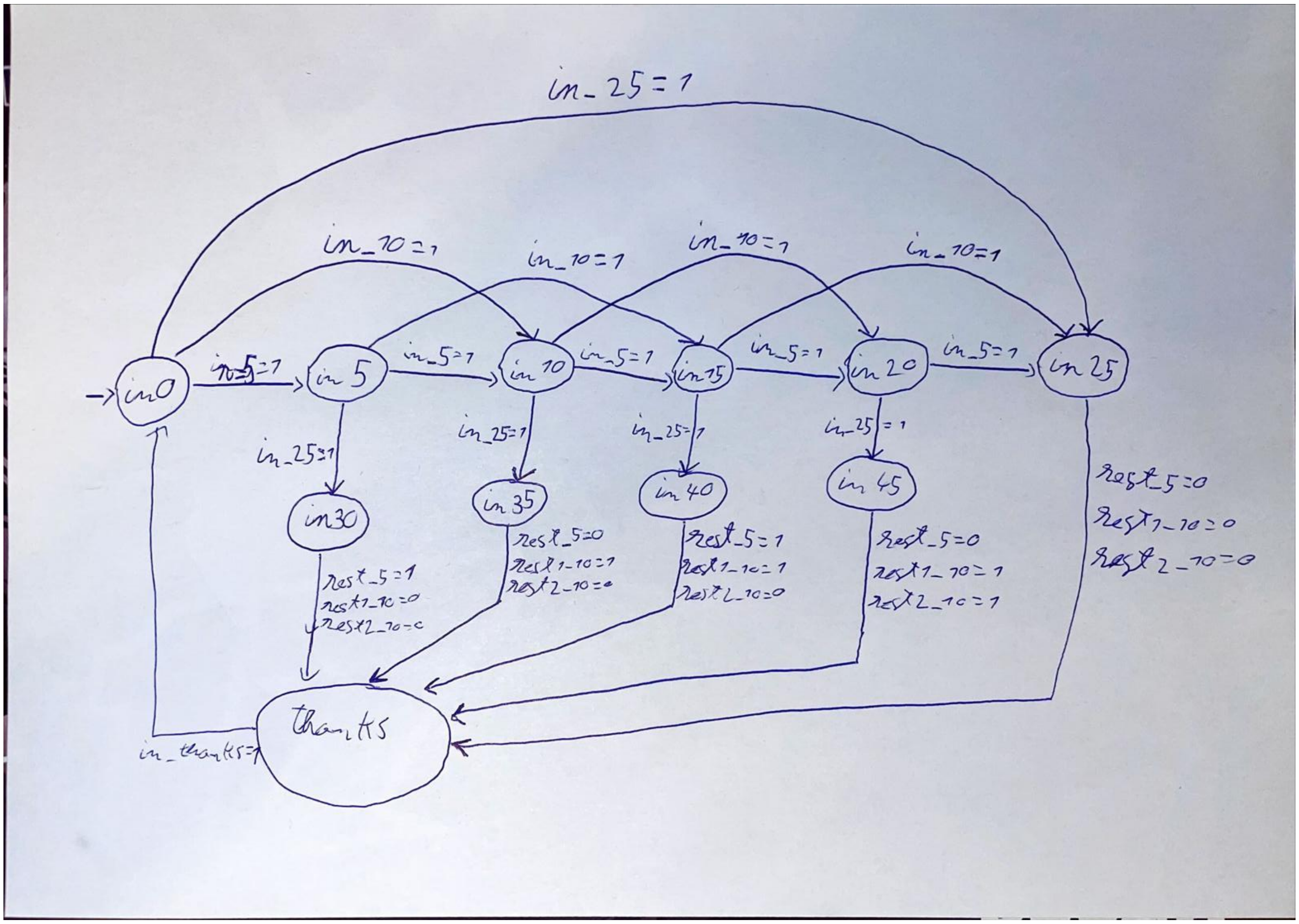
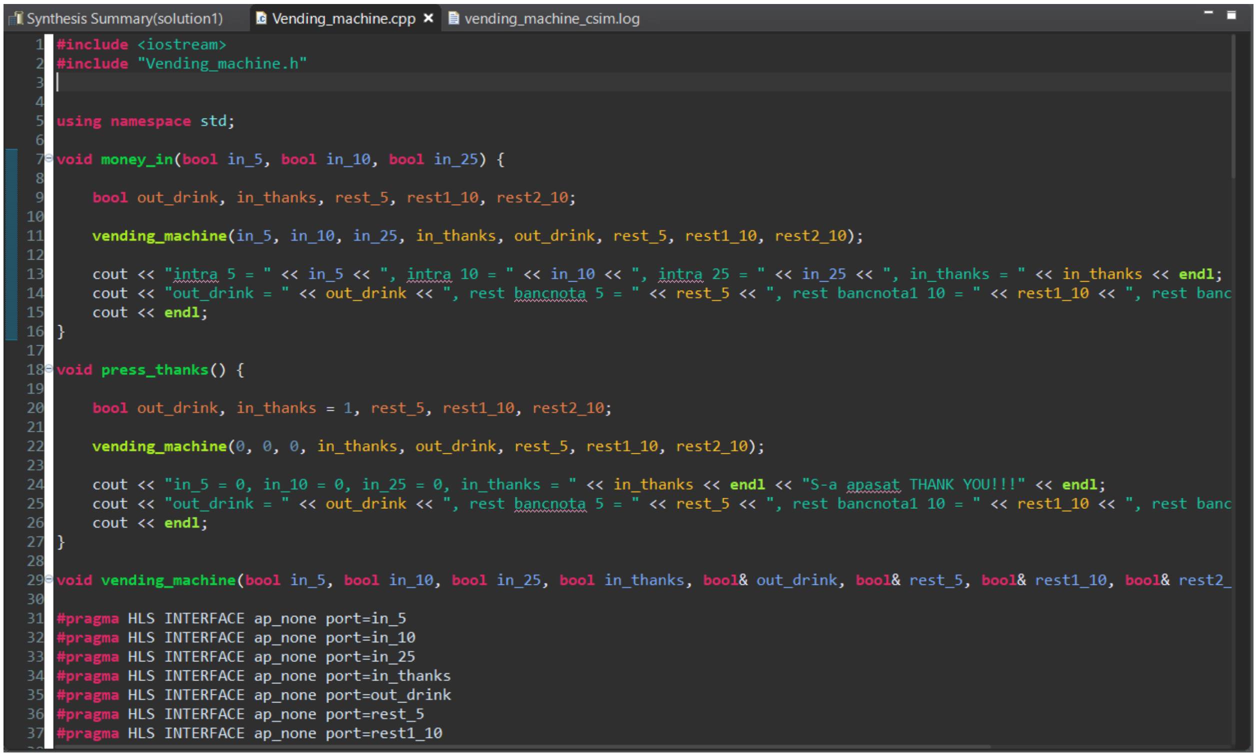
Tema 2 – Mereu Ioan Flaviu 333CC

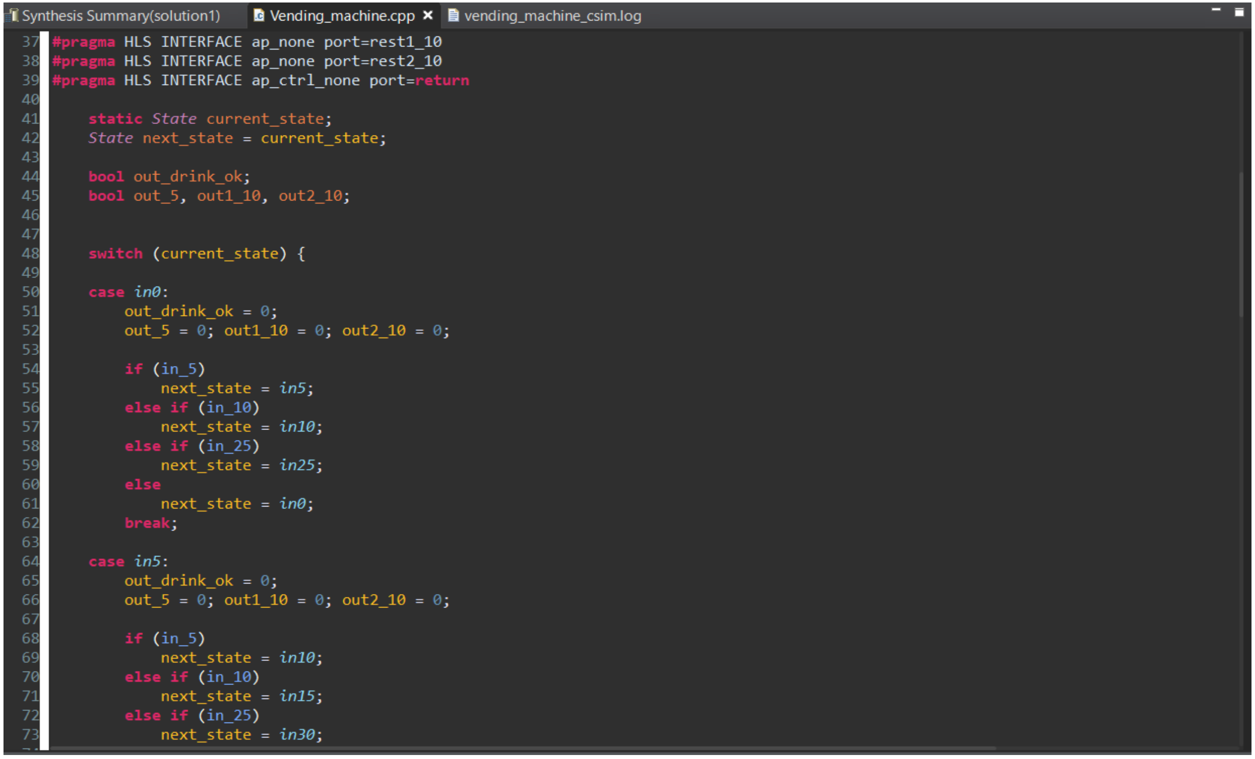
1. Diagrama de stari:



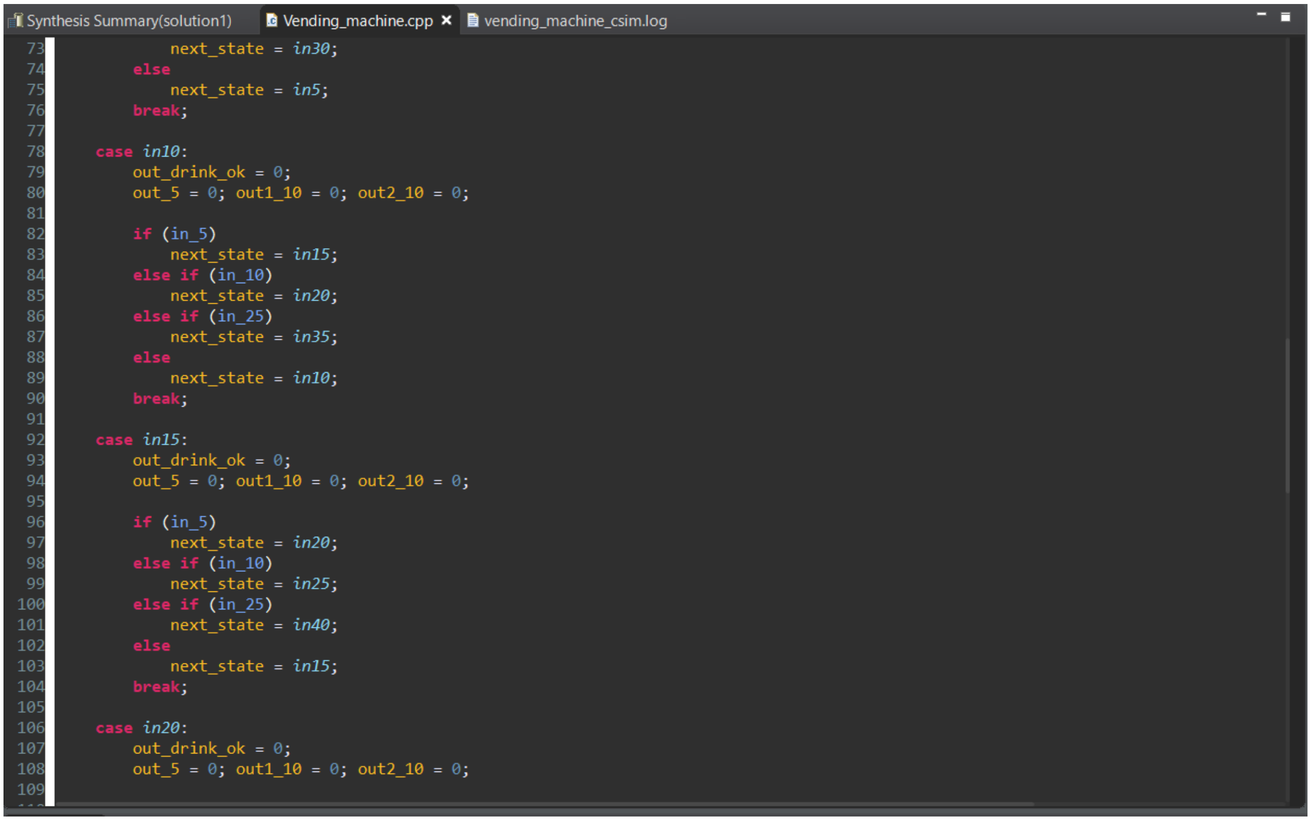
2. Cod CPP + rezultate



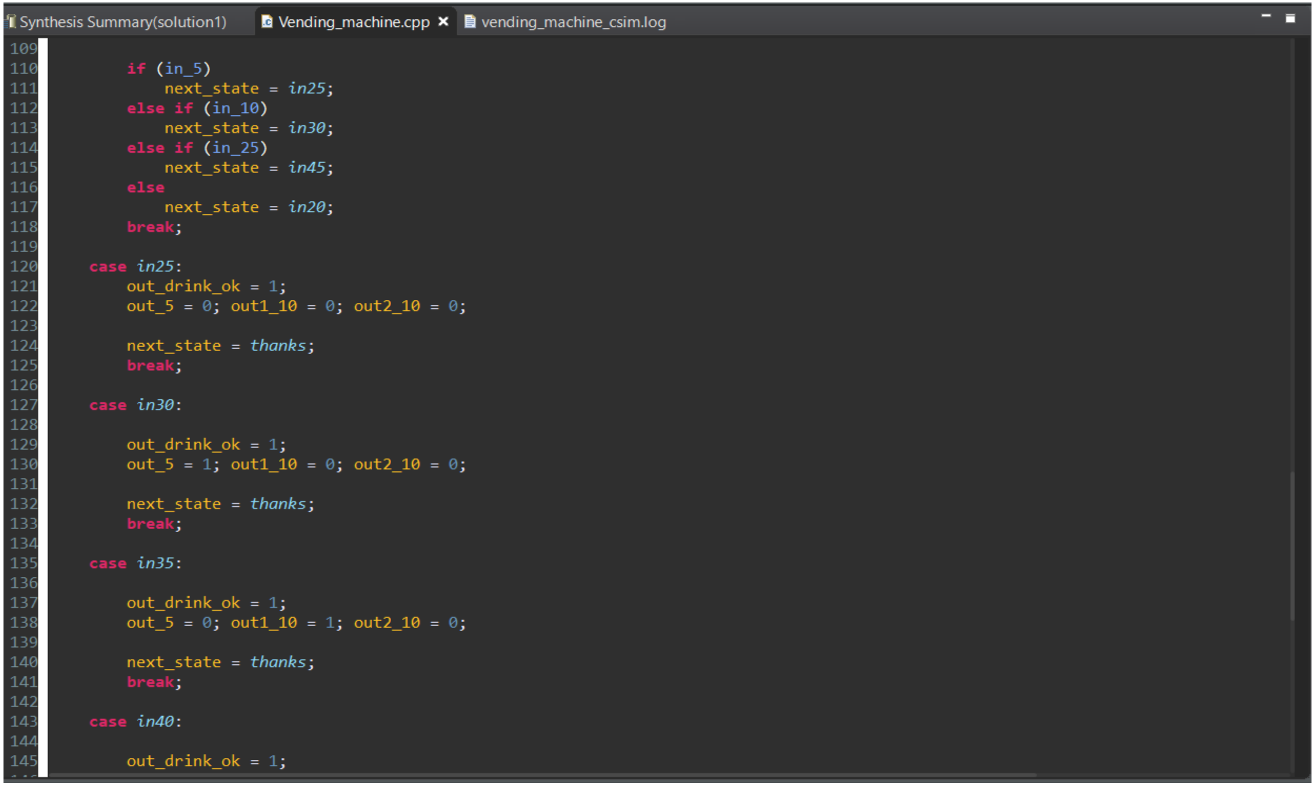
```cpp
#include <iostream>
#include "Vending_machine.h"


using namespace std;

void money_in(bool in_5, bool in_10, bool in_25) {

    bool out_drink, in_thanks, rest_5, rest1_10, rest2_10;

    vending_machine(in_5, in_10, in_25, in_thanks, out_drink, rest_5, rest1_10, rest2_10);

    cout << "intra 5 = " << in_5 << ", intra 10 = " << in_10 << ", intra 25 = " << in_25 << ", in_thanks = " << in_thanks << endl;
    cout << "out_drink = " << out_drink << ", rest bancnota 5 = " << rest_5 << ", rest bancnota1 10 = " << rest1_10 << ", rest banc
    cout << endl;
}

void press_thanks() {

    bool out_drink, in_thanks = 1, rest_5, rest1_10, rest2_10;

    vending_machine(0, 0, 0, in_thanks, out_drink, rest_5, rest1_10, rest2_10);

    cout << "in_5 = 0, in_10 = 0, in_25 = 0, in_thanks = " << in_thanks << endl << "S-a apasat THANK YOU!!!" << endl;
    cout << "out_drink = " << out_drink << ", rest bancnota 5 = " << rest_5 << ", rest bancnota1 10 = " << rest1_10 << ", rest banc
    cout << endl;
}

void vending_machine(bool in_5, bool in_10, bool in_25, bool in_thanks, bool& out_drink, bool& rest_5, bool& rest1_10, bool& rest2_

#pragma HLS INTERFACE ap_none port=in_5
#pragma HLS INTERFACE ap_none port=in_10
#pragma HLS INTERFACE ap_none port=in_25
#pragma HLS INTERFACE ap_none port=in_thanks
#pragma HLS INTERFACE ap_none port=out_drink
#pragma HLS INTERFACE ap_none port=rest_5
#pragma HLS INTERFACE ap_none port=rest1_10
```



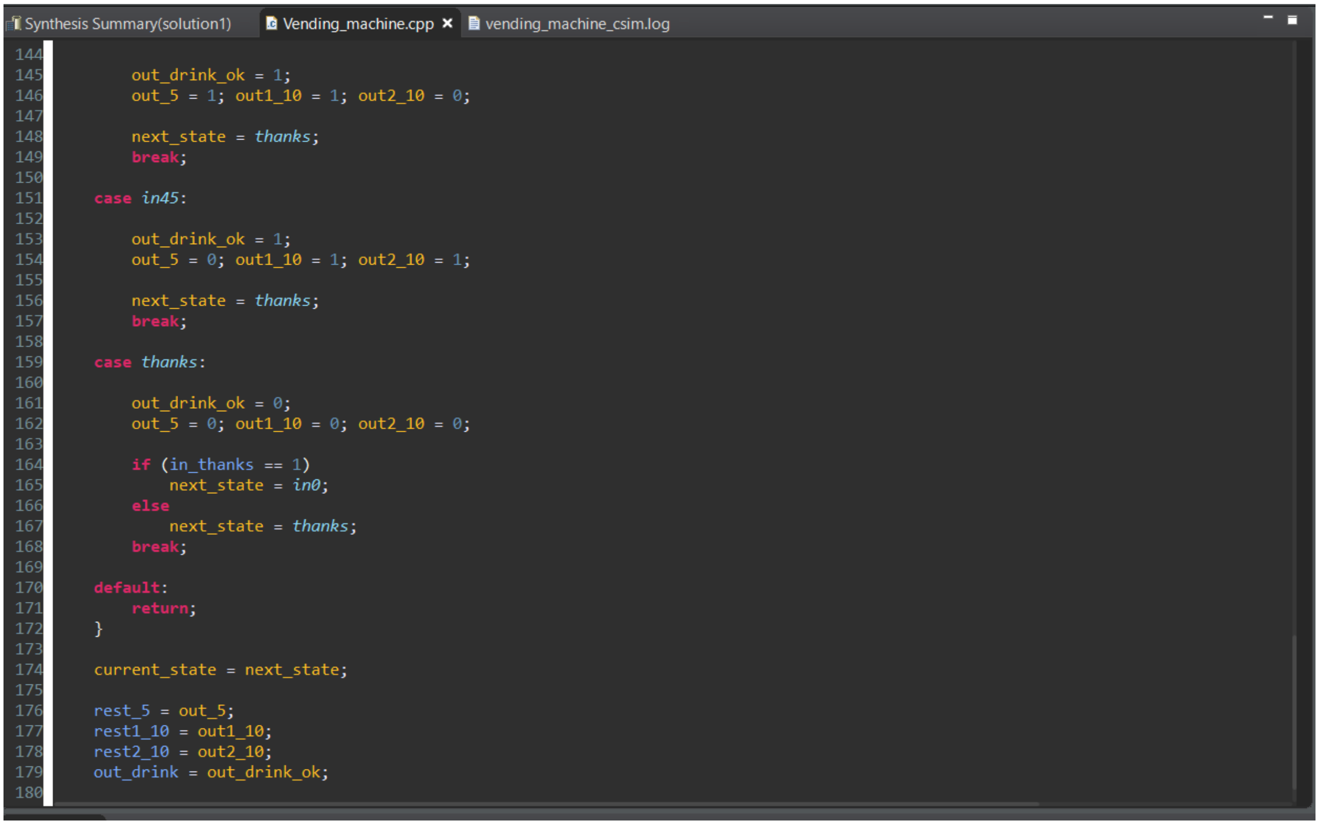
```cpp
#pragma HLS INTERFACE ap_none port=rest1_10
#pragma HLS INTERFACE ap_none port=rest2_10
#pragma HLS INTERFACE ap_ctrl_none port=return

    static State current_state;
    State next_state = current_state;

    bool out_drink_ok;
    bool out_5, out1_10, out2_10;


    switch (current_state) {

    case in0:
        out_drink_ok = 0;
        out_5 = 0; out1_10 = 0; out2_10 = 0;

        if (in_5)
            next_state = in5;
        else if (in_10)
            next_state = in10;
        else if (in_25)
            next_state = in25;
        else
            next_state = in0;
        break;

    case in5:
        out_drink_ok = 0;
        out_5 = 0; out1_10 = 0; out2_10 = 0;

        if (in_5)
            next_state = in10;
        else if (in_10)
            next_state = in15;
        else if (in_25)
            next_state = in30;
```



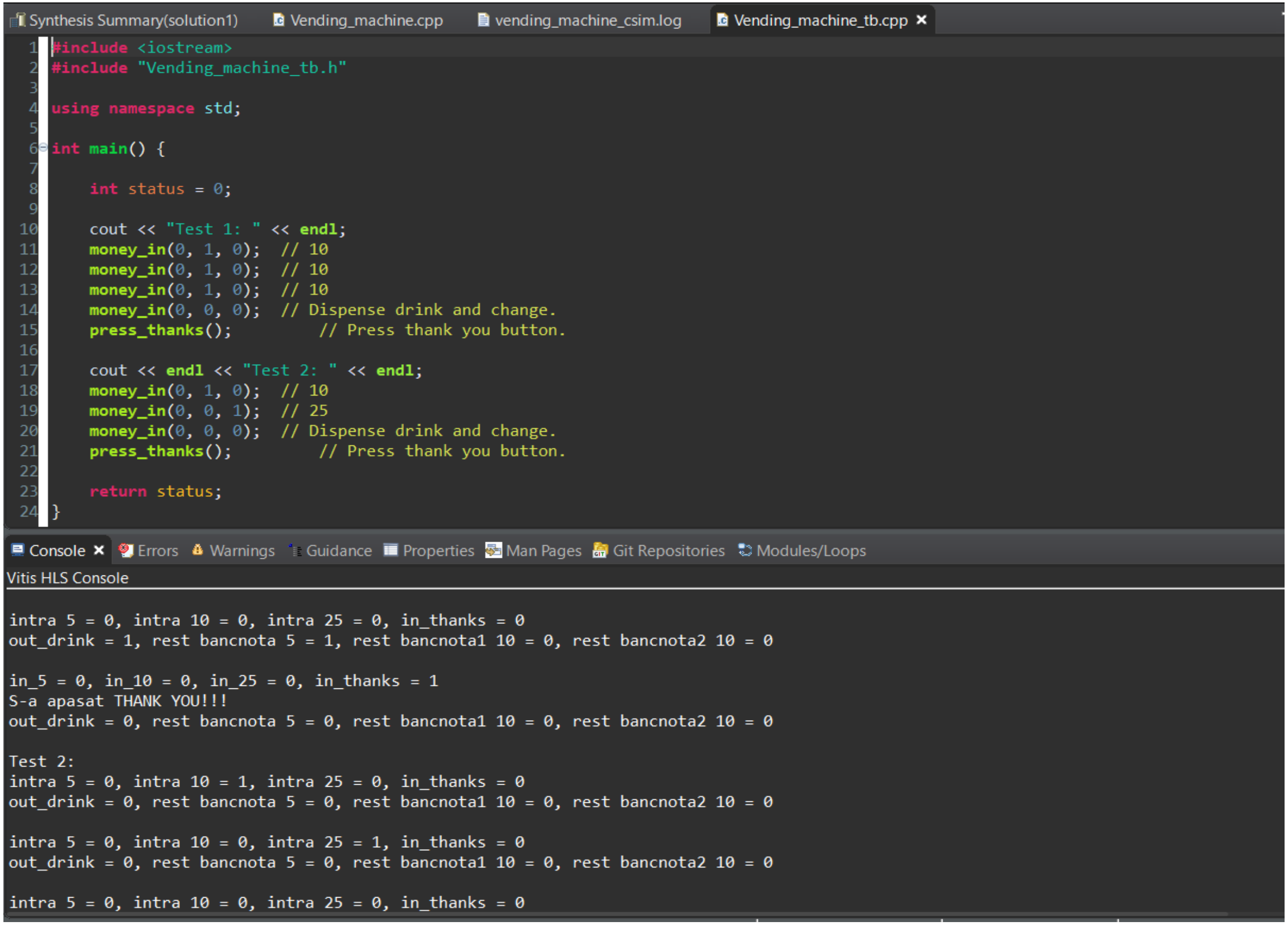
```
            next_state = in30;
        else
            next_state = in5;
        break;

    case in10:
        out_drink_ok = 0;
        out_5 = 0; out1_10 = 0; out2_10 = 0;

        if (in_5)
            next_state = in15;
        else if (in_10)
            next_state = in20;
        else if (in_25)
            next_state = in35;
        else
            next_state = in10;
        break;

    case in15:
        out_drink_ok = 0;
        out_5 = 0; out1_10 = 0; out2_10 = 0;

        if (in_5)
            next_state = in20;
        else if (in_10)
            next_state = in25;
        else if (in_25)
            next_state = in40;
        else
            next_state = in15;
        break;

    case in20:
        out_drink_ok = 0;
        out_5 = 0; out1_10 = 0; out2_10 = 0;

```



```

        if (in_5)
            next_state = in25;
        else if (in_10)
            next_state = in30;
        else if (in_25)
            next_state = in45;
        else
            next_state = in20;
        break;

    case in25:
        out_drink_ok = 1;
        out_5 = 0; out1_10 = 0; out2_10 = 0;

        next_state = thanks;
        break;

    case in30:

        out_drink_ok = 1;
        out_5 = 1; out1_10 = 0; out2_10 = 0;

        next_state = thanks;
        break;

    case in35:

        out_drink_ok = 1;
        out_5 = 0; out1_10 = 1; out2_10 = 0;

        next_state = thanks;
        break;

    case in40:

        out_drink_ok = 1;
```



```

        out_drink_ok = 1;
        out_5 = 1; out1_10 = 1; out2_10 = 0;

        next_state = thanks;
        break;

    case in45:

        out_drink_ok = 1;
        out_5 = 0; out1_10 = 1; out2_10 = 1;

        next_state = thanks;
        break;

    case thanks:

        out_drink_ok = 0;
        out_5 = 0; out1_10 = 0; out2_10 = 0;

        if (in_thanks == 1)
            next_state = in0;
        else
            next_state = thanks;
        break;

    default:
        return;
    }

    current_state = next_state;

    rest_5 = out_5;
    rest1_10 = out1_10;
    rest2_10 = out2_10;
    out_drink = out_drink_ok;

```



```
#include <iostream>
#include "Vending_machine_tb.h"

using namespace std;

int main() {

    int status = 0;

    cout << "Test 1: " << endl;
    money_in(0, 1, 0);  // 10
    money_in(0, 1, 0);  // 10
    money_in(0, 1, 0);  // 10
    money_in(0, 0, 0);  // Dispense drink and change.
    press_thanks();         // Press thank you button.

    cout << endl << "Test 2: " << endl;
    money_in(0, 1, 0);  // 10
    money_in(0, 0, 1);  // 25
    money_in(0, 0, 0);  // Dispense drink and change.
    press_thanks();         // Press thank you button.

    return status;
}
```

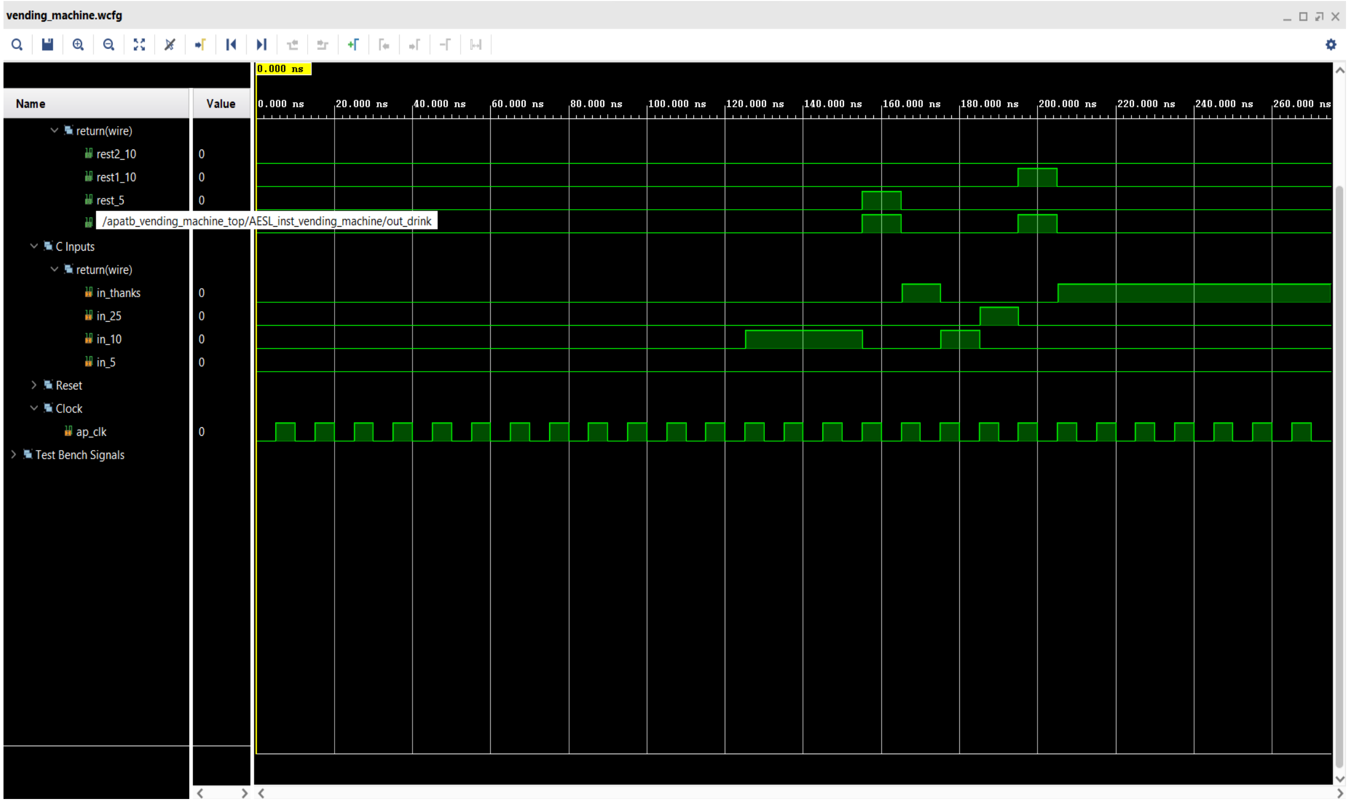
```
intra 5 = 0, intra 10 = 0, intra 25 = 0, in_thanks = 0
out_drink = 1, rest bancnota 5 = 1, rest bancnota1 10 = 0, rest bancnota2 10 = 0

in_5 = 0, in_10 = 0, in_25 = 0, in_thanks = 1
S-a apasat THANK YOU!!!
out_drink = 0, rest bancnota 5 = 0, rest bancnota1 10 = 0, rest bancnota2 10 = 0

Test 2:
intra 5 = 0, intra 10 = 1, intra 25 = 0, in_thanks = 0
out_drink = 0, rest bancnota 5 = 0, rest bancnota1 10 = 0, rest bancnota2 10 = 0

intra 5 = 0, intra 10 = 0, intra 25 = 1, in_thanks = 0
out_drink = 0, rest bancnota 5 = 0, rest bancnota1 10 = 0, rest bancnota2 10 = 0

intra 5 = 0, intra 10 = 0, intra 25 = 0, in_thanks = 0
```

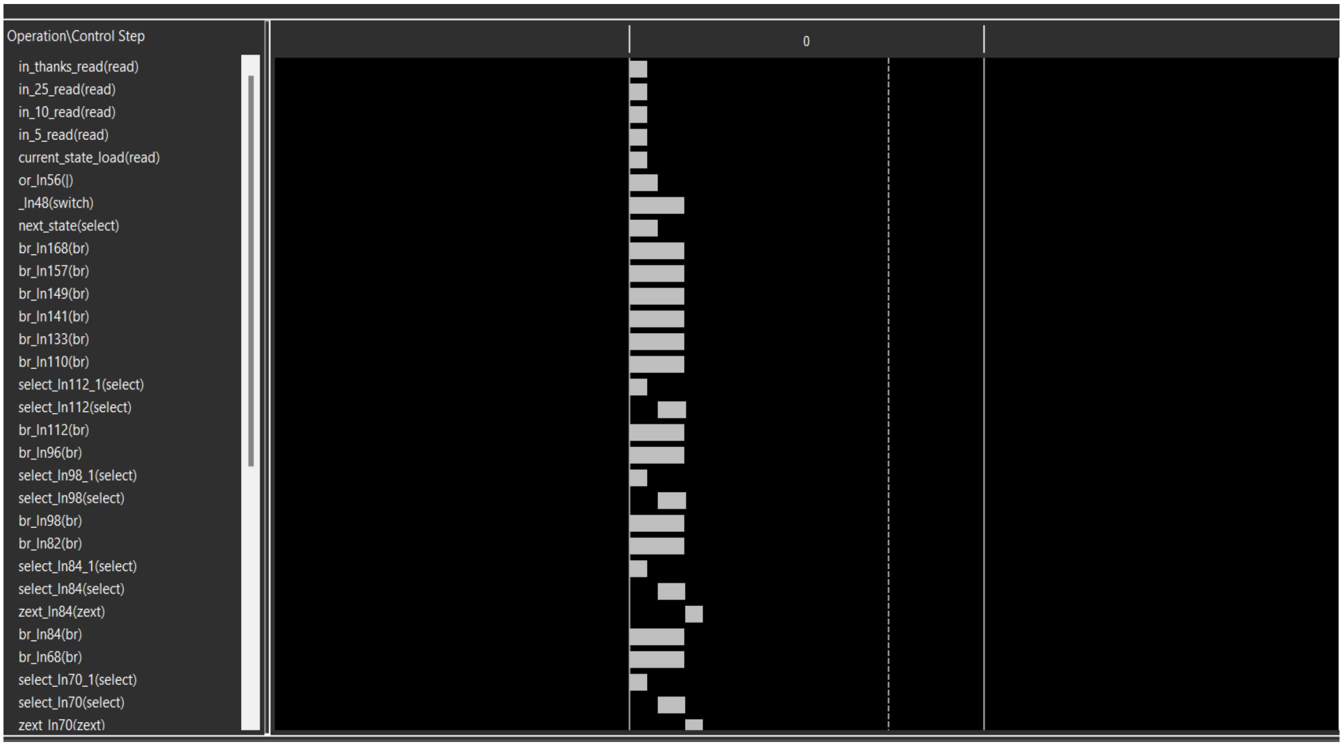
In primul test se introduc bancnote de 10,10,10 si da rest 5.

In al doilea test se introduc 10,25 si da rest 10.

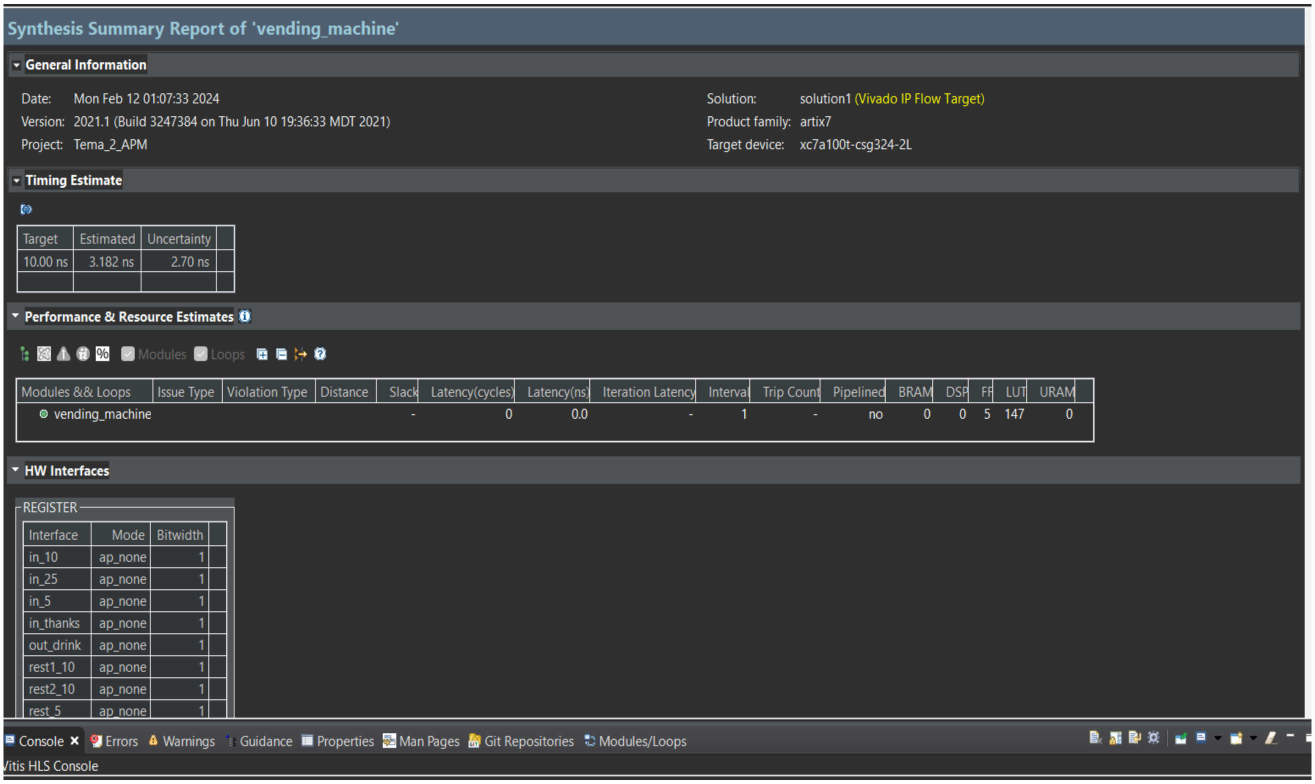
3. Raport care demonstreaza ca circuitul este secvential:

Formele de unda :





4. Raportul care arată resursele hardware folosite în cazul utilizării unui FPGA Artix 7



Synthesis Summary Report of 'vending_machine'

General Information

Date:	Mon Feb 12 01:07:33 2024	Solution:	solution1 (Vivado IP Flow Target)
Version:	2021.1 (Build 3247384 on Thu Jun 10 19:36:33 MDT 2021)	Product family:	artix7
Project:	Tema_2_APM	Target device:	xc7a100t-csg324-2L

Timing Estimate

Target	Estimated	Uncertainty
10.00 ns	3.182 ns	2.70 ns

Performance & Resource Estimates

Modules && Loops	Issue Type	Violation Type	Distance	Slack	Latency(cycles)	Latency(ns)	Iteration Latency	Interval	Trip Count	Pipelined	BRAM	DSP	FF	LUT	URAM
● vending_machine				-	0	0.0	-	1	-	no	0	0	5	147	0

HW Interfaces

REGISTER

Interface	Mode	Bitwidth
in_10	ap_none	1
in_25	ap_none	1
in_5	ap_none	1
in_thanks	ap_none	1
out_drink	ap_none	1
rest1_10	ap_none	1
rest2_10	ap_none	1
rest_5	ap_none	1