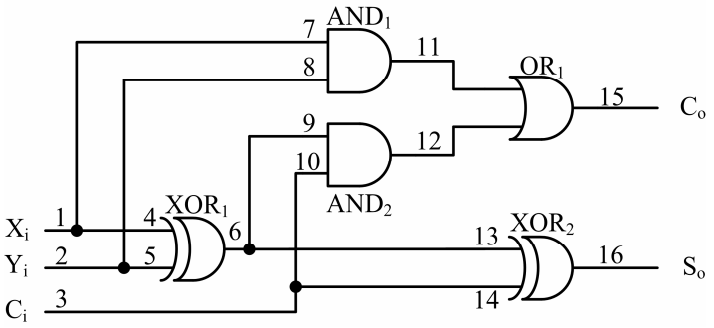
Laboratorijska vežba 1

Modelovanje grešaka u kombinacionim mrežama

Zadatak 1: Modelovanje grešaka i generisanje testova za kolo punog sabirača

Puni sabirač (*full adder*, FA) predstavlja osnovnu gradivnu jedinicu prilikom formiranja mreža za izvođenje složenijih aritmetičkih operacija, na primer sabirača, množača, delitelja, itd. Pun sabirač zapravo predstavlja kombinacionu mrežu sa tri ulaza, x_i, y_i, c_i, i dva izlaza, s_o, c_o.
Postoji čitav niz *gate-level* implementacija punog sabirača. Mi ćemo ovde razmatrati implementaciju prikazanu na Slici 1.



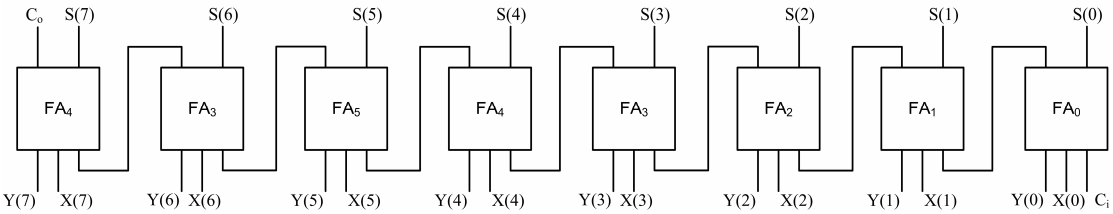
Slika 1. *Gate-level* implementacija punog sabirača

Za gate-level implementaciju punog sabirača sa Slike 1:

1) Odrediti ukupan broj *stuck-at* (SA) kvarova koji se mogu definisati za kolo punog sabirača.
2) Napisati VHDL model, uzimajući u obzir da su svi ulazni i izlazni signali 1-bitni.
3) Napisati *testbench* pomoću kojega je moguće modelovati pojavu SA kvarova na svakoj od veza koje postoje u kolu (uključujući primarne ulaze, primarne izlaze i unutrašnje veze). U ovu svrhu potrebno je koristiti *signal_force* i *signal_release* komande dostupne u ModelSim simulatoru. Detalji o načinu korišćenja ovih komandi mogu se pronaći u „**ModelSim User's Manual**“ dokumentu na stranama 755-761, koji se može pronaći u **...\modeltech_6.5\docs\pdfdocs** direktorijumu.
4) Koristeći razvijeni *testbench* i model punog sabirača generisati test vektore potrebne za detekciju SA kvarova u kolu punog sabirača. Kao algoritam pomoću kojega će biti generisani test vektori koristiti algoritam koji je predstavljen u okviru trećeg predavanja, „**Modeli kvarova i generisanje testova za njihovu detekciju u digitalnim kolima**“, na slajdovima 43-45. Proveriti da li je moguće deektovati sve SA kvarove koji se mogu pojaviti u kolu punog sabirača, i ukoliko to nije moguće, navesti koji SA kvarovi se ne mogu detektovati.
5) Koristeći algoritam za pronalaženje minimalnog skupa test vektora, predstavljen u okviru trećeg predavanja, „**Modeli kvarova i generisanje testova za njihovu detekciju u digitalnim kolima**“, na slajdovima 47-48, odrediti minimalni broj test vektora potreban za detekciju SA kvarova u kolu punog sairača.

Zadatak 2: Modelovanje grešaka i generisanje testova za kolo 8-bitnog *ripple-carry* sabirača

Kao što je poznato od ranije, najjednostavniji sabirač višebitnih brojeva može se formirati kaskadnim vezivanjem odgovarajućeg broja FA kola prikazanih na Slici 1. Ovako formirani sabirač poznat je pod imenom *ripple-carry* sabirač. U slučaju 8-bitnih operanada konceptualni blok dijagram *ripple-carry* sabirača prikazana je na Slici 2.



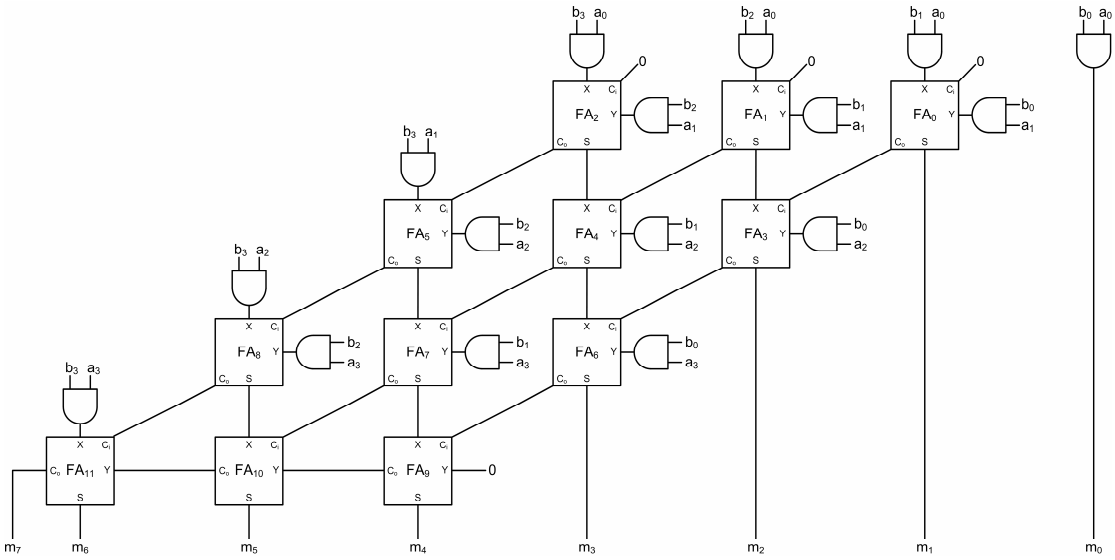
Slika 2. Blok dijagram 8-bitnog *ripple-carry* sabirača

Za 8-bitni sabirač prikazan na Slici 2:

1) Odrediti ukupan broj *stuck-at* (SA) kvarova koji se mogu definisati.
2) Napisati VHDL model, pri čemu je potrebno koristiti VHDL model punog sabirača razvijen u zadatku 1.
3) Napisati *testbench* pomoću kojega je moguće modelovati pojavu SA kvarova na svakoj od veza koje postoje u kolu (uključujući primarne ulaze, primarne izlaze i unutrašnje veze). U ovu svrhu potrebno je koristiti *signal_force* i *signal_release* komande dostupne u ModelSim simulatoru. Detalji o načinu korišćenja ovih komandi mogu se pronaći u „**ModelSim User's Manual**" dokumentu na stranama 755-761, koji se može pronaći u **...\modeltech_6.5\docs\pdfdocs** direktorijumu.
4) Koristeći razvijeni *testbench* i model 8-bitnog *ripple-carry* sabirača generisati test vektore potrebne za detekciju SA kvarova u kolu 8-bitnog *ripple-carry* sabirača. Kao algoritam pomoću kojega će biti generisani test vektori koristiti algoritam koji je predstavljen u okviru trećeg predavanja, „**Modeli kvarova i generisanje testova za njihovu detekciju u digitalnim kolima**", na slajdovima 43-45. Proveriti da li je moguće deektovati sve SA kvarove koji se mogu pojaviti u kolu 8-bitnog *ripple-carry* sabirača, i ukoliko to nije moguće, navesti koji SA kvarovi se ne mogu detektovati.
5) Koristeći algoritam za pronalaženje minimalnog skupa test vektora, predstavljen u okviru trećeg predavanja, „**Modeli kvarova i generisanje testova za njihovu detekciju u digitalnim kolima**", na slajdovima 47-48, odrediti minimalni broj test vektora potreban za detekciju SA kvarova u kolu 8-bitnog *ripple-carry* sabirača.

Zadatak 3: Modelovanje grešaka za kolo 4-bitnog *array* množača

Brzi paralelni množač takođe se može projektovati korišćenjem odgovarajućeg broja punih sabirača i dodatnih logičkih kola. Na Slici 3, prikazan je blok dijagram 4-bitnog *array* množača. Kao što se sa slike može videti množač je formiran korišćenjem dvodimenzionalne mreže punih sabirača uz upotrebu dodatnih dvoulaznih I logičkih kapija.



Slika 3. Blok dijagram 4-bitnog *array* množača

Za 4-bitni *array* množač prikazan na Slici 3:

1) Odrediti ukupan broj *stuck-at* (SA) kvarova koji se mogu definisati.
2) Napisati VHDL model, pri čemu je potrebno koristiti VHDL model punog sabirača razvijen u zadatku 1.
3) Napisati *testbench* pomoću kojega je moguće modelovati pojavu SA kvarova na svakoj od veza koje postoje u kolu (uključujući primarne ulaze, primarne izlaze i unutrašnje veze). U ovu svrhu potrebno je koristiti *signal_force* i *signal_release* komande dostupne u ModelSim simulatoru. Detalji o načinu korišćenja ovih komandi mogu se pronaći u „**ModelSim User's Manual**" dokumentu na stranama 755-761, koji se može pronaći u **...\modeltech_6.5\docs\pdfdocs** direktorijumu.
4) Koristeći razvijeni *testbench* i model 4-bitnog *array* množača generisati test vektore potrebne za detekciju SA kvarova u kolu 8-bitnog *ripple-carry* sabirača. Kao algoritam pomoću kojega će biti generisani test vektori koristiti algoritam koji je predstavljen u okviru trećeg predavanja, „**Modeli kvarova i generisanje testova za njihovu detekciju u digitalnim kolima**", na slajdovima 43-45. Proveriti da li je moguće deektovati sve SA kvarove koji se mogu pojaviti u kolu 4-bitnog *array* množača, i ukoliko to nije moguće, navesti koji SA kvarovi se ne mogu detektovati.
5) Koristeći algoritam za pronalaženje minimalnog skupa test vektora, predstavljen u okviru trećeg predavanja, „**Modeli kvarova i generisanje testova za njihovu detekciju u digitalnim kolima**", na slajdovima 47-48, odrediti minimalni broj test vektora potreban za detekciju SA kvarova u kolu 4-bitnog *array* množača.