Univerza v Ljubljani
Fakulteta za elektrotehniko

Klemen Perko

24-bitni Logični Analizator

Seminarska naloga

pri predmetu
Elektronska vezja

V Ljubljani, november 2002

VSEBINA

Stran

1 UVOD

Logični analizatorji so naprave, ki omogočajo analizo delovanja digitalnih vezij. Na tržišču so analizatorji, ki omogočajo več vrst analiz. Najbolj poznane so timing analiza ter state analiza. Prva nam ponuja podatke o časovnem zaporedju posameznih vhodnih linij. Podatke pri tej analizi vzorčimo asinhrono glede na opazovan sistem. Nekateri boljši analizatorji nam pri tej analizi ponujajo tudi detekcijo glitch-ev (to so neželjeni impulzi, ki so posledica logične sinteze, ob neupoštevanju realnih zakasnitev v samem vezju ter ostalih napak v samem procesu sinteze digitalnega vezja). Pri timing analizi dobimo na zaslonu naprave izrisane grafe časovnega poteka logičnih nivojev posameznih vhodnih linij. State analiza pa poteka sinhrono glede na opazovan sistem. Zato je potrebno na logični analizator priključiti tudi taktni signal opazovanega sistema. Ta analiza prikazuje, kako si v opazovanem sistemu sledijo logična stanja (tu je mišljeno logično stanje kot celotna vhodna beseda). S pomočjo te analize lahko npr. odkrivamo napake pri delovanju raznih avtomatov (Mealey in Moore) ipd.

1.1 Motivacija

Glavna napaka logičnih analizatorjev je njihova visoka cena. Večkrat se mi je že zgodilo, da bi potreboval logični analizator, vendar si ga nisem mogel nikjer sposoditi. Na fakulteti se je ponudila možnost, da dobim razvojni sistem za FPGA vezja Spartan –II. Tako sem se odločil, da bom s tem razvojnim sistemom izdelal 24-bitni logični analizator, ter s tem pokazal široke možnosti uporabe programljivih FPGA vezij. Za programiranje FPGA vezja sem uporabil višjenivojski jezik VHDL za opisovanje delovanja vezja.
Naprava, ki sem jo izdelal, vzorči in shranjuje podatke o 24 vhodnih linijah, nato pa se dobljeni podatki prenesejo na PC, kjer jih izdelan grafični vmesnik tudi prikaže. Preko grafičnega vmesnika se izvajajo tudi vse nastavitve in kontrola naprave. Komunikacija med analizatorjem in osebnim računalnikom poteka preko paralelnega vmesnika po protokolu EPP1.9. Grafični vmesnik sem izdelal s programskim orodjem Borland C++Builder 5. Naprava omogoča izvajanje samo timing analiz.

1.2 Funkcionalni opis vezja

Logični analizator je priklopljen na PC preko LPT porta in komunicira po protokolu EPP1.9. Preko grafičnega vmesnika se nastavljajo sledeči parametri delovanja analizatorja:

- način delovanja (8-bitni, 16-bitni, 24-bitni)
- hitrost vzorčenja (458Hz – 30MHz, koraki so vnaprej določeni v GUI)
- prožilna beseda – trigger (možnost nastavljanja logičnih vrednosti 0,1,X)
- pretrigger (dolžina vzorčenih dogodkov, ki se je zgodila pred triggerjem)

Grafični vmesnik je zgrajen po principu MDI (Multiple Dialog Interface). Odpremo lahko do 5 različnih meritev. Omogoča tudi časovno povečevanje in pomanjševanje slike (zoom) in časovno merjenje.

Sam analizator je sestavljen iz:
- razvojnega sistema s FPGA vezjem Spartan-II XC2S100
- zunanjega pomnilnika 192kB (v konfiguraciji 2x 64kB + 2x 32kB)
- treh vhodnih 8-bitnih sond
- paralelnega komunikacijskega vmesnika (kompatibilega z IEEE1284)

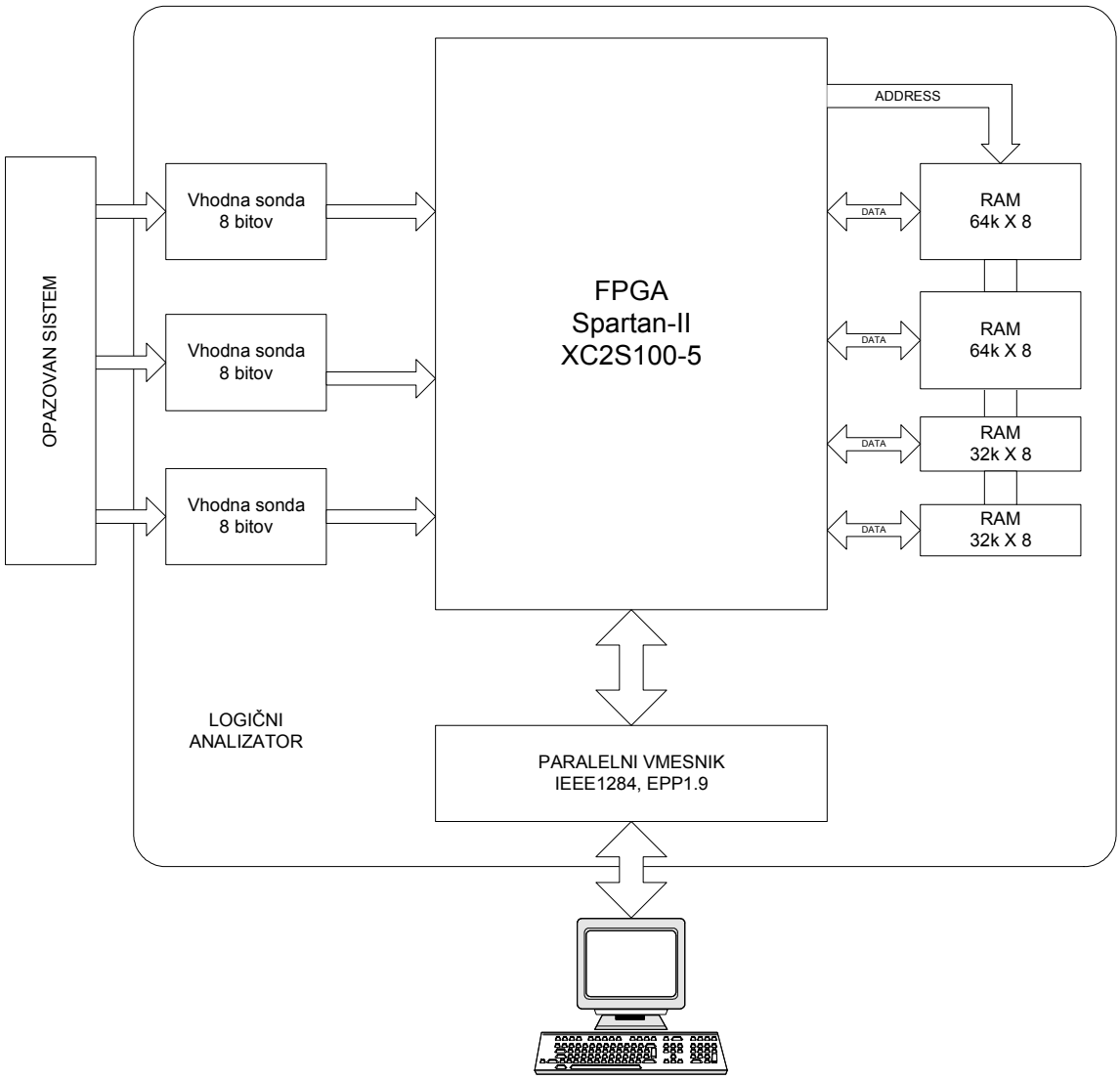
Nekaj karakteristik razvojnega sistema:
- vsebuje FPGA vezje Spartan-II XC2S100-5
- ima dvoje napajanj za FPGA:
  2.5V za interno napajanje FPGA vezja (core)
  3.3V za napajanje izhodnih blokov FPGA vezja
- FLASH pomnilnik za konfiguriranje FPGA vezja ob vklopu naprave (XC1701)
- dva sedem segmentna LCD prikazovalnika
- ...

Nekaj karakteristik FPGA vezja XC2S100-5PQ208C:

število logičnih celic	število sistemskih vrat (logičnih + RAM)	število I/O pinov	število RAM bitov
2,700	100,000	140	38,400

- max. vhodna frekvenca 200MHz
- max. interna frekvenca 333MHz
- štirje vgrajeni DLL (delay locked loop)
- štirje taktni vhodi (dedicated clock)
- možnost internega množenja vhodne frekvence (2x in 4x)
- 16 različnih standardov za I/O pine
- ...

Nekaj značilnosti zunanjega pomnilnika IS61C256AH-15 (podatki iz datasheet-a)
- High-speed access time: 15ns
- Low active power: 400 mW (typical)
- Low standby power
  55 mW (typical) TTL standby
- Fully static operation: no clock or refresh required
- TTL compatible inputs and outputs
- Single 5V power supply



Osnovni princip delovanja naprave je sledeč:

- PC nastavi začetne nastavitve (način delovanja hitrost vzorčenja, prožilna beseda, pretrigger)
- naprava odvisno od načina delovanja (8, 16, 24 bitno) vzorči vhodne podatke in jih sproti shranjuje v RAM
- od trenutka ko nastopi prožilna beseda, začne analizator šteti vzorce. Glede na način delovanja in nastavitev pretriggerja, je odvisno koliko vzorcev bo naprava še shranila od trenutka nastopa prožilne besede
- po končanem vzorčenju PC prebere shranjene vzorce in jih prikaže na zaslonu

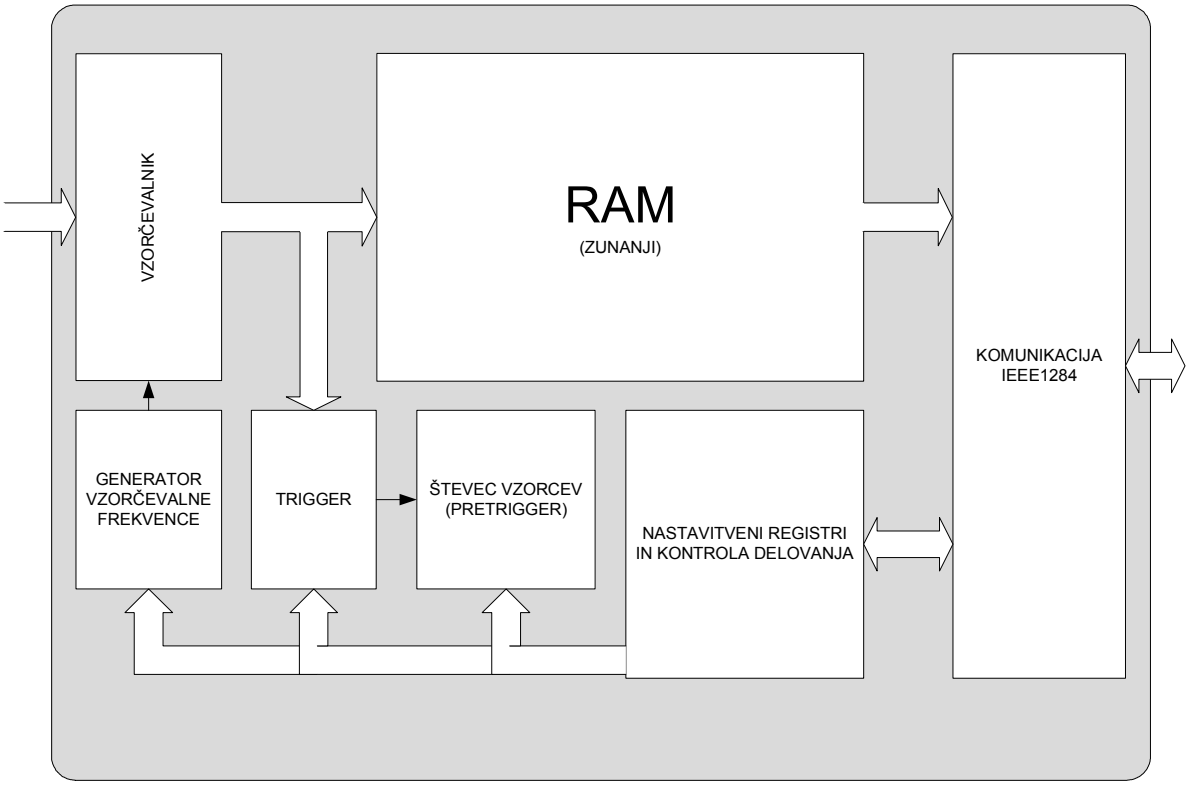
2 GLAVNI DEL

2.1 Opis delovanja podsklopov vezja s shematskim prikazom podsklopov

Kot je že iz prejšnje slike razvidno, je analizator sestavljen iz treh večjih podsklopov.

2.1.1 Razvojni sistem s FPGA vezjem.

V FPGA vezju je zgrajena celotna logika, ki je potrebna za delovanje. Implementiran je generator vzorčevalne frekvence, vzorčevalnik, prožilnik (trigger), vodila za zunanji pomnilnik, logika za razdeljevanje pomnilnika glede na način delovanja (8, 16 oz. 24 bitno) in komunikacijski vmesnik z dodatno logiko za upravljanje notranjih nastavitvenih registrov in vodil za dostop do zunanjega pomnilnika.



2.1.1.1 Nastavljanje hitrosti vzorčenja:

Analizatorju je možno nastaviti hitrost vzorčenja od 458Hz do 30MHz po 458Hz korakih. Ta del je izveden s pomočjo NCO (numerical controlled oscillator). Frekvenca vzorčenja je nastavljiva po naslednji enačbi:

$$frek = \frac{clk}{2} \times \frac{beseda}{65536}$$

Pri čemer pomeni:

- clk- takt
- beseda – 16 bitna beseda, s katero se nastavlja hitrost (nastavlja jo PC)

Ker sem želel dobiti frekvence vzorčenja do 30MHz, sem moral zato uporabiti en Spratanov DLL (delay locked loop) za interni dvig frekvence na 60MHz. Zato del logike deluje s taktom 60MHz, del pa s 30MHz. Ena od negativnih lastnosti NCO-ja je jitter. Negotovost periode željene frekvence je enaka eni periodi taktne frekvence NCO-ja. V konkretnem primeru je to 16.7nS, kar pa je za polovico manj od periode vzorčenja pri max. frekvenci.

2.1.1.2 Nastavljanje prožilne besede (trigger-ja):

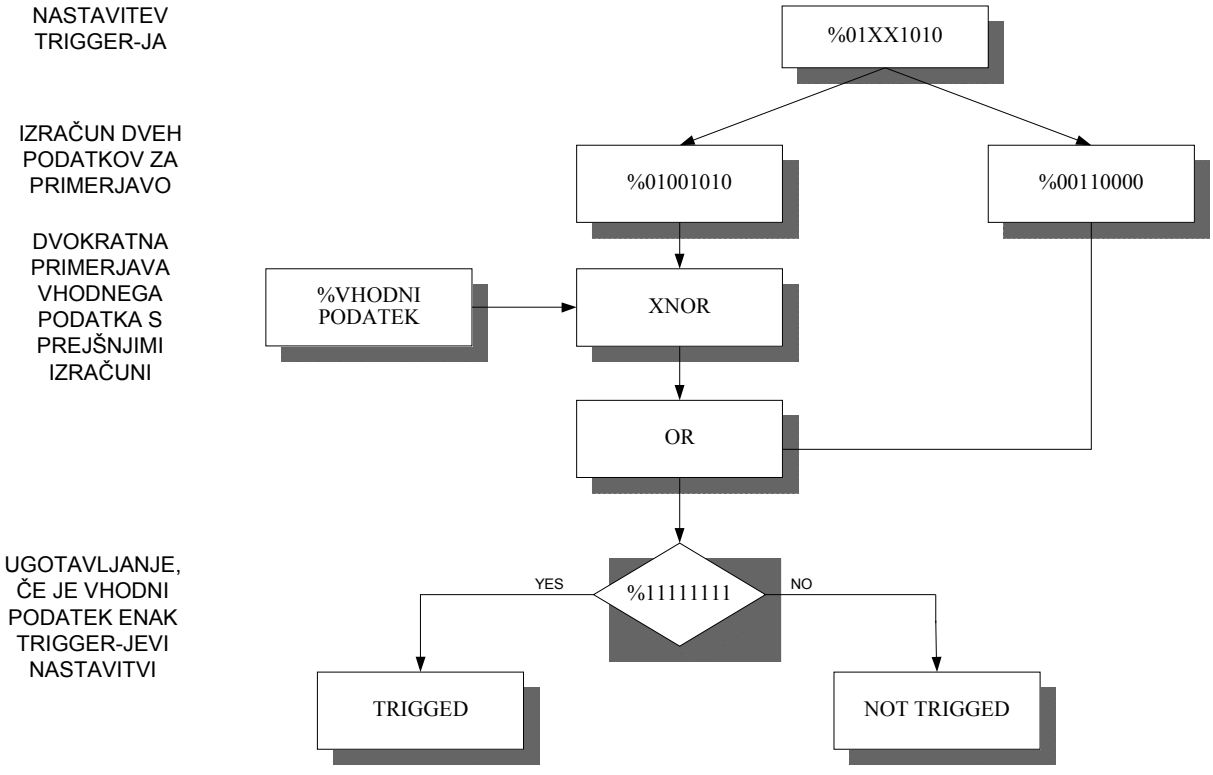
Dolžina prožilne besede je odvisna od načina delovanja analizatorja (8, 16, 24 bitov). Prožilna beseda lahko vsebuje stanja 0, 1, X. V grafičnem vmesniku se vpiše prožilna beseda sestavljena iz omenjenjih znakov. Program izračuna dva podatka iz podane besede:

- kjer se nahajajo '1' dobi eno binarno število (npr. 01XX1010 -> 01001010 - trig_A1)
- kjer se nahajajo 'X' dobi drugo binarno število (npr. 01XX1010 -> 00110000 - trig_A2)

Ti dve števili potem pošlje analizatorju. Tu se potem ob vsakem novem vzorčenju primerja vzorčen podatek s prej nastavljenimi števili po principu :

```
trig_A3<= (DATA_VHOD_A xnor trig_A1);
if ((trig_A3 or trig_A2) = "11111111" ) then
        trigged_a<='1';
else
```

Diagram poteka tega postopka:



2.1.1.3 Nastavljanje pretriggerja:

Preko grafičnega vmesnik1 se nastavi velikost pretriggerja. Ta je odvisna od načina delovanja. Max. velikosti pretriggerja so:

- 24 bitni način - 65535 vzorcev
- 16 bitni način - 98303 vzorci
- 8 bitni način - 196607 vzorcev

V primeru maksimalnih nastavitev, to pomeni, da se vzorčenje ob nastopu prožilne besede ustavi.

2.1.2 Zunanji statični pomnilnik

Drugi večji podsklop predstavlja zunanji statični pomnilnik. Ker ima konkretno FPGA vezje samo 40k bitov RAM-a (t.j. okrog 5kB), sem dodal zunanji ram (6x32kB=192kB). Ker sem želel narediti analizator, ki bi bil sposoben vzorčiti vhodne podatke s frekvenco oscilatorja na razvojni plošči, sem moral najti temu premerno hitre RAM-e (časi dostopa do podatkov (read in write) v rangu 30nS oz. hitrejše za frekvenco vzorčenja 30MHz). V trgovinah z elektronskimi komponentami imajo na zalogi samo statične RAM-e, ki so v rangu od 70nS in več, zato sem se odločil, da uporabim cache RAM-e s starih matičnih plošč (PC tipa 486). Uporabil sem 6 integriranih vezij (61C256) po 32kB, njihov max. dostopni čas pa je 15nS. Pomnilnik, ki sem ga dodal, sem razporedil v štiri enote. Dve sta sestavljeni iz 64kB, dve pa iz 32kB. S tem sem pridobil naslednje možnosti uporabe RAM-a v različnih načinih delovanja:

24-bitno delovanje:

64kB	– kanal A
64kB	– kanal B
32kb+32kb	– kanal C

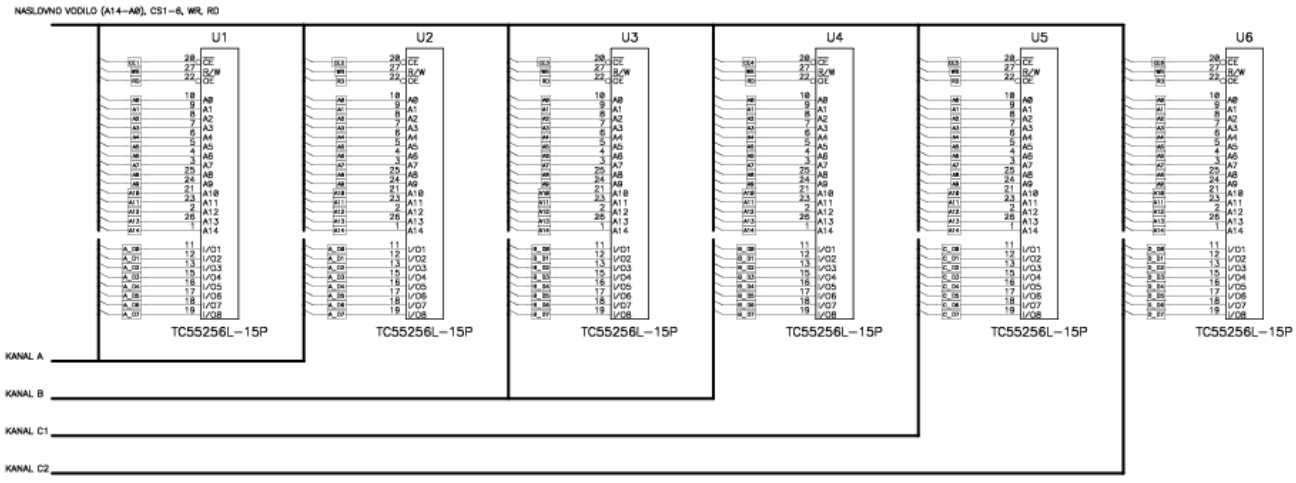
16-bitno delovanje

64kb + 32kb	– kanal A
64kb + 32kb	– kanal B

8-bitno delovanje

64kb + 64kb + 32kb + 32kb	– kanal A

(vsak kanal sestavlja 8 bitov!)



Tako je v vsakem od treh načinov vedno zagotovljena polna izkoriščenost pomnilnika. Od načina delovanja je odvisno, na katerih naslovih se nahajajo podatki o posameznem kanalu. To je potrebno upoštevati pri izrisu podatkov na PC-ju.

Uporabljeni RAM-i imajo eno veliko pomanjkjivost: kadar so aktivni (CE=’0’) imajo več kot 100mA tokovne porabe. V primeru da je naprava v 24 bitnem načinu, so

istočasno aktivirani trije RAM-i. To predstavlja porabo približno 300-350mA. Pri nižjih frekvencah vzorčenja (pod 10MHz), sem dodal poseben mehanizem, ki aktivira RAM-e en urni cikel pred vzorčenjem, ter jih izkjuči en urni cikel po vzorčenju. **S tem sem močno znižal povprečno porabo med vzorčenjem.** Nad frekvenco 10MHz je ta mehanizem izkjučen. Kadar analizator vzorči podatke in so RAM-i aktivrani, se z osciloskopom lahko vidi, kako močno se poveča šum na napajalnih žicah. Zaradi tega sem dlje časa testiral, če sestavljeno vezje deluje pravilno. To sem testiral tako, da sem s PC-jem vpisoval točno določene podatke na točno določene lokacije, ter jih potem bral nazaj in preverjal. Hotel sem preveriti, če omenjeni šum na napajanju RAM-ov predstavlja kakšne težave za vpisovanje podatkov. Uspešno sem prenesel približno 5GB.

2.1.3 Komunikacijski del

Analizator in PC komunicirata preko paralelnega vmesnika po protokolu EPP 1.9. Ta protokol omogoča ločen prenos naslova in podatkov. PC s pomočjo vmesnika tudi kontrolira, če je analizator usprešno prejel podatek oz. naslov. Po protokolu EPP1.9 mora periferija v manj kot 10µS odgovoriti PC-ju (po protokolu to stori s signalom WAIT), da je podatek/naslov sprejela/oddala, drugače vmesnik signalizira Time Out bit v statusnem registru paralelnega vmesnika (EPP 1.7 te opcije nima). Program, ki teče na PC-ju lahko s tem ugotovi, če je komunikacija z napravo dobra. Maksimalna hitrost prenosa, ki sem jo izmeril, je 250kB/S, (operacijski sistem – DOS, program na PC-ju ne preverja Time Out bita), oziroma 170kB/S (operacijski sistem – DOS, program na PC-ju preverja Time Out bit).

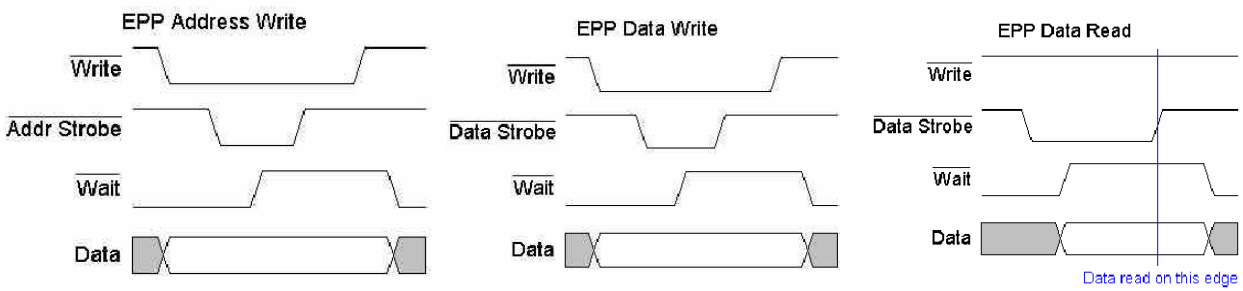
Spodnja tabela podaja lokacije posameznih signalov v načinu SPP in EPP.

Pin	SPP Signal	EPP Signal	In/Out	Function
1	Strobe	Write	Out	A low on this line indicates a Write, High indicates a Read
2-9	Data 0-7	Data 0-7	In-Out	Data Bus. Bi-directional
10	Ack	Interrupt	In	Interrupt Line. Interrupt occurs on Positive (Rising) Edge.
11	Busy	Wait	In	Used for handshaking. A EPP cycle can be started when low, and finished when high.
12	Paper Out / End	Spare	In	Spare - Not Used in EPP Handshake
13	Select	Spare	In	Spare - Not Used in EPP Handshake
14	Auto Linefeed	Data Strobe	Out	When Low, indicates Data transfer
15	Error / Fault	Spare	In	Spare - Not used in EPP Handshake
16	Initialize	Reset	Out	Reset - Active Low
17	Select Printer	Address Strobe	Out	When low, indicates Address transfer
18-25	Ground	Ground	GND	Ground

Pri komuniciranju PC-ja z analizatorjem uporabljam tri različne prenose podatkov:

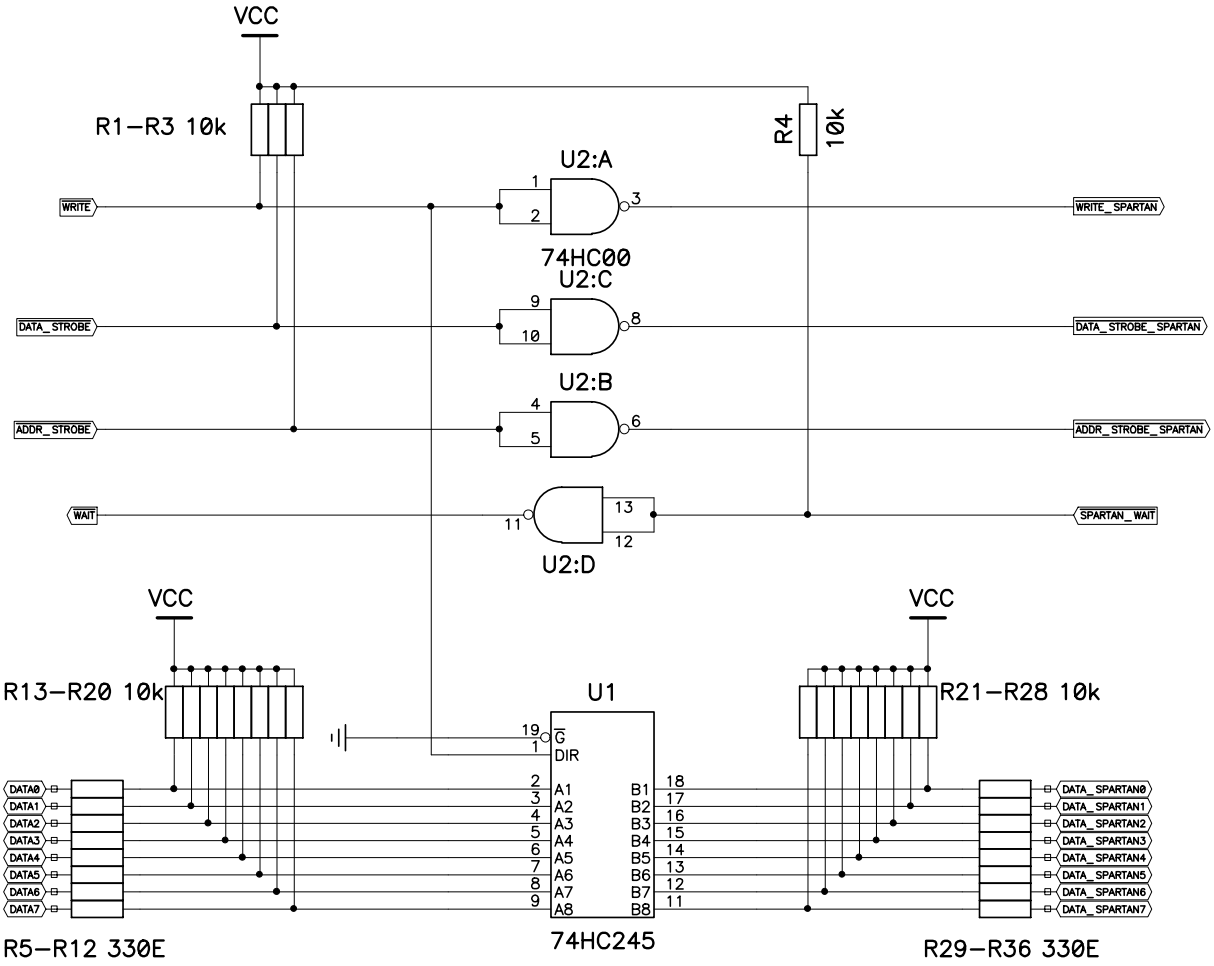
- vpis naslova v komunikacjski del analizatorja
- vpis podatka v analizator (nastavitveni registri)
- branje podatka iz analizatorja (branje pomnilnika)

Časovni poteki posameznih načinov prenosa podatkov so na spodnjih treh slikah:



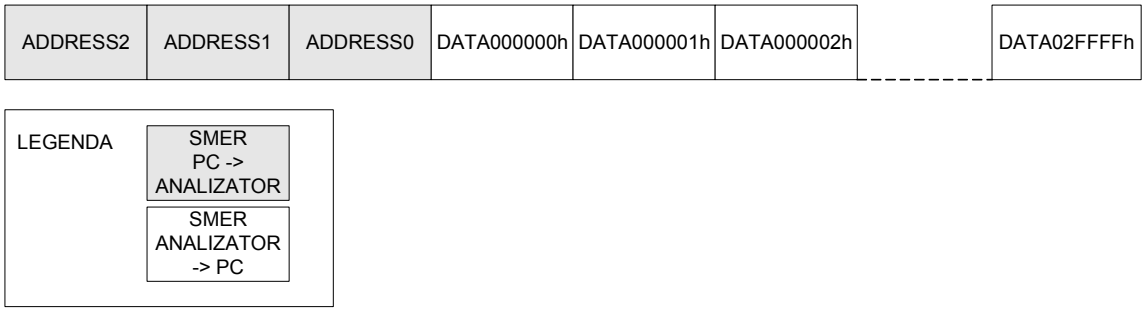
Podrobnejši opis delovanja vmesnika IEEE1284 po protokolu EPP se nahaja v [xyz].

HW del med PC-jem in Spartanom služi za prilagoditev nivojev. Sestavljen je iz line driverja 74HC245, 4-ih negatorjev, ter uporov. Shemo prikazuje spodnja slika:



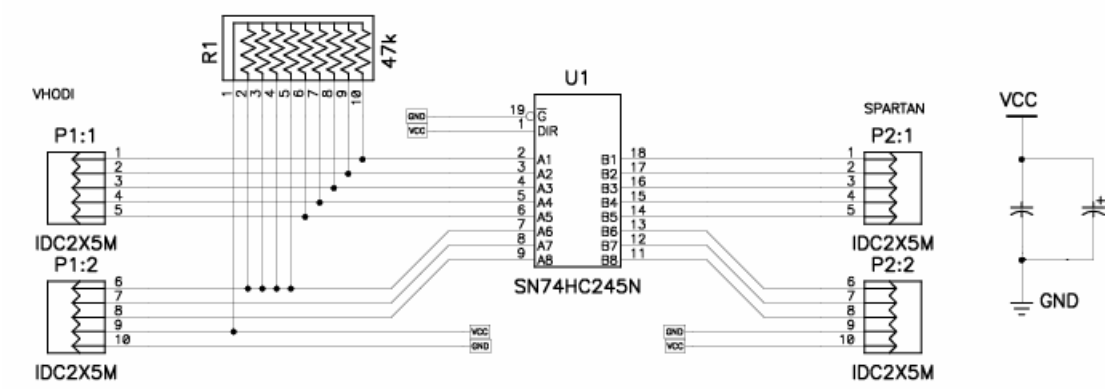
Zato, da se doseže hiter in učinkovit prenos podatkov med Spartanom in PC-jem, je bilo potrebno izdelati protokol, po katerem se prenašajo podatki. Protokol je sestavljen tako, da najprej PC pošlje Spartanu v treh zaporednih delih podatek o

naslovu (3x8bitov=24bitni naslov), s katerim bi PC želel operirati (branje/pisanje). S tem se nastavi interno naslovno vodilo v Spartanu. Nato pa PC pošlje/prebere podatek na/iz to lokacijo. Ob tem se vrednost internega vodila poveča za ena. Zato PC-ju ni potrebno za vsak podatek posebej povedati naslova, ampak lahko prebere celoten blok. To je smiselno, saj je potrebno ob vsaki meritvi prenesti 192kB. Spodnja slika prikazuje učinkovitost tega pristopa. Za branje pomnilnika je treba naslov za branje nastaviti le na začetku. Ob vsakem prebranem podatku pa se interno naslovno vodilo avtomatsko poveča za vrednost ena. Isti pristop velja za pisanje podatkov (nastavitev registrov).



2.1.4 Vhodne sonde

Vhodne sonde so sestavljene iz zelo šibkih pull-up uporov na vhodu ter line driverja 74HC245. Vezja vhodnih sond so vgrajena vsaka v svoje ohišje in se priključijo na logični analizator preko flat kabla.



2.2 Analiza delovanja

Preko grafičnega vmesnika se nastavljajo sledeči parametri delovanja analizatorja:

- način delovanja (8-bitni, 16-bitni, 24-bitni)
- hitrost vzorčenja (458Hz – 30MHz)
- prožilna beseda – trigger
- pretrigger

Ob sprožitvi začetka meritve, PC nastavi pripadajoče registre. Logični analizator začne takoj z vzorčenjem vhodnih linij in z zapisovanjem vzorcev v zunanji pomnilnik. Istočasno tudi preverja, če je nastopi na na vhodu prožilna beseda. Dokler ne nastopi prožilna besede, se vsi podatki krožno vpisujejo v pomnilnik. Ob nastopu

prožilne besede pa se začne štetje vzorcev. Glede na velikost nastavljenega pretriggerja je odvisno, koliko vzorcev bo logični analizator še sprejel. Npr., če je velikost pretriggerja nastavljena na 0 vzorcev, bo analizator vzorčil toliko časa, dokler ne bo celoten pomnilnik zopet osvežen, če pa je velikost pretriggerja nastavljena na maksimalno vrednost (odvisno od načina delovanja – 8, 16, 24), pa bo analizator ob nastopu prožilne besede ustavil vzorčenje in zaključil tekočo meritev. Ob zaključku meritve se v poseben register vpiše tudi naslov zadnjega vzorčenega podatka. S tem podatkom kasneje grafični vmesnik določi, kje v pomnilniku se nahaja prvi vzorec. Po zaključeni meritvi se vsi podatki prenesejo na PC. Grafični vmesnik izriše časovne poteke posameznih vhodnih linij. Posebej označi tudi vzorec, pri katerem je nastopila prožilna beseda.

2.3 Navodila za uporabo

Grafični vmesnik je zgrajen po principu MDI (Multiple Dialog Interface). Odpremo lahko do 5 različnih meritev. Vsako okno ima pripadajoče nastavitve. To nam omogoča lažje izvajanje različnih meritev, ki pa se lahko ponavljajo, ne da bi morali vedno znova vnašati njihove nastavitve. Pri preklapljanju med merilnimi okni, se spreminjajo tudi nastavitve meritev. Vmesnik omogoča tudi časovno povečevanje in pomanjševanje slike (zoom) in časovno merjenje. Osnovni parametri in njihov pomen, ki jih je potrebno pred meritvijo nastaviti so bili opisani že v prejšnjih razdelkih. Omeniti velja le še nekaj stvari:

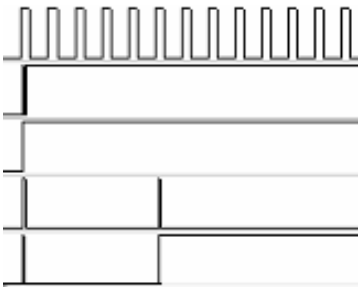
- pri nastavitvi vzorčevalne frekvence nam grafični vmesnik prikaže pripadajočo periodo
- pri nastavitvi pretriggerja pa nam glede na vzorčevalno frekvenco izpiše tudi časovni zamik
- vzorec, pri katerem je nastopila prožilna beseda, je označena z navpično svetlomodro črto
- nastavitev paralelnega vmesnika na osebnem računalniku se nastavi v meniju “Settings” (tu je možno tudi izključiti preverjanje komunikacije, s tem pa povečati njeno hitrost)

3 ZAKLJUČEK

3.1 Morebitne težave

Pri izdelavi te naprave sem naletel na veliko težav. Ena prvih je bila vzpostavitev hitre in zanesljive komunikacije med osebnim računalnikom in FPGA vezjem. Na začetku, sem mi je na vsakih nekaj tisoč pravilno prenešenih bytov zgodilo napaka pri prenosu. Ugotovil sem, da se pojavljajo glitch-i, zato sem izdelal posebne mehanizme za filtriranje (podobno kot debounce). Po uspešni izdelavi komunikacije in njenem testiranju, so se pojavile težave, kako zmanjšati porabo zunanjega pomnilnika. Delovanje tega mehanizma sem opisal že prej. Največ preglavic pa mi je naredila izdelava grafičnega vmesnika. Dokler nisem izdelal grafičnega vmesnika, nisem uspel dobro testirati samega delovanja analizatorja, saj sem imel na voljo le kup podatkov, ki pa so mi brez vizualne predstave bolj malo pomenili. Najprej sem za testiranje izdelal razne testne strukture znotraj FPGA vezja. S tem sem izločil možne motnje, ki

bi se lahko pojavile na vhodih vezja. Večinoma sem za te teste uporabljal razne števce. Šele po uspešnem zaključku tega koraka, sem priklopil zunanje sonde. Ravno tu pa sem odkril določene težave. Zunanje sonde sem priklopil preko približno 15cm dolgega flat kabla. Pri nizkih frekvencah (reda nekaj 100kHz) opazovanih signalov ni nikakršnjih težav. Pri višjih frekvencah pa nastopijo presluhi med sosednjimi linijami. Zato se ob spremembi logičnega stanja na eni liniji, lahko pojavi na sosednji liniji kratek impulz. Ta težava je lepo vidna na spodnji sliki.



Pri izdelavi grafičnega vmesnika sem naletel še na nekaj težav. Kot prvo sem ugotovil, da Borlandov C++ Builder V5 nima več vgrajenih funkcij inp() in outp(). S tem so želeli namreč preprečiti dostop do posameznih internih registrov v računalniki. Tega dostopa tudi ne dopuščajo operacijski sistemi zgrajeni na osnovi winNT. Pri profesionalni izdelavi vmesikov, je zato potrebno uporabljati razna DDK (device driver kit) orodja. Ker sam teh orodij ne znam uporabljati, sem moral najprej izdelati assemblerske nadomestke funkcij inp() in outp(), zato da sem lahko dostopal do registrov paralelnega vmesnika. Pri operacijskih sistemih w2k, wXP in winNT pa je potrebno zagnati kakšnega od gonilnikov, ki nam omogoči dostop do teh registrov (npr. PortTalk).

3.2 Sklepne ugotovitve

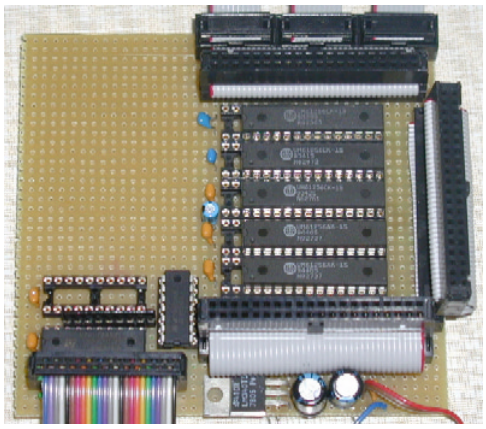
Glede na obsežnost zadane naloge, je končni izdelek relativno dobro uspel. Če bi želel analizator izpopolniti, bi moral izdelati novo TIV s FPGA vezjem in zunanjim pomnilnikom. Izdelava TIV nebi bila prezahtevna, saj bi bilo potrebno izdelati štiri slojno vezje, kar pa močno poenostavi izdelavo kvalitetnega napajanja za integrirana vezja (ground in power plane).

3.3 Možnosti nadgradnje

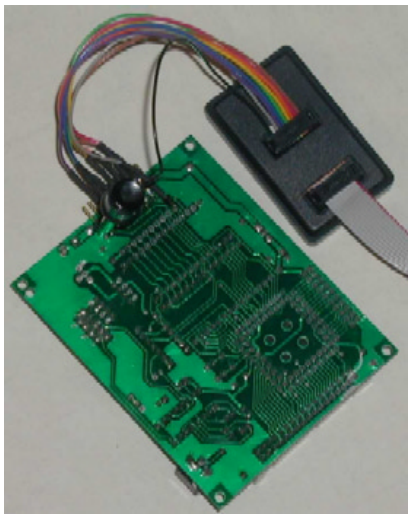
Pri izdelavi logičnega analizatorja se poraja veliko zanimivih idej.
Pri nadgradnji moje rešitve, bi bilo najprej dobro razmisliti, kako povečati velikost zunanjega pomnilnika. Obstaja več možnosti. Prva je, da se zamenja sedanji pomnilnik z nekoliko večjim, precej dražjim, a ravno tako statičnim pomnilnikom, druga pa je, da se ga zamenja s precej večjim, nekoliko dražjim dinamičnim pomnilnikom. Obe možnosti imata svoje prednosti in slabosti. Prva ima veliko prednost, da ni potrebna zahtevna logika za delo s pomnilnikom. ima pa ta varianta veliko pomankljivost. Razmerje med velikostjo pomnilnika in njegovo ceno je zelo neugodna. Druga varianta z uporabo dinamičnega pomnilnika ima to razmerje zelo ugodno. Za relativno malo denarja, se lahko pomnilnik razširi na nekaj MB. Vendar pa je pri uporabi dinamičnega pomnilnika potrebno izdelati poseben krmilnik za njegovo osveževanje. Potrebno je izdelati tudi hitri notranji cache pomnilnik, v

katerega analizator hitro zapisuje vzorce, potem pa jih bločno prenese v zunanji pomnilnik. Pred odločitvijo, katera varianta je ugodnejša, bi bilo dobro narediti nekaj izračunov in veliko premisleka.
Izboljšati bi bilo potrebno grafični vmesnik. Najbolje bi bilo, če bi to naredil programer, ki se s podobnimi nalogami srečuje vsakodnevno.
Ena od možnih izboljšav bi bila tudi zamenjava komunikacije. Ena od mikavnih izbir je USB protokol, še boljša pa je izbira Ethernet protokola. Ethernet sicer zahteva precej več dela, vendar pa je tudi precej boljši s stališča uporabe, saj je možno potem neko napravo uporabljati kar preko mreže ali interneta. V tem primeru bi bilo najbolje poleg FPGA vezja dodati enega izmed sodobnih mikrokontrolerjev, ki podpirajo Ethernet protokol.
Ostaja pa še ena zelo pomembna šibka točka. Izdelati bi bilo potrebno vhodne sonde, ki bi imele nizko vhodno kapacitivnost (za višje vhodne frekvence) in ne bi bile občutljive na motnje, presluhe,... Ocenjujem, da bi že uporaba drugačnega flat kabla presluhe močno zmanjšala. Uporabiti bi bilo mogoče flat kabel z dvojno gostoto žil, pri tem pa bi bilo potrebno vsako drugo žilo spojiti na maso.

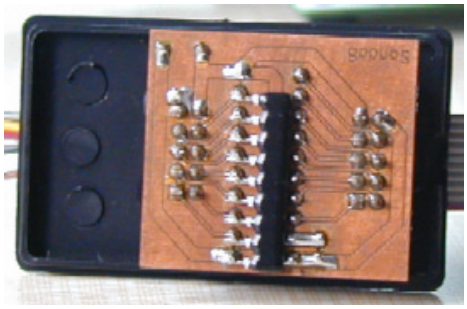
4 SLIKE



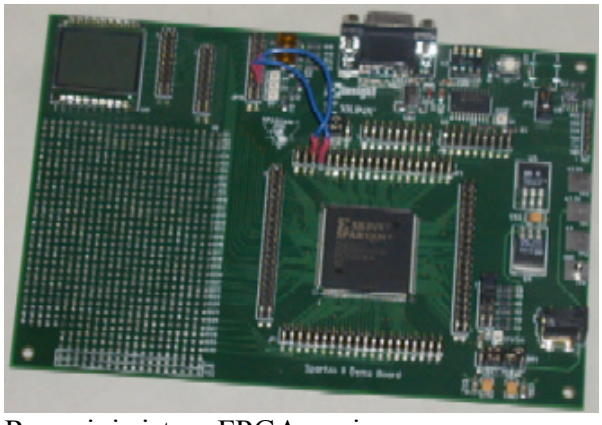
Zunanji pomnilnik



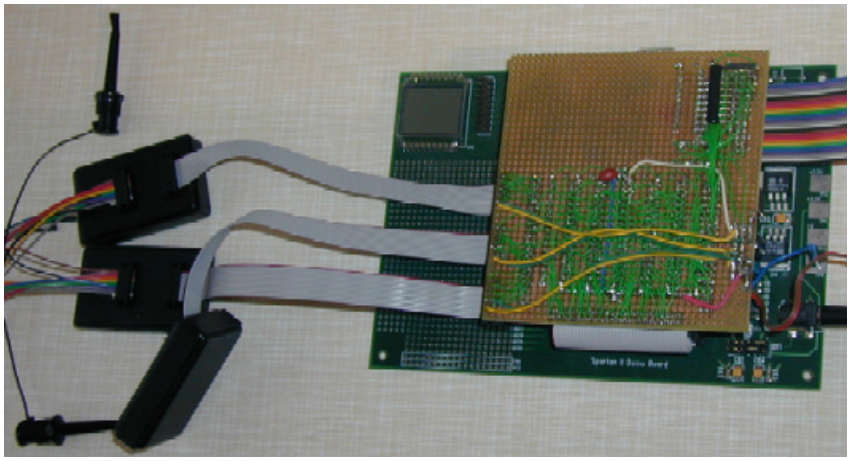
Priklop sonde na merjenec



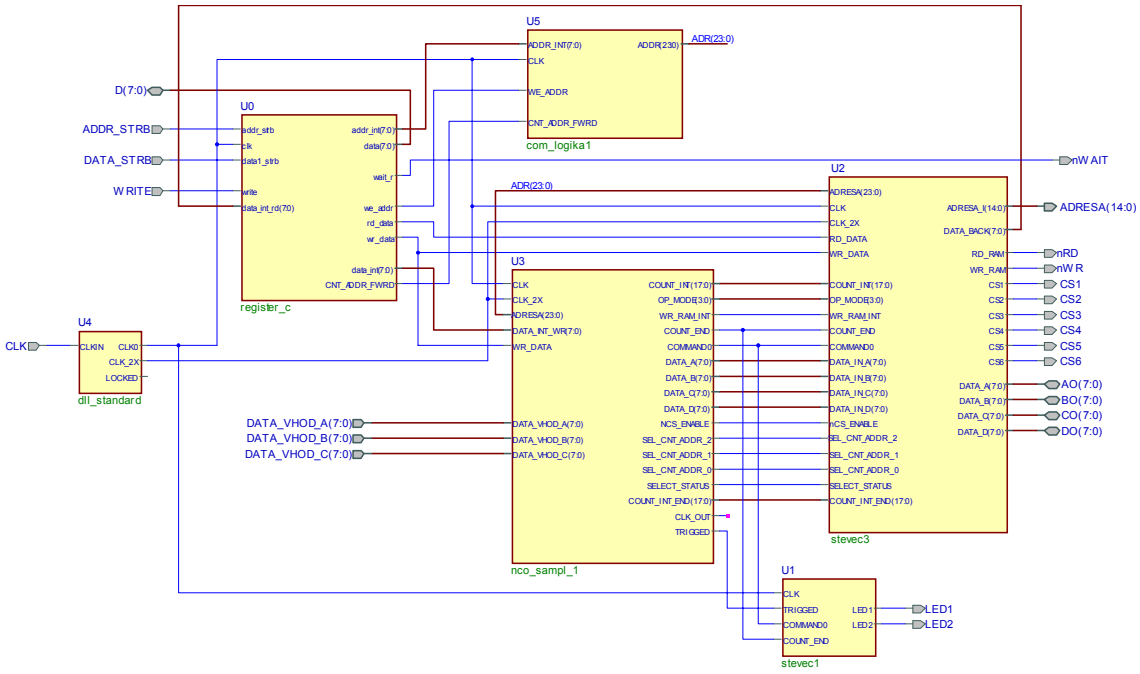
TIV sonde



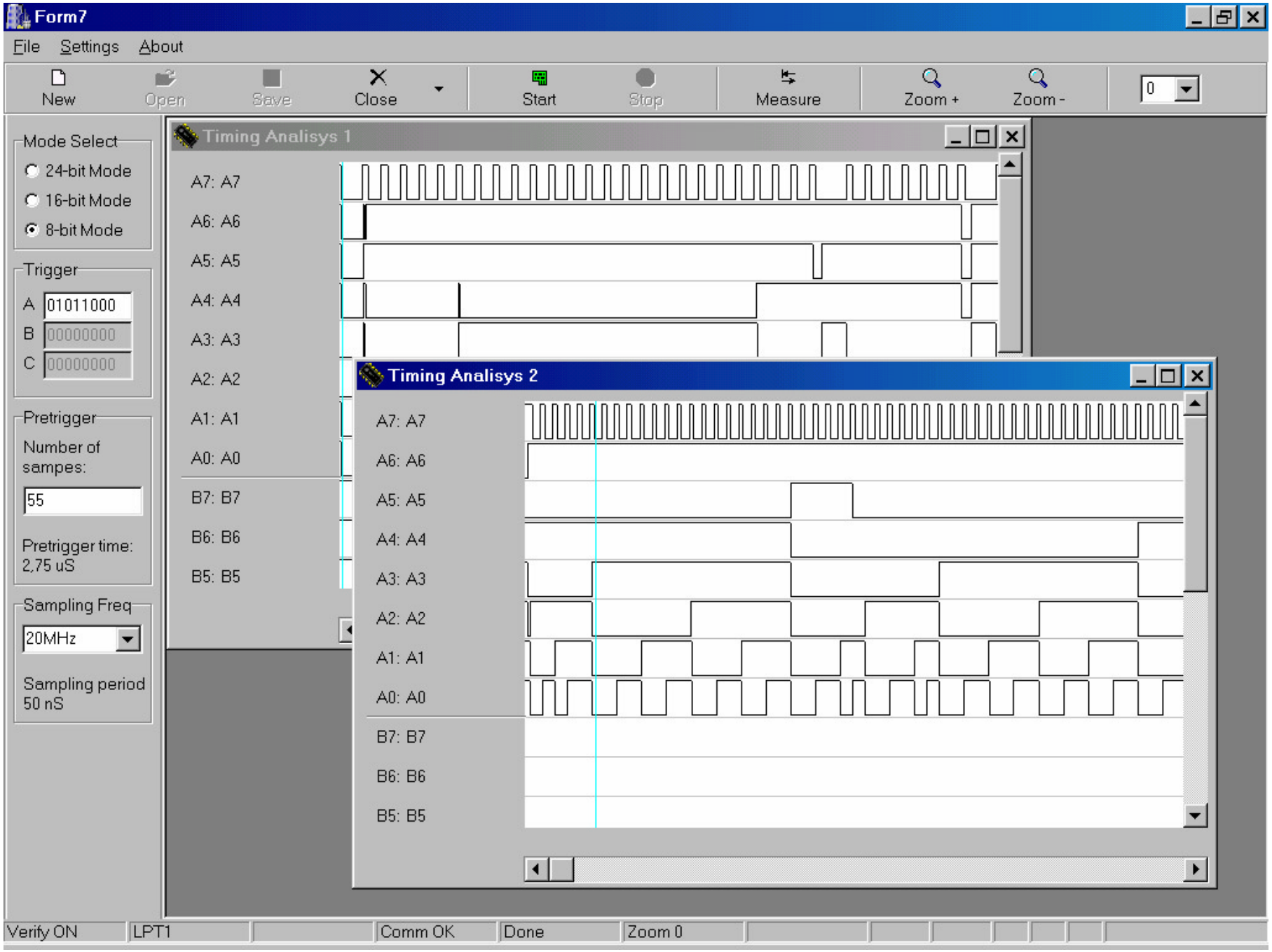
Razvojni sistem FPGA vezja



Celoten analizator



Povezava med posameznimi bloki v VHDL-u



Grafični vmesnik

5 REFERENCE:

Datasheet-i uporabljenih integriranih vezij:

- **74HC00** – nand logična vrata
  http://www.semiconductors.philips.com/acrobat/datasheets/74HC_HCT00_CNV_2.pdf
- **74HC245** – line driver
  http://www.semiconductors.philips.com/acrobat/datasheets/74HC_HCT245_CNV_2.pdf
- **IS61C256AH-15** – statični RAM pomnilnik
  http://www.issi.com/pdf/61C256AH.pdf
- **XC2S100-5PQ208C**
  http://direct.xilinx.com/partinfo/ds001_1.pdf
  http://direct.xilinx.com/partinfo/ds001_2.pdf
  http://direct.xilinx.com/partinfo/ds001_3.pdf
  http://direct.xilinx.com/partinfo/ds001_4.pdf

Opis delovanja paralelnega vmesnika:

- http://www.beyondlogic.org/epp/epp.htm
- http://www.eureca.de/pdf/english/EPP.pdf

Opis razvojnega sistema s FPGA XC2S100

http://www.insight-electronics.com
Xilinx: Spartan™-II Development Kit

http://www.insight-electronics.com/cgi-bin/bvutf8/memec/scripts/local/mc_loc_b.jsp?Div=INSIGHT&Reg=AMERICAS&Country=UNITED_STATES&Lang=EN&EDOID=187084&Manu=XILINX