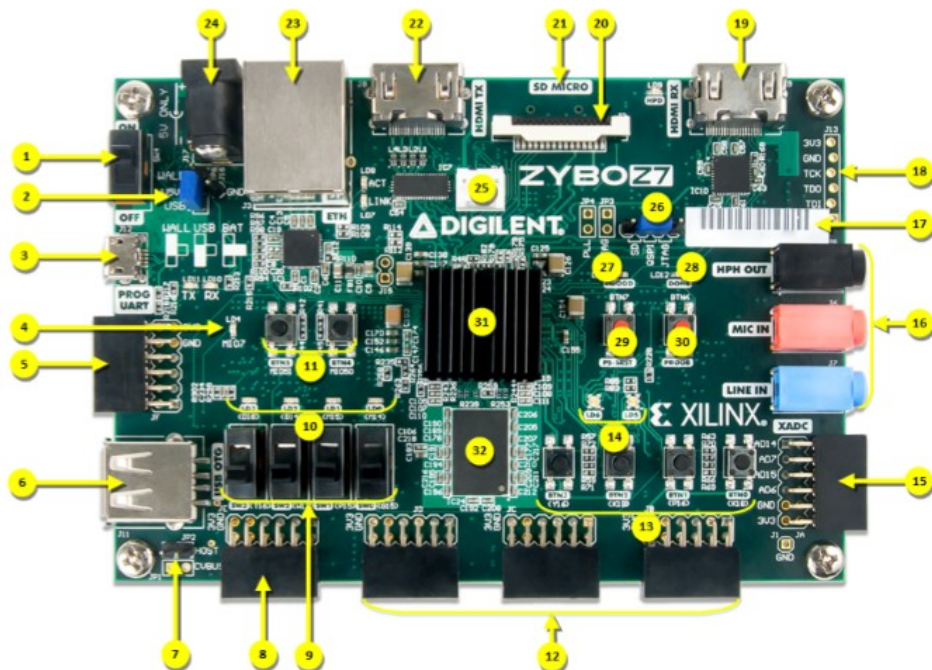


Vežba 7 - Zybo razvojan platforma i rad sa Vivado IP i Vitis alatima

Uvod

Ovaj materijal posvećen je Zybo razvojnoj platformi kao i alatima koji omogućavaju njeno programiranje. Sledeća slika prikazuje Zybo platformu kao i sve njene periferije:



*Zybo Z7-20 pictured

Callout	Description	Callout	Description	Callout	Description
1	Power Switch	12	High-speed Pmod ports *	23	Ethernet port
2	Power select jumper	13	User buttons	24	External power supply connector
3	USB JTAG/UART port	14	User RGB LEDs *	25	Fan connector (5V, three-wire) *
4	MIO User LED	15	XADC Pmod port	26	Programming mode select jumper
5	MIO Pmod port	16	Audio codec ports	27	Power supply good LED
6	USB 2.0 Host/OTG port	17	Unique MAC address label	28	FPGA programming done LED
7	USB Host power enable jumper	18	External JTAG port	29	Processor reset button
8	Standard Pmod port	19	HDMI input port	30	FPGA clear configuration button
9	User switches	20	Pcam MIPI CSI-2 port	31	Zynq-7000
10	User LEDs	21	microSD connector (other side)	32	DDR3L Memory
11	MIO User buttons	22	HDMI output port	* denotes difference between Z7-10 and Z7-20	

Slika 1. Zybo razvojna platforma

Centralni deo ove razvojne platforme je Zynq 7000 SoC (XC7Z010-1CLG400C) čip koji se deli na dve celine:

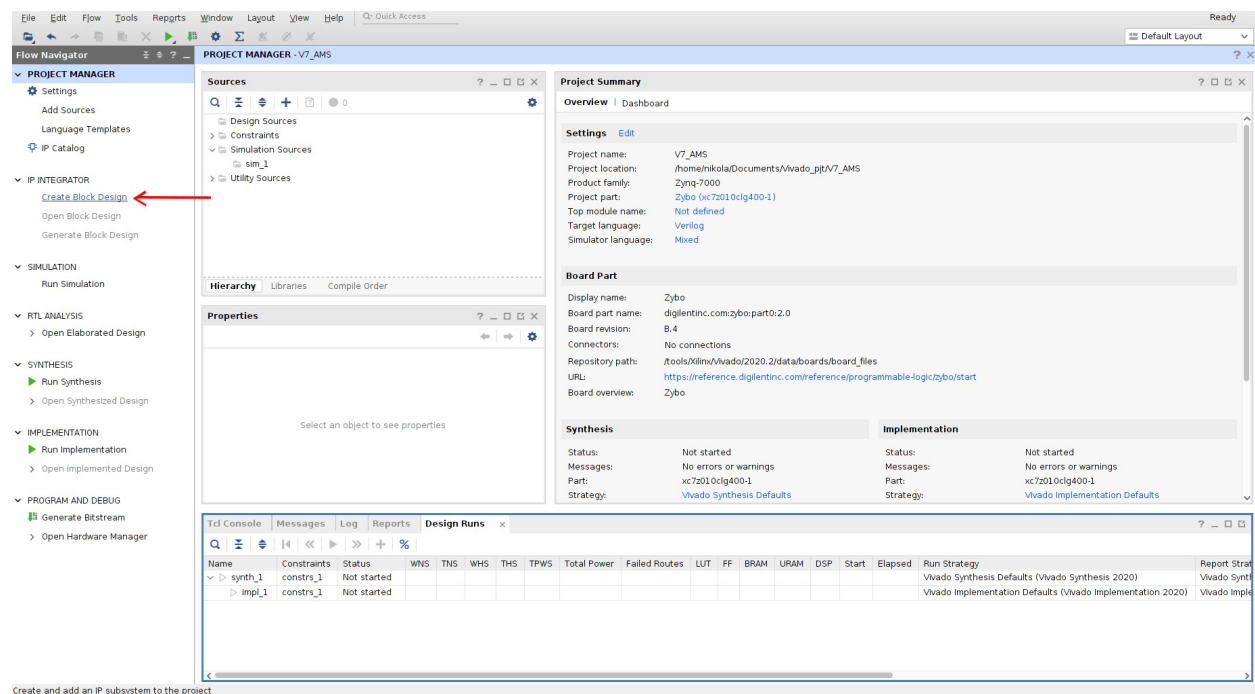
1. *Processing System* koji je fabrikovan kao ASIC i sadrži:
 - dvojezgarni Cortex-A9 processor @ 650Mhz
 - Svaki core ima po 32KB LVL1 Instr i 32KB LVL1 Data Cache
 - Ujedinjeni LVL2 Cache od 512KB
 - Interfejs ka off-chip 32-bitnoj DDR3 memoriji sa kapacitetom 512MB i propusnim opsegom od 1050Mbps.
2. FPGA logika koja pripada Xilinx 7-series familiji čipova i koja sadrži sledeće ćelije:
 - 4,400 logic slices, each with four 6-input LUTs and 8 flip-flops
 - 240 KB of fast block RAM
 - 80 DSP slices
 - Internal clock speeds exceeding 450MHz

U nastavku će biti objašnjeno kako se programira procesorski sistem, dok će u narednim materijalima biti objašnjeno programiranje FPGA celine kao i način komunikacije između njih.

Vivado IP integrator

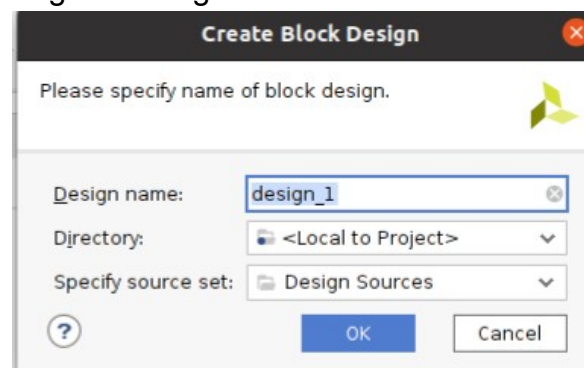
Pošto FPGA logika Zybo razvojne platforme pripada Xilinx seriji čipova, najjednostavniji način da se koriste procesorski sistem, FPGA logika i periferije ove platforme jeste pomoću Vivado IP integratora. On omogućava da, umesto kucanja HDL koda, kroz grafički korisnički interfejs (GUI) dodajete komponente koje želite da koristite.

Da bi se pristupilo ovom alatu prvo je neophodno napraviti novi Vivado projekat, nakon čega se IP integrator pokreće pritiskom na opciju *Create Block Design* (**Napomena: obavezno odabrati razvojnu platformu prilikom pravljenja Vivado projekta**):



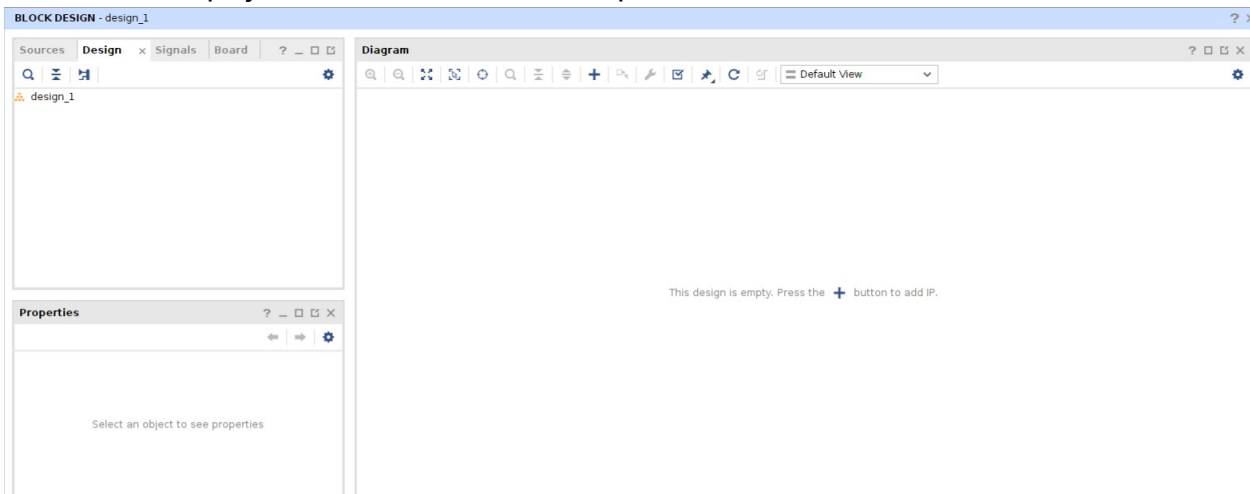
Slika 2. Pokretanje IP integrator alata

Odabirom ove opcije otvara se prozor u kome u polju *Design name* možete da postavite naziv dizajna modelovanog u IP integratoru:



Slika 3. Create block design prozor

Pritiskom na opciju **OK**, otvoriće se sledeći prozor:



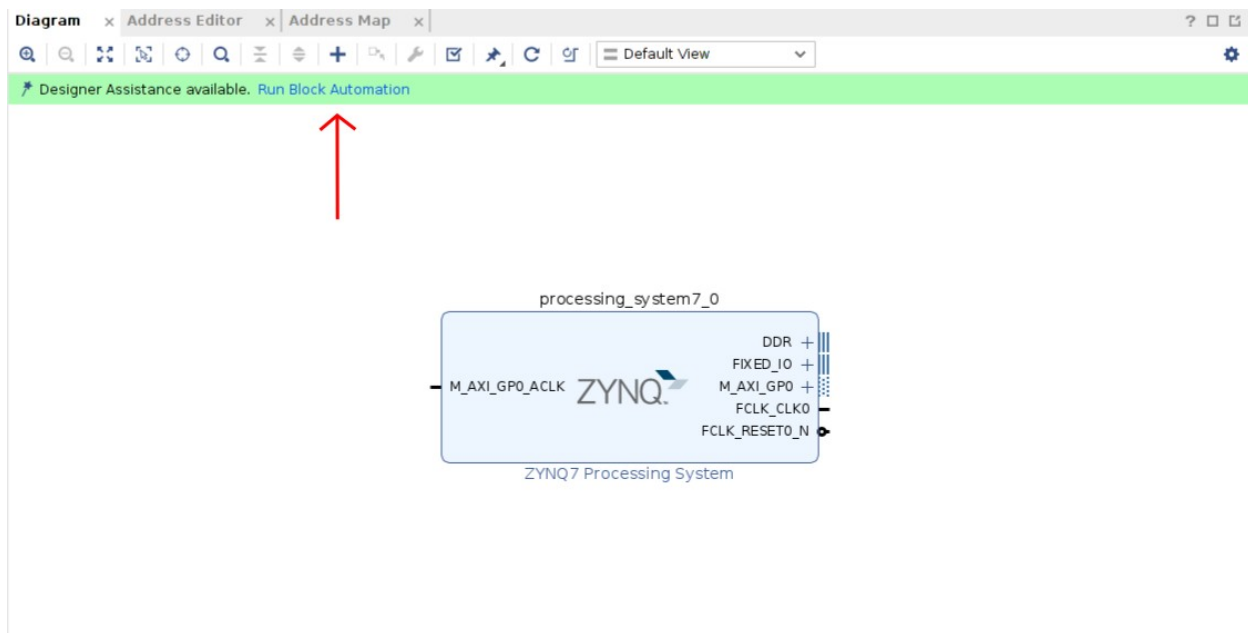
Slika 4. Blok dizajn prozor

Ukoliko pritisnete opciju + prikazanu na slici 4 ili desni klik -> Add IP, imate mogućnost da dodajete komponente u blok integrator. Te komponente mogu da budu:

- Xilinx IP blokovi - već gotovi moduli koji implementiraju određenu funkcionalnost (množači, memorije, itd).
- Custom blokovi - digitalni sistemi koje je napravio korisnik.
- Procesorski sistem - ukoliko se koriste platforme koje su SoC odnosno poseduju procesorska jezgra.

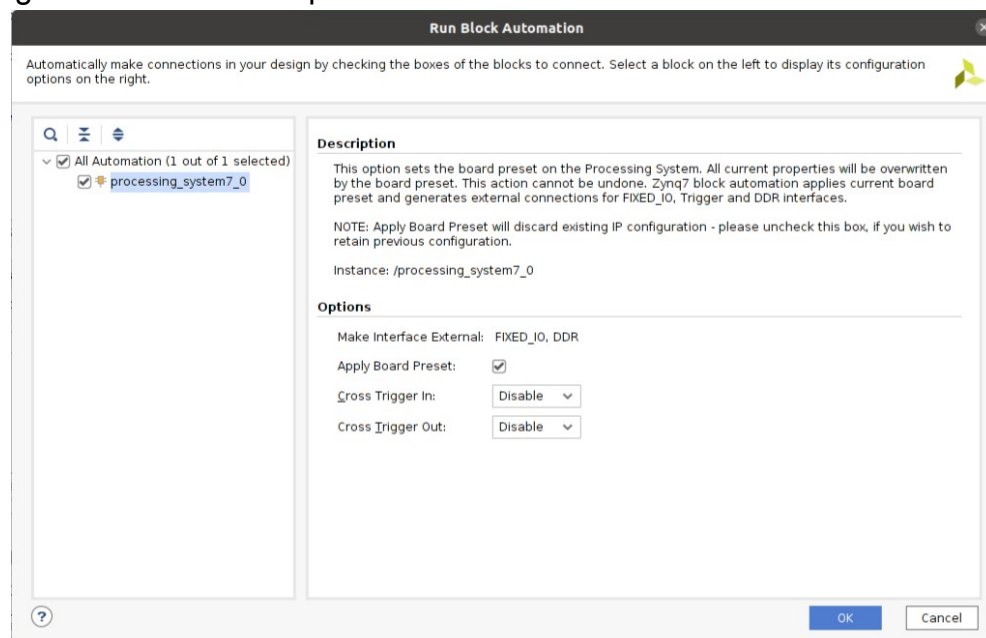
U ovom materijalu biće objašnjeno samo korišćenje procesorskog sistema, dok će u narednim materijalima biti objašnjeno korišćenje Xilinx IP blokova i komunikacije procesorskog sistema sa njima.

Na narednoj slici je u *Block Design* dodat *ZYNQ7 Processing System*:



Slika 5. Dodavanje Zynq procesorskog sistema

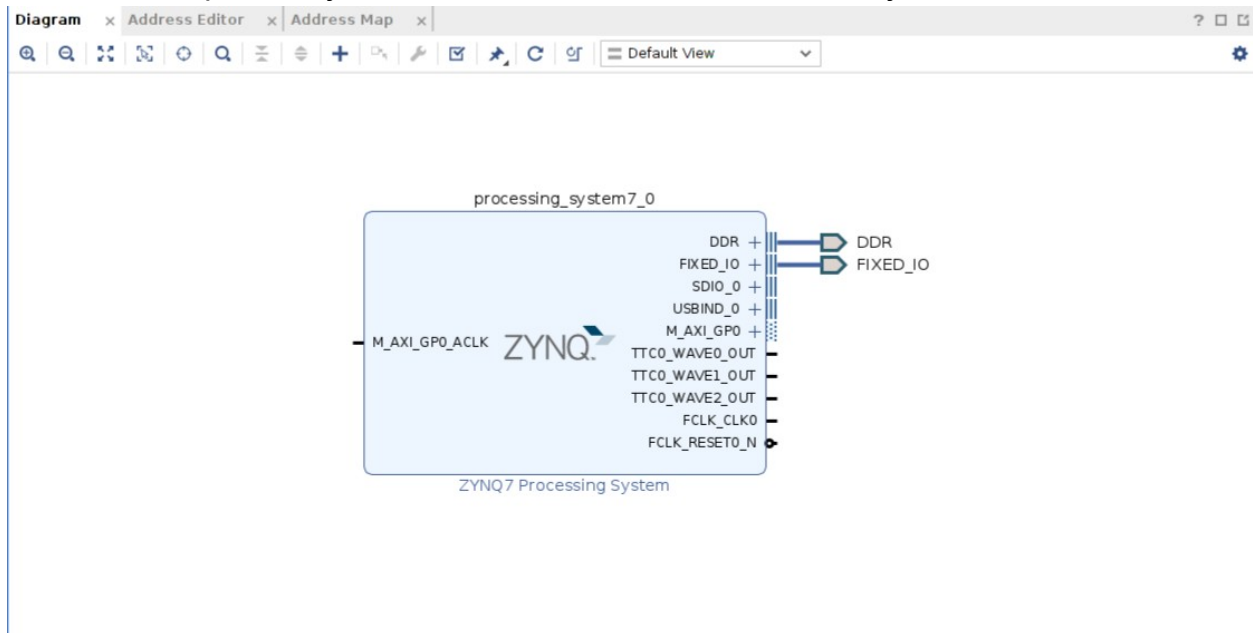
Ovaj blok predstavlja jedno jezgro na Zynq čipu i pomoću IP integratora se ono može konfigurirati, no pre toga, neophodno je pokrenuti komandu *Run Block Automation*. Nakon čega se otvora sledeći prozor:



Slika 6. Run block automation

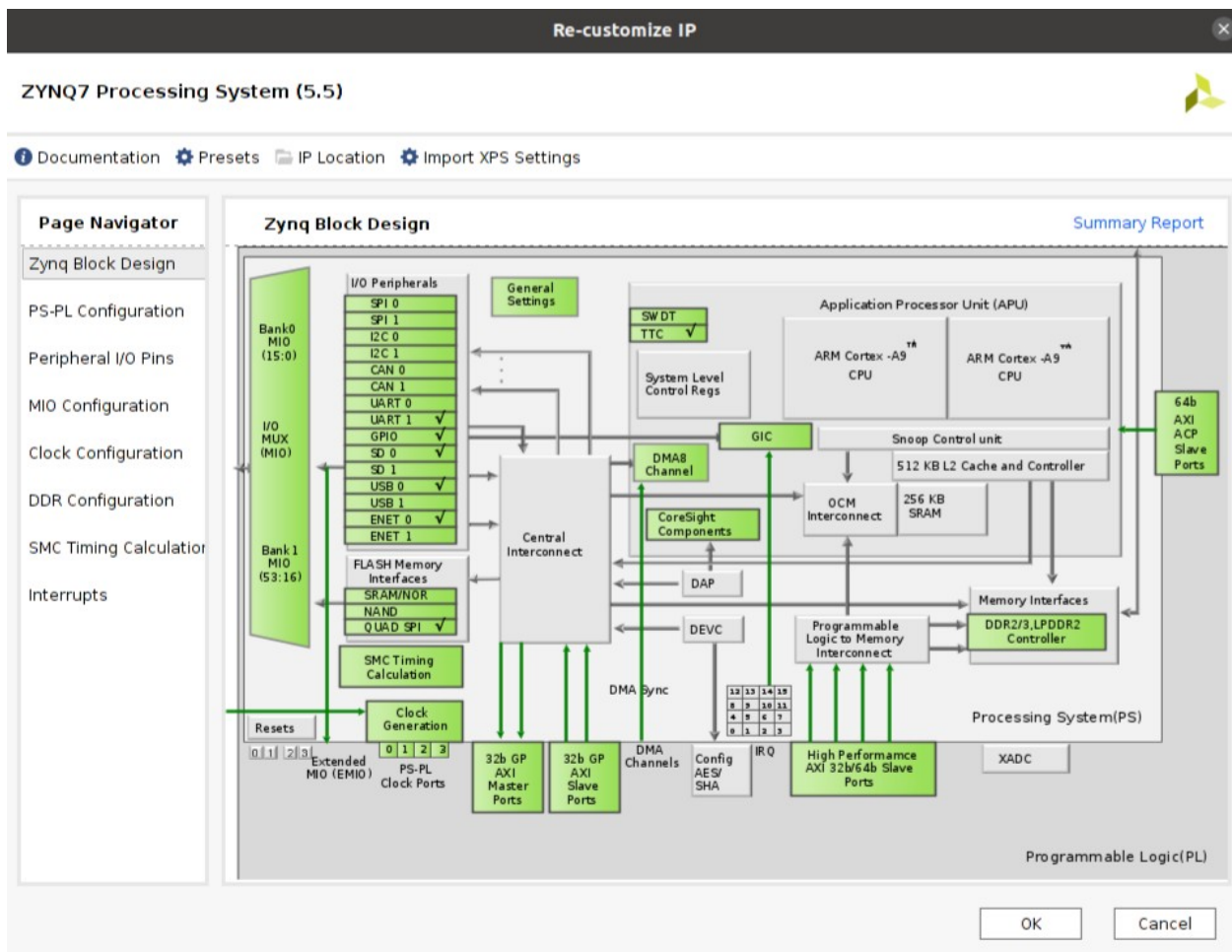
Ova komanda je jako bitna zato što automatski konfigurira zynq procesorsko jezgro (omogućava UART komunikaciju, konfigurira GPIO, itd) i automatski ga povezuje sa DDR interfejsom. Ako obratite pažnju na sliku 1, možete videti da se DDR čip nalazi van procesorskog jezgra i da bi se koristio procesor se preko određenih pinova mora povezati sa njim. Ti pinovi su već predefinisani, ali se u vivadu tačno mora naglasiti koji su. To se može uraditi pomoću .XDC fajla (fajla sa ograničenjima) ili pomoću run block automation komande koja to automatski za nas uradi.

Nakon pokretanja *Run Block Automation* komande dobija se sledeće:



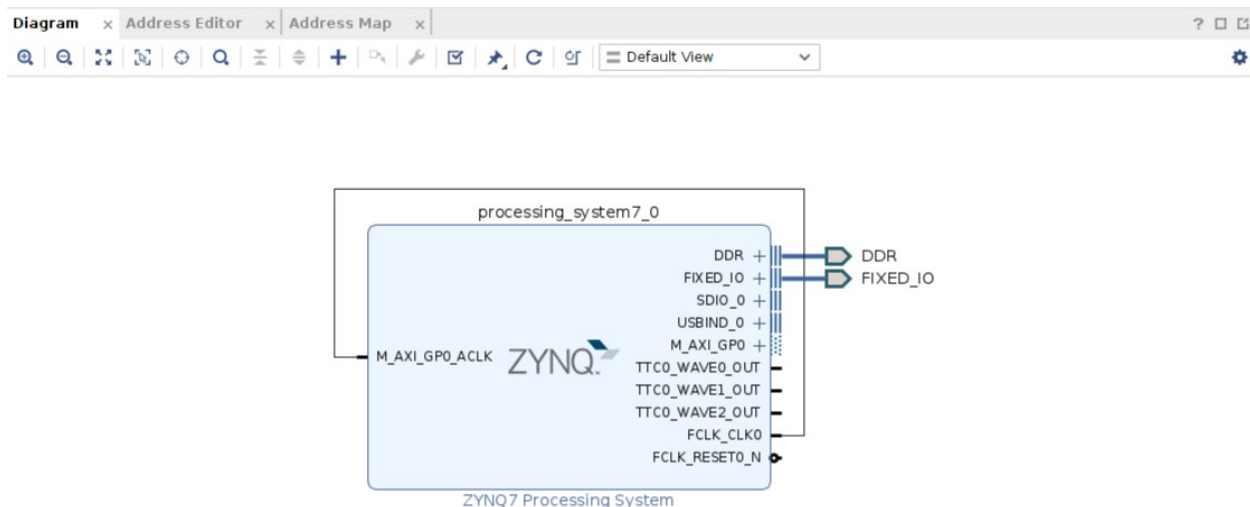
Slika 7. Zynq processing system Nakon pokretanja run automation komande

Može se videti da su dodati određeni interfejsi procesorskom jezgru, ali i da je DDR interfejs povezan. Dvo Klikom na Zynq jezgro ono se može dodatno konfigurirati.



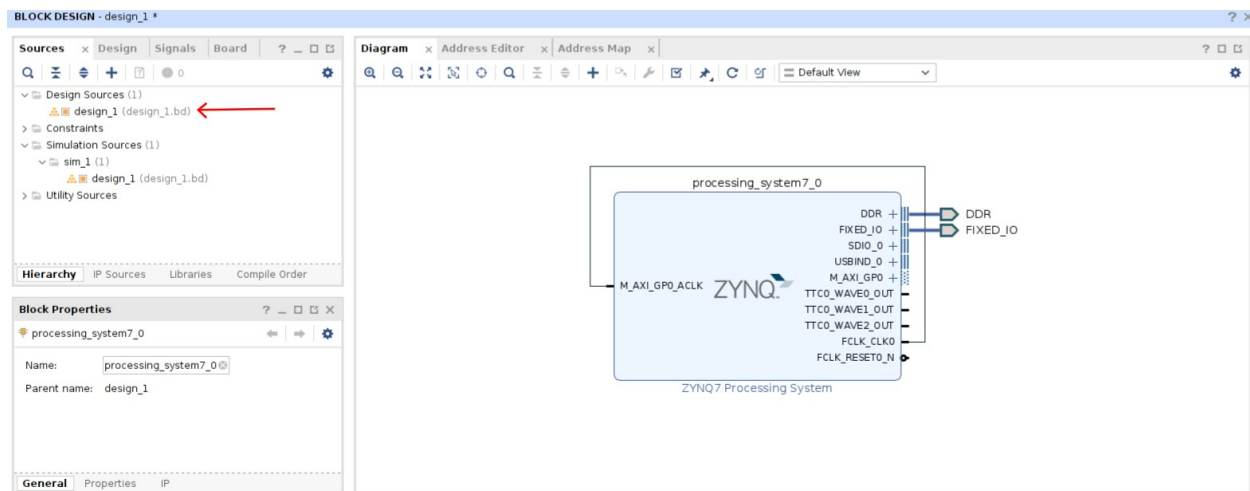
Slika 8. Konfigurisanje Zynq bloka

Run block Automation komanda je već konfigurisala većinu stvari (sve što je otkučeno sa leve strane nije bilo otkučeno pre pokretanja komande), ali dodatno se mogu vršiti i druge konfiguracije. **Za sada ne menjati ništa, u narednim materijalima će biti detaljnija objašnjenja nekih konfiguracija. Samo povezati FCLK_CLK0 port sa M_AXI_GPO_ACLK kao na sledećoj slici:**



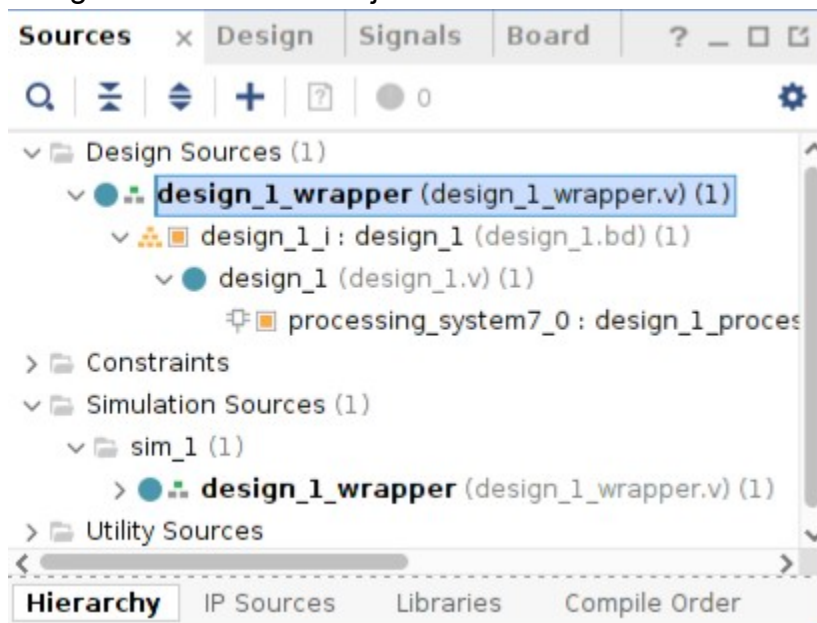
Slika 9. Povezivanje FCLK_CLK0 porta sa M_AXI_GPO_ACLK portom

Nakon što su sve komponente ubačene u *Block Design* prozor (u ovom slučaju samo zynq procesor) neophodno je generisati HDL wrapper, odnosno generisati kod na osnovu koga će se izvršiti sinteza i implementacija. Pritisnuti Desni klik na ime vašeg design bloka i odaberite opciju *Create HDL Wrapper* kao što je to prikazano na sledećoj slici



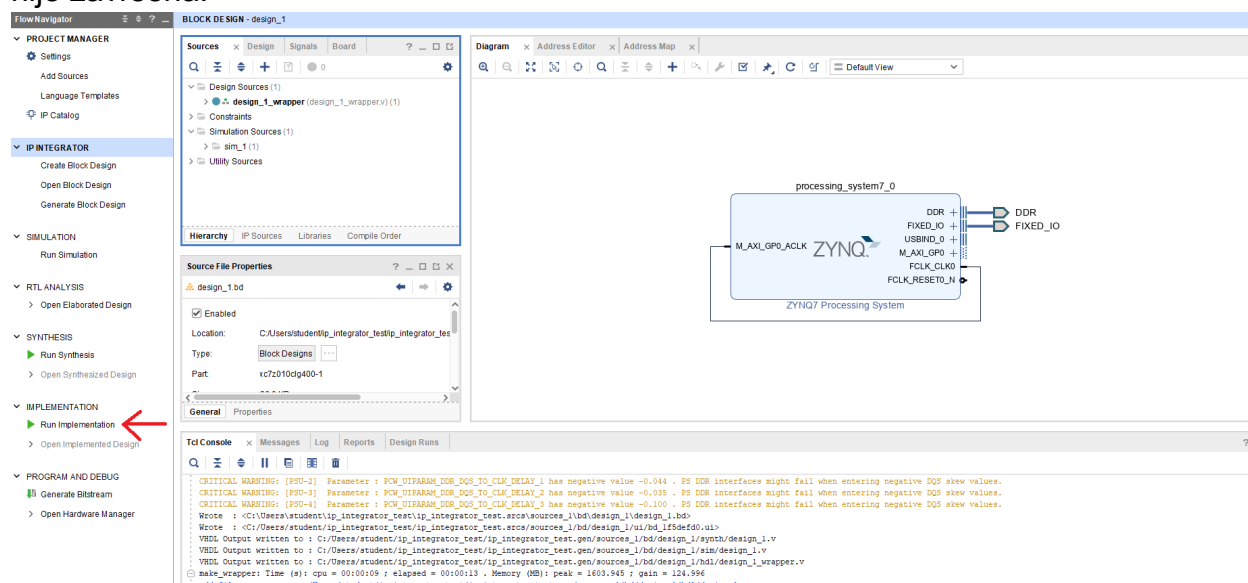
Slika 9. HDL wrapper

Odabirom ove opcije otvoriće se prozor u kome je dovoljno pritisnuti OK, nakon čega će source prozor da izgleda kao na sledećoj slici:



Slika 10. Nakon pokretanja HDL wrapper komande

Sada se može izvršiti sinteza i implementacija digitalnog sistema. Ukoliko pritisnete dugme *run implementation* pokrenuće se i sinteza i implementacija ukoliko sinteza već nije završena:



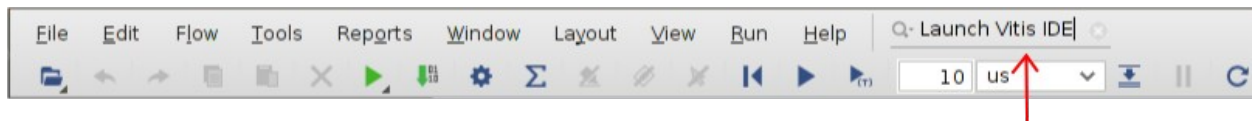
Slika 11. Pokretanje simulacije

Nakon što se implementacija završi (što može potrajati par minuta) pritisnuti dugme **generate bitstream**. Ova komanda generiše fajl pomoću koga se programira ploča. Pošto ćemo ovaj fajl koristiti u drugom alatu koji se zove Vitis, **neophodno je eksportovati ga**. To se radi pritiskom na:

file->Export->Export Hardwer

nakon čega je dovoljno pritisnuti *next*, odabrati opciju *include bitstream*, pritisnuti next i odabrati mesto gde će se **.xsa** fajl (fajl koji programira plogu) eksportovati.

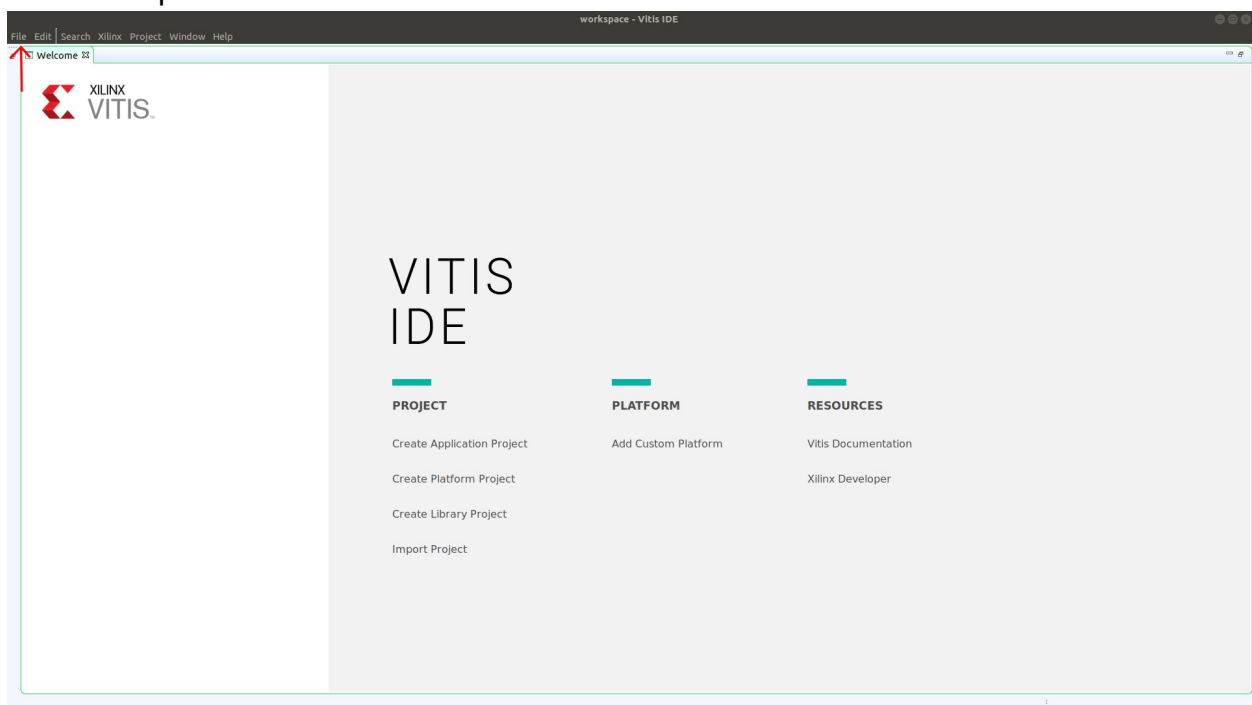
Nakon ovoga urađeno je sve što može da se uradi iz Vivado alata i neophodno je pokrenuti Vitis alat unutar koga ćemo kucati programe koji će se izvršavati na zynq procesorskom sistemu. Vitis može da se pokrene iz Vivada kucanjem “*Launch Vitis*” komande u quick access bar Vivada.



Slika 12. Pokretanje Vitis alata

VITIS

Prilikom pokretanja vitisa prvi prozor koji se otvori određuje gde će biti *workspace* u kome će se sačuvati projekat. Nakon što odaberete direktorijum i pritisnete OK, otvara se sledeći prozor:

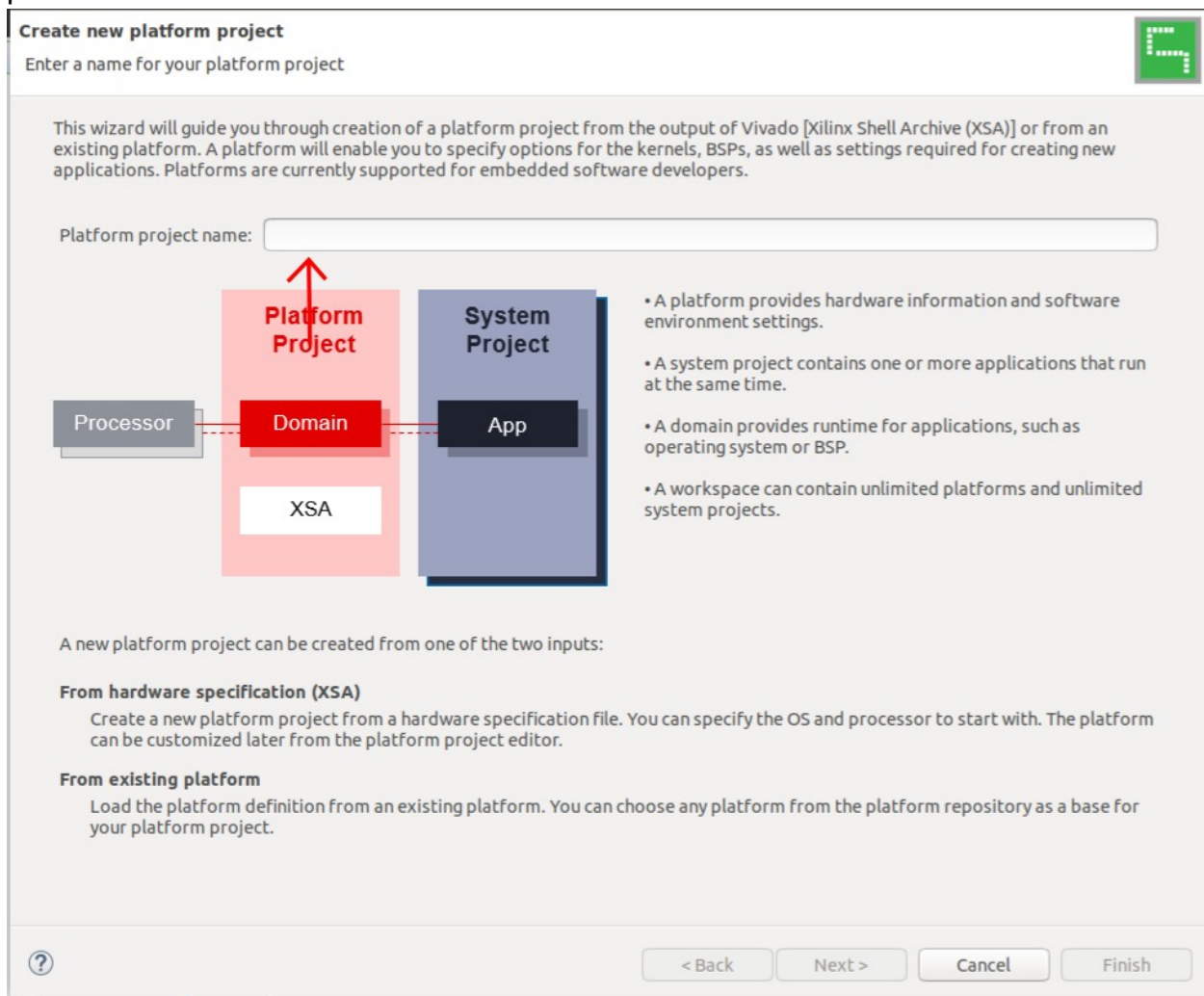


Slika 13. Vitis

Kako bi mogla da se programira određena platforma(Zybo u našem slučaju), neophodno je napraviti Vitis platform project. Pritiskom na:

File -> new -> Platform project

Otvoriće se sledeći prozor u kome semo treba da se unese ime platform projekta i pritisne next:



Slika 14. Pravljenje platform projekta

Sada je potrebno dodati u projekat .xsa fajl koji se koristi prilikom programiranja platforme iz Vitis alata. U prethodnoj sekciji smo eksportovali taj fajl, i sada je samo potrebno da ga ubacimo u vitis i se radi preko opcije *browse*, kao na sledećoj slici:

Platform

Please select a platform to create the project

Create a new platform from hardware (XSA) **Select a platform from repository**

Hardware Specification

XSA File: Provide your XSA file or use a pre-built board description

Software Specification

Specify the details for the initial domain to be added to the platform. More domains can be added after the platform is created by double clicking the platform.spr file

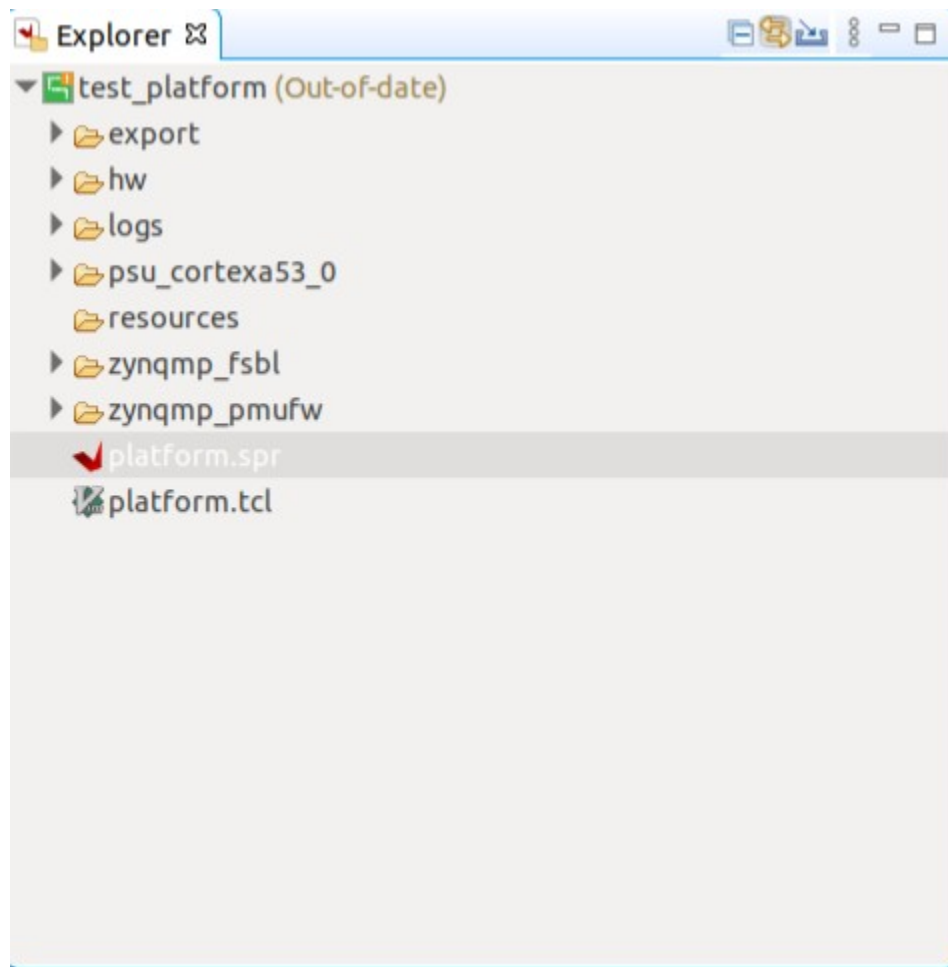
Operating system:

Processor:

Slika 15. Dodavanje .xsa fajla

Kada se .xsa fajl doda pritisnuti **next** i nakon toga **finish**.

Vitis će sada napraviti *platform project* i u Explorer prozoru će se nalaziti sledeće:



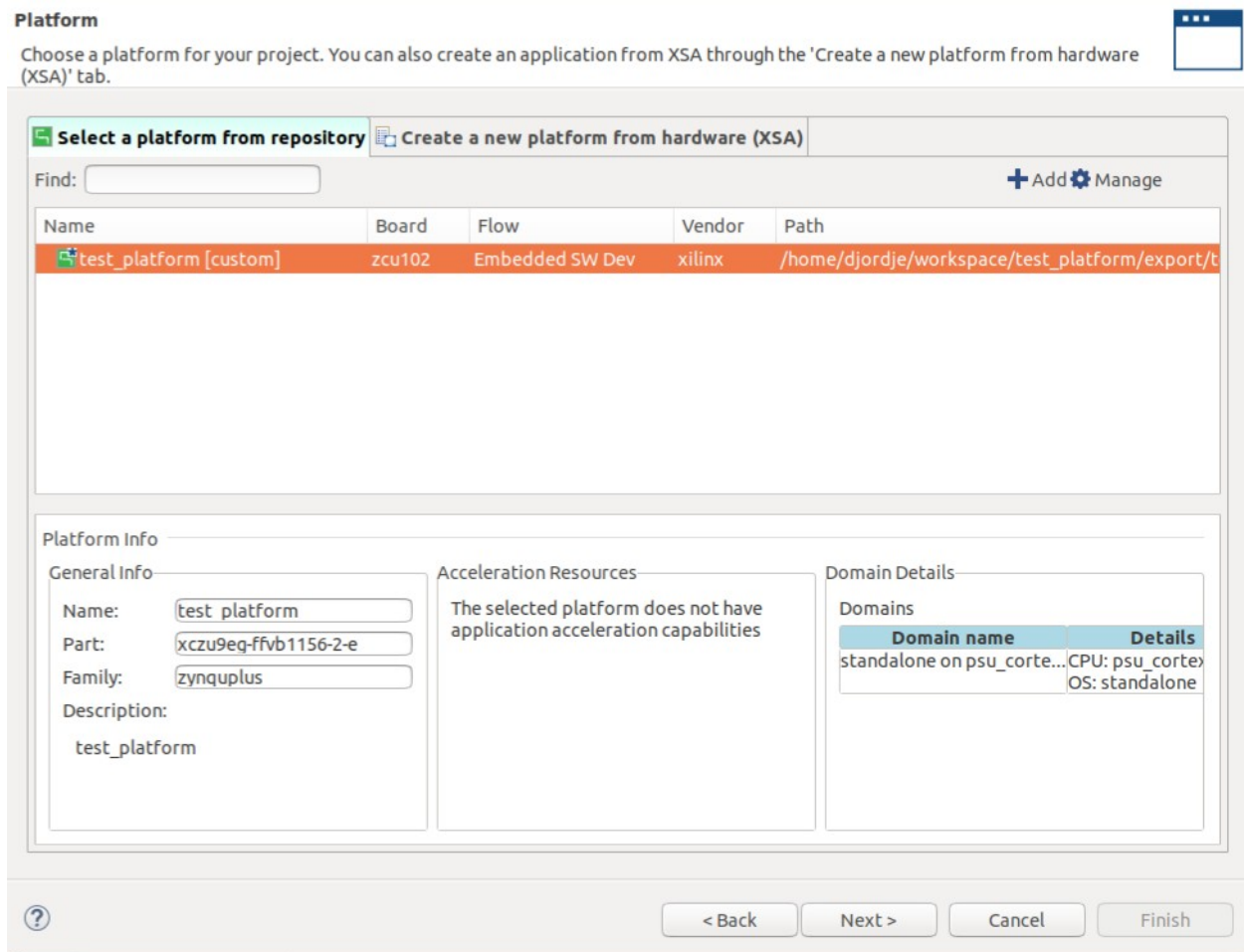
Slika 16. Vitis Explorer

Da bi se završilo pravljenje platform projekta neophodno je desnim klikom na platform project (test platform sa prethodne slike) otvoriti dodatne opcije i pritisnuti **build project**. Nakon što se ova komanda izvrši pored imena platform projekta više ne bi trebalo da stoji **out-of-date**.

Sada mora da se napravi application projekat i da se uveže sa platform projektom. To se radi sledećom komandom:

File -> new -> Application project

U prozoru koji se otvori pritisnuti **Next** kako bi se pojavilo sledeće:



Slika 17. Application project

Sada je potrebno uvezati application projekat sa određenim platform projektom. Kako njih može biti više treba odabrati pravi, ali u primeru sa slike 1 postoji samo jedan platform projekat. Nakon odabira pritisnuti **Next**. Sada uneti ime application projekta i klikatati na next sve dok se ne otvori prozor sa slike 19 :

Application Project Details

Invalid application project name. Project name must be specified

Application project name:

System Project

Create a new system project for the application or select an existing one from the workspace

Select a system project
+ Create new...

System project details
System project name:
Target processor
Select target processor for the Application project.

Processor	Associated applications
psu_cortexa53_0	

Show all processors in the hardware specification ☐

< Back Next > Cancel Finish

Slika 18. Unos imena application projekta

Templates

Select a template to create your project.

Available Templates:

Find:

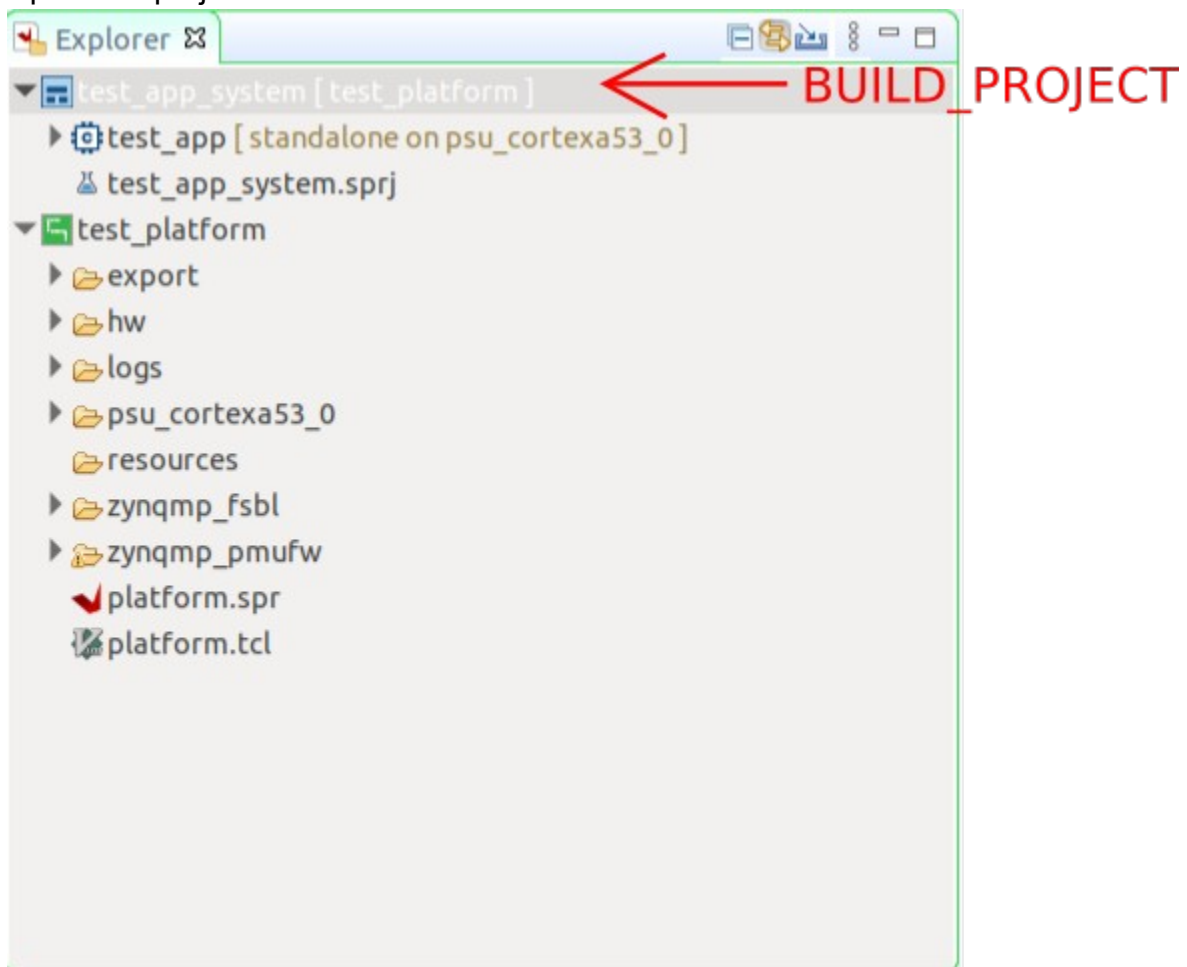
SW development templates
Empty Application
Empty Application (C++)
Hello World
Image Recovery
Image Selector
lwIP Echo Server
lwIP TCP Perf Client
lwIP TCP Perf Server
lwIP UDP Perf Client
lwIP UDP Perf Server
Memory Tests
Peripheral Tests
Zynq MP DRAM tests
Zynq MP FSBL

Hello World
Let's say 'Hello World' in C.

< Back Next > Cancel Finish

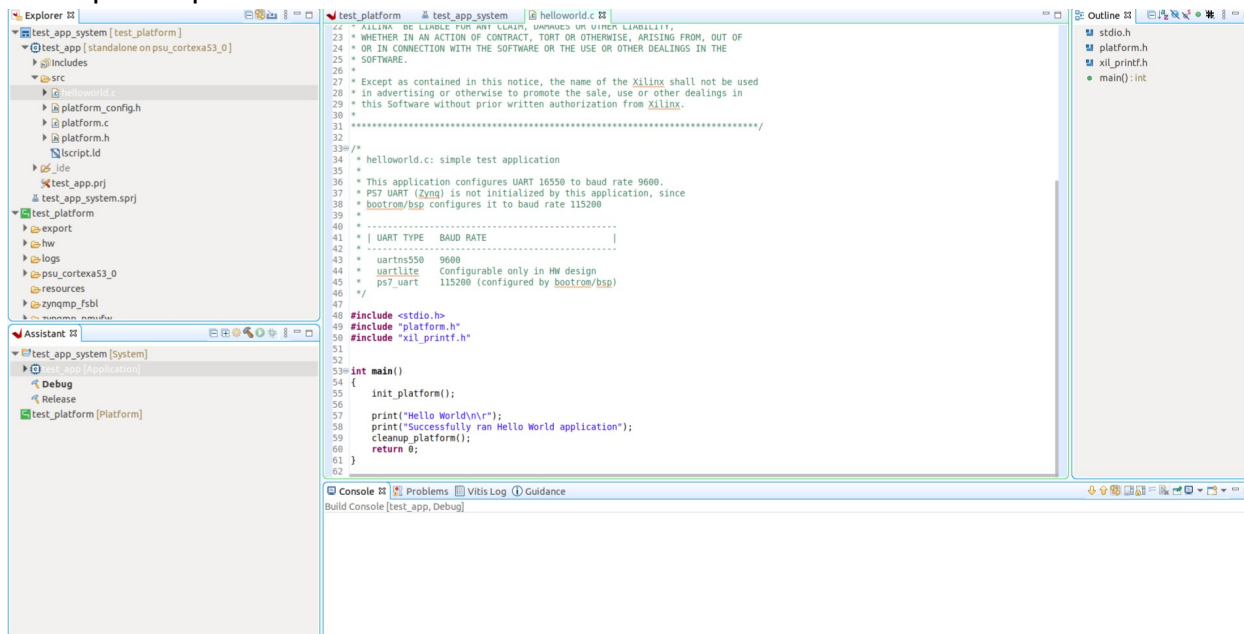
Slika 19. Odabir hello world aplikacije

Sada je napravljen application projekat i potrebno je pokrenuti komandu build isto kao i kod platform projekta:



Slika 20.

Posle svih ovih koraka urađeno je sve neophodno kako bi se programirala platforma. Sada može da se napiše C kod pomoću koga može da se programira jezgro koje smo dodali u IP integratoru. Za ovako napravljen Application Project, napraviće se hello world primer prikazan u nastavku:



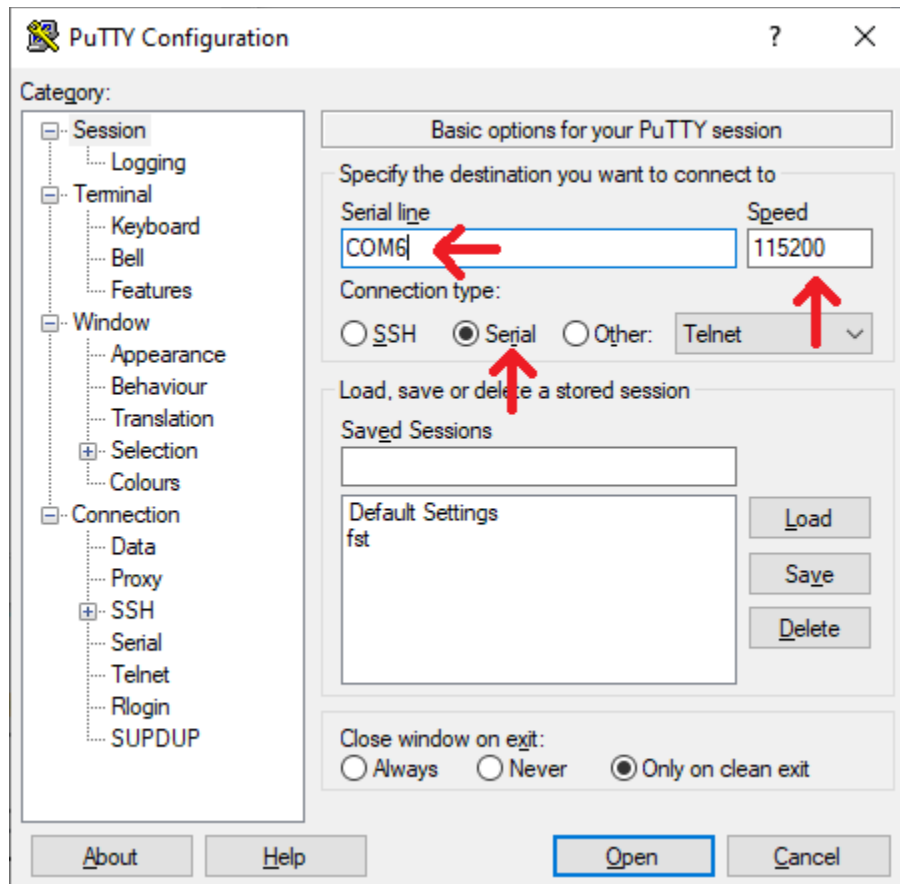
Slika 21. Hello world template

Pre nego što se pokrene ovaj kod, priključiti razvojnu ploču za računar (preko USB kabela) i otvoriti **putty** program. Ukoliko **Putty** nije instaliran možete ga preuzeti sa sledećeg linka:

<https://www.putty.org/>

Putty omogućava komunikaciju računara i razvojne platforme preko serijskog protokola i ovo je neophodno kako bi videli šta razvoja ploča radi (sve **printf** poruke se vraćaju našem računaru preko serijskog interfejsa).

Kada se pokrene putty otvoriće se sledeći prozor:



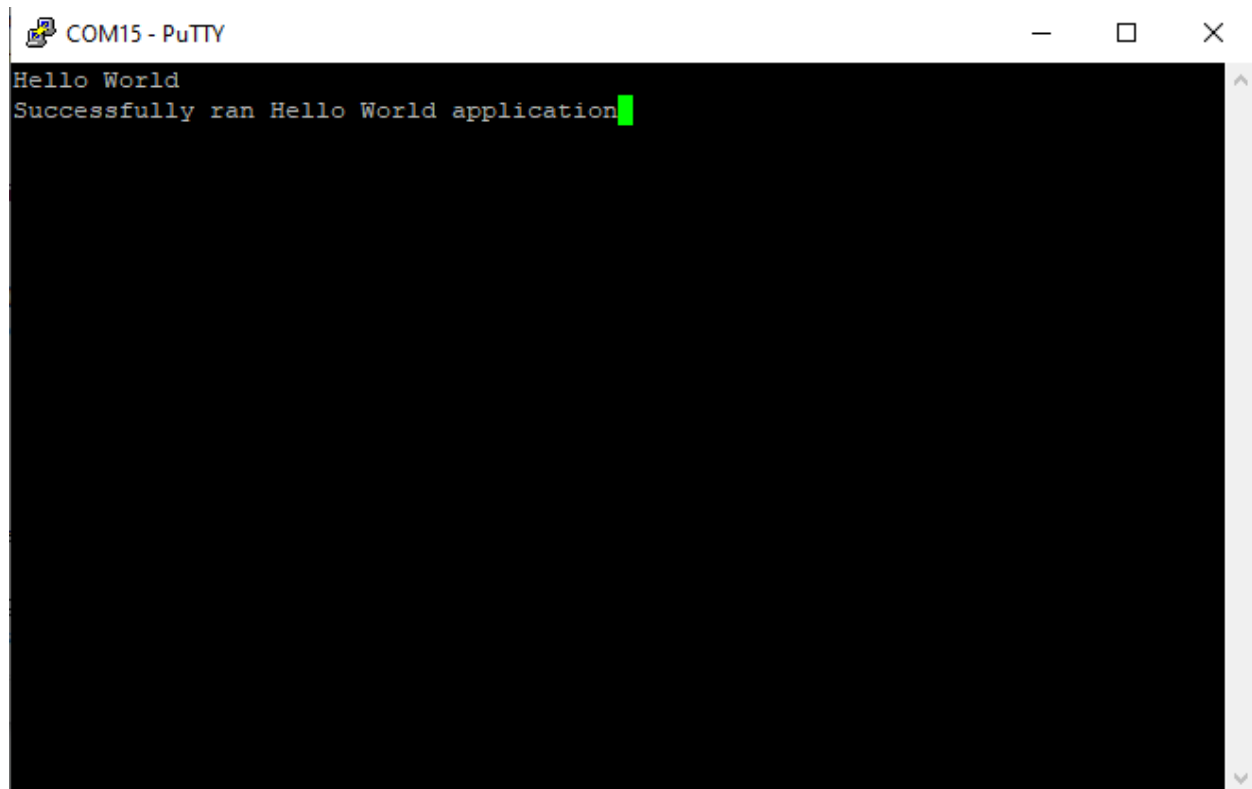
Slika 22. Putty

Podesiti opcije kako je naznačeno na slici 22. Prilikom odabira serial line opcije na windows mašinama proveriti u **device manager** alatu koji COM je razvojna ploča dobila.

Sada iz Vitisa može da se programira ploča odabirom sledeće opcije:

Desni klik na Application project -> run as -> Launch Hardware

Ako je sve urađeno kako treba u putty prozoru će se ispisati sledeće:



```
COM15 - PuTTY
Hello World
Successfully ran Hello World application
```

Slika 23. Ispis Hello World poruke