LABORATORIJSKI ZADATAK 4

Složeni digitalni sistemi – LCD banner

Potrebno predznanje

- Urađena pokazna vežba 5
- Osnovno razumevanje pojma protokola pri komunikaciji dva sistema
- Razumevanje ASCII kodova karaktera

Šta će biti naučeno tokom izrade vežbe?

U ovoj vežbi:

- Naučićete kako da napravite sistem koji komunicira sa drugim sistemom pomoću protokola,
- Implementiraćete sistem zasnovan na automatu koji komunicira sa drugim sistemom (LCD driver-om),
- Vežbaćete projektovanje složenih sistema sastavljenih od nekoliko modula,
- Simuliraćete složeni digitalni sistem i vežbati razumevanje simulacionih dijagrama složenih sistema.

Apstrakt i motivacija

Različiti sistemi često treba međusobno da komuniciraju. Kada priključite vašu Flash memoriju na USB port računara, PC i memorija treba da komuniciraju kako bi razmenjivali datoteke. Kada pričate sa vašim prijateljem, vaš računar i njen računar treba da komuniciraju i razmenjuju poruke. Komunikacija između sistema je moguća samo ukoliko obe strane poštuju isti protokol, tj. standard pri komunikaciji. U ovoj vežbi, projektovaćete sistem koji će komunicirati sa LCD driverom tako da vrši ispis karaktera na LCD ekranu E2LP platforme. Vaš sistem treba da prati sekvencu predefinisanih koraka u komunikaciji sa driverom, definisanu protokolom komunikacije sa njim. Takođe, napravićete i vrh hijerarhije sistema u kome će biti instancirani vaš sistem (banner), driver i clock generator.

Šta treba doneti na termin laboratorijske vežbe?

- Logičku i/ili blok šemu sistema na papiru ili računaru (za **LCD banner, tj. vašu komponentu unutar celog sistema**). *Koristite standardne kombinacione i sekvencijalne mreže kao komponente, nemojte crtati blokove koji su suviše apstraktni i nepotpuno definisani.*
- Graf prelaza stanja automata
- VHDL opis **krajnjeg koraka** u realizaciji sistema (međukoraci nisu potrebni) – krajnji korak je poslednji korak koji uspete da realizujete.
- Testbench za krajnji korak sistema.
- Generisanu *.bit* datoteku za konfigurisanje E2LP platforme za krajnji korak sistema.

TEORIJSKE OSNOVE

1. Protokol komunikacije sa LCD driverom

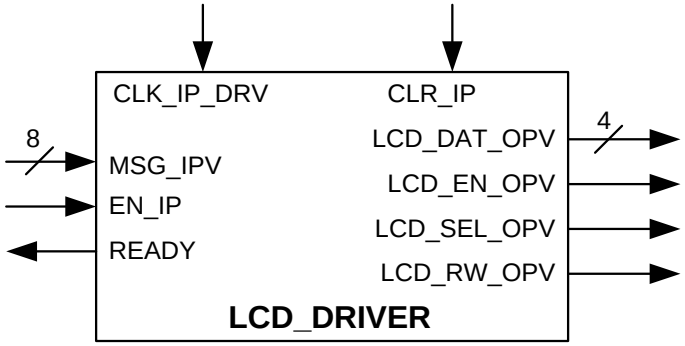
Često digitalni sistemi moraju da komuniciraju. Da li ste priključili nešto na USB port vašeg računara ili povezali dva računara, sistemi moraju da razmenjuju podatke na neki način. Jedini način da ova komunikacija ima šansu da bude uspešna je da svi sistemi koji međusobno komuniciraju poštuju isti skup pravila i koraka prilikom komunikacije. Taj skup zovemo **protokol** komunikacije između sistema.

Uglavnom se sistem sa kojim treba komunicirati ne poznaje sa stanovišta njegove arhitekture (što je i dobro prema principima abstrakcije i skrivanja informacija koje smo pominjali u prethodnoj vežbi). Komunikacija sa datim sistemom se obavlja preko njegovih prolaza. Tipično, vaš sistem treba da osluškuje vrednosti na nekim signalima i šalje podatke na drugim signalima. Nekada vaš sistem inicira komunikaciju, a nekada drugi sistem istu inicira na linijama koje vaš sistem osluškuje.

Često se komunikacija obavlja putem posrednika. Zadatak posrednika je da pojednostavi protokol komunikacije. Posrednici se nalaze na nivou apstrakcije koji je između vašeg sistema i sistema sa kojim komunicirate i pojednostavljuju protokol na nižem nivou apstrakcije (komunikacija sa sistemom) protokolom na višem nivou (komunikacija sa vašim sistemom) koji po pravilu ima jednostavniji skup signala. Korišćenjem posrednika, vi kao projektant možete da se koncentrišete na logiku vašeg sistema, a manje vremena da provedete analizirajući složen protokol komunikacije sa sistemom sa kojim želite da komunicirate.

U ovoj vežbi, vaš zadatak će biti da napravite sistem koji će ispisivati karaktere na LCD ekranu na E2LP platformi. Ovo se može uraditi pravljenjem sistema koji komunicira sa LCD kontrolerom koristeći 7 signalnih bita. Četiri od tih sedam bita su biti podataka, odn. karaktera koji se ispisuje, a preostala tri su kontrolni biti. Trebali bi da ste već primetili da ova komunikacija nije potpuno intuitivna, pošto su ASCII kodovi karaktera širine 7 ili 8 bita, a širina signala podatka prema LCD-u je 4 bita! Protokol komunikacije je verovatno prilično složen zbog ovoga.

Kako ne bi morali da radite sa ovim veoma neintuitivnim protokolom, dobićete posrednika u komunikaciji sa LCD kontrolerom – LCD driver. On će komunicirati sa LCD kontrolerom putem ovog komplikovanog protokola, a prema vama će postaviti jednostavnu spregu koja se sastoji od svega 2 kontrolna bita i 8 bita podatka, odn. karaktera koji želite da ispišete. Entitet ovog posrednika je prikazan na slici 1-1.



Slika 1-1. Posrednik pri komunikaciji sa LCD kontrolerom

Nakon sistemskog reseta, LCD driver inicijalizije LCD ekran i spremi ga za prihvatanje 2 reda od 16 karaktera. Driver koji ste dobili sa ovom vežbom inicira ekran tako da ne prikazuje kursor i spremi ga za prihvatanje 32 karaktera redom, koje ispisuje prvo u gornjem, pa u donjem redu, s leva na desno.

LCD driver je pisan poštujući princip skrivanja informacija tako da *ne moramo poznavati njegovu arhitekturu da bi ga koristili*. Međutim, entitet moramo poznavati jer ćemo sa njim komunicirati putem njegovih ulaza i izlaza. Pa analizirajmo njegov entitet. Kao što možete primetiti sa slike 1-1, sem standardnih takt i reset ulaza (CLK_IP_DRV je takt, a CLR_IP je reset), ovaj modul ima 7 izlaza koji se povezuju sa LCD kontrolerom i tih 7 bita izlaze iz FPGA i fizički su povezani sa ulazima LCD kontrolera na E2LP platformi. Ovih 7 bita služe za komunikaciju na nižem nivou, onim neintuitivnim protokolom koji želimo da izbegnemo. Sa leve strane predstave entiteta prikazano je 10 bita sa kojima treba da komunicira naš sistem. Ovi prolazi su:

- MSG_IPV – 8-bitni signal podatka koji predstavlja karakter koji želimo da ispišemo (ulaz)
- EN_IP – signal dozvole ispisa (ulaz)
- READY – signal spremnosti drivera da prihvati karakter (izlaz).

Samo posmatrajući ovaj skup signala, možemo zaključiti da je protokol komunikacije sa driverom mnogo intuitivniji nego onaj sa kontrolerom, jer podatak šaljemo kao 8-bitni vektor, što odgovara ASCII širini, a kontrola se vrši pomoću svega 2 bita.

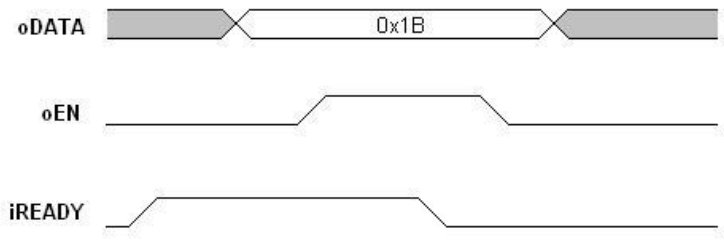
Signal dozvole ispisa se šalje **od naše komponente ka driveru**. Ovaj signal treba da generiše naš sistem i njime javi driveru da je karakter spreman na liniji podataka i da ga driver može preuzeti. Ovaj signal je aktivan na 1.

Signal spremnosti drivera se šalje **od drivera ka našoj komponenti**. Ovaj signal generiše driver onda kada je spreman da primi karakter od nas. Signal je takođe aktivan na 1. Ukoliko ovaj signal nije postavljen, driver će zanemariti karaktere koje mu mi šaljemo, tako da naša komponenta mora da sačeka da se ovaj signal aktivira, pre nego što pošalje karakter.

Okrenimo se sada našoj komponenti, koja treba da priča sa ovim driverom. Sa stanovišta komunikacije sa driverom, ona treba da ima sledeće prolaze:

- izlaz oDATA [7:0] koji povezujemo na MSG_IPV,
- izlaz oEN koji povezujemo na EN_IP,
- ulaz iREADY koji povezujemo na READY.

Primer vremenskog dijagrama komunikacije sa driverom je dat na slici 1-2.



Slika 1-2. Vremenski dijagram komunikacije sa LCD driverom

Kao što smo ranije rekli, naša komponenta ne sme da šalje karaktere driveru dok ne dobije signal da je driver spreman da te karaktere primi. Isto tako, ona ne sme slati karaktere brzo, jer će driver moći da ih preuzima samo jedan po jedan. Protokol komunikacije treba biti sledeći:

- Naša komponenta čeka signal READY,
- Kada se pojavi signal READY, naša komponenta postavlja karakter na liniju podatka i signalom dozvole javlja driveru da je prvi karakter spreman,
- Driver prihvata karakter i kreće da ga obrađuje, postavljajući signal READY na nulu,
- Naša komponenta mora da sačeka da se ovaj signal READY spusti na nulu, držeći isti karakter sve vreme,
- Nakon deaktiviranja READY signala, naša komponenta može da se pripremi za slanje novog karaktera,
- Ponovnim pojavljivaljem READY signala, šalje se sledeći karakter i ciklus se ponavlja.

Sem redovnih ASCII karaktera, LCD driver prihvata i specijalne *komandne karaktere* koji ne predstavljaju ASCII vrednost slova koji želimo da ispišemo nego odgovarajuću komandu koju šaljemo driveru. Jedina komanda koju ćemo koristiti u ovoj vežbi je:

- 0x1B – obriši sadržaj ekrana i postavi kursor na početak.

Dakle, pre pisanja bilo čega na ekranu, driveru treba poslati „karakter“ 0x1B kako bi se ekran spremio za ispis. Isti karakter treba poslati svaki put kada želite ispisati novi tekst na ekranu.

Na slici 1-3 dat je pregled ASCII kodova koje podržava naš LCD ekran. Ova tabela je nešto modifikovan ASCII, kako bi ekran podržao i još neke karaktere koji nisu u standardnoj ASCII tabeli.

Slika 1-3. Tabela karaktera

ZADACI

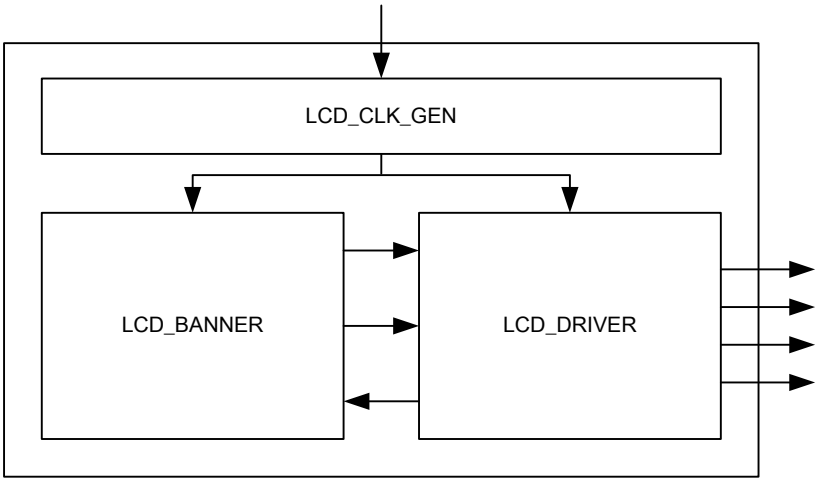
2. Statički tekst na LCD ekranu

Vaš prvi zadatak je da napišete vaše ime i prezime na LCD ekranu E2LP platforme. Ime treba biti ispisano u prvom redu, a prezime u drugom. Iako ovaj zadatak zvuči veoma jednostavno, on zahteva složen dizajn sistema koji prati protokol koji smo objasnili u prethodnoj sekciji.

Za početak, napravite novi projekat i usnimite ga na željenoj lokaciji.

U projekat dodajte opis LCD drivera (LCD_driver.vhd) koji ste dobili zajedno sa ovom vežbom. Postojeću datoteku u projekat možete ubaciti izborom **Project --> Add Source** i izborom odgovarajuće datoteke.

Na slici 2-1 je dat blok dijagram celog sistema. On se sastoji iz tri komponente – drivera koji ste upravo ubacili u projekat, generatora takta i LCD bannera koji će biti komponente koje ćete sami napraviti.



Slika 2-1. Vrh hijerarhije sistema

Nakon što ste pripremili teren, krenućemo sa lakšim delom i opisati generator takta. Napravite novu VHDL datoteku u vašem projektu i nazovite je LCD_CLK_GEN. Generator takta treba da ima sledeće prolaze:

- Ulazi: iCLK, inRST
- Izlazi: oCLK, onRST

Generator takta treba da podeli ulazni takt od 24 MHz na sporiji takt čija frekvencija treba da bude ispod 270 kHz. Ovo je zahtev od strane LCD kontrolera koji naš sistem mora da ispoštuje. Vaš zadatak je da odredite koji takt ćete koristiti i da ga generišete. Najjednostvniji način podele takta je pomoću brojača, na sličan način na koji smo podeli takt kod štoperice (drugi brojač se uvećavao jednom u sekundi, dakle tamo smo ulazni takt od 24 MHz podelili na 1 Hz). Jedan od bita brojača ćemo dodeliti izlaznom signalu oCLK. Ukoliko dodelite nulti bit brojača, izlazni takt će biti duplo sporiji (proverite zašto!). Izaberite odgovarajući bit tako da izlazni takt bude ispod 270 kHz.

Sem generisanja takta, generator mora da generiše i reset kako bi sve sinhrone komponente u sistemu bile resetovane ukoliko rade na ovom sporijem taktu. Za ovaj deo iskoristite proces i dodelu iz listinga 1-1. Ovaj proces zadržava reset nešto duže nego što ga korisnik drži pritisnutog.

Listing 1-1. Reset generator

```
process (iCLK, inRST) begin
    if (inRST = '0') then
        snRST <= '0';
    elsif (iCLK'event and iCLK = '1') then
        if (sCLK_DIV(7) = '1') then
            snRST <= '1';
        end if;
    end if;
end process;

onRST <= snRST;
```

Konačno, treća komponenta sistema je LCD banner – ovo je sistem koji treba da komunicira sa driverom i ispisuje karaktere. Sem redovnih ulaza takta i reseta, ova komponenta ima i prolaze za komunikaciju sa driverom: oDATA, oEN i iREADY kao što smo opisali u teorijskom delu.

LCD banner treba da bude zasnovan na automatu koji će pratiti protokol komunikacije sa driverom opisan u teorijskom delu vežbe. Ovog puta vaš je zadatak da projektujete ceo automat – definišete stanja, ulaze, izlaze, početno stanje, funkciju prelaza i funkciju izlaza.

Sem automata, banner treba da sadrži i neke pomoćne komponente, a vi se odlučite koje vam trebaju da bi sistem ispisivao karaktere. Preporučljivo je da iskoristite VHDL nizove za smeštanje konstante sa svim karakterima koje treba da ispisujete, a tokom rada sistema brojačem prolazite kroz niz i svaki put ispisujete drugi karakter, dok ne ispišete svih 32 (za svaki karakter treba ispratiti sekvencu iz protokola!). Index (brojač) treba da bude uvećavan svaki put kada jedan ciklus komunikacije bude završen, odn. u odgovarajućem stanju automata.

I kao poslednji korak, napravite četvrtu datoteku u kojoj ćete opisati vrh hijerarhije sistema. U vrhu hijerarhije treba samo instancirati tri komponente koje ste do sada ubacili u projekat, prema slici 2-1. Povezivanje komponenti treba izvršiti na sledeći način:

- Generator takta prihvata ulazni takt i reset i formira takt i reset za ostatak sistema,
- LCD banner radi na internom taktu i resetu i povezan je sa driverom signalima protokola,
- LCD driver radi na internom taktu i resetu, povezan je sa bannerom preko signala protokola, a generiše izlaz ka LCD kontroleru.

Povezivanje sa pinovima FPGA je sumirano u tabeli 2-1.

Table 2-1. Povezivanje prolaza na komponente E2LP platforme

Prolaz	Smer	Signal na E2LP platformi
iCLK	in	CLK
inRST	in	RESET
oLCD_DAT_OPV [3:0]	out	LCD_D [3:0]
oLCD_EN_OPV	out	LCD_EN
oLCD_RW_OPV	out	LCD_R/#W
oLCD_SEL_OPV	out	LCD_RS

Prolaze sistema povezati sa E2LP platformom prema Tabeli 2-1.

Pre pokretanja implementacije, sistem treba simulirati. Simulaciju možete uraditi na dva načina:

- simulirati samo LCD banner – ovo je dobar prvi korak provere da li banner radi korektno jer simulacija čitavog sistema nije lako čitljiva; pisanje test bencha je u ovom slučaju nešto složenije jer unutar test bencha treba da „odglumite“ driver, odn. njegov READY signal; preporučujemo da ovaj signal generišete periodično, slično kao što je generisan takt; ovo ne oslikava potpuno kako se on ponaša, ali može dobro da posluži kako bi proverili da li vaš banner prolazi kroz stanja onako kako želite.
- simulirati čitav sistem – preporučljiv način simulacije nakon što ste proverili osnovnu funkcionalnost bannera; pokazaće da li vaš sistem stvarno poštuje protokol komunikacije sa driverom; dijagram će biti dosta složeniji za analizu, ali test bench je u ovom slučaju trivijalan – treba samo resetovati sistem i pustiti ga da radi.

Prilikom implementacije sistema nemojte zaboraviti da vaš vrh hijerarhije postavite kao top module, desnim klikom na njegovu datoteku u Design prozoru i izborom opcije **Set as Top Module**.

3. Pokretni tekst na LCD ekranu

Kao drugi zadatak, napravite da vaše ime i prezime rotiraju na ekranu ulevo svakih pola sekunde. Jedan način da se ovo uradi je modifikacijom niza (koji više nije konstanta nego signal) u kome su smešteni karakteri koje ispisujete. *Vodite računa da karakter 0x1B uvek mora da se ispiše prvi, kako bi se ekran obrisao pre ispisa ostalih 32 karaktera*.

Primer:

```
Name------------     =>     ame------------N     =>     me------------Na

Surname---------     =>     urname---------S     =>     rname---------Su
```

Nije teško zaključiti da će vam ponovo trebati brojač koji meri pola sekunde i određuje kada će se izvršiti rotacija.

OPŠTE NAPOMENE

Prilikom crtanja šeme na papiru ili računaru, koristiti blokove za standardne kombinacione i sekvencijalne mreže – nema potrebe da poznate komponente crtate na nivou logičkih kola. Bitno je da logička šema ispravno opiše logiku sistema. Sve nestandardne komponente koje koristite morate na neki način definisati – bilo istinitosnom tablicom ili opisom na nivou logičke funkcije.

Implementaciju sistema izvršiti za poslednji urađen korak zadatka. Nije neophodno imati urađenu implementaciju za svaki korak. Simulaciju treba raditi posle svakog koraka, jer na sledeći korak treba preći jedino ukoliko je prethodni funkcionalno proveren.

Na termin vežbe doneti **jedan** VHDL opis i **jedan** testbench (nije neophodno donositi ni opis ni testbench međukoraka). Međukoraci su tu da bi vama olakšali put ka kompletnom sistemu i omogućili da vaš sistem projektujete inkrementalno – počev od jednostavnijih komponenata ka složenijim.

Broj poena koje možete osvojiti na vežbama zavisi od koraka do kojeg ste doveli sistem – svaki korak nosi 1 poen (do maksimalno 2), a viši poeni (3, 4) se mogu dobiti izradom modifikacija i malih dopuna vašeg sistema koje ćete dobiti na terminu vežbe. Dodatni zadatak na terminu vežbe vam može doneti i dodatni, peti poen.

ZAKLJUČAK

Ukoliko vaše ime srećno rotira na LCD ekranu, čestitamo na uspešnoj realizaciji ovog veoma složenog sistema! U ovoj vežbi ste usavršili nekoliko veoma važnih koncepata u dizajnu digitalnih sistema – podela takta, komunikacija pomoću protokola, projektovanje sistema sastavljenog od više modula, „multipleksiranje“ kontrole i podataka (LCD driver je kontrolne karaktere prihvatao na istim linijama kao i podatke!) i korišćenje nizova kao veoma korisnog načina da prikažete memoriju u vašem sistemu.