



## 1. Introducere in Xilinx Vivado 2018

### 1.1 Crearea unui proiect

În acest scop, vom urmări pașii de mai jos:

- Create project
- Next
- Alegem nume și locație pentru noul proiect.
- Alegem RTL project type

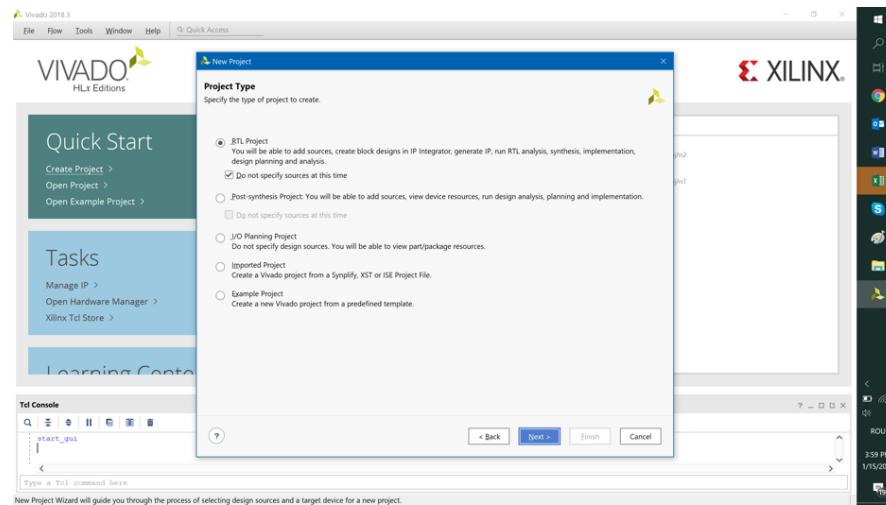


Figura 1.1: Crearea unui proiect

În pasul următor vom selecta tipul plăcii pe care o vom folosi pentru implementarea proiectului. Astfel filtrele pe care le vom folosi pentru a utiliza placuta hardware din dotarea laboratorului sunt:

- Family: **Artix-7**
- Package: **csg324**

- Speed: -1
- Board: xc7a100tcs324-1
- Apoi apăsăm Next și Finish

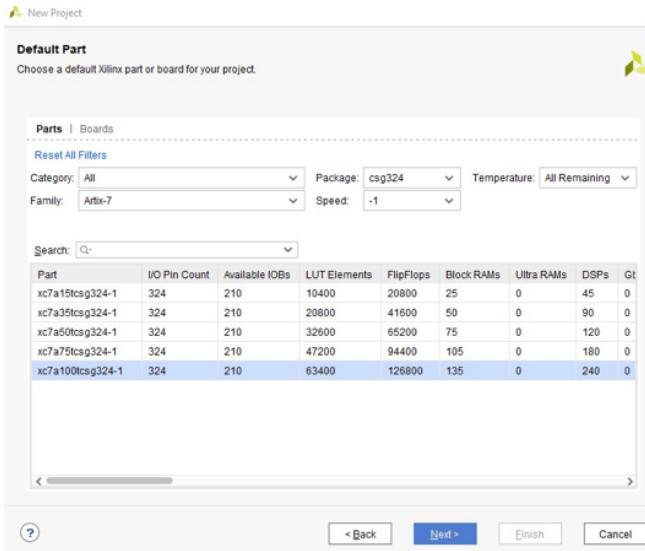


Figura 1.2: Tipul placii

Proiectul a fost astfel creat și vor fi deschise cele 5 frame-uri din Figura 1.2:

- Flow Navigator unde apar toate procesele prin care trece proiectul dezvoltat de noi
- Ierarhia fișierelor sursă
- Proprietăți
- Consolă Tcl pentru mesajele apărute în timpul rulării și diverse comenzi
- Spațiul de lucru (editare/design)

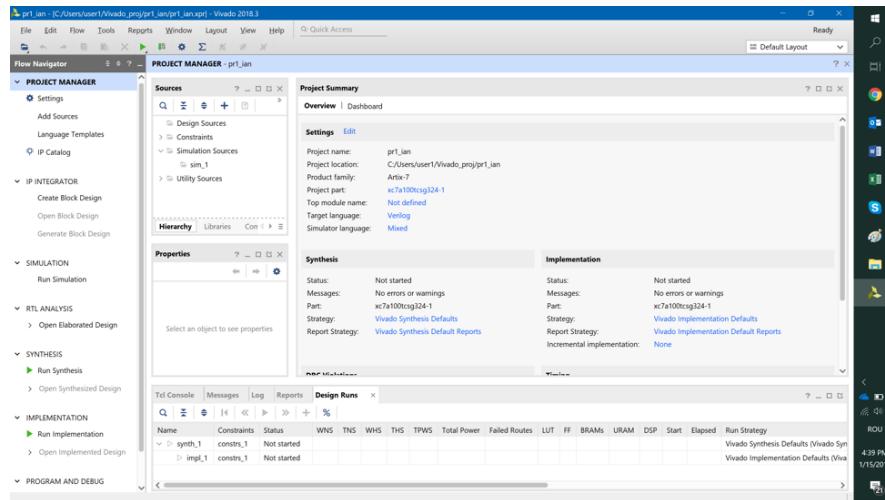


Figura 1.3: Fereastra principală

În frame-ul dedicat surselor (Figura 1.3, se observă semnul “+” care ne va da posibilitatea sa

alegem tipul de fișier pe care vrem sa îl creăm sau adăugam în proiectul nostru (Figura 1.4b):

- Adăugarea sau crearea fișierului de constrângeri
- Adăugarea sau crearea fișierului de tip design (proiectare cu componente)
- Adăugarea sau crearea fișierului pentru simulare

Pentru crearea și adăugarea de fișiere putem alege și opțiunea Add sources din Project Manager (frame-ul Flow Navigator).

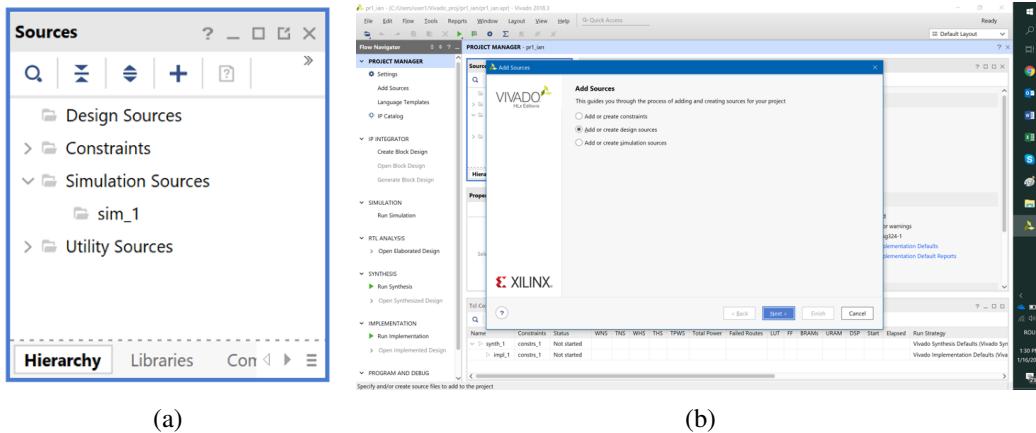


Figura 1.4: Ierarhia de surse

## 1.2 Proiectarea cu ajutorul componentelor din librăria Xilinx

Pentru a proiecta un modul cu ajutorul componentelor existente in librăria Xilinx, se utilizează opțiunea **Create Block Design** din **IP Integrator** (Figura 1.5) Apoi se alege un nume fișierului de tip block design care va avea extensia \*.bd si va fi vizibilă in fereastra surselor.

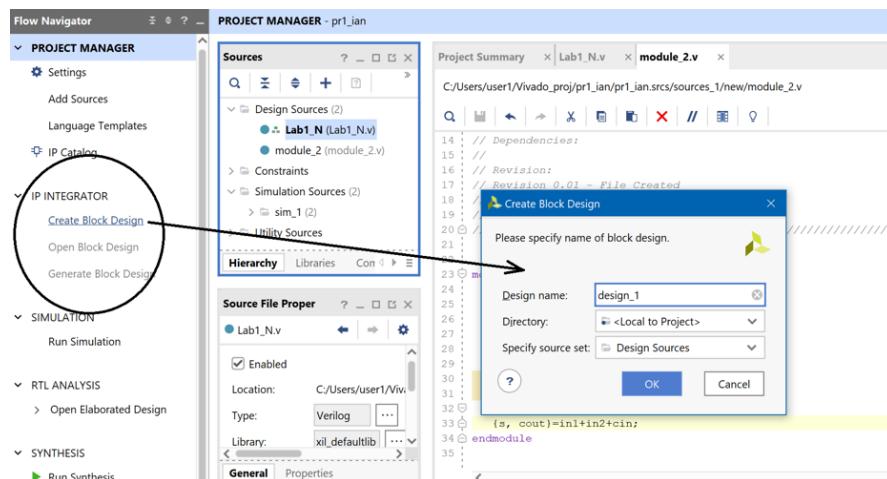


Figura 1.5: Adăugarea unui bloc

În noul mediu de lucru deschis se selectează “+” din meniu si astfel apare lista componentelor Xilinx ce pot fi utilizate (Figura 1.6).

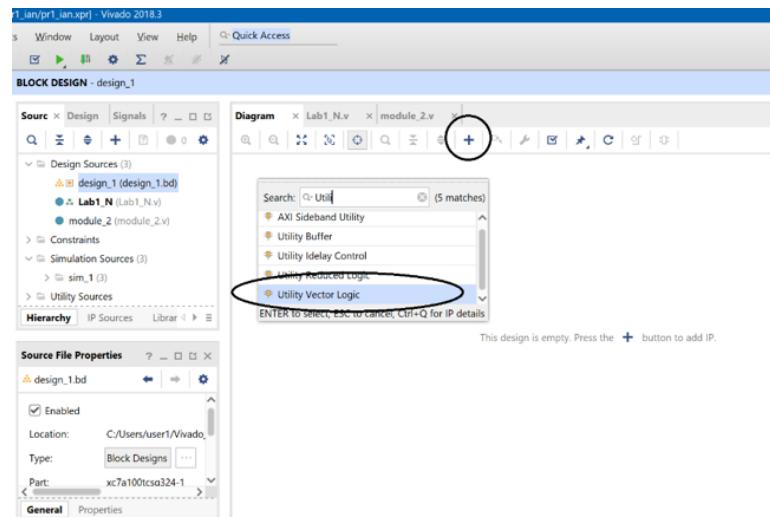


Figura 1.6: Listă componente Xilinx

Pentru a lucra cu porti logice, se selectează din listă Utility Vector Logic. După Enter, va putea fi observată componenta plasată în mediul de lucru. Ea poate fi selectată, iar cu ajutorului apăsării click dreapta al mouse-ului, apare o listă cu opțiuni astfel încât componenta poate fi modificată conform cerințelor design-ului. Selectați Make External pentru a adăuga porturi de intrare și ieșire componentei (Figura 1.7a). Denumirea porturilor se poate schimba în fereastra External Port Properties. Pentru a modifica tipul porturilor și dimensiunea porturilor, dublu-click pe componentă apoi selectați opțiunile dorite (Figura 1.7b)).

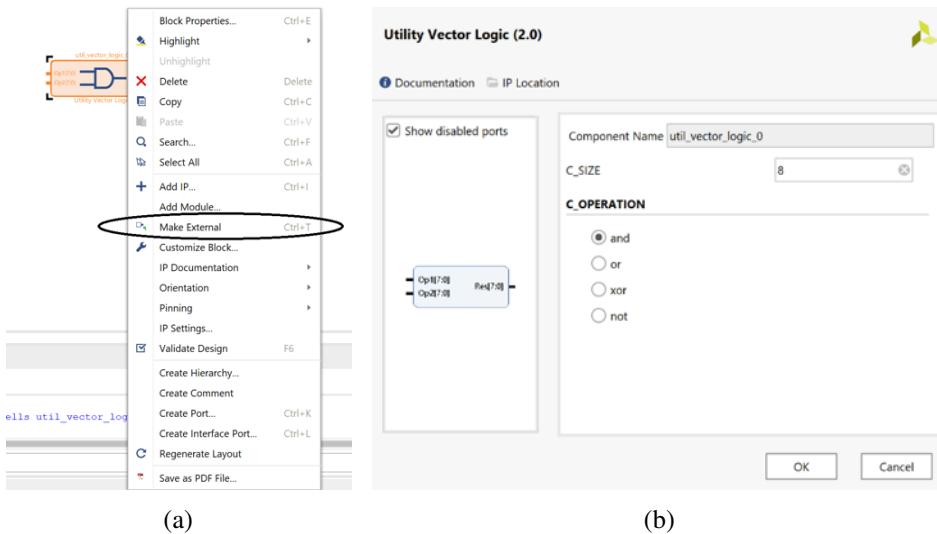


Figura 1.7: Configurare porti

Un exemplu de circuit descris în Block Diagram (design\_1.bd) este prezentat în Figura 1.8.

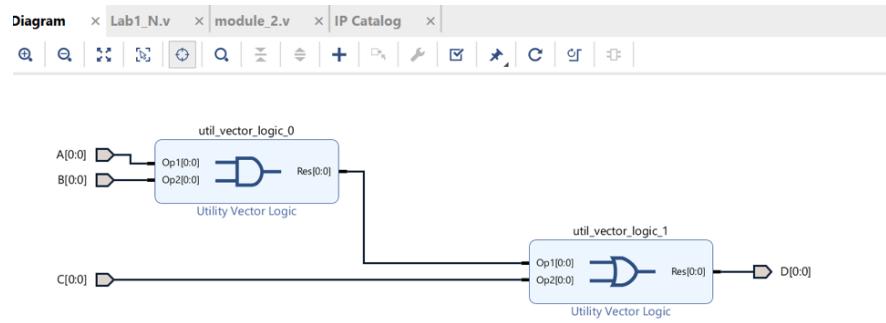


Figura 1.8: Exemplu circuit

### 1.3 Realizarea constrângerilor

Pentru a putea simula circuitul nostru pe plăcuța hardware este necesar să facem o mapare între resursele de I/O ale placii și pinii de I/O ale circuitului nostru. Pentru aceasta, trebuie ca circuitul proiectat să fie transformat într-un fișier Verilog (sau VHDL). De aceea vom apăsa click dreapta pe fișierul dorit și vom selecta **Create HDL Wrapper**. Astfel, în surse a apărut un nou fișier Verilog pentru circuitul design\_1.bd, care se numește design\_1\_wrapper.v.

De această dată vom apăsa **Add sources** și vom selecta **Add or Create Constraints**. În fereastra următoare alegem **Add file** și vom adăuga fișierul Nexys4DDR\_Master.xdc. De asemenea trebuie să avem selectată opțiunea **Copy constraints files into project** (Figura 1.9).

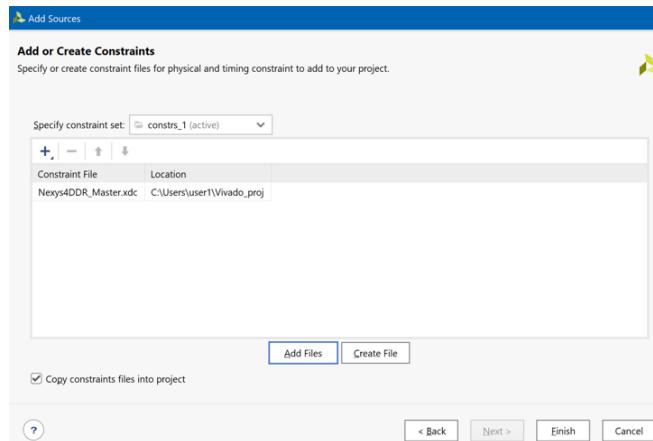


Figura 1.9: Realizarea constrângerilor

După Finish, acest fișier de constrângeri a apără în fereastra surselor și, pentru a-l adapta designului nostru, trebuie să îl deschidem (dublu click) și să deselectăm liniile de cod aferente porturilor noastre de intrare și ieșire prezente în design. Pentru că avem 3 intrări și o ieșire de un bit, vom avea nevoie de 3 switch-uri și un led. Așadar, deselectăm liniile următoare (Figura 1.10) și folosim denumirile porturilor noastre:

Astfel s-a creat o legătura între pinii placii pe care o vom utiliza și porturile noastre A, B, C, D.

```

7 #set_property -dict { PACKAGE_PIN E3   IOSTANDARD LVCMS33 } [get_ports { CLK100MHZ }]; #IO_L12P_T1_MRCC_35 Sch=clk100mhz
8 #create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports {CLK100MHZ}];
9
10
11 ##Switches
12
13 set_property -dict { PACKAGE_PIN J15   IOSTANDARD LVCMS33 } [get_ports { A[0] }]; #IO_L24N_T3_R50_15 Sch=sv[0]
14 set_property -dict { PACKAGE_PIN L16   IOSTANDARD LVCMS33 } [get_ports { B[0] }]; #IO_L3N_T0_D05_EMCLR_14 Sch=sv[1]
15 set_property -dict { PACKAGE_PIN M13   IOSTANDARD LVCMS33 } [get_ports { C[0] }]; #IO_L6N_T0_D08_VREF_14 Sch=sv[2]
16 #set_property -dict { PACKAGE_PIN R15   IOSTANDARD LVCMS33 } [get_ports { SW[3] }]; #IO_L13N_T2_MRCC_14 Sch=sv[3]
17 #set_property -dict { PACKAGE_PIN R17   IOSTANDARD LVCMS33 } [get_ports { SW[4] }]; #IO_L12N_T1_MRCC_14 Sch=sv[4]
18 #set_property -dict { PACKAGE_PIN T18   IOSTANDARD LVCMS33 } [get_ports { SW[5] }]; #IO_L7N_T1_D10_14 Sch=sv[5]
19 #set_property -dict { PACKAGE_PIN U18   IOSTANDARD LVCMS33 } [get_ports { SW[6] }]; #IO_L17N_T2_A15_D29_14 Sch=sv[6]
20 #set_property -dict { PACKAGE_PIN R13   IOSTANDARD LVCMS33 } [get_ports { SW[7] }]; #IO_L5N_T0_D07_14 Sch=sv[7]
21 #set_property -dict { PACKAGE_PIN T8    IOSTANDARD LVCMS33 } [get_ports { SW[8] }]; #IO_L24N_T3_34 Sch=sv[8]
22 #set_property -dict { PACKAGE_PIN U8    IOSTANDARD LVCMS33 } [get_ports { SW[9] }]; #IO_L25_35 Sch=sv[9]
23 #set_property -dict { PACKAGE_PIN R16   IOSTANDARD LVCMS33 } [get_ports { SW[10] }]; #IO_L15P_T2_D05_RDWR_B_14 Sch=sv[10]
24 #set_property -dict { PACKAGE_PIN T19   IOSTANDARD LVCMS33 } [get_ports { SW[11] }]; #IO_L23P_T3_A03_D19_14 Sch=sv[11]
25 #set_property -dict { PACKAGE_PIN H6    IOSTANDARD LVCMS33 } [get_ports { SW[12] }]; #IO_L24P_T3_35 Sch=sv[12]
26 #set_property -dict { PACKAGE_PIN U12   IOSTANDARD LVCMS33 } [get_ports { SW[13] }]; #IO_L20P_T3_A08_D24_14 Sch=sv[13]
27 #set_property -dict { PACKAGE_PIN U11   IOSTANDARD LVCMS33 } [get_ports { SW[14] }]; #IO_L19N_T3_A09_D25_VREF_14 Sch=sv[14]
28 #set_property -dict { PACKAGE_PIN V10   IOSTANDARD LVCMS33 } [get_ports { SW[15] }]; #IO_L21P_T3_DQS_14 Sch=sv[15]
29
30
31 ## LEDs
32
33 set_property -dict { PACKAGE_PIN H17   IOSTANDARD LVCMS33 } [get_ports { D[0] }]; #IO_L18P_T2_A24_15 Sch=led[0]
34 #set_property -dict { PACKAGE_PIN K15   IOSTANDARD LVCMS33 } [get_ports { LED[1] }]; #IO_L24P_T3_R51_15 Sch=led[1]
35 #set_property -dict { PACKAGE_PIN J13   IOSTANDARD LVCMS33 } [get_ports { LED[2] }]; #IO_L17N_T2_A25_18 Sch=led[2]

```

Figura 1.10: Fișierul de constrângeri

## 1.4 Crearea fișierului bitstream

După ce salvăm, apăsăm **Generate bitstream** din **Program and Debug**, fereastra **Flow Navigator**. În **Launch Runs**, putem selecta 4 la nr. of jobs, acest lucru depinzând de performanța procesorului (Figura 1.11a).

După generarea cu succes a fișierului bit, apare fereastra din Figura 1.11b unde se pot realiza diverse acțiuni în pașii următori. De exemplu, selectând **Open Implemented Design**, se poate observa configurația board-ului FPGA utilizat, iar selectând **Open Hardware Manager** vom începe setul de pași necesari realizării download-ului în placă.

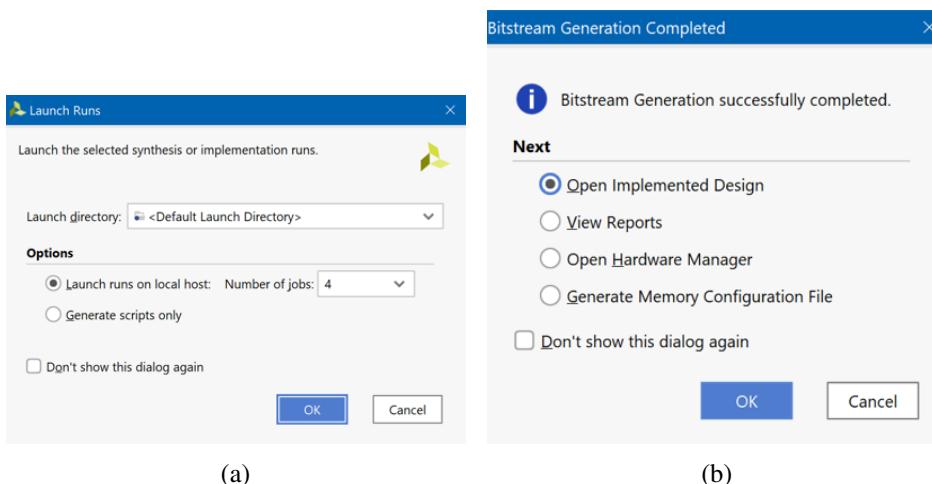


Figura 1.11: Generare fișier bitstream

## 1.5 Download în placa FPGA

Pentru această ultimă etapă, se selectează **Open Hardware Manager** din **Program and Debug (Flow Navigator)**, apoi **Open Target**, **Autoconnect** și **Program Device** (Figura 1.12). Verificăm dacă avem fișierul bit corect, în cazul exemplului prezentat fiind `design_1_wrapper.bit` (Figura 1.13). Apoi apăsăm **Program** și fișierul este download-at în placă.



Figura 1.12: Programare placă

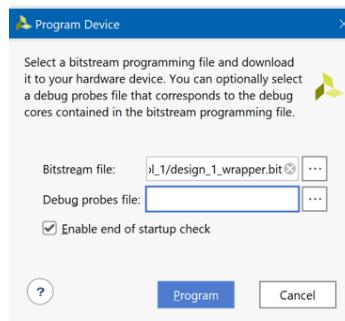


Figura 1.13: Selectare fișier bitstream

Rezultatele simulării sunt prezentate în figurile de mai jos (Figura 1.14a și Figura 1.14b). Se poate observa că, în cazul în care primele 2 switch-ri sunt setate pe 1, ieșirea devine 1 prin aprinderea led-ului 1.

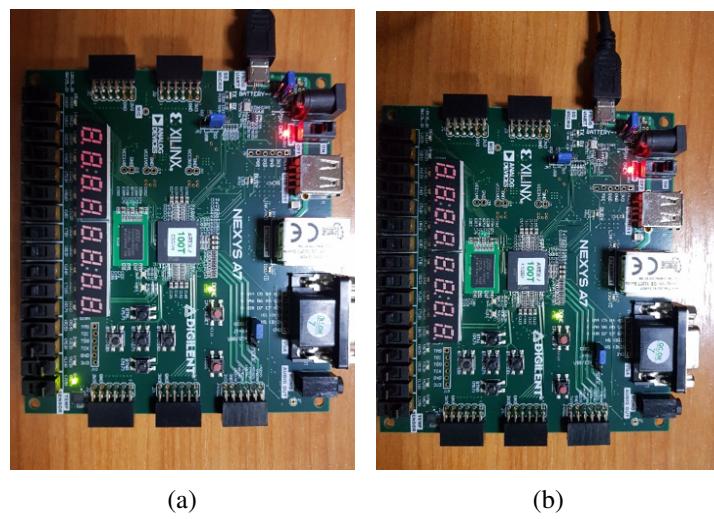


Figura 1.14: Rezultatele simulării

Pentru a încheia laboratorul, verificați funcționarea completă a circuitului proiectat.