Vlad-Costin NEDELCU (78559) - POV Clock

Autorul poate fi contactat la adresa: **Login pentru adresa**

Introducere

POV (Persistence of vision) este o iluzie optica ce are loc atunci cand perceptia vizuala a unui obiect nu inceteaza pentru ceva timp dupa ce razele de lumina pornind de la acesta au incetat sa mai intre in ochi. Astfel, acea perceptie persista pe retina.

In proiectarea mea, LED-urile se vor invarti cu o viteza foarte mare, astfel incat o persoana oarecare va percepe existenta mai multor LED-uri decat exista in realitate. Aprinderea lor se va face intr-o maniera sincronizata, pentru a da impresia afisarii de cifre. Folosind aceste cifre va fi indicata ora curenta.

Descriere generală

Atat placa de baza, cat si cea cu LED-uri vor fi fixate de motor prin intermediul unui suport din lemn. Toate componentele vor trebui fixate foarte bine pentru a nu se desprinde in timpul rotatiei rapide. Pentru alimentare, placuta de baza va fi conectata la o baterie externa prin USB (asezata tot pe suport), iar motorul la calculator sau direct la priza.

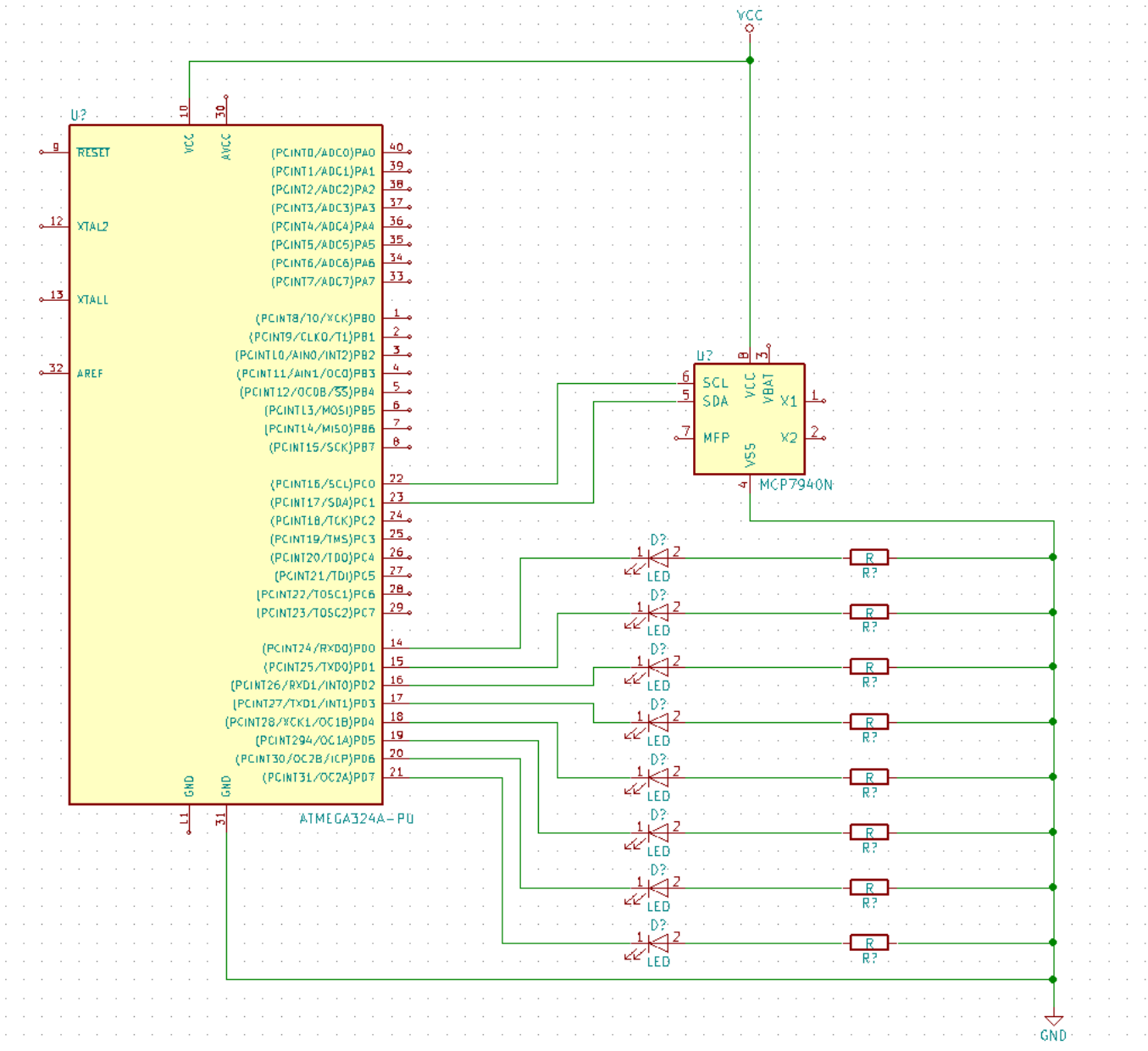
Pentru a reda ora curenta voi folosi un modul RTC ce va tine ora exacta intr-un registru accesibil din exterior. Modulul isi va actualiza ora intern la fiecare secunda indiferent daca placa este sau nu alimentata.

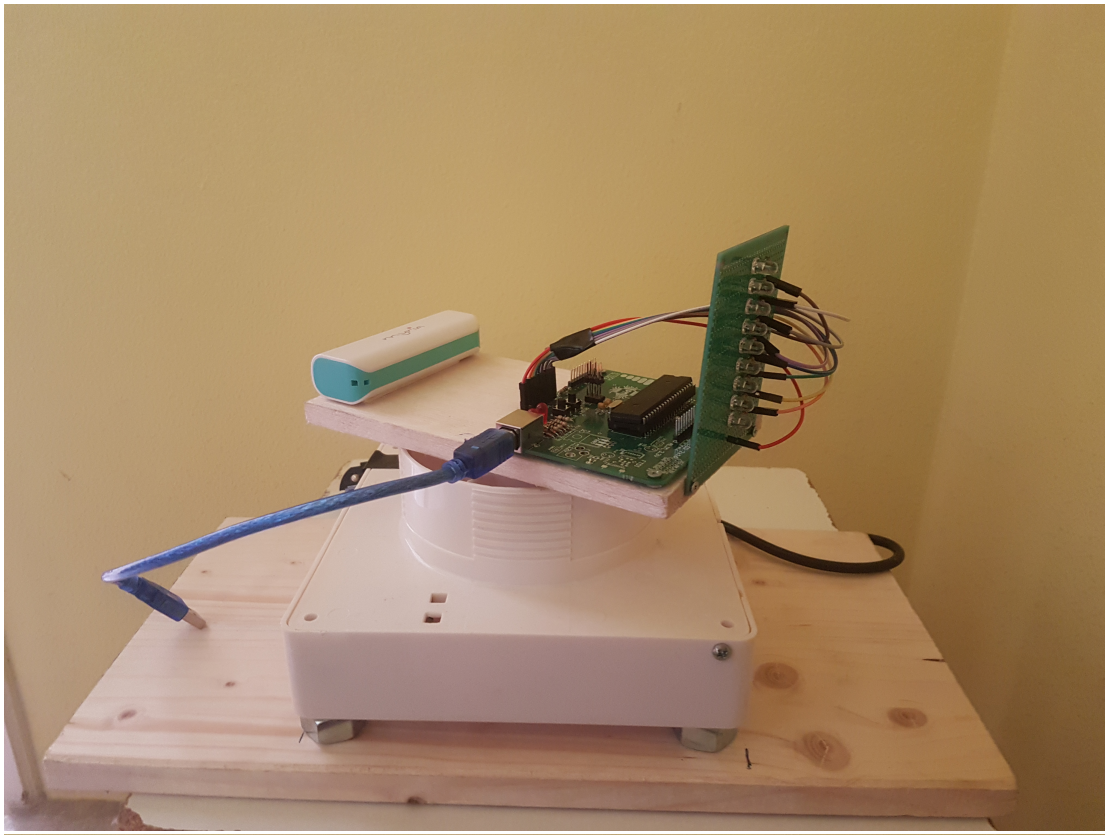
Hardware Design

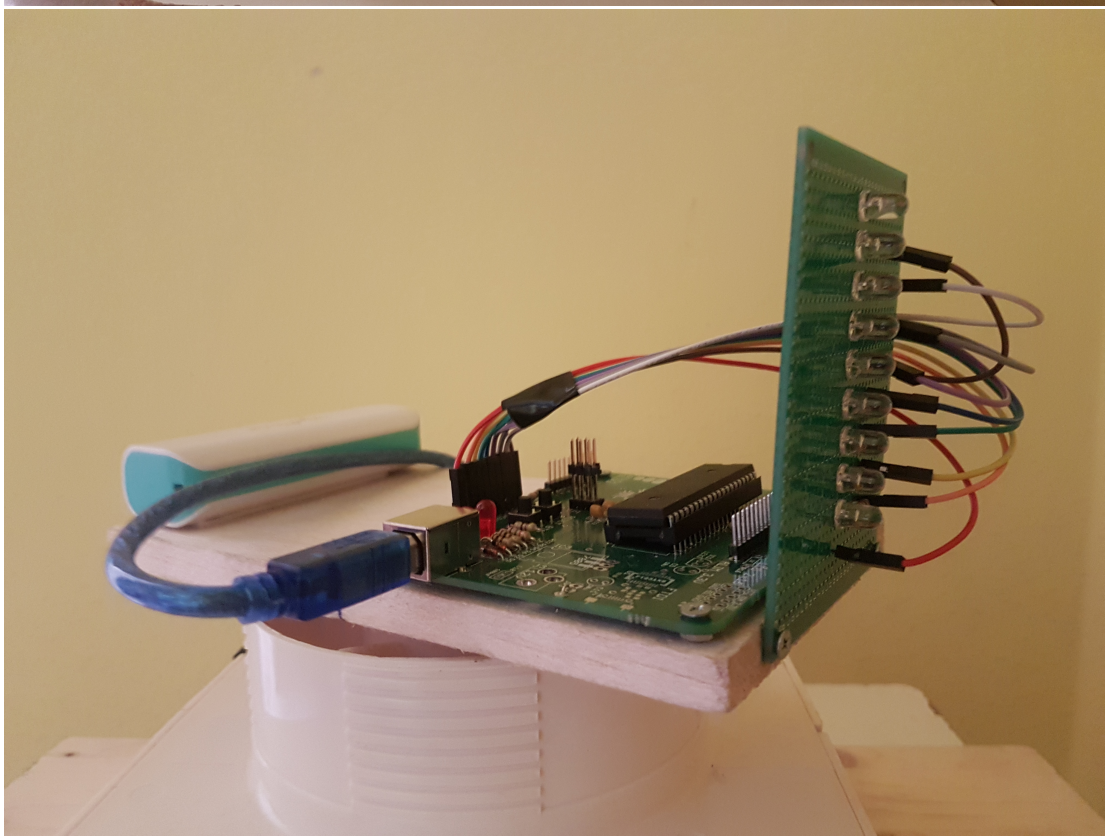
Componente	Cantitate
ATMega324 (placa de baza)	1
Componente de baza	1
Motor (de la un cooler de procesor sau ventilator)	1
Baterie externa USB	1
Suport de lemn	1
LED-uri colorate	8
Rezistente de 1k pentru LED-uri	8
Placa secundara (pentru montarea LED-urilor)	1

Modul RTC (Real Time Clock)	1

Schema electrica







Software Design

Codul aplicatiei a fost dezvoltat pe Windows, compilat cu avr-g++ si incarcat pe placa folosind interfata GUI a lui HIDBootFlash, iar principalele biblioteci folosite au fost *avr/io.h*, *util/delay.h* si *avr/interrupt.h*.

Cifrele vor fi formate dintr-o secventa ce contine 5 stari, fiecare stare avand o parte din cele 7 LED-uri aprinse. Pentru memorarea pattern-urilor de LED-uri ce vor forma cifrele am folosit o matrice. Valorile matricii vor putea fi atribuite direct registrului PORTD pentru simplitate.

```
unsigned char patterns[10][dimension] = {
                                    {0x7f, 0x41, 0x41, 0x41, 0x7f}, //0
                                    {0x00, 0x00, 0x7f, 0x00, 0x00}, //1
                                    {0x79, 0x49, 0x49, 0x49, 0x4f}, //2
                                    {0x49, 0x49, 0x49, 0x49, 0x7f}, //3
                                    {0x0f, 0x08, 0x08, 0x08, 0x7f}, //4
                                    {0x4f, 0x49, 0x49, 0x49, 0x79}, //5
                                    {0x7f, 0x49, 0x49, 0x49, 0x79}, //6
                                    {0x01, 0x01, 0x01, 0x01, 0x7f}, //7
                                    {0x7f, 0x49, 0x49, 0x49, 0x7f}, //8
                                    {0x4f, 0x49, 0x49, 0x49, 0x7f}  //9
                                        };
```

Functia **displayPattern** afiseaza o cifra folosind intrarea corespunzatoare din matricea de sus, iar intre fiecare configurare din secventa este introdus un delay pentru ca cifra sa apara mai clar.

```
void displayPattern(unsigned char* pattern, unsigned char dim) {
        unsigned char j;
        for (j = 0; j < dim; j++) {
                PORTD = pattern[j];
                _delay_ms(2);
                _delay_us(500);
        }
}
```

Timerul 1 este setat in modul *FastPWM* cu *top la OCR1A* si *prescaler 1024* astfel incat sa se obtina o intrerupere la fiecare secunda.

```
void initTimer() {
    TCCR1B = (3 << WGM12) | (5 << CS10);
    TCCR1A = (3 << WGM10);

    TIMSK1 = (1 << OCIE1A);
    OCR1A = 18825;
}
```

Cand intreruperea este activata va fi rulata functia **changeTime**, care actualizeaza ora curenta.

```
void changeTime() {
    secunda++;

    if (secunda == 60) {
        minut++;
        secunda = 0;

        if (minut == 60 ) {
```

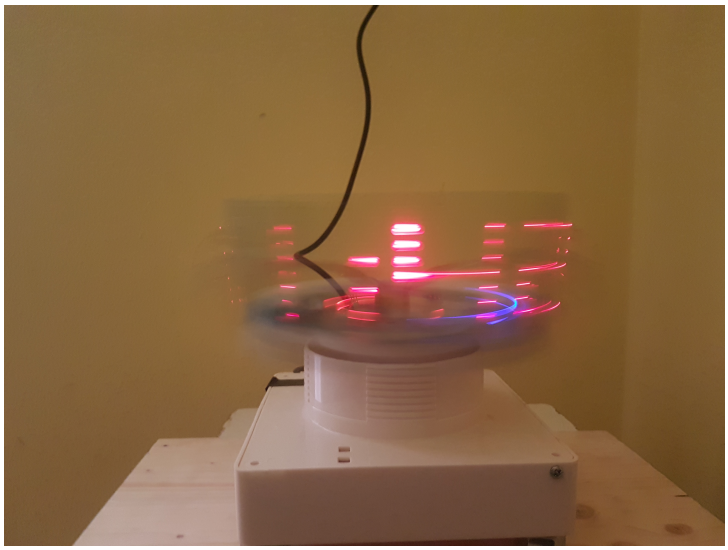
```
            ora++;
            minut = 0;

            if (ora == 12 )
                ora = 0;
        }
    }
}
```

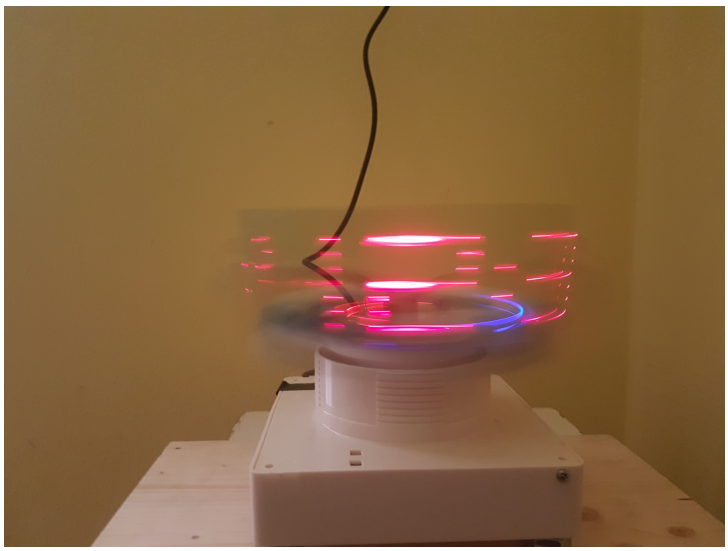
Dupa realizarea initializarilor, in bucla while(1) din main se vor afisa caracterele din componenta orei curente (HH:MM:SS = 8 caractere), fiecare caracter fiind urmat de o scurta perioada de timp in care se sting toate LED-urile (ca un fel de spatiu).

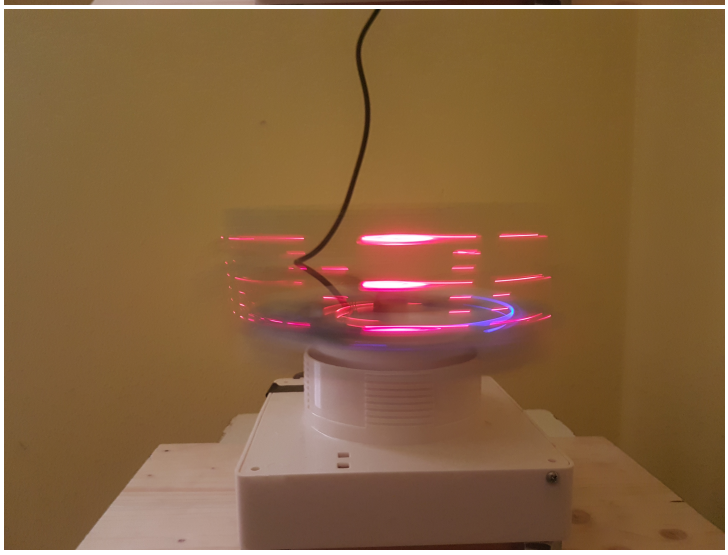
Rezultate Obţinute

Dupa mai multe incercari am reusit sa stabilizez ansamblul si sa obtin o viteza suficient de mare pentru ca ceasul sa poata aparea clar. Avand in vedere ca nu am avut un motor cu turaj reglabil textul nu sta fixat, se roteste cu timpul, dar acest lucru poate fi considerat si un feature in functie de cum privesti lucrurile In plus, precizia obtinuta pentru ceas folosind intreruperile este aproape perfecta.

In poze nu prea se vede bine din cauza ca am folosit LED-uri rosii







Concluzii

Sunt multumit de rezultatul pe care am reusit sa il obtin, cu mentiunea ca daca as fi folosit un motor reglabil as fi putut tine si textul fix. Principala dificultate am avut-o cu motorul, a trebuit sa incerc 3 pana sa gasesc unul suficient de puternic pentru a roti ansamblul la viteza dorita (si care sa poata fi si stabilizat).

Download

336cb_nedelcuvlad_pm2018_pov_clock.zip

Bibliografie/Resurse

Resurse Software

- Laboratoarele 2 si 3 de anul acesta
- Proiectele asemanatoare din alti ani

Resurse Hardware

- Datasheet ATmega324A doc8272.pdf

Documentația în format PDF