Vežba8 - rad sa periferijama

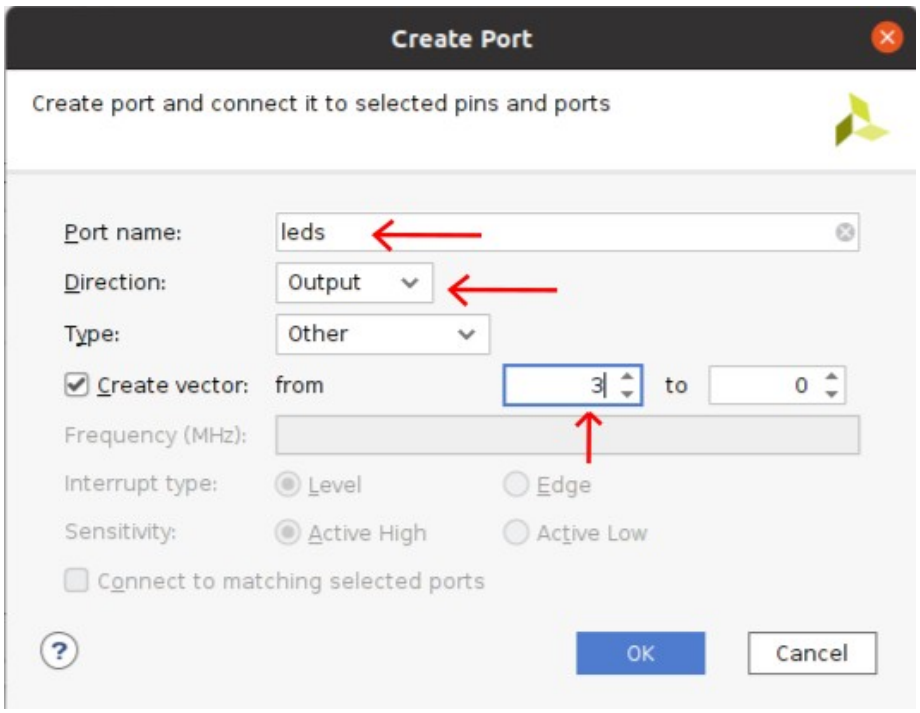
Uvod

Na prethodnim vežbama je objašnjen način korišćenja *Zynq* procesorsog bloka u Vivado IP integratoru, kako se on konfiguriše i kako se programira pomoću Vitis alata. Sledeći korak jeste rad sa periferijama *Zybo* razvojne platforme kao što su LED diode, tasteri i prekidači.

Kreiranje portova u IP integratoru

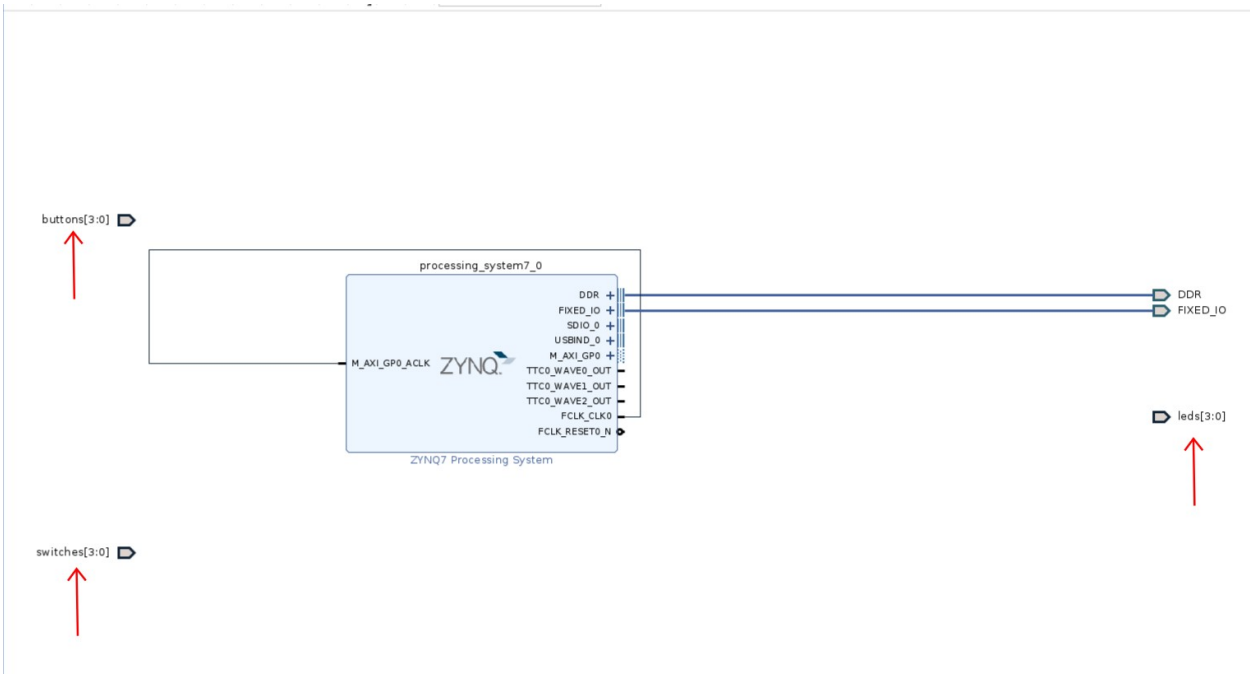
Da bi se omogućila komunikacija sa periferijama koje se nalaze na razvojnoj platformi neophodno je proširiti Vivado projekat sa prethodne vežbe. Prva stvar koju dodajemo jesu portovi preko kojih će se pristupati periferijama. Na *Zybo* platformi postoje 4 tastera, 4 prekidača i 4 LED diode, tako da će nam trebati tri četvorobitna porta kako bismo komunicirali sa njima (**dva ulazna i jedan izlazni**).

Kreiranje portova se realizuje desnim klikom u IP integratoru i odabirom opcije *Create Port* nakon čega se otvara sledeći prozor:



Slika 1. Create port

Kada se naprave 3 porta IP integrator treba da ima sledeći izgled:



Slika 2. IP integrator nakon dodavanja portova

Constraint fajl (Fajl sa ograničenjima)

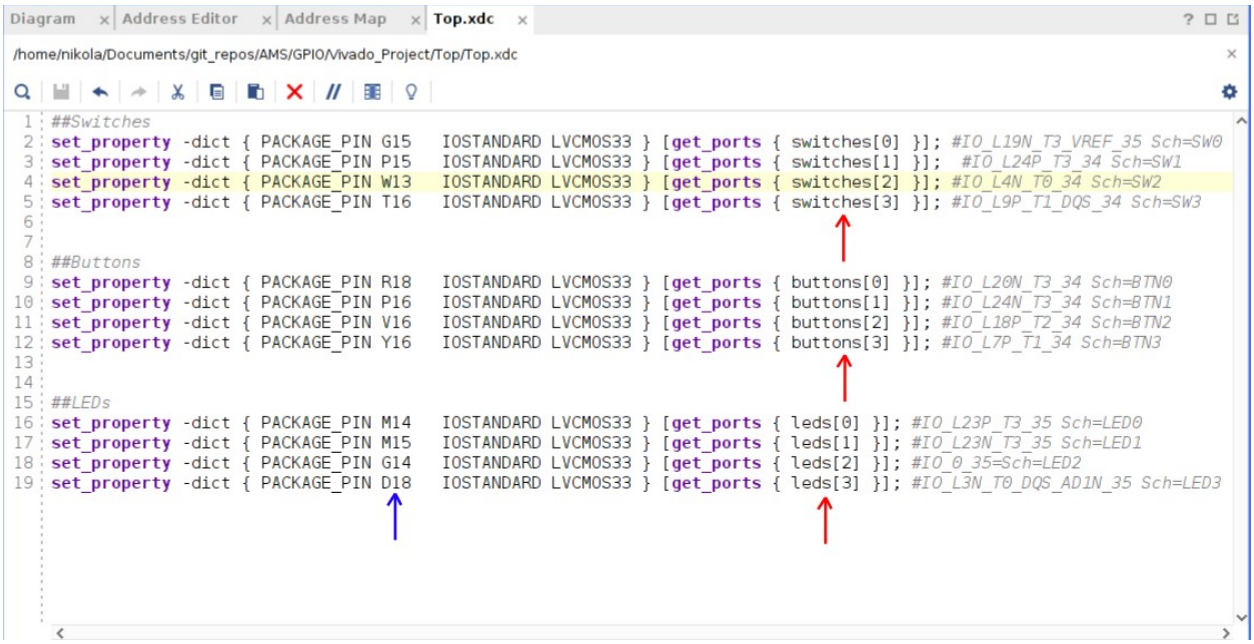
Portovi koji su napravljeni treba da se povežu sa stvarnim fizičkim portovima na koje su povezane diode, prekidači i tasteri. Da bi se ovo omogućilo neophodno je napraviti *Constraint* fajl u kome će se naznačiti na koji fizički port se dovodi port iz IP integratora. Constraint fajl se pravi odabirom opcije:

Add sources -> constraint file

U zavisnosti od platforme na kojoj se radi oznake fizičkih pinova se mogu razlikovati. Kako ćemo na laboratorijskim vežbama raditi sa dve platforme Zybo i Zybo Z710 mora se voditi računa o oznakama pinova. Na sledeća dva linka imate primer *Constraint* fajla za obe razvojne platforme i neophodno je samo odkomentarisati određene linije kako bi se koristile željene periferije:

Zybo *constraint file*: link
Zybo Z710 *constraint file:* link

U našem slučaju treba odkomentarisati *switches, buttons* i *leds* segmente i promeniti imena portova tako da se poklapaju sa onim u IP integratoru:



Slika 3. *Constraint File* (fajl sa ograničenjima)

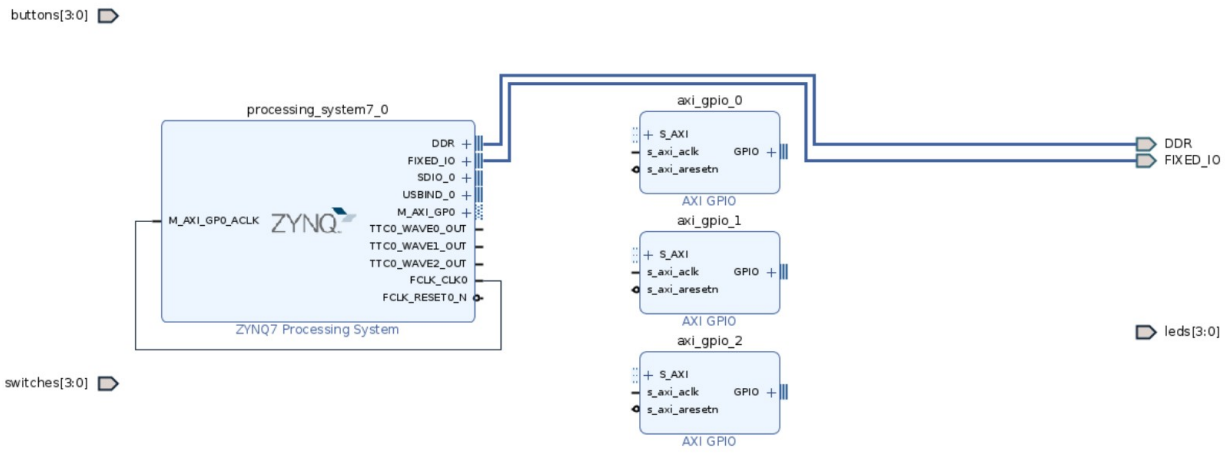
Crvene strelice naznačavaju koji portovi su povezani na fizičke pinove, dok plava strelica pokazuje koji fizički pin je iskorišćen. **Obratiti pažnju da imena naznačena crvenom strelicom moraju da se poklapaju sa imenima portova u IP integratoru.** Sada će nakon sinteze i implementacije alat znati da portove navedene u IP integratoru treba da mapira na fizičke portove.

Povezivanje Zynq procesorskog sistema sa portovima

Cilj ove vežbe je da se omogući procesorskom sistemu da komunicira sa prethodno pomenutim periferijama. To znači da Zynq procesorski sistem treba da se poveže sa njima. Kako on nema portove koji omogućavaju direktno povezivanje sa periferijskim portovima moraju se ubaciti dodatne komponente i te dodatne komponente su AXI GPIO moduli. Oni sa jedne strane poseduju AXI lite interfejs a sa druge magistralu proizvoljne širine. Ovo je neophodno uraditi jer procesor za komunikacija sa komponentama mora da koristi standardne AXI interfejse i GPIO moduli su tu da prihvate podatke koji dolaze sa AXI lite interfejsa i prebace ih na nama prihvatlji interfejs (4-bitna magistrala). Ovi moduli se u IP integratoru ubacuju na sledeći način:

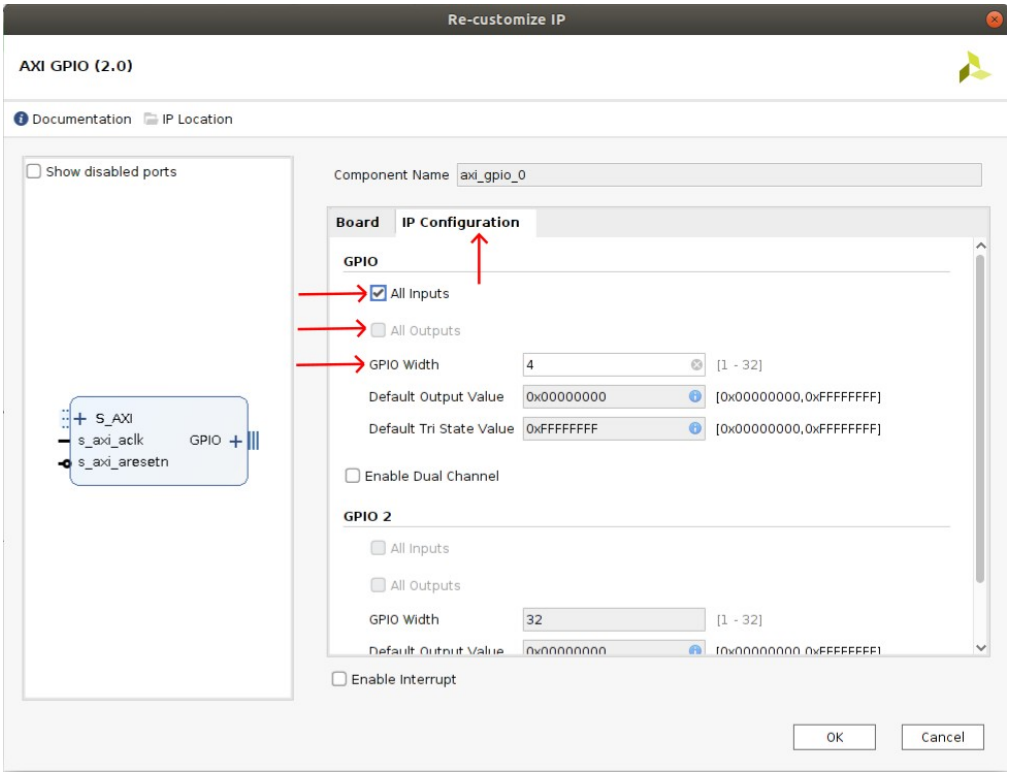
Desni klik na radnu površinu -> add IP -> AXI GPIO

Dodati 3 ovakva modula nakon čega IP integrator treba da ima sledeći izgled(otprilike):



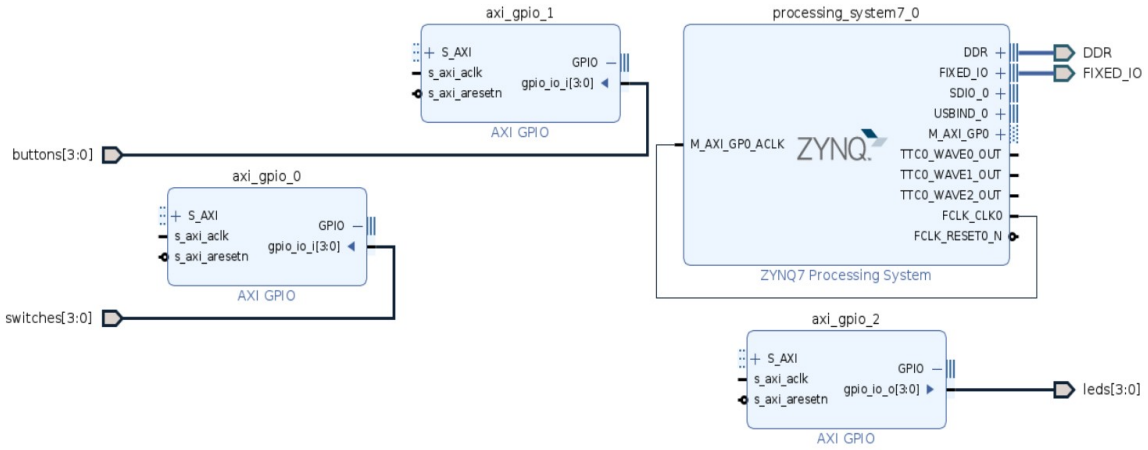
Slika 4. IP integrator sa AXI GPIO modulima

Sada ove module treba povezati sa napravljenim portovima, ali pre nego što se to može uraditi svaki GPIO modul se mora konfigurisati. To se radi tako što se dvoklikom na njih otvara prozor za podešavanja:



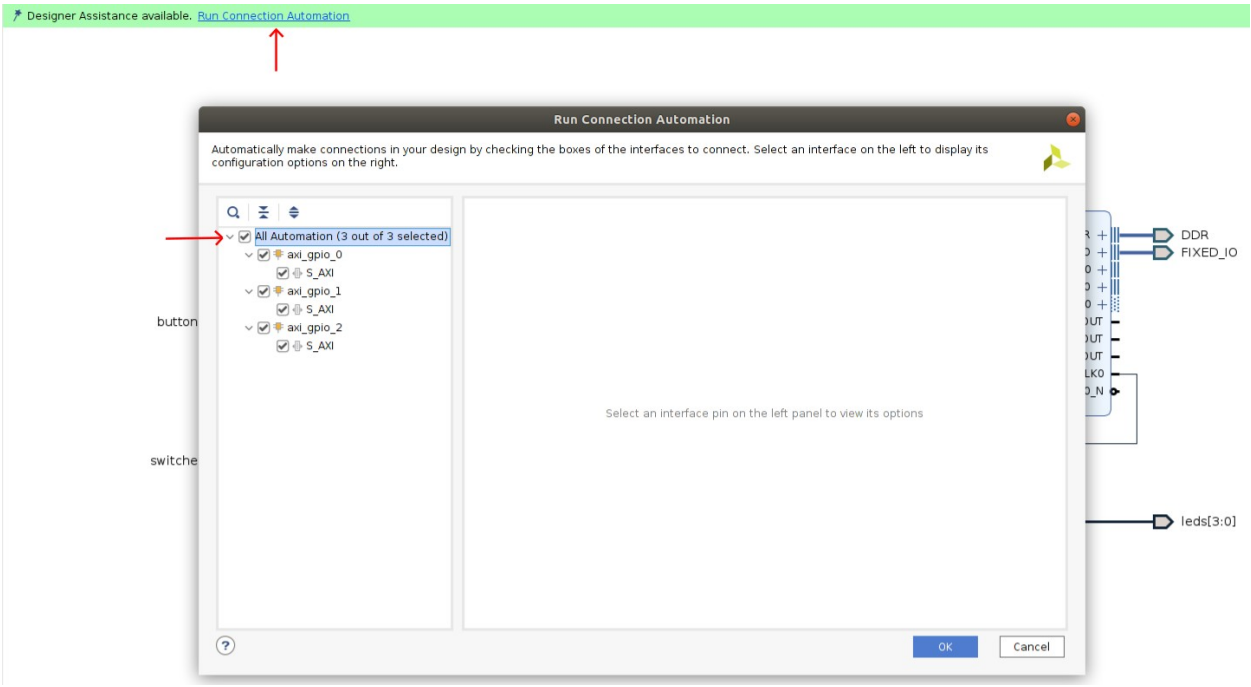
Slika 5. Konfigurisanje GPIO modula

GPIO modul povezan sa Leds portom konfigurisati da bude izlazni, dok ostala dva konfigurisati da budu ulazna. Povezati GPIO module sa periferijskim portovima na sledeći način:



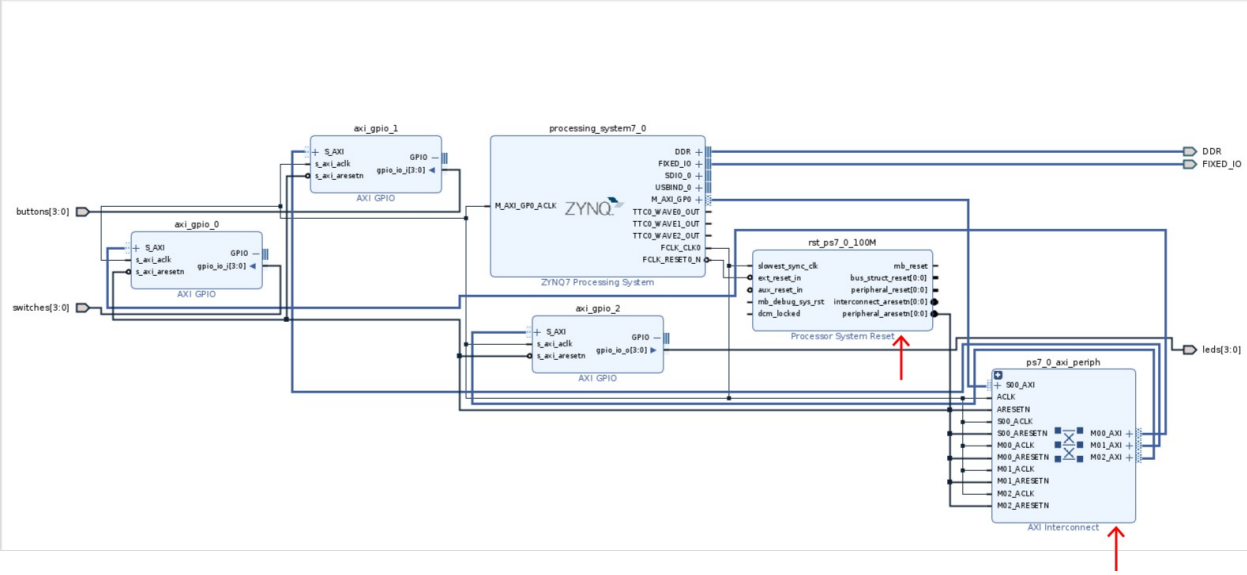
Slika 6. Povezivanje GPIO modula sa periferijskim portovima

Sada se procesor može povezati sa GPIO modulima. U Vivadu se ovo može uraditi na jednostavan način pritiskom na opciju “Run connection Automation”, koja će automatski povezati sve za nas:



Slika 7. Automatsko povezivanje procesora sa GPIO modulima

Kada se pritisne ok u IP integratoru bi trebalo da se pojavi sledeće:



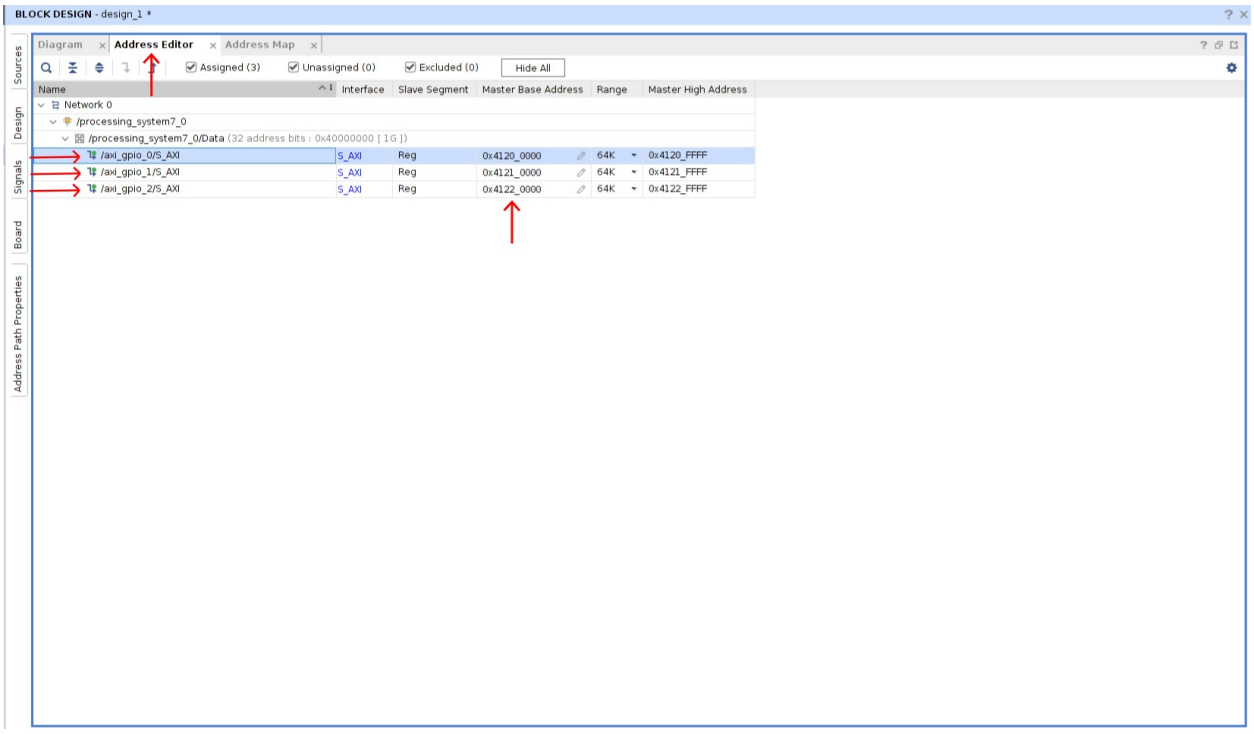
Slika 8. Nakon pokretanja komande za automatsko povezivanje

Sa slike se može videti da je Vivado ubacio dve dodatne komponente: Processor System Reset i AXI interconnect. *Axi interconnect* omogućava da se jedan AXI interfejs procesorskog modula (M_AXI_GPIO) poveže sa više komponenti koje poseduju AXI interfejs, dok Processor System Reset razvodi procesorski reset (fclk_reset0_N) do svih komponenti.

Sa ovim je završeno projektovanje u IP integratoru i može se izvršiti *HDL wrapping*, sinteza, implementacija, generisanje bitstream fajla i njegovo eksportovanje.

Programiranje iz Vitis alata

Pre nego što pokrenete Vitis obratite pažnju u Vivadu na address Editor prozor:



Slika 9. Address editor

U ovom prozoru su jedinstveni adresni opsezi svakog GPIO modula koje ćemo koristiti kako bi komunicirali sa periferijama. Svi moduli koji poseduju AXI lite slave ili AXI full slave interfejs dobijaju ove jedinstvene adrese.

Sada kreirati platform i application project kao u materijalu sa prethodnih vežbi. Primer koda pomoću koga može da se manupuliše LED diodama, tasterima i prekidačima je dat u nastavku:

```
#include <stdio.h>
#include "platform.h"// biblioteka u kojoj se nalaze Xil_in32 i
Xil_Out32 funkcije
#include "xil_printf.h"
#include "xil_io.h"
#include "xparameters.h"
#include "sleep.h"


//Obavezno Upisati bazne adrese koje vivado generiše!
```

```c
#define LED_BASEADDR     0x41200000
#define BUTTON_BASEADDR 0x41210000
#define SWITCH_BASEADDR 0x41220000

int buttons = 0;
int button3,button2,button1,button0 = 0;
int switches = 0;
int switch3,switch2,switch1,switch0 = 0;
int main()
{
  init_platform();

  print("Hello GPIO\n\r\n\r");

  print("***************************************\n\r");
  print("Working With LED! LOOK AT THEM! \n\r");
  print("***************************************\n\r");

  // For loop goes trough all 16 combinations of LEDs turned
ON/OFF
  for(int i=0; i<16; i++)
    {
      Xil_Out32(LED_BASEADDR,i); // Xil_Out32 sets values on a
chossen address
      sleep(1); // delay 1s
    }
  sleep(2);
  print("***************************************\n\r");
  print("Working With BUTTON! PRESS BUTTONS !!!\n\r");
  print("***************************************\n\r");

  sleep(2);
  // For loop prints out 10 times, the status of buttons
  for(int i=0; i<16; i++)
    {
      buttons = Xil_In32(BUTTON_BASEADDR);
      button3 = (buttons&0x8) ? 1 : 0; //0x8 = 0b1000
      button2 = (buttons&0x4) ? 1 : 0; //0x4 = 0b0100
      button1 = (buttons&0x2) ? 1 : 0; //0x2 = 0b0010
```

```
      button0 = (buttons&0x1) ? 1 : 0; //0x1 = 0b0001
      printf("BUTTON3= %d, BUTTON2= %d, BUTTON1= %d, BUTTON0 =
%d\n\r",button3,button2,button1,button0);
      sleep(1); // delay 1s
    }
  sleep(2);
  print("*************************************\n\r");
  print("Working With SWITCHES! MOVE SWITCHES !!!\n\r");
  print("*************************************\n\r");
  sleep(2);
  // For loop prints out 10 times, the status of buttons
  for(int i=0; i<16; i++)
    {
      switches = Xil_In32(SWITCH_BASEADDR);
      switch3 = (switches&0x8) ? 1 : 0; //0x8 = 0b1000
      switch2 = (switches&0x4) ? 1 : 0; //0x4 = 0b0100
      switch1 = (switches&0x2) ? 1 : 0; //0x2 = 0b0010
      switch0 = (switches&0x1) ? 1 : 0; //0x1 = 0b0001
      printf("SWITCH3= %d, SWITCH2= %d, SWITCH1= %d, SWITCH0 =
%d\n\r",switch3,switch2,switch1,switch0);
      sleep(1); // delay 1s
    }
  sleep(2); // delay 1s
  print("Successfully ran Hello GPIO application");
  cleanup_platform();
  return 0;
}
```

Obratiti pažnju na Xil_In32 i Xil_Out32 funkcije. Prva učitava vrednost sa određene adrese, u našem slučaju ono što treba čitati jeste stanje tastera i prekidača. Postavljanjem bazne adrese odgovarajućeg GPIO modula može se pročitati stanje periferije.
Xil_Out32 omogućava slanje vrednosti na odgovarajuću adresu. U našem slučaju jedina periferija kojoj ima smisla poslati vrednost jeste LEDs.