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

-2020-

CONTENTS

1. Project specification
2. Project description
3. 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:

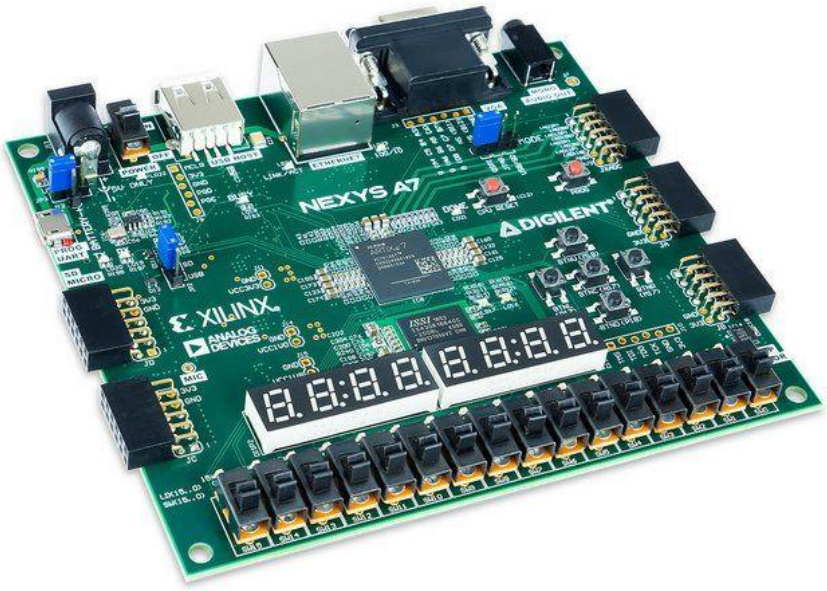
- cash withdrawal
- cash entry
- display balance
- change pin

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

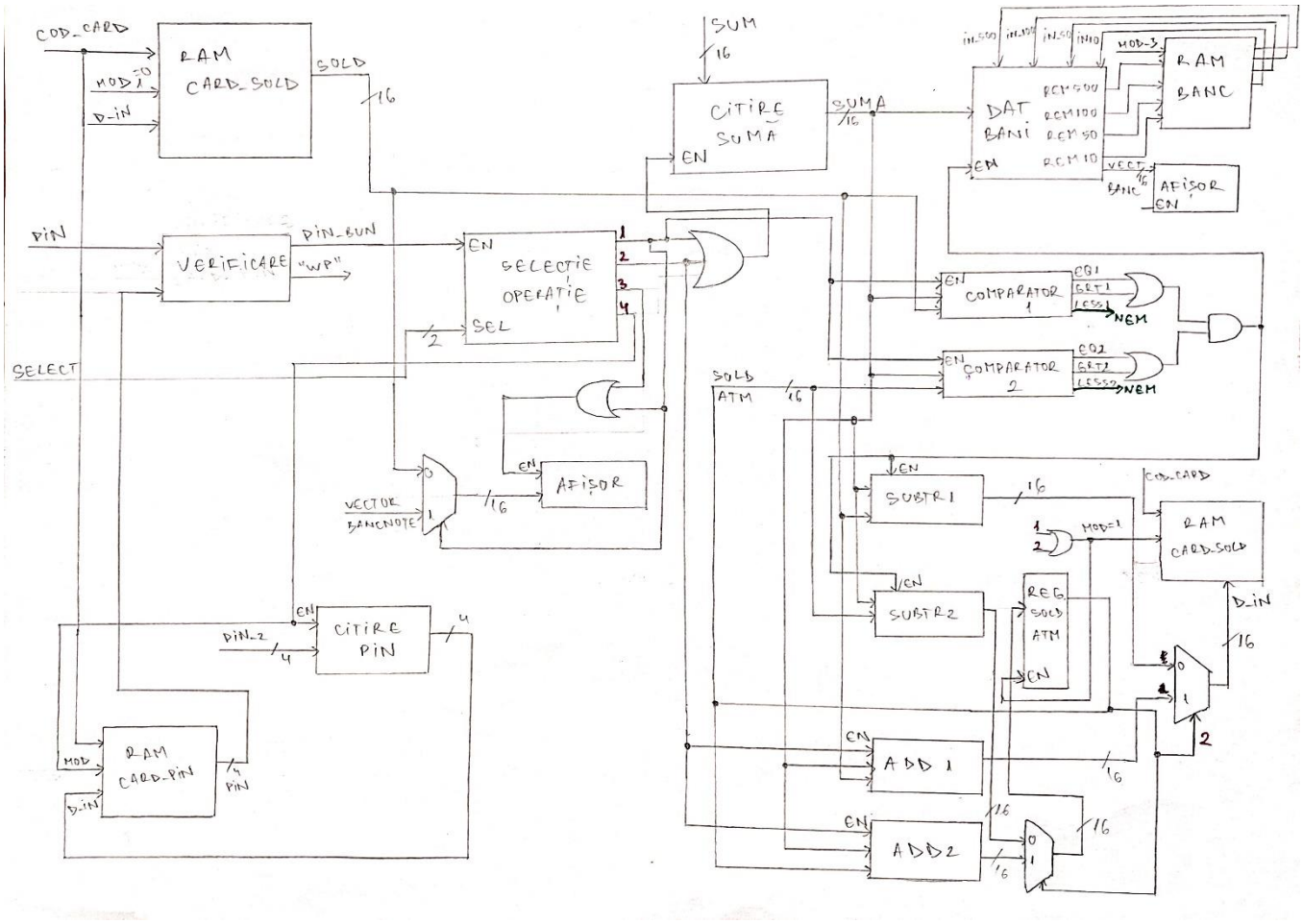
The third operation is to display the balance, it aims to display the amount present in an account.

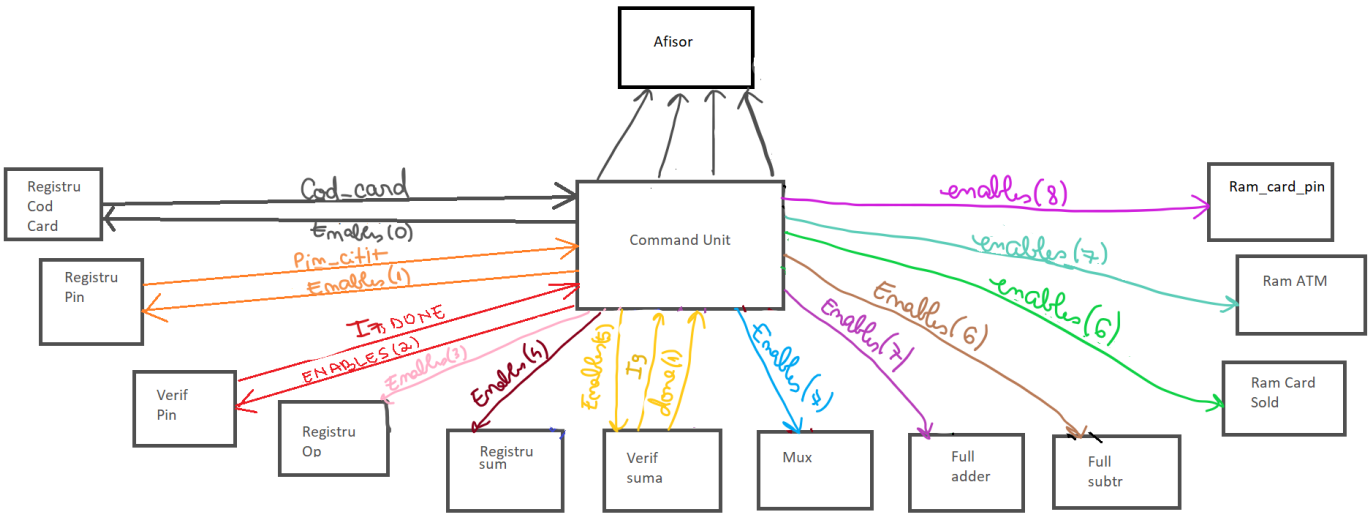
The fourth operation is to change the pin of an account: change the pin in the memory and replace it with a new one.



2. Project description

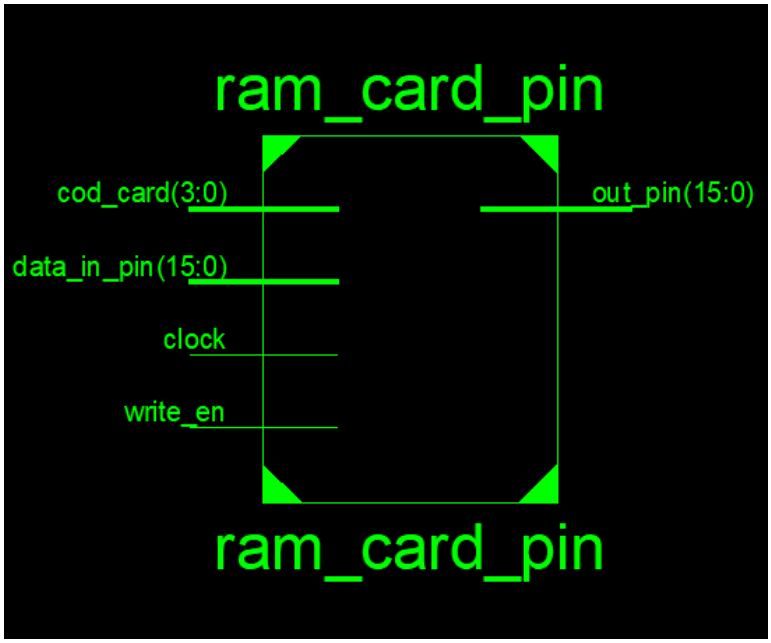
2.1 BLOCK DIAGRAM

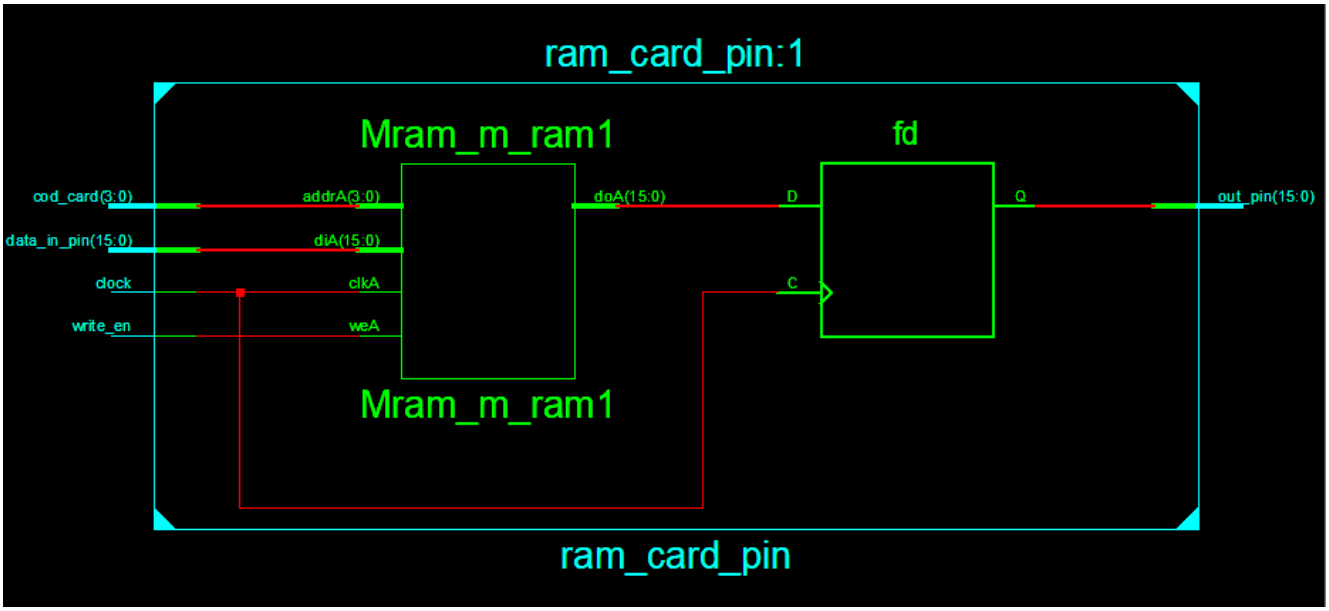




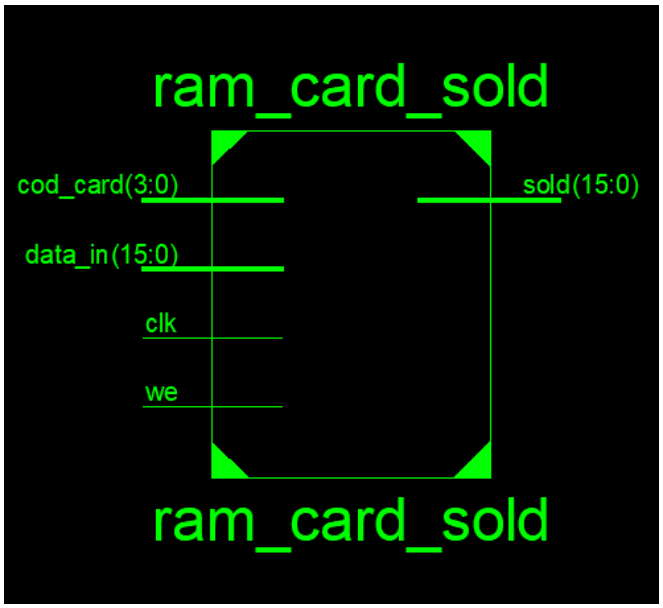
2.2 EXECUTION UNIT - COMPONENTS

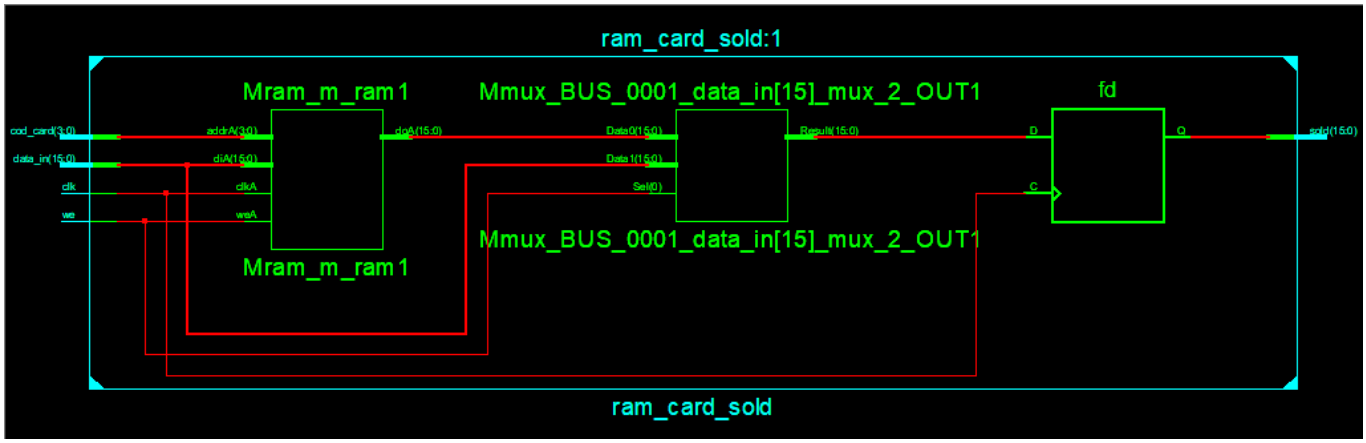
CARD PIN RAM: this RAM contains the description of every card, the id code which corresponds to a specific pin. In our case, we have 15 such cards and the possibility to introduce more.



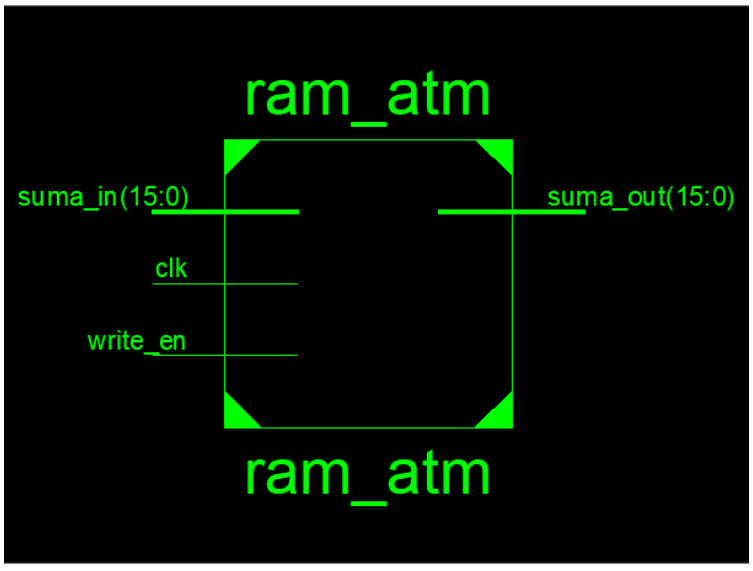


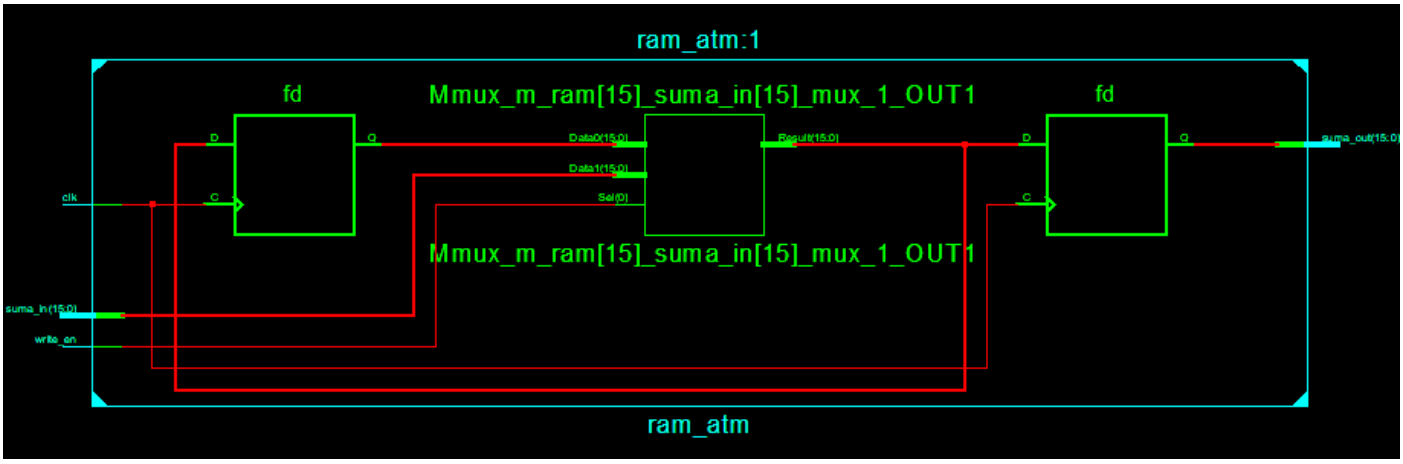
BALANCE CARD RAM: contains the amount of money corresponding to every card.



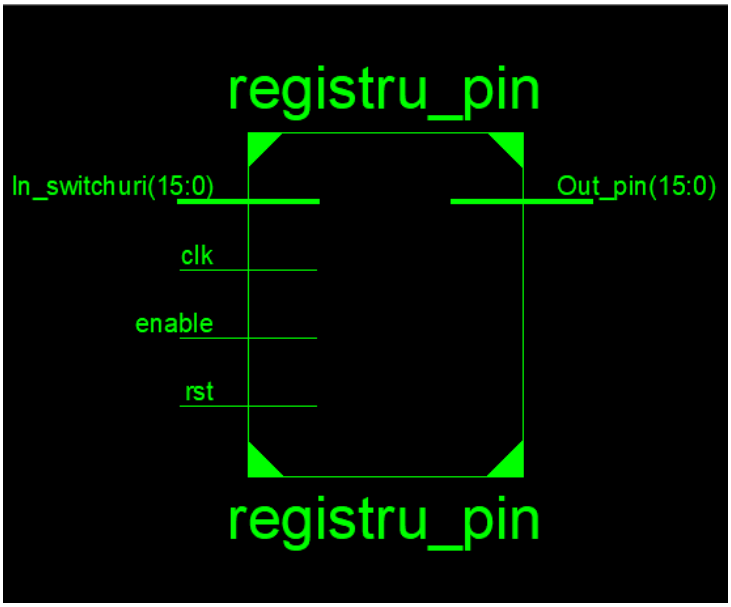


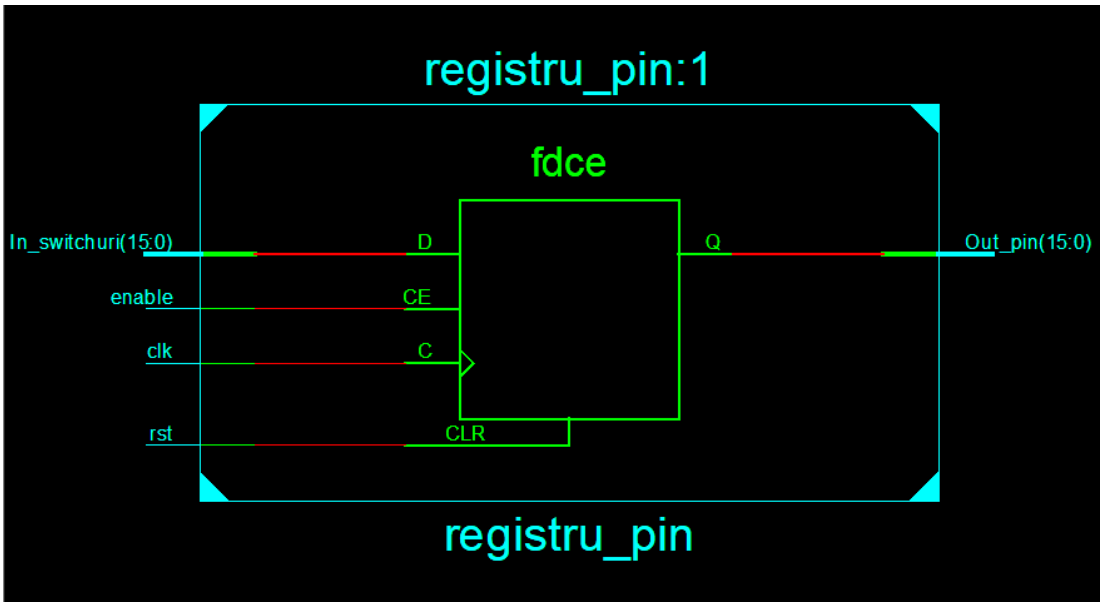
RAM FOR THE EXISTING AMOUNT OF MONEY: this memory stores the available sum that the ATM has.



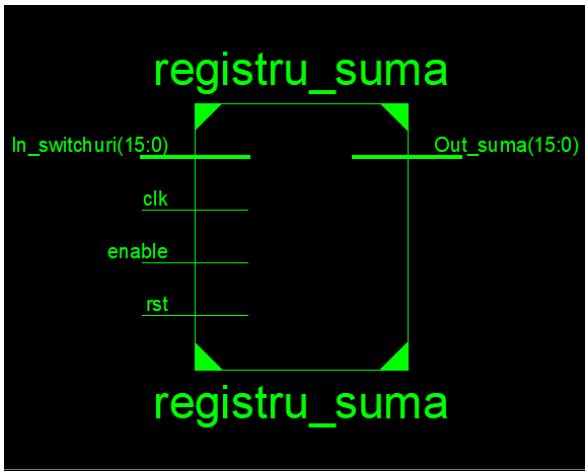


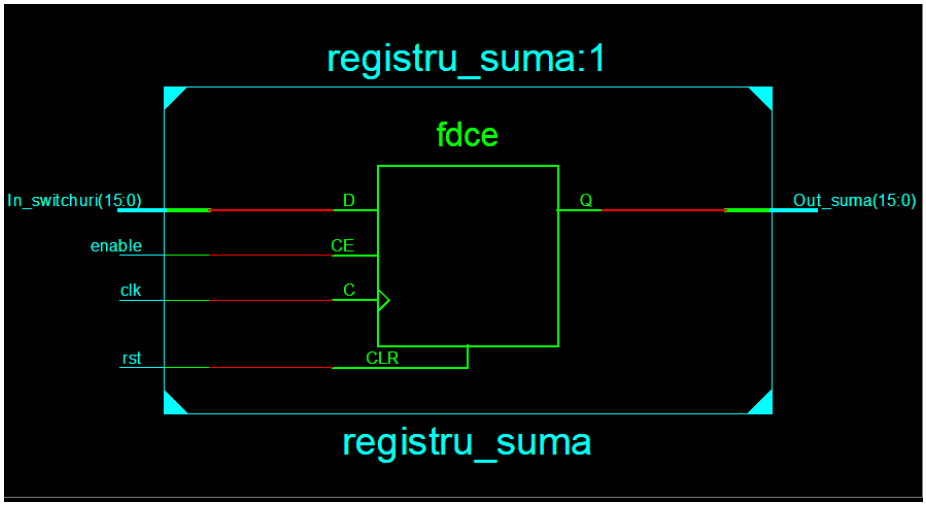
READING PIN REGISTER:



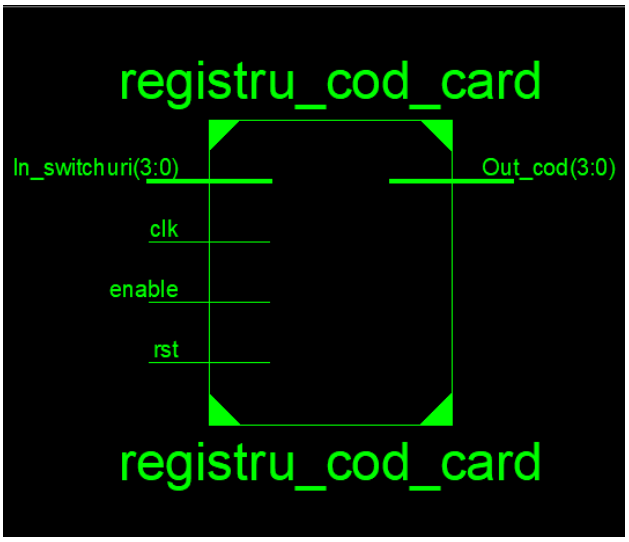


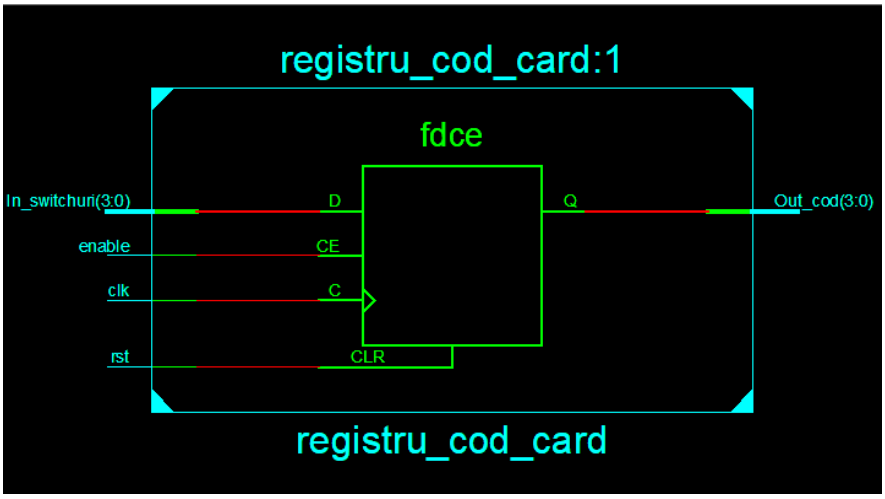
REGISTER FOR READING THE SELECTED SUM: if the user selects cash withdrawal or cash entry option, he is required to introduce the sum of money to add or extract from the account. This component reads the introduced sum of money.



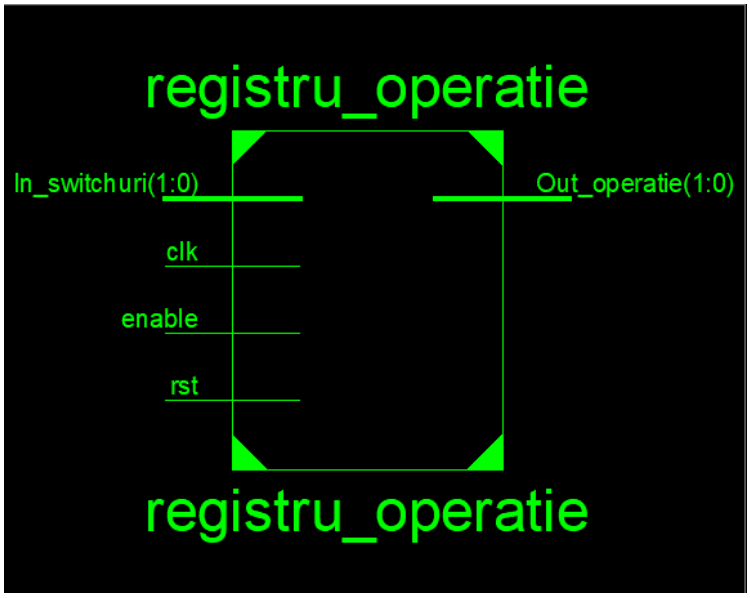


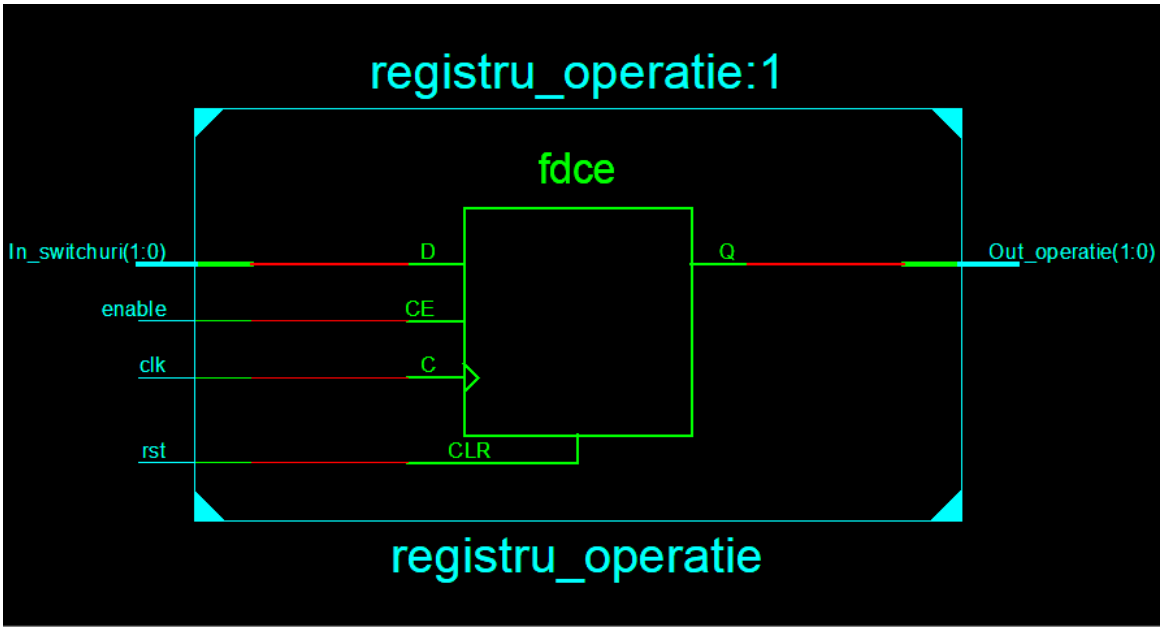
REGISTER FOR READING THE CARD CODE:



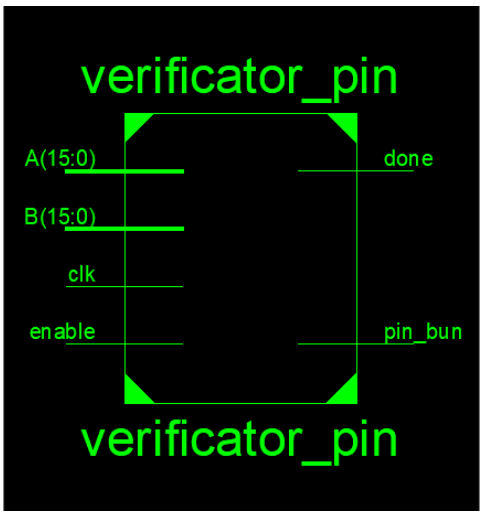


REGISTER FOR READING THE OPERATION:

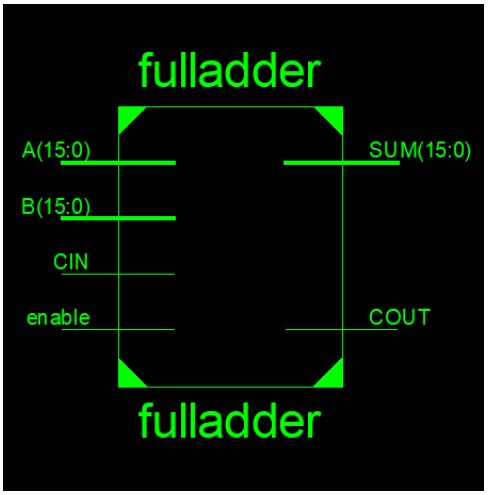




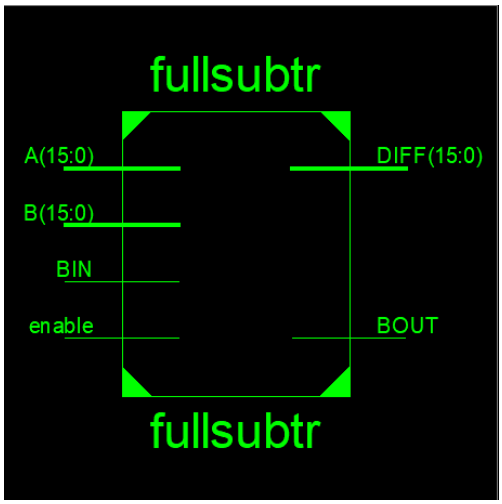
SELECTED PIN VALIDATION: it verifies if the introduced pin corresponds to the card's id. If they match,it moves to the next state and it allows the user to select an operation.



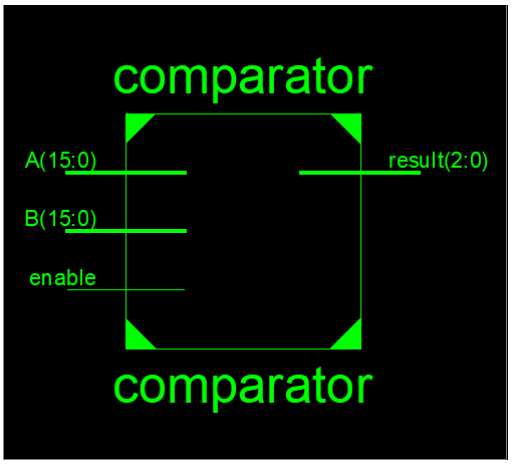
FULL ADDER: adds the introduced sum to the existing one

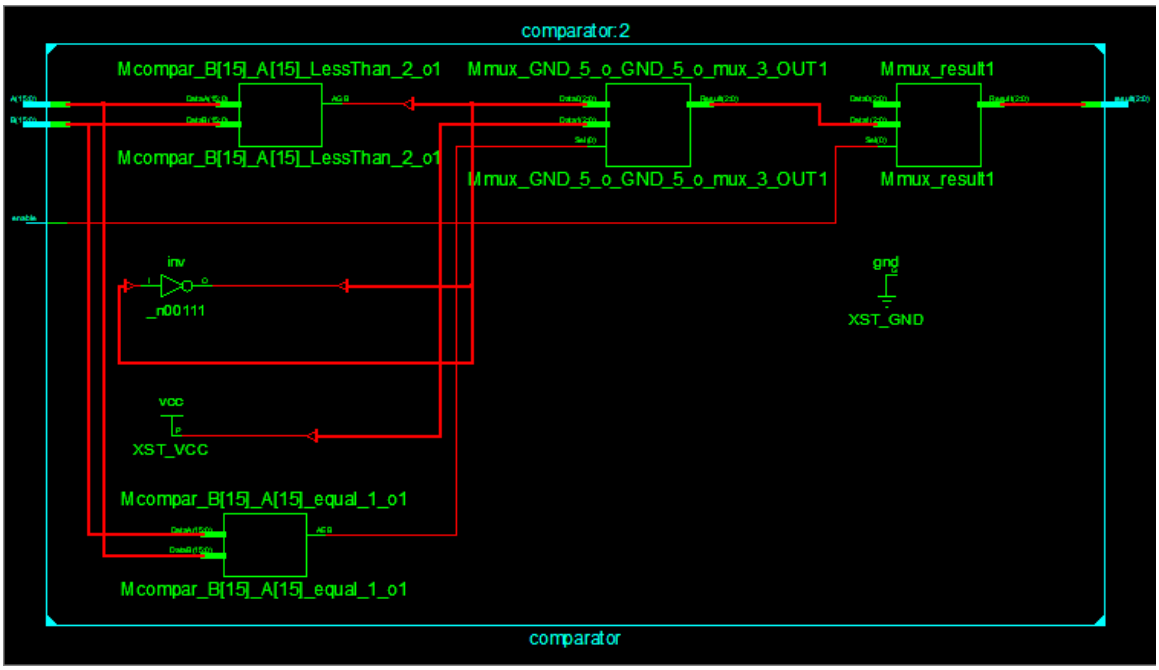


FULL SUBTRACTOR: subtracts the introduced sum from the existing one



COMPARATOR: compares two numbers of 16 bits length



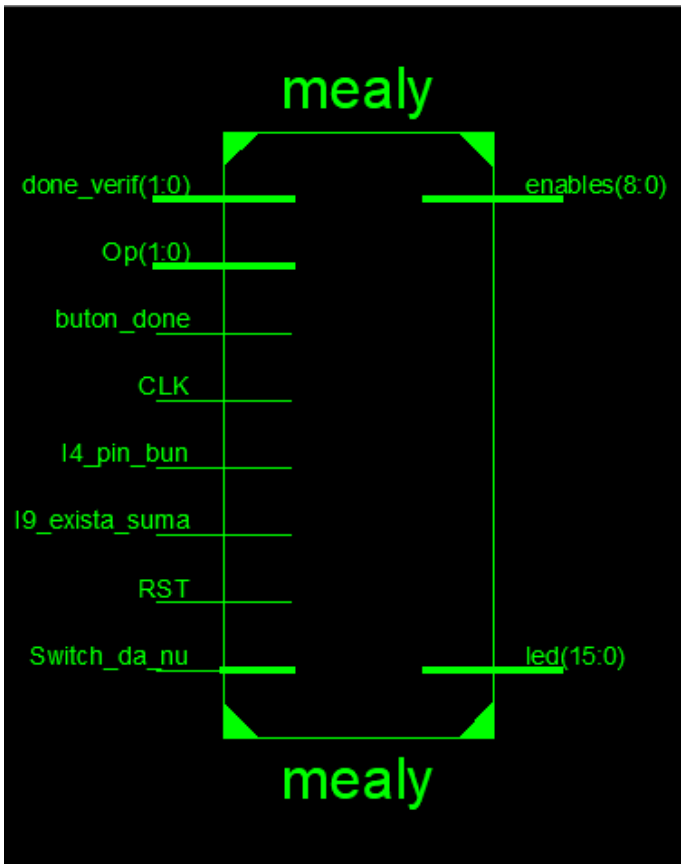


AMOUNT CHECKER: this component verifies if the sum introduced is smaller or equal to the existing sum from ATM and the card's balance memory and 1000 euro. Is composed of 3 comparators.

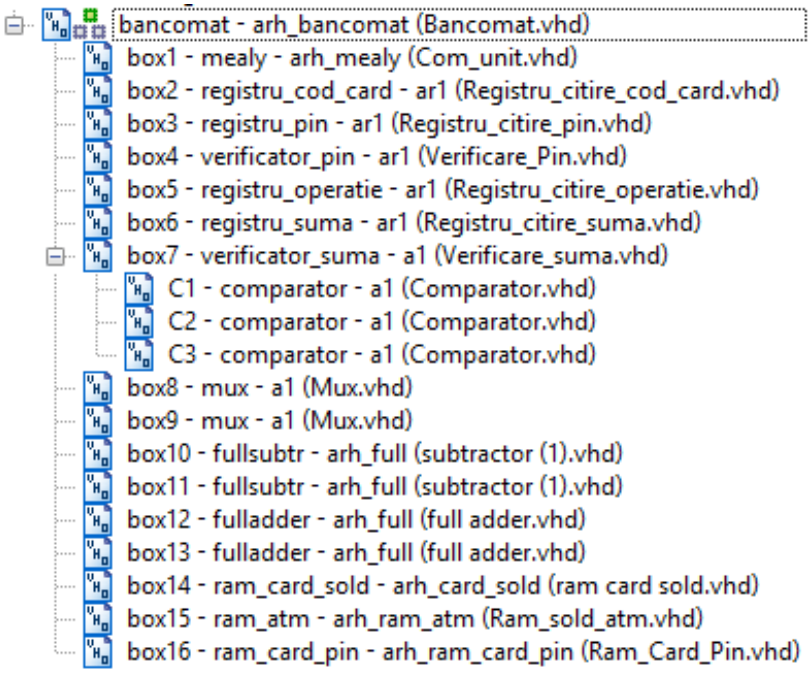
MULTIPLEXER: to select the inputs for the rams (if we perform addition or subtraction)

2.3 COMMAND UNIT

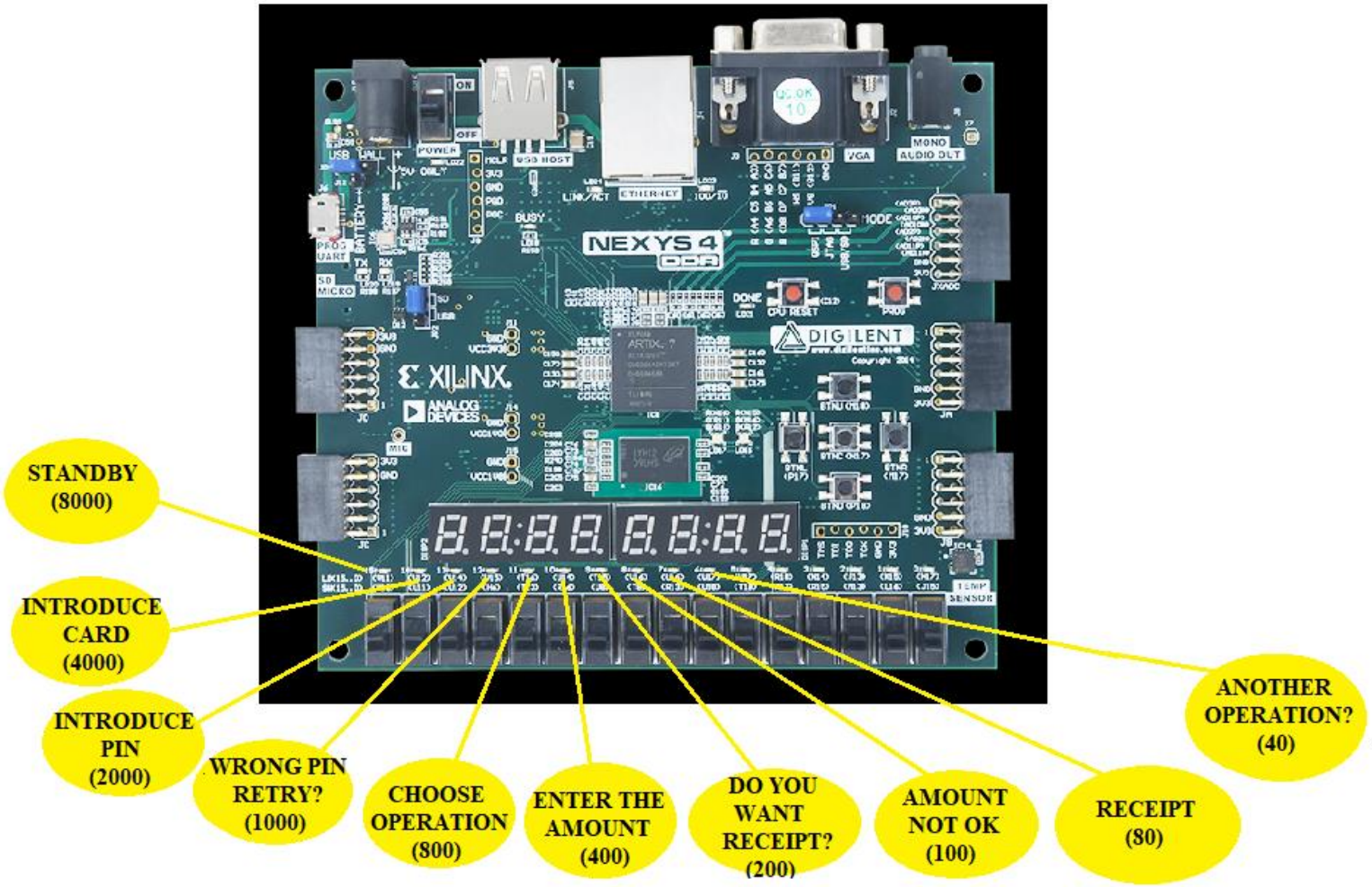
Performs the transitions from a state to another, and computes the enables for the execution unit, based on it's current state and on the value of it's inputs (signals coming from the execution unit).

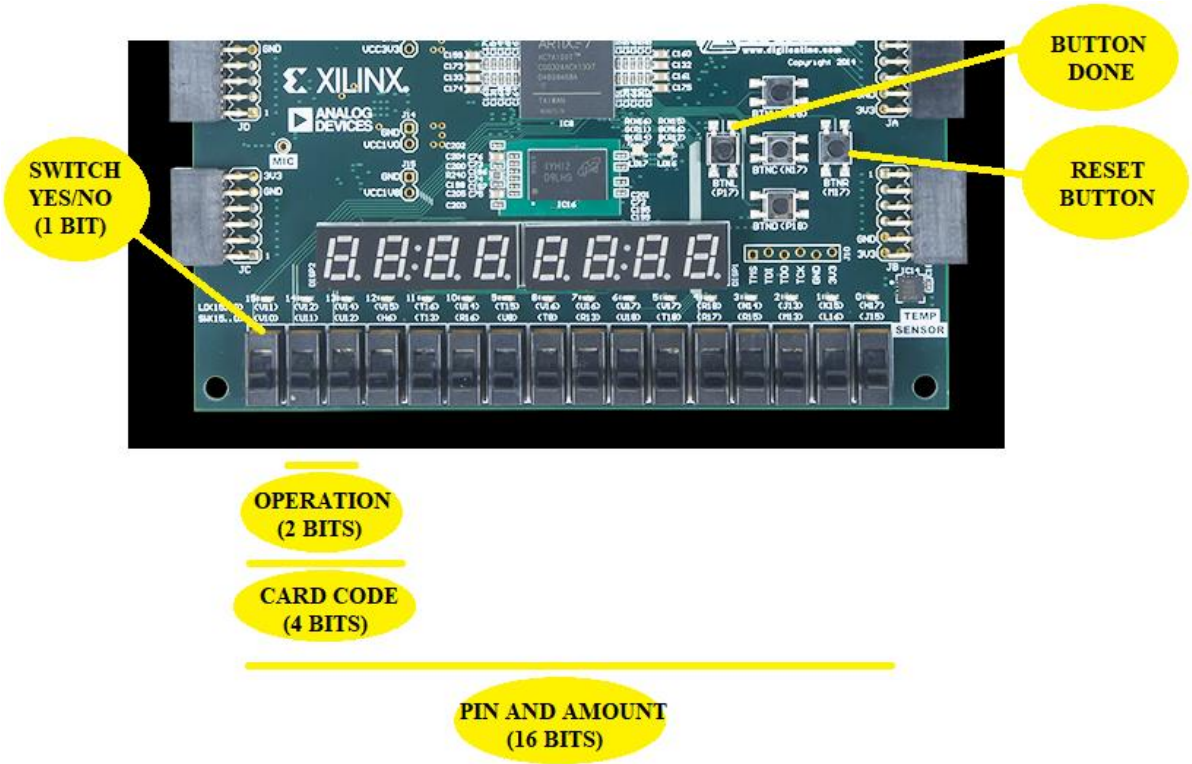


2.4 ALL TOGETHER



3. Project implementation





RAM CARD – PIN

Address (cod card) = 0001, write enable = 0

=>Data out = 0000000100110000 (130) as in the memory

=>ok

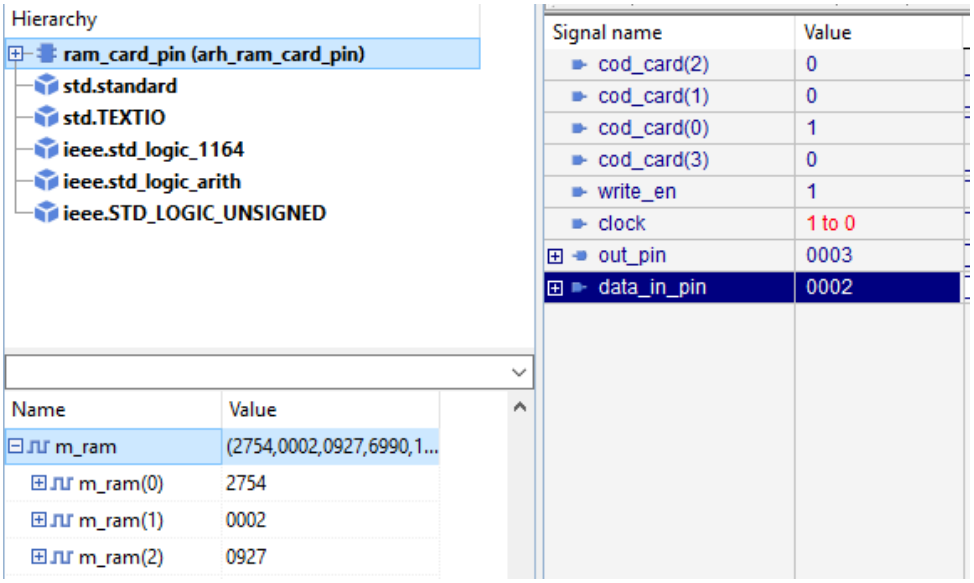
Write enable = 1, Address = 1

Data in = 2

=>M_ram(1) = 2 => ok

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

=>ok

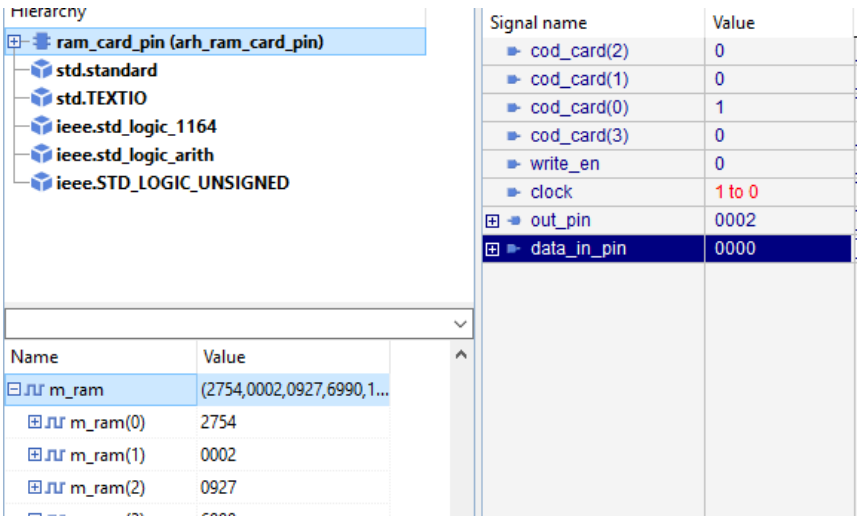


Address = 1, Write en = 0, Data in = 0

=>M_ram(1) = 2 (not changed)

=>Out_pin = 2

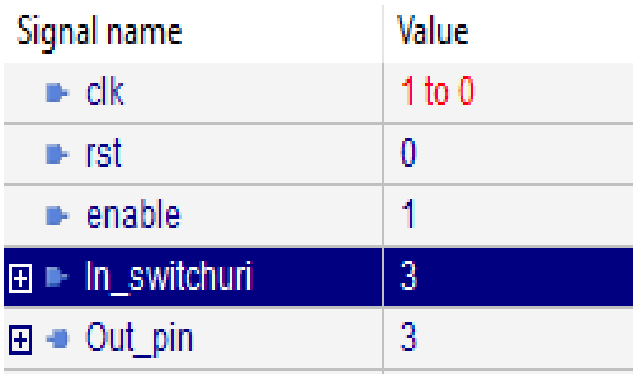
=> ok



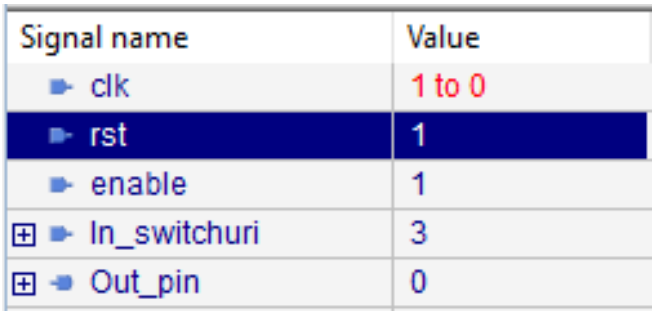
REGISTER CARD CODE

Reset = 0, Enable = 1, In switch = 3

=> Out = 3 => ok



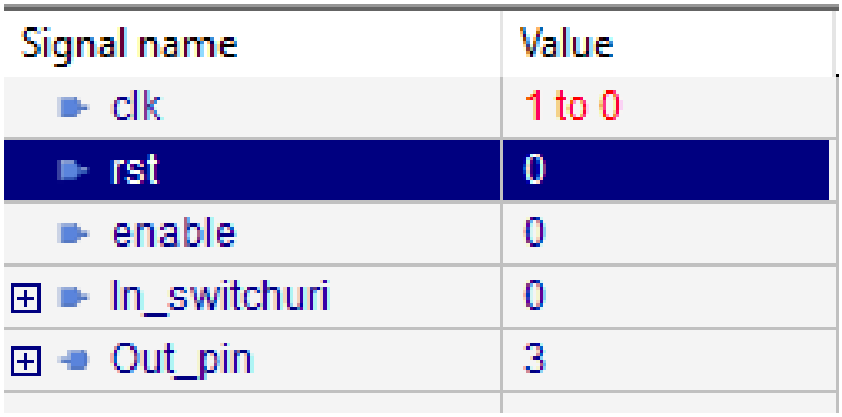
Reset = 1 => Out = 0 => ok



Reset = 0, Enable = 0, In = 0

Out = 3 (previous value, not changed)

=> ok

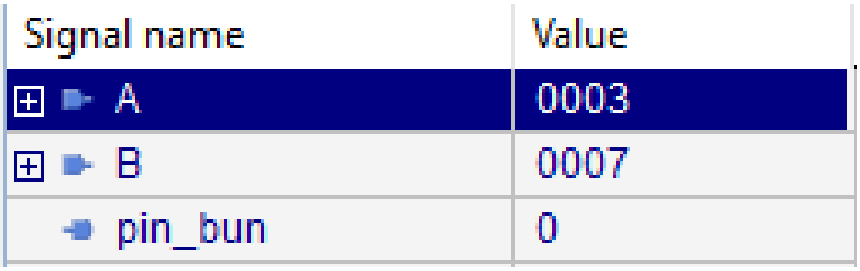


PIN CHECKER

A=3, B=7

=> pin bun = 0

=> ok



A=3, B=3

=> pin bun = 1

=> ok

Signal name	Value
A	0003
B	0003
pin_bun	1

COMPARATOR

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
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
Extended_sum	0000F
N	16

FULL SUBTRACTOR

Signal name	Value
A	000F
B	0004
en	1
BIN	0
DIFF	000B
BOUT	0
Extended_diff	0000B
N	16

Existing sum A=1111(F)

Sum to extract B=0100(4)

En=1

BIN=0

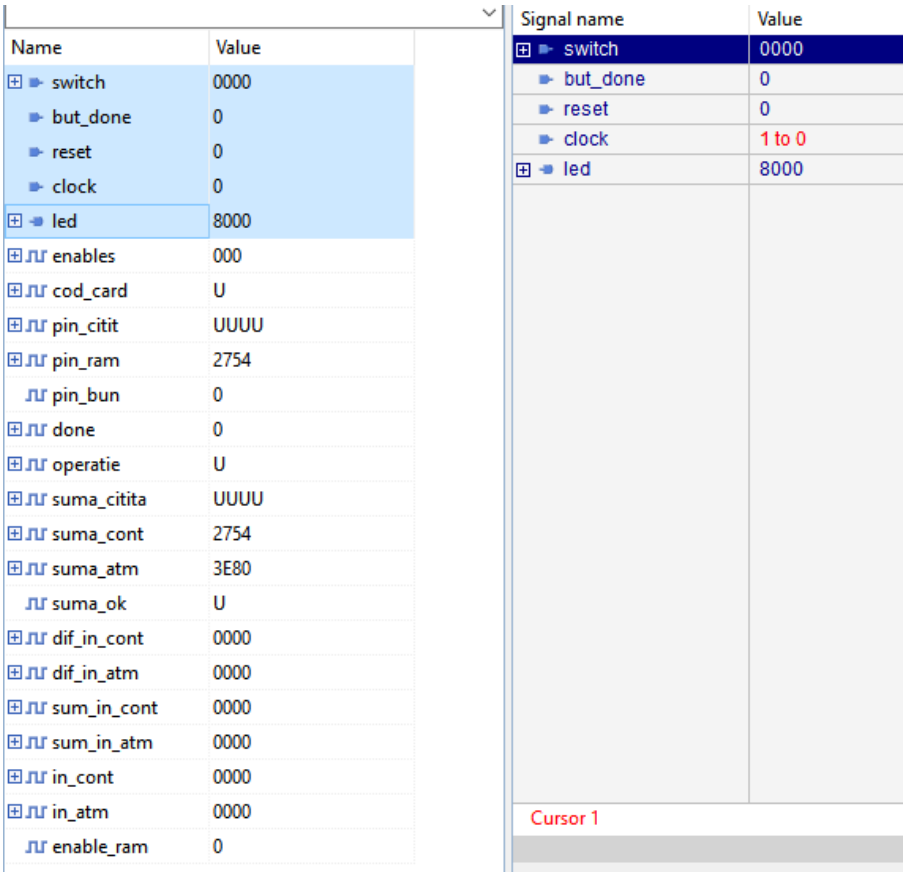
DIFF=A-B-BIN=1011(B)

DIFF=Extended_diff=1011(B)

BOUT=0

STATE=STANDBY

=>LED = 8000



Name	Value
switch	0000
but_done	0
reset	0
clock	0
led	8000
enables	000
cod_card	U
pin_citit	UUUU
pin_ram	2754
pin_bun	0
done	0
operatie	U
suma_citita	UUUU
suma_cont	2754
suma_atm	3E80
suma_ok	U
dif_in_cont	0000
dif_in_atm	0000
sum_in_cont	0000
sum_in_atm	0000
in_cont	0000
in_atm	0000
enable_ram	0

Signal name	Value
switch	0000
but_done	0
reset	0
clock	1 to 0
led	8000

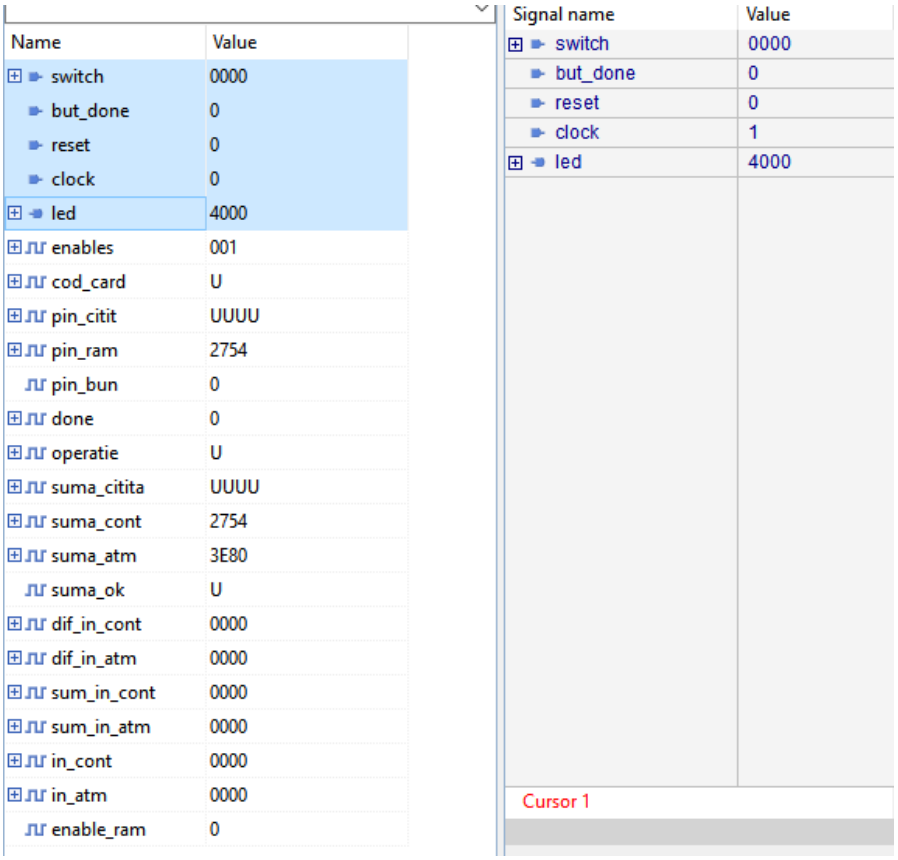
Cursor 1

BUTON DONE : 0->1->0

=>STATE = INTRODUCE CARD CODE

=> LED = 4000

ENABLES=1 (REGISTER CARD CODE)



Name	Value
switch	0000
but_done	0
reset	0
clock	0
led	4000
enables	001
cod_card	U
pin_citit	UUUU
pin_ram	2754
pin_bun	0
done	0
operatie	U
suma_citita	UUUU
suma_cont	2754
suma_atm	3E80
suma_ok	U
dif_in_cont	0000
dif_in_atm	0000
sum_in_cont	0000
sum_in_atm	0000
in_cont	0000
in_atm	0000
enable_ram	0

Signal name	Value
switch	0000
but_done	0
reset	0
clock	1
led	4000

Cursor 1

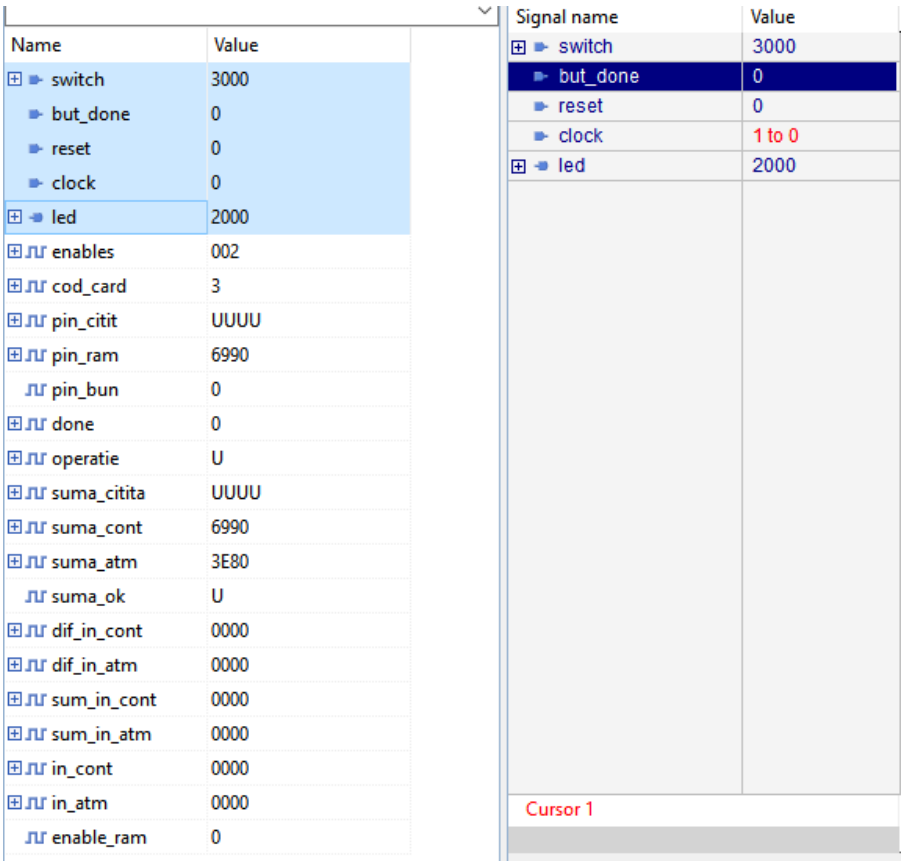
CARD CODE = 3

BUTON DONE: 0->1->0

=>STATE = INTRODUCE PIN

=> LED = 2000

=>ENABLES = 2 (REGISTER PIN)



Name	Value
switch	3000
but_done	0
reset	0
clock	0
led	2000
enables	002
cod_card	3
pin_citit	UUUU
pin_ram	6990
pin_bun	0
done	0
operatie	U
suma_citita	UUUU
suma_cont	6990
suma_atm	3E80
suma_ok	U
dif_in_cont	0000
dif_in_atm	0000
sum_in_cont	0000
sum_in_atm	0000
in_cont	0000
in_atm	0000
enable_ram	0

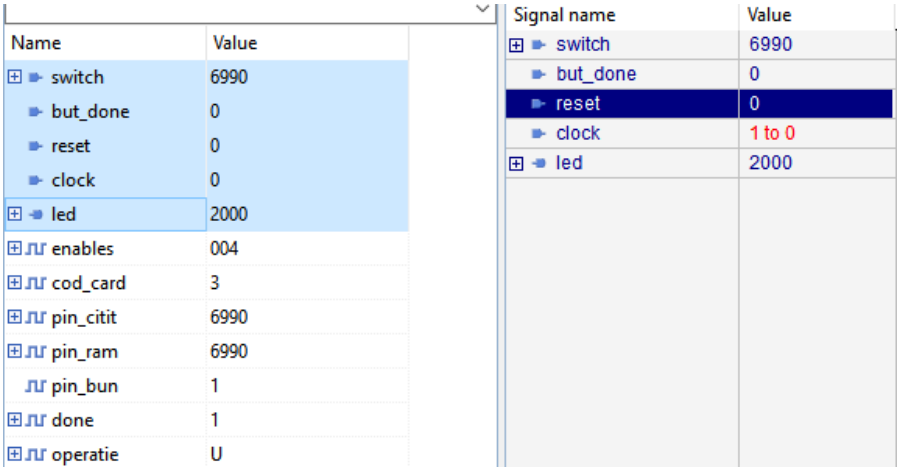
Signal name	Value
switch	3000
but_done	0
reset	0
clock	1 to 0
led	2000

Cursor 1

PIN=6990

ENABLES = 4 (CHECK PIN)

PIN BUN = 1



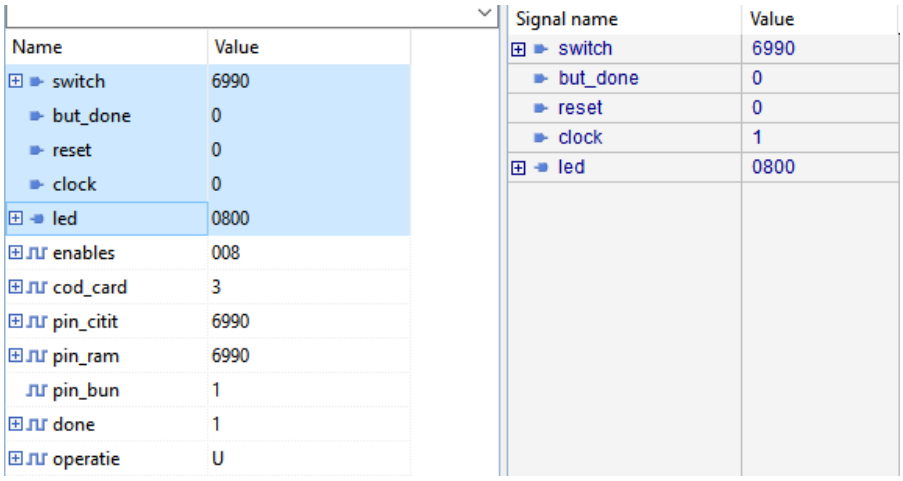
Name	Value
switch	6990
but_done	0
reset	0
clock	0
led	2000
enables	004
cod_card	3
pin_citit	6990
pin_ram	6990
pin_bun	1
done	1
operatie	U

Signal name	Value
switch	6990
but_done	0
reset	0
clock	1 to 0
led	2000

STATE = CHOOSE OPERATION

⇨ LED = 800

ENABLES = 8 (REGISTER OPERATION)



Name	Value
switch	6990
but_done	0
reset	0
clock	0
led	0800
enables	008
cod_card	3
pin_citit	6990
pin_ram	6990
pin_bun	1
done	1
operatie	U

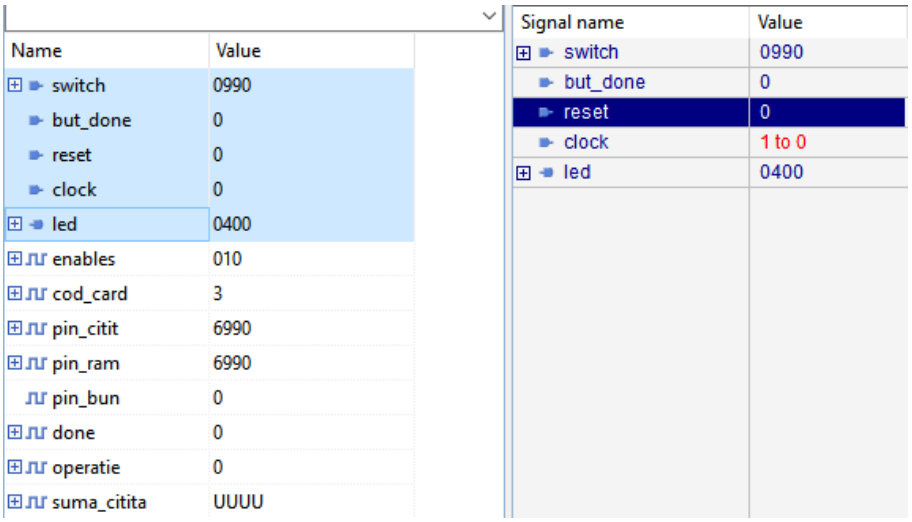
Signal name	Value
switch	6990
but_done	0
reset	0
clock	1
led	0800

OPERATION = 0 (CASH \WITHDRAWAL)

STATE = ENTER AMOUNT

LED = 400

ENABLES = 10 (REGISTER AMOUNT)



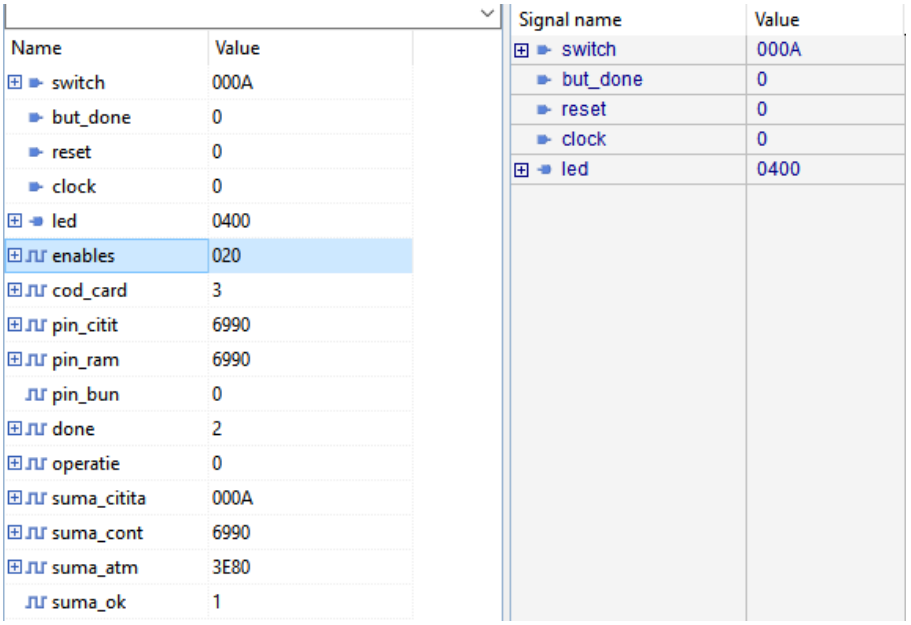
Name	Value
switch	0990
but_done	0
reset	0
clock	0
led	0400
enables	010
cod_card	3
pin_citit	6990
pin_ram	6990
pin_bun	0
done	0
operatie	0
suma_citita	UUUU

Signal name	Value
switch	0990
but_done	0
reset	0
clock	1 to 0
led	0400

AMOUNT READ = A(10)

ENABLES = 20 (AMOUNT CHECK)

=>AMOUNT_OK = 1



Name	Value
switch	000A
but_done	0
reset	0
clock	0
led	0400
enables	020
cod_card	3
pin_citit	6990
pin_ram	6990
pin_bun	0
done	2
operatie	0
suma_citita	000A
suma_cont	6990
suma_atm	3E80
suma_ok	1

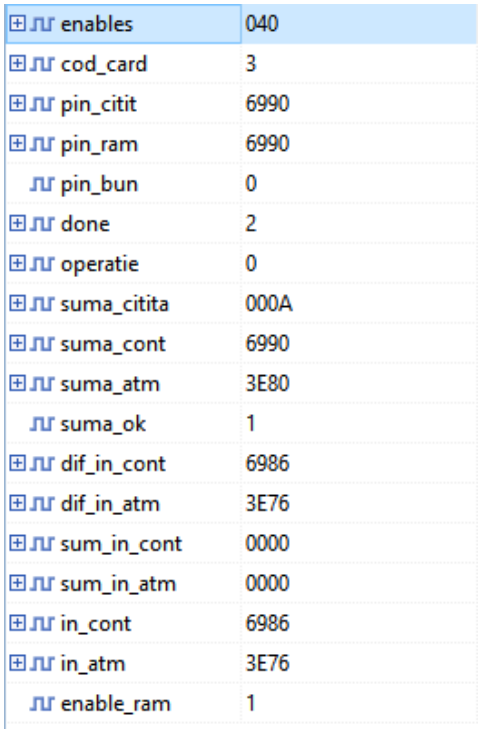
Signal name	Value
switch	000A
but_done	0
reset	0
clock	0
led	0400

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)



enables	040
cod_card	3
pin_citit	6990
pin_ram	6990
pin_bun	0
done	2
operatie	0
suma_citita	000A
suma_cont	6990
suma_atm	3E80
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
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	000A
but_done	0
reset	0
clock	0
led	0200
enables	000
cod_card	3
pin_cititt	6990
pin_ram	6990
pin_bun	0
done	0
operatie	0
suma_cititta	000A
suma_cont	6986
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
enable_ram	0

switch	000A
but_done	0
reset	0
clock	1 to 0
led	0200

Cursor 1

RESET = 1

⇨ ALL CLEARED EXCEPT FOR RAMS

⇨ STATE = STANDBY

Name	Value
switch	000E
but_done	0
reset	1
clock	0
led	8000
enables	000
cod_card	0
pin_cititt	0000
pin_ram	2754
pin_bun	0
done	0
operatie	0
suma_cititta	0000
suma_cont	2754
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
enable_ram	0

Signal name	Value
switch	000E
but_done	0
reset	1
clock	0
led	8000

Cursor 1

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.