Zadatak – Verilog

Sastaviti na jeziku Verilog modul koji predstavlja jednostavan računarski sistem. Sistem se sastoji od procesora i memorije. Koristiti prekidač SW9 za potrebe asinhronog reseta.

1. Faza (6 poena, pločica) – Aritmetička jedinica

Implementirati aritmetičku jedinicu koja prihvata dva operanda. U zavisnosti od vrednosti kontrolnog signala aritmetička jedinica vrši operaciju sabiranja ili operaciju oduzimanja operanada. Aritmetička jedinica pored rezultata operacije pruža i podatak o prenosu tj. pozajmici. Širina operanada aritmetičke jedinice mora biti skalabilna.

Ukoliko četvrta faza nije u potpunosti ispravno urađena, za potrebe ocenjivanja se mora izvršiti povezivanje aritmetičke jedinice sa prekidačima, dugmićima i diodama na odgovarajući način koji će omogućiti prikaz svih njenih funkcionalnosti.

2. Faza (6 poena, pločica) - Registar

Implementirati registar sa mogućnošću paralelnog upisa i inkrementiranja. Širina registra mora biti skalabilna.

Ukoliko četvrta faza nije u potpunosti ispravno urađena, za potrebe ocenjivanja se mora izvršiti povezivanje registra sa prekidačima, dugmićima i diodama na odgovarajući način koji će omogućiti prikaz svih njegovih funkcionalnosti.

3. Faza (6 poena, pločica) - Memorija

Implementirati memorijski modul koji koristi interne memorijske elemente FPGA čipa. Alat za sintezu Quartus II prepoznaje tj. zaključuje da dizajner zahteva interne memorijske elemente FPGA čipa ukoliko je izlaz memorijskog modula sinhron u odnosu na signal takta. Veličina memorijskog modula i širina memorijske reči moraju biti skalabilni.

Ukoliko četvrta faza nije u potpunosti ispravno urađena, za potrebe ocenjivanja se mora izvršiti povezivanje memorijskog modula sa prekidačima, dugmićima i diodama na odgovarajući način koji će omogućiti prikaz svih njegovih funkcionalnosti. Inicijalna vrednost memorijskih lokacija unutar memorijskog modula se može definisati prilikom konfiguracije FPGA čipa koristeći datoteku za inicijalizaciju memorije (memory initialization file). Naziv datoteke za inicijalizaciju memorije je neophodno navesti kao **atribut sinteze** uz **signal koji predstavlja niz memorijskih lokacija**:

```
(* ram_init_file = MIF_FILE_NAME *) reg [7 : 0] mem_signal [255 : 0];
```

4. Faza (6 poena, pločica) – Procesor (prvi deo)

Implementirati jednostavan osmobitni procesor koristeći prethodno implementirane module. Procesor ima osam bitova za adresiranje memorijskih lokacija, osam bitova za čitanje podataka iz memorije, osam bitova za upis podataka u memoriju, signal za upis podataka u memoriju, osam pinova za čitanje podataka iz ulazno/izlaznog adresnog prostora (na koje su povezani prekidači), osam pinova za upis

podataka u ulazno/izlazni adresni prostor (na koje je povezan registar koji pogoni diode) i signal za upis podataka u ulazno/izlazni adresni prostor. Procesor poseduje LOAD/STORE arhitekturu i osmobitni akumulator. Instrukcijski set se sastoji od jednobajtnih i dvobajtnih instrukcija datih u nastavku:

```
|0|0|0|0|X|X|X|X|   |8|b|i|t|A|d|d|r|    LD (LOAD)          : ACC <= mem(address)

|0|0|0|1|X|X|X|X|   |8|b|i|t|A|d|d|r|    ST (STORE)         : mem(address) <= ACC

|0|1|0|0|X|X|X|X|   |8|b|i|t|A|d|d|r|    ADD (ADDITION)     : ACC <= ACC + mem(address)

|0|1|0|1|X|X|X|X|   |8|b|i|t|A|d|d|r|    SUB (SUBTRACTION) : ACC <= ACC - mem(address)

|0|0|1|0|X|X|X|X|   IN : ACC <= SW[7:0]

|0|0|1|1|X|X|X|X|   OUT : LED_REGISTER <= ACC

|1|1|1|1|X|X|X|X|   TRAP
```

Instrukcija TRAP zaustavlja izvršavanje instrukcija sve dok se ne pritisne taster BTN2.

5. Faza (6 poena, pločica) – Procesor (drugi deo)

Proširiti instrukcijski set procesora iz prethodne faze instrukcijama uslovnog skoka i bezuslovnog skoka koje su realizovane kao apsolutni skokovi (skokovi na zadatu adresu).

```
|1|0|0|0|X|X|X|X|   |8|b|i|t|A|d|d|r|    JZ (IF ZERO) : if(ZF == 1) PC <= address

|1|0|0|1|X|X|X|X|   |8|b|i|t|A|d|d|r|    JNZ (IF NOT ZERO) : if(ZF == 0) PC <= address

|1|0|1|0|X|X|X|X|   |8|b|i|t|A|d|d|r|    JMP (UNCONDITIONAL) : PC <= address
```