

FACULTY OF AUTOMATION AND COMPUTER SCIENCE

DIGITAL SYSTEM DESIGN

VHDL PROJECT

AUTOMATED TELLER MACHINE

(ATM)

COMMAND UNIT AND EXECUTION UNIT

DESIGNED AND IMPLEMENTED BY:

Chereji Iulia

Iepure Ana

COORDINATING TEACHER:

Miclea Vlad

GROUP:

30414

CONTENTS

- 1. Project specification
- 2. Project description3. Project implementation
- 4. Final conclusions

1. Project specification

Design a banking machine for withdrawals in EURO.

We have implemented a banking machine, which performs the following operations:

- cash withdrawal
 - cash entry
- display balance
 - change pin

In case of cash withdrawal, the existence of the necessary amount in the account is verified, and then the respective amount is extracted.

In case of entering the amount, enter the desired amount, then add the amount to the account

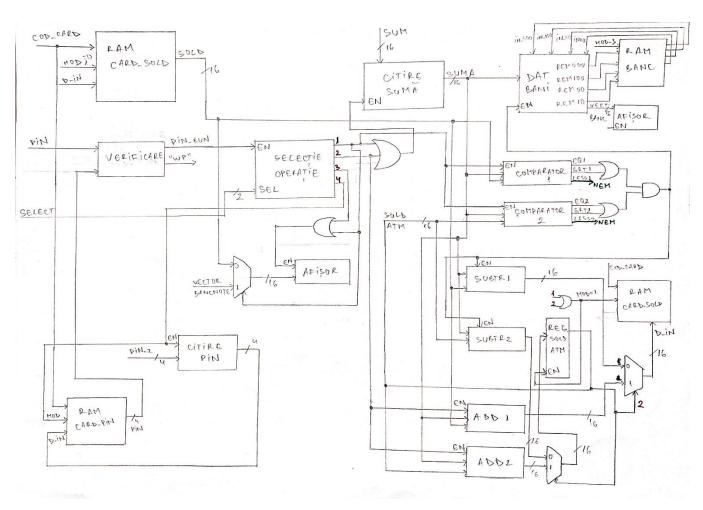
The third operation is to display the balance, it aims to display the amount present in an account.

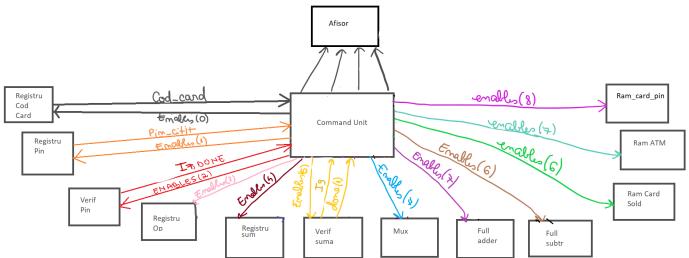
The fourth operation is to change the pin of an account: change the pin in the memory and replace it with a new one.



2. Project description

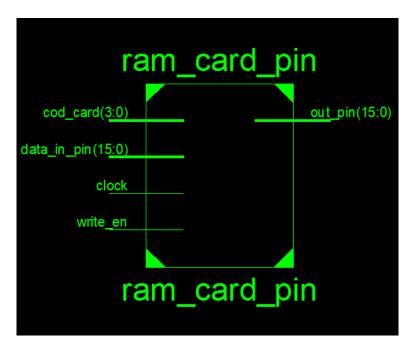
2.1 BLOCK DIAGRAM

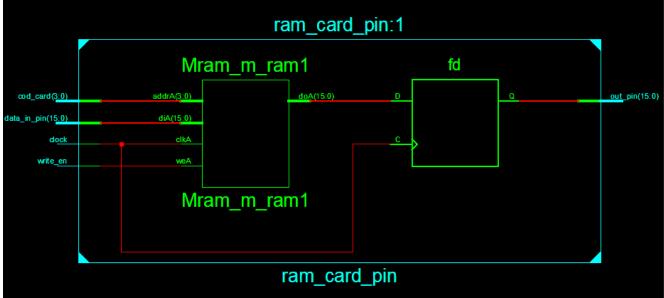




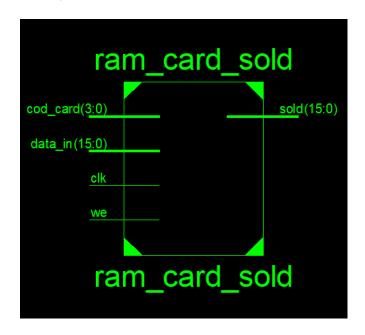
2.2 EXECUTION UNIT - COMPONENTS

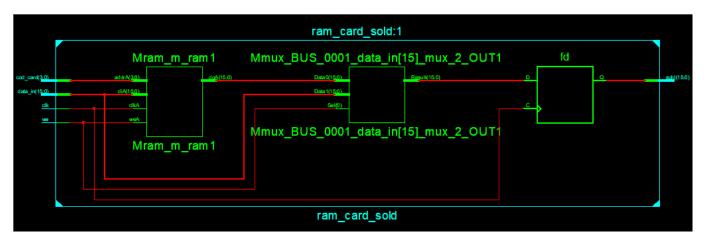
CARD PIN RAM: this RAM contains the description of every card, the id code which corresponds to a specific pin. In our case, we have 15 such cards and the possibility to introduce more.



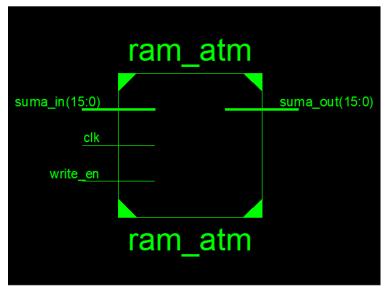


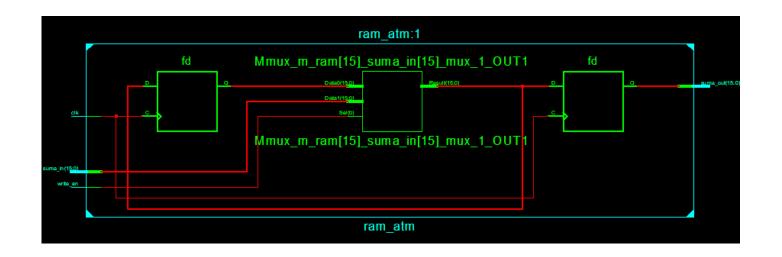
BALANCE CARD RAM: contains the amount of money corresponding to every card.



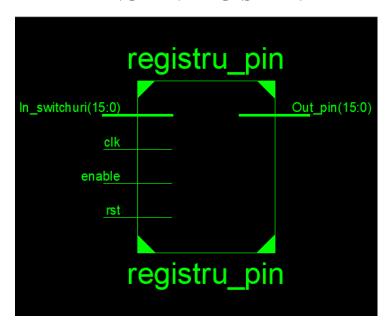


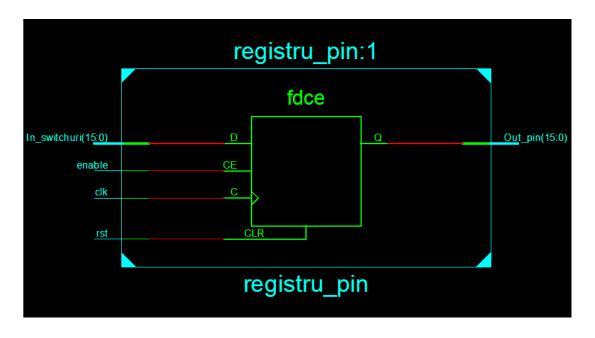
RAM FOR THE EXISTING AMOUNT OF MONEY: this memory stores the available sum that the ATM has.



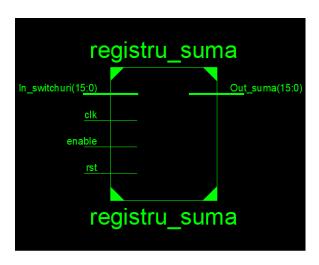


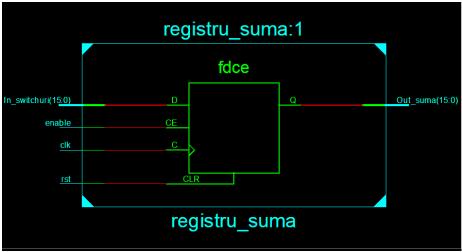
READING PIN REGISTER:



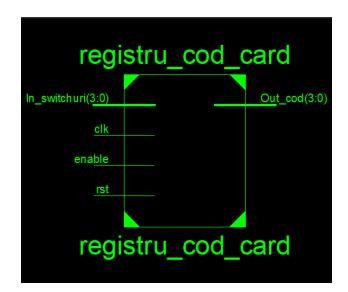


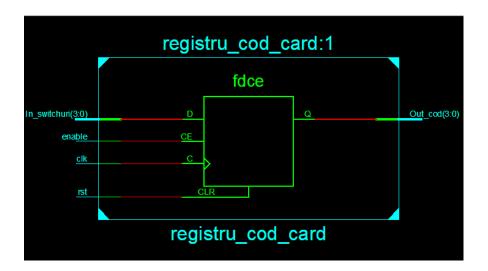
REGISTER FOR READING THE SELECTED SUM: if the user selects cash withdrawal or cash entry option, he is required to introduce the sum of money to add or extract from the account. This component reads the introduced sum of money.



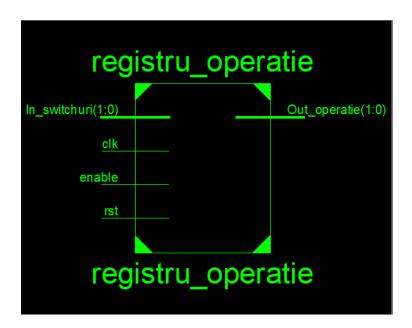


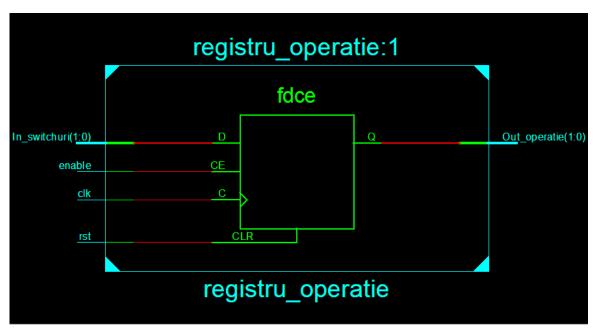
REGISTER FOR READING THE CARD CODE:



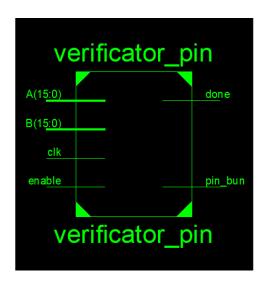


REGISTER FOR READING THE OPERATION:

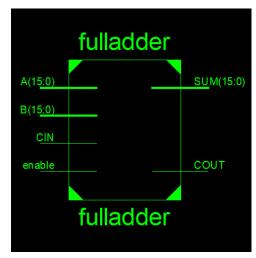




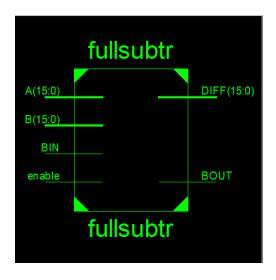
SELECTED PIN VALIDATION: it verifies if the introduced pin corresponds to the card's id. If they match, it moves to the next state and it allows the user to select an operation.



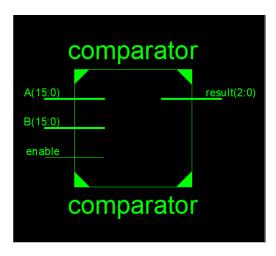
FULL ADDER: adds the introduced sum to the existing one

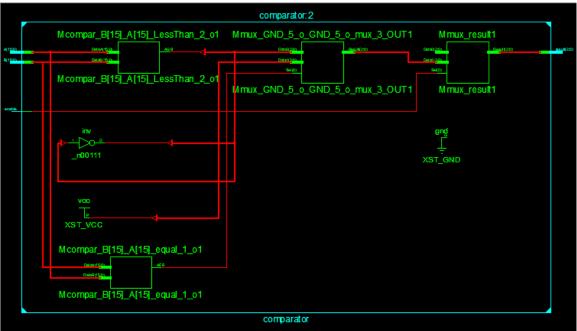


FULL SUBTRACTOR: subtracts the introduced sum from the existing one



COMPARATOR: compares two numbers of 16 bits length



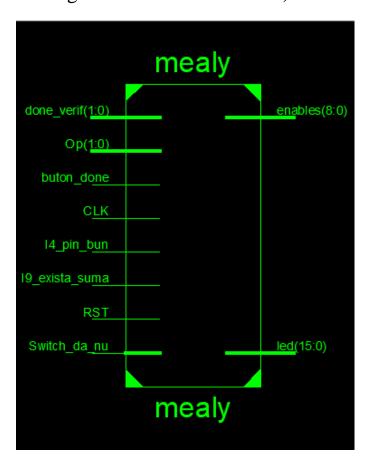


AMOUNT CHECKER: this component verifies if the sum introduced is smaller or equal to the existing sum from ATM and the card's balance memory and 1000 euro. Is composed of 3 comparators.

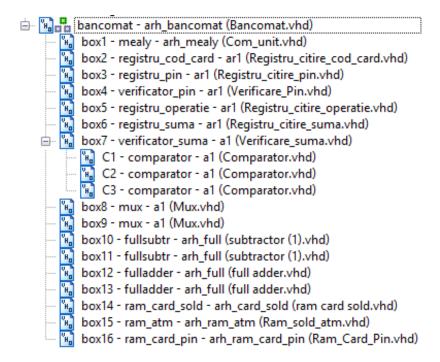
MULTIPLEXER: to select the inputs for the rams (if we perform addition or subtraction)

2.3 COMMAND UNIT

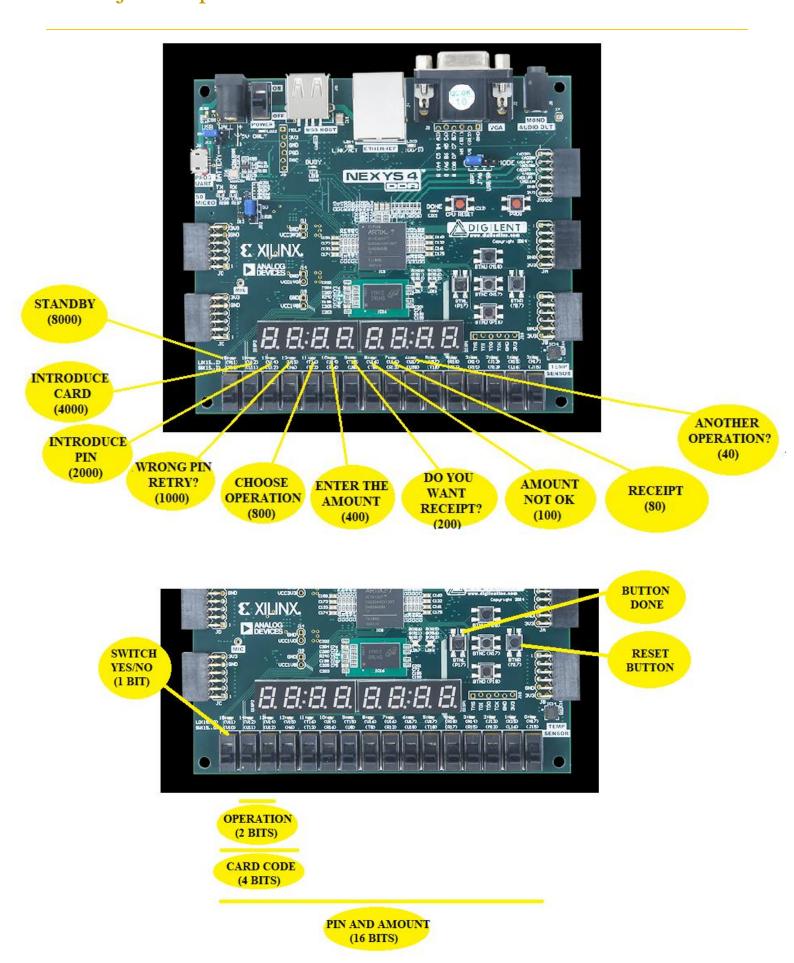
Performs the transitions from a state to another, and computes the enables for the execution unit, based on it's current state and on the value of it's inputs (signals coming from the execution unit).



2.4 ALL TOGETHER



3. Project implementation



3.1 SIMULATIONS

RAM CARD - PIN

Address (cod card) = 0001, write enable = 0 =>Data out = 000000100110000 (130) as in the memory =>ok

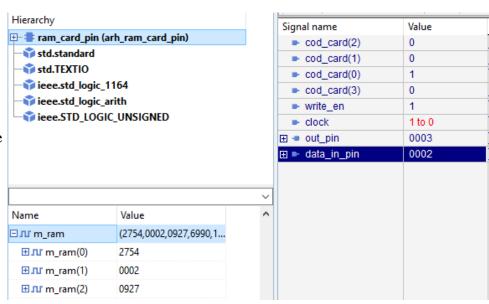
Signal name	Value
cod_card(2)	0
cod_card(1)	0
■ cod_card(0)	1
■ cod_card(3)	0
write_en	0
■ clock	1 to 0
■ out_pin	0130

Write enable = 1, Address = 1

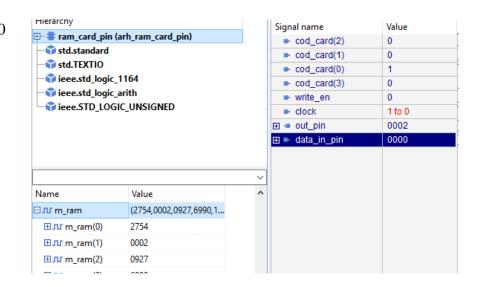
Data in = 2

=>M_ram(1) = 2 => ok

=>Out_pin = 3 (what was before becaure we write, not read)
=>ok



Address = 1, Write en = 0, Data in = 0
=>M_ram(1) = 2 (not changed)
=>Out_pin = 2
=> ok



REGISTER CARD CODE

Reset = 0, Enable = 1, In switch = 3=> Out = 3 => ok

Reset =	1 ->	Out -	0 - >	٥k
Neset –	1 一/	Out –	ひー/	OK

Signal name	Value
► clk	1 to 0
rst	0
■ enable	1
	3
■ Out_pin ■ Out_pin	3

Signal name	Value
► clk	1 to 0
rst rst	1
enable	1
■ In_switchuri ■ In_switchuri	3
■ Out_pin	0

Reset = 0, Enable = 0, In = 0

Out = 3 (previous value, not changed)

=> ok

Signal name	Value
► clk	1 to 0
▶ rst	0
enable	0
■ In_switchuri ■ In_switchuri	0
■ Out_pin	3

PIN CHECKER

A=3, B=7

 \Rightarrow pin bun = 0

=> ok

Signal name	Value
± ► A	0003
⊞ ■ B	0007
■ pin_bun	0

A=3, B=3

=> pin bun = 1

=> ok

Signal name	Value
± ► A	0003
⊞ B	0003
■ pin_bun	1

COMPARATOR

Signal name	Value
⊟ ► A	A
► A[3]	1
▶ A[2]	0
▶ A[1]	1
► A[0]	0
⊟ ▶ B	В
■ B[3]	1
■ B[2]	0
■ B[1]	1
■ B[0]	1
■ enable	1
► clk	1 to 0
⊟ ● result	1
■ result[2]	0
■ result[1]	0
■ result[0]	1

A(amount in cont) = 10, B(amount requested) = 11

r[2] (A>B) = 0

r[1](A=B)=0

r[0] (A < B) = 1

=>ok

AMOUNT CHECKER

Signal name	Value	T
suma_ceruta	FFC0	
■ suma_cont	000F	
■ suma_atm	FFFF	
► clk	0	
■ enable	1	
suma_ok	0	
done_verif	1	

Suma ceruta (amount requested) <

Suma cont (amount in cont),

Enable=1,

=>suma_ok = 0

=>done_verif = 1

=>ok

FULL ADDER

Existing Sum A=1010 (A)

Sum to introduce (B)= 0100 (B)

CIN=1

SUM=A+B+CIN=15(F)

 $SUM=Extended_sum=15(F)$

COUT=0

Signal name	Value
⊕ A	000A
⊕ B	0004
■ CIN	1
⊕ → SUM	000F
COUT	0
⊞ л Extended_sum	0000F
(x) N	16

FULL SUBTRACTOR

Signal name	Value
⊕ ► A	000F
⊕ B	0004
▶ en	1
■ BIN	0
■ DIFF	000B
■ BOUT	0
	0000B
(x) N	16

Existing sum A=1111(F)

Sum to extract B=0100(4)

En=1

BIN=0

DIFF=A-B-BIN=1011(B)

DIFF=Extended_diff=1011(B)

BOUT=0

MAIN SIMULATION

STATE=STANDBY

=>LED = 8000

		Signal name	Value
Vame	Value	switch switch	0000
∃ ► switch	0000	■ but_done	0
but_done	0	▶ reset	0
▶ reset	0	■ clock	1 to 0
■ clock	0	⊕ → led	8000
∃ → led	8000		
. III enables	000		
∃ л г cod_card	U		
∄ Љ pin_citit	UUUU		
∄ лг pin_ram	2754		
лг pin_bun	0		
∃ л ɪ done	0		
∃ 1 operatie	U		
∄ ™ suma_citita	UUUU		
∃ л suma_cont	2754		
∃ . II suma_atm	3E80		
лг suma_ok	U		
∄	0000		
∄ л r dif_in_atm	0000		
∄ Љ sum_in_cont	0000		
∃.π sum_in_atm	0000		
∃	0000		
∃ J J in_atm	0000	Cursor 1	
™ enable_ram	0		

BUTON DONE : 0->1->0

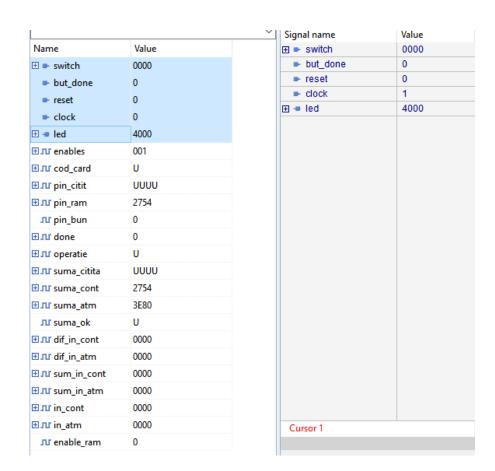
=>STATE = INTRODUCE CARD

CODE

=> LED = 4000

ENABLES=1 (REGISTER CARD

CODE)



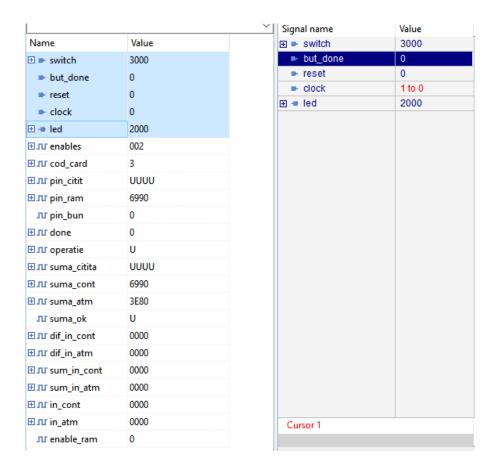
 $CARD\ CODE = 3$

BUTON DONE: 0->1->0

=>STATE = INTRODUCE PIN

=> LED = 2000

=>ENABLES = 2 (REGISTER PIN)



PIN=6990

ENABLES = 4 (CHECK PIN)

PIN BUN = 1

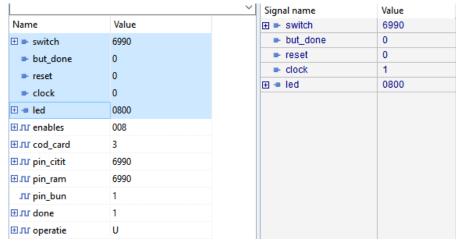
Name	Value
± ► switch	6990
■ but_done	0
▶ reset	0
■ clock	0
± → led	2000
∄ .π enables	004
I	3
	6990
	6990
лг pin_bun	1
∄	1
⊞ Л operatie	U

STATE = CHOOSE OPERATION

⇒ LED = 800

ENABLES = 8 (REGISTER

OPERATION)



OPERATION = 0 (CASH
\WITHDRAWAL)

STATE = ENTER AMOUNT

LED = 400

ENABLES = 10 (REGISTER

AMOUNT)

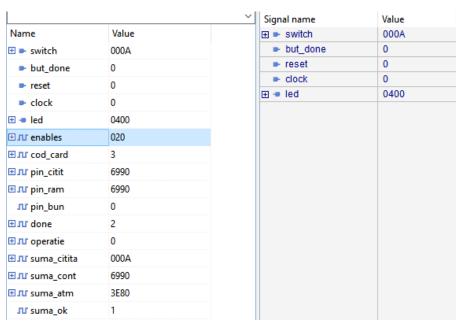
Signal name Value Name Value switch
 switch 0990 0 0990 but_done switch

 switch 0 but_done 1 to 0 clock 0 ■ reset ⊕ • led 0400 clock 0 🕀 🛥 led 0400 **⊞ J** ∎ enables 010 3 **⊞л**г pin_citit 6990 **⊞** лг pin_ram 6990 ☐ pin_bun 0 **⊞** J∐ done 0 **⊞ J** J operatie **⊞ J** suma_citita UUUU

AMOUNT READ = A(10)

ENABLES = 20 (AMOUNT CHECK)

=>AMOUNT_OK = 1



DIF_IN_CONT AND DIF_IN_ATM MODIFIED
IN_CONT, IN_ATM (DECIDED IF WE MODIFY WITH A SUM
OR A DUBTRACTION) MODIFIED
SUMA_CONT, SUMA_ATM NOT YET MODIFIED
ENABLES= 40 (RAMS)

⊞ лг enables	040
⊞ I II cod_card	3
⊞ Л pin_citit	6990
⊞лг pin_ram	6990
™ pin_bun	0
⊞ J I done	2
⊞ π operatie	0
⊞ ЛГ suma_citita	000A
⊞ J suma_cont	6990
⊞ . II suma_atm	3E80
™ suma_ok	1
⊞ лг dif_in_cont	6986
⊞ л dif_in_atm	3E76
⊞ J sum_in_cont	0000
⊞ J sum_in_atm	0000
⊞ I II in_cont	6986
⊞ л г in_atm	3E76
лг enable_ram	1

RAMS MODIFIED

ENABLES RAM = 0

STATE = WANT RECEIPT?

LED = 200

- ⇒ SAY NO => STATE =
 ANOTHER OPERATION?
- ⇒ SAY YES => STATE = CHOOSE OPERATION

Name	Value
switch switch	000A
but_done	0
▶ reset	0
▶ clock	0
⊕ → led	0200
⊞ 	000
⊞ J J Cod_card	3
⊞	6990
⊞	6990
.π pin_bun	0
⊞	0
⊞ лг operatie	0
⊞ .⊓ suma_citita	000A
⊞ JJI suma_cont	6986
⊞ J II suma_atm	3E76
Л suma_ok	1
⊞	6986
⊞	3E76
⊞ J sum_in_cont	0000
⊞	0000
⊞ J in_cont	6986
⊞.лг in_atm	3E76
☐ enable_ram	0

	000A
but_done	0
▶ reset	0
■ clock	1 to 0
⊕ → led	0200
Cursor 1	

RESET = 1

- ⇒ ALL CLEARED EXCEPT FOR RAMS
- \Rightarrow STATE = STANDBY

■ switch 000E but_done 0 reset 1 clock 0 led 8000 lnr enables 000 lnr enables 000 lnr pin_citit 0000 lnr pin_titit 0000 lnr operatie 0 lnr suma_citita 0000 lnr suma_cont 2754 lnr suma_titia 0000 lnr suma_ok 1 lnr dif_in_cont 6986 lnr sum_in_cont 0000 lnr sum_in_cont 0000 lnr sum_in_atm 0000 lnr sum_in_atm 0000 lnr sum_in_cont 6986 lnr sum_in_cont 6986			Signal name	Value
■ but_done 0 ■ reset 1 ■ clock 0 ■ led 8000 ■ If enables 000 ■ If cod_card 0 ■ If pin_citit 0000 ■ If pin_ram 2754 ■ If pin_bun 0 ■ If one 0 ■ If suma_citita 0000 ■ If suma_citita 0000 ■ If suma_cont 2754 ■ If suma_cont 2754 ■ If in_cont 6986 ■ If dif_in_atm 3E76 ■ If sum_in_cont 0000 ■ If sum_in_atm 0000	Name	Value	switch switch	000E
■ reset 1 ■ clock 0 ■ led 8000 ■ In enables 000 ■ In cod_card 0 ■ In pin_ram 2754 In pin_bun 0 ■ In operatie 0 ■ In suma_citita 0000 ■ In suma_cont 2754 ■ In suma_cont 2754 ■ In suma_cont 6986 ■ In dif_in_cont 6986 ■ In sum_in_cont 0000 ■ In sum_in_cont 0000 ■ In sum_in_cont 0000 ■ In sum_in_cont 0000 ■ In sum_in_cont 6986 ■ In sum_in_atm 0000	⊕ ► switch	000E		
■ clock 0 ■ clock 0 ■ reables 000 ■ reables 000 ■ reables 0000 ■ reables 00000 ■ r	but_done	0		
■ clock 0 ■ reables 8000 □ reables 000 □ repin_cod_card 0 □ repin_citit 0000 □ repin_tram 2754 repin_bun 0 □ reportie 0	▶ reset	1		
### Pur enables	■ clock	0	± → led	8000
Enr cod_card 0 Enr pin_citit 0000 Enr pin_ram 2754 In pin_bun 0 Enr done 0 Enr operatie 0 Enr suma_citita 0000 Enr suma_cont 2754 Enr suma_titia 3E76 In suma_ok 1 Enr dif_in_cont 6986 Enr dif_in_atm 3E76 Enr sum_in_cont 0000 Enr sum_in_cont 0000 Enr sum_in_atm 0000 Enr in_cont 6986 Enr in_cont 6986 Enr in_atm 3E76 Cursor 1	⊞ → led	8000		
### Dr pin_citit	⊞ лг enables	000		
######################################	⊞ π cod_card	0		
### pin_bun 0 #### pin_bun 0 #### done 0 #### noperatie 0 #### suma_citita 0000 #### suma_cont 2754 #### suma_atm 3E76 #### suma_ok 1 #### dif_in_cont 6986 #### sum_in_cont 0000 #### sum_in_atm 0000 #### sum_in_atm 0000 #### sum_in_atm 0000 ##### sum_in_atm 0000 ##### sum_in_atm 0000 ################################	⊞	0000		
#	⊞ лг pin_ram	2754		
■	л pin_bun	0		
### Suma_citita	⊞ Л I done	0		
# III suma_cont 2754 # III suma_atm 3E76 III suma_ok 1 # III dif_in_cont 6986 # III dif_in_atm 3E76 # III sum_in_cont 0000 # III sum_in_atm 0000 # III in_cont 6986 # III in_atm 3E76 Cursor 1	⊞ 1 operatie	0		
### 3E76 ### suma_atm 3E76 ### suma_ok 1 #### dif_in_cont 6986 ### dif_in_atm 3E76 ### sum_in_cont 0000 #### sum_in_atm 0000 #### in_cont 6986 #### in_atm 3E76 Cursor 1	⊞ лг suma_citita	0000		
#####################################	⊞ J I suma_cont	2754		
⊕	⊞ Л Г suma_atm	3E76		
### Ju dif_in_atm 3E76 ### Ju sum_in_cont 0000 ### Ju sum_in_atm 0000 #### Ju in_cont 6986 #### Ju in_atm 3E76 Cursor 1	™ suma_ok	1		
⊕	⊞ л dif_in_cont	6986		
⊕	⊞ л dif_in_atm	3E76		
∄	⊞	0000		
⊕лг in_atm 3E76 Cursor 1	⊞ Л sum_in_atm	0000		
- Culsor 1	⊞	6986		
лг enable_ram 0	⊕ лг in_atm	3E76	Cursor 1	
		0		

4. Final conclusions

Why did we implement the project this way?

We adopted a structured and well-organized implementation style: we gave the variables and signals representative names, and in order to highlight each state of the ATM, we chose to display representative messages together. Thus, it will be easier for any user to perform the desired operation.

Why did we design this way?

We tried to have a design mode as optimal as possible. In solving the project, multiple modules are used, which are linked in a main module. We chose to use the modules both for structuring the program and for a good understanding of the instructions used to solve the problem. This structured and organized way is favorable to both the designer and the user.

How did we create the block diagram?

The block diagram is chosen so as to be as representative, simple and legible as possible. Thus, we highlighted every possible state of the ATM. This type of representation is explicit and easy to understand for each user.

Project development possibilities

To develop the project, several operations could be added.

A new operation could be the transfer of a sum of money from one account to another.

Another operation could be the extraction of the amount in different banknotes of 5,10,20,100 euros, as well as in coins.

It would be an interesting idea to have the possibility to pay taxes or to buy something directly with the money from the credit card.