

Vásárhelyi József

Vivado IP alapú fejlesztő rendszer

Segédlet

Bevezetés

Jelen segédlet célja megismertetni Xilinx Vivado FPGA fejlesztő alapfokú használatát (tervek kezelése, kapcsolási rajz szerkesztés, VHDL állomány és szimuláció). A Vivado IP Integrator (IPI) előre gyártott szabványos digitális áramköri egységeket (Intellectual Property - IP) és a felhasználó által létrehozott elemeket felhasználva valósít meg egy tervet.

A laboratóriumi gyakorlat terv megvalósítása során különböző tervezési lépéseket ismerünk meg, úgymint: IP tömb és HDL burok (wrapper) létrehozása felhasználói paraméterezés (user constraint), szimuláció, szintézis, huzalozás, bittérkép létrehozás, terv funkcionalitás ellenőrzése, stb.

A segédlet a gyakorlati megvalósítása során a Nexys4 DDR oktatási kártyát használja.

Célok

Az alábbiakban leírt lépések alapján a hallgató a következő kompetenciákat szerzi meg:

- Vivado terv létrehozása adott FPGA áramkör felhasználásával, amelyet a Nexys4 DDR kártya tartalmaz.
- Felhasználói paraméterek szerkesztése a megadott Xilinx Design Constraint (XDC) paraméter állomány alapján.
- Egyéb paraméterezés a TCL scriptek felhasználásával.
- Áramköri szimuláció az XSim szimulátor használatával.
- Terv szintézise és huzalozása.
- Bittérkép – bitstream – létrehozása.
- FPGA konfigurálás a bittérkép letöltésével és funkcionalitás ellenőrzése.

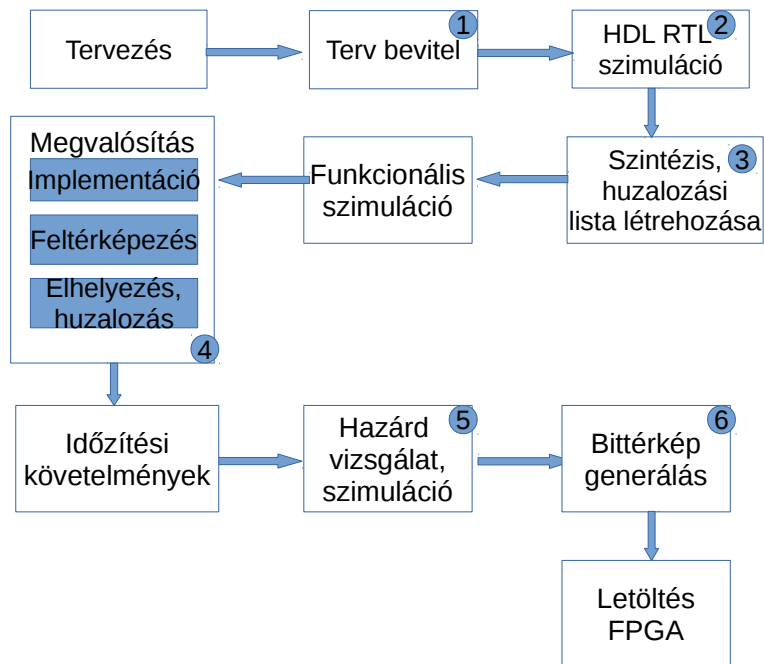
A tipikus tervezési folyamat az 1. Ábrán látható.

Módszertan

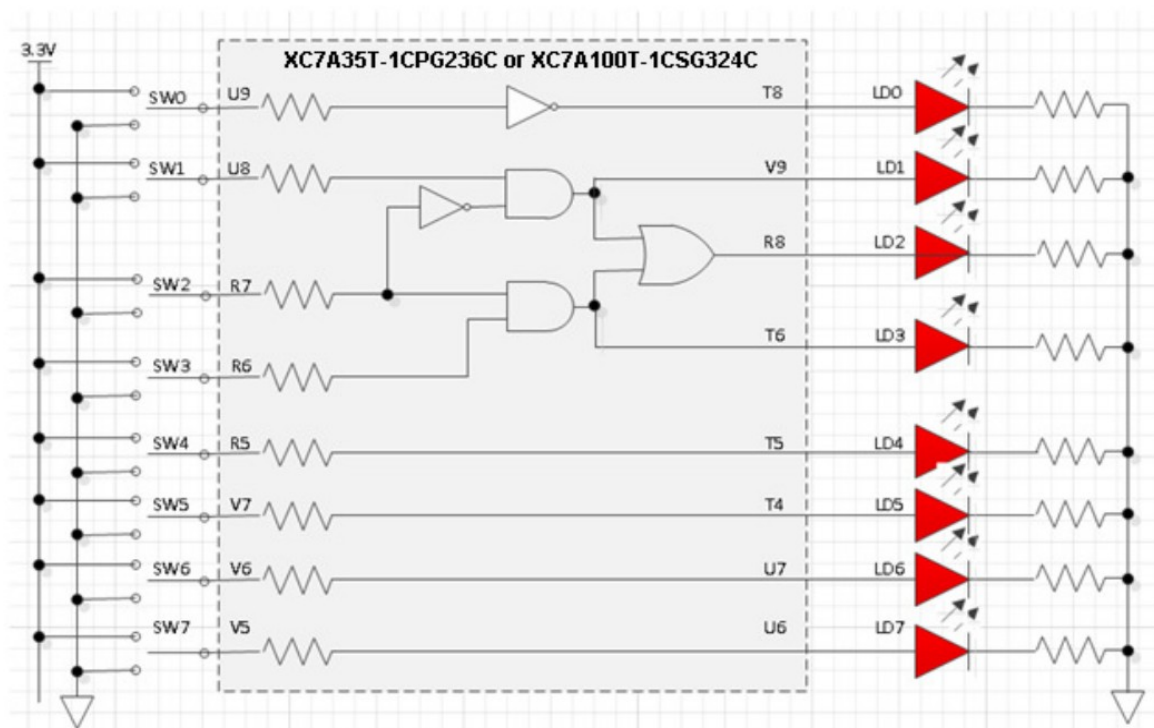
A segédlet áttekinti, majd részletesen leírja az egymás után következő és követendő tervezési-szimulációs lépéseket. Kövesse lépésről lépésre a segédletben leírtakat!

A segédlet terve

Az első terv összeköt néhány bemenetet (kapcsolók) és kimenetet (ledek). Néhány bemenettel pedig egyszerű logikai függvényt valósít meg, amelynek eredményét ugyancsak ledekkel jelezzük, ahogyan az, az 2. Ábrán látható.



1. Ábra: Tipikus tervezési lépések



2. Ábra: Megvalósított kapcsolási rajz.

A segédletben követendő lépések:

- Vivado terv létrehozása és IP könyvtár beállítása;
- Kapcsolási rajz létrehozása;
- A kapcsolási rajz HDL fordítása és lábkiosztás megadása;
- Funkcionális szimuláció megvalósítása XSim szimulátorral;
- Terv szintézis;
- Terv fordítás – implementáció;
- Időfüggő viselkedés vizsgálat szimulátorral;
- Funkcionális ellenőrzés az oktató kártya használatával.

1. Vivado Terv létrehozása az IDE használatával

1. lépés

1.1 A Vivado grafikus környezet indítása és az első terv létrehozása;

Céláramkör: **XC7A 100T CSG324C – 1 !!!** Feladat: Az első HDL terv létrehozása. A mellékelt tutorial.vhd és a tutorial.xdc forrásállományok használata [1.]

- 1.1.1. Indítsa el a Vivado 2016 programot. Az asztalon található parancsikon segítségével, (lásd 3. Ábra) vagy Linux operációs rendszer alatt a **vivado.sh** parancssal terminál ablakból.



3. Ábra: Vivado parancs ikon

- 1.1.2. Új terv létrehozása: kattintás **Create New Project** a terv varázsló elindításához. Kattintás: **Next**

Megjegyzés: Windows: A gyakorlatok során minden hallgatói munkát az S lemezegységre mentsen! ([\\193.6.4.39\student](http://193.6.4.39/student))! Az S lemezegységen hozzon létre egy saját könyvtárat - **neptun kód vagy ékezet nélküli név!!!** A gyakorlat elvégzéséhez szükséges állományok itt megtalálhatók:

<http://www.xilinx.com/support/university/vivado/vivado-teaching-material/hdl-design.html>.

Linux: a saját munkakönyvtárban hozzon létre egy **xup** könyvtárat (**mkdir xup** paranccsal) és a továbbiakban a gyakorlaton végzett munkákat ide mentse! A gyakorlathoz szükséges állományok megtalálhatók:

<http://www.xilinx.com/support/university/vivado/vivado-teaching-material/hdl-design.html>

Az tömörített alkatrészkönyvtár letölthető a Xilinx honlapjáról vagy:

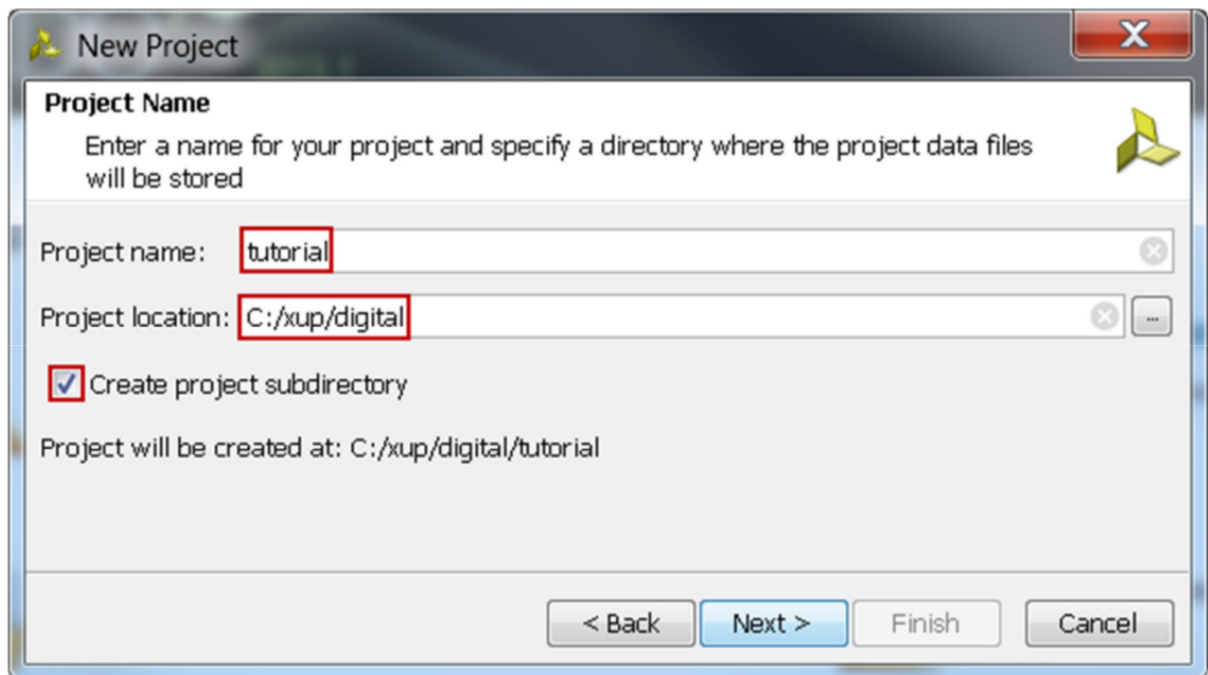
http://mzsola.iit.uni-miskolc.hu/DATA/storages/files/_afKfpEI__beIGGE.zip

Válassza a VHDL állományokat!

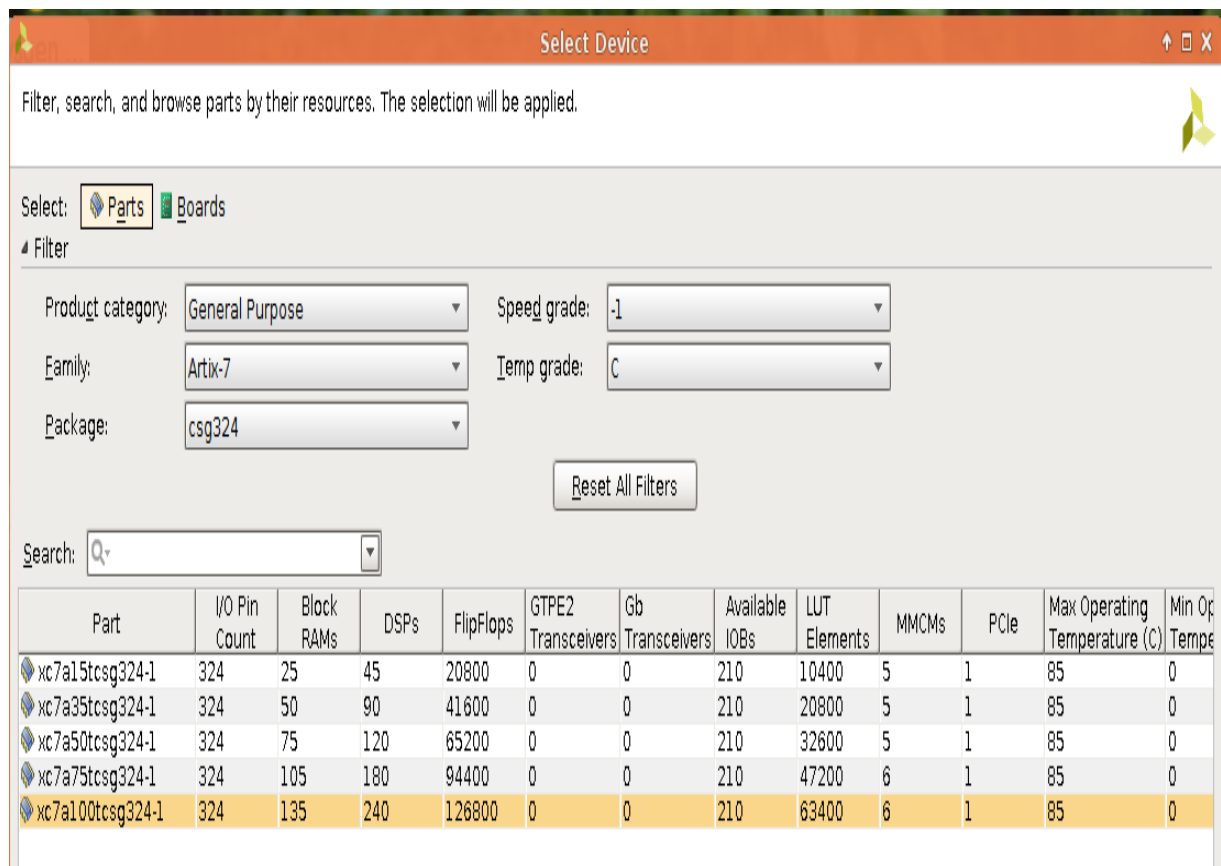
A terv neve legyen „tutorial” (vagy „tanmese”), kiválasztott hardverleíró nyelv pedig VHDL!

- 1.1.3. Kattintás: **Browse** az S lemezegység kiválasztásához. Hozzon létre egy saját könyvtárat. Pl: [s:\vajo](#); kattintás: **Select**
- 1.1.4. Írjuk be a terv nevét pl: tutorial (*tanmese*) és ellenőrizze, hogy kijelölte a **Create Project Subdirectory** (alkönyvtár létrehozása) parancsot (4. Ábra). Kattintás: **Next**
- 1.1.5. **Terv típus: RTL Project** és kattintás **Next**
- 1.1.6. Terv leíró nyelve: **VHDL**, szimuláció: vegyes (mixed)
- 1.1.7. Kattintás: **Next**
- 1.1.8. Kattintás: **Next**, a következő ablak a paraméter állományok hozzáadása (Constraints File). A paraméterező állomány segítségével adjuk meg a lábkiosztást is – Kapcsolók, LED-ek, stb. Most ezt mellőzzük. Kattintás: **Next**

1.1.9. **Áramkör típus kiválasztása (5. Ábra). A legördülő opciókból Válassza ki az ábrán látható áramkört: XC7A100TCSG324-1. Kattintás: Next**

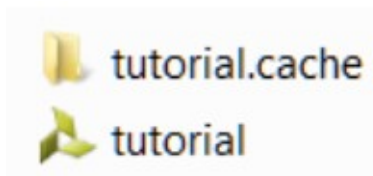


4. Ábra: Terv létrehozása - név és könyvtár beállítás



5. Ábra: Áramkör típusának kiválasztása.

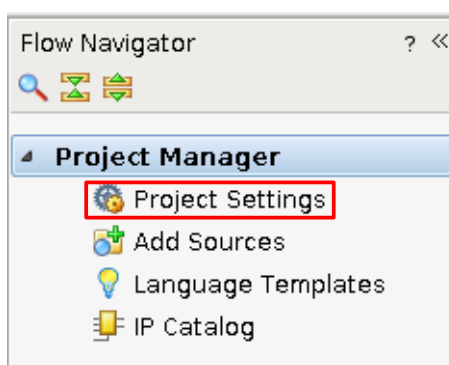
- 1.1.10. Kattintás a terv létrehozásához: **Finish**. Előbb azonban ellenőrizze a beállításokat. Ezek később is módosíthatók! Vizsgálja meg a terv létrehozása után a könyvtár struktúrát. A terv könyvtárban található egy „*terv_neve.cache*” könyvtár és a „*terv_neve.xpr*” projekt állomány (6. Ábra)



6. Ábra: Terv könyvtár tartalma.

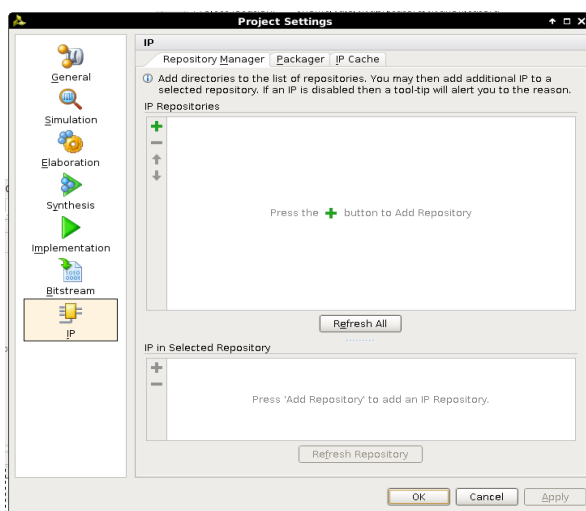
1.2. Az alkatrész könyvtár elérési útvonalának beállítása.

- 1.2.1. Az „XUP.zip” tömörített alkatrészkönyvtárat csomagolja ki a terv könyvtárba. Az elérési útvonalat jegyezze meg, mert erre szüksége lesz. A példában a kicsomagolt könyvtár a /opt/Xilinx/XUP_Lib útvonalon lehet elérni! (linux)
- 1.2.2. A projekt kezelő ablakban kattintson a terv beállításra a **Flow Navigator** → **Project Manager** → **Project Settings** parancsra



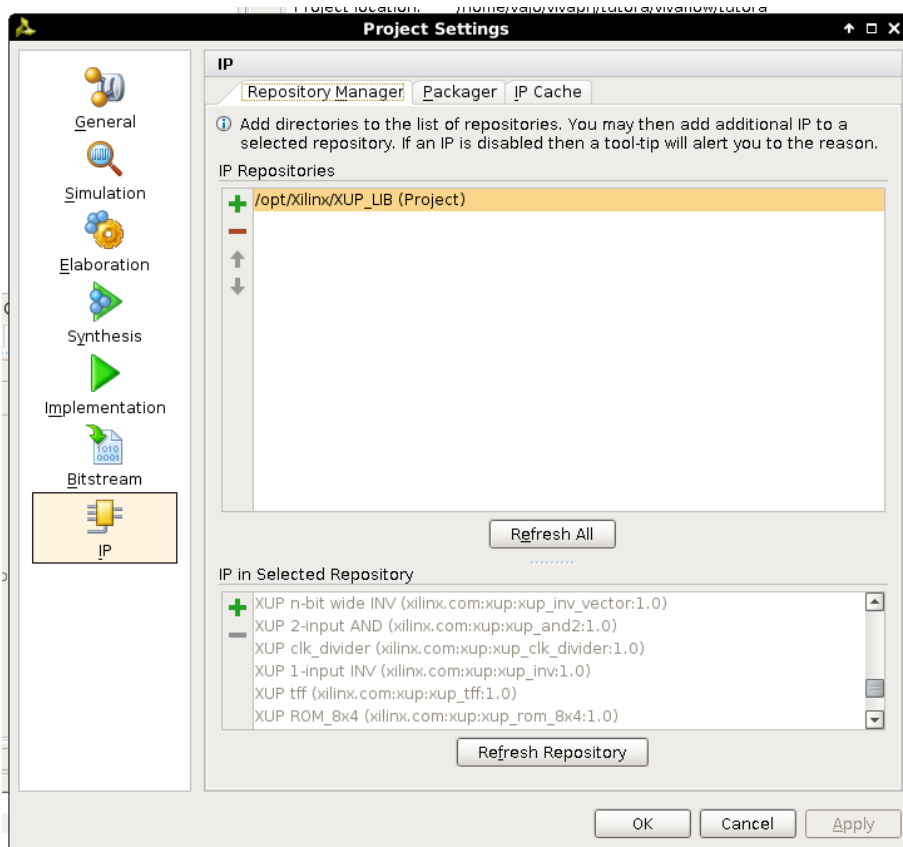
7. Ábra: Terv paraméterek beállítása
- alkatrész könyvtár elérési útvonal

- 1.2.3. A terv beállítások ablakban kattintson az IP beállításokra (8. Ábra). Majd kattinson a „+” (Press the + button to Add Repository) az IP elemkönyvtár kiválasztásához.



8. Ábra: IP beállítás

- 1.2.4. Válassza ki az s:\XUP\XUP_LIB könyvtárat és adja hozzá az IP könyvtárakhoz. Kattintás: **Select**. A fejlesztő rendszer beolvassa és felsorolja az IP elemeket .



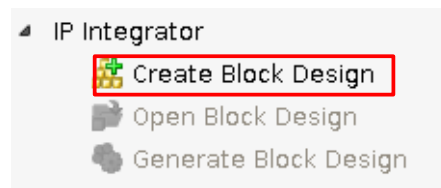
9. Ábra: IP alkatrészek beolvasása

- 1.2.5. Kattintás: **OK**

2. Modulokból álló kapcsolási rajz létrehozása 2. lépés

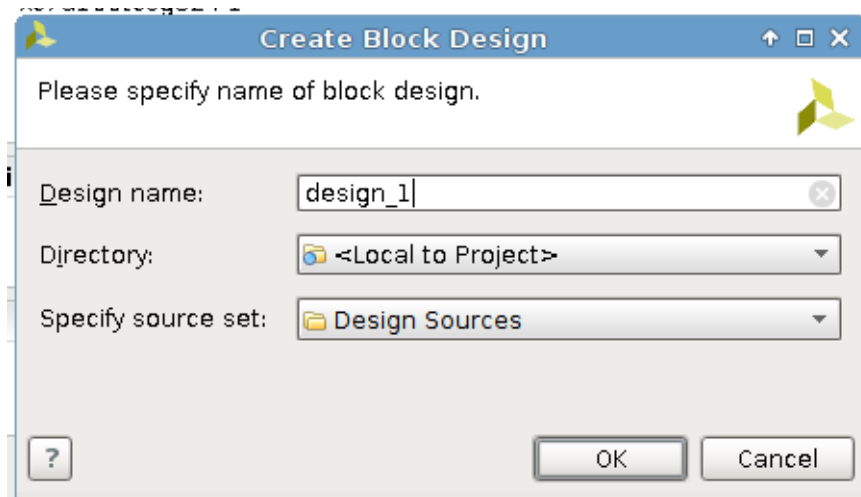
2.1. Modulokból álló kapcsolási rajz megvalósítása

- 2.1.1. A projekt kezelő ablakban kattintson a modul rajz létrehozása ikonra **Flow Navigator** → **IP Intergrator** → **Create Block Design**




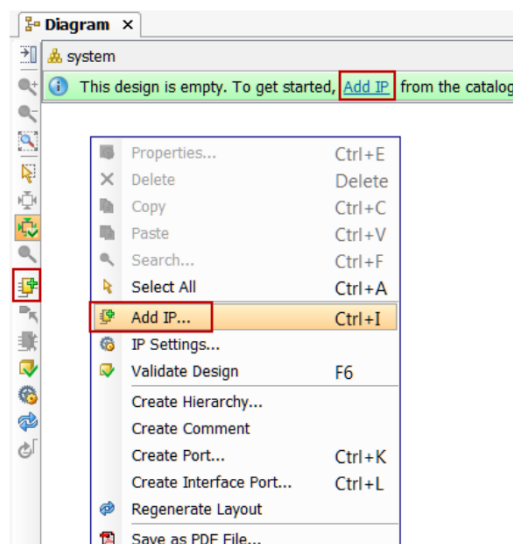
10. Ábra: Modul rajz létrehozása

- 2.1.2. Kattintás: **OK** a „design1” nevű blokk rajz létrehozásához – lásd 11. Ábra.



11. Ábra: Blokk rajz létrehozása

- 2.1.3. A diagram panelben kattintson az IP elem hozzáadása ikonra **Add IP** vagy CTRL+I billentyűzet kombináció vagy kattintás az  ikonra.

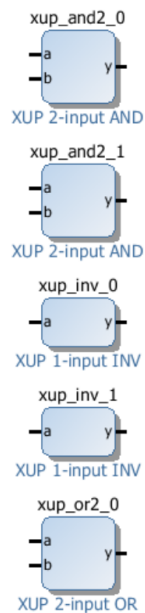


12. Ábra: IP elem hozzáadása

- 2.1.4. Az IP elemkönyvtár kereső ablakába írjuk be az egyes alkatrészek neveit, amelyeket keresünk sorrendben: **inv** (XUP 1-input INV), **and** (XUP 2-input AND), **or** (XUP-2 input OR), majd **2 x kattintás** a kiválasztott elemre vagy **ENTER** leütése.

2.1.5. Az elemek amelyeket a rajzra felteszünk az 13. Ábrán láthatók

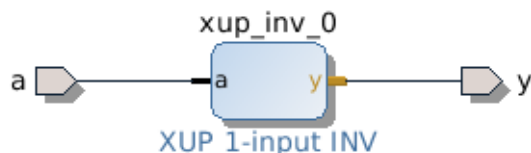
A további lépésekben befejezzük a rajzot (lásd 2. Ábra)



13. Ábra: Felhasznált IP elemek.

2.2. Modulokból álló rajz készítése

2.2.1. Válassza ki az xup_inv_0 bemeneti portját és **kattintás** egér jobb gombbal és külső csatlakozásként határozzuk meg a portot: **Make External**.



14. Ábra: Külső port csatlakozások

2.2.2. **Kattintás** az „a” portra és változtassa meg a port nevét „SW0”-ra (lásd 2. Ábra)

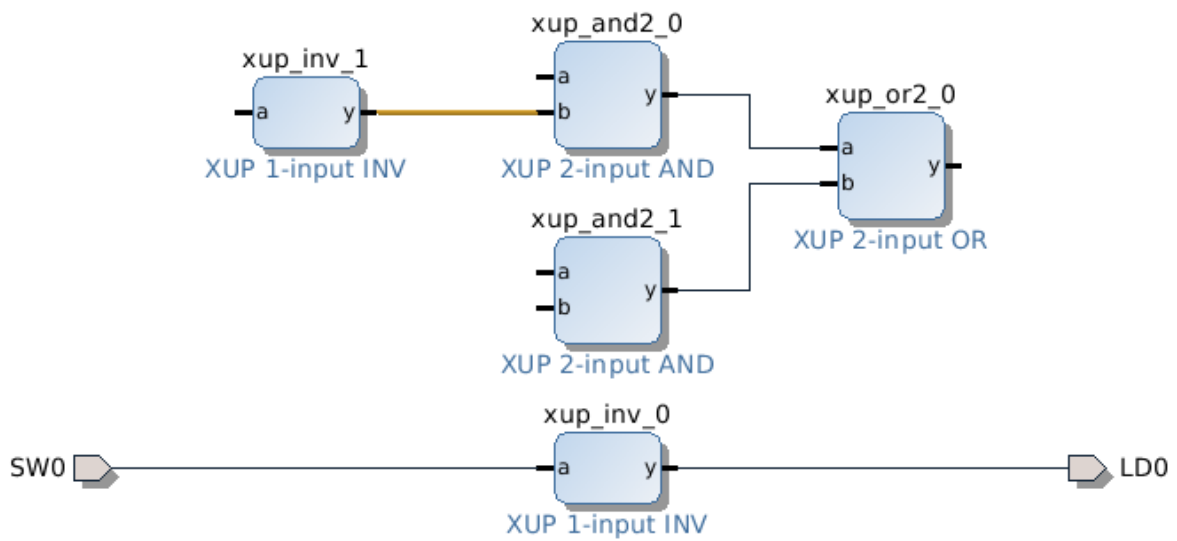
2.2.3. **Kattintás** az „b” portra és változtassa meg a port nevét „LD0”-ra (lásd 2. Ábra)

2.2.4. Rendezze az OR2 és AND2 kapukat az ábrának megfelelő elrendezésben - 15. Ábra.

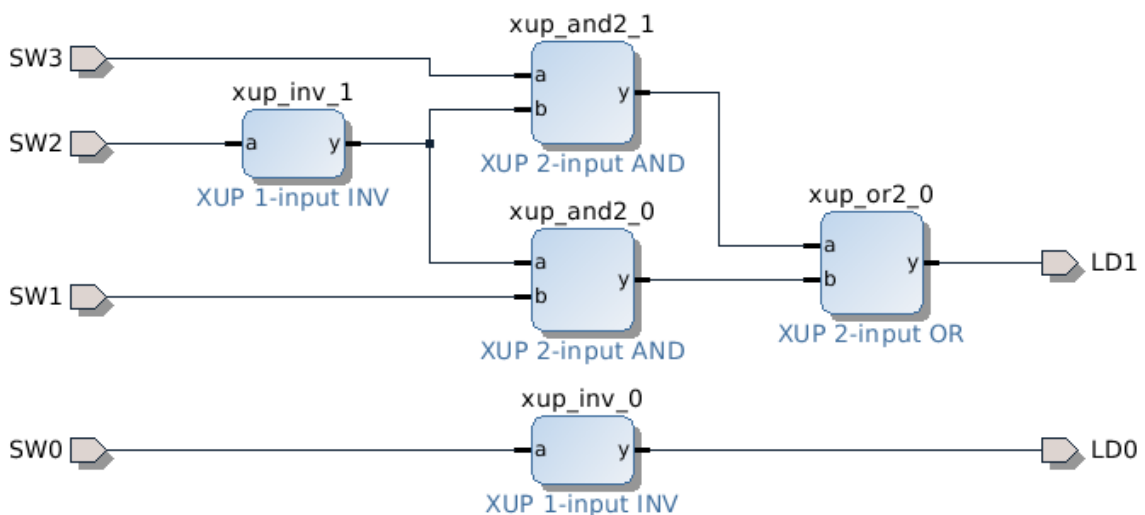
2.2.5. Csatlakoztassa a 15. Ábra-ának megfelelően az egyes elemeket.

Megjegyzés: Amikor az egyes elemek lábaihoz közelítjük az egeret az egérmutató átvált „ceruzára” – huzalozási üzemmódba. Lenyomott egér bal nyomógombbal megvalósítható a huzalozás. Kattintás egér jobb gombbal a portra „Block Pin Properties” megadhatók a port tulajdonságai -elnevezése, stb.

2.2.6. Hasonló módon mint az SW0 és LD0 esetében valósítsuk meg a külső csatlakozásokat az SW1(a), SW2 (a_1), SW3 (b) és LD2(y) lábakhoz (16. Ábra)

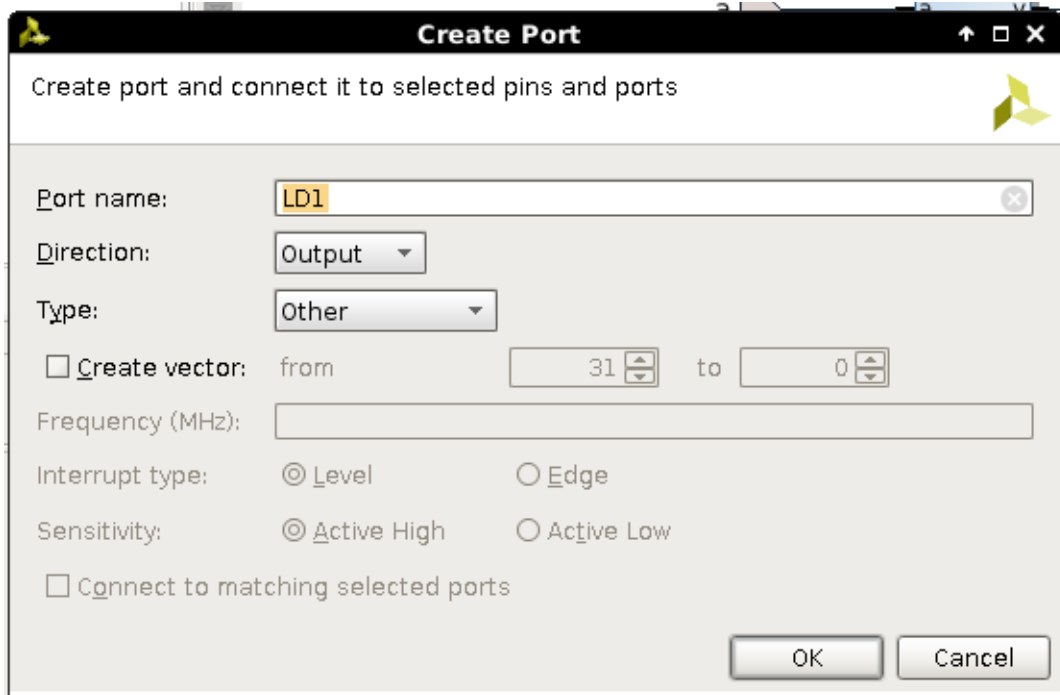


15. Ábra: A kapcsolási rajz egy félkész részlete

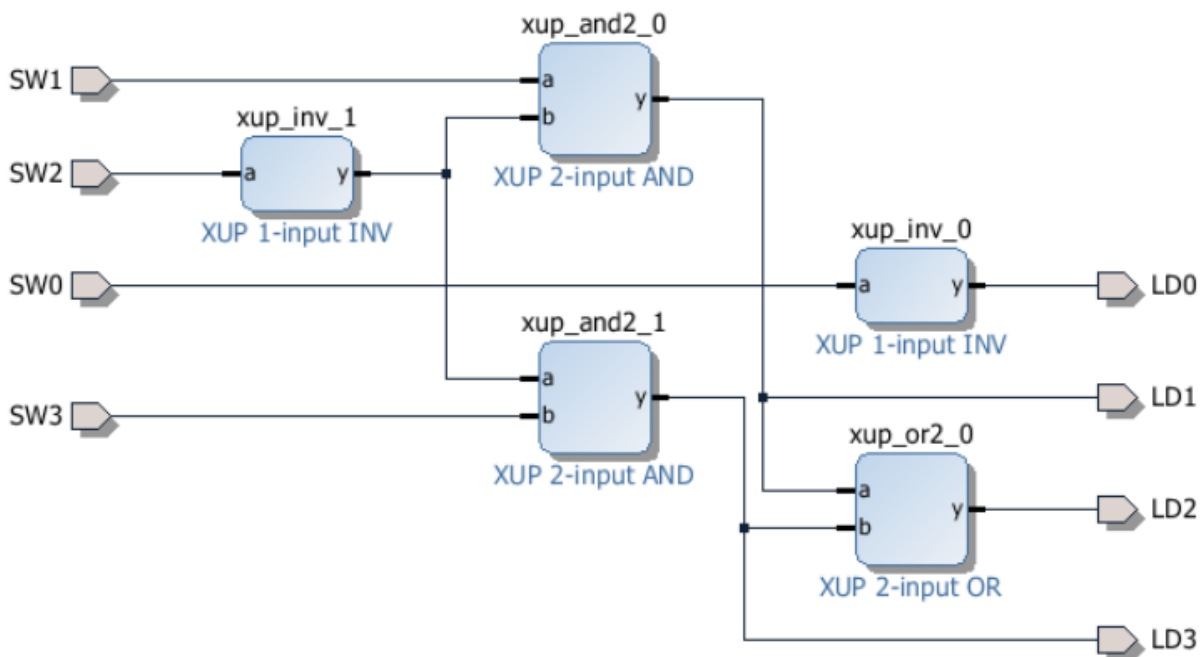


16. Ábra: Külső portok csatlakoztatása

- 2.6.1. **Kattintson** a rajzfelületen egér jobb gombbal és a felugró menüből válassza ki a port létrehozása parancsot: **Create Port**, hatására megjelenik a port létrehozása ablak (17. Ábra). A port neve legyen LD1, iránya kimenet (output).
- 2.6.2. Hasonlóan hozza létre az LD3 portot és csatlakoztassa megfelelő módon



17. Ábra: Port létrehozása ablak kép

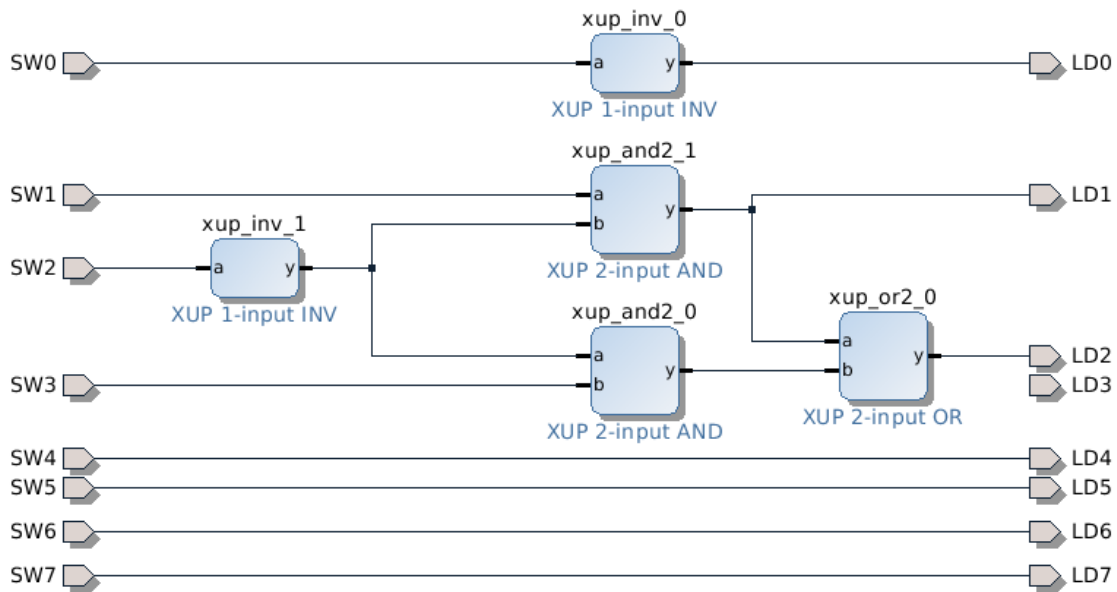


18. Ábra: Félig kész modul rajz

2.2.7. **Kattintás** a kapcsolás újra rajzolása ikonra 

2.3. Modul rajz befejezése (19. Ábra).

- 2.3.1. Hozza létre az SW4, Sw5, SW6, SW7 bemeneti portokat és az LD4, LD5, LD6, LD7 kimenteti portokat.
- 2.3.2. A huzalozó eszközt használva csatlakoztassa az egyes portokat az ábrán megadott módon. **Kattintás** Újrarajzolás. A befejezett rajz az 19. Ábrának megfelelő.
- 2.3.3. A Befejezett rajz elmentése: **File > Save Block Design** paranccsal történik.



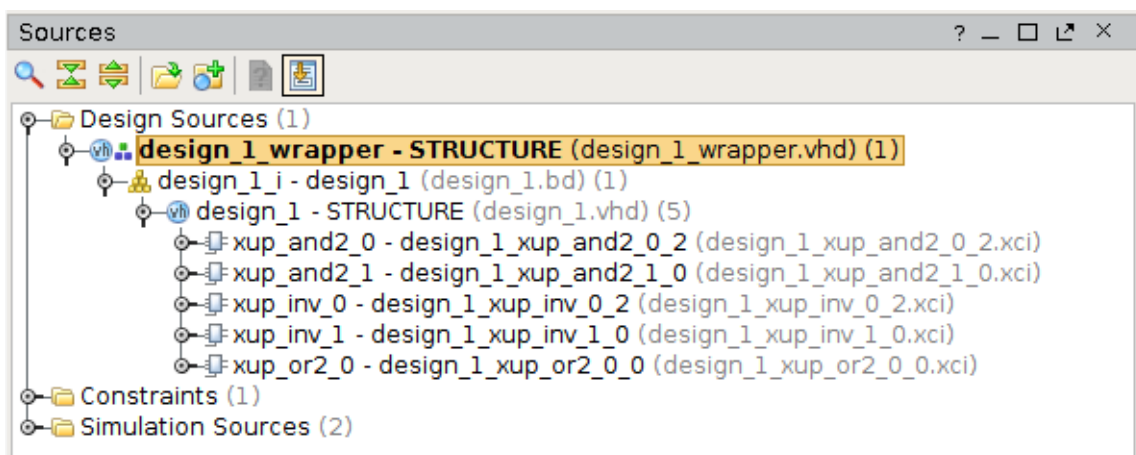
19. Ábra: Befejezett modul terv

3. HDL burok (wrapper) létrehozása és lábkiosztás 3. lépés

3.1. HDL burok létrehozás és hierarchia analízis

3.1.1. A forrás állományok ablakban kattintson egér jobb gombbal a modul rajzra „design1.bd” állományra és Válassza ki a HDL burok létrehozása parancsot: **Create HDL Wrapper**. A parancs hatására létrejön a HDL burok, amikor megjelenik az üzenet ablak (készítse el a HDL burkot?) Válassza az automatikus lehetőséget: **„Let Vivado manage wrapper and auto-update”**. Majd: **Kattintás: OK**

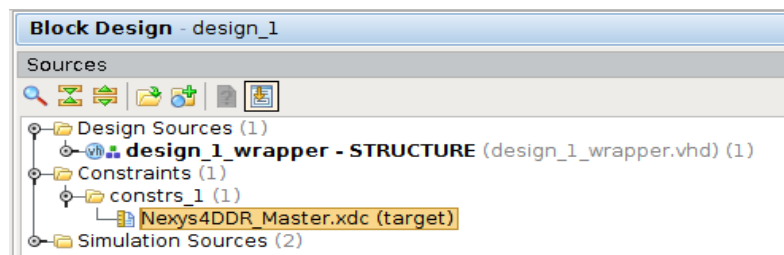
3.1.2. A parancs eredményeként két állományt hozott létre a fejlesztő rendszer. Az egyik állomány a HDL burok (design_1_wrapper.vhd). Ez az állomány a kapcsolási rajz modulként történő leírását valósítja meg. A másik állomány a (design_1.vhd) a kapcsolási rajz elemei közötti kapcsolatokat, alkatrészeket írja le (20. Ábra).



20. Ábra: HDL burok és terv hierarchia

3.2. Láb kiosztás és paraméterezés állomány hozzáadás

- 3.2.1. A *Flow Navigator* ablakban (Folyamat feldolgozás), *Project Manager* alcsoportban kattintson a forrás hozzáadás parancsra: **Add or Create Sources**,
- 3.2.2. Majd válassza ki a láb kiosztás és paraméterezés opciót: **Add or Create Constraints**
- 3.2.3. Adjuk hozzá a **Nexys4DDR_Master.xdc** (Nexys4DDR) paraméterező állományt amely szintén letölthető a következő címről:
<http://mzsola.iit.uni-miskolc.hu/M.edu/targyak/Nexys4DDR%20Mester%20XDC>
- 3.2.4. Kattintás: **OK**, majd a befejezéshez - Kattintás: **Finish**. (az eredmény az 21. Ábrán látható)



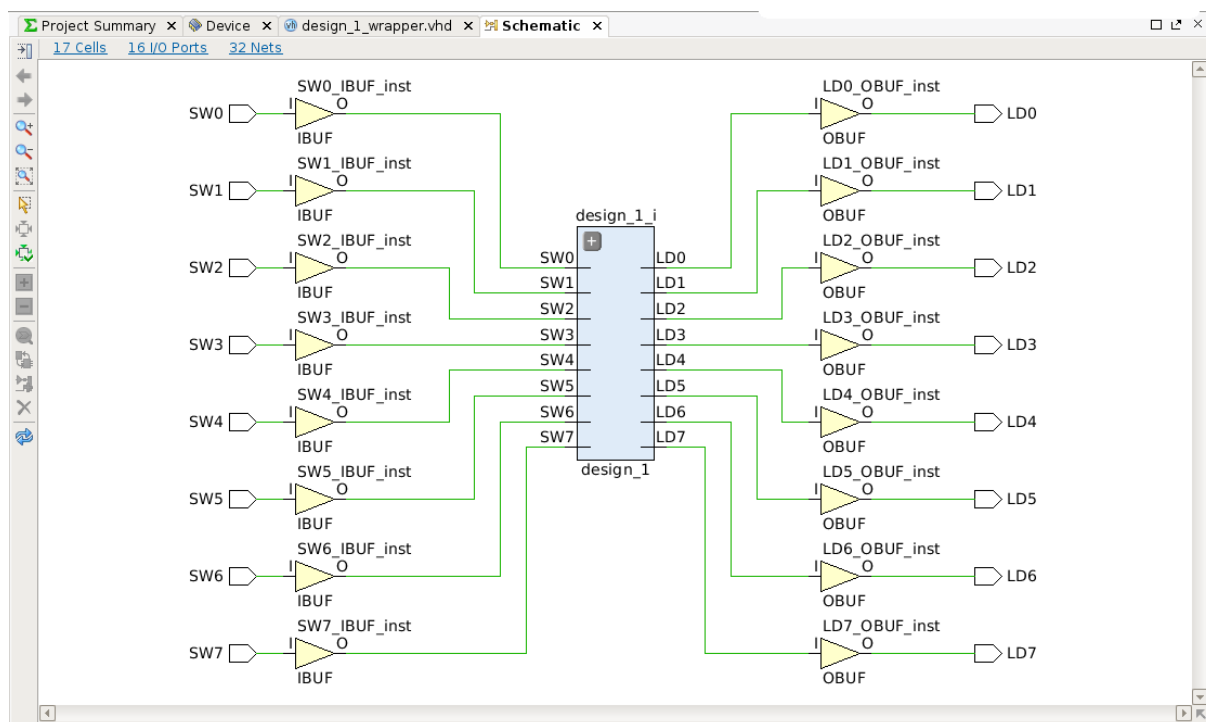
21. Ábra: Paraméterezés, láb kiosztás

- 3.2.5. A terv ablakban kétszeres kattintással nyissa ki a **Nexys4DDR_Master.xdc** állományt.
- 3.2.6. Keresse meg a kapcsolókra vonatkozó leírást **##Switches** – paraméterrel kezdődő 11-27. sorokat. A 12-19. sorok a technológiához történő illesztést (LVCMOS33) illetve a 20-27. sorok a láb kiosztást határozzák meg - SW[6:0].
- 3.2.7. A 46-62. sorok pedig a LED-ekre vonatkozó paraméterezést adják meg - LD[6:0].
- 3.2.8. Az SW7 és a LD7-re vonatkozó leírás célzottan hiányzik, annak érdekében, hogy megtanuljon más paraméterező módszert is.

3.3. RTL analízis

- 3.3.1. A *Flow Navigator* ablak, **RTL analysis** (analízis) csoport az **Open Elaborated Design** alcsoportban kattintson a **Schematic** (kapcsolási rajz) parancsikorra. Kattintson a mentés **Save**-re ha szükséges. A parancs hatására modul terv logikai nézetét mutatja a Vivado (22. Ábra).
- 3.3.2. Kattintson a rajzon a szimbólum + ikonjára, amelynek hatására megjelenik a modul mögötti kapcsolási rajz. Ha egy-egy szimbólum + ikonjára kattint végül láthatóvá válik a teljes terv hierarchia. Használjuk a nagyítót, hogy részleteiben is lássuk a kapcsolást.

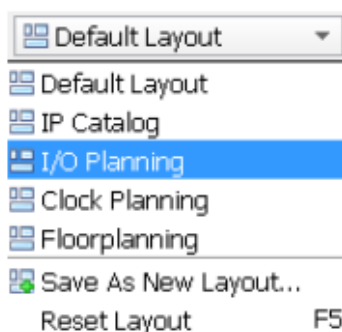
Megjegyzés: Figyelje meg, hogy néhány jel a bemenettől a kimenetig vezetékkel van összekötve!



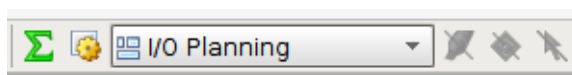
22. Ábra: Logikai kapcsolási rajz nézet.

3.4. Lábkiosztás megvalósítása chip szerkesztővel

- 3.4.1. A következő lépésekben a hiányzó lábkiosztást - kapcsoló (SW7) és LED (LD7) - valósítja meg.
- 3.4.2. Az RTL analízis megvalósításával, a többi szerkesztési mód is elérhetővé válik. Kattintson a szerkesztési mód kiválasztásra és a legördülő lehetőségek közül Válassza a lábkiosztás tervezési-szerkesztési módot - **I/O Planning layout** (lásd 23. Ábra és 24. Ábra.)

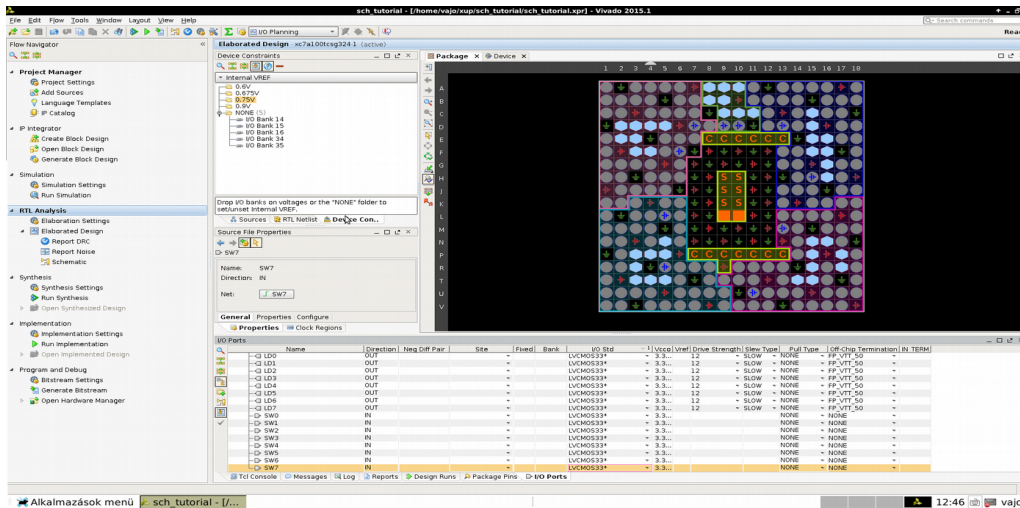


23. Ábra: I/O Planning szerkesztési mód választása



24. Ábra: I/O Planning szerkesztési mód

Megjegyzés: A szerkesztő ablakban megjelenik az áramkör tokozásának megfelelő nézet. Az I/O portok (ki/bemeneti lábak) táblázata megjelenik a tervező rendszer alsó részén (üzenetek ablak – Console View). A tervben használt lábakhoz (SW* és Ld*) rendelt portok (IOB) táblázatosan is felsorolásra kerültek és mindenik esetében többféle áramkörti technológia kiválasztása lehetséges. Ha az egeret a tokozás nézetben mozgatjuk az egyes lábaknál megjelenik a láb típusa és a hozzá tartozó lábszám (User IO, GND, VCCO..., stb.) - 25. Ábra, 27. Ábra.



25. Ábra: Tokozás és lábkiosztás nézet.

All ports (16)						
Scalar ports (16)						
LD0	Output		T8	<input checked="" type="checkbox"/>	34	LVCNMOS33*
LD1	Output		V9	<input checked="" type="checkbox"/>	34	LVCNMOS33*
LD2	Output		R8	<input checked="" type="checkbox"/>	34	LVCNMOS33*
LD3	Output		T6	<input checked="" type="checkbox"/>	34	LVCNMOS33*
LD4	Output		T5	<input checked="" type="checkbox"/>	34	LVCNMOS33*
LD5	Output		T4	<input checked="" type="checkbox"/>	34	LVCNMOS33*
LD6	Output		U7	<input checked="" type="checkbox"/>	34	LVCNMOS33*
LD7	Output					LVCNMOS18
SW0	Input		U9	<input checked="" type="checkbox"/>	34	LVCNMOS12
SW1	Input		U8	<input checked="" type="checkbox"/>	34	LVCNMOS15
SW2	Input		R7	<input checked="" type="checkbox"/>	34	LVCNMOS18
SW3	Input		R6	<input checked="" type="checkbox"/>	34	LVCNMOS25
SW4	Input		R5	<input checked="" type="checkbox"/>	34	LVCNMOS33
SW5	Input		V7	<input checked="" type="checkbox"/>	34	LVTTL
SW6	Input		V6	<input checked="" type="checkbox"/>	34	MOBILE_DDR
SW7	Input					

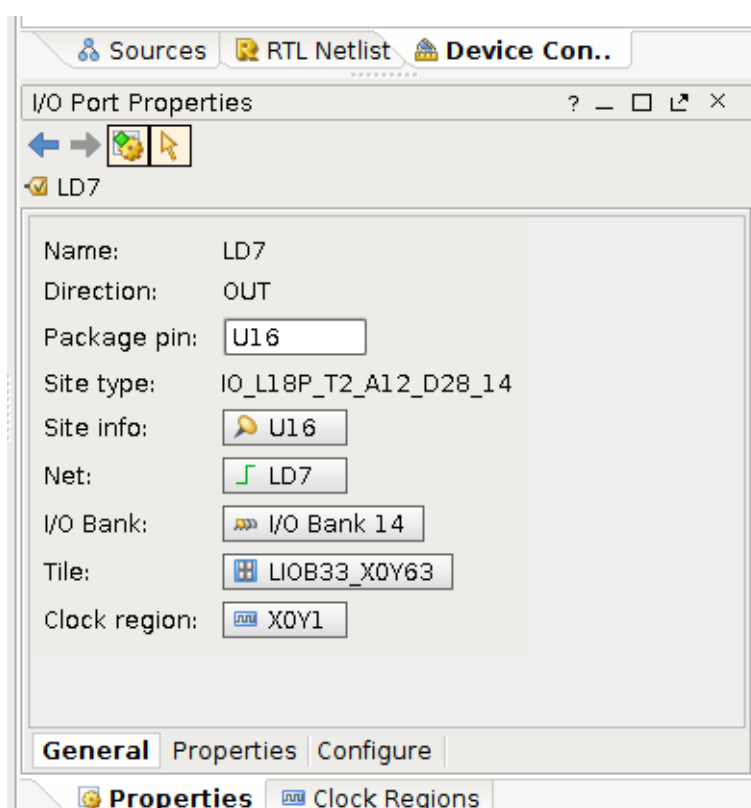
26. Ábra: Lábkiosztás és technológiához illesztés szerkesztő ablak.

- 3.4.3. Válassza ki az LD7 sort (26. Ábra) és a lábszám oszlopban - **Package Pin** - írja be az **U16**-os lábszámot, míg a technológiai szabványhoz való illesztés oszlopban - **I/O std** rendelje hozzá az LD7-es kimenetehz az LVCNMOS33 áramkörti technológiát.
- 3.4.4. Tcl parancsból is megoldhatjuk a paraméterezést. **Tcl** (eredetileg **Tool Command Language**) de úgy is ismert mint **Ticle** script. A Tcl Console ablakban írjuk be:

```
set_property package_pin V5 [get_ports SW7]
set_property iostandard LVCMOS33 [get_ports [list SW7]]
```

3.4.5. Figyelje meg a jel-láb I/O szabvány és I/O port változást. Megvalósíthatja a láb kiosztást úgy is, hogy az áramköri tokozás nézetben – **Package view** – Kattintson az SW7 jelre és egérrel áthúzzuk az IC tok megfelelő lábára. Úgy is megvalósítható a láb kiosztás, hogy kattint a táblázatban az LD7 sorra. Az IO Port ablakban megjelennek a lábhoz rendelt tulajdonságok. Ekkor ebben az ablakban szerkessze/rendelje a lábhoz a kívánt paramétereket (27. Ábra)

3.4.6. Mentse el a paraméter változásokat és a láb kiosztást.
Menü: **File** → **Save Constraints** állománynév: Nexys4DDR_Master.xdc

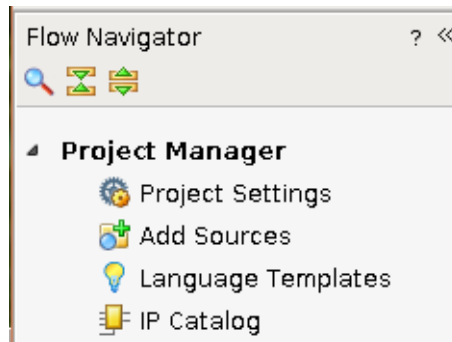


27. Ábra: LD7-es port paramétereinek beállítása ablak.

4. Áramköri Szimuláció XSim Szimulátorral (VHDL). 4. lépés

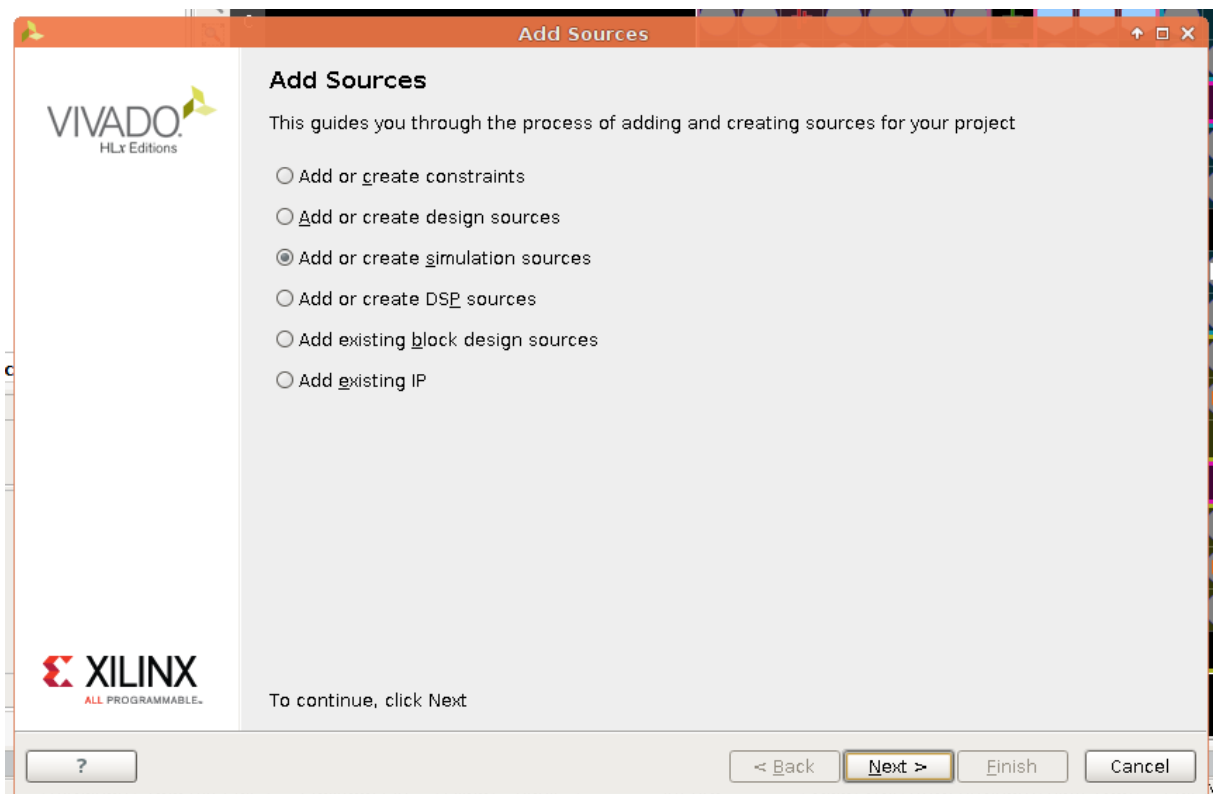
4.1. Tesztvektorok: tutorial_tb.vhd szimulációs állomány hozzáadása

4.1.1. A **Flow Navigator**, **Project Manager** parancsoknál kattintson az **Add Sources** – állomány hozzáadása parancsra.

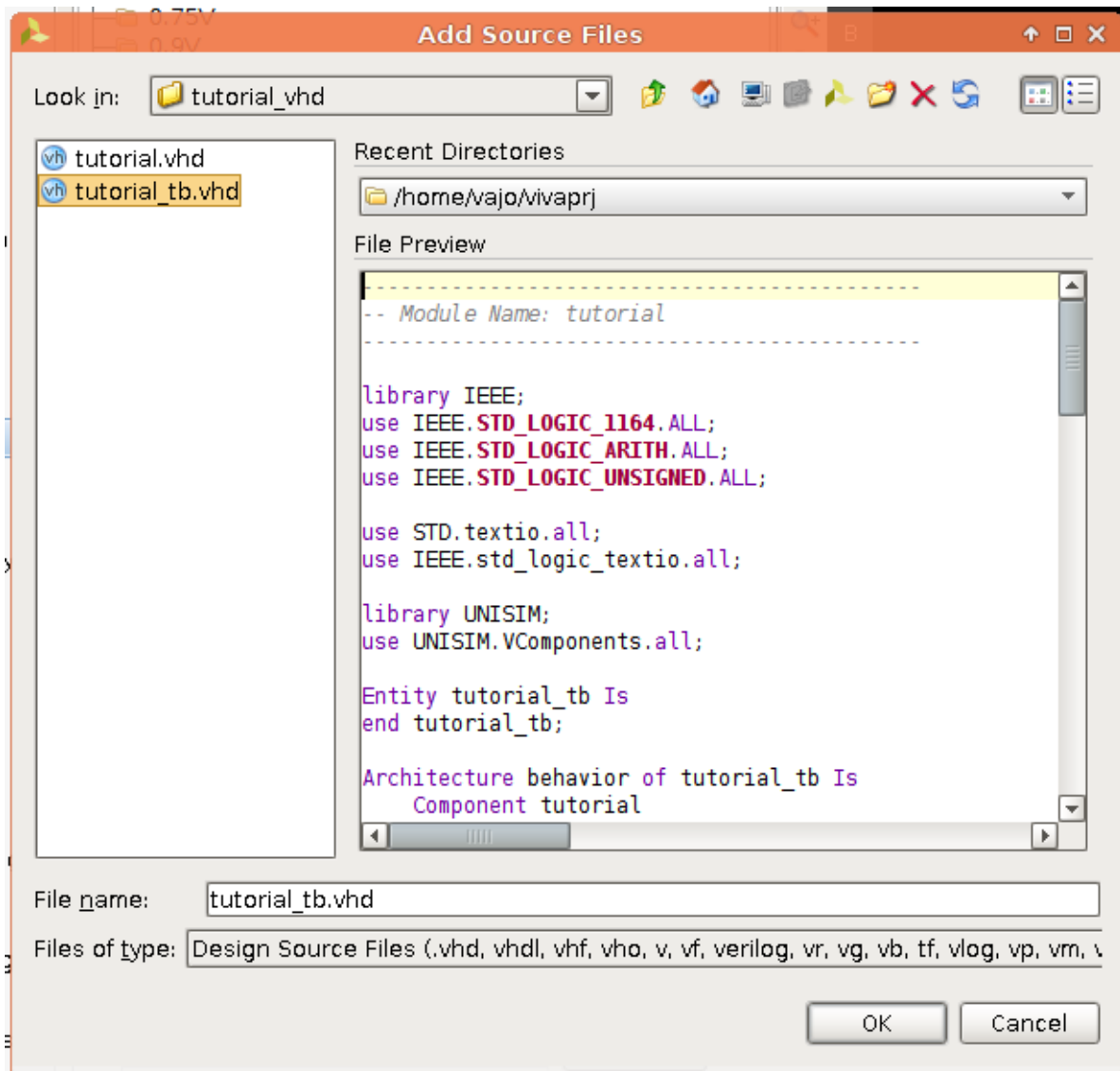
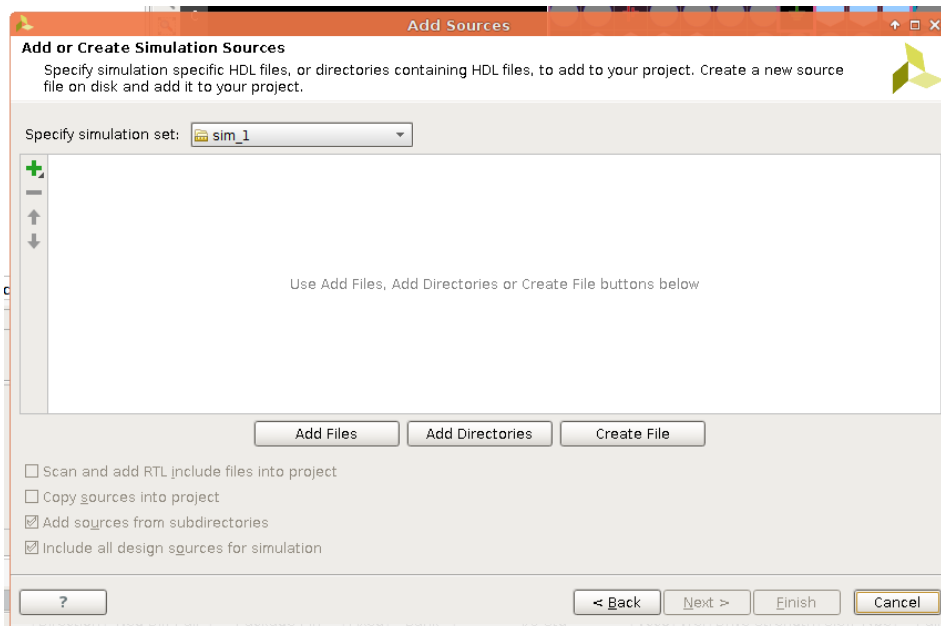


28. Ábra: Forrás hozzáadása parancs

- 4.1.2. Válassza ki a szimulációs állomány hozzáadása és létrehozása opciót. **Add or Create Simulation Sources**, majd kattintás: **Next** – lásd 28. Ábra, 29. Ábra, 30. Ábra, 31. Ábra.
- 4.1.3. Kattintson az állomány hozzáadás **Add Files** gombra. Majd keresse meg az /xup/ könyvtárban a tutorial_tb.vhd állományt és adjuk hozzá a tervhez. Kattintás: **OK**

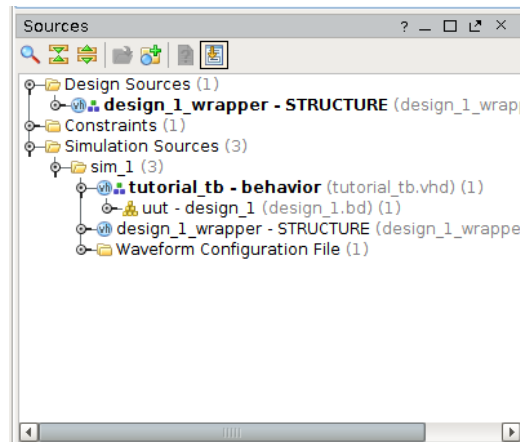


29. Ábra: Szimulációs állomány hozzáadása opció kiválasztása 1. lépés



4.1.4. Kattintás: **Finish**

4.1.5. Válassza ki a terv forrás állományok ablakban a szimulációs forrásokat, ezen belül a tutorial_tb.vhd állományt (32. Ábra)



32. Ábra: Szimulációs források

4.1.6. A tutorial_tb.vhd állomány a szimulációs források csoportjában található és a system_wrapper_1.vhd állomány automatikusan a tesztelendő alkatrész (uut – unit under test) lesz – lásd 32. Ábra. Használja az operációs rendszer fájl kezelőjét (Windows Explorer, Thunar, TotalComander, DoubleComander, mc) és ellenőrizze a terv könyvtár struktúrájában a sim_1 könyvtár tartalmát.

4.1.7. Nyissa meg a tutorial_tb.vhd állományt és vizsgálja meg tartalmát.

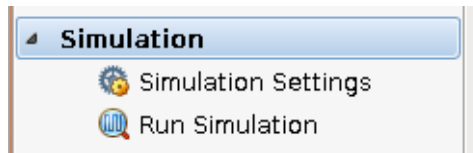
Figyelje meg, hogy a tesztelő állomány entitása sem paraméterekkel sem portokkal nem rendelkezik (üres). A tesztelendő állományt pedig alkatrészként határozza meg (uut). Elemezze az állományt és írja le hogyan történik a kapcsolók állapot változása?

4.2. Áramköri Szimuláció XSIM szoftverrel

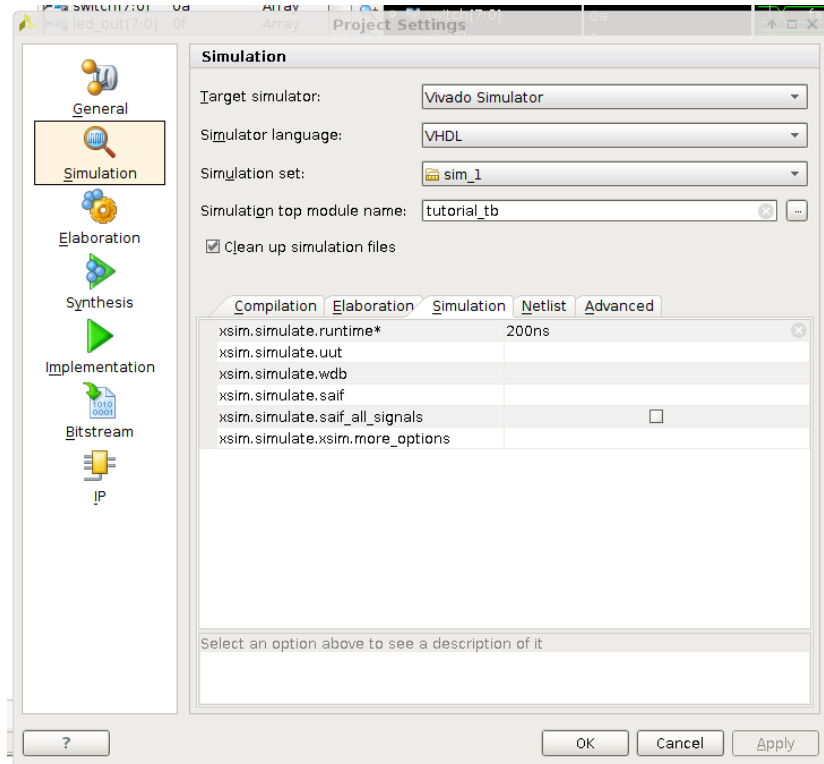
4.2.1. Válassza ki a szimulációs beállításokat (**Simulation Settings**) a Flow Navigátor ablakban - lásd 33. Ábra. A szimulációs paraméterek beállításához indítsa el a **Simulation Settings** parancsot a menüből.

4.2.2. Válassza ki a szimuláció tulajdonságait és a **Simulation** fülnél állítsuk be a szimuláció időtartamát 200 ns-ra, majd jóváhagyás az **OK** gombra történő kattintással.

4.2.3. Indítsuk el a szimulátort kattintással a **Run Simulation** paranccsal (33. Ábra)

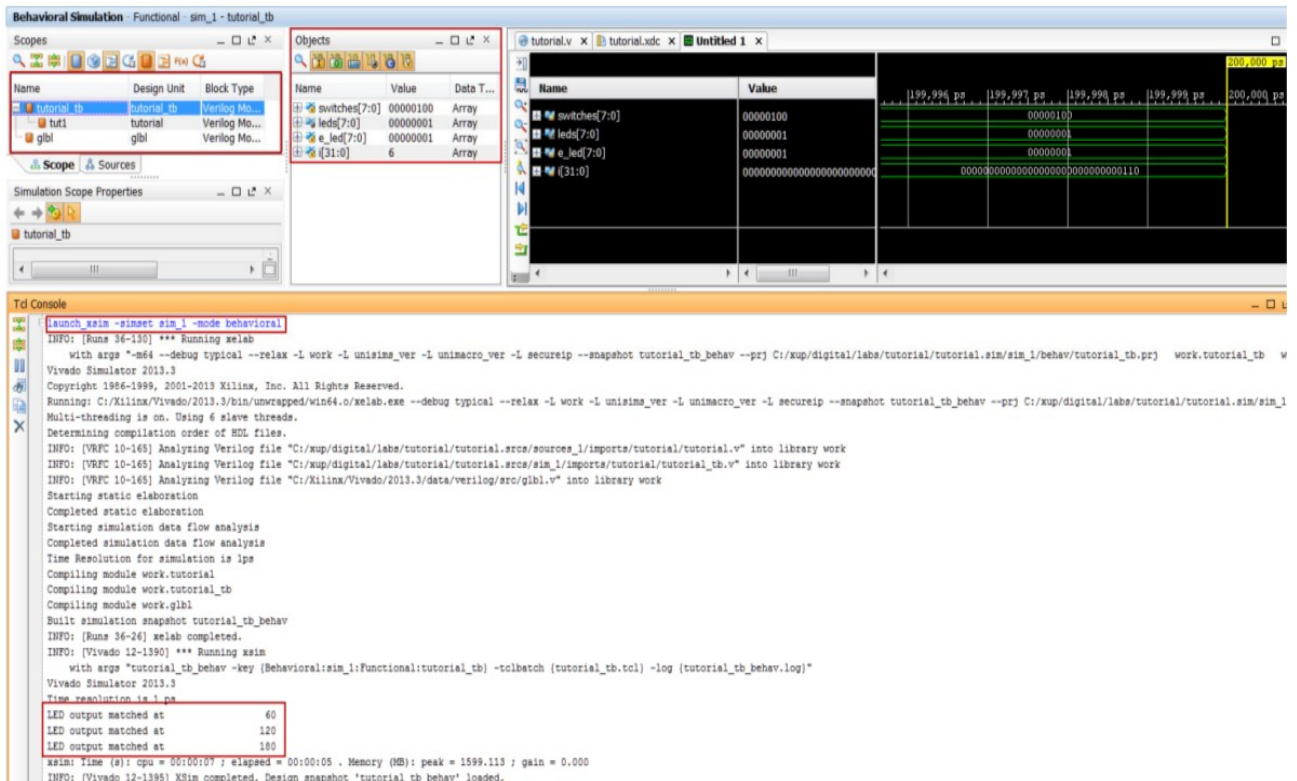


33. Ábra: Szimuláció menü.



34. Ábra: Szimulációs paraméterek beállítási kép.


- 4.2.4. Run Szimuláció → Run Behavioral Simulation – viselkedés vizsgálatával. A tesztelő és forrás állomány fordításával – feltételezve, hogy az állományok hibamentesek – elindul az XSIM szimulátor és a szimuláció eredménye is megjelenik (35. Ábra).

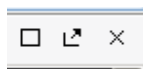


35. Ábra: A feladat szimulációjának eredménye

A szimuláció eredményét többféle módon ábrázolja a szoftver: (i) felsorolja a hierarchiában szereplő és a globális jeleket amelyek a teszt állomány részei -testbench; (ii) felsorolja a legfelső szintű jeleket; (iii) ábrázolja grafikus képen (virtuális szkóp) a jelek és globális jelek időbeni változását; (iv) Tcl ablak, ahol a szimulátor parancsok és az azokra adott válaszok szövegesen láthatók.

Megjegyzés: (A) A szimulátor ellenőrzi a „test-bench” állományt és annak függvényében ábrázolja az eredményt, közvetlenül a szimulátor elindítása után. **(B)** A tutorial.sim könyvtárat és annak tartalmát a szimulátor hozza létre.

4.2.5. Az idődiagramon használhat nagyítót , hogy növelje a felbontást. A „szkóp ablak” kiemelhető a keretrendszerből és vissza is helyezhető



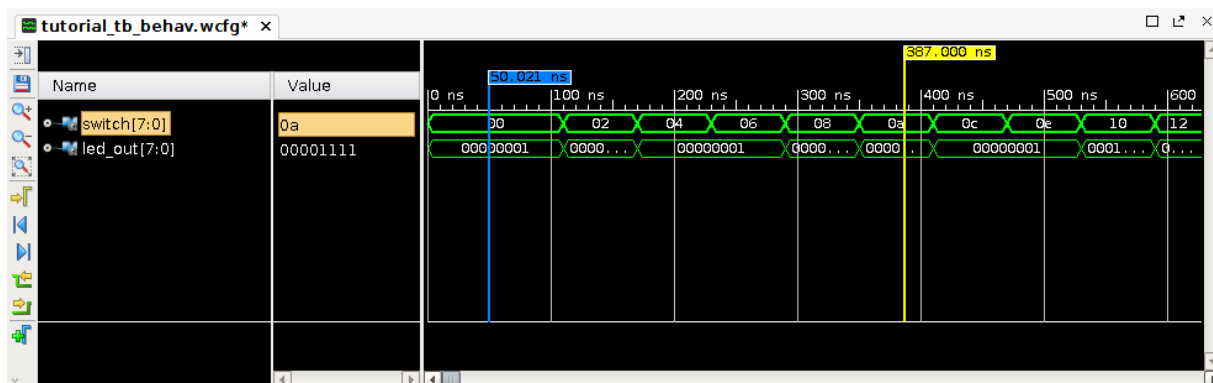
36. Ábra: Kiemelés - Float



37. Ábra: Horgonyzás - Dock

4.3. Adatformátum kijelzésének megváltoztatása

- 4.3.1. Az sínrendszerbe szervezett jelek ábrázolási adatformátuma, megváltoztatható ha kattint egér jobb gombbal a jelre és kiválasztja a „*Default Radix*” vagy a „*Radix*” parancsot. Itt megadható, a hogy a jelet milyen számrendszerben ábrázolja a szimulátor.
- 4.3.2. Például az 38. Ábra switch[7:0] jele hexadecimális, míg a led_out[7:0] bináris számrendszerben ábrázoltuk.



38. Ábra: Jelek különböző ábrázolási számrendszerben.

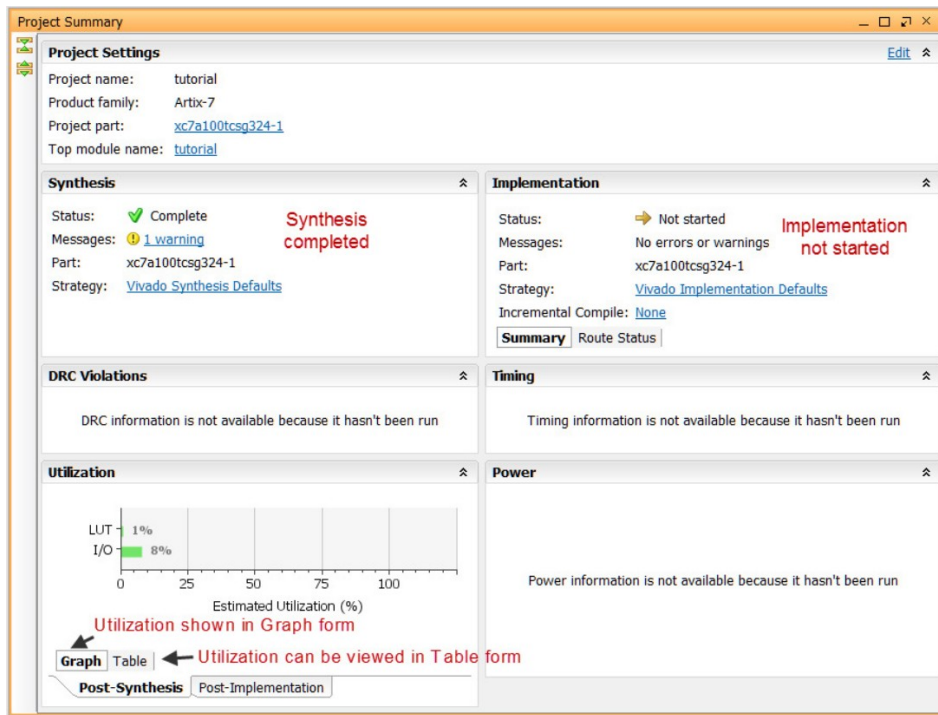
- 4.3.3. Kilépés az XSIM szimulátorból: **File** → **Close Simulation** parancssal.

5. A terv szintézise

5. lépés

5.1. A Vivado szintetizáló segítségével lefordítjuk a tervet és elemezzük a kapott eredményt.

- 5.1.1. A Flow Navigator menüben kattintson a **Run Synthesis** parancsra a Synthesis menüben. A szintetizáló program teljes hierarchiájában feldolgozza a tervet. Amikor a művelet befejeződik a megjelenő üzenet ablakban válassza a szintetizált terv vizsgálatát **Open Synthesized Design** és kattintás: **OK** gombra. Ennek eredményeként megjelenik az áramkör szerkesztő ablak. Válassza a **Project Summary** fület a szintézis eredményeinek elemzése érdekében (39. Ábra).



39. Ábra: Szintetizált tervben felhasznált LUT és I/O erőforrások.

- 5.1.2. Kattintson a táblázat nézetre, hogy a rendszer táblázatos formában jelentesse meg a felhasznált erőforrásokat - 39. Ábra. Az eredmény a 40. Ábrán látható, amint azt az ábra is mutatja 5 darab keresőtábla (LUT) és 16 darab IO be-kimeneti lábát használt fel.

Resource	Estimation	Available	Utilization %
LUT	5	63400	0.01
IO	16	210	7.62

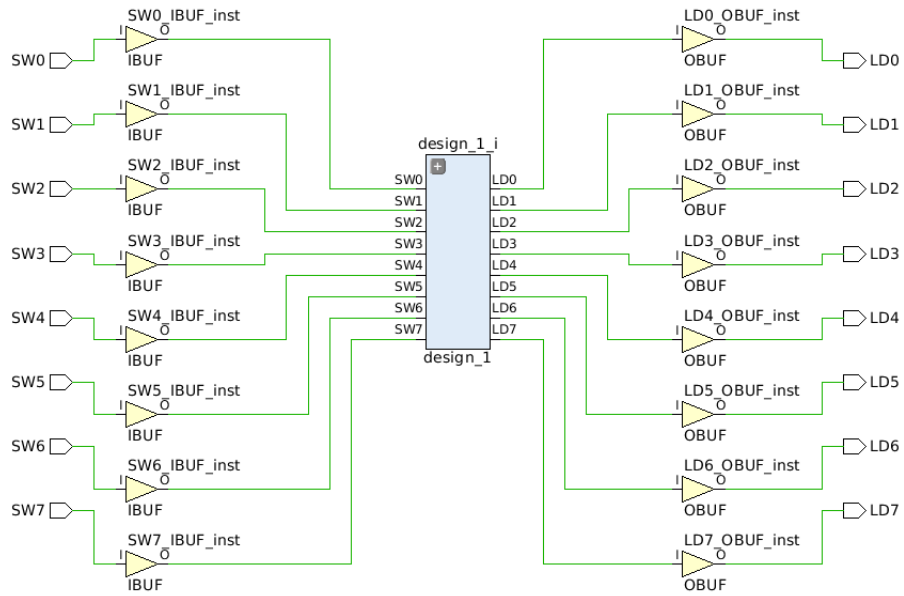
Utilization - Post-Synthesis

Graph **Table**

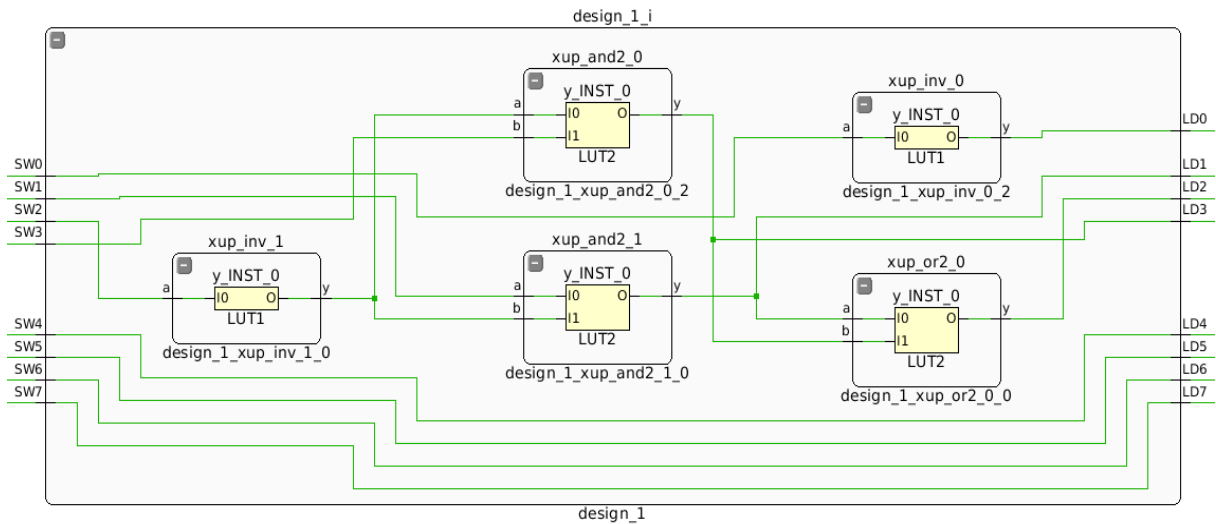
Post-Synthesis Post-Implementation

40. Ábra: Felhasznált erőforrások - táblázatos nézet

- 5.1.3. Kattintson az **Flow Navigator** ablakban a **Synthesis** → **Synthesized Design** menüben a kapcsolási rajz nézetre **Schematic**-lásd 41. Ábra. Az ábrán kattintson a design_1_i modul „+” jelére, hogy láthatóvá tegye a további kapcsolási rajz részleteket (42. Ábra). Figyelje meg, hogy a keretrendszer a be-kimeneti lábához automatikusan csatlakoztatta az IBUF, OBUF meghajtható puffereket.



41. Ábra: Szintetizált terv kapcsolási rajza



42. Ábra: A kapcsolási rajz részletes nézete

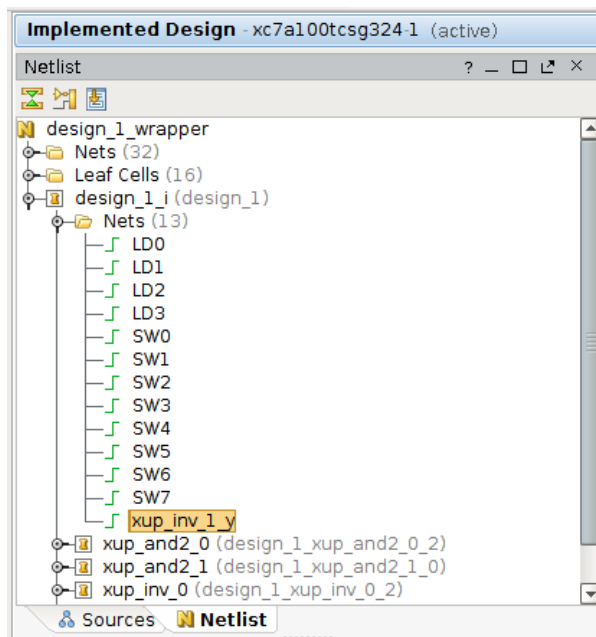
A fenti ábrán megfigyelhető, hogy a logikai kapukat az FPGA keresőtáblába helyezte el a szintézis. Egy fájlkezelő ablak segítségével megtekinthető a terv könyvtárban a *tutorial.runs* könyvtár tartalma, amely a szintézis eredményét tartalmazza.

6. A terv megvalósítása - „implementálás”

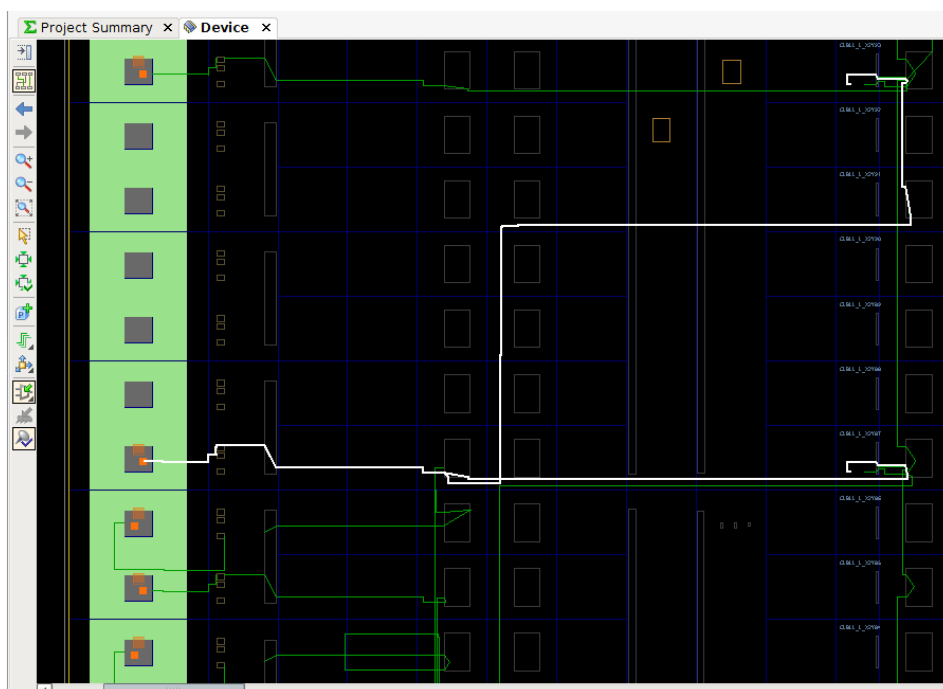
6. lépés

6.1. A terv megvalósítása a Vivado keretrendszer alap beállításával és a megvalósítás elemzése.

- 6.1.1. A **Flow Navigátor** ablakban a **Implementation** menüben kattintással indítsa el a **Run Implementation** – megvalósítás parancsot. A megvalósítás, áramkört huzalozás, feltérképezés befejezése után az üzenetablakban látható a művelet befejezése üzenet – **Implementation Completed**. A felugró ablakban válassza a megvalósítás megtekintése opciót – **Open implemented design**. Kattintás: **OK**. Ugyanakkor zárja be a szintézis nézet ablakot. 44. Ábra



43. Ábra: Vezeték kiválasztása a huzalozási listából



44. Ábra: Megvalósított huzalozott terv kép - részlet

6.1.2. A huzalozási listából válasszon ki egy vezetéket (43. Ábra), amelyet aztán a magvalósított áramkörben kövessen (44. Ábra).

6.1.3. Ha bezárja a megvalósított terv ablakot vagy átkapcsol a terv állapot ablakra - **Project Summary tab** (Default Layout nézet), akkor megtekinteti a felhasznált erőforrásokat hasonlóan, mint ahogyan a szintézis során is tette (5.1.2). Vizsgálja meg az eredményt! Látható, hogy a valóban felhasznált LUT elemek száma nem egyezik az előző eredménnyel – szintézis, optimalizálás, megvalósítás; a felhasznált IOB-k száma 16. Megjegyezzük, hogy az „implementációt” az alapértelmezett paraméterekkel valósítottuk meg – ne állítson időzítési korlátokat.

6.1.4. A Fájltrekezelővel nyissa ki a terv könyvtárakat és vizsgálja meg az impl_1 könyvtár tartalmát, ahol különböző jegyzőkönyvek találhatók a megvalósításról. Ezek a tervező rendszerben is megtekinthetők, amennyiben az **Implementation** → **Implemented Design** egyes al-parancsaira kattint. Így megtekintheti az áramkör által felvett villamos teljesítményt, az egyes zónák hőmérsékletét, időzítések állapotát, szerkeszthetők a fordítási paraméterek, stb. Megjegyezzük, hogy a tervünk egy kombinációs hálózat ezért egyetlen regisztert sem használt fel a rendszer.

7. Időarányos Szimuláció

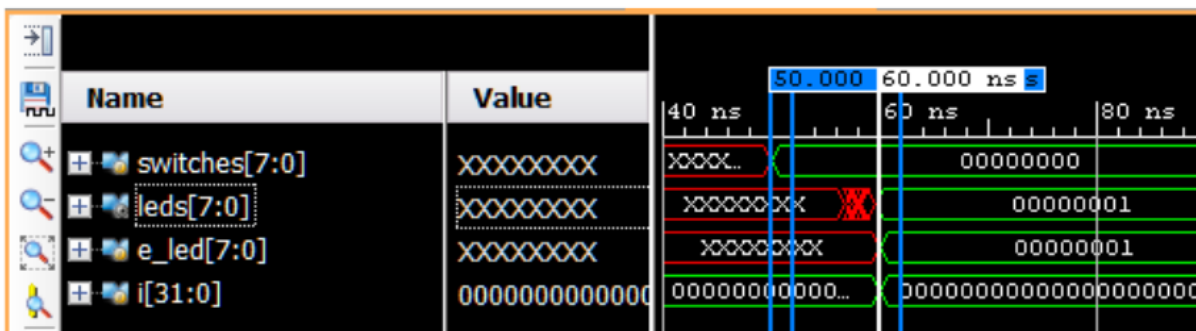
7. lépés

7.1. Időarányos szimuláció megvalósítása.

7.1.1. Kattintson a **Flow Navigator** → **Simulation** menüjében a **Run Simulation** parancsra. Válassza a **Run Simulation** → **Run Post-Implementation Timing Simulation** opciót. Az XSim szimulátor elindul és a *tutorial_tb* tesztelő állományt fogja futtatni. A fájlkezelővel vizsgálja meg, hogy a terv könyvtárban a *tutorial.sim* → *sim_1* → *impl* könyvtárat a Vivado létrehozta és a könyvtár tartalmazza az időarányos szimulációhoz szükséges állományokat.

7.1.2. Ha az XSIM elindult és megjelent a szimuláció eredménye a szkóp ablakban, akkor kattintson a nagyítás ikonra (**Zoom Fit**)

7.1.3. Kattintás egér jobb gombbal az 50 ns pozícióba (a switch bemenet) és Válassza a **Select Markers** → **Add Marker** parancsot. Hasonlóan szúrjon be egy markert az 55 ns és a 60 ns környékén is.



45. Ábra: Időarányos szimuláció eredménye.

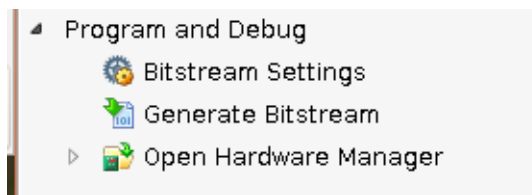
7.1.4. Zárjuk be a szimulátort **File** → **Close Simulation** paranccsal

8. Bittérkép létrehozása és funkcionális ellenőrzés

8. lépés

8.1. Csatlakoztassa a Nexys4DDR kártyát és kapcsolja be a tápfeszültséget. Bittérkép generálása és letöltés a hardverbe – FPGA programozás.

8.1.1. A Flow Navigator ablakban kattintson a bittérkép létrehozás parancsra **Program and Debug** → **Generate Bitstream**. A bittérkép létrehozása után az üzenetablakban megadott három lehetőség közül válassza a hardware manager elindítása opciót - **Open Hardware Manager**.

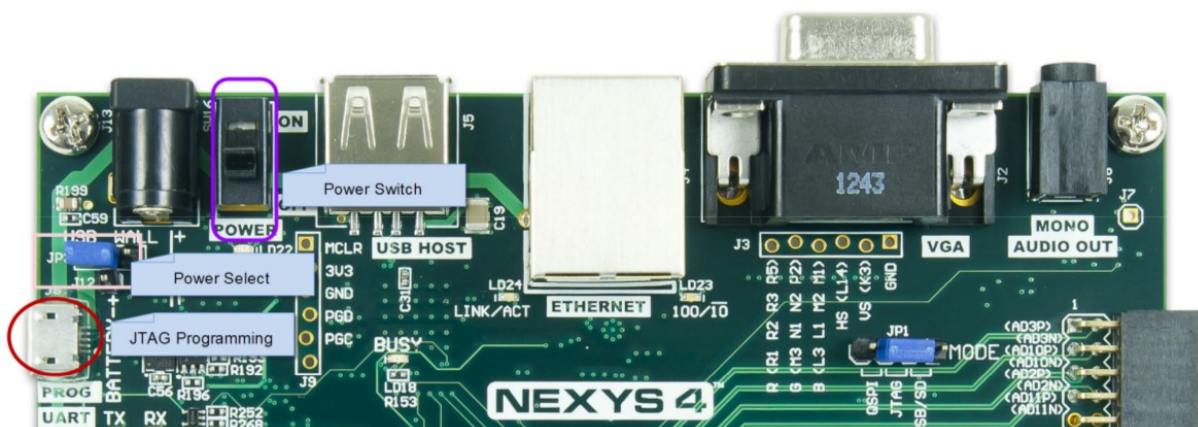


46. Ábra: Bittérkép generálása parancs kiválasztása

A bittérkép generátor létrehozta a design_1_wrapper.bit állományt, amely a terv könyvtárban az *tutorial.run* → *impl_1* alkönyvtárban található.

8.1.2. Bizonyosodjon meg róla, hogy a tápfeszültség jumper „tápfeszültség az USB-n keresztül” pozícióban van és a micro-USB kábelt csatlakoztatta a PC és a kártya között.

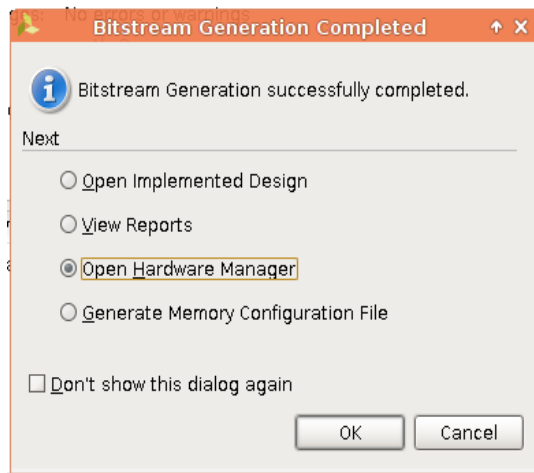
Megjegyzés: A tápegységet nem szükséges csatlakoztatni, a kártya táplálása az USB kábelen keresztül is megvalósítható.



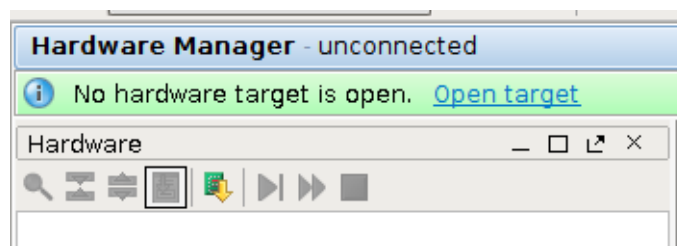
47. Ábra: Csatlakoztatás a PC-hez micro-USB-vel és tápfeszültség jumper pozíciója

8.1.3. Kapcsolja be a tápfeszültséget a kártyán.

8.1.4. válassza ki az üzenetablakban az **Open Hardware Manager** – hardver kezelés - opciót és kattintás **OK** lásd 48. Ábra. A hardver kezelőből elindíthatja a letöltést.

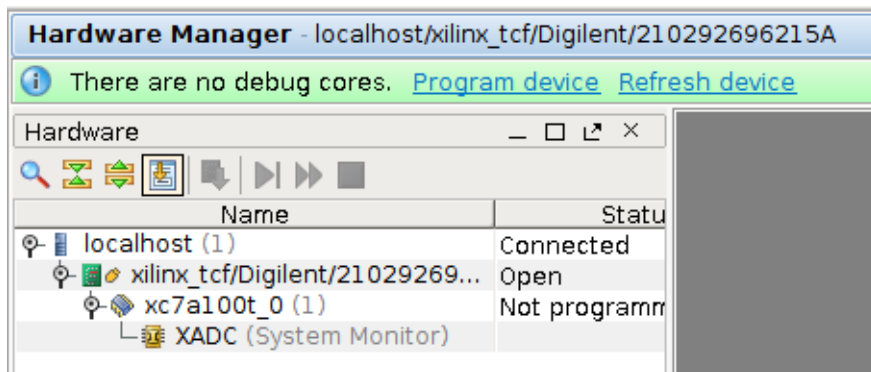


48. Ábra: Bittérkép sikeres létrehozása üzenet ablak.



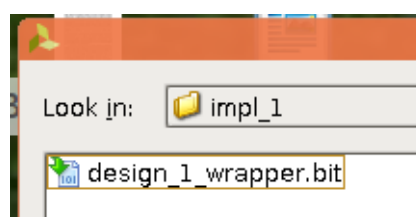
49. Ábra: Hardver manager ablak.

- 8.1.5. Kattintson az **Open target** – kapcsolat a célhardverrel – parancsra. válassza az automatikus kapcsolódás opciót.

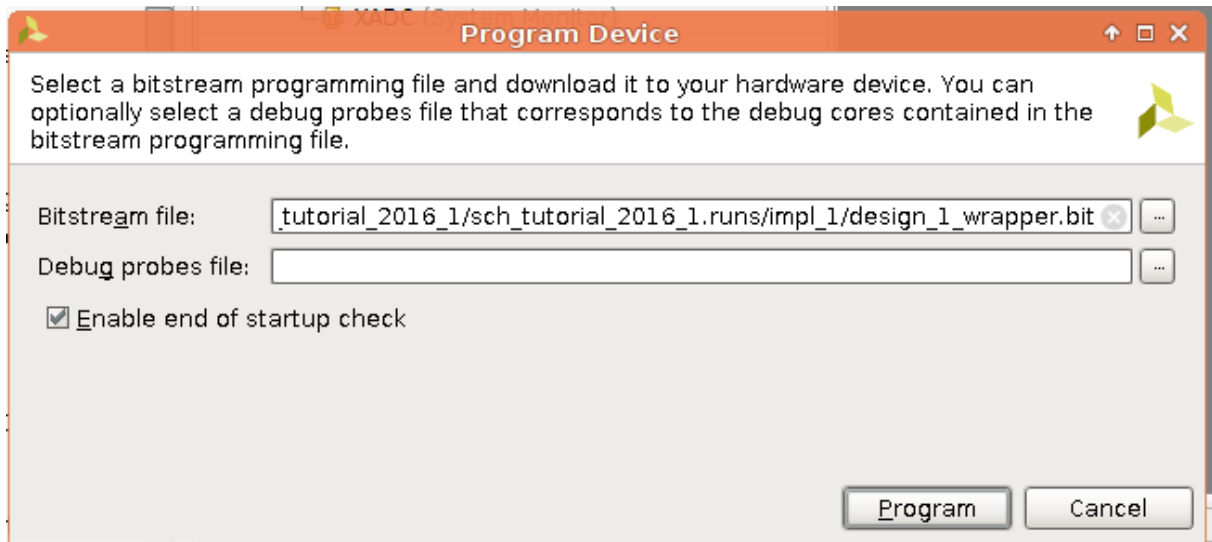


50. Ábra: Kapcsolat a célhardverrel.

- 8.1.6. Kattintson az **xc7a100t_0** FPGA áramkörre és válassza a **Program device** - eszköz konfigurálás parancsot (50. Ábra).
- 8.1.7. Válassza ki a felugró ablak által ajánlott `desig_1_wrapper.bit` állományt, amely a tervezési könyvtár `impl_1` könyvtárában található (51. Ábra).



51. Ábra: Létrehozott bittérkép állomány az `impl_1` könyvtárban.



52. Ábra: FPGA programozás parancs indítás

- 8.1.8. Kattintson a programozás parancsra
- 8.1.9. Ellenőrizze a kártyán az áramkör működését a kapcsolók ki és bekapcsolásával.
- 8.1.10. Zárja be a hardverkezelő ablakot. Kattintás: **OK** – bezárás.
- 8.1.11. Kacsolja ki a kártya tápfeszültségét.
- 8.1.12. Zárja be a Vivado programot: **File** → **Exit**. Kattinás: **OK**

9. Következtetések

A Vivado fejlesztő rendszer az FPGA fejlesztés minden fázisát támogatja. A tervet az XUP IP alkatrész könyvtárelemeiből valósította meg (IPI elemek és paraméterezés). Viselkedés és időarányos szimulációval vizsgálta az áramkör működését. Szintézis, huzalozás és bittérkép generálás után a Nexys4 kártyát konfigurálta a generált bitfolyam állománnyal. Megismerte az FPGA fejlesztés főbb lépéseit.

Irodalom

- [1.] *** Xilinx Vivado segédlet források:
http://mazsola.iit.uni-miskolc.hu/DATA/storages/files/_akfnfpfmfm_eaDDBJ.zip
- [2.] *** Digilent Nexys 4 DDR:
http://mazsola.iit.uni-miskolc.hu/DATA/storages/files/_doBfICA_eqTTUR.pdf
- [3.]