

**Sistem cu Procesor 8086**

**Autor: Paulescu Andrei**

**An universitar: 2022-2023**

**Disciplina: Proiectarea microsistemelor digitale**

**Sectia: Calculatoare si Tehnologia Informatiei**

***Tema proiectului:***

Să se proiecteze un microsistem cu următoarea structură:

* unitate centrală cu microprocesorul 8086;
* 128 Ko memorie EPROM, utilizând circuite 27C512;
* 64 Ko memorie SRAM, utilizând circuite 62256;
* interfaţă serială, cu circuitul 8251, plasată în zona 05A0H – 05A2H
* sau 0DA0H – 0DA2H, în funcţie de poziţia microcomutatorului S1;
* interfaţă paralelă, cu circuitul 8255, plasată în zona 0890H – 0896H
* sau 0C90H – 0C96H, în funcţie de poziţia microcomutatorului S2;
* o minitastatură cu 16 contacte;
* 18 LED-uri;
* un modul de afişare cu 7 segmente, cu 10 ranguri (se pot afişa maxim
* 10 caractere hexa simultan).

**DESCRIEREA HARDWARE**

***Unitatea centrală***

Caracteristici ale microprocesorului 8086:

* registrele interne şi magistrala de date externă sunt pe 16 biţi
* posibilitatea de a adresa direct 1 Mo de memorie
* viteză mărită de lucru datorită atât frecvenţei tactului cât şi unei structuri interne bazată pe conceptul de suprapunere care permite aducerea din memorie, în avans, a instrucţiunilor în timpul unor cicluri fără acces la magistrale
* poate acoperi o gamă largă de aplicaţii datorită celor două moduri de lucru ale sale: minim şi maxim
* magistralele de date şi adrese sunt multiplexate iar o parte dintre terminalele de comandă au rol dublu; aceasta a permis încapsularea circuitului într-o capsulă cu doar 40 terminale

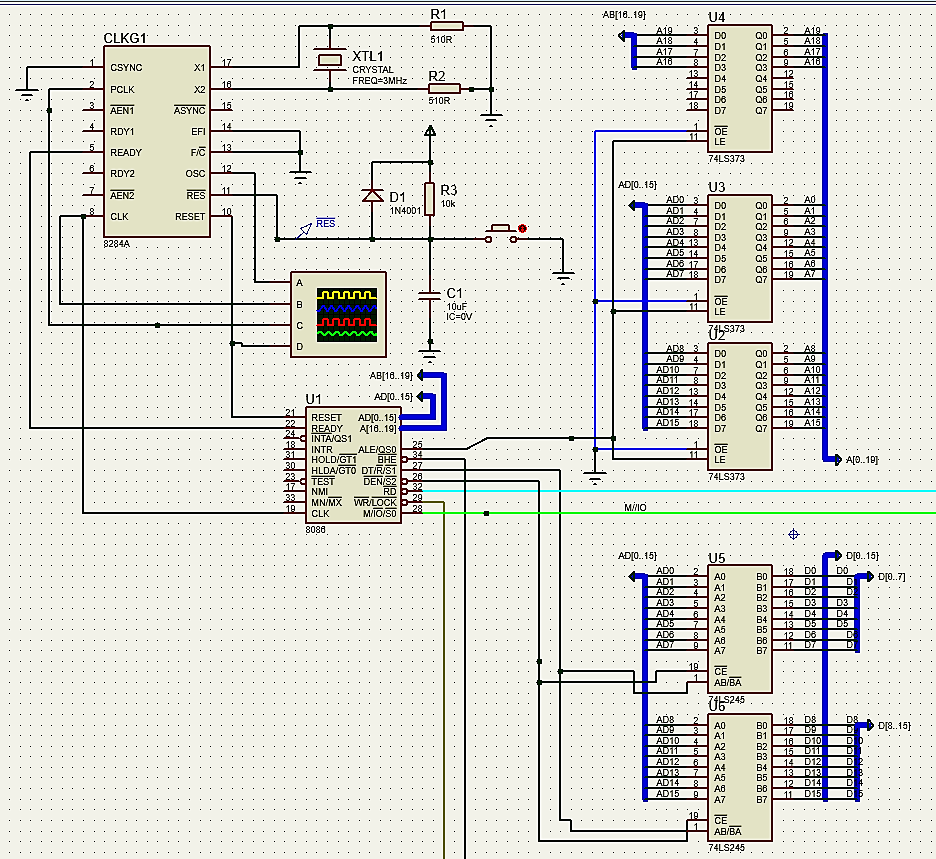
Moduri de lucru:

* minim: pentru aplicaţii relativ simple, în care microprocesorul generează el însuşi semnalele necesare transferurilor cu memoriaşi cu porturile de intrare/ ieşire
* maxim: pentru aplicaţii complexe, inclusiv sisteme multiprocesor, în care semnalele de comandă pentru memorii şi porturi sunt generate de un controler de magistrală, 8288 nu oferă privilegii diferite ci ele se recomandă în anumite configuraţii hardware, pentru tipuri de aplicaţii diferite, trecerea dintr-un mod în altul se face prin hardware: există terminalul MN/ /MX la care, prin 1 logic se cere modul minim iar prin 0 logic se cere modul maxim.

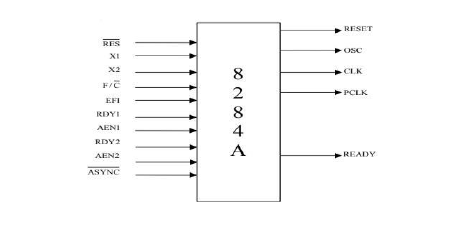
***Terminale:***

******

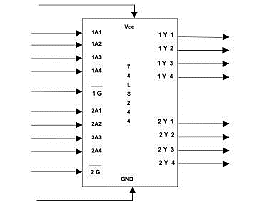
***Schema unității centrale:***

(generator de tact, microprocesor, circuite de separare/demultiplexare)

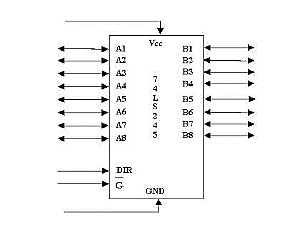
**Generatorul de tact 8284:**

Circuitul 8284A generează tactul către microprocesor şi pentru circuitele specializate pentru interfeţe, generează semnalul READY către microprocesor, sincronizându-l cu tactul şi generează semnalul de iniţializare, RESET, către microprocesor, sincronizându-l cu tactul.

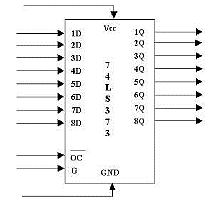
**Circuitul amplificator/separator unidirectional 74x244**

* este un circuit folosit pentru amplificarea/separarea magistralelor unidimensionale

**Circuitul amplificator/separator bidirectional 74x245:**

* este un circuit folosit pentru amplificarea/separarea magistralelor bidimensionale ale microprocesoarelor

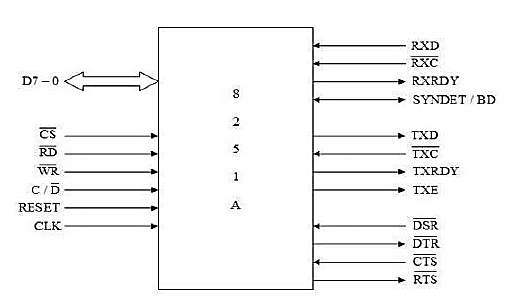
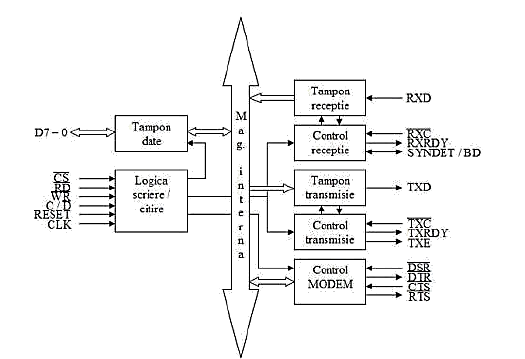
**Circuitul registru 74x373**

* registru cu 8 ranguri, cu 3 stari

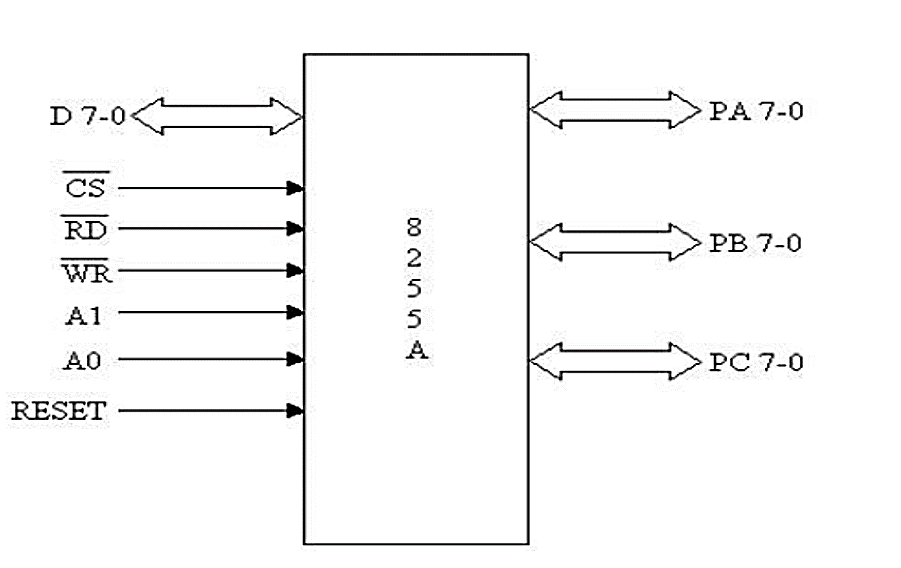
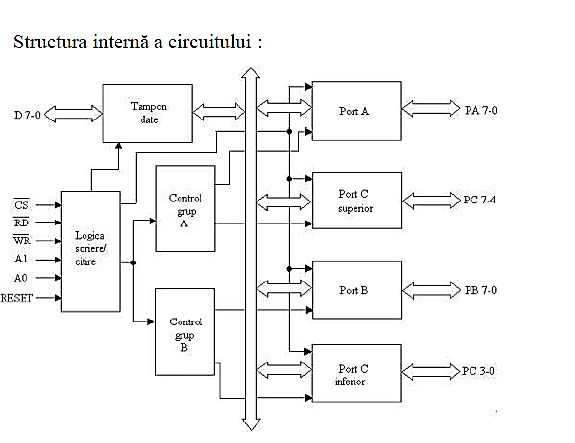
***Interfața serială 8251:***

* Specializat pentru transferurile seriale
* Face parte din categoria circuitelor de tip USART

(“Universal Synchronous Asynchronous Receiver Transmitter”);

* Moduri de lucru: Sincron si Asincron)
* Poate să primească un octet în paralel de la unitatea centrală, să-l serializeze şi să-l transmită la un echipament serial.
* Poate să preia de pe linie, de la un echipament periferic serial, un octet, să-l asambleze şi să-l predea, în paralel, unităţii centrale.
* Circuitul comunică unităţii centrale când are un caracter gata pentru ea sau când a terminat de transmis un octet şi poate prelua altul, poate comunica prin program si prin întreruperi.

**Interfața paralelă 8255:**

* Spre deosebire de transferul serial, la care transferul datelor se face bit după bit, la transferul paralel se transferă 8 biţi simultan iar transferul este însoţit şi de semnale de dialog.
* Există un semnal de dialog către modulul care primeşte datele prin care acesta este anunţat că datele sunt stabile pe linii şi pot fi preluate.
* De asemenea există cel puţin un semnal de dialog de la modulul care primeşte datele prin care acesta comunică primirea acestora sau eventuala indisponibilitate a sa de a primi datele.

***Decodificarea memoriilor :***

* **Harta memoriei**

Memoria este alcatuita din: **🡪** 128 KB EPROM (2\* 64 KB block **27C512**)

**🡪**  64 KB SRAM (2\* 32KB block **62256**)

|  |  |  |
| --- | --- | --- |
| **Block number** | **A19 ………………………………..……………………………. A0** | **Adress hex** |
| 1  (par) | 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 | **00000h** |
| ………… ..……… ……….. ………… ……….. | **..…………** |
| 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 | **1FFFEh** |
| 1  (impar) | 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 | **00001h** |
| ………… ..……… ……….. ………… ……….. | **..…………** |
| 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 | **1FFFFh** |
| 2  (par) | 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 | **20000h** |
| ………… ..……… ……….. ………… ……….. | **..…………** |
| 0 0 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 | **2FFFEh** |
| 2  (impar) | 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 | **20001h** |
| ………… ..……… ……….. ………… ……….. | **..…………** |
| 0 0 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 | **2FFFFh** |

**SELB1 = A19’ A18’ A17’ si SELB2 = A19’ A18’ A17 A16’**

**Daca consideram A19 caz de “don’t care” selectiile devin:**

**SELB1 = A18’A17’ Astfel avem conectati la decodificator**

**SELB2 = A18‘ A17 bitii A18 A17 si ca semnale**

**de selectie pt fiecare bloc :**

**| CSB1  = Y0 | unde, Y0 si Y1 sunt iesiri dintr-un**

**| CSB2 = Y1 | decodificator de 2 la 4 ( 74LS139 )**

**Folosind decodificarea incompleta vom avea pentru fiecare bloc 2 locatii in memorie unde va fii “vizibil”:**

**B1( par ) : [00000h-1FFFEh] , [80000h-9FFFEh]**

**B1(impar): [00001h-1FFFFh] , [80000h-9FFFFh]**

**B2( par ) : [20000h-2FFFEh], [A0000h-AFFFEh]**

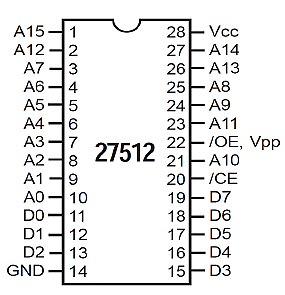
**B2(impar): [20001h-2FFFFh],[A0001h-AFFFFh],**

**\*Pentru a nu suprscrie datele blocurilor din memorie in cazul de fata vom folosii adrese de la 00000h pana la 30000h.\***

**Circuitul EPROM 27C512**

EPROM (*erasable programmable read only memory*) 27C512 este un tip de memorie care îşi păstrează conţinutul odată întrerupta alimentarea. Principalele sale caracteristici sunt:

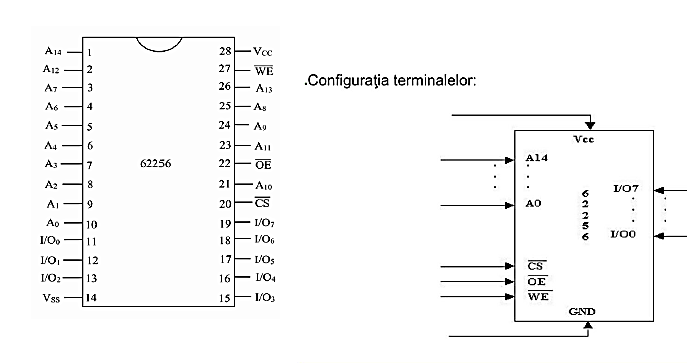
* capacitatea 64KiB,
* timpul de acces de 90-200ns,
* viteza mare
* compatibilitate cu CMOS.

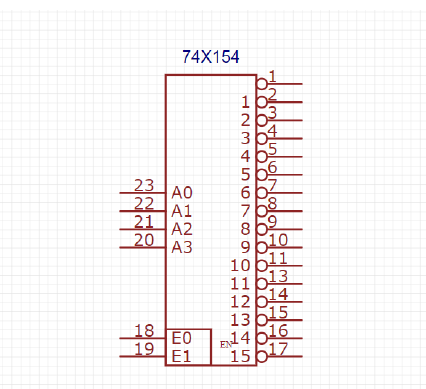
Circuitul 27C512 are 16 linii de adrese, A15 – 0, 8 linii de date (octet) pentru accesarea datelor la adresa respectivă, D7 – D0, biţii de control: o linie de selecţie, /CE, o linie de validare, /OE, necesară în cazul în care pe aceeaşi linie de date mai folosim si alte memorii, rezultând astfel multiplexarea mai multor memorii, o intrare pentru tensiunea de programare, Vpp şi tensiune de alimentare Vcc si GND.

**Memoriile SRAM**

**SRAM** este un tip de memorie care își păstrează starea atâta timp cât este alimentată. Aceasta, spre deosebire de DRAM, nu mai are nevoie de un ciclu periodic pentru reîmprospătarea datelor. Acest lucru este posibil deoarece memoriile SRAM folosesc circuite logice combinaționale pentru a memora fiecare bit. Termenul SRAM indică faptul că datele depuse în memorie sunt

**Circuitul de memorie 62256**

* capacitatea memoriei este de 32 Ko
* timpul de acces = 45 – 84 ns

**Circuitul 74LS154 implementează decodificatorul de porturi**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| A15 | A14 | A13 | A12 | A11 | A10 | A9 | A8 | A7 | A6 | A5 | A4 | A3 | A2 | A1 | A0 | Adresss |
| **0** | **0** | **0** | **0** | 0 | 1 | **0** | 1 | 1 | **0** | **1** | **0** | **0** | 0 | **0** | 0 | **05A0h** |
| **0** | **0** | **0** | **0** | 0 | 1 | **0** | 1 | 1 | **0** | **1** | **0** | **0** | 0 | **1** | 0 | **05A2h** |
| **0** | **0** | **0** | **0** | 1 | 1 | **0** | 1 | 1 | **0** | **1** | **0** | **0** | 0 | **0** | 0 | **0DA0h** |
| **0** | **0** | **0** | **0** | 1 | 1 | **0** | 1 | 1 | **0** | **1** | **0** | **0** | 0 | **1** | 0 | **0DA2h** |
| **0** | **0** | **0** | **0** | 1 | 0 | **0** | 0 | 1 | **0** | **0** | **1** | **0** | **0** | **0** | 0 | **0890h** |
| **0** | **0** | **0** | **0** | 1 | 0 | **0** | 0 | 1 | **0** | **0** | **1** | **0** | **1** | **1** | 0 | **0896h** |
| **0** | **0** | **0** | **0** | 1 | 1 | **0** | 0 | 1 | **0** | **0** | **1** | **0** | **0** | **0** | 0 | **0C90h** |
| **0** | **0** | **0** | **0** | 1 | 1 | **0** | 0 | 1 | **0** | **0** | **1** | **0** | **1** | **1** | 0 | **0C96h** |

**Decodificare Porturi**

* **Interfaţa serială**, cu circuitul 8251, plasată în zona 05A0H – 05A2H sau 0DA0H – 0DA2H, în funcţie de poziţia microcomutatorului S1.

Pt S1 in pozitia 0 --> adresa **= 0000 0101 1010 00[0]0** => port date (05A0H)

adresa = **0000 0101 1010 00[1]0** => port com/stari(05A2H)

Pt S1 in pozitia 1 --> **0000 1101 1010 00[0]0** => port date (0DA0H)

**0000 1101 1010 00[1]0** => port com/stari(0DA2H)

* **interfaţă paralelă**, cu circuitul 8255, plasată în zona 0890H – 0896H sau 0C90H – 0C96H, în funcţie de poziţia microcomutatorului S2.

Pt S2 in pozitia 0 : **0000 1000 1001 0[00]0** => port A (0890H)

**0000 1000 1001 0[01]0** => port B (0892H)

**0000 1000 1001 0[10]0** => port C (0894H)

**0000 1000 1001 0[11]0** => port com(0896H)

Pt S2 in pozitia 1 : **0000 1100 1001 0[00]0** => port A (0C90H)

**0000 1100 1001 0[01]0** => port B (0C92H)

**0000 1100 1001 0[10]0** => port C (0C94H)

**0000 1100 1001 0[11]0** => port com(0C96H)

***Semnale:***

***`E1 = M/`IO;***

***C/`D = A[1]; (Seriala)***

***A1 = A[2]; (Paralela porturi)***

***A0 = A[1]; (Paralela porturi)***

***A11 A10 A4 = selectie interfete seriala/paralela***

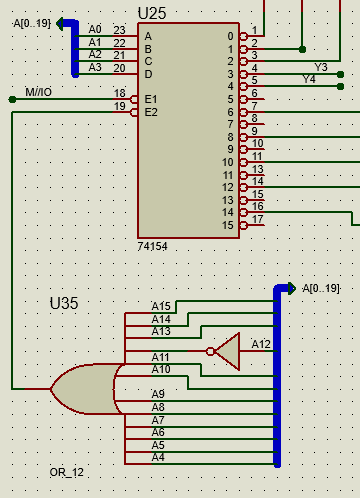
***Decodificare leduri/tastatura/afisaje***

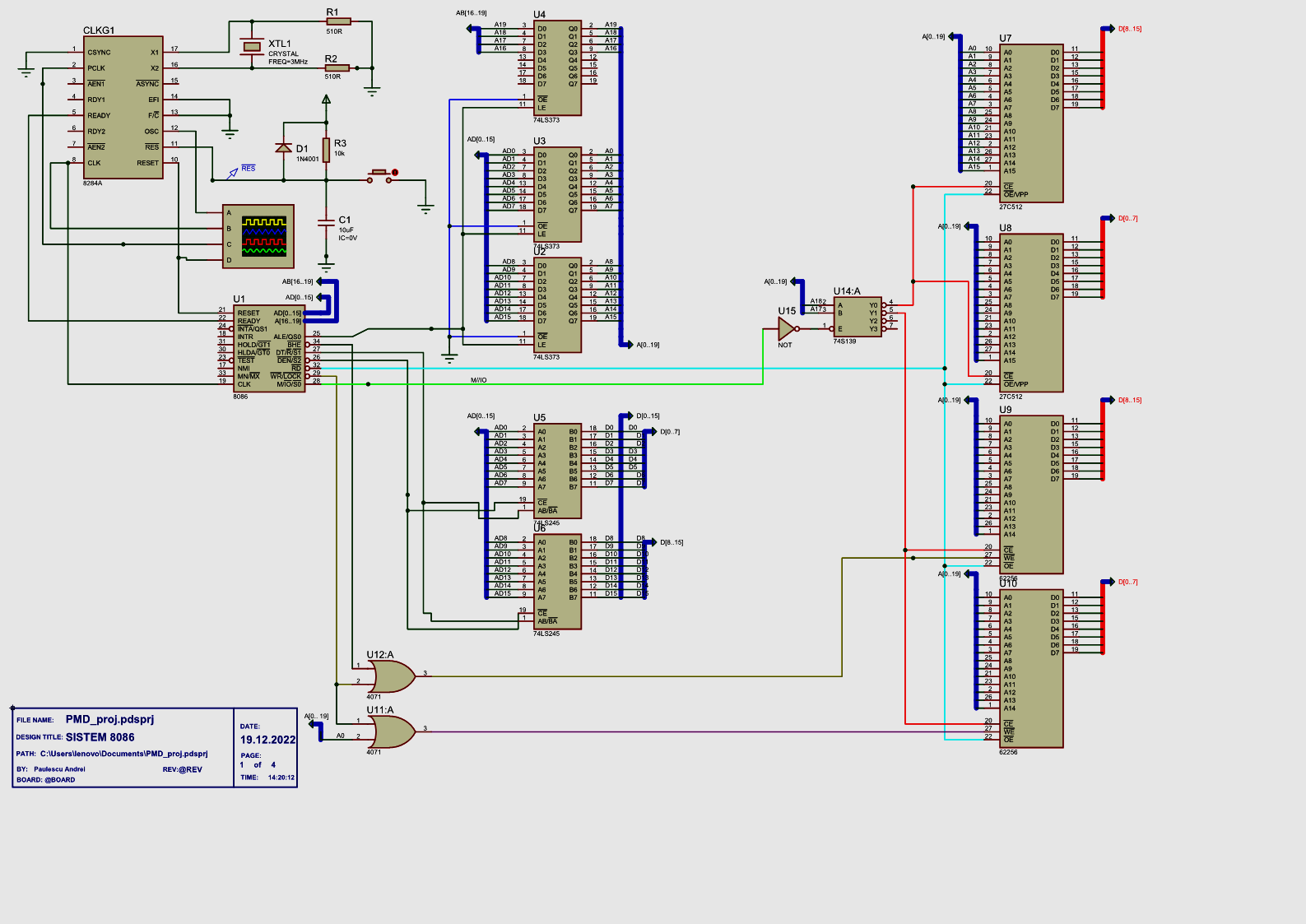
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| A15 | A14 | A13 | A12 | A11 | A10 | A9 | A8 | A7 | A6 | A5 | A4 | A3 | A2 | A1 | A0 | Adresss |
| 0 | 0 | 0 | **1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **0** | **0** | **0** | **0** | **1000h L** |
| 0 | 0 | 0 | **1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **0** | **0** | **0** | **1** | **1001h L** |
| 0 | 0 | 0 | **1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **0** | **0** | **1** | **0** | **1002h L** |
| 0 | 0 | 0 | **1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **0** | **0** | **1** | **1** | **1003h T** |
| 0 | 0 | 0 | **1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **0** | **1** | **0** | **0** | **1004h T** |
| 0 | 0 | 0 | **1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **0** | **1** | **1** | **0** | **1006h A** |
| 0 | 0 | 0 | **1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **1** | **0** | **0** | **0** | **1008h A** |
| 0 | 0 | 0 | **1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **1** | **0** | **1** | **0** | **100Ah A** |
| 0 | 0 | 0 | **1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **1** | **1** | **0** | **0** | **100Ch A** |
| 0 | 0 | 0 | **1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **1** | **1** | **1** | **0** | **100Eh A** |

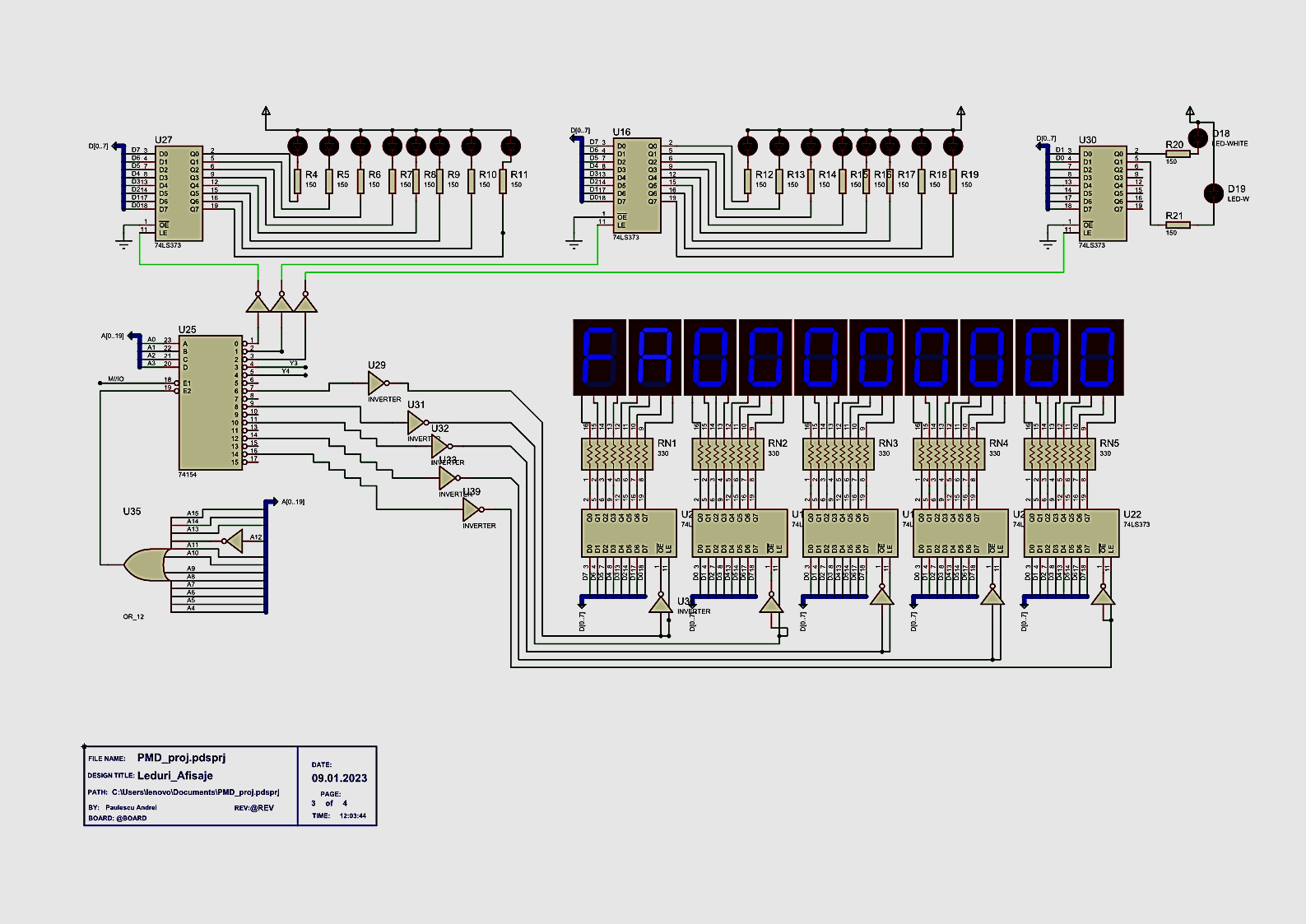
***Avem :***

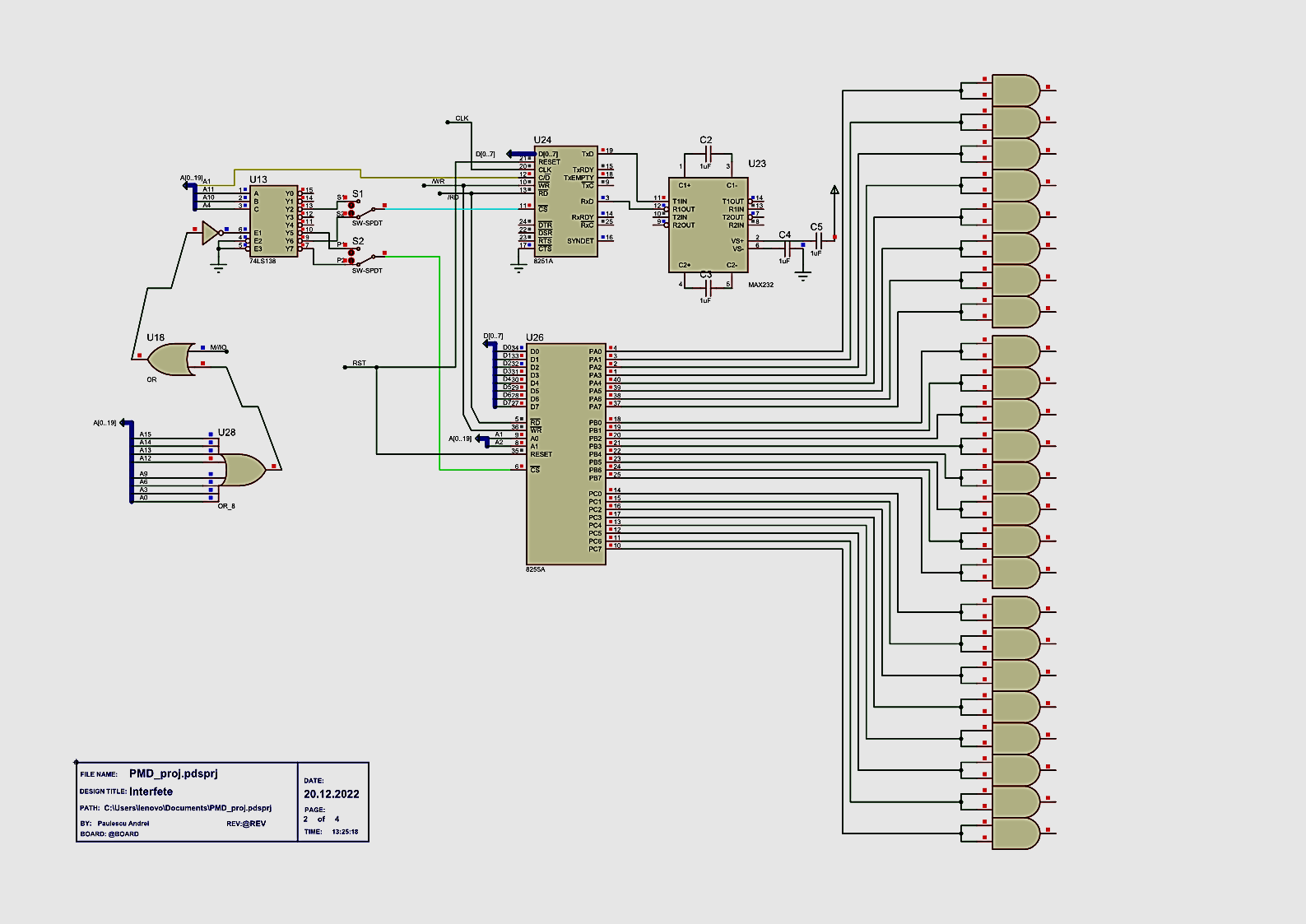
* 3 adrese(1000h,1001h,1002h) pt leduri ( cate 8 grupuri pe adresa)
* 2 adrese (1003h si 1004h) pt tastatura cu 16 contacte
* 5 adrese pt Afisajele cu 7 segmente, 2 ranguri pe adresa

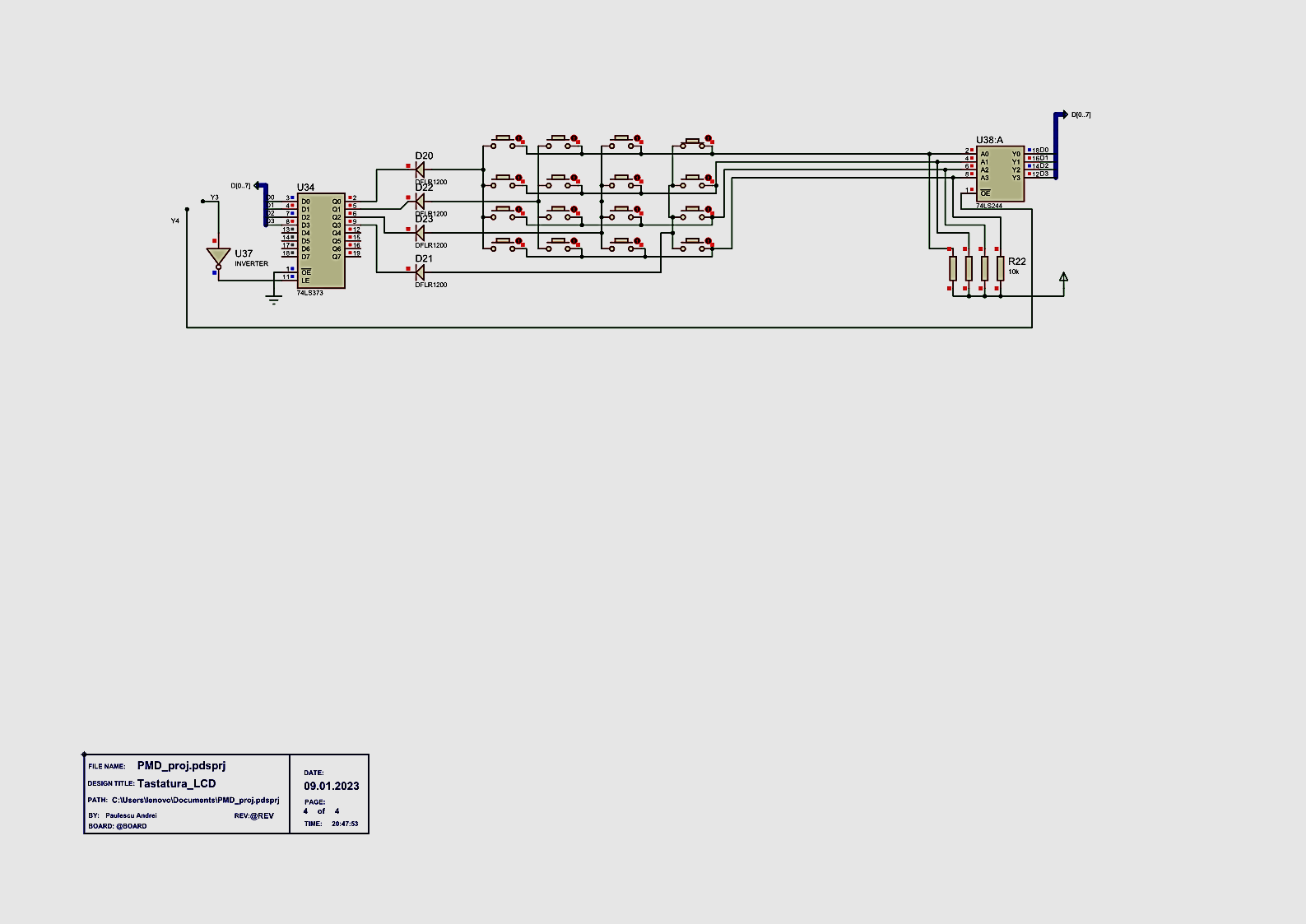
Bitii **A15 , A14 , A13 , /A12 , A11 , A10 , A9 , A8 , A7 , A6 , A5 , A4** ii filtram pintr-o poarta si cu 12 intrari, iesirea portii este enable pt decodificatorul acestora.

Bitii **A3 , A2 , A1 si A0** intra in dec 4 la 16, care selecteaza circuitul dorit.

**Schema Unitate Centrala + Memorii:**

**Schema Leduri\_Afisaje:**

**Schema Interfete:**

**Schema Tastatura:**

## **8251:**

## ;Rutina de programare 8251

## MOV DX,05A0H ;setare adresa periferic sau ;0CB2H

## MOV AL,0CEH ;cuvant de mod

## OUT DX,AL

## MOV AL,15H ;cuvant de comanda

## OUT DX,AL

## RET

## ;Rutina de transmisie de caracter

## TR:

## IN AL,DX ;citire si testare rang TxRDY

## RCR AL,1 ;rotire dreapta cu carry

## JNC TR

## MOV AL,CL

## MOV DX,05A0H

## OUT DX,AL

## RET

## ;Rutina de receptie de caracter

## REC:

## IN AL,DX ;citire si testare rang RxRDY

## RCR AL,2

## JNC REC

## MOV DX,05A0H

## IN AL,DX ;preia date de la 8251

## MOV CL,AL

## RET

## 

## **8255:**

## ;Rutina de programare 8255

## MOV DX, 0896

## MOV AL,21H

## OUT DX,AL

## RET

## ;Rutina de emisie a unui caracter

## PAR\_TR:

## IN AL,DX ;citire si testare BUSY RCR AL,1

## JNC PAR\_TR

## MOV AL,CL ;se preia caracterul din CL

## MOV DX, 0890H

## OUT DX,AL

## OR AL,01H

## MOV DX, 0892H

## OUT DX,AL ;STB=1

## AND AL,00H

## OUT DX,AL ;STB=0

## OR AL,01H

## OUT DX,AL ;STB=1

## RET

## • Rutina de scanare a minitastaturii Tasta apasata se stocheaza in CL

## 0 1 2 3

## 4 5 6 7

## 8 9 A B

## C D E F

## SCAN\_KEYBOARD:

## ; verificare taste 0,4,8,C; se pune 0 pe prima coloana

## MOV AL,0FEH

## OUT 1003H,AL ; verificare tasta 0

## IN AL,1003H ;

## AND AL,01H ; AL&01H

## JZ TASTA0

## ; verificare tasta 4

## IN AL,1003H

## AND AL,02H

## JZ TASTA4

## ; verificare tasta 8

## IN AL,1003H

## AND AL,04H

## JZ TASTA8

## ; verificare tasta C

## IN AL,1003H

## AND AL,08H

## JZ TASTAC

## ; verificare taste 1,5,9,D, se pune 0 pe a doua coloana

## MOV AL,0FDH

## OUT 1003H,AL

## ; verificare tasta1

## IN AL,1003H

## AND AL,01H

## JZ TASTA1

## ; verificare tasta 5

## IN AL,1003H

## AND AL,02H

## JZ TASTA5

## ; verificare tasta 9

## IN AL,1003H

## AND AL,04H

## JZ TASTA9

## ; verificare tasta D

## IN AL,1003H

## AND AL,08H

## JZ TASTAD

## ; verificare taste 2,6,A,E; se pune 0 pe a treia coloana

## MOV AL,0FBH

## OUT 1003H,AL

## ; verificare tasta 2

## IN AL,1003H

## AND AL,01H

## JZ TASTA2

## ; verificare tasta 6

## 

## IN AL,1003H

## AND AL,02H

## JZ TASTA6

## ; verificare tasta A

## IN AL,1003H

## AND AL,04H

## JZ TASTAA

## ; verificare tasta E

## IN AL,1003H

## AND AL,08H

## JZ TASTAE

## ; verificare taste 3,7,B,F;se pune a patra coloana pe 0

## MOV AL,0F7H

## OUT 1003H,AL

## ; verificare tasta 3

## IN AL,1003H

## AND AL,01H

## JZ TASTA3

## ; verificare tasta 7

## IN AL,1003H

## AND AL,02H

## JZ TASTA7

## ; verificare tasta B

## IN AL,1003H

## AND AL,04H

## JZ TASTAB

## ; verificare tasta F

## IN AL,1003H

## AND AL,08H

## JZ TASTAF

## JMP SCAN\_KEYBOARD

## ; verificare evenimentul tasta apasata TASTA0:

## CALL DELAY ;astept stabilizarea

## T0:

## IN AL,DX ;citirea liniei

## AND AL,01H

## JZ T0

## CALL DELAY

## MOV CL,00H ;CL=nr corespunzator tastei RET

## 

## TASTA1:

## CALL DELAY

## T1:

## IN AL,DX

## AND AL,01H

## JZ T1

## CALL DELAY

## MOV CL,01H

## RET

## TASTA2:

## CALL DELAY

## T2:

## IN AL,DX

## AND AL,01H

## JZ T2

## CALL DELAY

## MOV CL,02H RET

## TASTA3:

## CALL DELAY

## T3:

## IN AL,DX

## AND AL,01H

## TASTA4:

## CALL DELAY

## T4:

## IN AL,DX

## AND AL,02H

## JZ T4

## CALL DELAY

## MOV CL,04H

## RET

## Rutina de afisare a unui caracter in hexa pe un rang de segmente

MOV DX,1006H ;1006h-primele 2 ranguri

MOV AL,01H ;pun 01 in afisaje

OUT DX,AL

MOV DX,1008H ;1008h-rang 3 si 4

MOV AL,02H ;pun 02 in afisaje

OUT DX,AL

MOV DX,100AH ;100Ah-rang 5 si 6

MOV AL,0AH ;pun 0A in afisaje

OUT DX,AL

MOV DX,100CH ;100Ch-rang 7 si 8

MOV AL,0FH ;pun 0F in afisaje

OUT DX,AL

MOV DX,100EH ;100Eh-rang 9 si 10

MOV AL,FFH ;pun FF in afisaje

OUT DX,AL

## **Rutina de aprindere/stingere LED**

## LEDS:

## INT 26H ; AL = 0 -> aprindere led-uri; AL = 1-> stingere led-uri

## MOV DX,1000H ; DX = adresa primului grup de LED-uri

## IN AL,DX ; preiau starea in AL

## CMP AL,255 ;compar cu FF pentru a sti daca sunt aprinse sau stinse

## JE APRINDE

## MOV DX,1001H ;DX=adresa celui de-al doilea grup IN AL,DX ;preiau starea in AL

## CMP AL,255

## JE APRINDE

## MOV DX,1002H ;DX=adresa celui de-al doilea grup IN AL,DX ;preiau starea in AL

## CMP AL,255

## JE APRINDE

RET

APRINDE:

MOV AL,000H ;aprind LED-urile

OUT DX,AL

RET