Proiectarea cu Microprocesoare
Curs 1

Andrei Voinescu

POLITEHNICA University of Bucharest

22 Februarie 2017

Ce este Proiectarea cu Microprocesoare?

- Proiectarea de sisteme care să conțină cel puțin un microprocesor
  - Ce este un microprocesor?
  - În zilele de azi, toate procesoarele sunt microprocesoare
- Considerăm un sistem cu microprocesor a fi orice de la un sistem de control pentru o mașină de spălat vase, la sistemul de navigație al unui avion, la o placă de bază pentru un PC
- Accentul se pune pe proiectare, ceea ce dă un caracter profund practic materiei
- Vrem să aducem la viață sisteme de calcul, și tocmai asta veți face în cadrul laboratorului și proiectului!

De ce facem Proiectarea cu Microprocesoare?

- Sisteme de calcul cu microprocesoare se găsesc peste tot
- Este important pentru o formație de inginer să știe ce se află în calculatorul lui, cum se interconectează componentele măcar la un nivel teoretic
- Proiectarea de mici dispozitive hardware este din ce în ce mai relevant în contextul tehnologic actual:
  - Legea lui Moore a încetinit [1], dar tendințele de miniaturizare continuă
  - Internet of Things este buzzword-ul decadei și propune miliarde de dispozitive mici interconectate
  - Aceste dispozitive ar fi construite atât de firme mari, startup-uri sau chiar hobby-iști

[1] https://www.technologyreview.com/s/601441/moores-law-is-dead-now-what/

Ce facem la Proiectarea cu Microprocesoare

- Curs
  - Prezentarea diferitelor periferice și metode de comunicare folosite în laborator
  - Metode de programare pentru microcontroller-ul de laborator
  - Proiectarea unei plăci *de bază*, bazată pe 8086 :)
  - Altele
- Laborator (primele 7 săptămâni)
  - Programarea pe microcontroller-e, aplicată pe Atmel AVR
  - O să învățați să folosiți perifericele disponibile pe ATMega324a
- Proiect (ultimele 7 săptămâni)
  - Construirea unui dispozitiv hardware cu o anumită funcție
    - Roboți
    - Jocuri pe LCD (Gameboy, nu Nintendo DS)
    - Mașini de făcut clătite, mașinuțe teleghidate, etc

- Nu tot conținutul va fi pe slide-uri
- Toate materialele pentru laborator și proiect se găsesc pe wiki la http://cs.curs.pub.ro/wiki/pm

Notare

- 5p Examen
- 1p Activitate Laborator
- 3p Proiect
- 1p Colocviu în săptămâna 7
- 1p Prezență
  - Sub formă de lucrări neanunțate cu subiect din cursul curent

- 3p din punctajul final
  - 20p wiki[2]
  - 20p placa de bază[3]
  - 30p implementare hardware
  - dintre care 5p pentru lucrat ordonat
  - 30p implementare software
  - dintre care 5p pentru cod scris ordonat
  - -20p nerespectare milestone-uri (5p x 4)
- Motivații pentru *
  - Nu vrem documentații fără proiect
  - Placa de bază este doar un punct intermediar, nu un scop în sine

[1]Dacă proiectul este dezvoltat dincolo de placa de bază

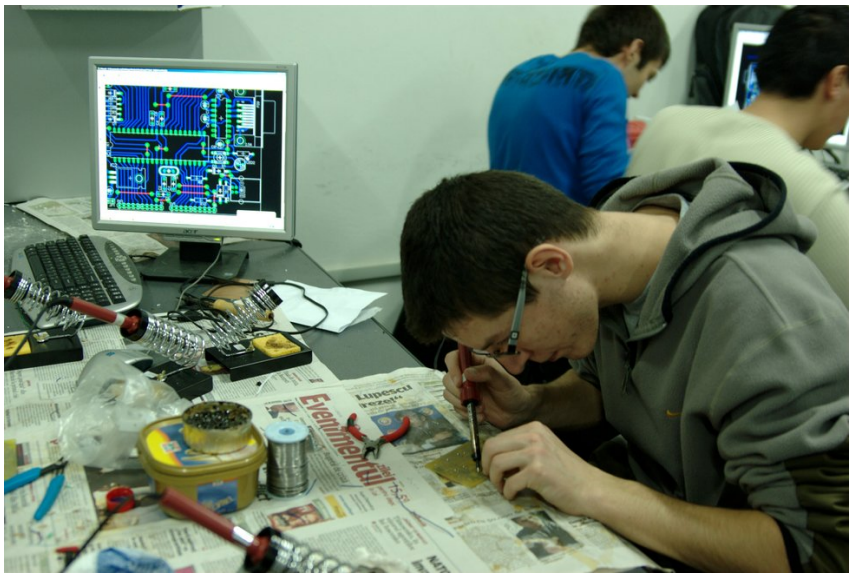
[2]Dacă proiectul este dezvoltat dincolo de placa de bază, exceptând placa de bază făcută pe placă de test (cu găurele), care este punctată ca atare

- 4 Milestone-uri
  - Sunt aici ca să vă ajute!
  - de pe wiki:
    - *"PS:Nu lasati proiectul pe ultima suta de metri.Lucrati din timp!"*
    - *"trebuia sa ma apuc de proiect cu mult mult timp inainte"*
  - Încercați să vă țineți de ele, dar nu fiți limitați de ele
    - Puteți să le îndepliniți și mai devreme!

- Reprezintă ocazia unică din facultate de a construi un dispozitiv hardware
- Beneficiați de ajutorul echipei de PM în atingerea acestui scop
- Are un preț asociat
  - Suportul (PCB) pentru placa de bază este 7 RON
  - Piesele pentru placa de bază $\simeq$ 40 RON
  - Piesele pentru restul proiectului depind de ce alegeți
- Considerăm că prețul este justificat pentru cât învățați din acest proiect

- Proiectul de PM există din timpuri imemoriale
  - No really, chiar nu știu de când e, 15 ani pe puțin
  - Cronicile pe wiki încep abia din 2009
- Aveți pe wiki un Hall of Fame
  - `http://cs.curs.pub.ro/wiki/pm/halloffame`

- Discuții cu asistentul de proiect
- Lipit, testat hardware
- Scris cod, debugging
- **ATENȚIE** - vă rugăm insistent să păstrați pe cât posibil ordinea în laborator



Calendar

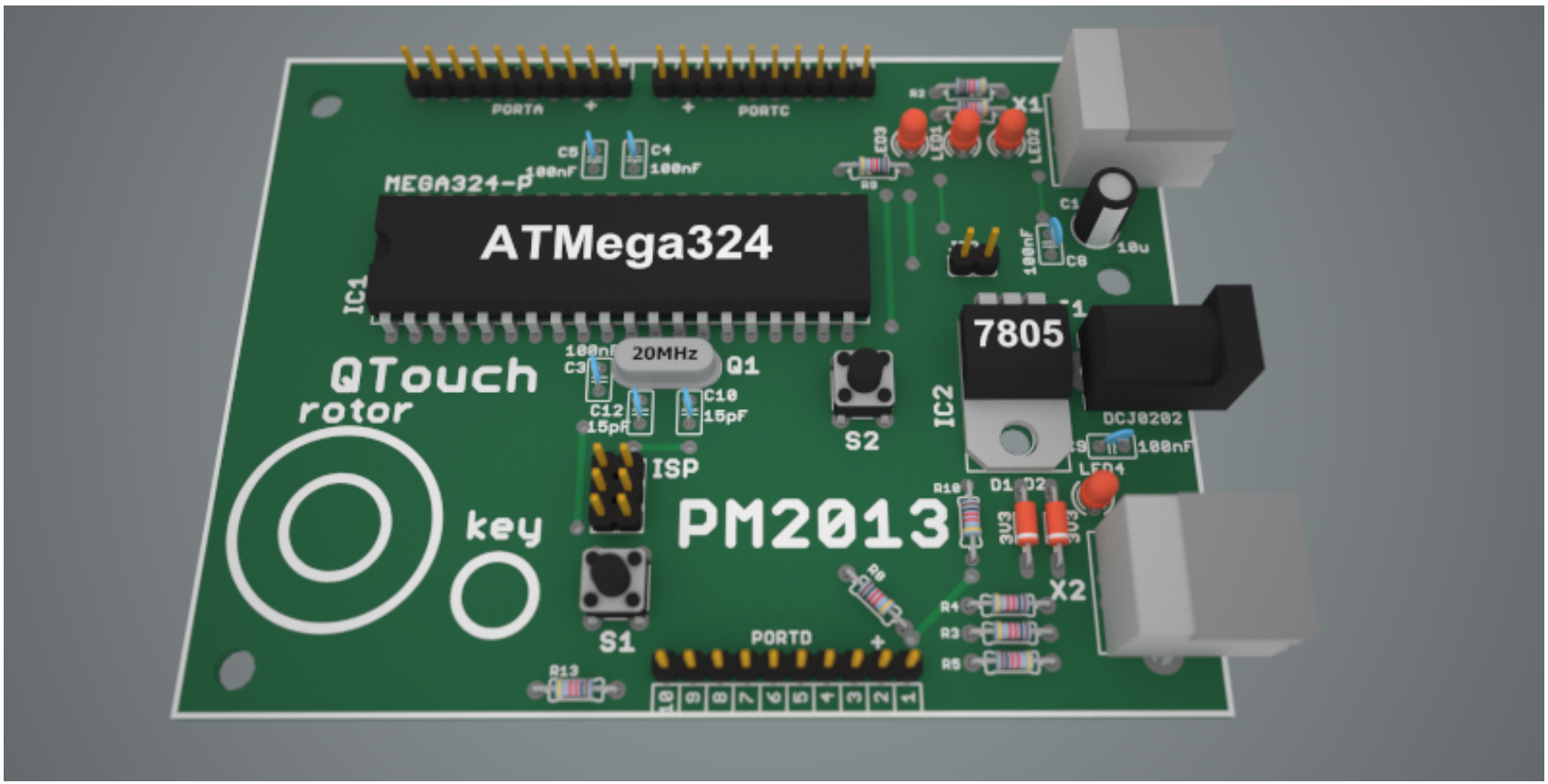
- Săptămâna 8: Demararea proiectului
- Săptămâna 9: Milestone 1
- ...
- Săptămâna 14: Prezentarea proiectului/PM Fair
- Aveți calendarul pe pagina principală a wiki-ului

Săptămâna 9 - Alegerea temei, schema bloc și lista de piese

- The sky is the limit!
- Ideal - vă gândiți de pe acum
  - Aveți proiecte din 2009-2016, știți ce se poate face
  - Instructables, Sparkfun, Avrfreaks, Make, Hackaday
- Săptămâna 8 - vă întâlniți cu asistentul de proiect, discuții
- Asistentul de proiect trebuie să aprobe tema
  - Nu o să se aprobe: ceasuri digitale, termometre digitale etc.
  - Dacă totuși aprobă un proiect considerat simplu, acesta va fi considerat la 80% din notă
  - Rule of thumb: Dacă e mai simplu decât un lab de PM, nu e bine (Fără Arduino IDE!)
- Schema bloc (*Atenție*, nu electrică)
  - 3-4 componente sau câte aveți în proiectul vostru, cu conexiuni simple între ele

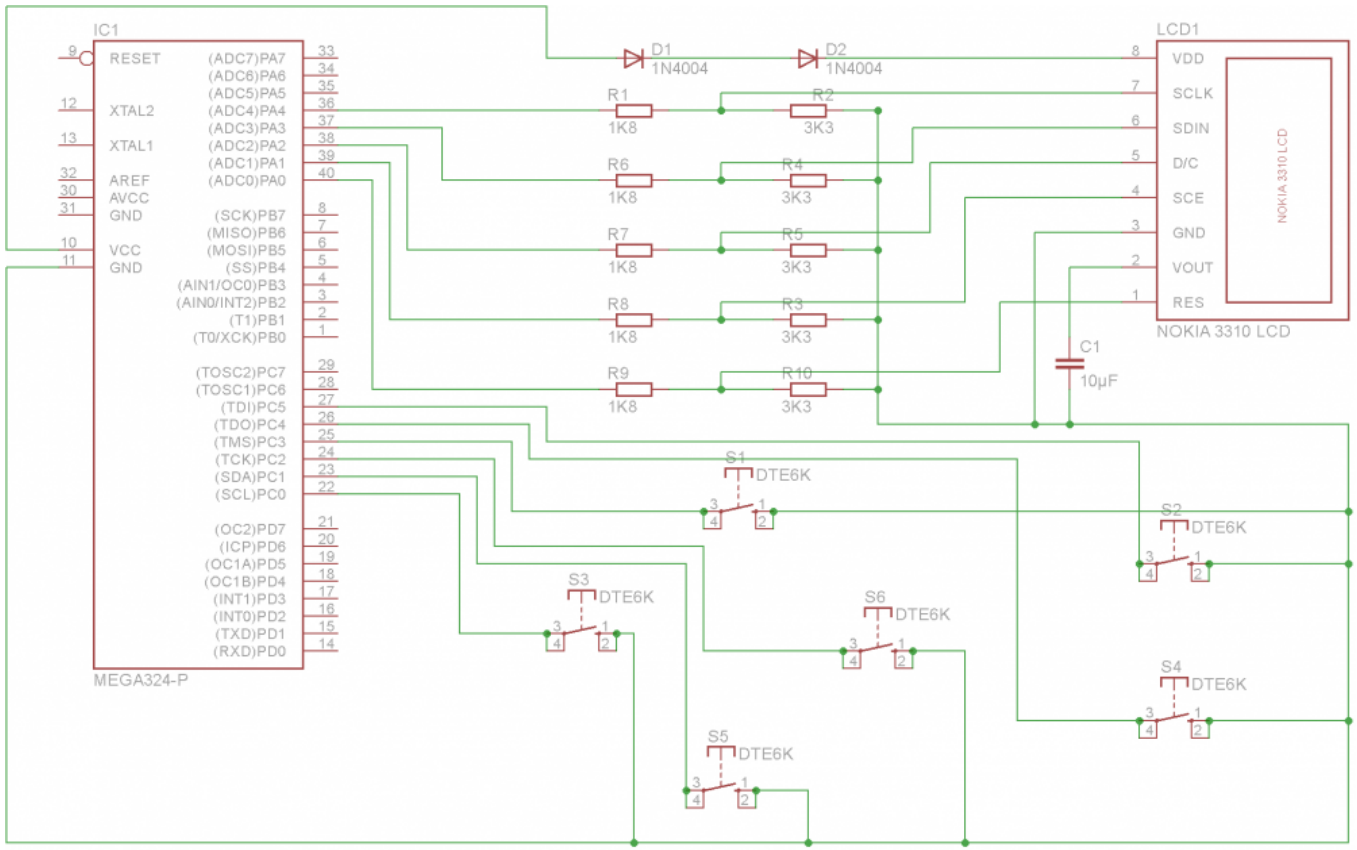
- Nu trebuie să fie lista finală, dar trebuie să conțină ce aveți nevoie pentru partea a doua a proiectului
- Listă de piese
  - Fără piesele din prima parte
  - Așa da: *senzor de temperatură LM35, rezistență* $1k\Omega$*, fire*
  - Așa nu: *rezistențe, condensatori*
  - Implică documentare
  - Găsit supplier
- Maica Domnului, Robofun, Farnell, Okazii, Ebay, Optimus Digital
- Trade-off între timp și bani
  - Placă de motor făcută de voi ∼20RON
  - Cumpărată de pe robofun ∼80RON
- de pe wiki
  - *"Te ajută foarte mult să îți faci lista de piese înainte ca să nu fii nevoit să te dui de un milion de ori să iei piese"*[sic]

- Trebuie prezentată placa de bază funcțională
- Plăcuța rulează programul de test
- Placa din 2013 într-o randare 3D (anul acesta va fi diferită):

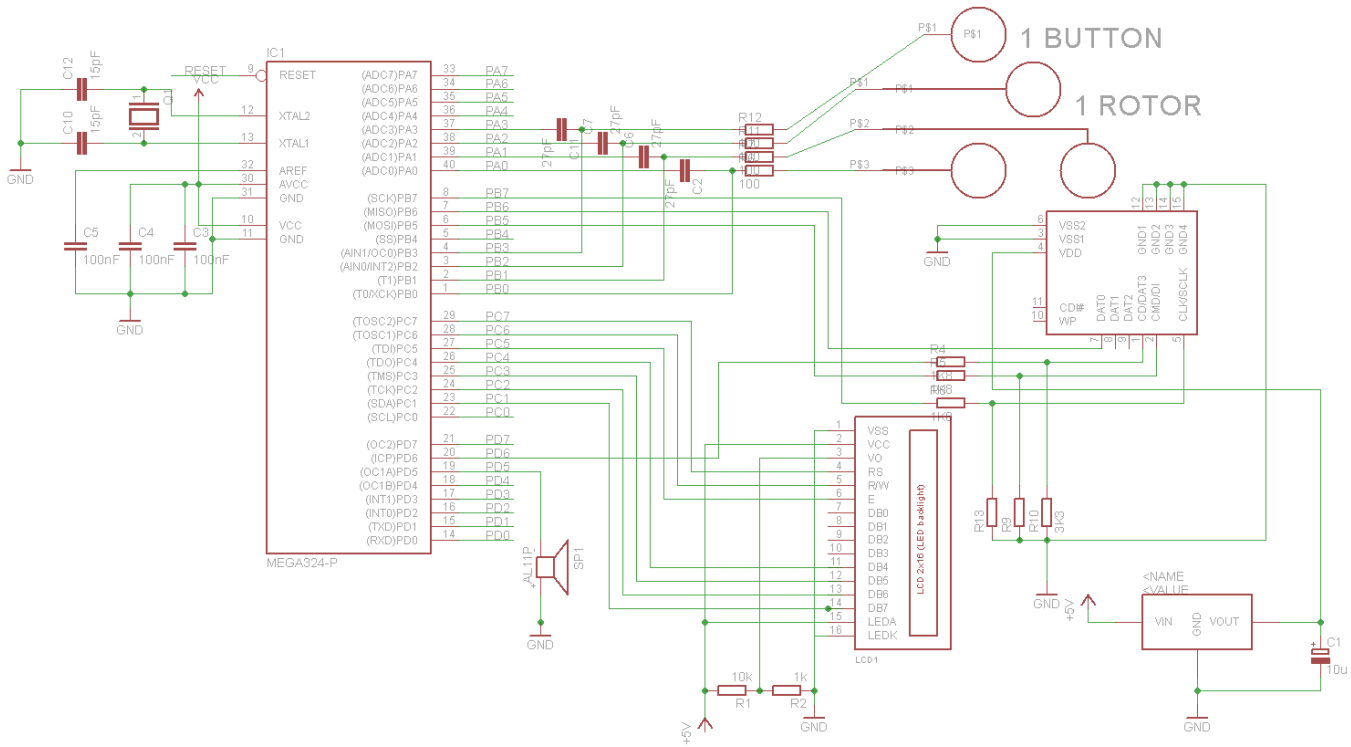


- Cristal 16MHz
- Butoane User, RESET
- LED User, Power
- USB pentru bootloader și serială (V-USB)
- QTouch (slider)
- Header-e expansiune
- Header pentru LCD 16x2
- Header expansiune compatibil pt breakout LCD 3310
- *USB prin FTDI (serială rapidă)* - opțional
- *Alimentare externă la 12V* - opțional
- Cost estimat piese ∼40RON - lista de piese soon^{TM}
- cablajul și lista de piese pentru anul acesta: comanda va veni soon^{TM} ∼7RON

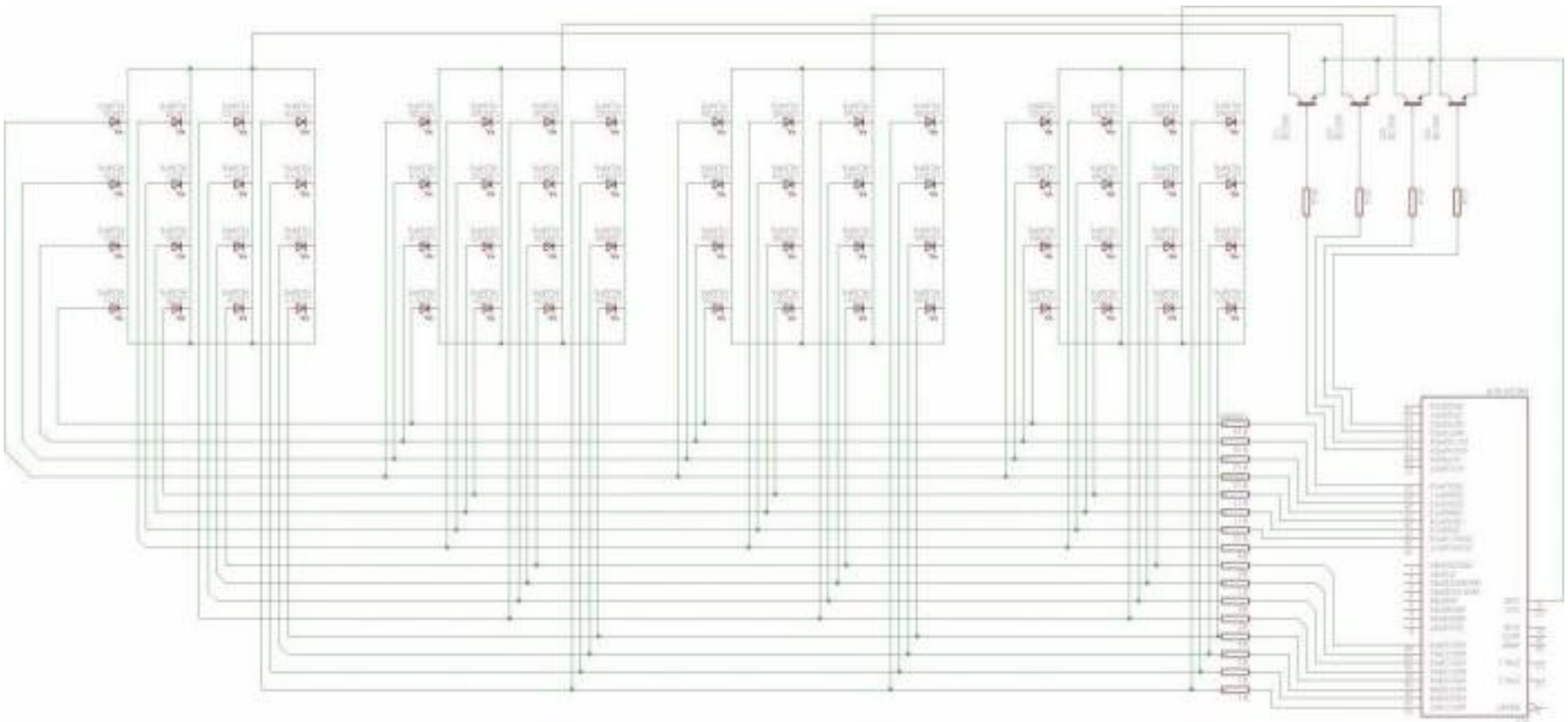
- Nu trebuie să fie finală, dar trebuie să fie completă
- Recomandăm Eagle, nu trebuie făcut decât schematic
- Poate fi mică



- Nu trebuie să fie finală, dar trebuie să fie completă
- Recomandăm Eagle, nu trebuie făcut decât schematic
- Poate fi medie



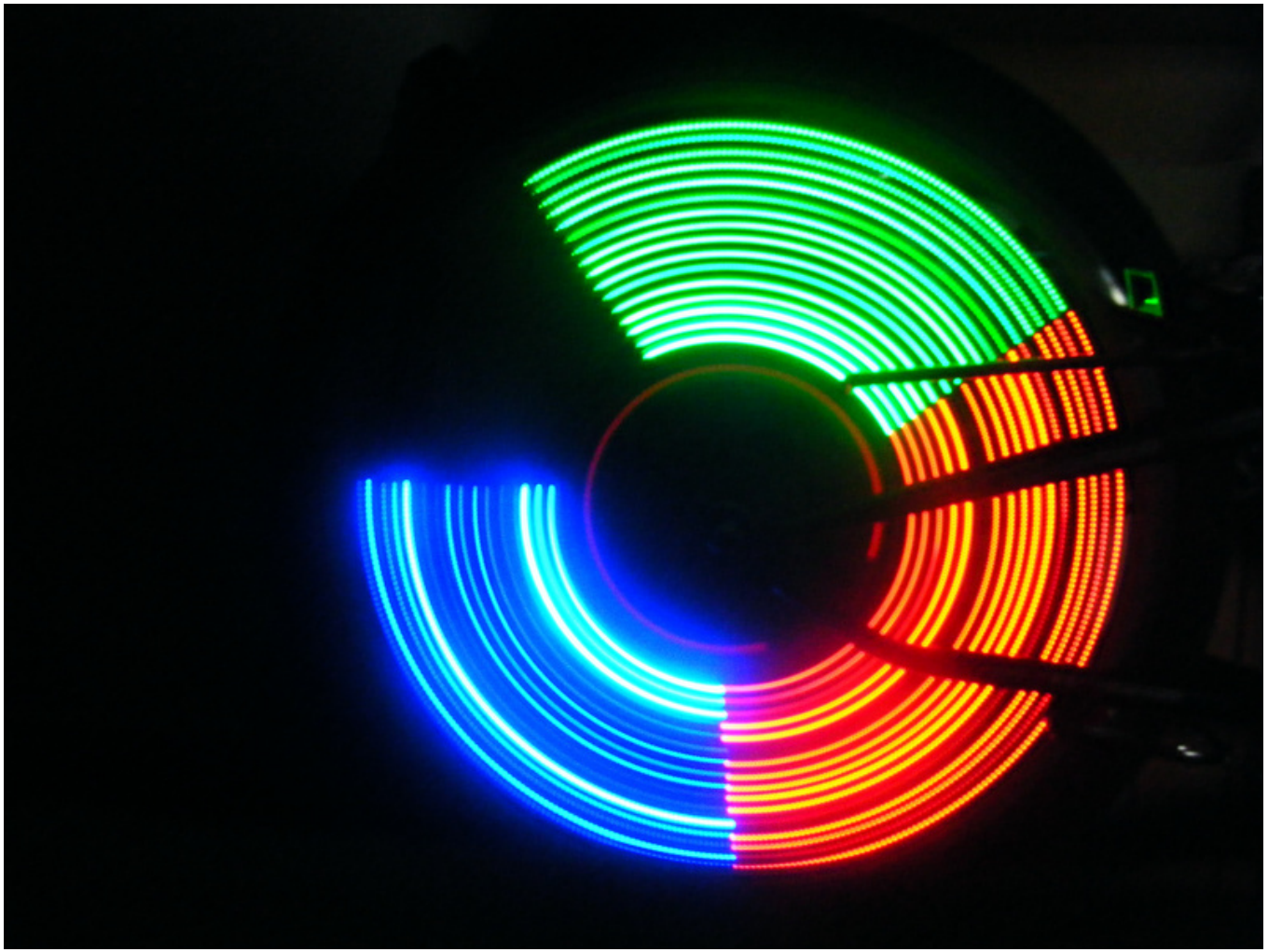
- Nu trebuie să fie finală, dar trebuie să fie completă
- Recomandăm Eagle, nu trebuie făcut decât schematic
- Poate fi mare



- Măcar în stadiu incipient
- Trebuie să fi început să lucrați la hardware/software
  - Nu se pot face milestone-uri mai mici... deci lucrați și voi din timp, chiar înainte de acest deadline

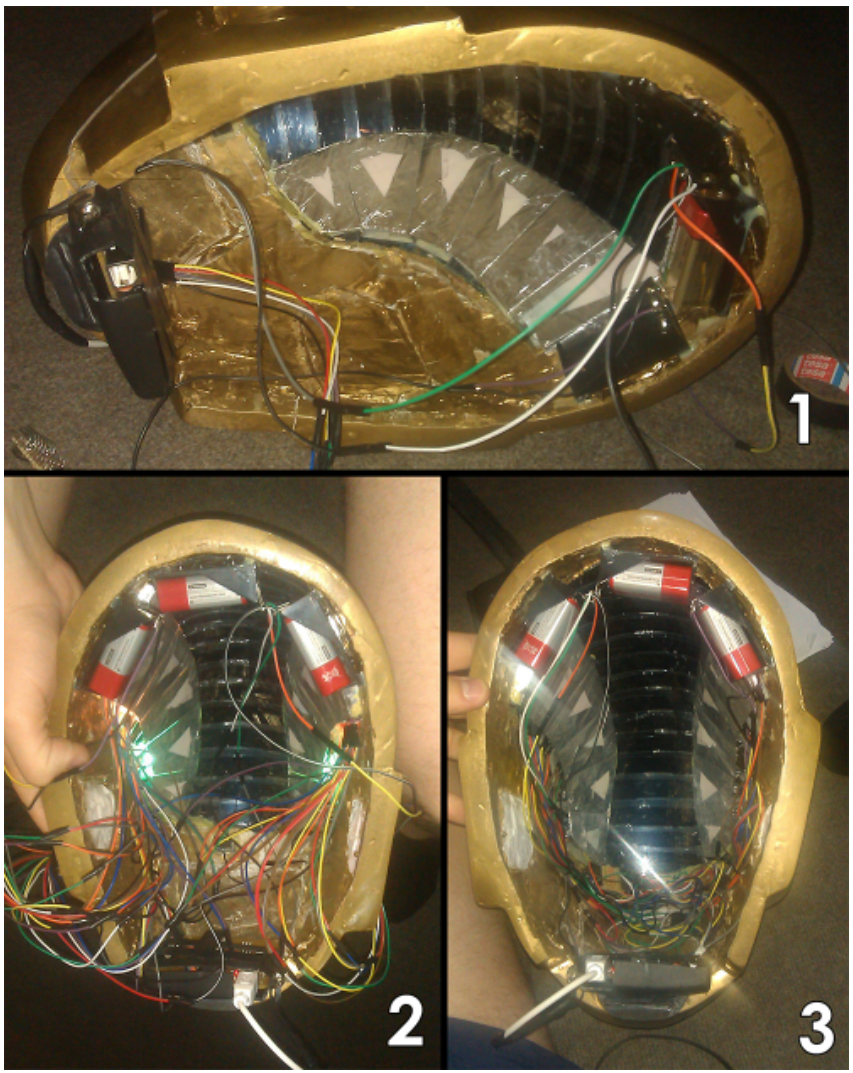
- Mașina de făcut clătite

- POV pentru bicicletă - Persistence of Vision

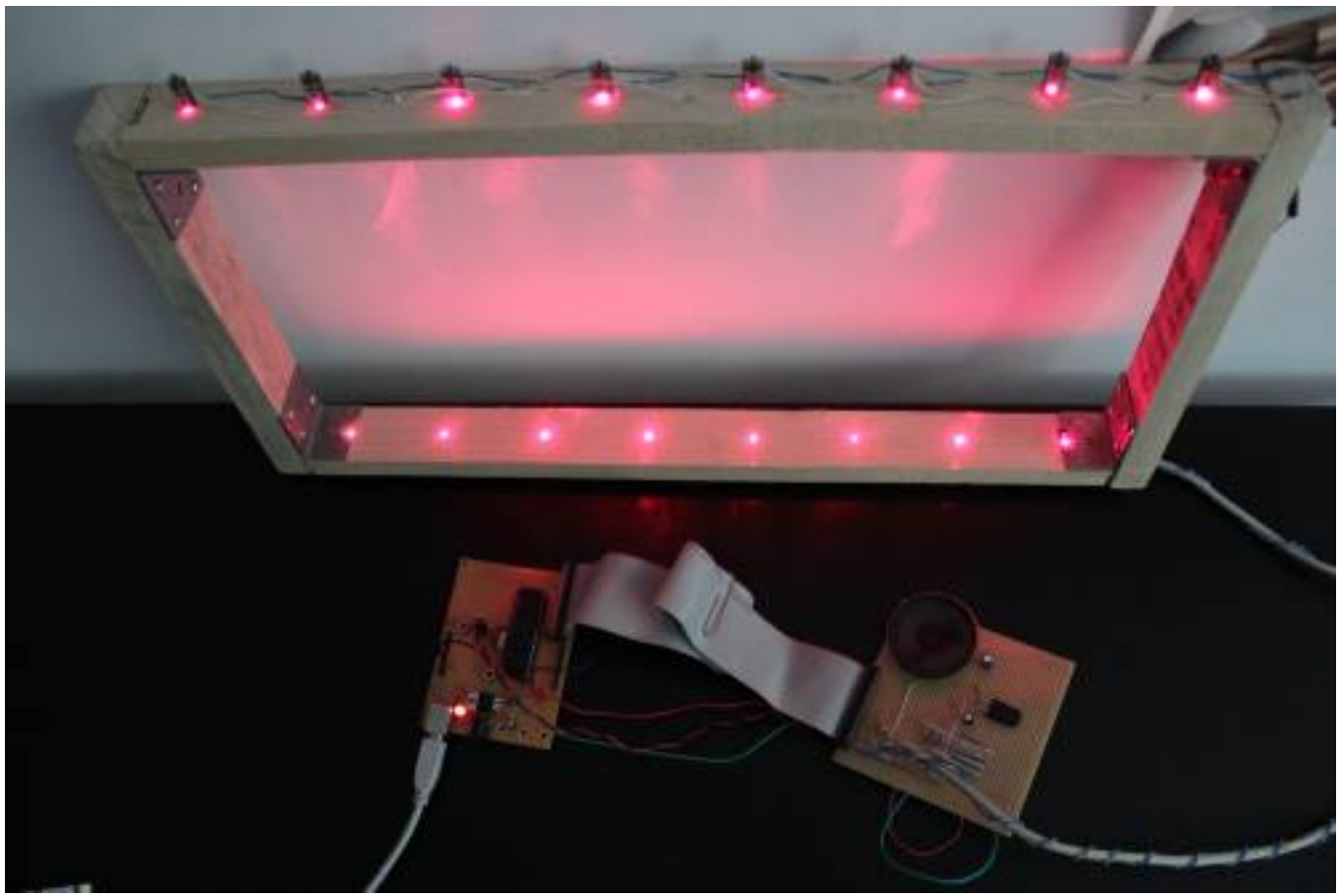


- Air POV - Persistence of Vision

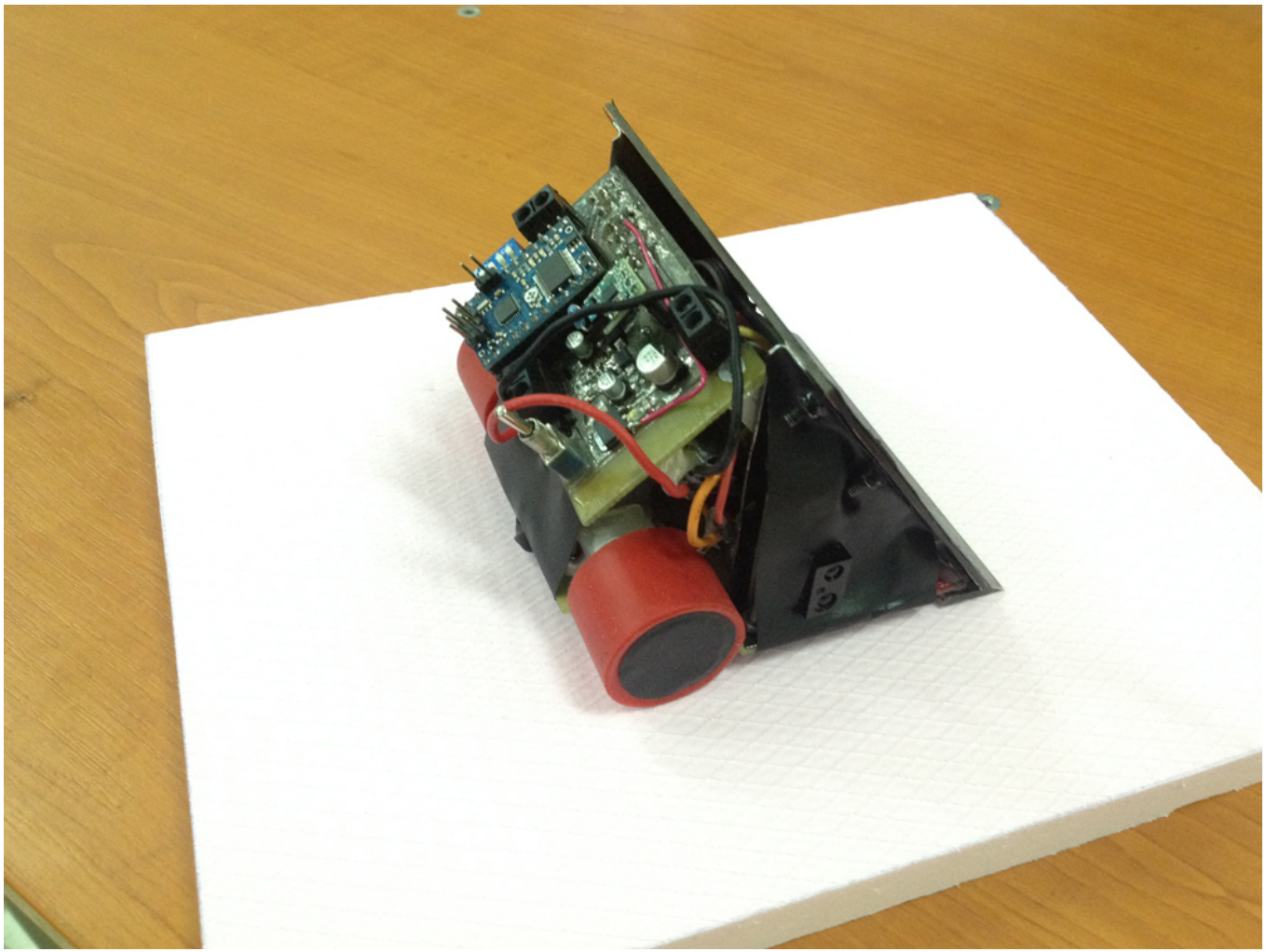
- Cască Daft Punk



- Harpă Laser

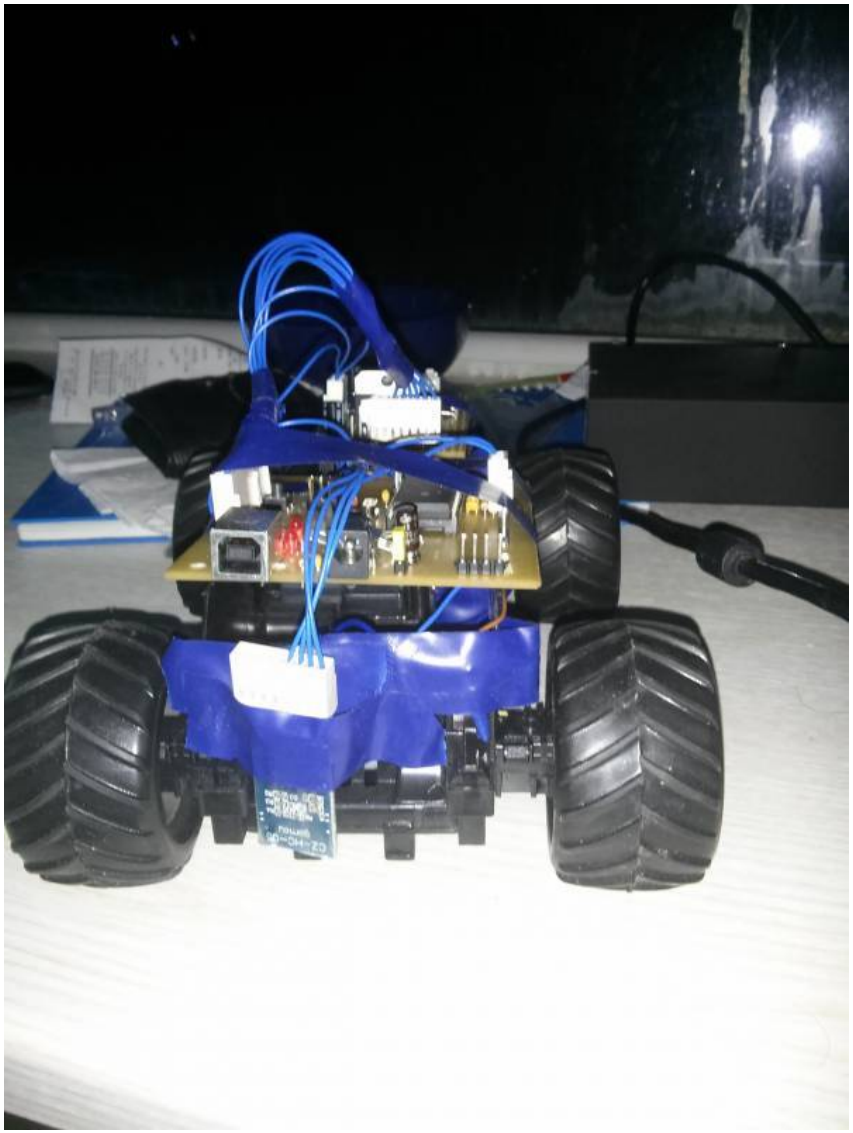


- Robot Mini-sumo

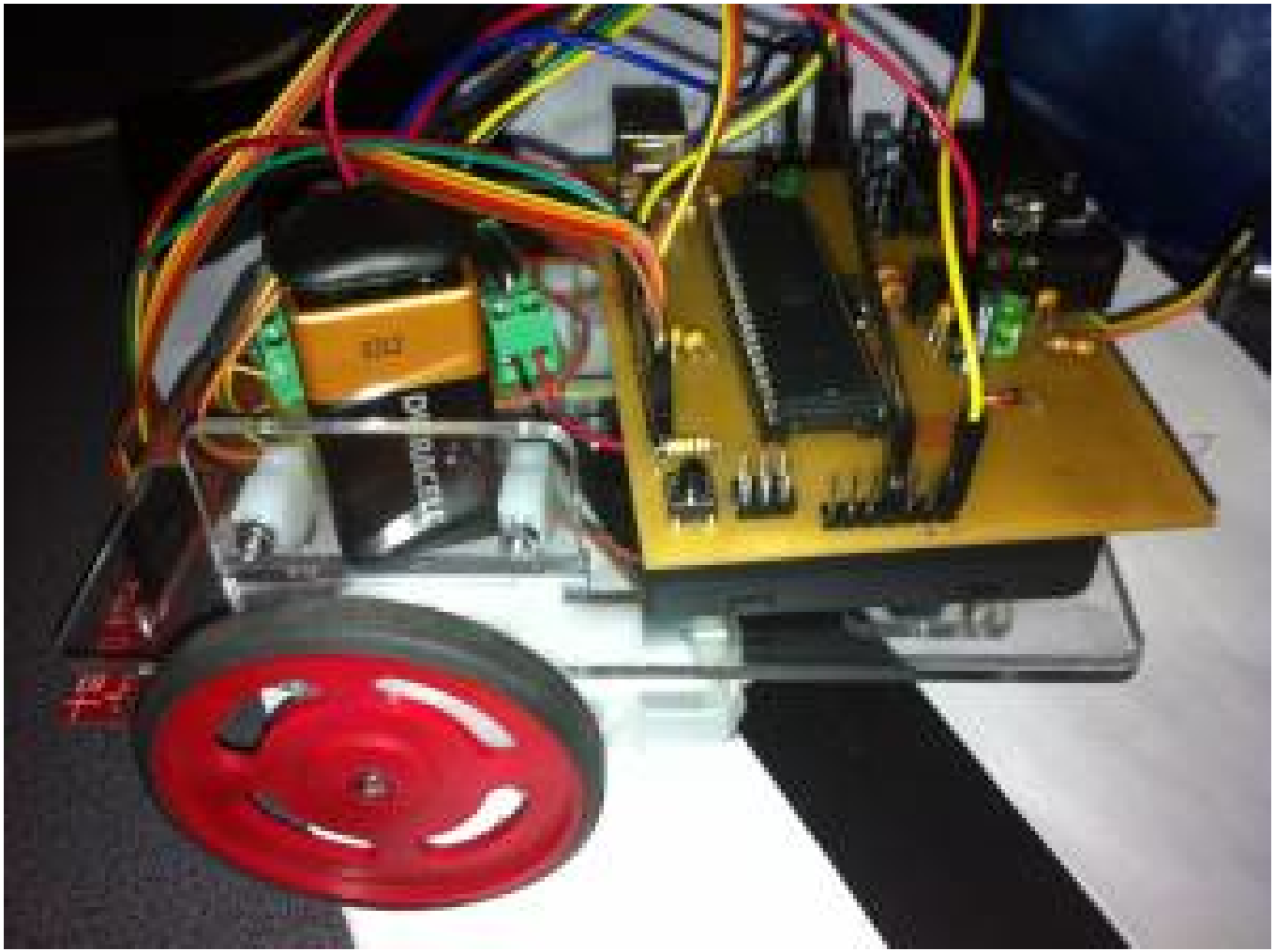


- Joc 2048 pe LCD

- Mașină comandată pe Bluetooth

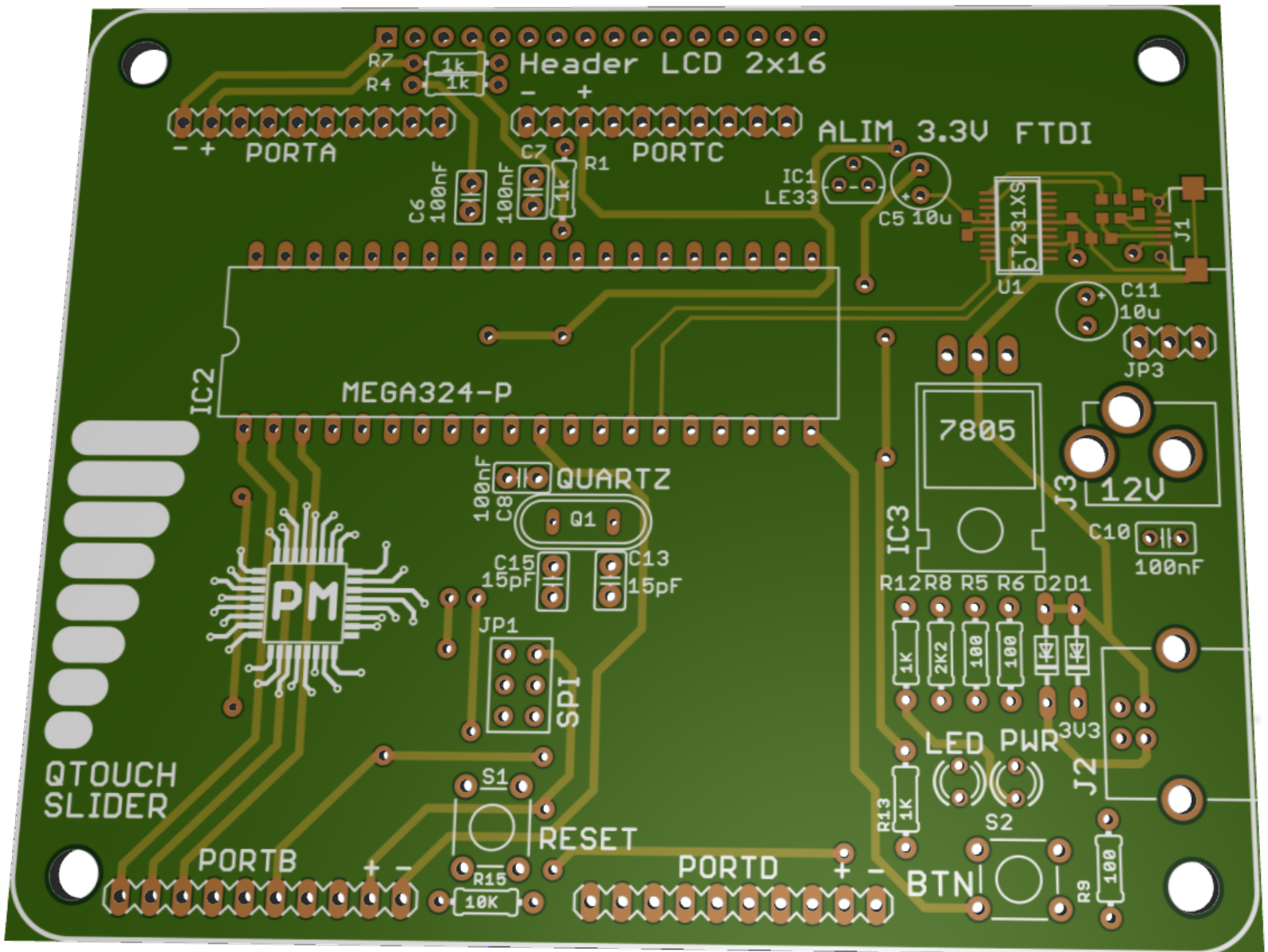


- Line Follower Robot



- Braț robotic

Punctul de plecare - PCB 2017

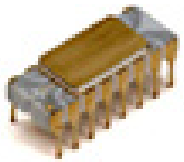


- Un dispozitiv electronic pe o singură plachetă de siliciu care execută instrucțiuni
- De unde încarcă instrucțiuni?
- De unde preia intrările?
- Unde stochează ieșirile?

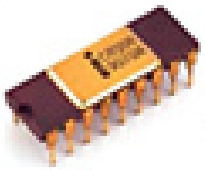
- 1947 - dezvoltarea primului tranzistor în cadrul Bell Laboratories
- 1959 - prima "integrare" a tranzistoarelor pe același substrat
- 1969 - primul chip INTEL - o memorie de 1Kbit RAM
- 1971 - primul microprocesor INTEL - 4004, pe 4 biți

- 1972 - primul microprocesor pe 8 biți - 8008
  - 800 kHz
  - 3500 tranzistoare
  - 6 registre
  - spațiu de adrese de 64KB (PC de 14 biți)
- 1974 - Brian Kernighan și Dennis Ritchie dezvoltă C
- 1978 - Intel lansează 8086
  - Primul microprocesor pe 16 biți
  - Magistrală de date de 16 biți, dar adrese pe 20 de biți - poate accesa 1MB de memorie

Alte procesoare din aceeași perioadă



Intel 4004 (1971)
0.1 MHz, 4-bit
The world's first single-chip microprocessor



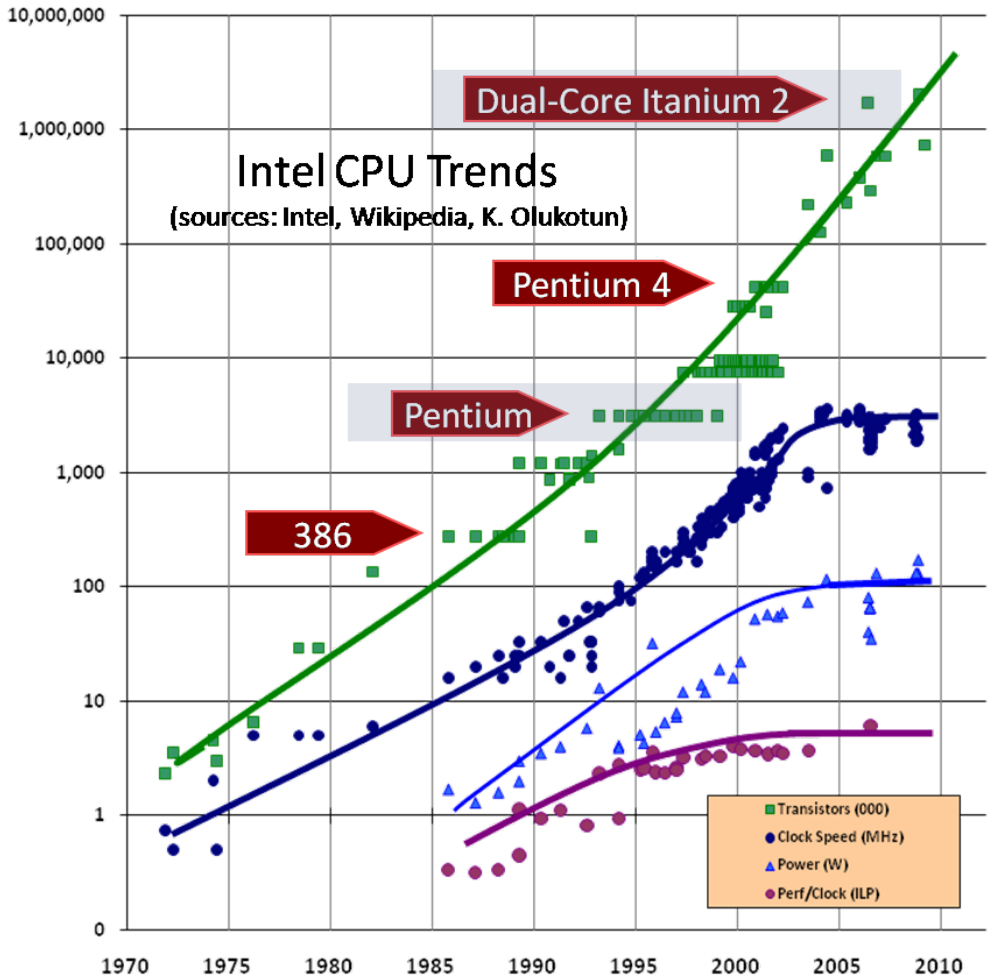
Intel 8008 (1972)
0.2 MHz, 8-bit

Intel 8080 (1974)
2 MHz
Space Invaders machines

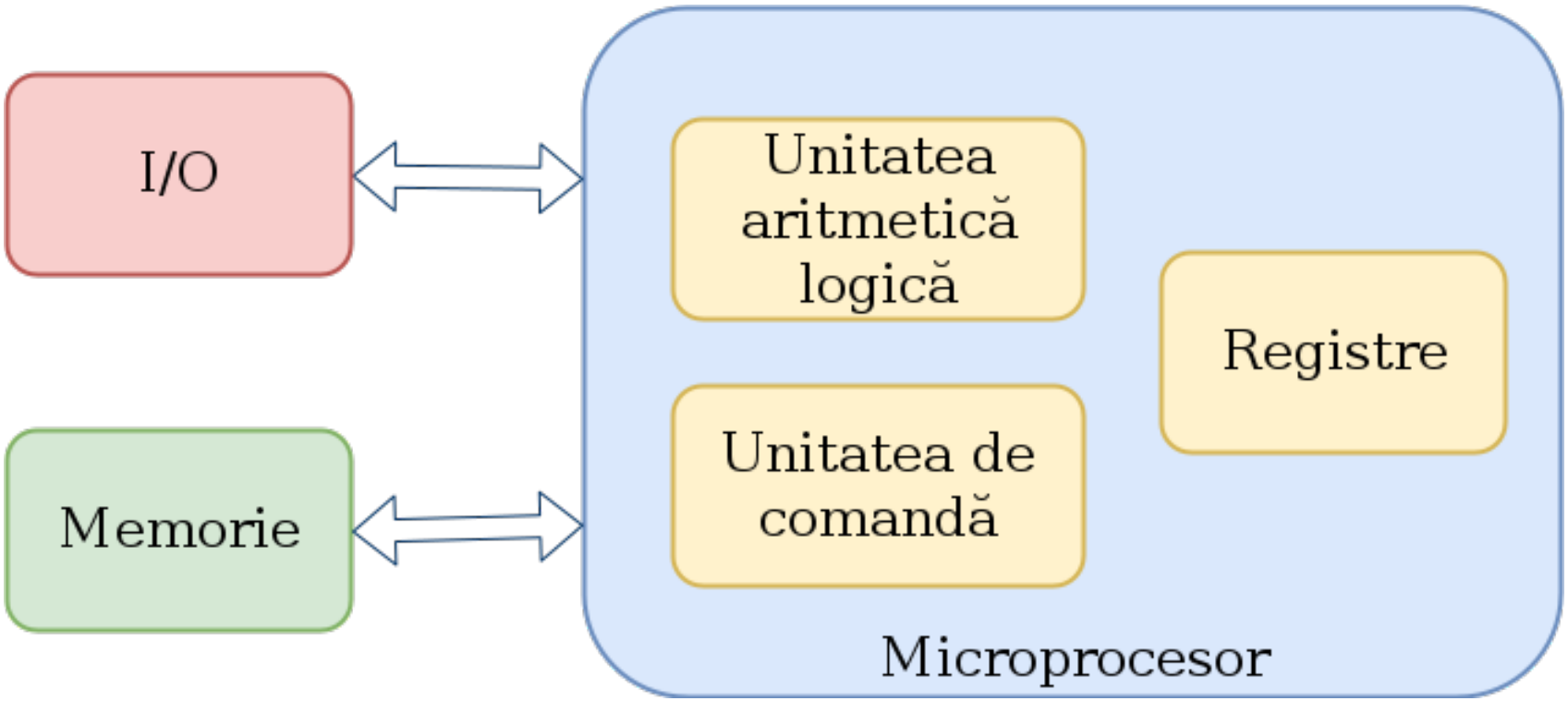
Zilog Z80 (1976)
2 – 4 MHz
ZX81 & ZX Spectrum;
TRS-80; CP/M machines
& Amstrad PCW

MOS 6502 (1975)
1 – 4 MHz
Apple I & II, BBC Micro,
Atari VCS, Nintendo NES,
Commodore 64, and many
early arcade machines

Creșterea procesoarelor de-a lungul timpului

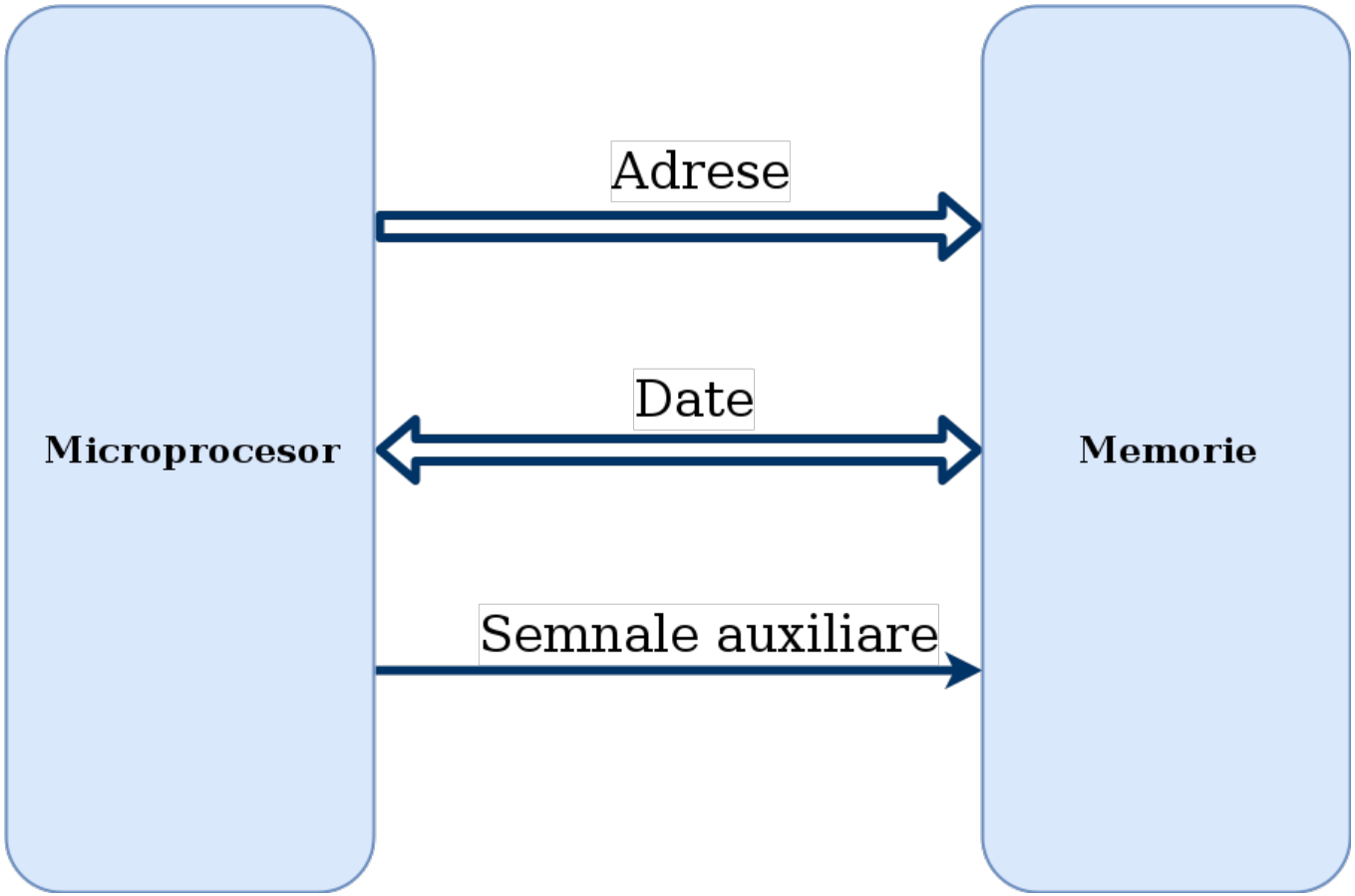


- Din punctul de vedere al accesului la memorie, sunt 2 arhitecturi:
  - *von Neumann*, unde memoria conține atât instrucțiuni cât și date. PC-urile actuale sunt toate von Neumann
  - *Harvard*, unde accesul la memorie se face pe magistrale separate, una pentru date, una pentru instrucțiuni. Microcontrollere AVR de la laborator sunt arhitectură Harvard


I/O
Memorie
Unitatea aritmetică logică
Unitatea de comandă
Registre
Microprocesor

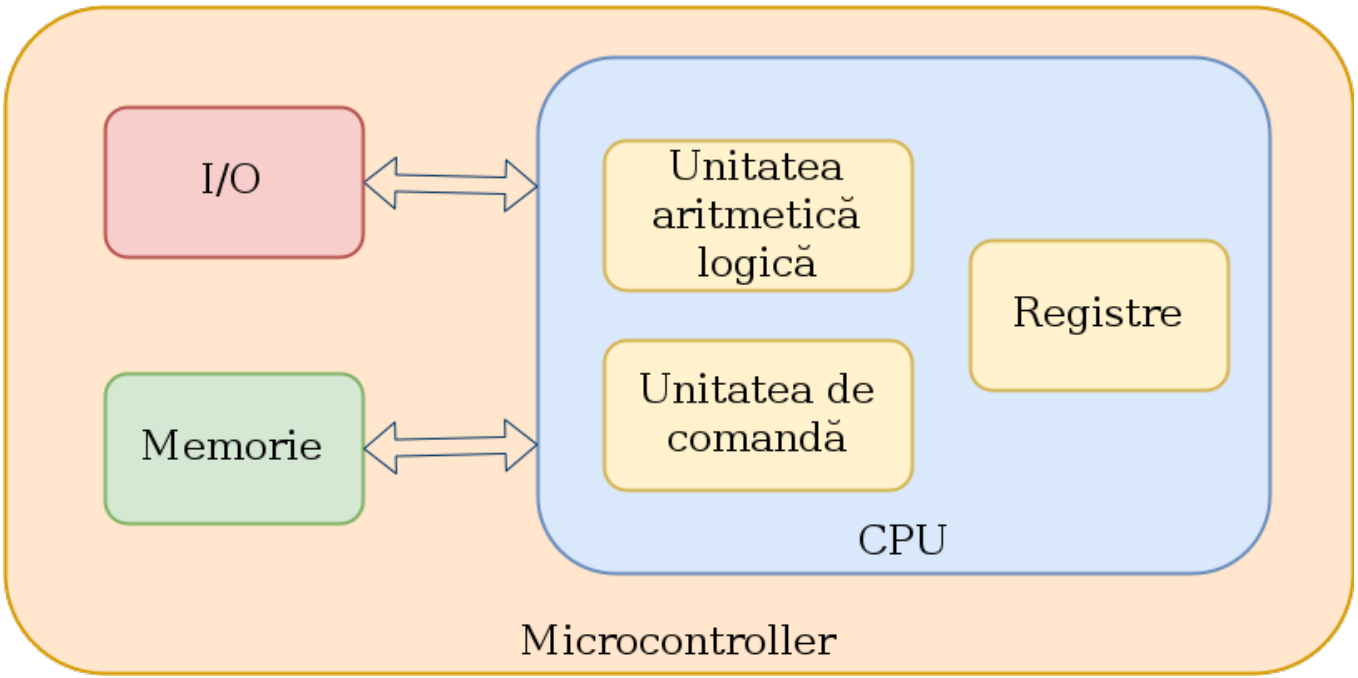
- Un microprocesor are nevoie de:
  - Memorie
  - Periferice I/O
  - Circuit de ceas
  - Circuit de RESET
  - ...
- Există componente care nu sunt strict necesare, dar fac viața mai ușoară
  - DMA
- Toate acestea presupun componente externe chip-ului care trebuie așezate pe o placă

- Magistrala asigură transferul de date bidirecțional între procesor și o locație din memorie
- Accesul procesorului la memorie are loc doar prin magistrală

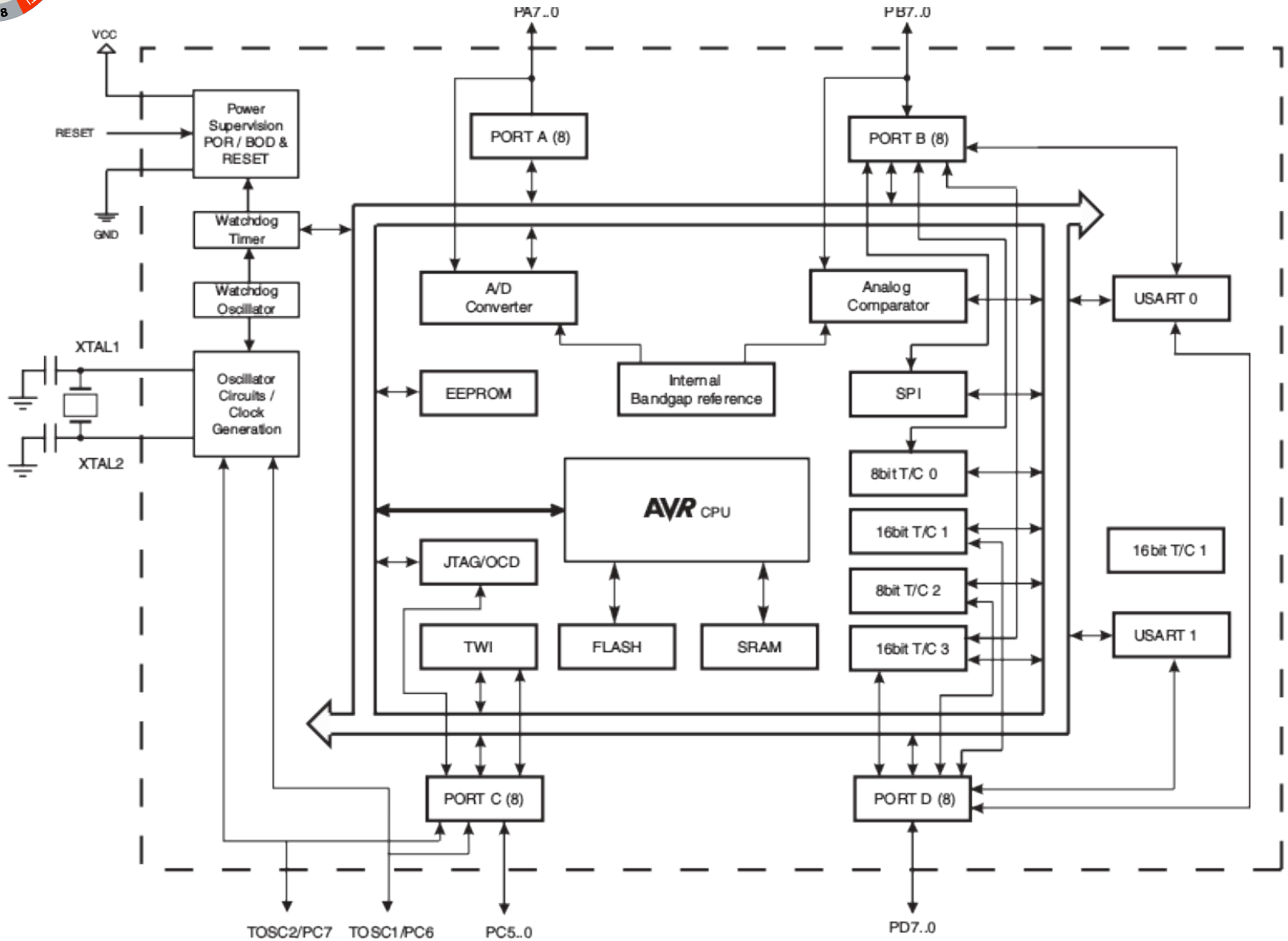


- Sunt două metode de a accesa periferice I/O:
- Memory-mapped I/O:
  - Perifericul I/O are o adresă, registrele lui sunt accesibile la acea adresă
- Port-mapped I/O:
  - Microprocesorul are instrucțiuni speciale de acces al perifericelor I/O, care se traduc la exteriorul chip-ului prin alte semnale auxiliare decât cele pentru citire/scriere din/în memorie

- Microcontroller-ul este un chip integrat care conține un microprocesor, cu memorii și cu periferice I/O
- Unde microprocesorul are nevoie de o placă de bază întreagă, microcontroller-ul este self-contained
- un uP poate fi 100$, un uC poate fi $1



Schemă bloc ATMega324a



- Arhitectură Harvard:
  - Există o memorie de program (32kB)
  - Există o memorie de date (2kB)
- Intrările și ieșirile sunt *port-mapped*
  - Există instrucțiuni speciale (`in` și `out`) de acces la memoria I/O
  - Compilatorul se ocupă singur de diferențierea între scriere în memorie și scriere către I/O

- Încărcarea programului
  - Cu ajutorul unui programator extern sau a unui bootloader, binarul de AVR trebuie să ajungă în memoria de program
- Lansarea în execuție
  - La resetarea microcontroller-ului, se pornește execuția începând cu adresa 0[4]
- Fire de execuție
  - Unul singur, dat de registrul `PC` (Program Counter)
- Interacțiune cu exteriorul
  - Doar prin instrucțiuni `in`/`out` (port-mapped I/O)

[4] diferă în cazul bootloader-ului

- În C, foarte similar cu programarea pe PC
  - Nu avem consolă
  - GCC este configurat pentru un sistem pe 8 biți (`int` și pointerii sunt pe 16 biți)
  - Avem macrodefiniții pentru toate adresele relevante de I/O, compilatorul știe automat să folosească instrucțiunile dedicate pentru scrierile și citirile de la aceste adrese

```
#define PORTC *(volatile uint8_t *) (0x020)
```

```
#include <avr/io.h>
#define F_CPU 16000000UL
#include <util/delay.h>

int main()
{
    DDRC = (1 << PC4);


    while (1)
    {
        PORTC ^= (1 << PC4);
        _delay_ms(250);
    }
}
```

- Header-ul cu registrele I/O
- Frecvența de lucru (determinată de placă)
- Setarea pinului PC4 ca ieșire
- Alternarea stării pinului PC4 o dată la 250 de milisecunde

Cum arată rezultatul?

```
00000000 <__vectors>:
   0:   0c 94 3e 00     jmp 0x7c            ; 0x7c <__ctors_end>
                                            ; care face jump catre main dupa initializar
00000094 <main>:
  94:   80 e1           ldi r24, 0x10       ; (1 << 4)
  96:   87 b9           out 0x07, r24       ; scriere in DDRC
  98:   90 e1           ldi r25, 0x10       ; (1 << 4)
  9a:   88 b1           in  r24, 0x08       ; citire din PORTC
  9c:   89 27           eor r24, r25        ; r24 = r24 XOR r25
  9e:   88 b9           out 0x08, r24       ; scriere in PORTC
  a0:   2f ef           ldi r18, 0xFF       ; 255
  a2:   34 e3           ldi r19, 0x34       ; 52
  a4:   8c e0           ldi r24, 0x0C       ; 12
  a6:   21 50           subi    r18, 0x01   ; 1 ciclu
  a8:   30 40           sbci    r19, 0x00   ; 1 ciclu
  aa:   80 40           sbci    r24, 0x00   ; 1 ciclu
  ac:   e1 f7           brne    .-8         ; 0xa6 <main+0x12>
                                            ; 1 ciclu daca nu face conditia, 2 altfel
                                            ; practic se numara de la 0x0C34FF la 0
                                            ; 0x0C34FF * 5 cicli - 1 de iesire
  ae:   00 c0           rjmp    .+0         ; 0xb0 <main+0x1c>
  b0:   00 00           nop                 ; nop + rjmp = 3 cicli
  b2:   f3 cf           rjmp    .-26        ; 0x9a <main+0x6>
```