UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA

PROIECT

SISTEM CU COD CIFRU PENTRU SECURIZAREA DULAPURILOR

CUPRINS

1. SPECIFICAȚII

Descriere: Sa se implementeze un sistem numeric care permite utilizatorului adăugarea unui cifru din 3 caractere distincte pentru securizarea unui dulap (asemănător dulapurilor folosite la vestiarele de la sălile de sport, mall, etc)

Cerințe funcționale:

1. Un led LIBER_OCUPAT va avea funcția de a semnala faptul ca dulapul este liber(led stins) sau ocupat(led aprins).

2. Utilizatorul va apasă un buton ADAUGA_CIFRA pentru a semnala începerea introducerii codului. Un led INTRODU_CARACTERE se va aprinde pentru a marca starea.

3. Utilizatorul va adaugă pe rând 3 caractere cu ajutorul butoanelor UP si DOWN.

4. Caracterele permise sunt cuprinse in intervalul 0-1-..-8-9-A-B-..-F

5. Caracterul curent introdus este afișat pe afișor cu 7 segmente (SSD).

6. Pentru trecerea la următorul caracter utilizatorul va apasă butonul ADAUGA_CIFRA.

7. Caracterul anterior introdus rămâne afișat.

8. Următorul caracter este vizibil pe afișaj pe poziția următoare.

9. După introducerea celui de al treilea caracter, la apăsarea butonului ADAUGA_CIFRA, afișajul SSD se va stinge iar cifru va fi in starea blocat prin aprinderea ledului LIBER_OCUPAT.

10. Ledul INTRODU_CARACTERE se va stinge

11. Existenta unui buton/switch RESET in timpul introducerii cifrului pentru revenire in starea inițiala(ledul LIBER_OCUPAT se va stinge, afișajul SSD este gol, ledul INTRODU_CARACTERE se va stinge)

12. Utilizatorul va apasă butonul/switch ADAUGA_CIFRA pentru a începe introducerea codului pentru deblocarea cifrului

13. Se vor relua pașii 2-8.

14. La introducerea ultimului caracter, la apăsarea butonului ADAUGA_CIFRA se va face verificarea, daca codul introdus corespunde cu codul anterior.

15. In cazul de egalitate, ledul LIBER_OCUPAT se va stinge, ledul INTRODU_CARACTERE se va stinge, afișajul SSD se golește.

16. In cazul de inegalitate, ledul LIBER_OCUPAT va rămâne aprins, ledul INTRODU_CARCATERE se va stinge, afișajul SSD se golește.

Cerințe non-funcționale:

- Implementare pe plăcută
- Utilizare SSD
- Utilizare switch-uri, led, butoane

Exemplu use case:

Utilizatorul alege un dulap cu ledul Liber_ocupat stins. Apasă pe butonul ADAUGA_CIFRA pentru a introduce caracterele. Caracterul “0” este vizibil pe SSD. Introduce primul caracter “2” prin apăsarea de 2 ori a butonului UP. Pe SSD se modifica afișajul o data cu apăsarea butonului si anume: 0->1->2. Utilizatorul apasă din nou pe ADAUGA_CIFRA pentru a introduce al doilea caracter “1”. Utilizatorul apasă din nou pe ADAUGA_CIFRA pentru a introduce al doilea caracter “3”. Utilizatorul apasă din nou ADAUGA_CIFRA, codul este salvat, conținutul SSD este gol, ledul LIBER_OCUPAT este aprins, ledul INTRODU_CARACTERE se va stinge.

2.PROIECTARE

2.1. Schema bloc

Schema bloc prezinta 5 intrări si 4 ieșiri.

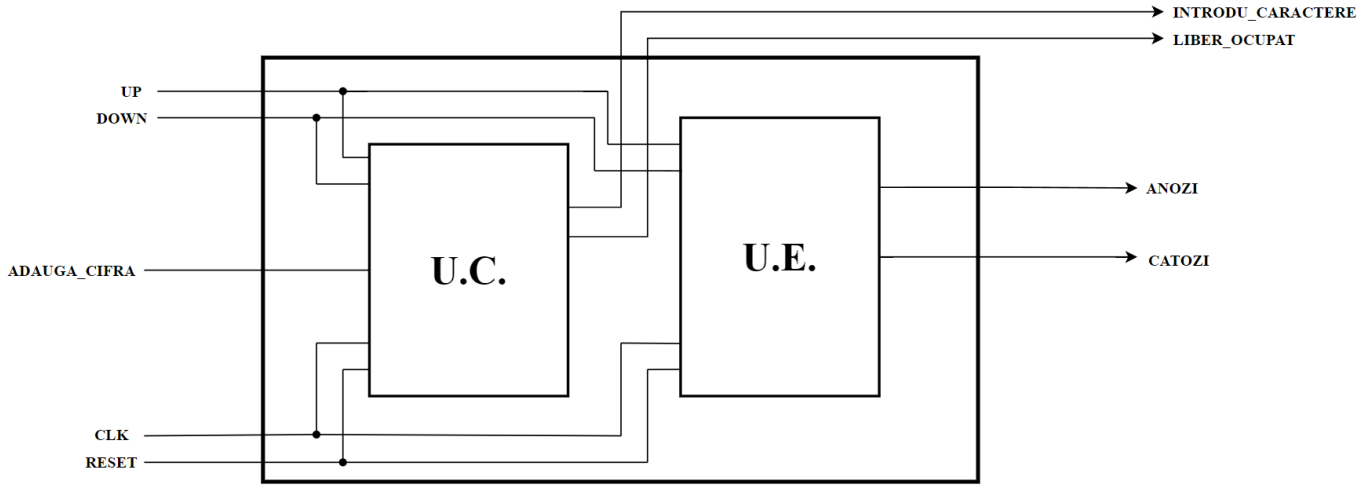
Principalele intrări sunt cele care influențează direct stările:

- Butoanele UP si DOWN sunt pentru deplasarea si afișarea cifrelor înainte de selecție si memorare.
- ADAUGA_CIFRA este butonul de selecție a cifrelor care constituie codul dulapului.

Ieșirile sunt:

- Starea INTRODU_CARACTERE care este prezentata de un led care ne arata ca putem începe sa setam cifrul.
- Led-ul care reprezintă starea de LIBER_OCUPAT a dulăpiorului.

2.2. Unitatea de Control si Unitatea de Execuție



Cutia neagra trebuie descompusa pentru a putea găsi componentele implementabile. Vom face o descompunere **TOP-DOWN** a problemei pana când ajungem la circuite cunoscute, iar apoi vom implementa **BOTTOM-UP**.

Prima descompunere a oricărui sistem este una in care vom diferenția intre **logica de control** din sistem si **resursele sistemului**. Logica de control este reprezentata de UC iar resursele sunt reprezentate de UE.

2.2.1. Maparea intrărilor si ieșirilor cutiei mari pe cele doua componente UC si UE

Putem împărți atât intrările cât și ieșirile în 2 categorii: *de date si de control*. Aceasta separare este esențială la început.

- ***Intrări de date:*** valori pentru diferite lucruri (reset, selecțiile UP si DOWN)
- ***Intrări de control:*** buton de resetare, buton de confirmare
- ***Ieșiri de date:*** valori de afișat pentru utilizator (cifrele selectate pentru cifru)
- ***Ieșiri de control:*** semnale de avertizare sau atenționare a utilizatorului, prin care noi putem sa controlam și îndrumam utilizatorul prin funcționarea sistemului.(ledurile LIBER_OCUPAR, INTRODU_CARACTERE)

2.2.2. Determinarea resurselor (UE)

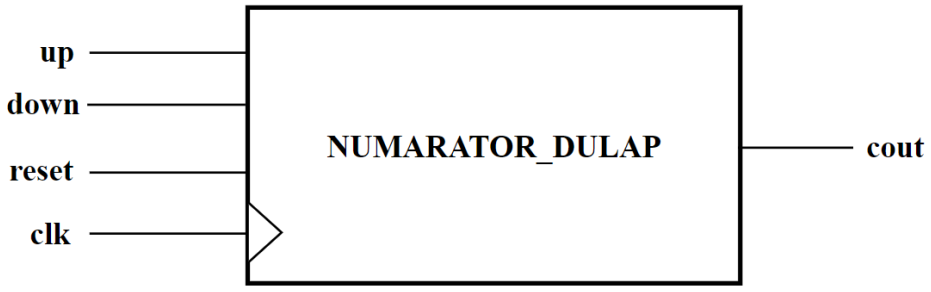
Pentru a stabili mai departe legăturile dintre UC și UE trebuie mai întâi să identificăm resursele pe baza cărora luăm decizii sau resursele care devin informații pentru utilizator. Aceste resurse pot sa genereze semnale către unitatea de control și pot fi controlate de UC prin semnale de Enable sau Reset.

Orice informație pe baza căreia se ia decizii trebuie sa vină de la o resursă care generează acea informație și o transmite mai departe UC.

Resursele pot fi circuite simple, care pot fi implementate direct (numărător, registru etc) sau resurse complexe. Aceste resurse complexe pot apărea în prima descompunere cu cutii negre cărora trebuie sa le stabilim intrări si ieșiri, dar ulterior trebuie descompuse mai departe (de obicei tot în UC și UE) pana când ajungem la circuite cunoscute.

RESURSE

- **Numărător**



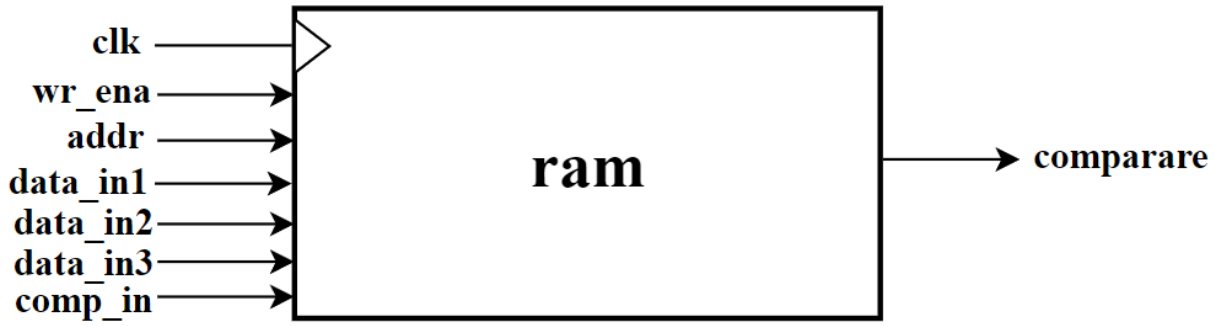
Componenta ***"NUMARATOR_DULAP"*** numără in mod secvențial de la 0 la 15. Acesta are următoarele intrări si ieșiri:

- ***"up"*** (intrare) : Semnal numărare crescătoare
- ***"down"*** (intrare) : Semnal numărare descrescătoare
- ***"reset"*** (intrare) : Semnalul de reset pentru a reseta valoarea contorului la zero
- ***"clk"*** (intrare) : semnalul de ceas pentru a sincroniza operațiile contorului
- ***"cout"*** (ieșire) : afișează valoarea actuală a contorului în format binar pe 4 biți

În arhitectura ***"arh_numarator"***, comportamentul componentei este definit astfel:

- Se utilizează ***MPG-ul*** pentru a detecta marginile de urcare ale semnalului "up" și "down" și a genera semnalele corespunzătoare pentru a sincroniza operațiile contorului;
- Semnalul intern "c" este un vector de 4 biți care reprezintă valoarea curentă a contorului;
- Când se activează semnalul de reset (reset = '1'), contorul este resetat la valoarea de start "0000";
- Se verifică dacă s-au produs margini ascendente ale semnalului de ceas (rising_edge(up_F))
- Valoarea curentă a contorului "c" este atribuită semnalului de ieșire "output";

- **Memorie RAM**



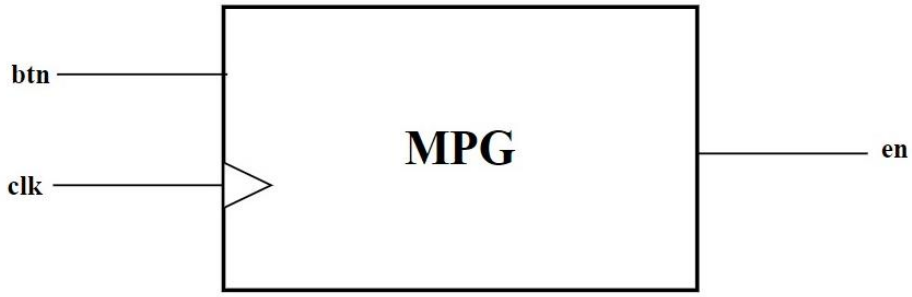
Componenta ***"ram"*** este o memorie RAM care are si functionalitatea de a scrie si de a compara. Aceasta are urmatoarele intrari si iesiri:

- ***"clk"*** (intrare) : Semnalul de ceas care sincronizează operațiile componente.
- ***"wr_ena"*** (intrare) : Semnal de scriere care activează scrierea în memoria RAM.
- ***"addr"*** (intrare) : Adresa la care să se facă operația de scriere.
- ***"data_in1", "data_in2", "data_in3"*** (intrări) : Datele de intrare care trebuie scrise în memoria RAM pentru fiecare dintre cele trei cifre.
- ***"comp_in"*** (intrare): Semnal care declanșează operația de comparare între datele salvate în memorie și cele introduse.
- ***"comparare"*** (ieșire): Ieșirea care indică rezultatul comparării. Este setată la '1' dacă datele introduse și cele din memorie coincid, altfel este setată la '0'.

Această memorie RAM este una generica, parametrul size, indică numărul de locații de memorie, iar fiecare locație este de lungime d_width (lungimea cuvântului).

Procesul din arhitectura "arh_ram" realizează scrierea datelor în memorie la falling_edge(clk) și compararea datelor la rising_edge(clk), când semnalul comp_in este activat. Compararea este efectuată între datele salvate în memorie (reprezentate de ram(0), ram(1), ram(2)) și cele introduse recent (reprezentate de ram(3), ram(4), ram(5)). rezultatul este scris în ieșirea ***comparare***.

- **MPG**



Componenta “MPG” generează un impuls la ieșirea ***“en”*** atunci când se detectează o secvență specifică de impulsuri de ceas și când butonul ***“btn”*** este activat.

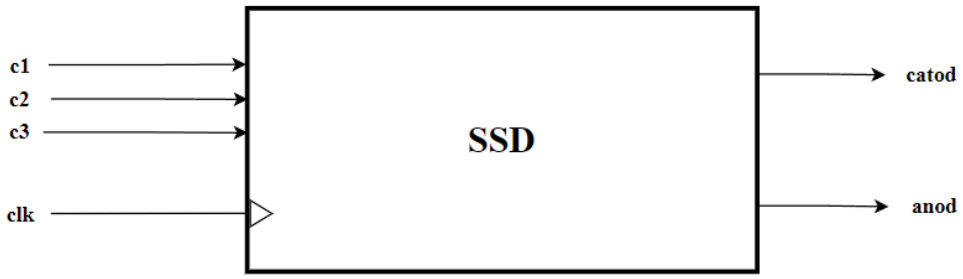
Aceasta are următoarele intrări si ieșiri:

- ***“btn”*** (intrare) : semnalizează dacă butonul este activat sau nu
- ***“clk”*** (intrare) : semnalul de ceas care controlează funcționarea componentei
- ***“en”*** (ieșire) : furnizează impulsul generat în urma detectării secvenței specifice de impulsuri de ceas și activării butonului

In cadrul arhitecturii ***“arh_MGP”*** au loc următoarele acțiuni:

- Este definit pentru a incrementa contorul la fiecare front de urcare al semnalului de ceas;
- Se atribuie semnalului t valoarea '1' când contorul count atinge valoarea hexazecimală FFFF, altfel, acesta rămâne '0';
- Este definit pentru a măsura starea semnalului btn și a o atribui semnalului q1 în momentul în care semnalul t devine '1';
- Se propagă semnalele q1, q2 și q3 la fiecare front de urcare al semnalului de ceas;
- Semnalul **en** este atribuit ca produs logic între semnalul q2 și negarea semnalului q3.

- **SSD**



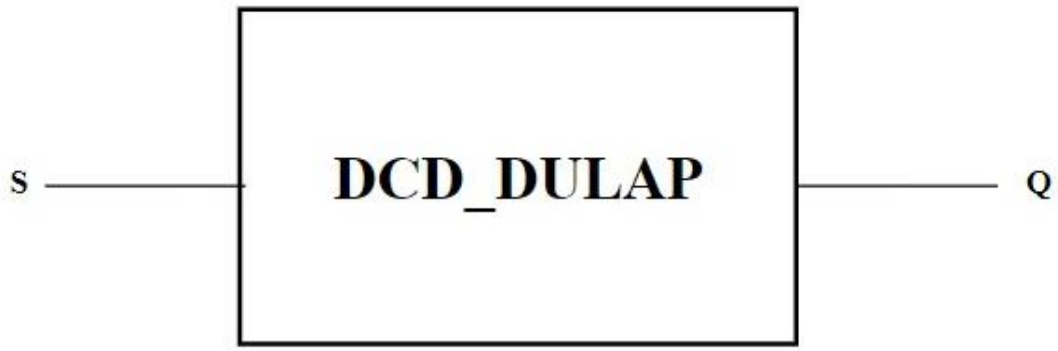
Componenta “SSD” controlează un afișaj cu șapte segmente multiplexat pentru a afișa trei cifre hexazecimale.

Aceasta are următoarele intrări si ieșiri:

- ***“clk”*** (intrare): acesta este semnalul de ceas de intrare utilizat pentru sincronizarea operațiunilor modulului.
- ***“c1, c2, c3”*** (intrari)
- ***“anod”*** (ieșire): acesta este semnalul de control pentru anozii cifrelor afișajului cu șapte segmente. Fiecare bit corespunde unei cifre, unde 0 activează cifra respectivă.
- ***“catod”*** (ieșire): acesta este semnalul de control pentru segmentele afișajului. Fiecare bit controlează un segment specific, unde 0 aprinde segmentul respectiv.

Când semnalul clk se schimbă de la 0 la 1 **(**rising edge), contorul count este incrementat. Valoarea celor mai semnificativi 2 biți ai contorului este utilizată pentru a determina care dintre cele patru cifre ale afișajului trebuie activată și care 4 biți din data trebuie afișați pe cifra respectivă.

- **Decodificator**



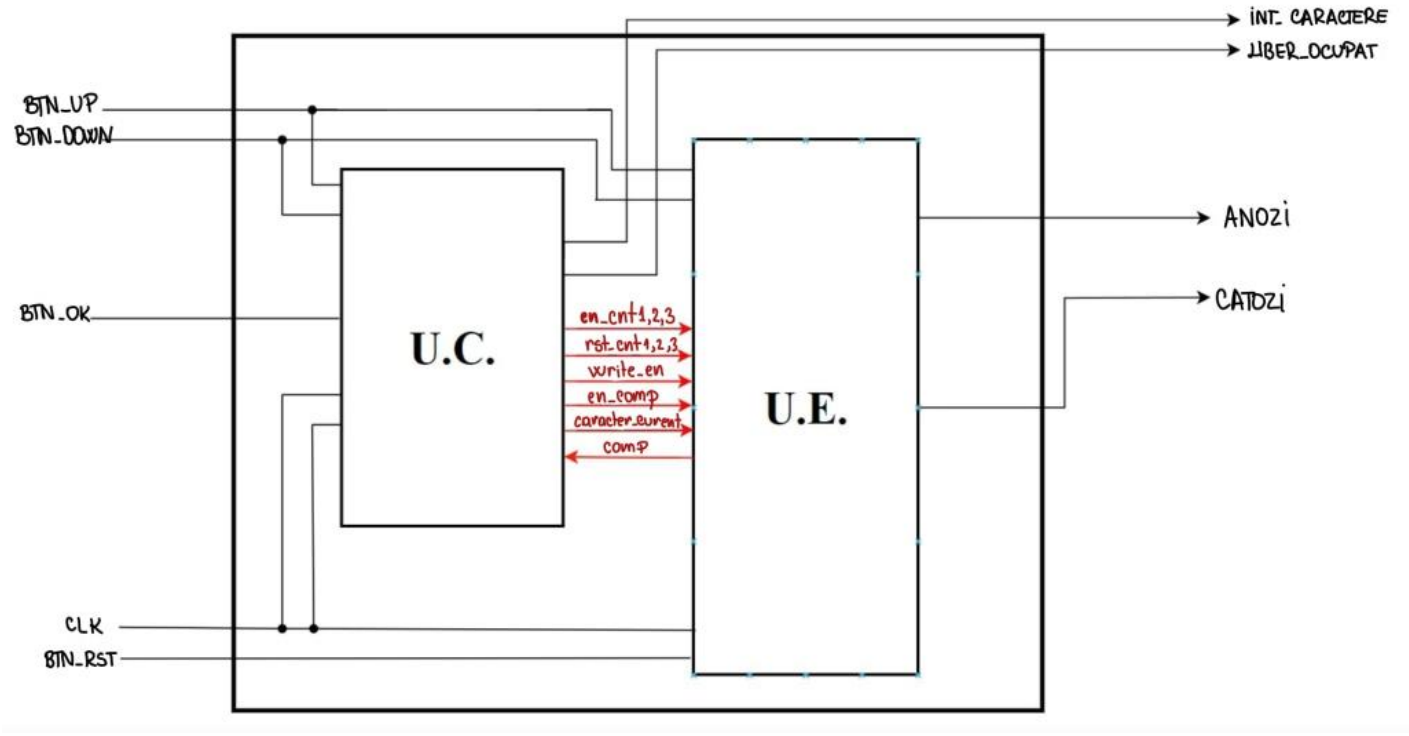
Componenta “DCD” definește un decoder pentru un afișaj cu 7 segmente, care poate afișa cifrele și câteva litere și simboluri.

Acesta are următoarele intrări si ieșiri:

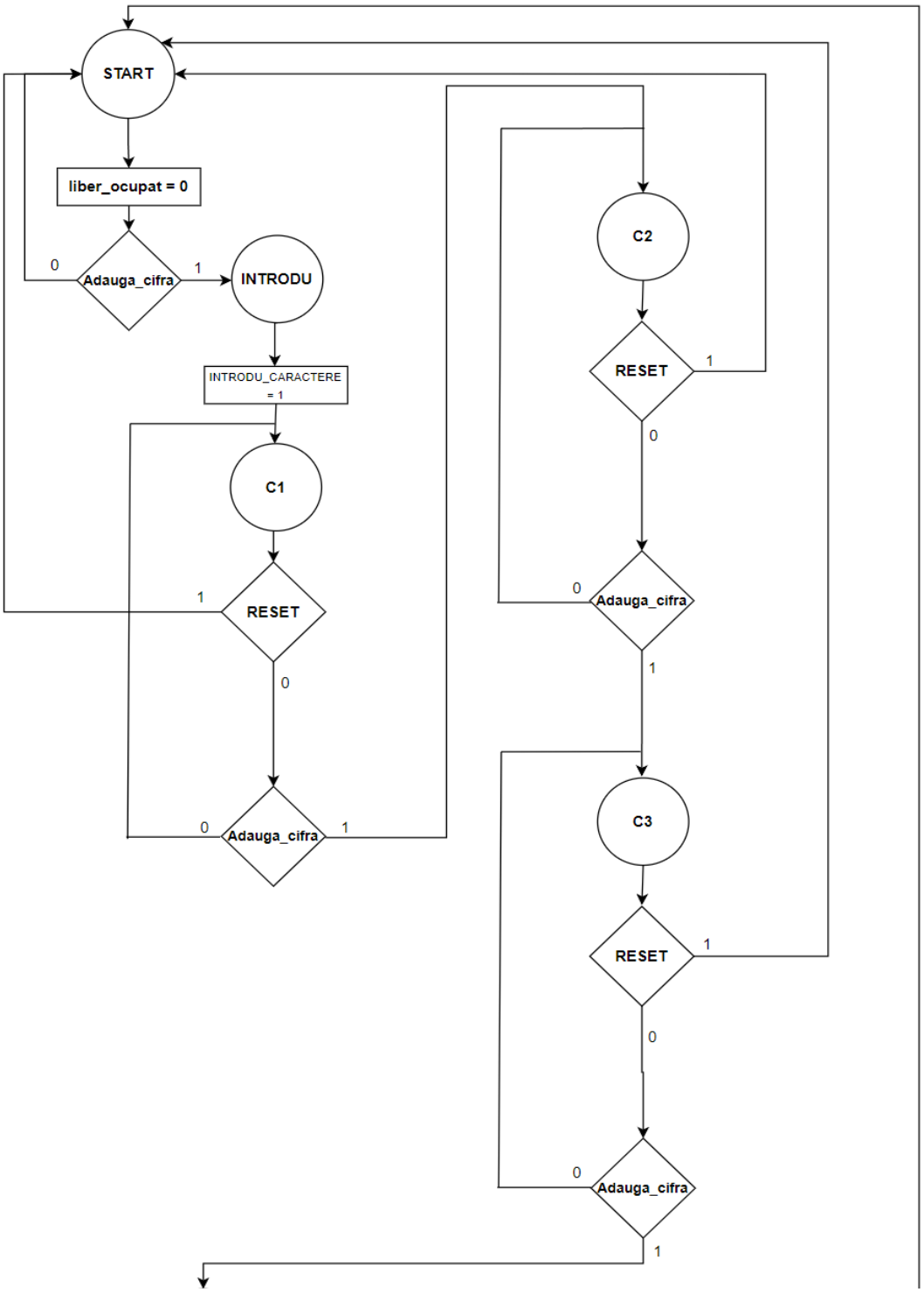
- ***“S”*** (intrare): un vector logic cu 4 biți, utilizat pentru a selecta cifra care trebuie afișată.
- ***“Q”*** (iesire): un vector logic cu 7 biți, care reprezintă afișarea pe display a cifrei selectate.

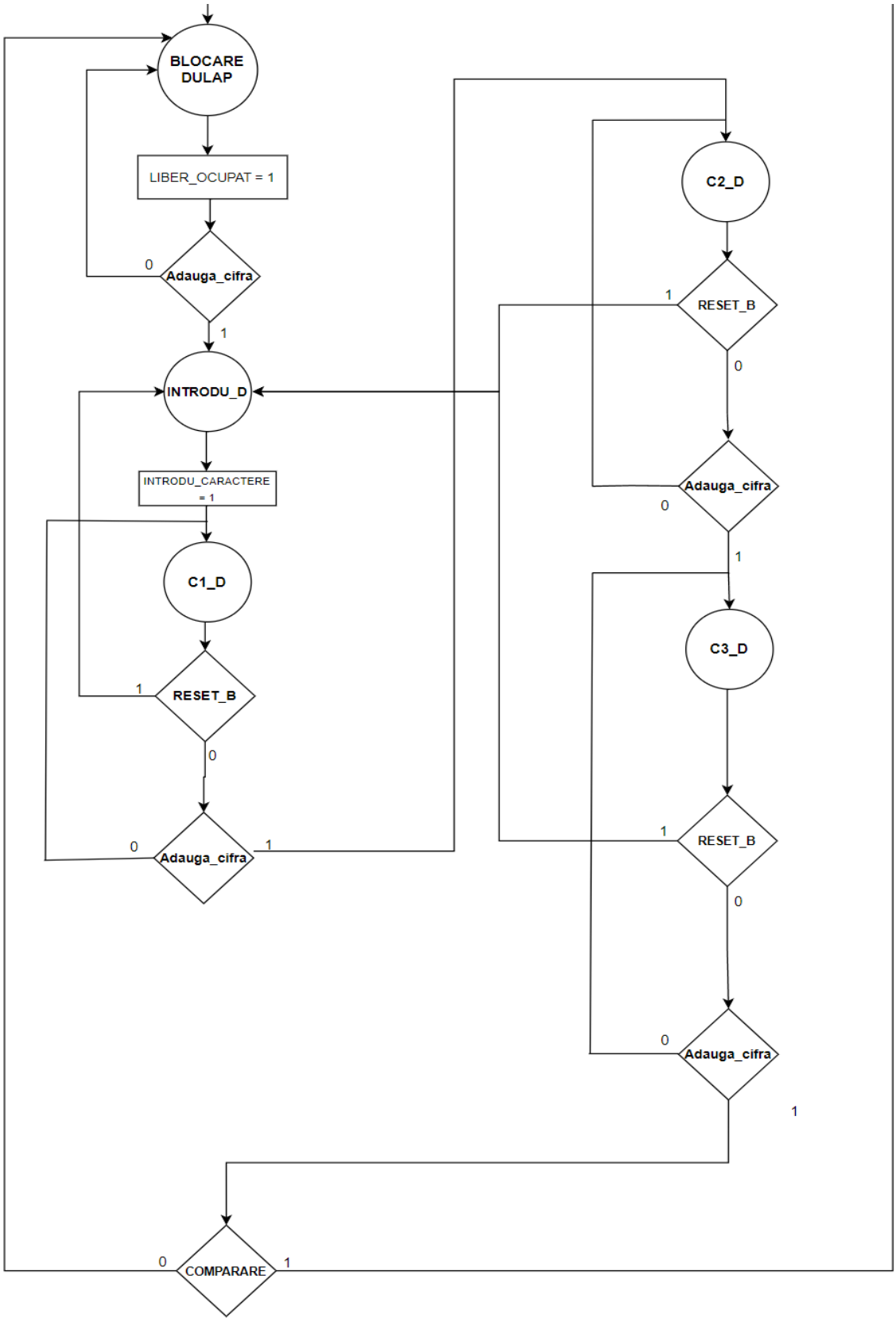
Definește un decoder pentru un afișaj cu 7 segmente, care poate afișa cifrele și câteva litere și simboluri. Intrarea S selectează cifra sau simbolul care trebuie afișat, iar ieșirea Q activează/dezactivează segmentele necesare pentru a afișa acea cifră sau simbol.

2.2.4. Schema bloc a primei descompuneri

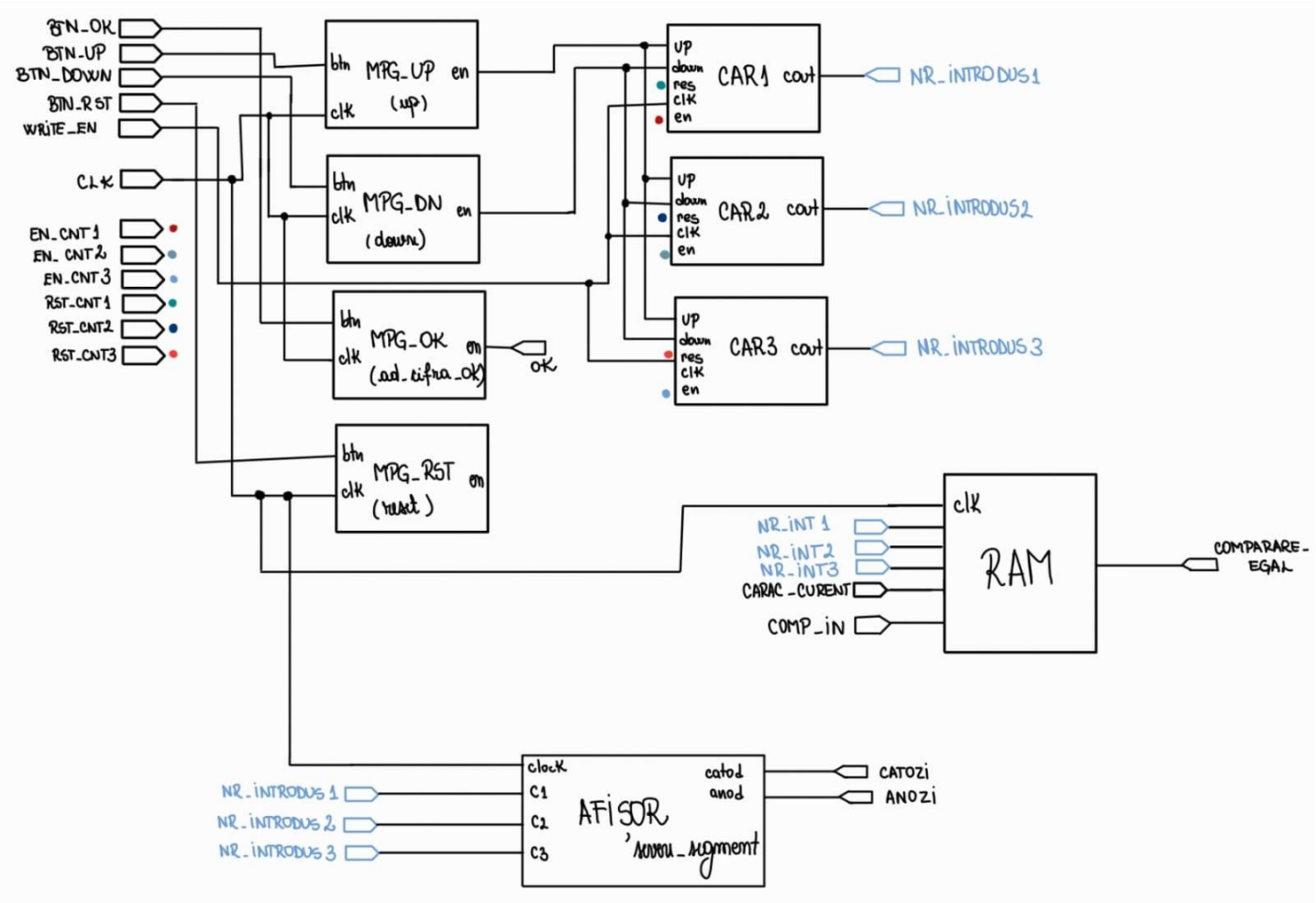


2.2.4. Reprezentare UC prin diagramă de stări (organigramă)




BLOCARE DULAP
LIBER_OCUPAT = 1
Adauga_cifra
0
1
INTRODU_D
INTRODU_CARACTERE = 1
C1_D
RESET_B
1
0
Adauga_cifra
0
1
C2_D
RESET_B
1
0
Adauga_cifra
0
1
C3_D
RESET_B
1
0
Adauga_cifra
0
1
COMPARARE
0
1

2.2.5. Schemă de detaliu a proiectului



3.MANUAL DE UTILIZARE SI INTRETINERE

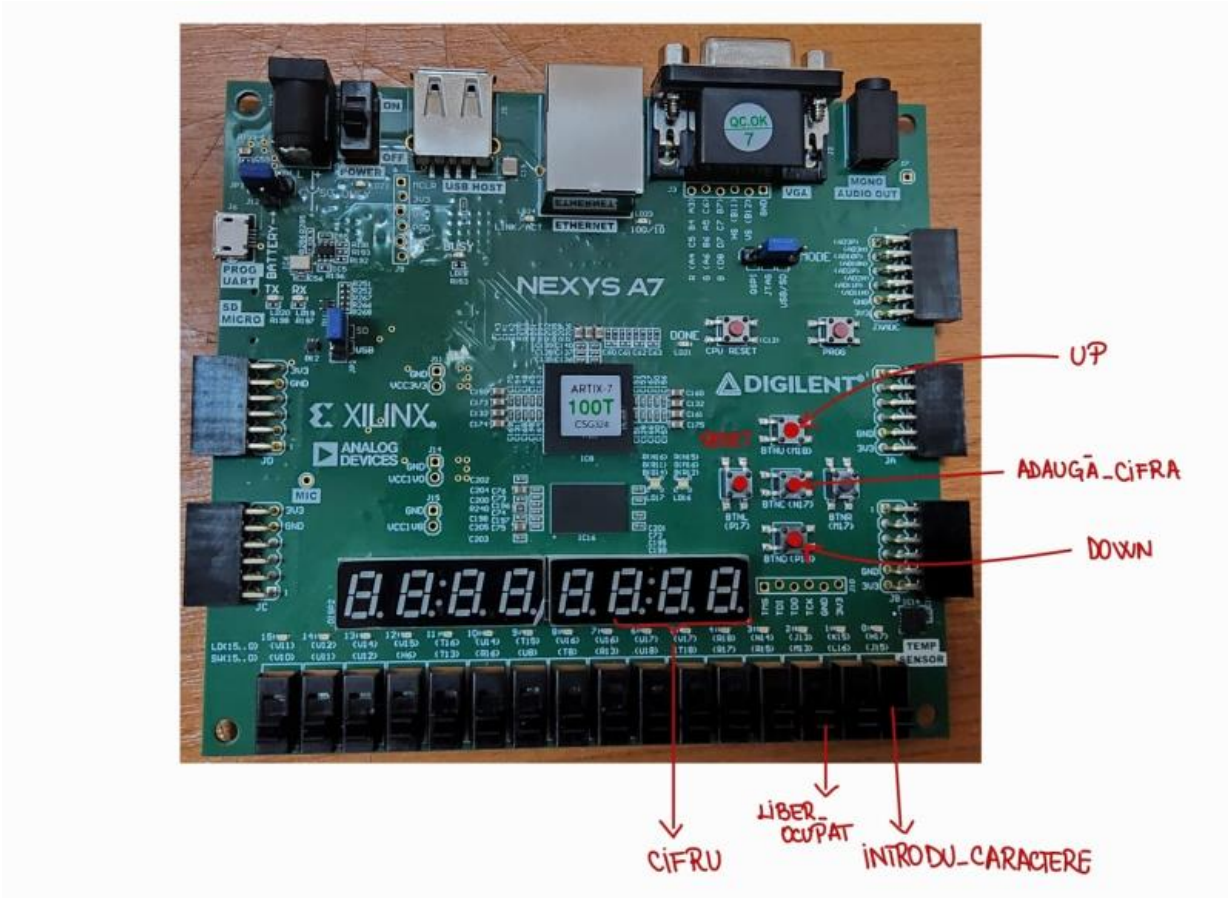
Utilizatorul va apăsa butonul **adauga_cifra** pentru a începe setarea codului de 3 cifre pentru dulăpior, iar ledul **introdu_caractere** se va aprinde.

Selectarea cifrei dorite se va face cu cele doua butoane de direcție, **up** si **down**, iar pentru a trece la următoarea cifra se va apasă din nou butonul **adauga_cifra**.

Dupa inserarea tuturor cifrelor ledul **introdu_caractere** se va stinge, iar ledul **liber_ocupat** se va aprinde. In acest moment dulăpiorul e blocat.

Pentru deblocare se vor repete primii doi pași, iar in caz de succes, dupa 5 apăsări ale butonului **adauga_cifra**, led-ul **liber_ocupat** se va stinge, iar circuitul va trebui blocat cu un nou cifru. In ca contrar nu se va întâmpla nimic, si va trebui sa încercam din nou introducerea unui cifru.

In eventualitatea unei greșeli in momentul in care se setează codul exista butonul **reset**, care ne va aduce in starea inițiala indiferent de stadiul in care ne aflam.



4. JUSTIFICAREA SOLUTIEI ALESE

Pentru implementarea memorie am ales sa folosim o memorie RAM in care am memorat toate cele 6 caractere introduse (3 pentru cifrul care a blocat dulapul si inca 3 pentru cifrul al doilea) . Operatia de comparare se efectueaza tot in cadrul memoriei, intre primele 3 caractere introduse si urmatoarele 3.

In momentul afisarii numerelor pe SSD am decis sa folosim un decodificator pentru a ne fi mai usor sa le afisam.

După ce facem compararea salvam într-o variabila rezultatul, in caz favorabil deblocam dulapul, iar in caz contrar ne întoarcem in starea de deblocare.

5.POSIBILITATEA DE DEZVOLTARE ULTERIOARA

În ceea ce privește dezvoltarea ulterioară a sistemului cu cod cifru pentru securizarea dulapurilor, există mai multe direcții posibile de îmbunătățire și extindere a funcționalităților, printre care:

- **Integrarea cu un sistem de alertă**:

  Adăugarea unui modul care să trimită notificări în cazul în care se încearcă deblocarea dulapului cu un cod greșit de mai multe ori consecutiv. Aceste notificări pot fi sub formă de alerte sonore sau mesaje trimise către un telefon mobil.

- **Extinderea capacității de memorare**:
  Creșterea numărului de combinații posibile de cifru prin extinderea setului de caractere disponibile sau mărirea numărului de caractere din codul cifru.

BIBLIOGRAFIE

- Cursurile domnului profesor Dragoș Florin Lisman
- Octavian Creţ, Lucia Văcariu – Limbajul VHDL. Îndrumător de laborator, Ediţia a treia completată şi revizuită.
- Manual de referință a limbajului VHDL, IEEE Std 1076, 2000 Edition