**„Kratka“ uputa za četvrti labos, (by eott25)**

Zadana je funkcija f(A,B,C,D), koju moramo realizirati multipleksorom 16/1. Na raspolaganju imamo multipleksore 2/1, pa kako ćemo to napraviti?

Prvo, da skužimo princip po kojem funckioniraju multipleksori, naime, kad uzmemo za primjer ovaj naš mali 2/1, on na sebi ima 2 podatkovna ulaza, 1 adresni, enable i izlaz f. Multipleksor radi kao sklopka tako da u ovisnosti o vrijednosti adresnog bita, kojeg ćemo nazvati S0, multipleksor na izlaz spaja onaj ulaz koji po poretku odgovara vrijednosti s0.

E

D0 0

f

D1 1

S0

Ako je S0=0, multipleksor kao sklopka propušta D0 i njegovu vrijednost stavlja u f, znači ako je D0 =1, a S0=0-> izlaz će biti 1.

E

D0 =1 0

f=1

D1 1

S0=0

E sad imamo još nešto, imamo ulaz E(enable) taj ulaz nam govori hoće li uopće naš multipleksorčić raditi, ako je Enable =1, onda nam multipleksor radi, i za navedene vrijednosti D0 i S0( D0=1,S0=0) izlaz bi bio 1, no ukoliko je E=0, izlaz bi bio 0 bez obzira na vrijednosti D0 i S0. Ako pobrojimo ulaze, na multipleksoru 2/1 imamo 4 ulaza i 1 izlaz.

Napravimo sad istu stvar za multipleksor 4/1, idemo razmisliti koliko bi on mogao imati ulaza i izlaza. Već smo prije utvrdili da prvi broj u nazivu tipa multipleksora znači broj podatkovnih bitova, nadalje imamo onaj adresni bit s0 i enable, pa kako bi sad ovo funkcioniralo?

E

D0 0

D1 1

f

D2 ?

D3 ?

S0

Zašto upitnici, problem je sada što imam 4 bita, znači četiri ulaza, moja sklopka S0 može poprimiti samo vrijednosti 0 i 1, a treba mi četiri kombinacije, kako ću to riješiti?

E

D0 00

D1 01

f

D2 10

D3 11

S1 S0

Doveo sam još jedan adresni bit, pošto i on može poprimiti vrijednosti 0 i 1, sada mogu ostvariti sve 4 kombinacije, 00,01,10,11 ☺ s tim da mi je teži bit u ovom slučaju s1, pa kad bih upisao s1=1, s0=0, izlaz bi bila vrijednost na ulazu D2.

Mogli smo reći da je i s0 najteži bit( ili samo zamijeniti ulaze adresnih bitova) i onda bi nam za vrijednost s1=1 i s0=0 izlaz bila vrijednost na ulazu D1.

Multipleksor 4/1 može se napraviti i od više manjih 2/1 multipleksora, tada oni svi zajedno čine multipleksorsko stablo, primjetite da broj podatkovnih ulaza na multipleksoru može biti uvijek samo potencija broja dva (2/1, 4/1,16/1...) a izlaz je bez obzira na broj ulaza uvijek jedan. Naravno, sveukupan broj ulaza je broj podatkovnih+ broj adresnih+1 (enable).

Naše multipleksorsko stablo izgleda ovako:

E

D0 0

E

D1 1

0 f

1

E

D2 0 S1

D3 1

S0

E sad ,kako će funkcionirati ovo, recimo da je Enable= 1 na svim multipleksorima, znači, svaki će dobro raditi. Na ulaze D(0) do D(3) ( usput, ovako kako sam ja označio D-ove, tako se označavju vektorski kada trebate u Arhitekturi označiti gjde priljučiti koji od bitova vektora, možda vam sada to ne znači mnogo, ali kad budete radili labos u Vhdl-u, trebat će vam ;) ), znači na sva ta 4 ulaza dovedene su neke vrijednosti, i sad krenemo od kraja, za s0 uzmemo naprimjer vrijednost 0( s0=0), i što će nam ići dalje?

Dakle, s0 je priključen na 2 multipleksora, ti multipleksori su u istoj liniji i oni su znači pod kontrolom adresnog bita s0, tj. S0 odlučuje što gdje kako kada :D. Znači s0 je 0, on ide na najgornji multipleksor i gleda, moram pustiti onu vrijednost koja dolazi na ulazu 0, a je vrijednost D0. I sad s0 razmišlja što još mora učiniti, i vidi da je spojen i na doljnji multipleksor, tamo isto mora na izlaz propustiti ono što je na 0-tom ulazu, a to je vrijednost od D2, dakle prošli su D0 i D2. Te oni dolaze na završni multipleksor.

E

D0 0

D0 E

D1 1

D0 0 f

D2 1

E

D2 0 S1

D2

D3 1

S0

I sada koji od njih dvojce će izići na izlaz f, to naravno ovisi o s1, dakle, ako je s1 =1, prolazi donji, odnosno, prolazi D2, ako s1 = 0 prolazi d0;

Idemo sad jedan primjer: recimo na ulaz nam je doveden D=“1011“, što će biti izlaz ako nam je s=“01“, što ovo znači, uzimat ću da se D definira ovako D0D1D2D3, a ne obrnuto, znači iz toga-> D0=1,D1=0,D2=1,D3=1; nadalje, s mi je s1s0, znači s1=0, a s0=1, i sada idemo po redu( uzimamo da je E= 1) Prvo gledamo vrijednost s0, on je jedan, znači propuštaju se vrijednosti na ulazima koji su označeni sa 1, a to su D1=0 i D3=1, dalje, te vrijednosti dolaze na završni multipleksor kojeg kontrolira s1, a njegova je vrijednost 0, znači izlazi gornji ulaz, a to je D1; i izlaz f= 0 onda. Primjetimo da s=“01“ u dekadskom zapisu ima vrijednost 2, a nama je na kraju izašao drugi po redu izlaz D1( prvi je D0), znači taj vektor s nam određuje vrijednost kojeg ulaza nam izlazi van.

E

D0=1 0

i0=0 E

D1=0 1

0

1 f=0

E

D2=1 0 S1=0

i1=1

D3=1 1

S0=1

Označio sam izlaze ova dva multipleksora koja su u liniji sa i0 te sa i1 zbog toga što je praksa da se tako označavaju signali, jer je napravilno napisati da je i ulaz i izlaz nekog sklopa ista varijabla, a i praktičnije je ovako ;), ugl, signal nije onaj početni ulaz, on je samo zadobio vrijednost toga ulaza **(ovo je podvučeno zato što nisam siguran jel točno ili samo pričam gluposti)**

Bacimo se na našu vježbu:

Znamo da veće multipleksore možemo izgraditi od manjih multipleksora tako da od više tih manjih napravimo multipleksorsko stablo. Što znači da je multipleksor tipa 16/1.

To znači da on ima na ulazu 16 podatkovnih ulaza i jedan podatkovni izlaz, no uz to, na sučelje tog našeg 16/1 multipleksora dolaze i adresni bitovi , te bit E (enable) koji nam govori hoće li naši multipleksorčići(2/1) uopće raditi ili ne. A koliko imamo adresnih bitova, i koji su to bitovi?

Pošto naša funkcija ima 4 varijable, to su A,B,C i D. To znači da će naše sučelje isto tako imati 4 adresna bita. Kao što smo mogli primijetiti 21 broj ulaza-> 1 adresni, 2 2-> 2 adresna bita, tako ćemo i za 16/1 imati 4 adresna bita( napomena: ovaj broj adresnih bitova je zbog ovako postavljenog zadatka, da smo naprimjer ovu funkciju morali ostvariti muxom 8/1, koristili bi 3 adresna bita, a imamo f(a,b,c,d), znači, četiri varijable, to znači da postoji mogučnost da nam se osim nula i jedinica na ulazima nađe i vrijednost varijable koja nije adresni bit, recimo a,b,c su adresni, na ulazu nam se može pojaviti onda d ili not d osim nula i jedinica, ali više o tome imate na auditornim kod Čupića (ferko ) i u ostaloj literaturi)

Znači 4 adresna bita, dakle imat ćemo 4 linije muxova(multipleksora) u m

Multipleksorskom stablu. Mogli smo ovaj zadatak riješiti i s 1 multipleksorom 16/1, ali nama je zadano da takav multipleksor moramo izgraditi s 2/1 multipleksorom, pa da vidimo koliko ih je. U zadnjoj liniji( vrh stabla) se nalazi jedan multipleksor i njegov izlaz je ujedno i izlaz funkcije, na njega je od adresnih bitova spojen onaj naviše težine, ako imamo s3s2s1s0, tamo će biti spojen s3, no ako adresne označimo s s0s1s2s3, onda je najviše težine s0.

Zašto spominjem obje verzije, pa u zadatku nam piše da vektor moramo zapisati kao

Imevektora: std\_logic\_vektor(0 to n) , znači kad bi zapisali u s da je jednako 1000, onda bi dobili s0s1s2s3=1000, znači, možemo gledati s0 kao najteži bit, ili možemo u testiranju i simulaciji našeg sklopa samo zapisivati brojeve unatrag. Dakle ako nas asistent pita da isprobamo možemo li vrijednost ualza d1 dovesti na izlaz f, nećemo zapisati 0001, nego 1000, što inače ima vrijednost 8... ugl, stvara problem, ali za tu verziju sam siguran da je ispravna. Za verziju s0s1s2s3, dakle da je s0 najteži bit nisam siguran da ju smijemo koristiti, ali je puno jednostavnije jer ako vas traži da isprobamo hoće li prespojiti prvi ulaz na izlaz f, upisali bi ono što i ima vrijednost ulaza 1,a to je s=“0001“ s0s1s2s3=0001. Budem provjerio smije li se ta jednostavnija verzija pa javim, ja sam osobno koristio tu drugu verziju ☺

Znači na vrh stabla ide najteži bit, na drugu s lijeva liniju koja se sastoji od 2 multipleksora dovodimo drugi najteži bit, na treću liniju koja se sastoji od 4 multipleksora 3. najteži bit ( 4 ih dovodimo jer svaki od ova dva u drugoj liniji i imaju po 2 ulaza, a svaki mux 2/1 ima jedan izlaz, da bi se na svaki ulaz spojio po jedan izlaz nekog multipleksora moramo u treću staviti 4 muxa), a u zadnju liniju ide 8 mux-a, primjetimo 8 mux-a, svaki po dva ulaza, to je sveukupno 16 ulaza, što smo i htjeli dobiti. To su ulazi vekotora D, D: IN STD\_LOGIC\_VECTOR(0 TO 15), na svaki od tih ide Enable, ali u zadatku se uzima da je E: IN STD\_LOGIC, pošto nije vektor znači da će svi muxovi imati istu vrijednost E-a, znači ili svi rade ili ni jedan ne radi ☺, ako E postavimo u 0, neće ni jedan mux raditi, izlaz f mora biti 0.

Broj signala bit će vam broj izlaza iz svih multipleksora- glavni izlaz, pošto svaki multipleksor ima 1 izlaz, broj mux-a je 8+4+2+1= 15, znači 15-1= 14, dakle imamo 14 signala, definicija je

SIGNAL i: std\_logic\_vector(0 to 13);

Svaku liniju naših multipleksora znači kontrolira po **jedan** adresni bit!

Na ulaze dolazi vektor D, vrijednost D-a je ona vaša funkcija koju dobijete u ovisnosti o vašem JMBAG-u, znači samo dignete 1 u testbenchu tamo gdje su mintermi, i to su ti ulazi, a adresni vam kažu koji od tih ulaza izlazi, ako vrijednost adresnih odgovara onom mintermu koji ima vrijednost 1, izlaz će biti jedan jer će vrijednost tog minterma s nizom sklopki biti spojena na izlaz. Usput, kašnjenje izlaza mora biti sveukupno 40 ns, dakle nemojte se čuditi što kasni, svaki mux ima kašnjenje 10 ns, a pošto moramo proći kroz 4 linije, 4\*10=40

I ostatak morate sami jer ako vam do kraja sve riješim nećete imati pojma na eliminacijskim pitanjima i neće vam vrijediti trud, nadam se da sam nekome bio od pomoći! ☺

**ZA SVAKU INFORMACIJU OVDJE KOJA VAM JE SUMNJIVA, PROVJERITE JU, JA ISTO PRVI PUT RADIM OVAJ LABOS I POSTOJE MOGUĆNOSTI DA SAM NEŠTO KRIVO REKAO, JA SMATRAM DA TAKO TREBA IĆI, MOŽDA SE GRDNO VARAM, AKO BUDU MASOVNE, TOPLO PREPORUČAM, I BTW. Pripremite pitanja za masovne da sve što vam nije jasno tamo riješite**  ;)