**Automat Bancar**

-FPGA Basys2-

Proiect realizat de Angheluș Diana și Jișa Diana

Facultatea de Automatică și Calculatoare

An I de studiu

Grupa 30214

Profesor coordonator

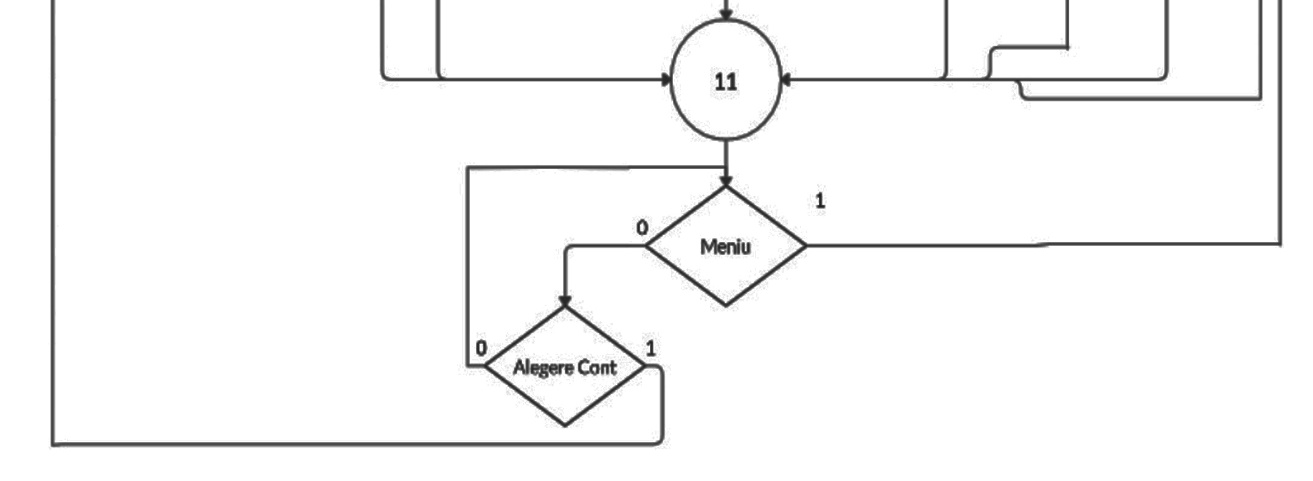
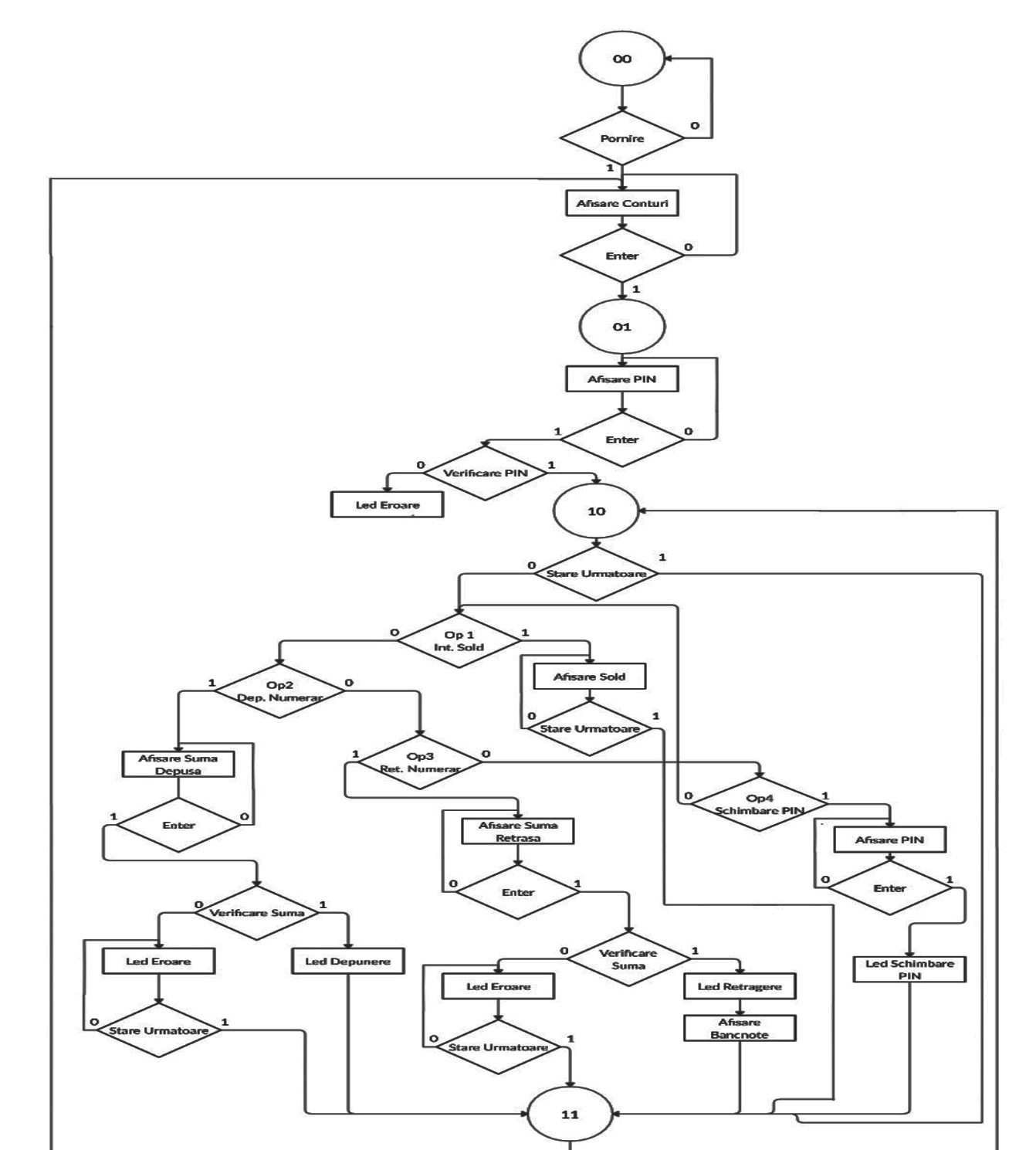
As. Ing. Diana Pop

**Cuprins**

1. **Specificație proiect**
2. **Diagrama block**
3. **Schema bloc**
4. **Proiectare**
5. **Componente utilizate**
6. **Semnificația notațiilor**
7. **Simularea**
8. **Instrucțiuni de utilizare și folosire**
9. **Justificarea soluției alese și posibilități de dezvoltare ulterioară**
10. **SPECIFICAȚII ALE PROIECTULUI**

Să se proiecteze un automat bancar care permite extrageri de sume în EURO. Se presupune că suma maximă care poate fi extrasă o dată este de maximum 1.000 EURO. Inițial se efectueaza identificarea cardului si se alege operația. Vor fi suportate minim 4 carduri/conturi diferite si se vor implementa minim 4 operații diferite. Automatul dispune de o casă in care inițial se introduce o anumită sumă (numar de bacnote de diferite valori). În cazul cererii de eliberare de numerar se introduce suma, se verifica existența sumei cerute, se vizualizeaza tipurile de bacnote emise si se actualizează contul. Apoi se eliberează cardul, suma si, eventual, chitanța. Proiectul va fi realizat de 2 studenți.

1. **DIAGRAMA BLOC**



000

001

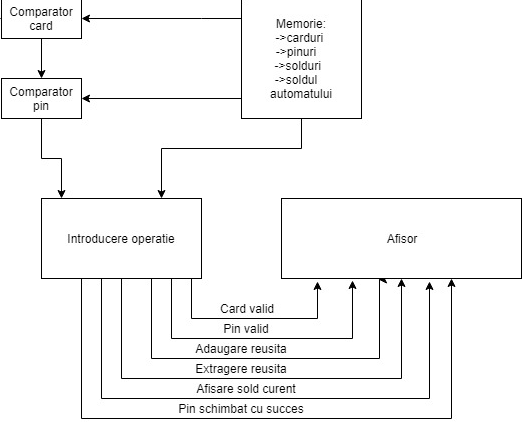
010

011

100

101

1. **SCHEMA BLOC**



1. **PROIECTARE**

Proiectul realizat de noi este un simulator al unui Automat Bancar care execută 4 operațiuni: interogare sold,depunere numerar,retragere numerar,schimbare pin.

Întrucât nu avem cum sa implementăm practic un scanner de card am implementat o optiune care alege 4 carduri cu o suma si un pin corespunzător fiecăruia.

Se alege cardul/contul, apoi se introduce pin-ul respectiv cardului iar în cazul în care pin-ul este corect se va trece în starea urmatoare a automatului care este cea de alegere a operațiunilor. S-au implementat 4 operatiuni cu posibilitatea de a executa mai multe operațiuni într-o simulare pe un cont ales.

În cazul în care este selectata depunerea numerar trebuie introdusă suma de depunere care poate fi maxim 1000 EURO si apoi se trece in starea urmatoare de decizie unde se poate reveni la o alta operațiune sau se poate încheia procesul de simulare.

Daca este selectata retragerea numerar atunci trebuie introdusă suma de retragere,iar daca aceasta este mai mică decât suma din cont si exista bancnote disponibile în automat atunci se elibereaza un număr de bancnote care vor fi afișate pe anozii cu segmente.

La operațiunea de interogare sold se va afișa pe anozi suma respectiva din contul ales la început.

Partea de schimbare pin va fi asemanatoare cu cea a introducerii pin-ului numai ca aici pin-ul va fi modificat în baza de date.

Am decis ca la toate operatiunile sa se afișeze cifrele pe LED-urile cu 7 segmente, iar la fiecare operațiune care are succes se aprinde un led.

1. **COMPONENTE UTILIZATE**

1. Divizor de frecvență pentru afișare

2. Divizor de frecvență 1Hz

3. Inițializator Bancomat

4. Debouncer

5. Counter Alegere Cont

6. Afisor Alegere Cont

7. Cititor PIN

8. Afisor PIN

9. Comparator PIN

10. Afișor Sold

11. Afișor BCD

12. Cititor Suma Introdusă

13. Afișor Suma

14. Sumator pe N biți(depunere)

15. Cititor Suma Retrasă

16. Afișor Sumă Retrasă

17. Scazător N biți (retragere)

18. Distribuitor Bancnote

19. Afișor Bancnote

20. Cititor PIN nou

21. Afișor PIN nou

**1. Divizor de frecvență 1Hz**

Primul divizor(cel principal) e folosit pentru procesele interne, emițând un semnal de tact pentru fiecare 50.000 de semnale emise de clock-ul sistemului. Divizorul poate fi resetat. Ca intrare va fi **system\_clk** care reprezintă semnalul de tact la frecvența de **100Mhz** al plăcii. **Clk** este ieșirea componentei adică semnalul de tact divizat la frecvența de **1Hz.**Se poate da și reset acestei componente folosind intrarea **clear** pe **1.**

clear

freq\_divider

clock\_out

system\_clk

**2. Divizor de frecvență pentru afișare 1KHz**

Cel de-al doilea divizor este folosit pentru debouncer și la afișarea conturilor disponibile, ce sunt ilustrate pe un afișor BCD 7 segmente pe rând, odată la **0.001s**. Ca intrare va fi **system\_clk** care reprezintă semnalul de tact la frecvența de **100Mhz** al plăcii. **Clk** este ieșirea componentei adică semnalul de tact divizat la frecvența de **1KHz**

cLK\_1KHZ

system\_clk

clk

**3.Inițializator Bancomat**

Înainte de a fi pornit de către utilizator bancomatul este inițializat. Dispozitivul dispune de patru conturi, fiecare cu PIN specific și cu suma în euro prezentă în el și de un anumit număr de bancnote de 100, 50 respectiv 10 euro. În urma modificărilor aduse valorilor inițiale, componenta va înregistra fiecare schimbare (sumă nouă/PIN nou) și o va face disponibilă pentru următoarea operațiune.**Cont0,cont1,cont2,cont3,pin0,pin1,pin2,pin3** sunt conturile și pinurile respective conturilor din baza de date a automatului, **a0,a1,a2,a3** sunt semnale de decizie pentru suprascrierea conturilor sau a pinurilor în baza de date în cazul modificării lor după operațiuni, enb este un enable pentru inițializare ( activ pe **0** ) și pentru suprascriere ( activ pe **1** ). **Nr100,nr50,nr10** sunt vectori care rețin numărul de bancnote de **100e**,**50e** și **10e** disponibile în automat, iar **n1,n2,n3** sunt copii ale acestor numere. **Pinales** și **contales** sunt vectori care vor fi folosiți la suprascrierea în baza de date a pin-ului și a contului ales după executarea unor operațiuni.

**Pin0 --4721**

**Pin1--5832**

**Pin2--6943**

**Pin3 --1967**

**Cont0--1722 E**

**Cont1--2356E**

**Cont2--324E**

**Cont3--50 303E**

**nr100--100 de bancnote de 100e**

**nr50--150 de bancnote de 50e**

**nr10--200 de bancnote de 10e**

**4.Debouncer**

Pentru a evita anumite erori cauzate de suprapunerea nedorită a unor comenzi sau de apasarea într-o manieră repetată a aceluiași buton se folosește un debouncer pentru butonul de ENTER(la introducerea PIN-ului/sumei), a butonului de NEXT, a butonului MENIUP, al ALEGERI unui alt cont și al RESETĂRII unei introduceri (sumă/pin).**Clk** este intrare și reprezintă semnalul de tact la frecvența de **1Hz**, **d** reprezintă butonul conectat de pe placăși **butout** reprezintă ieșirea butonului care va fi folosită în procesele proiectului.

clk

butout

d

debouncer

**5.Counter alegere cont**

Componenta numară de la 0 la 3 și transmite valoarea mai departe spre un afișor.Se va alege aici un cont/card disponibil. 0 fiind indicele pentru primul cont disponibil, iar 3 pentru cel de-al patrulea cont disponibil în baza de date. **Alegere** este un semnal de enable al componentei active pe **1**, **clk** este semnalul de tact la frecvența de **1Hz**, și **t** este un vector folosit pentru incrementare.

t (1:0)

alegere

clk

alegere

**6.Afișor alegere cont**

Primind valoarea de la counter-ul anterior, componenta pune la dispoziție pe cel mai din dreapta afișor **BCD 7 segmente** numărul contului pe care utilizatorul îl poate accesa. Se afișează pe rand cifrele de la **0** la **3** la interval **de 0.001 secunde** între ele(determinat de divizorul de **1 Khz**). Ulterior, utilizatorul poate folosi butonul **ENTER** pentru a alege contul (ex. Daca apasă **ENTER** cand e afișata cifra 2, va accesa cel de-al treilea cont disponibil). Pentru afișare s-au folosit cei patru anozi disponibili(corespunzători celor patru afișoare disponibile) și catozi pentru aprinderea led-urilor corecte.**C1** este numărul contului ales, **clk** este semnalul de tact la frecvența de **1Khz**, **enable** este un enable al acestei componente ( activ pe **1** ) și **an** și **catod** sunt vectori de 4 respectiv de 7 biți corespunzători numărului de anozi folosiți pentru afișare și numărului de catozi ai fiecărui anod.

c1 (1:0)

an (3:0)

afisare\_contalegere

clk

catod (6:0)

enable

**7.Cititor PIN**

Primind semnale de la switch-urile **1,2,3 și 4** (**unități, zeci, sute, mii**) manevrate de utilizator, componenta memoreaza valoarea temporară PIN-ului introdus și o transmite mai departe spre afișare și verificare. Un switch activat reprezintă incrementarea valorii unității respective cu unu(ex. Switch-ul 1 activat pentru 2 secunde determină valoarea **0002**). Se pornește de la **0000**, iar componenta ramane activă pana la apăsarea butonului **ENTER. Clk** este semnalul de tact la frecvența de **1Hz**, **pin\_introdus** este vectorul care reține pin-ul introdus, **pinok** este enable la această componentă activ pe **0** și **reset** poate fi folosit optional pentru a reseta introducerea în caz că se greșește pin-ul.

clk

mii

pinok

pin\_introdus (15:0)

introdpin

reset

sute

unități

zeci

**8.Afișor PIN**

Primind valorile de la componenta anterioară, afișorul pune la dispoziție PIN-ul introdus până în acel timp pe afișoarele **BCD 7 segmente**. Se afișează pe rand cifrele pin-ului (**c1,c2,c3,c4)** în ordinea miilor, sutelor, zecilor și unităților la interval **de 0.001 secunde** între ele(determinat de divizorul de **1 Khz**). **Clk** este semnalul de tact la frecvența de **1Khz**, **enable** este un enable al acestei componente ( activ pe **1** ) și **an** și **catod** sunt vectori de **4** respectiv de **7** biți corespunzători numărului de anozi folosiți pentru afișare și numărului de catozi ai fiecărui anod.

c1 (3:0)

c2 (3:0)

an (3:0)

c3 (3:0)

afisare\_introdpin

c4 (3:0)

catod (6:0)

clk

enable

**9.Comparator PIN**

Componenta compară PIN-ul introdus cu cel al contului ales și decide următoarea stare a sistemului in funcție de verificare.**Checkpin** este enable pentru componentă active pe **1**, **clk** este semnal de tact la frecvența de **1Khz**, **pindeverif** este un vector care reține pin-ul introdus și **pinsetat** este un vector ce reține pin-ul actual al cardului din baza de date.Dacă pin-ul introdus coincide cu cel setat atunci semnalul **pinegal** va deveni **1.**

checkpin

clk

pinegal

pindeverif (15:0)

verifpin

pinsetat (15:0)

**10.Afișor Sold**

Primind valorile de la inițializatorul bancomatului, componenta pune la dispoziție soldul curent, în hexazecimal pe afișoarele **BCD 7 segmente.** **Clk** este semnalul de tact la frecvența de **1Khz**, **enb** este enable pentru componentă activ pe **1**, **sumacont** este un vector care transmite suma contului ales pentru afișare și **an** și **catod** sunt vectori de **4** respectiv de **7** biți corespunzători numărului de anozi folosiți pentru afișare și numărului de catozi ai fiecărui anod.

clk

an (3:0)

interogaresold

enb

catod (6:0)

sumacont (15:0)

**11.Cititor sumă introdusă**

Primind semnale de la switch-urile **2,3** și **4** (**unități, zeci, sute, mii**) manevrate de utilizator, componenta memorează valoarea temporară sumei introduse și o transmite mai departe spre afișare și depunere. Un switch activat reprezintă incrementarea valorii unității respective cu zece, o sută sau o mie (ex. Switch-ul **4** activat pentru **2 sec** determină valoarea **07D0** (**2000** în zecimal). Se pornește de la **0000**, iar componenta ramane activă pana la apăsarea butonului **ENTER**. Suma poate fi resetata de utilizator. **Clk** este semnalul de tact la frecvența de **1Hz**, **suma** este vectorul care reține suma introdusă, **sumaok** este enable la această componentă activ pe **0** și **reset** poate fi folosit optional pentru a reseta introducerea în caz că se greșește suma.

clk

mii

reset

suma (9:0)

introdsuma

sumaok

sute

zeci

**13.Afișor sumă introdusă**

Primind valorile de la cititorul anterior, componenta folosește afișoarele **BCD 7 segmente** pentru a-i dispune utilizatorului suma pe care a introdus-o până la un moment dat(în hexazecimal) . **Clk** este semnalul de tact la frecvența de **1Khz**, **enable** este enable pentru componentă activ pe **1**, **c1,c2,c3** sunt vectori care transmit suma introdusă pentru afișare și **an** și **catod** sunt vectori de **4** respectiv de **7** biți corespunzători numărului de anozi folosiți pentru afișare și numărului de catozi ai fiecărui anod.

c1 (1:0)

an (3:0)

c2 (3:0)

c3 (3:0)

catod (6:0)

afisare\_introdsuma

clk

enable

**14.Sumator N biți**

Sumatorul este format din cascadarea mai multor sumatoare complete pe un bit și este folosit pentru a însuma valoarea introdusă la suma deja prezentă în cont.**Cin** este carry la însumare ce va fi mereu **0**, și **x și y** sunt vectorii ce urmează a fi însumați, rezultatul fiind scris în vectorul **s.**

cin

x (9:0)

s (n:0)

sumatorn

y (n:0)

**15.Cititor sumă retrasă**

Primind semnale de la switch-urile **2,3 și 4** (**zeci, sute, mii**) manevrate de utilizator, componenta memorează valoarea temporară sumei introdue și o transmite mai departe spre afișare și retragere. Un switch activat reprezintă incrementarea valorii unității respective cu zece,o sută sau o mie (ex. Switch-ul **3** activat pentru 2 secunde determină valoarea **00C8** (**200** în zecimal). Se pornește de la **0000**, iar componenta ramane activă pana la apăsarea butonului **ENTER**. Suma poate fi resetata de utilizator. **Clk** este semnalul de tact la frecvența de **1Hz**, **suma** este vectorul care reține suma introdusă, **sumaok** este enable la această componentă activ pe **0** și **reset** poate fi folosit optional pentru a reseta introducerea în caz că se greșește suma.

clk

mii

reset

suma (15:0)

introdsumaret

sumaok

sute

zeci

**16.Afișor sumă retrasă**

Primind valorile de la cititorul anterior, componenta foloseste afișoarele **BCD 7 segmente** pentru a-i dispune utilizatorului suma pe care a introdus-o pana la un moment dat(în hexazecimal) . **Clk** este semnalul de tact la frecvența de **1Khz**, **enable** este enable pentru componentă activ pe **1**, **c1,c2,c3** sunt vectori care transmit suma introdusă pentru afișare și **an** și **catod** sunt vectori de **4** respectiv de **7** biți corespunzători numărului de anozi folosiți pentru afișare și numărului de catozi ai fiecărui anod.

c1 (1:0)

c2 (3:0)

an (3:0)

afisare\_introdsumaret

c3 (3:0)

clk

catod (6:0)

enable

**17.Scăzător N biți**

Scăzătorul este format din cascadarea mai multor scăzătoare complete pe un bit și este folosit pentru a scade din suma deja prezentă în cont valoarea introdusă.**Bin** este borrow in pentru scădere care va fi mereu **0,** și **x și y** sunt vectori care trebuie utilizați la scădere.Se va executa operația “**x-y”** și se va reține rezultatul în vectorul **d.**

bin

x (n:0)

d (n:0)

scazatorn

y (n:0)

**18.Distribuitor bancnote**

Componenta primește ca date de intrare numărul de bancnote de **10,50** respectiv **100** de **euro** disponibile și suma ce trebuie formata din acele bancnote. Folosind un algoritm greedy, se efectuează distribuirea bancnotelor.Dacă se pot distribui bancnote de **100euro** atunci se vor aduna atâtea bancnote posibile cât permite suma de retragere apoi se trece la următorul tip de bancnote și anume cele de **50euro**, analog și pentru cele de **10euro**. **Clk** este semnalul de tact la frecvența de **1Khz**, **enb** este enable la această componentă activ pe **1. Give100, give50, give10** sunt vectori care rețin numărul de bancnote distribuite de **100,50** respectiv **10 euro**, iar **nr100,nr50,nr10** sunt vector ce memorează numărul de bancnote de **100,50 și 10 euro** disponibile în automat.

give10 (3:0)

clk

give100 (3:0)

givebanc

give50 (3:0)

enb

nr10 (7:0)

nr100 (6:0)

suma (9:0)

nr50 (7:0)

**19.Afișor Bancnote**

Folosindu-se de datele determinate de distribuitor, componenta afișează pe placă numărul fiecarui tip de bancnotă folosit în ordine de la stânga la dreapta (**100, 50, 10**). **Clk** este semnalul de tact la frecvența de **1Khz**, **enable** este enable pentru componentă activ pe **1**, **c1,c2,c3** sunt vectori care transmit numărul de bancnote pentru afișare și **an** și **catod** sunt vectori de **4** respectiv de **7** biți corespunzători numărului de anozi folosiți pentru afișare și numărului de catozi ai fiecărui anod.

c1 (3:0)

c2 (3:0)

an (3:0)

afisare\_banc

c3 (3:0)

catod (6:0)

clk

enable

**20.Cititor PIN nou**

Primind semnale de la switch-urile **1,2,3 și 4** (**unități, zeci, sute, mii**) manevrate de utilizator, componenta memorează valoarea temporară a PIN-ului introdus și o transmite mai departe spre afișare și verificare. Un switch activat reprezintă incrementarea valorii unității respective cu unu (ex. Switch-ul **1** activat pentru **2 sec** determină valoarea **0002**). Se pornește de la **0000**, iar componenta ramane activă pana la apăsarea butonului **ENTER**.**Clk** este semnalul de tact la frecvența de **1Hz** și **pin\_introdus** este un vector ce reține pin-ul nou introdus.

clk

mii

pinok

introdpinschmb

pin\_introdus (15:0)

reset

sute

unități

zeci

**21.Afișor PIN nou**

Primind valorile de la componenta anterioară, afișorul pune la dispoziție PIN-ul introdus până în acel timp pe afișoarele BCD 7 segmente. **Clk** este semnalul de tact la frecvența de **1Khz**, **enable** este enable pentru componentă activ pe **1**, **c1,c2,c3,c4** sunt vectori care transmit cifrele pin-ului nou introdus pentru afișare și **an** și **catod** sunt vectori de **4** respectiv de **7** biți corespunzători numărului de anozi folosiți pentru afișare și numărului de catozi ai fiecărui anod.

c1 (3:0)

c2 (3:0)

an (3:0)

afisare\_introdschmbpin

c3 (3:0)

c4 (3:0)

catod (6:0)

clk

enable

1. **SEMNIFICAȚIILE NOTAȚIILOR**

Sunt prezentate toate semnalele folosite, cele din entitatea si din arhitectura unității centrale de comandă.

**unit,zeci,sute,mii**: *in* std\_logic; ---pentru introducere suma **(switch-uri) (1,2,3,4)** **op1,op2,op3,op4**:*inout* **std\_logic;** ----1: depunere numerar  **(switch5)** ----2: retragere numerara **(switch6)**

----3: interogare sold **(switch7)**

----4: schimbare pin **(switch8)**

**startstop**: *in* **std\_logic:='0';** -------------------------pornire automat **(switch)**

**meniubuton**: *in* **std\_logic:='0';** ------------------------------buton meniu operațiuni placă **(jos)**

**nextstate**: *in* **std\_logic:='0';**-----------------------------------buton placă next stare **(buton dreapta)**

**resetsumap**:*inout* **std\_logic:='0';** -----------------resetare sumă/pin introducere **(după debounce)**

**enterrr**: *in* **std\_logic;** -------------------------------------------button enter placa **(mijloc)**

**enterps**:*inout* **std\_logic:='0';** ----enter pentru introducere suma sau introducere pin **(dupa debounce)**

**system\_clk**: *in* **std\_logic;** -------------------------------clock placa **100mhz**

**catod** :*inout* **std\_logic\_vector(6 downto 0):="0000000";** ---------------------catozi afisare  **(pentru netlist)**

**an:***inout* **std\_logic\_vector(3 downto 0):="0111";** ------------------------anozi afisare **(pentru netlist)**

**pinled**:*inout* **std\_logic:='0';** ------------------------------------------- -led aprins dacă pin e corect; **1**

**leddep**: *inout* **std\_logic:='0';** ---------------------------------led aprins dacă a avut loc depunerea  **2**

**ledret**: *inout* **std\_logic:='0';** ----------------------------------led aprins dacă a avut loc retragerea **3**

**schimbpinled**: *inout* **std\_logic:='0';** ------------ led aprins dacă a avut loc schimbarea pin-ului **4**

**errled**: *inout* **std\_logic:='0';** --------------- -led aprins în caz că nu se poate efectua operațiunea  **5**

**signal clk\_1hz**: **std\_logic:='0';** ------------------------------clock divizat de 1 Hz

**signal** clk\_afisare: **std\_logic:='0';** --------------------------clock pentru afișare 1Khz

**signal** initbank: **std\_logic:='0';** ---pentru initializare bancomat

**type** DATE **is array(7 downto 0) of std\_logic\_vector(15 downto 0);**

**signal** BAZA: **DATE**; -------------------bază de date pentru automat

**signal** a1,a2,a3,a4:**std\_logic:='0';** ---------------------decidere alegere cont/card

**signal** pinintrodus:**std\_logic\_vector(15 downto 0):="0000000000000000";** --- pin de introdus

**signal** nextstare**: std\_logic:='0';**----------------------next stare dupa operațiuni**(după debounce)**

**signal** contales: **std\_logic\_vector(15 downto 0):="0000000000000000";** ---cont ales pentru op

**signal** pinales: **std\_logic\_vector(15 downto 0):="0000000000000000";** ---pin ales pentru op

**signal** catod1:**std\_logic\_vector(6 downto 0):="0000000";** ---------------------catozi afisare

**signal** an1: **std\_logic\_vector(3 downto 0):="0111";** ---------------------anozi afisare

**signal** catod2:**std\_logic\_vector(6 downto 0):="0000000";** ---------------------catozi afisare

**signal** an2: **std\_logic\_vector(3 downto 0):="0111";** ---------------------anozi afisare

**signal** catod3:**std\_logic\_vector(6 downto 0):="0000000";** ---------------------catozi afisare

**signal** an3:**std\_logic\_vector(3 downto 0):="0111";** ---------------------anozi afisare

**signal** catod4: **std\_logic\_vector(6 downto 0):="0000000";** ---------------------catozi afisare

**signal** an4:**std\_logic\_vector(3 downto 0):="0111";** ---------------------anozi afisare

**signal** catod5: **std\_logic\_vector(6 downto 0):="0000000";** ---------------------catozi afisare

**signal** an5:**std\_logic\_vector(3 downto 0):="0111";** ---------------------anozi afisare

**signal** catod6: **std\_logic\_vector(6 downto 0):="0000000";** ---------------------catozi afisare

**signal** an6:**std\_logic\_vector(3 downto 0):="0111";** ---------------------anozi afisare

**signal** countcont: **std\_logic\_vector(1 downto 0):="00";** ---------------pentru alegere (numarare)

**signal** stare: **std\_logic\_vector(2 downto 0):="000";** --prima stare a automatului

**signal** give100:**std\_logic\_vector(3 downto 0):="0000";** ---nr. banc de **100E** distribuite

**signal** give50:**std\_logic\_vector(3 downto 0):="0000";** -----nr. banc de **50E** distribuite

**signal** give10: **std\_logic\_vector(3 downto 0):="0000";** ------nr. banc de **10E** distribuite

**signal** nr100: **std\_logic\_vector(6 downto 0):="1100100";** --100 de bancnote de 100e

**signal** nr50: **std\_logic\_vector(7 downto 0):="10010110";** --150 de bancnote de 50e

**signal** nr10: **std\_logic\_vector(7 downto 0):="11001000";** --200 de bancnote de 10e

**signal** n1: **std\_logic\_vector(6 downto 0):=nr100;** ---copie bancnote de 100e

**signal** n2: **std\_logic\_vector(7 downto 0):=nr50;** -----copie bancnote de 50e

**signal** n3: **std\_logic\_vector(7 downto 0):=nr10;** -----copie bancnote de 50e

**signal** meniup: **std\_logic;** -----------------meniu principal **(alegere operațiuni)**

**signal** verifpinn: **std\_logic;** -----------enable la verificare pin

**signal** pinegal: **std\_logic;**-------------daca pin egal

**signal** interogares: **std\_logic;** --------enable interogare sold

**signal** depunerenr: **std\_logic;** --------enable la depunere

**signal** introdsumd: **std\_logic;** --------enable la intrdoducere suma**(depunere)**

**signal** sumadep: **std\_logic\_vector(11 downto 0):="000000000000" ;** ---suma introdusă la depunere

**signal** sumadepfinal: **std\_logic\_vector(15 downto 0):="0000000000000000";**-----suma finala după depunere

**signal** introdsumr: **std\_logic;**--------enable la introducere suma **(retragere)**

**signal** sumaret: **std\_logic\_vector(11 downto 0):="000000000000" ;** ---suma introdusă la depunere

**signal** sumaretfinal: **std\_logic\_vector(15 downto 0):="0000000000000000";**-----suma finală după depunere

**signal** retragerenr: **std\_logic:='0';**--------enable la retragere

**signal** distret: **std\_logic:='0';** ---------------enable la distribuire bancnote

**signal** introdpinschimb: **std\_logic;**------------------enable la introducere pin **(schimbare pin)**

**signa**l pinschimb: **std\_logic\_vector(15 downto 0):="0000000000000000" ;** ---pin introdus la schimbare

**signal** schimbanr**: std\_logic;**----------------------enable la schimbare pin (pentru afișare)

**signal** introdpinc: **std\_logic;** ---------------------enable la afișare introducere pin

**signal** choosepinc: **std\_logic;** -------------------enable la afișare introducere pin

**signal** introdpc: **std\_logic;** ------------------------enable la alegere cont

**signal** afispc: **std\_logic;** ---------------------------enable la afișare alegere cont

**signal** schimbpin: **std\_logic:='0';** --------------------enable la schimbare pin

**signal** resetsumap: **std\_logic;** --resetare pin introducere( dupa debounce)

**signal** resetsumad: **std\_logic;** --resetare pin introducere( dupa debounce)

**signal** resetsumar: **std\_logic;** --resetare pin introducere( dupa debounce)

**signal** resetsumasp: **std\_logic;** --resetare pin introducere( dupa debounce)

1. **SIMULAREA**

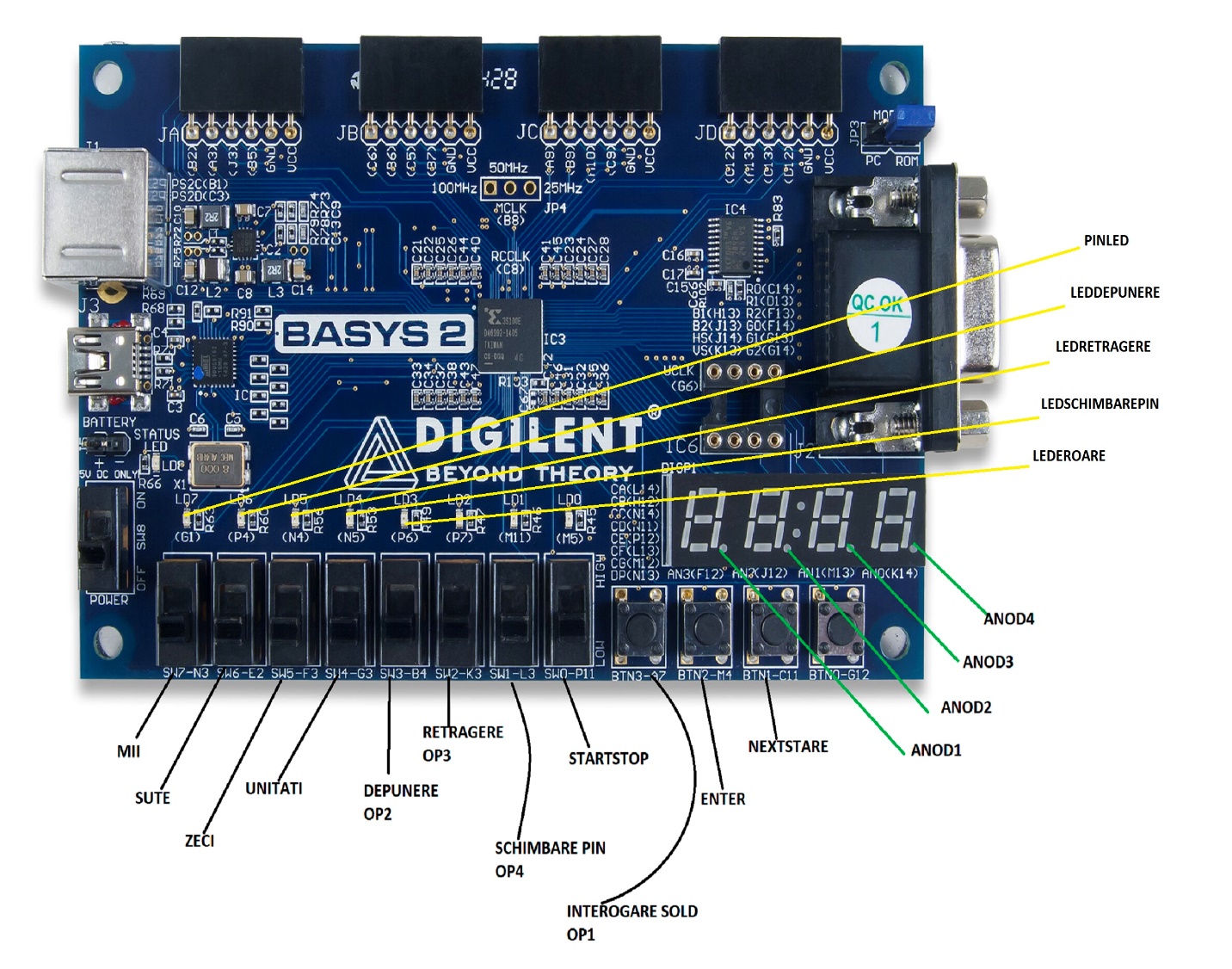
Automatul trece prin mai multe stări însa cele principale sunt date de vectorul stare și de switch-ul startstop.

Când vectorul stare are valoarea **“000”** atunci automatul se afla în starea de inițializare.Aici se decide ce cont si ce pin se va folosi pentru simulare, se resetează la valorile inițiale, se alege contul/cardul sau se oprește automatul în cazul în care am executat ulterior niște operațiuni.

La valoarea **“001”** se va introduce pin-ul respectiv cardului ales și daca pin-ul este corect se trece în starea **“010”** urmată de aprinderea led-ului pinled.

În starea **“010”** se se poate alege una din operațiunile interogare sold,depunere **(“011”),** retragere **(„100”) ,** schimbare pin **(„101”)** folosind comenzile aferente ( switch-urile pentru depunere, retragere, schimbare pin și butonul pentru interogare sold**).**

Pentru revenirea la meniul principal se comută switch-ul startstoppe 0 apoi pe 1.

1. **INSTRUCȚIUNI DE UTILIZARE ȘI FOLOSIRE**

Pentru ca simulatorul să funcționeze trebuie ca **switch**-ul **startstop** să fie setat pe **1.** Apoi

va porni un numărător ce va număra și va afișa pe segmente de la **0** la **3**, ce corespund alegerii cardurilor disponibile (numerotate de la **1** la **4)**. Pentru a selecta cardul dorit se apasă **enter** de pe buton și atunci se va trece în starea următoare.

**Pin0 --4721**

**Pin1--5832**

**Pin2--6943**

**Pin3--1967**

**Cont0--1722 E**

**Cont1--2356E**

**Cont2--324E**

**Cont3--50 303E**

**nr100--100 de bancnote de 100e**

**nr50--150 de bancnote de 50e**

**nr10--200 de bancnote de 10e**

În următoarea stare se va introduce pin-ul corespunzător cardului ales, de la **switch**-urile **unități,zeci,sute,mii**. Daca **switch**-urile sunt setate pe **1** atunci se transmite enable la niște numărătoare care vor număra o data pe secundă și va apărea pe afișoare pin-ul. Atunci cand numărul dorit apare pe afișor, **switch**-ul corespunzător numărătorului **(unități,zeci,sute,mii)** se comută pe 0. Când pin-ul introdus corespunde cu cel al cardului se apasă din nou enter, iar daca este egal se va aprinde un led care va semnala ca pin-ul introdus este correct și se va trece în starea următoare unde se vor alege operațiuni, în caz contrar se va aprinde un led de eroare care corespunde unui pin incorect iar în acest caz va fi nevoie sa introduceți alt pin.

Dupa aceea va fi nevoie sa selectați operațiunea folosind un **switch** corespunzător operațiunilor (**op1**= interogare sold, **op2** = depunere numerar, **op3** = retragere numerar, **op4** = schimbare pin). Pentru a alege, se folosesc comenzile aferente ( **switch-urile pentru** **depunere, retragere, schimbare pin și butonul pentru interogare sold).Dupa selectarea operatiunii dorite se apasa butonul nextstare.**

În cazul interogării soldului,se apasa butonul corespunzator operatiei si se va afișa pe segmente suma din contul ales, iar daca se dorește o alta operațiune este nevoie sa se revina la meniu, comutand switch-ul de startstop,se va selecta din nou contul si se va introduce din nou pinul dupa care **se comută swtich-ul operațiunii dorite pe 1 și apoi** se va apăsa pe butonul **nextstare.**

La depunere numerar va trebui din nou introdusă suma, introducerea fiind bazată pe același principiu ca la introducerea pinului, numai că aici cand numărătorul este pornit va aduna **10** pentru **switch**-ul de **zeci**, **100** pentru **switch**-ul de **sute**, și **1000** pentru **switch**-ul de **mii**.Se va introduce pe rând ( ori **zeci**, ori **sute**, ori **mii**), iar cand suma dorită apare pe afișor se va opri **switch**-ul. Se va apăsa apoi  **enter,** iar daca suma este mai mică decât **1000E** atunci se va aprinde un led care semnalează că operațiunea a avut loc cu succes si se trece în starea următoare, iar în caz contrar se aprinde led-ul corespunzător unei erori și va fi nevoie de introducerea unei noi sume.

La retragere numerar va trebui din nou introdusă suma, introducerea fiind bazată pe același principiu ca la introducerea pinului, cand numărătorul este pornit va aduna **10** pentru **switch**-ul de **zeci**, **100** pentru **switch**-ul de **sute**, și **1000** pentru **switch**-ul de **mii**.Se va introduce pe rând ( ori **zeci**, ori **sute**, ori **mii**), iar cand suma dorită apare pe afișor se va opri **switch**-ul. Se va apăsa apoi  **enter,** iar daca suma este mai mică decât **1000E** și există bancnote pentru a plăti suma dorită și există acea sumă în cont atunci se va aprinde un led care semnalează că operațiunea a avut loc cu succes si se trece în starea următoare, iar în caz contrar se aprinde led-ul corespunzător unei erori și va fi nevoie de introducerea unei noi sume.

Ultima operațiune este cea de schimbare a pin-ului unde se introduce un nou pin, introducere bazată pe principiul anterior al introducerii pin-ului în primă fază (introducerea pentru verificare a pin-ului). Când pin-ul dorit a fost introdus, se apasă **enter**, și se va semnala cu un led că schimbarea pin-ului a avut loc cu succes.

Se poate opri simulatorul prin comutarea **switch**-ului **startstop** pe **0**, și revenirea la meniul principal din nou prin **comutarea switch-ului startstop pe 1.**

Graphical user interface, application

Description automatically generatedPentru a integra proiectul pe placă este nevoie de un program de sinteză și mapare a semnalelor folosite în cod, iar noi am decis să folosim Xilinx ISE Design Suite 14.7. La **Design Properties** se va alege ca unitate de execuție placa **SPARTAN3E XA3S100E-4CPG132.** Va fi nevoie de fișierul de constrângeri a semnalelor din entitate, după care se va da dublu-click pe **Generate Programming File** după care se va deschide o fereastră.

Graphical user interface, application

Description automatically generated

Se va executa **dublu-click** pe **Boundary Scan,** apoi **click-dreapta** pe **Initialize Chain**. Va apărea apoi o imagine ce simbolizează arhitectura plăcii **Basys2**  cu denumirea **XC3S100E**.

Ultima parte constă în adăugarea fișierului **.bit** din proiect după care se execută **click-dreapta** pe simbolul **XC3S100E** și se va alege **Program.**

1. **JUSTIFICAREA SOLUȚIEI ALESE ȘI POSIBILITĂȚI DE DEZVOLTARE ULTERIOARĂ**

Am ales acest tip de implementare pentru acest automat bancardeoarece am considerat ca in acest mod putem folosi un numar minim de switch-uri si butoane pentru a usura experienta utilizatorului.Atat pentru introducerea unei sume cat si pentru introducerea pinului am ales sa utilizam numaratore in locul registrelor, reusind astfel sa reducem timpul necesar introducerii pinului si a unei sume.Totodata,pentru compunerea unei sume am ales sa utilizam algoritmul greedy,acesta fiind cel mai des intalnit si usor de implementat.

Ca dezvoltari ulterioare,se pot adauga mai multe carduri in memorie si operatiuni destinate utilizatorilor(de exemplu transferul bancar).Trecerea de la o operatiune la alta ar putea fi realizata fara a mai fi nevoie sa se revin la meniul principal si sa se introduca din nou pinul. Se poate face un decodificator din hexazecimal în zecimal, pentru a observa toată suma în zecimal pe afișare deoarece ea este afișată în hexzecimal.La operatiunile de depunere si extragere numerar s-ar putea afisa soldul nou al cardului imediat dupa efectuarea operatiunii fara a mai fi nevoie sa se efectueze operatiunea de interogare sold.

# BIBLIOGRAFIE

* Îndrumator de Laborator VHDL (<http://users.utcluj.ro/~lucia/activdid.html>)
* <https://www.wikipedia.com/>
* <https://www.ics.uci.edu/~jmoorkan/vhdlref/>
* <https://reference.digilentinc.com/reference/programmable-logic/nexys-4-ddr/reference-manual>