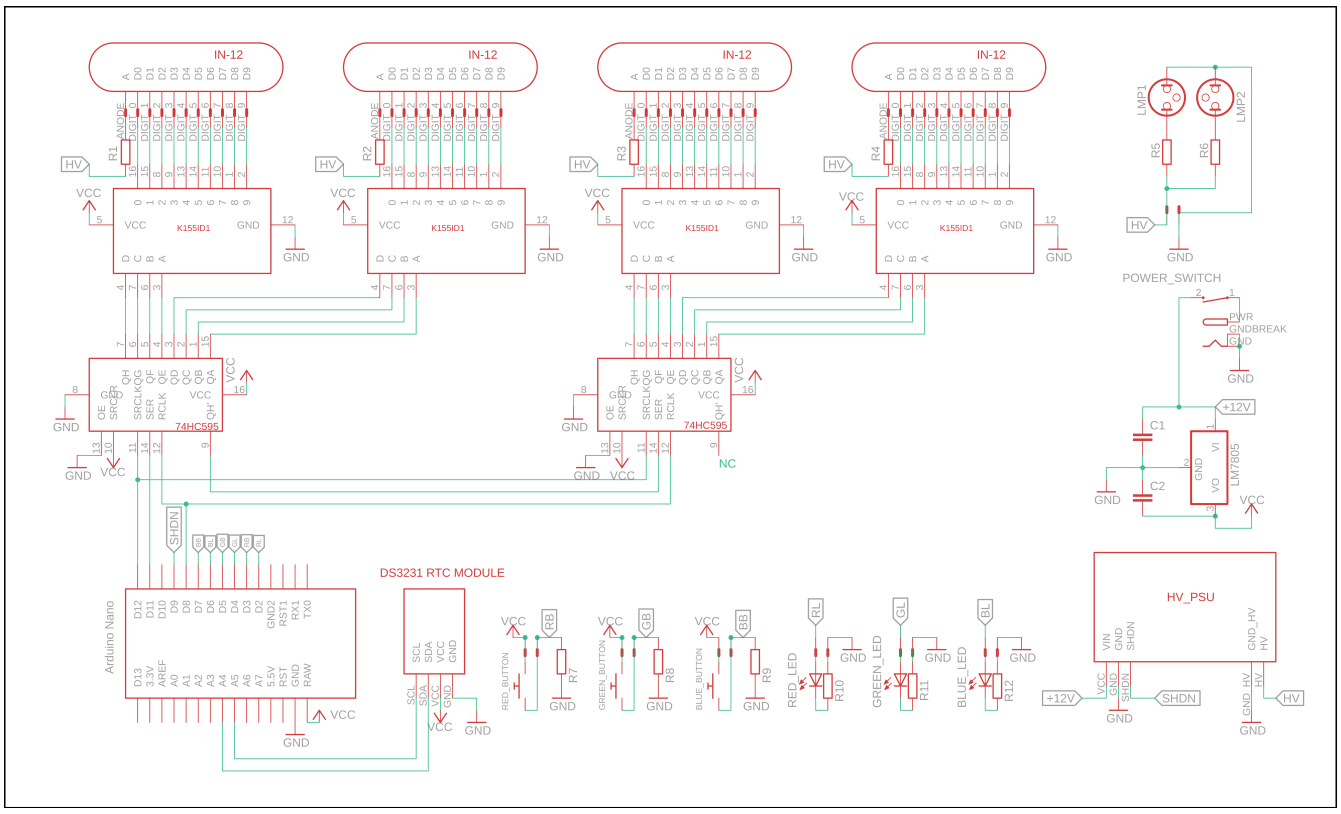
Ora exacta va fi pastrata de un modul RTC DS3231 ce va fi conectat la Arduino prin I2C.

Sistemul va fi alimentat la 12V. Sursa de tensiune inalta (170V) se va alimenta direct la 12V, iar partea de logica si control (Arduino si circuite integrate) vor fi alimentate la 5V, cu ajutorul unui regulator linear LM7805.

Hardware Design

Piese necesare:

- 4 x tub Nixie IN12b
- 4 x driver pentru tuburi Nixie K155ID1
- 2 x shift register 74HC595
- 4 x headere cu 11 pini
- 4 x cablu panglica cu 11 pini, mufat
- 6 x soclu DIP-16
- 2 x lampi neon
- 1 x LM7805
- 2 x condensator 100uF, 20V
- 1 x mufa DC mama
- 1 x mufa DC tata
- 3 x butoan cu led
- 7 x header cu 2 pini
- 4 x cablu cu 2 pini, mufat
- 1 x modul RTC DS3231
- 4 x rezistor 86k
- 2 x rezistor 220
- 1 x rezistor 100
- 3 x rezistor 10k
- 2 x header de pini mama
- 1 x placa prototipare 15×9 cm
- 1 x toggle switch



Unul dintre obiectivele principale ale acestui design este longevitatea tuburilor Nixie, motiv pentru care acestea sunt alimentate la un curent mai mic (1.5 mA) decat cel recomandat in datasheet (2.5 mA).

Carcasa a fost realizata din placi de acrilic si este formata din doua cutii: una pentru circuitul principal, ce are pe lateral butoane, mufa de alimentare si una pentru tuburi, ce este prinsa de prima cutie cu niste balamale, astfel incat sa poata fi ajustat unghiul la care sunt indreptate tuburile, similar unui ecran de laptop.

Software Design

Flow-ul general al buclei principale a programului este urmatorul:

- se obtine ora de la modulul RTC prin I2C
- in functie de modul de afiseare, se trimit datele necesare la shift registere
- se verifica daca au fost apasate butoanele
- se verifica daca ceasul ar trebui sa intre in modul de standby

DS3231 RTC

Modulul RTC comunica prin intermediul I2C, folosind functii din biblioteca Wire si DS3231. Codul care initializeaza modulul este:

```
        Wire.begin();
```

```
        DS3231_init(DS3231_CONTROL_INTCN);
```

Pentru a obtine ora de la RTC, se apeleaza functia DS3231_get(ts *t), populeaza campurile structului primit ca argument cu datele de la RTC. In cazul de fata, sunt necesare doar campurile pentru ora, minute si secunde.

Shift register

Trimiterea datelor la shift registere este realizata de urmatoarea functie:

```
void display_time() {
  uint16_t time = 0;

  //form time value to display
  if (!display_seconds)
    time |= ((((t.hour % 10) << 4) | (t.hour / 10)) << 8) | (((t.min % 10)
<< 4) | (t.min / 10));
  else
    time |= ((((t.min % 10) << 4) | (t.min / 10)) << 8) | (((t.sec % 10) <<
4) | (t.sec / 10));

  //shift out bits one by one, MSB first
  digitalWrite(latchPin, LOW);
  for (uint16_t i = 0; i < 16; i++) {
    digitalWrite(dataPin, !!(time & (1 << (15 - i))));

    digitalWrite(clockPin, HIGH);
    digitalWrite(clockPin, LOW);
  }
  digitalWrite(latchPin, HIGH);
}
```

In functie de modul de afisare selectat, este formata o variabila pe 16 biti, care sunt apoi trimisi spre shift register. Conform datasheet-ului pentru 75HC595, timpul de setup necesar pentru semnalul de intrare este de minim 19ns la 4.5V, iar timpul de propagare maxim este de 35 ns, insa Arduino functioneaza cu un ceas de 16MHz, deci perioada este de 62.5 ns, iar intarzierea introdusa de functia digitalWrite asigura timpii necesari astfel incat sa nu apara hazarde.

Butoane

Inputul de la cele trei butoane este trecut printr-un debouncer, dupa care este executata functia corespunzatoare fiecaruia (verde – ora++, albastru – minute++, rosu – mod display/ standby). De exemplu, pentru butonul verde, codul este urmatorul:

```
current_time = millis();
```

```
    if (current_time - last_pressed_blue > debounce_threshold) {
      if (digitalRead(blue_button) == HIGH) {
        increment_minute();
        last_pressed_blue = current_time;
        on_time = 0;
      }
    }
```

In cazul butonului rosu, pentru a distinge intre apasare scurta si lunga se masoara cat timp butonul este apasat. O apasare lunga (80ms) va opri sursa tuburilor Nixie si va pune microcontrollerul in modul de sleep, iar o apasare scurta va schimba modul de afisare (hhmm sau mmss). Pentru o mai buna interactiune cu utilizatorul, fiecare buton are un led in forma de inel ce se va aprinde cand este apasat si se va stinge cu un o intarziere de 500ms fata de momentul cand butonul este eliberat. Codul pentru led-ul butonului verde:

```
if (green_led && current_time - last_pressed_green > led_delay &&
digitalRead(green_button) == LOW) {
      green_led = 0;
      digitalWrite(green_button_led, LOW);
    }
```

Modul Standby

O a doua metoda prin care se asigura longevitatea tuburilor Nixie este modul de standby. Dupa o perioada de timp configurabila (implicit, 600s), sursa tuburilor Nixie se va opri automat. Pentru a masura aceasta perioada e timp, timer1 este configurat astfel incat sa genereze o intrerupere la fiecare secunda:

```
  TCCR1A = 0;
  TCCR1B = 0;
  TCNT1 = 0;

  OCR1A = 15624; //set compare value, 16MHz / 1024 = 15624
  TCCR1B |= 1<<WGM12; //ctc mode
  TCCR1B |= (5<<CS10); //set prescaler to 1024
  TIMSK1 |= 1<<OCIE1A; //enable interrupt on compare match

  ISR(TIMER1_COMPA_vect)
  {
    on_time++;
  }
```

In bucla principala este verificata valoarea contorului on_time la fiecare iteratie, iar atunci cand a trecut perioada setata, sursa se va opri:

```
if (on_time > display_on_time && !hv_enable) {
    hv_enable = HIGH;
    on_time = 0;
```

```
    digitalWrite(hv_enable_pin, hv_enable);
    goToSleep();
  }
```

Contorul este resetat la apasarea oricaruia dintre butoane pentru a preveni oprirea tuburilor atunci cand un utilizator interactioneaza cu ceasul.

Modul Sleep

Avand in vedere ca ora este retinuta de catre modulul RTC, in momentu in care sursa tuburilor este oprita, microcontrollerul nu mai este necesar, motiv pentru care ar fi mult mai eficient sa fie pus in modul sleep cu ajutorul functiilor din bibliotecile avr/sleep si avr/power.

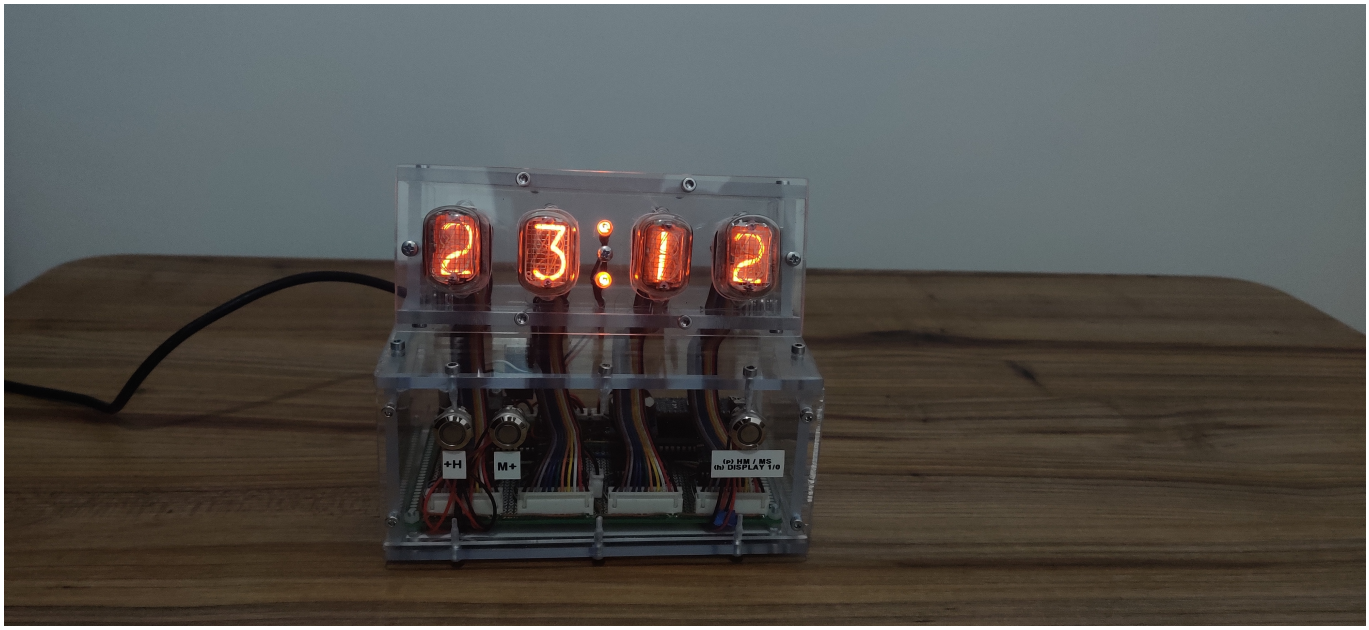
```
void goToSleep() {
  digitalWrite(red_button_led, LOW);
  red_led = 0;
  set_sleep_mode(SLEEP_MODE_PWR_DOWN);
  power_all_disable();  // power off ADC, Timer 0 and 1, serial interface
  sleep_enable();
  sleep_cpu();
  sleep_disable();
  power_all_enable();  // power everything back on
}
```

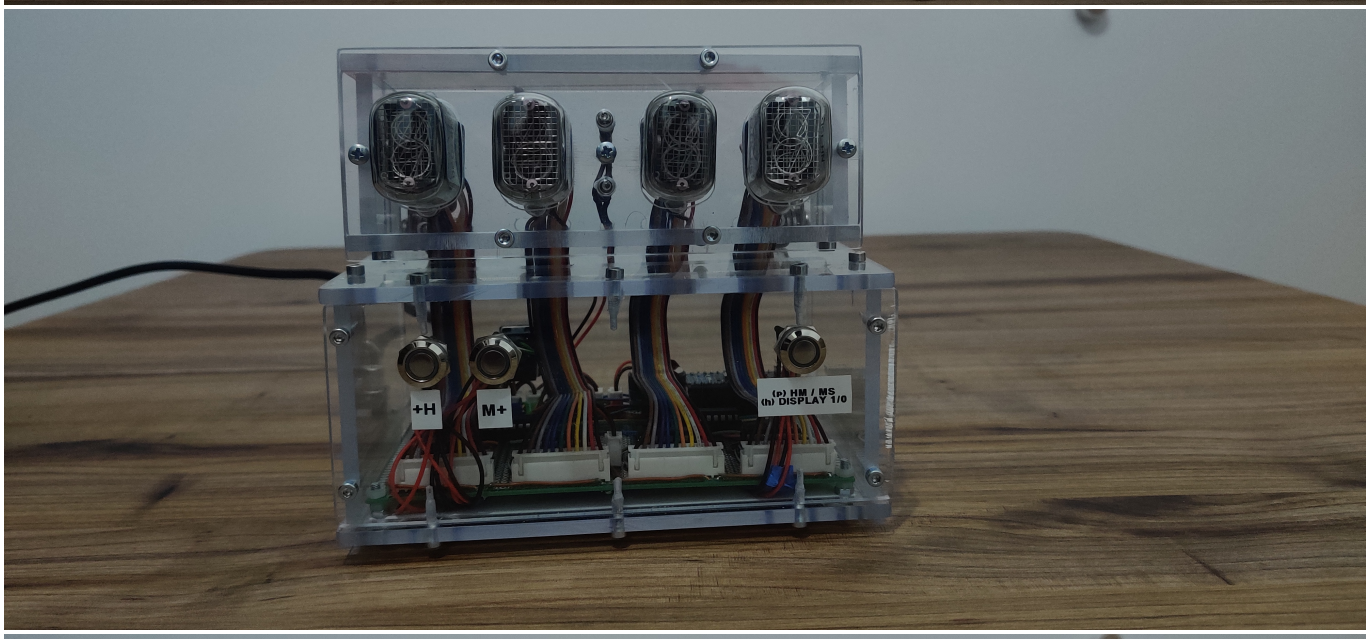
Microcontroller-ul executa codul pana la apelul functiei sleep_cpu(), unde intra in modul sleep si asteapta o intrerupere pentru a se trezi, dupa care reia executia codului. Pentru a folosi butonul rosu drept sursa pentru intrerupere externa, se activeaza PCINT19 (butonul rosu este legat la pinul PD3):

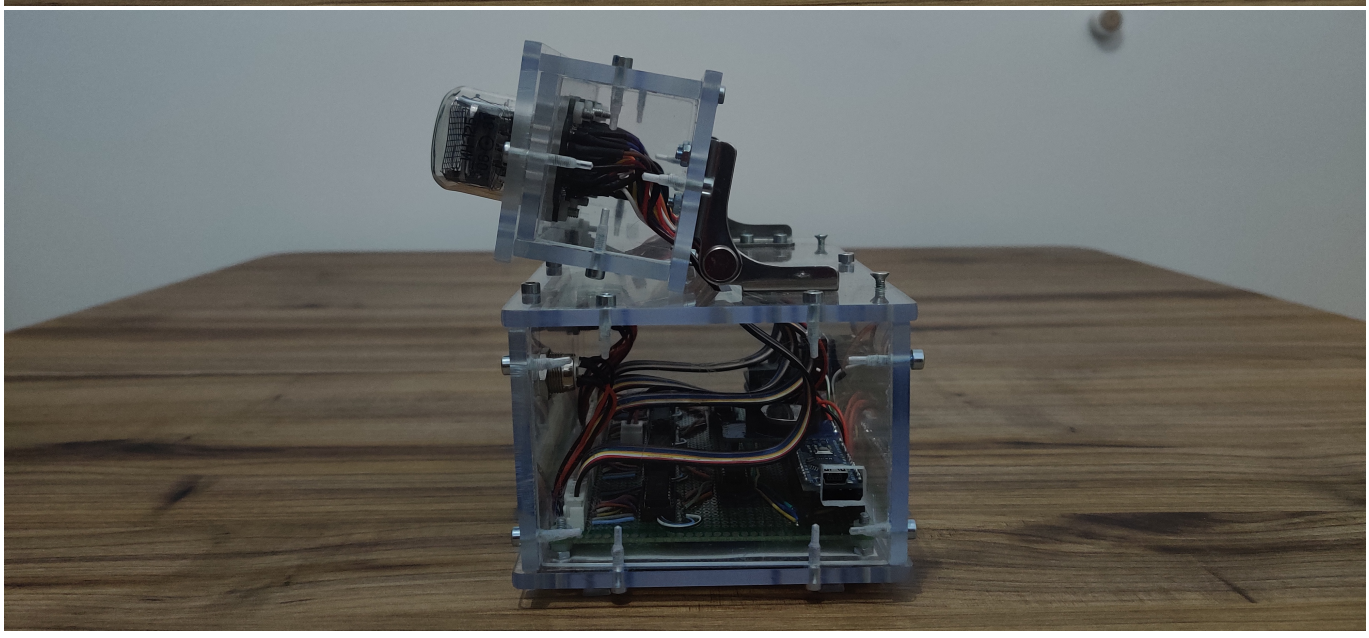
```
  PCICR |= B00000100;
  PCMSK2 |= B00001000;
```

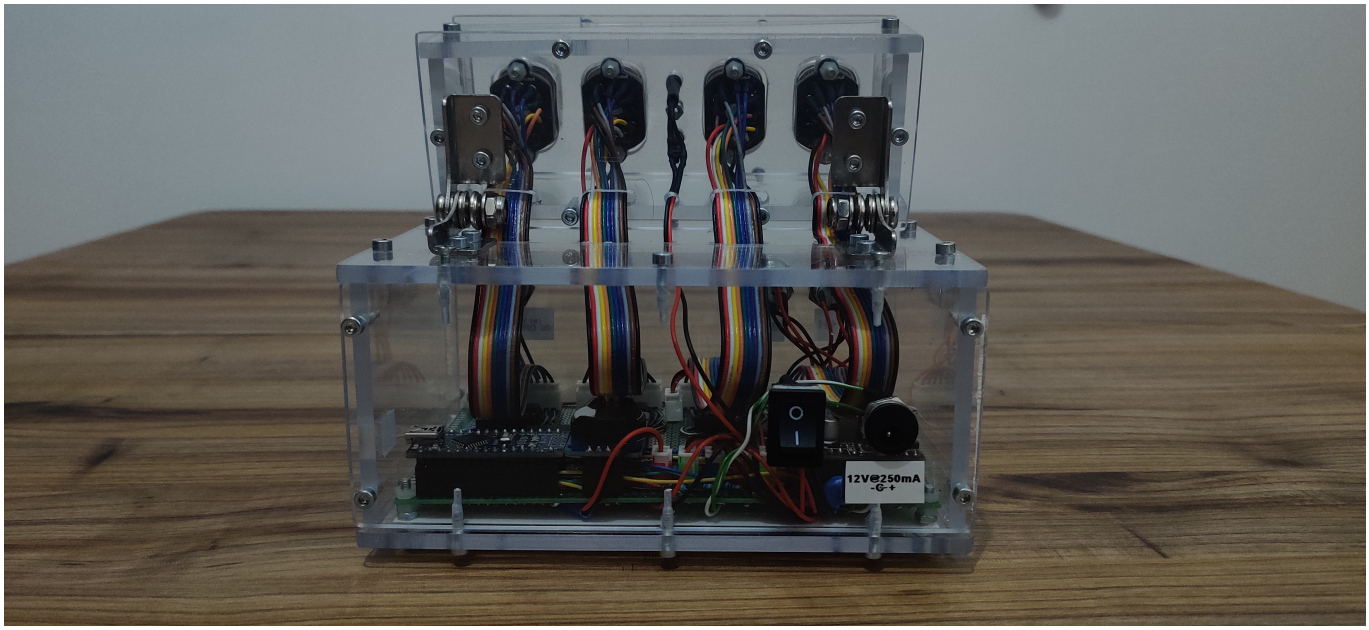
Rezultate Obţinute

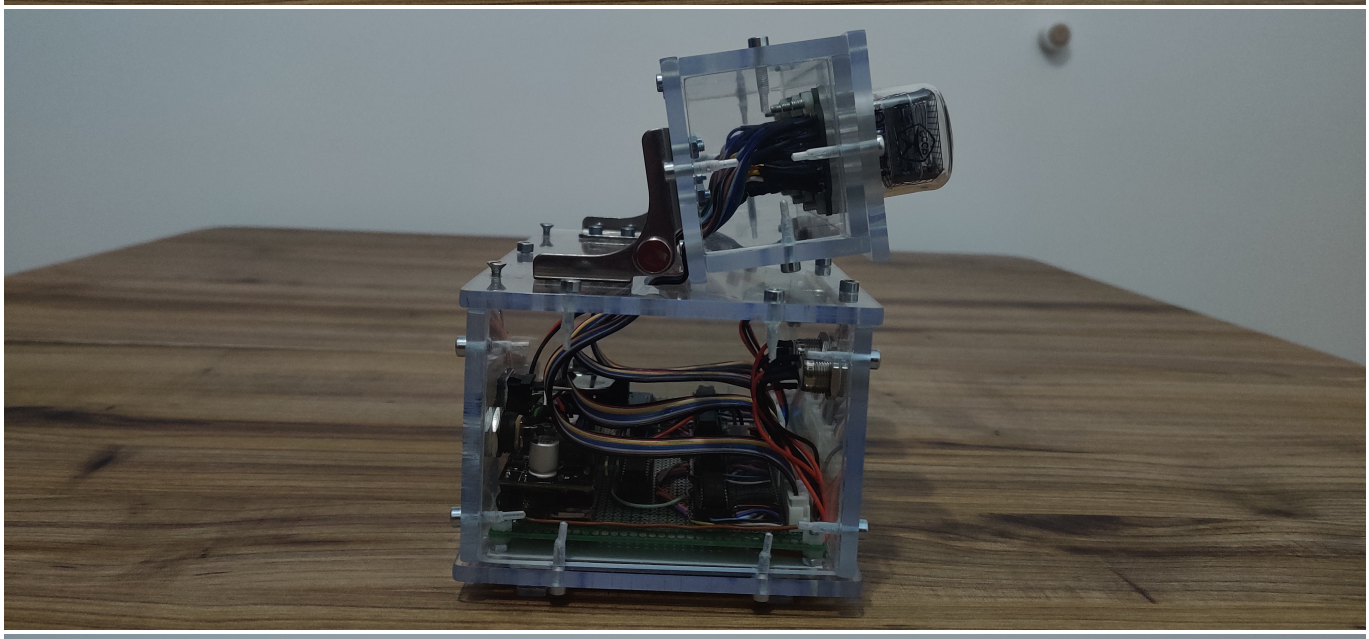
Video: https://www.youtube.com/watch?v=cYXRmdE6KUI

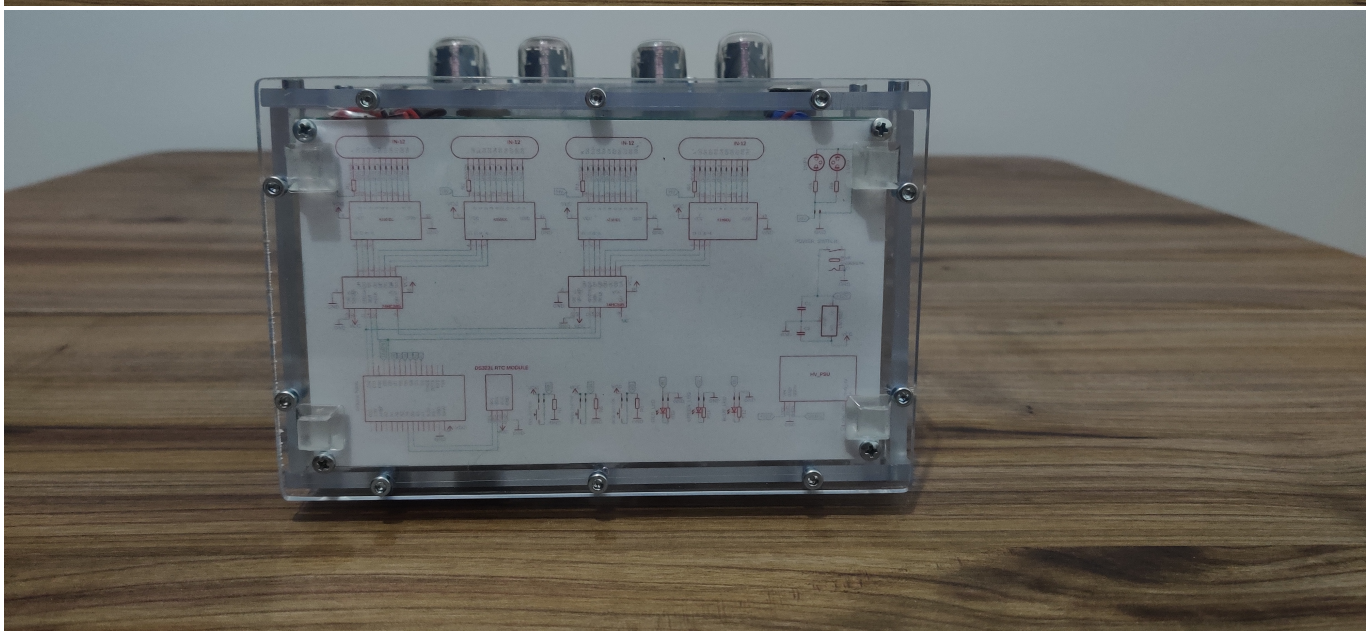

+H
M+
(p) HM / MS
(h) DISPLAY 1/0


+H
M+
(p) HM / MS
(h) DISPLAY 1/0




12V@250mA





Concluzii

Ceasul obtinut a indeplinit toate obiectivele proiectului. Asamblarea placii nu a fost simpla, insa constructia carcasei din acrilic (realizata complet manual) a fost cea mai dificila parte a proiectului, fiind zona in care am cea mai putina experienta.

Download

Cod sursa, diagrame si imagini

Bibliografie/Resurse

1. Datasheet tuburi Nixie IN12B: http://www.tube-tester.com/sites/nixie/dat_arch/IN-12A_IN-12B_03.pdf
2. Datasheet drivere tuburi K155ID1: https://eandc.ru/pdf/mikroskhema/k155id1.pdf
3. Datasheet shift register SN74HC595: https://www.ti.com/lit/ds/symlink/sn74hc595.pdf
4. Datasheet ATmega328P: https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

Export to PDF