Универзитет у Крагујевцу
Факултет инжењерских наука

Семинарски рад из предмета:
Основи рачунарске технике 2

Тема:
Симулација таксиметра на „Spartan 3E FPGA” плочи

Студенти:
Никола Самарџић 562/2015
Анђела Благојевић 558/2015

Предметни професор:
Александар Пеулић

Крагујевац 2017.

Садржај:

Увод

У овом пројекту разматраћемо архитектуру „Nexys 2 Spartan 3E FPGA” плоче, као и пример практичне примене исте помоћу Xilinx софтверског алата.
Тема семинарског рада је таксиметар, који је реализован помоћу верилога. Износ који муштерија треба да плати приказан је на седмо-сегментном дисплеју, паљењем прекидача таксиметар инкрементира вредност, уз кашњење од 2.5 секунде, почевши од 60 (што је старт у таксију), гашењем прекидача вредност престаје да се инкрементира и то је вредност коју муштерија треба да плати. Помоћу дугмета „reset” седмо-сегментни дисплеј приказује вредност од 60, чиме је такси спреман да прими нову муштерију и одвезе је на жељену дестинацију. Недостатак пројекта је то што таксиметар мери време вожње, не пређени пут током исте.

„Spartan 3E FPGA[1]” породица је специфично дизајнирана како би задовољила потребе великог обима потрошача, својом ниском ценом и применом. „Spartan 3E“ наставља успех раније „Spartan 3“ фамилије повећавајући обим логичких кола по I/O, приметно смањујући цену по логичкој ћелији. Нове особине побољшавају преформанце исте, као и трошак конфигурације. Нова „Spartan-3E FPGA“ побољшања дају већу функционалност породици и повећава њену доступност ниским ценама, чиме поставља нови стандард у својој индустрији.

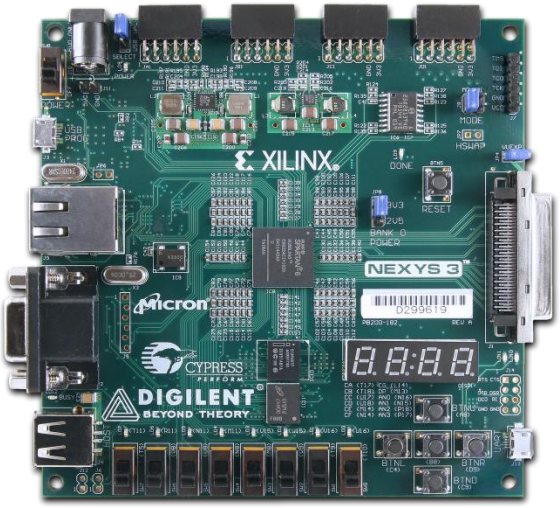
[1] FPGA (Field-Programmable Gate Array) представља интегрисано коло чија се унутрашња структура може конфигурисати од стране крајњег корисника.

Архитектура

Особине

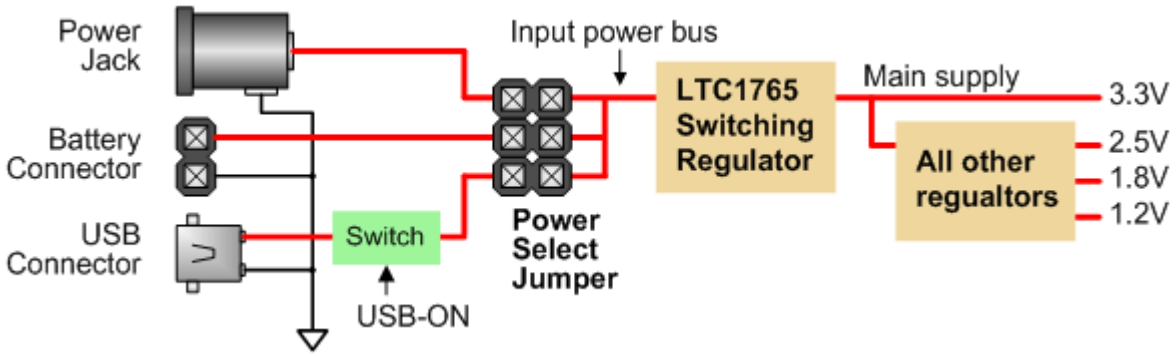
- 500K-gate Xilinx Spartan 3E FPGA
- USB2-базирано на FPGA конфигурацији и брзом преносу података(користећи бесплатан Adept Suite софтвер)
- USB-напајање
- 16MB Micron PSDRAM i 16MB Intel StrataFlash ROM
- Xilinx Platform Flash за трајну FPGA конфигурацију
- Ефикасан прекидачки извор напајања
- 50MHz осцилатор, плус место за други осцилатор
- 60 FPGA I/O повезаних на проширене конекторе (брзи Hirose FX2 конектор са 43 сигнала i 4 2x6 Pmod конектора)
- 8 LED диода, Четвороцифарни седмо-сегментни дисплеј, 4 тастера, 8 прекидача

Nexys2 доноси водеће технологије платформом коју свако може користити да би стекао искуство у дигиталном дизајну. Nexys2 плоча је компатибилна са свим верзијама Xilinx ISE алата, укључујући бесплатан WebPack. Сада свако може изградити праве дигиталне системе по цени мањој од цене уџбеника.



Извор напајања

„Nexys2” плоча се може напајати помоћу USB кабла, 2.1 милиметарским зидним прикључком или батеријом. „Power select” блоком, који се налази на плочи, бирамо извор напајања. USB кола се увек напајају USB каблом, уколико није прикључен USB кабал, USB коло остаће без напајања.



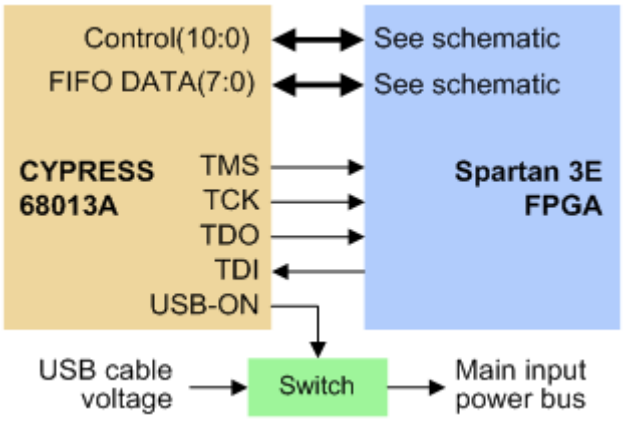
Магистрала за извор напајња користи регулатор од 3.3 волта, који снабдева целу плочу. Неким уређајима је потребан додати извор од 2.5V, 1.8 V или 1.2 V, поред главног од 3.3V, и ти додатни извори су направљени помоћу регулатора који користе главни извор од 3.3V. Примарни извори се генеришу помоћу ефикасних прекидачких регулатора, који користе Линеарну технологију. Ови регулатори не само да користе ефикасно снагу USB-а, већ дозвољавају да Nexys2, уз помоћ батерије, ради дуже.

Укупна струја плоче зависи од FPGA конфигурације , фреквенције осцилатора, и спољних веза. У струјним колима са отприлике 20 000 повезаних улазних кола, са осцилатором фреквенције од 50MHz, и свим диодама упаљеним, „тече“ струја од 200mA уколико је извор 1.2 волта, 50mA за извор од 2.5 волта и 100mA за извор од 3.3 волта. Струја ће се повећати ако су већа кола конфигурисана у FPGA, и ако су спољне плоче прикључене.

Клок

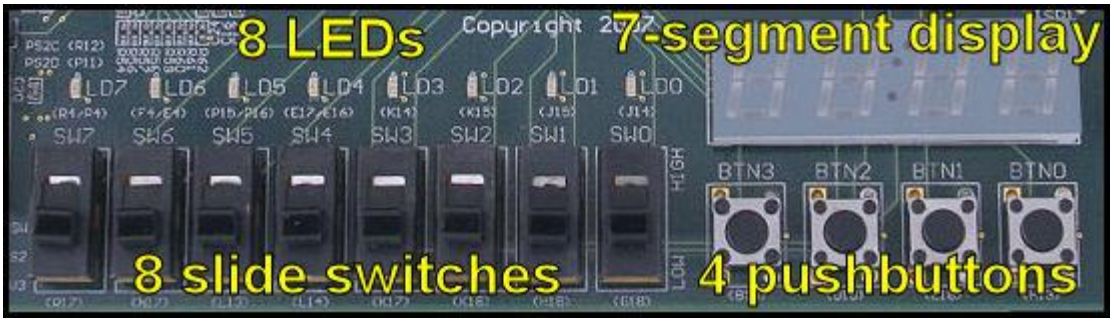
Nexys2 плоча укључује осцилатор од 50MHz и место за други осцилатор. Клок сигнали из осцилатора повезани су са глобалним улазним пиновима клока на FPGA како би могли да користе слободне клок блокове на FPGA. Клок **synthesizers** (звани DLL, или кашњење

закључане петље) пружа могућност управљања клоком који укључују двоструки или четвороструки унос фреквенције, дељењем улазне фреквенције било којим бројем(integer), и дефинише прецизне фазе и односе кашњења између различитих клок сигнала.



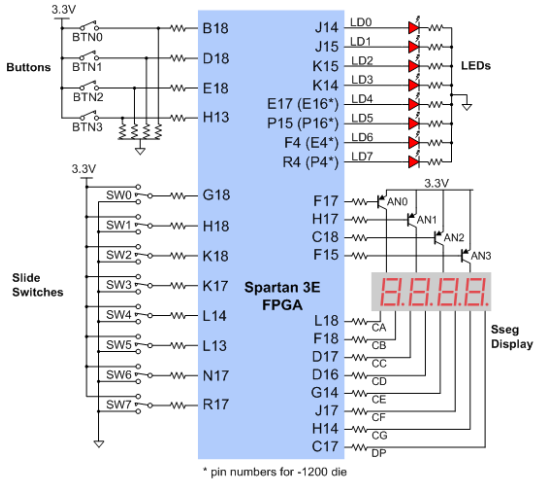
Кориснички I/O

Nexys2 плоча садржи више улазних и излазних уређаја, као и портове за податке. Такође, дозвољава имплементирање многих других дизајнова, без употребе других компоненти.



Улази: Прекидачи и тастери

Четири тастера и осам прекидача служе као улази логичких кола. Улаз тастера је на почетку логичка „0“, његово стање се мења у логичку „1“ притискањем истог. Прекидачи генеришу константу „1“ или „0“, што зависи од њиховог положаја. Тастери и прекидачи користе серијске отпорнике као заштиту од кратког споја.



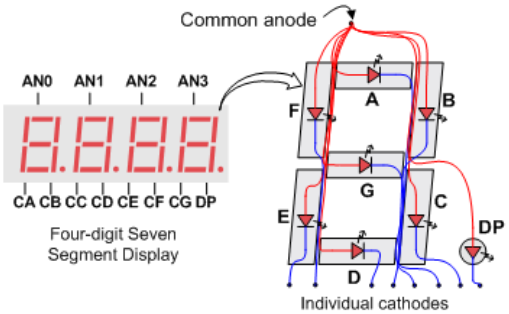
Излазне LED дидоде

Осам LED диода обезбеђено је за излазе кола. LED аноде покрећу се преко FPGA преко отпорника од 390Ω, тако да ће логичка ‘1’ на излазу осветлити са струјом од 3-4mA. Девета LED диода представља укључење LED, док десета LED диода представља индикатор статуса програмирања FPGA.

Излази: Седмо-сегментни дисплеј

Nexys2 плоча садржи четвороцифрену аноду седмо-сегментног LED дисплеја. Свака од четири цифре се састоји од седам сегмената, са убаченим LED-ом. LED-ови сегмента се могу одвојено осветлити (упалити), тако да сваки, од 128 шаблона, може приказати, паљењем или гашењем одређених сегмената. Од 128 шаблона, 10 који одговарају природним бројевима су најкориснији.

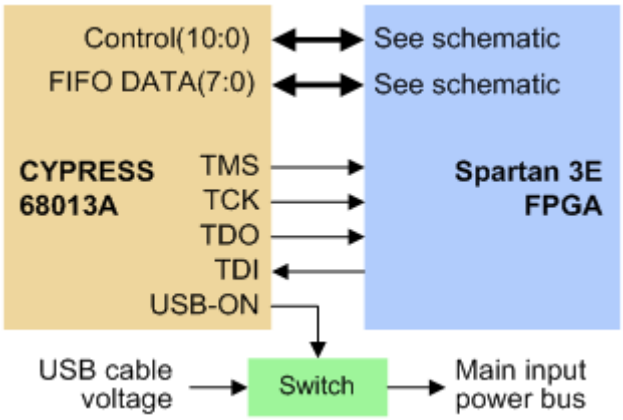
Аноде седам LED-ова, које формирају сваку цифру, спојени су у заједничку аноду, али LED катоде остају раздвојене. Сигнал заједничке аноде је доступан преко четири “digit enable” улазна сигнала. Катоде сличних сегмената повезане су, на сва четири дисплеја, у седам чворова кола, означених са CA до CG. Ових седам сигнала катода доступни су као улази за четвороцифрени дисплеј.



Како би свака од четири цифре светлела константно (или бар да тако делује људском оку), свака од четири цифре треба да се пали једном у интервалу од 1 до 16 ms, што значи да фреквенција њиховог паљења треба бити у опсегу од 1 KHz до 60 Hz. На пример, за фрекевенцују од 60Hz, цео дисплеј би се палио (гасио) на 16 ms, тј. свака цифра би се осветлила на сваку четвртину циклуса, или 4 ms.

USB Порт

Nexys2 укључује USB2 порт високе брзине који се базира на Cypress CY7C68013A USB контролеру. USB порт се може користити за програмирање на плочи Xilinx уређаја, и извршава пренос података брзином 38Mbytes/sec, и то обезбеђује напајање плоче. Програмирање се постиже са бесплатним Digilent Adept Suite софтвером. Преношење података се може извршити користећи Adept софтвер, или прилагођеним корисничким софтвером који је написан користећи Digilent API за приступ Nexys2 USB конекцији.

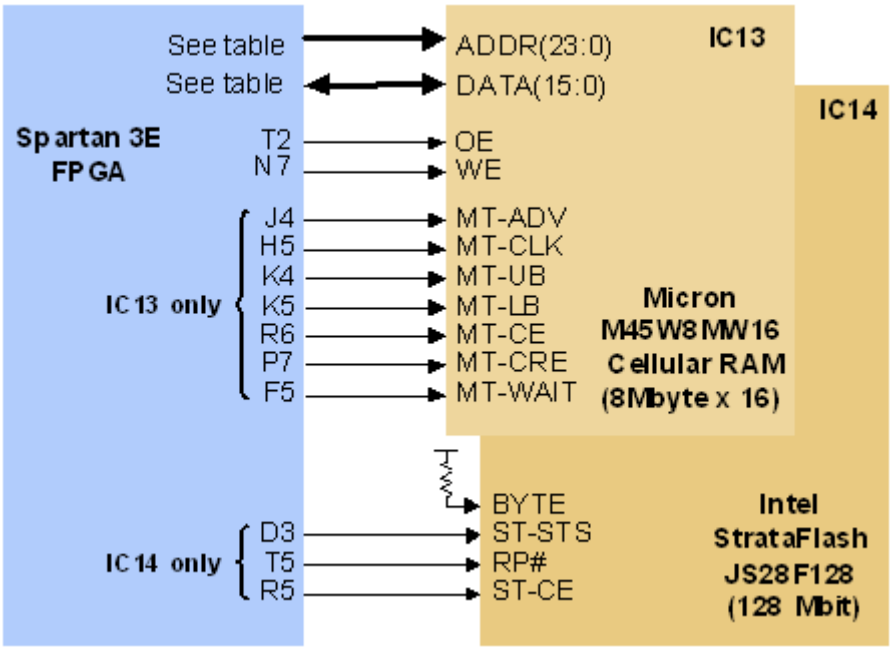


USB порт такође може да напаја Nexys2 плочу ако је изабрани извор напајања “USB”. USB спецификација омогућава да прикључени уређаји не користе више од 100mA док то не захтевају, након тога се може извући до 500mA. Када се први пут повеже на USB, Nexys2 плоча захтева 500mA, потом активира прекидач транзистор за повезивање USB кабла за главну магистралу извора напајања. Nexys2 плоча обично узима око 300mA са USB кабла, и треба водити рачуна о томе да се не извуче више од 500 mA (посебно када се користе периферне плоче).

Меморија

Nexys2 плоча има екстерне RAM и ROM уређаје. Екстерни RAM је „128Mbit Micron M45W8MW16 Cellular RAM pseudo-static DRAM“ уређај организован као 8 Мбајта x 16 битова. Може да ради као типични асинхрони SRAM са пиши и бриши циклусом од 70 ns, или као асинхрона меморија са магистралом од 80 MHz. Када функционише као асинхрони SRAM, „Cellular“ RAM аутоматски „рифрешује“ (освежава) унутрашње DRAM низове, дозвољавајући поједностављени дизајн меморијског контролера у FPGA. Када ради у синхроном маниру, могућ је непрекидан трансфер до 80 MHz.

Екстерни ROM је „128Mbit Intel TE28F128J3D75-110 StrataFlash“ уређај, организован као 8 Мбајта x 16 битова. Садржи 128 блокова, који се могу појединачно брисати, и садржи циклус од 110 ns за читање. Садржи 32-битни „пиши“ бафер, којим се може писати са циклусом од 70 ns, и 32-битни бафер се може пренети на флеш низ за 218 us.

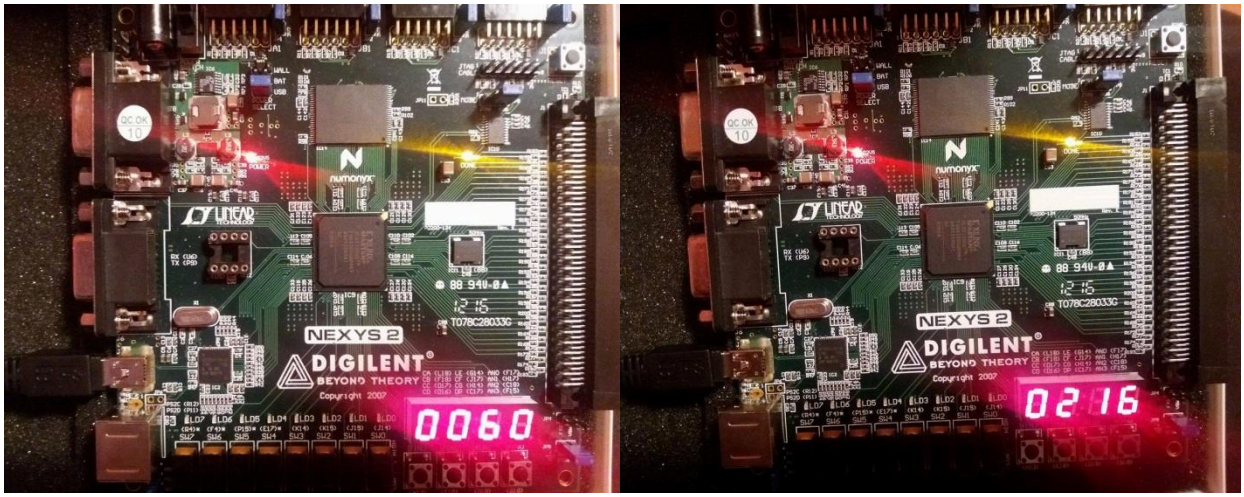


Оба уређаја деле заједничку 16-битну магистралу података и 24-битну адресну магистралу. Ћелијски RAM адресира помоћу битова, користећи сигнале вишег и нижег бита (MT-UB и MT-LB), али StrataFlash је конфигурисан за само 16 битне операције. “Output enable (OE)” и “write enable (WE)” сигнале имају оба уређаја, али сваки уређај има посебне “chip enable (CE)” сигнале. Поред тога, ћелијски RAM има клок (MT-CLK), чекање (MT-WAIT), адресу валидности (MT-ADV) и ”control register enable (MT_CRE)” сигнале који су на

располагању FPGA за употребу код синхроног трансфера, StrataFlash има “reset” (RP#) и “status” (STS) сигнале усмерене на FPGA.

Пројектни задатак

Дакле, као што је наведено у уводу, овај пројекат се бави реализацијом таксиметра у софтверском алату Xilinx, конкретније *Verilog-y*. Таксиметар ради по принципу штоперице, разлика је у томе што он своје одбројавање почиње од 60 (што је цена старта таксија) и инкрементирање има кашњење од 2.5 секунди, за разлику од штоперице која, јелте, има кашњење од 1 секунде. То је уједно и недостатак нашег таксиметра, јер реалан таксиметар вредност мери у односу на пређени пут, не узимајући у обзир само временски интервал вожње. Дакле, идеја је да паљењем прекидача почиње инкрементирање, гашењем истог добијамо коначни износ, који треба платити. Након завршене вожње, притискањем тастера тј. *button-a* број се ресетује назад на 60. Вредност, коју муштерија треба да плати, приказана је на седмо-сегментном дисплеју.



Реализација пројектног задатка

Први проблем на који смо наишли током реализације, јесте увођење колико-толико тачног кашњења. Одокативном методом смо одредили да то буде 2.5 секунде (та вредност нам делује као најтачнија). С обзиром да Nexys2 плоча има осцилатор (клок) од 50MHz, што значи да се циклус клока понови 50 милиона пута у временском интервалу од 1 секунде. Како бисмо направили одговарајуће кашњење, помножили смо фреквенцију осцилатора са 2.5:

$$50000000 \times 2.5 = 125000000$$

Овиме смо одредили до којег броја треба да „иде“ наш бројач. Коришћењем “assembler” програмског језика, тј. калкулатора који је уграђен у њему, добили смо вредност бројача у бинарном бројевном систему и избројали број битова тог броја, што је 27.
Дакле, притискањем тастера „reset” бројач треба да се врати на 0, исти исход добија се ако бројач изброји до 125000000, у било ком другом случају он се инкрементира.

```
36  always @ (posedge clk or posedge reset)
37  begin
38   if(reset)
39
40    brojac <= 0;
41
42   else if(brojac == 125000000) //ako stigne do max dozvoljene vrednosti, resetuj ga
43    brojac <= 0;
44   else if(start) //pocni da brojis, kad prekidac start dodje u stanje viseg bita
45    brojac <= brojac + 1;
46  end
```

Седмо-сегментни дисплеј није у могућности да вишецифрене бројеве посматра као целину, што значи да смо сваку јединицу, десетицу, стотину или хиљаду морали да представимо као појединачну цифру, како би дисплеј могао да је учита. Те појединачне цифре смо складиштили у посебним регистрима, који су се инкрементирали до броја 9, када би достигли тај број, ресетовали би се на 0, и тада би се инкрементирање наставило на следећем регистру. Тиме када одброји до 9, дисплеј ће показивати 10, а не „A“.

```
50  always @ (posedge clk or posedge reset)
51  begin
52   if (reset)
53    begin
54     reg_d0 <= 0;
55     reg_d1 <= 6;     // start u taxiju, 60 dinara
56     reg_d2 <= 0;
57     reg_d3 <= 0;
58    end
59
60   else if (click) //inkrementiraj na svaki click
61    begin
62     if(reg_d0 == 9) //xxx9 - kad dodjes do 9, vrati na 0
63     begin  //if_1
64      reg_d0 <= 0;
65
66      if (reg_d1 == 9) //xx99
67      begin  // if_2
68       reg_d1 <= 0;
69       if (reg_d2 == 5) //x599
70       begin //if_3
71        reg_d2 <= 0;
72        if(reg_d3 == 9) //9599
73         reg_d3 <= 0;
74        else
75         reg_d3 <= reg_d3 + 1;
76       end
77       else //else_3
78        reg_d2 <= reg_d2 + 1;
79      end
80
81      else //else_2
82       reg_d1 <= reg_d1 + 1;
83     end
84
85     else //else_1
86      reg_d0 <= reg_d0 + 1;
87    end
88  end
```

Као што се види у коду изнад, притискањем тастера ресет, цифре се ресетују на број „60“ (старт таксија).
У линији кода,

```
48  assign click = ((brojac == 125000000)?1'b1:1'b0);
```

увођењем нове жице *click* „преварили“ смо *Verilog*. Кад год би бројач одбројао до 125000000 жица би добила вредности логичке јединице, у супротном логичке нуле.

Касније у коду *click* се понаша као тастер, иако то није, и тиме „одобрава“ инкрементирање.

Следећи проблем био је како представити све претходно наведено на седмо-сегментном дисплеју. Раније је напоменуто да цифре на дисплеју треба толико брзо да се крећу да људско око такву промену не запази. Ту улогу добио је нови бројач, под називом *counter.*

```
reg [18:0]count; // 18-bitni brojac kako bi posmatrac stekao utisak da displej sve vreme radi
                 // ne da se pali/gasi
                 // pri frekvenciji ~ 1000Hz stice se takav utisak

always @ (posedge clk or posedge reset)
 begin
  if (reset)
   count <= 0;
  else
   count <= count + 1;
 end
```

Цифре седмо-сегментног дисплеја треба посматрати појединачно, не као целину. Када би бројач одбројавањем стигао до последња два бита почео би редом да укључује сваки од дисплеја, толиком брзином да посматрачу делује као да све четири раде истовремено.

```
always @ (*)
 begin
  case(count[18:17])        // kada brojac stigne do poslednje 2 cifre, pali odredjene cifre displeja

   2'b00 :
    begin
     sseg = reg_d0;          // prva cifra
     an_temp = 4'b1110;       // upaljena
     reg_dp = 1'b1;
    end

   2'b01:
    begin
     sseg = reg_d1;        // druga cifra
     an_temp = 4'b1101;    // upaljena
     reg_dp = 1'b1;
    end

   2'b10:
    begin
     sseg = reg_d2;       // treca cifra
     an_temp = 4'b1011;   // upaljena
     reg_dp = 1'b1;
    end

   2'b11:
    begin
     sseg = reg_d3;      // cetvrta cifra
     an_temp = 4'b0111;  // upaljena
     reg_dp = 1'b1;
    end
  endcase
 end
assign an = an_temp;
```

Коришћењем табеле,

Segments Inputs							7 Segment Display Output
a	b	c	d	e	f	g	
0	0	0	0	0	0	1	0
1	0	0	1	1	1	1	1
0	0	1	0	0	1	0	2
0	0	0	0	1	1	0	3
1	0	0	1	1	0	0	4
0	1	0	0	1	0	0	5
0	1	0	0	0	0	0	6
0	0	0	1	1	1	1	7
0	0	0	0	0	0	0	8
0	0	0	0	1	1	0	9

реализовали смо седмо-сегментни дисплеј, помоћу *case-a.*

```
143  reg [6:0] sseg_temp;
144  always @ (*)
145   begin
146    case(sseg)
147     0 : sseg_temp = 7'b1000000; // 0 na displeju
148     1 : sseg_temp = 7'b1111001; // 1 na displeju
149     2 : sseg_temp = 7'b0100100; // 2 na displeju
150     3 : sseg_temp = 7'b0110000; // 3 na displeju
151     4 : sseg_temp = 7'b0011001; // 4 na displeju
152     5 : sseg_temp = 7'b0010010; // 5 na displeju
153     6 : sseg_temp = 7'b0000010; // 6 na displeju
154     7 : sseg_temp = 7'b1111000; // 7 na displeju
155     8 : sseg_temp = 7'b0000000; // 8 na displeju
156     9 : sseg_temp = 7'b0010000; // 9 na displeju
157
158    endcase
159   end
160  assign {g, f, e, d, c, b, a} = sseg_temp;
161  assign dp = reg_dp; // dp- dot pointer, manipulisanje tackama na displeju
162
```

Закључак

У овом пројекту приказана је реална симулација таксиметра. Наш таксиметар функционише по принципу правог, укључивањем прекидача креће одбројавање рачуна, искључивањем истог добија се коначан исход који треба платити. Пројекат је примењив у пракси, искључујући наведени недостатак.

Литература

1. http://simplefpga.blogspot.rs
2. https://www.xilinx.com/support/documentation/data_sheets/ds312.pdf
3. https://reference.digilentinc.com/reference/programmable-logic/nexys-2/reference-manual
4. https://reference.digilentinc.com/reference/programmable-logic/nexys-2/reference-manual
5. http://www.electronicshub.org/seven-segment-displays/