## PROJEKAT

## iz Digitalnih sistema otpornih na greške

### TEMA PROJEKTA:

|  |
| --- |
| FIR filtar proizvoljnog reda koji je otporan na greške. |

### TEKST ZADATKA:

|  |
| --- |
| Implementirati FIR filtar proizvoljnog reda na koji je primijenjena *N-modular Redundancy* tehnika otpornosti na otkaz. Odbirci koji se filtriraju prvo se smeštaju u BRAM memoriju, kao i nakon što se obrade. Potrebno je parametrizovati BRAM memorije tako da prije sinteze može da se naznači broj odbiraka koji se filtrira, kao i red filtra. |

Mentor Student

Nikola Kovačević Ksenija Radonjić E161/2022

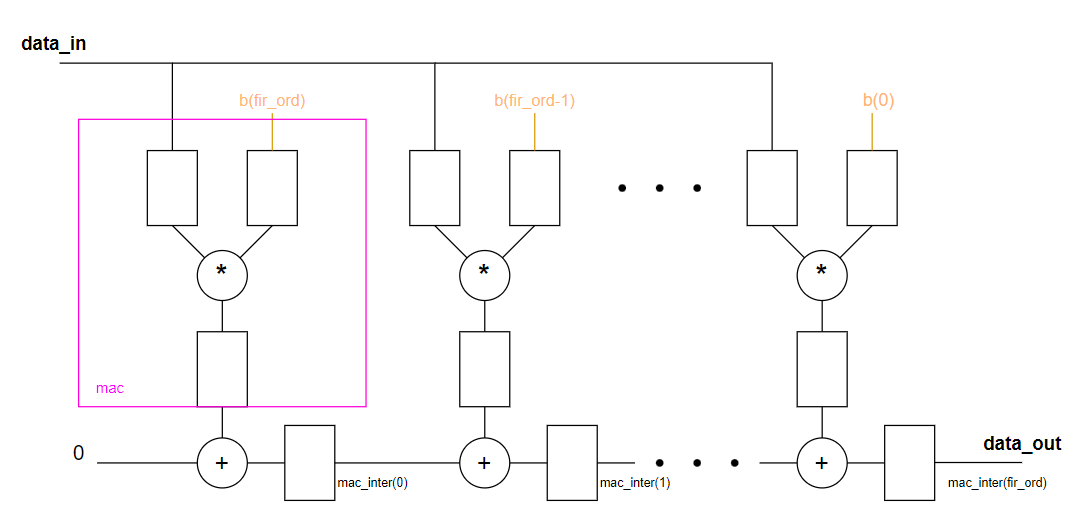
U Novom Sadu, februar 2023.

# 1. Funkcionalna i dizajn specifikacija sistema

U ovom projektu potrebno je projektovati digitalni filtar (FIR filtar) koji treba da ima sledeće funkcionalnosti:

* Filtar treba da bude proizvoljnog reda
* Potrebno je da širina podataka bude proizvoljna
* Koristiti transponovanu direktnu formu
* Primijeniti *N-modular Redundancy* tehniku otpornosti na otkaz
* Koristiti BRAM memoriju za skladištenje podataka, sa parametrizovanim brojem odbiraka koji se filtrira

Blok šema FIR sistema je prikazana na slici 1.



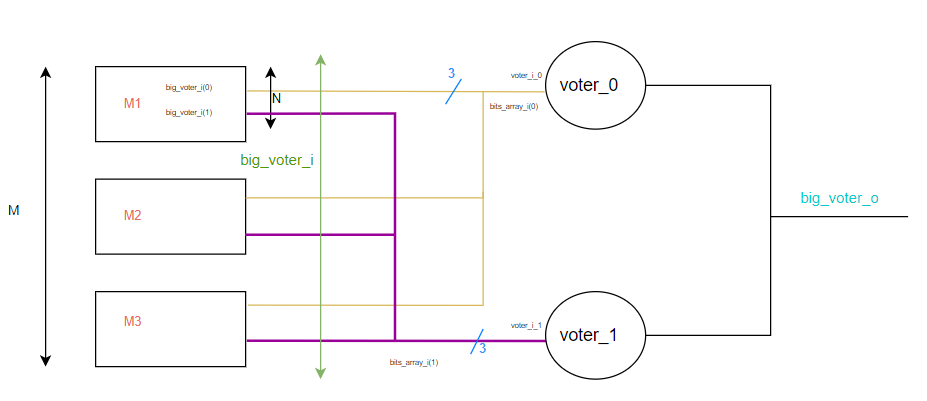
Slika 1. Blok šema FIR-a

FIR se sastoji iz MAC jedinica čiji broj određuje red filtra i koje se mapiraju na DSP ćelije. Proizvoljni red filtra je određen parametrom *fir\_ord,* broj obiraka sa *number\_samples\_g*, a širina podataka sa *input/output\_width.*

Na slici možemo vidjeti da se su ulazi u MAC modul:

* data\_i – odbirci ulaznog signala
* b\_i – koeficijenti filtra
* mac\_inter(i) – izlaz prethodnog MAC modula (za prvi MAC modul ovom signalu dodjeljujemo vrijednost 0 jer nema prethodnog)
* data\_o - izlazni podatak

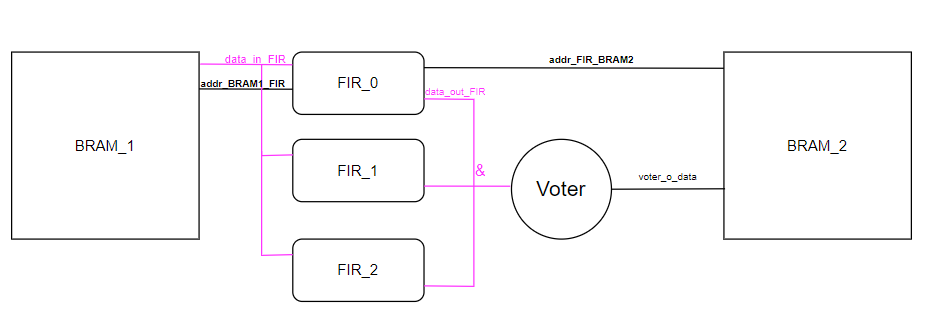
Zadatak ovog projekta jeste implementirati *N-modular Redundancy* tehnikuotpornosti na greške, koja podrazumijeva repliciranje FIR filtra proizvoljan (obavezno neparan) broj puta, i čiji se izlazi povezuju sa glasačem (*voter*). On izglasava za svaki bit podatka pojedinačno da li će biti 1 ili 0. Način na koji je logika glasača implementirana u ovom projektu prikazan je na slici 2. Parametar *M* predstavlja broj redudatnih modula, a *N* broj bita izlaznog podatka (zapravo jednak *output\_width*).



Slika 2. Prikaz logike glasača

Glasač (*big\_voter*) se sastoji iz *voter\_N* malih glasača koji porede određeni bit podatka od različitih modula. Broj malih glasača je jednak broju bita izlaza FIR-a (*output\_width*), a širina ulaza za svaki - broju modula (*M*). *I*-ti bit svih modula se konkatenira i proslijedi do *voter\_i*, koji prebroji da li je većina bita jednaka 0 ili 1, i tu vrijednost postavi na svoj jednobitan izlaz. Na kraju, izlazi malih glasača se konkateniraju i formiraju izlazni podatak. Ova redudantna tehnika je tolerantna za M/2 grešaka, jer će za svaki M/2 + 1 isti bit, i kada je netačan, on biti izglasan većinom.

Prikaz blok šeme i korišćenih komponenti u cijelom sistemu se nalazi na slici 3. Istaknut je način komunikacije sa memorijom, gdje se vidi da prvi FIR modul kontroliše čitanje i upis podataka.

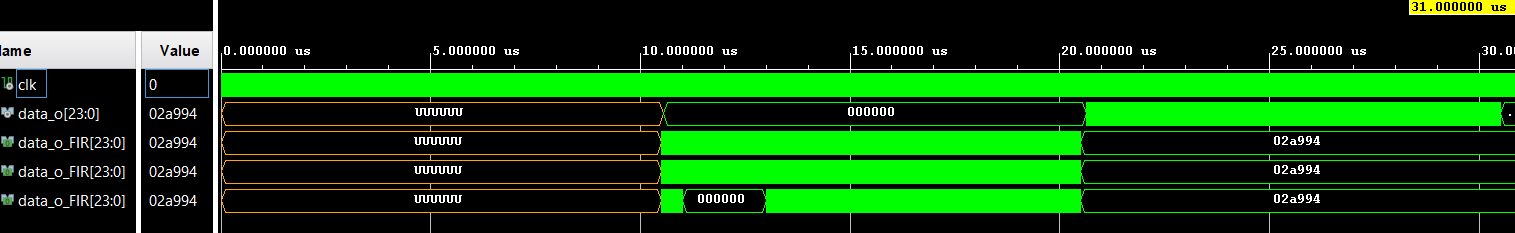


Slika 3. Blok šema sistema sa redudantnim komponentama i memorijom

# 2. Simulacija i forsiranje grešaka

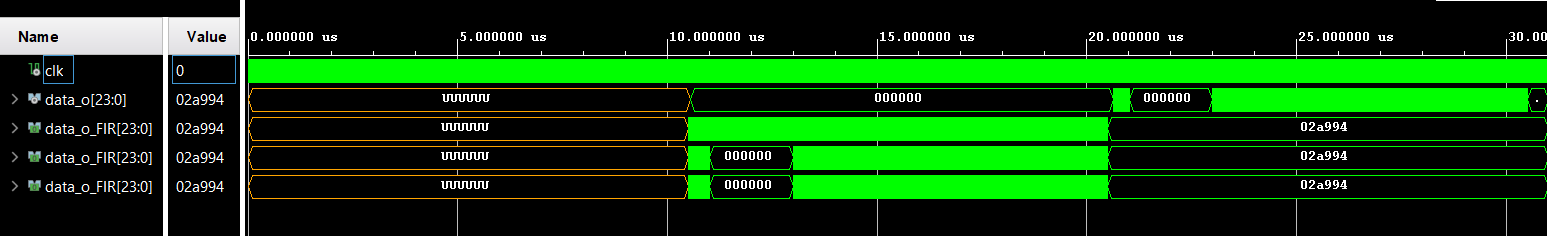
Funkcionalnost razvijenog sistema je provjerena korišćenjem testbenča. U njemu se prvo učitavaju koeficienti iz fajla *coef.txt,* a zatim i odbirci ulaznog signala iz fajla *input.txt* koji se upisuju u BRAM memoriju. Nakon toga se signal *start* postavi na 1, što daje znak FIR filtru da može da započne obradu podataka i spusti *ready* signal na 0. On iščitava podatke iz prvog BRAM-a, filtrira ih i nakon izglasavanja većine zapisuje izlazne podatake u drugi BRAM. U testbenču se čeka dok ne očita da se ready signal podigao na 1, nakon čega se iščitavaju podaci iz drugog BRAM-a. Radi provjere ispravnosti izlaznih odbiraka filtra, generisan je *Matlab* program koji implementira FIR i dobijene rezultate čuva unutar fajla *expected.txt*. Te rezultate poredimo sa iščitanim rezultatima iz BRAM-a, i u slučaju da se razlikuju, simulacija se prekida sa generisanjem poruke “*Result mismatch!*”.

Kako bi se provjerila tolerantnost sistema na greške koristi se *tcl* skripta *force.tcl*. Ona je testirana za različite parametre i za forsiranje grešaka u različitim djelovima dizajna, pri čemu je rezultat uvijek bio ispravan. Na sledećim slikama će biti prikazana simulacija za slučaj: *fir\_ord* =20, *input/output\_width* =24, *number\_samples\_g* = 1000, *M*=3. Na slici 4. su prikazani signali u *waveform* prozoru za slučaj forsiranja izlaza trećeg FIR modula na 0, u periodu od 10000ns – 12000ns. Na izlazu sistema se vidi da su zadržane ispravne vrijednosti, odnosno da sistem toleriše greške u M/2 modula.



Slika 4. Prikaz signala za uspješno tolerisanje unijete greške

Na slici 5. je prikazano neupješno tolerisanje unijete greške za *N-modular Redudancy* tehniku, odnosno za slučaj greške unutar M/2 +1 ili više modula. U ovom primjeru forsirani su izlazi i drugog i trećeg modula FIR-a na 0, za isti vremenski interval kao i u prošlom primjeru. Može se primijetiti da je i izlazni signal generisao grešku, tako da se simulacija završava sa report porukom “*Result mismatch!*”.



Slika 5. Prikaz signala za neuspješno tolerisanje unijete greške

# 3. Analiza utrošenih resursa i frekvencije rada

Kako bi prikazali da su MAC jedinice ispravno mapirane na DSP-ijeve, širina ulaznih/izlaznih podataka mora biti manja ili jednaka 16 bita , inače će alat iskoristiti više DSP jedinica za mapiranje. Stoga su u tabeli 1. prikazani rezultati analize resursa i frekvencija za slučaj 16-bitne širine ulaznih/izlaznih podataka. Pri analizi se koristi 500 odbiraka ulaznog signala. U tabeli 2. je prikazana analiza za 24-bitnu širinu podataka.

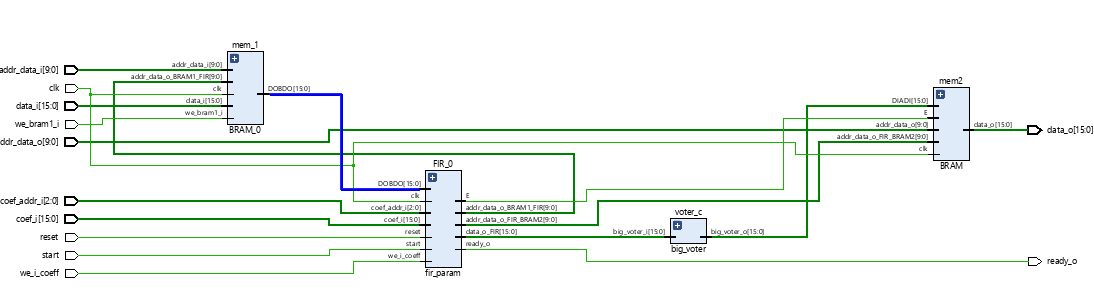
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **M**  (broj redudansi) | **fir\_ord** | **LUT** | **FF** | **BRAM** | **DSP** | **f** [MHz] |
| 1 | 5 | 46 | 21 | 1.0 | 5 | 225 |
| 20 | 59 | 19 | 1.0 | 20 | 222 |
| 3 | 5 | 166 | 57 | 1.0 | 15 | 222 |
| 20 | 182 | 57 | 1.0 | 60 | 222 |
| 5 | 5 | 362 | 95 | 1.0 | 25 | 212 |
| 20 | 383 | 1447 | 1.0 | 100 | 196 |

Tabela 1. Analiza utrošenosti resursa sa 16-bitnom širinom podataka

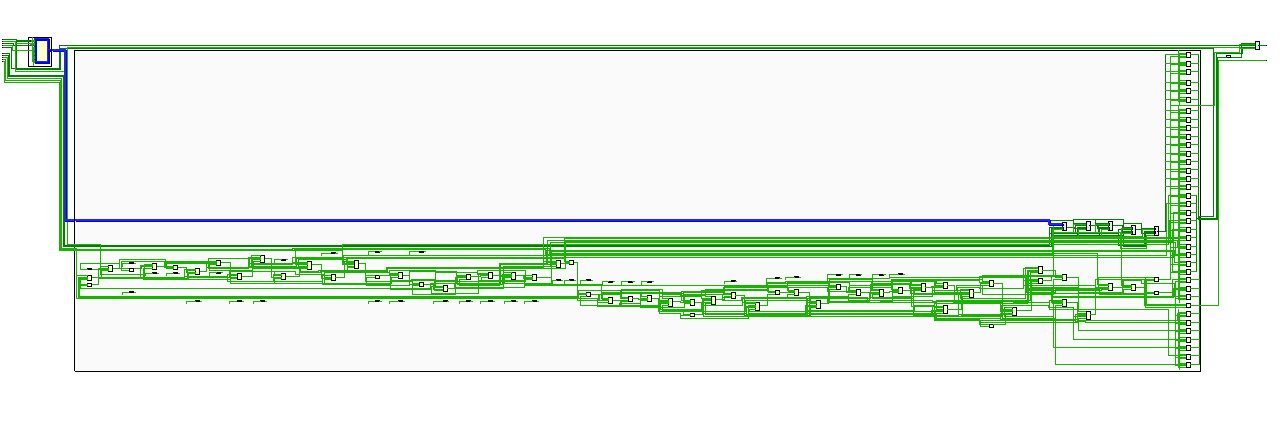
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **M**  (broj redudansi) | **fir\_ord** | **LUT** | **FF** | **BRAM** | **DSP** | **f** [MHz] |
| 1 | 5 | 54 | 21 | 1.0 | 15 | 135 |
| 3 | 5 | 209 | 57 | 1.0 | 45 | 135 |
| 5 | 5 | 474 | 95 | 1.0 | 75 | 133 |

Tabela 2. Analiza utrošenosti resursa sa 24-bitnom širinom podataka

Red filtra povećava broj iskorišćenih DSP-ijeva i LUT-ova, ali ne mijenja mnogo frekvenciju. Možemo primijetiti da se i sa povećavanjem broja redudantnih modula takođe povećava utrošenost resursa, a frekvencija malo smanjuje. Kritična putanja je razlog tome, jer je za sve slučajeve slična i ne povećava se sa povećanjem modula ili koeficienata. Na slikama 6. i 7. prikazana je radi jednostavnosti za samo jedan modul FIR-a.

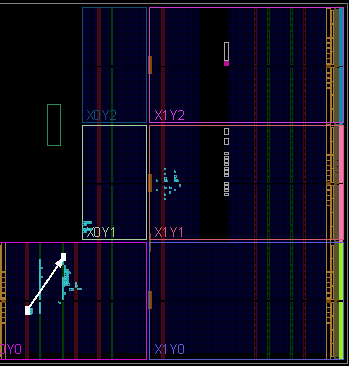


Slika 6. Kritična putanja na dizajnu



Slika 7. Kritična putanja na šematiku

Dakle nalazi se između memorije i MAC modula unutar FIR-a (ide od RAM-a unutar memorije, zatim kroz FIR,dolazi do MAC jedinice, i registra *a\_reg* koji čuva vrijednost izlaza). Na slici 8. prikazana je na dizajnu čipa FPGA.



Slika 8. Kritična putanja prikazana na FPGA čipu