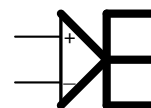


**FAKULTET TEHNIČKIH NAUKA
INSTITUT ZA ENERGETIKU, ELEKTRONIKU I TELEKOMUNIKACIJE
KATEDRA ZA ELEKTRONIKU
NOVI SAD
TRG DOSITEJA OBRADOVIĆA 6**



**(021) 459-449
kel@uns.ns.ac.yu**

UPUTSTVO ZA LABORATORIJSKE VEŽBE IZ MIKROPROCESORSKE ELEKTRONIKE

**- PROJEKTOVANJE EMBEDDED SISTEMA SA FPGA MODELOM
MIKROKONTROLERA 8051 -**

Novi Sad,
decembar 2007.

Fakultet tehničkih nauka
Katedra za elektroniku
Milan Lukić
Jovan Radak

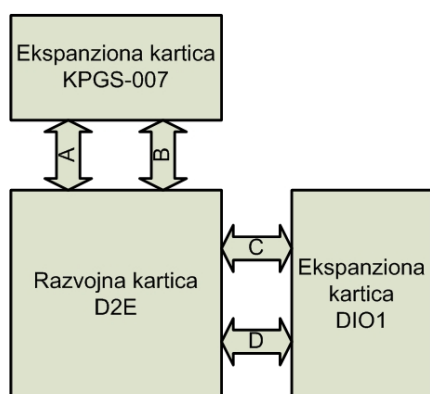
UVOD

Ovaj ciklus laboratorijskih vežbi će se baviti problematikom projektovanja digitalnih sistema zasnovanih na mikrokontroleru 8051. U daljem tekstu će biti opisan način realizacije digitalnih sistema korišćenjem modela mikrokontrolera 8051 implementiranog na FPGA čipu.

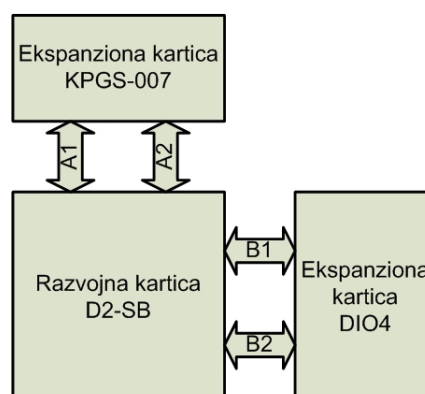
1. Razvojni sistem za projektovanje hardvera i softvera za 8051 sa FPGA kolom

U cilju ovladavanja metodologijom projektovanja mikrokontrolerskih digitalnih sistema (embedded sistemi), bilo je potrebno realizovati okruženje koje omogućava brzo i jednostavno programiranje i povezivanje mikrokontrolera sa većim brojem različitih perifernih uređaja. Pri tome korisnik (student) treba da ima "kreativnu slobodu" koja podrazumeva mogućnost konfigurisanja hardvera, tj načina na koji je kontroler povezan sa periferijama, kao i programiranja samog kontrolera. Takvu fleksibilnost je moguće postići korišćenjem FPGA kola u kojem se nalazi model mikrokontrolera 8051. FPGA kolo se nalazi na razvojnoj ploči, na koju je moguće spolja priključiti ekspanzione ploče, koje sadrže perifernijske uređaje kojima želimo da upravljamo pomoću kontrolera. Digitalni sistem se realizuje tako što se portovi kontrolera na FPGA kolu povežu sa odgovarajućim signalima za upravljanje periferijama na ekspanzionim karticama.

Za realizaciju razvojnog sistema za projektovanje hardvera i softvera za 8051 sa FPGA kolom iskorišćene su dve vrste razvojnih ploča sa *Xilinx SPARTAN XC2S200E* čipom. To su *Digilab 2E* i *Digilab D2-SB*. Obe varijante razvojne ploče se isporučuju zajedno sa odgovarajućom ekspanzionom karticom (*DIO1* kartica za *D2E* i *DIO4* kartica za *D2-SB*). Pored njih, na razvojne ploče moguće je povezati i dodatnu ekspanzionu karticu (*KPGS-007*) na kojoj se nalazi veći broj perifernih uređaja kojih nema na *DIO1* i *DIO4*. Prema tome, postoje dve konfiguracije razvojnog sistema u zavisnosti od toga koja je razvojna ploča korišćena. Na slici 1. prikazana je konfiguracija razvojnog sistema sa *D2E* pločom, a na slici 2. sa *D2-SB* pločom.



Slika 1. Razvojni sistem sa D2E pločom



Slika 2. Razvojni sistem sa D2-SB pločom

Ekspanzione kartice *DIO1* i *DIO4* na sebi imaju sledeće periferijske uređaje:

- Sedmosegmentni LED displej sa 4 cifre
- 8 LED dioda
- 3-bitni VGA port
- 8 prekidača
- 5 tastera
- PS/2 port

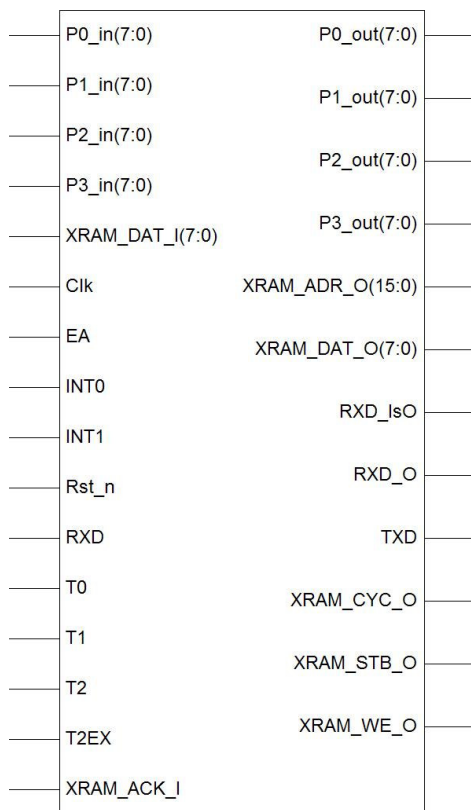
Ekspanzion kartica *KPGS-007* sadrži:

- Generator takt signala frekvencije 18.432MHz
- Interfejs za serijsku RS-232 komunikaciju
- Bidirekcioni PS/2 port
- Matrični LED displej 8x8
- 3 niza od po 8 LED dioda, u različitim bojama (crvena, žuta, zelena)
- SRAM memorija kapaciteta 128KB
- Konektor za tekstulani LCD displej (20x4 karaktera)
- Konektor za grafički LCD displej (128x64 pixela)
- Konektori za hibridni AM predajnik i prijemnik
- 8-bitni A/D konvertor sa osmokanalnim analognim multiplekserom

Načini povezivanja ekspanzionih kartica sa odgovarajućim razvojnim pločama dati su u prilogu, a mogu se naći sa detaljnim objašnjenjima i u zasebnim dokumentima koji se nalaze na sajtu [katedre za elektroniku](#).

2. Model mikrokontrolera 8051

Kao što je već naznačeno, razvojni sistem se bazira na modelu mikrokontrolera 8051 koji je implementiran unutar FPGA čipa. Ovaj model je realizovan u obliku skupa komponenti opisanih u jeziku VHDL. Model podražava logiku i celokupan set instrukcija standardnog 8052 kontrolera (verzija 8051 sa tajmerom 2 i 256 bajta internog RAM-a). Ipak, postoje izvesne modifikacije realizovanog modela u odnosu na standardnu kontrolersku arhitekturu familije 8051. Te modifikacije će biti razmotrene u nastavku i o njima treba voditi računa u procesu projektovanja sistema.



Slika 3. FPGA model mikrokontrolera 8051

2.1 Portovi (P0, P1, P2 i P3)

Kao što je poznato, u arhitekturi mikrokontrolera 8051 postoje četiri osmobitna ulazno-izlazna porta, označena sa P0, P1, P2 i P3. U svom prirodnom habitatu kontroler je zapakovan u 40-pinsko kućište, što znači da portovi zauzimaju 32 od 40 raspoloživih pinova. Ograničen broj pinova povlači sa sobom posledicu da pojedini pinovi pored toga što služe kao ulazno-izlazni pinovi opšte svrhe imaju dodeljene i alternativne funkcije poput tajmerskih i prekidnih ulaza, signala za upravljanje eksternom memorijom i sl. Uz to, bitno je napomenuti da su svi pinovi realizovani u tzv. *open drain* logici.

Nasuprot standardnoj arhitekturi, pri realizaciji modela kontrolera na FPGA kolu nije postojalo pomenuto ograničenje broja pinova, pa su stoga alternativnim funkcijama dodeljeni zasebni pinovi. To znači da za razliku od uobičajene situacije gde su kod kontrolera 8051 multipleksirani ulazno-izlazni pinovi sa pinovima specijalne namene, kod FPGA modela ove funkcije pinova su razdvojene, što rezultuje nešto većim brojem pinova.

Takođe, zbog nemogućnosti realizacije dvosmerne *open drain* logike, potpuno su razdvojeni ulazni od izlaznih portova. Svakom portu je dodeljen skup od po 8 ulaznih i 8 izlaznih linija. Naprimera, portu P0 je dodeljeno 8 ulaznih linija (P0_in(7:0)) i 8 izlaznih linija (P0_out(7:0)). Pri programiranju, treba voditi računa o tome da instrukcije koje očitavaju stanje portova čitaju stanje ulaznih linija, a nasuprot njima instrukcije koje izbacuju vrednost na port menjaju stanje izlaznih linija. Read-modify-write instrukcije kao i kod standardnog kontrolera 8051 čitaju izlazno stanje, modifikuju ga i ponovo ga ispisuju na izlaz. U tabeli 1. je prikazan efekat izvršenja pojedinih ulazno-izlaznih instrukcija.

Instrukcija	Efekat izvršenja
MOV P0, A	P0_out(7:0) <- A
MOV A, P3	A <- P3_in(7:0)
ANL P2, #0FEh	P2_out(7:0) <- P2_out & #0FEh

Tabela 1. Ulazno-izlazne instrukcije

2.2 Linije za povezivnje sa eksternom RAM memorijom

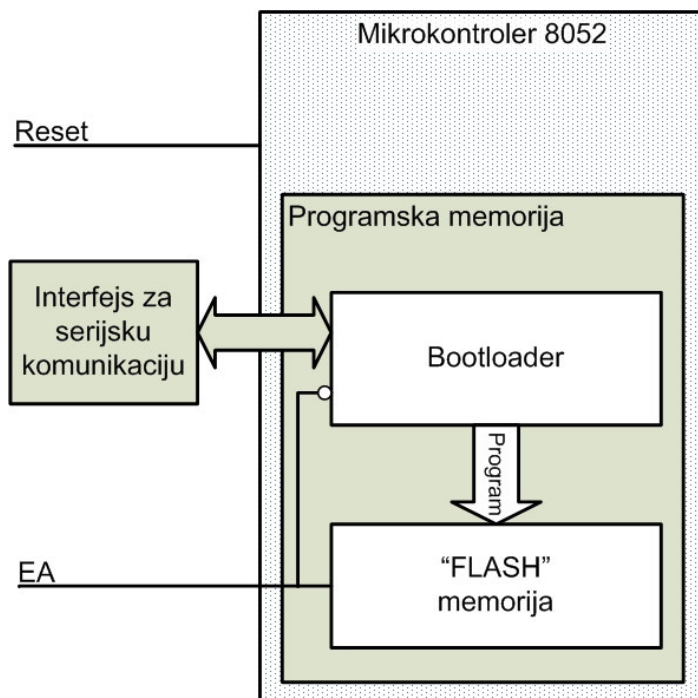
Još jedna razlika između standardnog mikrokontrolera 8051 i njegovog FPGA modela ogleda se u načinu njegovog sprežanja sa eksternom RAM memorijom. Kod standardne arhitekture postoje magistrala podataka i adresna magistrala koje se delimično preklapaju (nižih 8 bita adresne magistrale se preklapaju sa magistralom podataka), kao i tri kontrolna signala (ALE, RD i WR). FPGA model implementira tzv. *Wishbone Bus*. Ovo je standardan način međusobnog sprežanja modula koji se nalaze na istom integrisanom kolu koji se koristi u brojnim *open source* FPGA projektima. Model kontrolera 8051 ima implementiran *Wishbone* memorijski interfejs zbog kompatibilnosti sa drugim uređajima koji ga takođe koriste i koji bi se mogli naći u sistemu. Signali koji čine *Wishbone bus* su:

- XRAM_WE_O – signal indikacije da li je u pitanju ciklus čitanja ili upisa u eksternu memoriju; “0” na ovoj liniji označava čitanje, a “1” upis
- XRAM_STB_O – signal indikacije validnog prenosa podataka
- XRAM_CYC_O – signal indikacije da je validan ciklus na magistrali u toku
- XRAM_ACK_I – signal indikacije završetka ciklusa, od strane mem. uređaja
- XRAM_DAT_I (7:0) – ulazne linije magistrale podataka (8 bita)
- XRAM_DAT_O (7:0) – izlazne linije magistrale podataka (8 bita)
- XRAM_ADR_O (15:0) – adresna magistrala (16 bita)

Ovakav način povezivanja kontrolera sa eksternom memorijom ima još jednu specifičnost o kojoj treba voditi računa. Instrukcije čitanja ili upisa u eksternu memoriju su realizovane na taj način da kontroler **ne prelazi na sledeću instrukciju sve dok ne očita visok logički nivo na liniji XRAM_ACK_I**. Dakle, jednom započet ciklus upisa ili čitanja traje sve dok memorija ne signalizira kontroleru da je ciklus završen, postavljanjem linije XRAM_ACK_I na visok logički nivo. To znači da ako npr. želimo da upišemo podatak pomoću instrukcije *MOVX @dptr, a* u memoriju ili memorijski mapiranu periferiju koja nema signal kojim bi obavestila kontroler da je ciklus upisa završen, moglo bi se desiti da program kontrolera ostane “zaglavljen” u instrukciji upisa. Ovaj problem moguće je prevazići tako što se linija XRAM_ACK_I kratko spoji na logičku jedinicu.

2.3 Programska memorija

FPGA model programske memorije bi po svojoj prirodi trebalo da modeluje FLASH memoriju kapaciteta 4KB u kojoj se nalaze programske instrukcije. To na prvi pogled znači da bi programska memorija mogla biti reprezentovana jednom relativno jednostavnom ROM strukturom obuhvaćenom modelom kontrolera 8051. Međutim, ovakav pristup ima jednu veliku manu koja se ispoljava u procesu razvoja softvera. Za svaku, makar i najmanju izmenu programskog koda bilo bi potrebno prvo iskonvertovati novu izvršnu verziju programa (.hex fajl) u njen odgovarajući reprezent (.vhd fajl) uključen u FPGA model kontrolera, a zatim bi bilo potrebno rekompajlirati ceo projekat u *Xilinx ISE* okruženju i na kraju spustiti izmenjeni model u FPGA kolo. Ovaj proces je nedopustivo spor, pa je bilo neophodno pronaći rešenje koje pojednostavljuje proces programiranja kontrolera, po cenu nešto složenije organizacije programske memorije. Konačna realizacija programske memorije je šematski prikazana na slici 4.



Slika 4. Programska memorija

Programska memorija je podeljena na dva dela. Prvi deo je ROM u kojem se nalazi program koji se naziva *bootloader*. Uloga *bootloadera* je da preko serijskog porta kontrolera prima instrukcije korisničkog programa i smešta ih u drugi deo programske memorije. Taj drugi deo je RAM memorija koja reprezentuje FLASH memoriju. Mikrokontroler u svakom trenutku čita programske instrukcije iz jednog od ova dva dela programske memorije u zavisnosti od EA signala. Kada je EA signal na logičkoj nuli, izvršava se *bootloader*, a kada je na jedinici izvršava se korisnički program.

Ovakva organizacija programske memorije znatno olakšava razvoj softvera. Kada je hardver isprojektovan i konačno iskompajliran i pušten u FPGA kolo, korišćenjem *bootloadera* je moguće unositi neograničen broj izmena u program, bez potrebe za rekompajliranjem celog FPGA projekta. Proces programiranja kontrolera podrazumeva sledeće korake:

1. Inicijalno programiranje FPGA kola, čime se konfiguriše hardver sistema
2. Spuštanje EA linije na logičku nulu, a zatim reset kontrolera. Na ovaj način se pokreće *bootloader* i programska memorija je spremna da primi program.
3. *Bootloader* preko serijskog porta kontrolera komunicira sa aplikacijom na računaru. Ova aplikacija omogućava učitavanje korisničkog hex fajla, tj. programskog koda u “flash” memoriju kontrolera.
4. Pošto je program učitao u programsku memoriju, potrebno je promeniti stanje EA linije na logičku jedinicu i resetovati kontroler. Na ovaj način kontroler počinje sa izvršavanjem korisničkog koda.

3. Projektovanje sistema sa FPGA modelom 8051

Pošto su u prethodnom poglavlju objašnjene specifičnosti, odnosno ključne razlike FPGA modela i klasične arhitekture mikrokontrolera 8051, ostaje da se kaže još par reči o procesu projektovanja embedded sistema zasnovanog na modelu 8051.

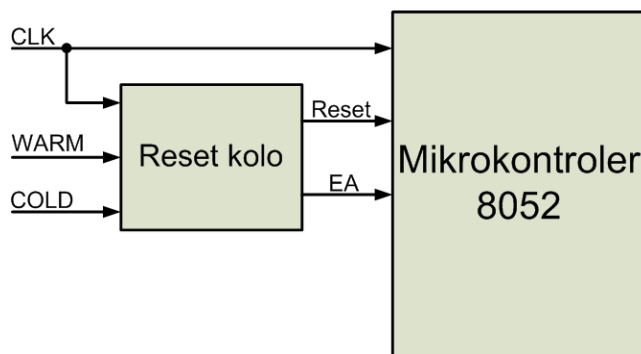
Projektovanje sistema korišćenjem razvojnog sistema sa *Xilinx SPARTAN XC2S200E* čipom podrazumeva kreiranje projekta u *Xilinx ISE* okruženju. Projekat treba da uključi model kontrolera 8051 koji se povezuje sa perifernim uređajima na ekspanzionim karticama. Na sajtu [katedre za elektroniku](#) se nalazi projekat *Maketa.isc* koji u sebe već ima uključene dve komponente:

- Model kontrolera 8051
- Reset kolo

U projekat je potrebno dodati top modul koji je hijerarhijski iznad ova dva već postojeća modula. Top modul ih instancira i povezuje sa perifernim uređajima na način koji odgovara samoj arhitekturi sistema koji želimo da realizujemo. Njegova implementacija može biti u jeziku VHDL ili korišćenjem šematskog editora uključenog u *Xilinx ISE*.

3.1 Reset kolo

Reset kolo koje je uključeno u projekat *Maketa.isc* samo po sebi nije neophodno, ali je stavljeno da bi olakšalo korisniku prelazak iz režima *bootloadera* na izvršavanje korisničkog programa i obrnuto. Na slici 5. šematski je prikazano reset kolo i način njegovog povezivanja sa modelom kontrolera 8051.



Slika 5. Povezivanje reset kola sa modelom kontrolera

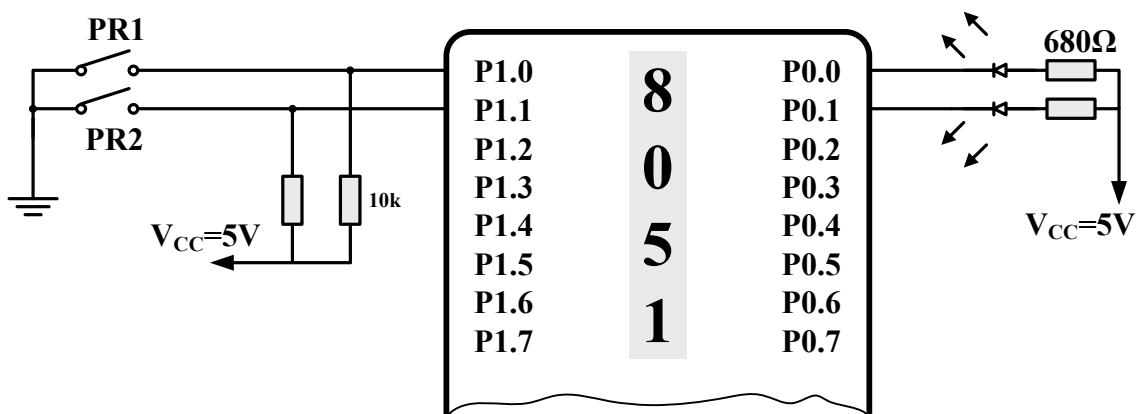
Dve ulazne linije reset kola (WARM i COLD) potrebno je povezati sa tasterima na ekspanzionoj kartici. Pritiskom na taster povezan sa COLD linijom, korisnik izaziva tzv. “hladni reset”. To podrazumeva istovremeno spuštanje EA linije na logičku nuli i reset kontrolera. Na ovaj način se pokreće *bootloader* i kontroler je spreman da preko serijskog porta primi korisnički program (hex fajl) i da ga pohrani u programsku memoriju. Kada je program učitao u programsku memoriju, potrebno je pritisnuti taster povezan na WARM liniju, što dovodi do “vrućeg reset”, odnosno postavljanja EA linije na jedinicu uz

istovremeni reset kontrolera. Po otpuštanju tastera kontroler počinje sa izvršavanjem korisničkog programa.

Dakle, najosnovnija varijanta embedded sistema treba da u sebe uključi model mikrokontrolera 8051 i reset kolo. Ulazi reset kola se povezuju sa tasterima na ekspanzionoj DIO ploči. Da bi bilo moguće korišćenje *bootloadera*, potrebno je povezati RxD i TxD linije na kontroleru sa interfejsom za serijsku komunikaciju na ekspanzionoj ploči KPGS-007. Sa iste ploče se dovodi i takt signal frekvencije 18.432MHz.

U nastavku će biti opisan tok projektovanja kompletnog sistema, sa stanovišta projektovanja hardvera i softvera. U jednoj od prethodnih vežbi je urađen primer sa dva prekidača, povezana na portu 1, i dve diode (crvene i zelene) povezane na port 0. Za početak ćemo se podsetiti kako je izgledao program napisan u C-u koji izvršava upravljanje radom LED dioda u skladu sa stanjem prekidača. Nakon toga ćemo pristupiti dizajnu sistema realizovanog na FPGA kolu, koji će imati istu funkcionalnost.

PRIMER: Kontrola rada LED. Sa mikrokontrolerom 8051, čija je frekvencija spoljašnjeg oscilatora 18.432MHz izvršiti kontrolu rada dve LED diode. Izbor LED diode se vrši prekidačem PR1 (kada je PR1 u stanju logičke "1" trepće crvena, a kada je u stanju logičke "0" trepće zelena LED dioda), a brzina treperenja LED diode se određuje prekidačem PR2 (kada je PR2=1 dioda treba da trepće učestanošću od 5Hz, a kada je PR2=0, dioda treba da trepće učestanošću od 0.5Hz). Obe LED diode treba da se uključe na visok logički nivo nožice P0.0 (crvena) i P0.1 (zelena).



Slika 6. Kontrola dve LED diode pomoću prekidača

Rešenje:

Oscilator koji se koristi na maketi ima frekvenciju 18,432MHz. Uz tako visoku frekvenciju oscilatora neophodno je tajmerom meriti frekvenciju 100Hz, a onda brojanjem prekida doći do željene frekvencije od 5, odnosno 0.5Hz. Prema tome, tajmer treba podesiti u skladu sa frekvencijom 100Hz: $(18,432\text{MHz}/12)/(100\text{Hz}/2)=7680$. Kada je prekidač otvoren treba da se desi 20 prekida, a kada je zatvoren 200 prekida pre nego što se izviši promena logičkog nivoa signala na nožici P0.0 ili P0.1!

U Timer0 je potrebno uneti vrednost: **0-7680=0000h-1E00h=E200h.**

```
#include <reg51.h>
//def. globalnih promenljivih
unsigned char brojac; // brojač prekida
unsigned char frekv;
unsigned char maska;
sbit PR1 = P1^0;
sbit PR2 = P1^1;
sbit CRVENA = P0^0;
sbit ZELENA = P0^1;

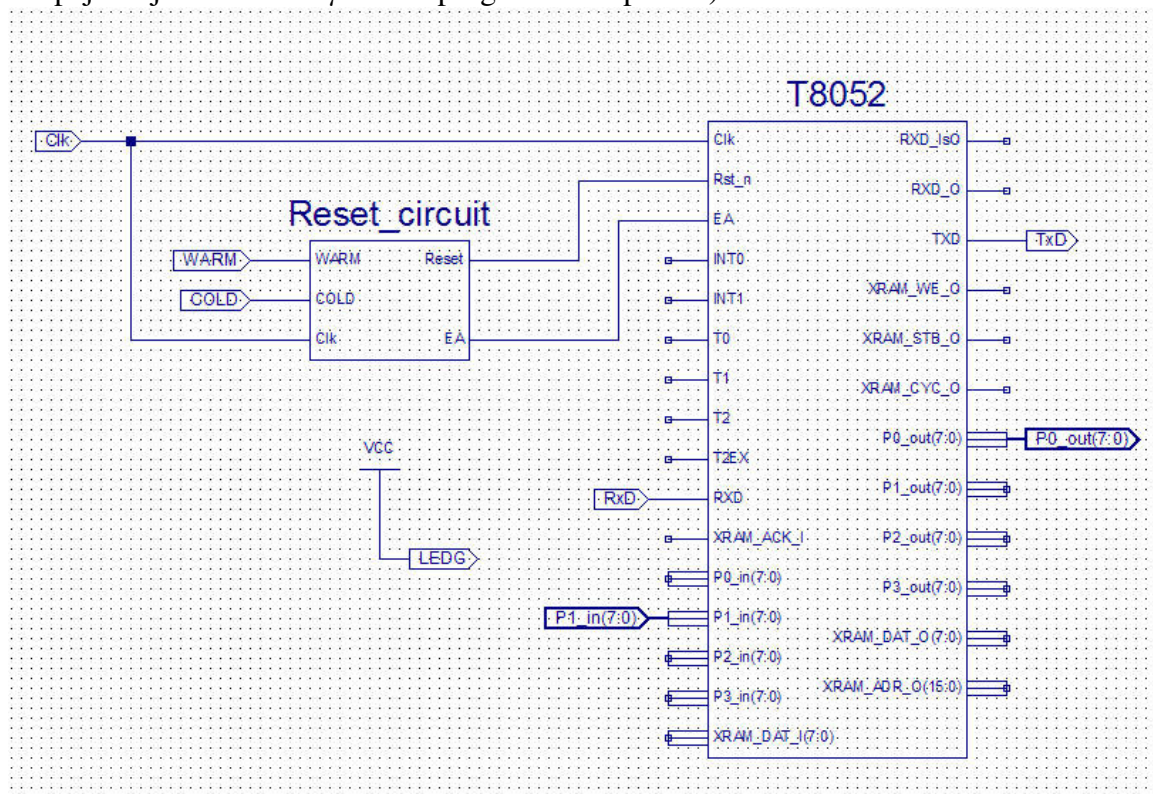
void Inicijalizacija (void)
{
    EA = 1; // dozvoljeno je bit adresibilno postavljanje
interrupta
    ET0 = 1; // dozvola interrupta prekoračenja Timera0
    TMOD = 0x01; // T0 je 16-bitni Timer
    TH0 = 0xE2; // gornji bajt Timera0
    TL0 = 0x00; // donji bajt Timera0
    brojac = 1;
    frekv = 1;
    TR0 = 1; // Timer0-on
}

void timer0 (void) interrupt 1 using 2 // prekidni potprogram
{
    TR0 = 0; // tajmer je potrebno zaustaviti pre punjenja
    TH0 = 0xE2; // potrebno je ponovo inicijalizovati Timer0 jer on
    TL0 = 0x00; // kad jednom odbroji nastavlja da broji od 0
    TR0 = 1; // tajmer se ponovo startuje
    if (--brojac)
    {
        brojac = frekv;
        P0 ^= maska;
    }
}

void main(void) //glavni program
{
    Inicijalizacija(); // poziv potprograma za inicijalizaciju tajmera0 i porta P0
    while(1) // glavni program se vrti u ovoj petlji i proverava stanje
    {
        // spoljašnjih prekidača i čeka na prekide tajmera 0
        if (PR1) // crvena
        {
            maska = 1;
            ZELENA = 0;
        }
        else // zelena
        {
            maska = 2;
            CRVENA = 0;
        }
        if (PR2) // 10Hz
            frekv = 20;
        else // 0.5Hz
            frekv = 200;
    }
}
```

Jedan od načina da se kompletan sistem realizuje na FPGA kolu jeste da se model mikrokontrolera, zajedno sa sadržajem programske memorije sintetiše korišćenjem Xilinx Foundation ISE-a. Ovaj proces zahteva relativno mnogo vremena (čak i kada se sinteza izvršava na računarima sa najboljim performansama). Problem postaje još ozbiljniji kada je neophodno više puta uzastopno konfigurisati FPGA kolo, nakon što se primeti da program ne radi u skladu sa očekivanom funkcionalnošću. Zbog toga bi bilo idealno ako bismo uspjeli da na neki način pojednostavimo mehanizam učitavanja novog programa u programsku memoriju. Tada bismo u FPGA kolu mogli da sintetišemo mikrokontroler sa svim neophodnim periferijama, a sadržaj programske memorije bi se mogao naknadno menjati proizvoljan broj puta bez nekih prevelikih vremenskih zahteva.

Da bi ova ideja mogla da bude realizovana, neophodno je dizajnirati hardver koji će biti jednom sintetisan u FPGA kolo, a koji će moći da na jednostavan i brz način omogući promenu sadržaja programske memorije u skladu sa HEX fajlom (dobijenim kompajliranjem C koda u μ Vision programskom paketu).

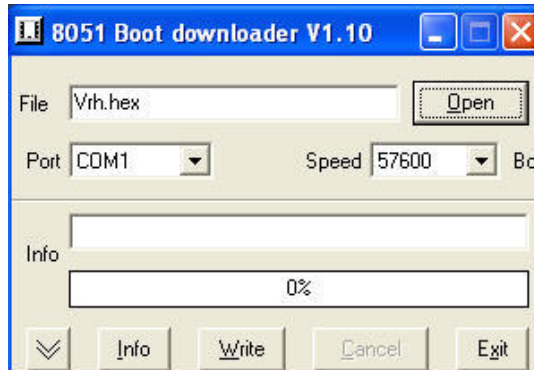


Slika 7. Šema hardvera koji će biti implementiran u FPGA kolu

Hardver koji će biti fiksiran u FPGA kolu je prikazan na slici 7. Prikazani hardver je konfigurisan tako da se nakon aktiviranja COLD reseta pokreće tzv. **bootloader**, mali program koji je već implementiran unutar prikazanog modela mikrokontrolera 8052. Njegova uloga je da od softvera na PC računaru, putem serijskog porta, primi HEX fajl kojim želimo da isprogramiramo programsku memoriju mikrokontrolera. Nakon prijema, bootloader upisuje sadržaj HEX fajla u programsku memoriju. Po završetku ovog procesa, pritiskom na WARM reset, pokreće se program koji je upravo upisan u programsku memoriju (a ne bootloader-u skladu sa logikom opisanom na slici 5).

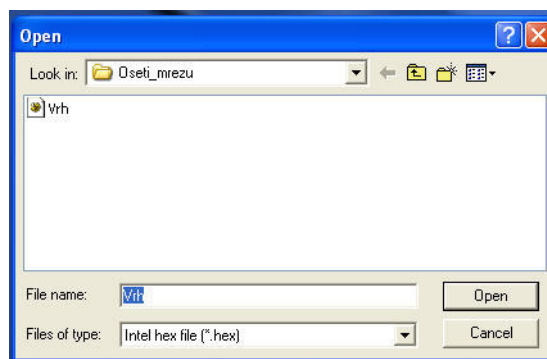
Način korišćenja bootloader-a:

1. Startuje se program bootloader.exe koji se nalazi na računaru (slika 8).



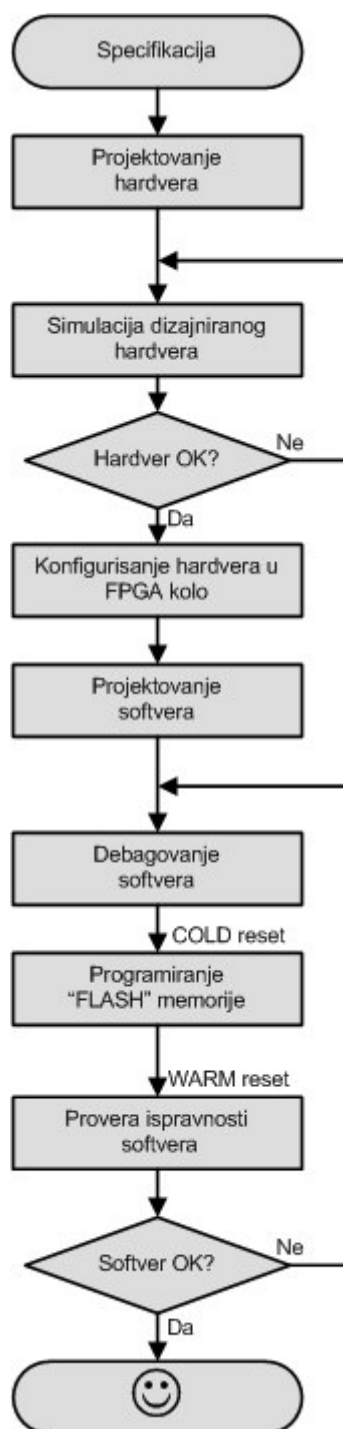
Slika 8. Program bootloader.exe

2. Podesi se **Port** na COM1 i brzina serijske komunikacije (**Speed**) na 57600 Bd
3. Klikom na **Open** otvara se dijalog u kome treba odabrati željeni HEX fajl (slika 9).



Slika 9. Dijalog za izbor .hex fajla

4. Pritisnuti taster za "**COLD reset**" na maketi (čime omogućavamo upis odabranog HEX fajla u memoriju)
5. Klikom na **Write** započinje transfer HEX fajla od računara ka FPGA kolu (odnosno ka mikroprocesoru 8052 sintetisanim unutar njega)
6. Kada se transfer završi, pritiskom na taster "**WARM reset**" na maketi počinje izvršavanje upravo primljenog HEX fajla.



Slika 10. Koraci u projektovanju sistema

PRILOG: veza pinova FPGA sa ekspanzionim pločama

P2 konektor (A2 na D2-SB, odnosno B na 2E)				P1 konektor (A1 na D2-SB, odnosno A na 2E)			
Pin #	Signal	FPGA pin		Pin #	Signal	FPGA pin	
		D2-SB	2E			D2-SB	2E
1	N.C. (not connected)	-	-	1	GND	-	-
2	N.C.	-	-	2	VU	-	-
3	N.C.	-	-	3	N.C.	-	-
4	-ADC_INT	p162	p194	4	TxD	p112	p68
5	PS2_CLK	p161	p193	5	RxD	p111	p64
6	LCD_RS	p160	p192	6	A15	p110	p63
7	PS2_CLK_DIR	p152	p191	7	-WR	p109	p62
8	LCD_RW	p151	p189	8	A16	p108	p61
9	PS2_DATA	p150	p188	9	A13	p102	p60
10	GLCD_RST	p149	p187	10	A14	p101	p59
11	*GCLK2	-	p185	11	A8	p100	p58
12	N.C.	-	-	12	A12	p99	p57
13	PS2_DATA_DIR	p146	p181	13	A9	p98	p56
14	N.C.	-	-	14	A7	p97	p55
15	N.C.	-	-	15	A11	p96	p49
16	N.C.	-	-	16	A6	p95	p48
17	N.C.	-	-	17	-RD	p94	p47
18	N.C.	-	-	18	A5	p93	p46
19	N.C.	-	-	19	A10	p89	p45
20	N.C.	-	-	20	A4	p181	p44
21	N.C.	-	-	21	D7	p87	p43
22	N.C.	-	-	22	A3	p180	p42
23	N.C.	-	-	23	D6	p179	p41
24	N.C.	-	-	24	A2	p178	p40
25	N.C.	-	-	25	D5	p176	p36
26	N.C.	-	-	26	A1	p175	p35
27	N.C.	-	-	27	D4	p174	p34
28	N.C.	-	-	28	A0	p173	p33
29	N.C.	-	-	29	D3	p169	p31
30	N.C.	-	-	30	D2	p168	p30
31	N.C.	-	-	31	S0	p167	p29
32	N.C.	-	-	32	D1	p166	p27
33	N.C.	-	-	33	S1	p165	p24
34	N.C.	-	-	34	D0	p164	p23
35	N.C.	-	-	35	S2	p163	p22
36	N.C.	-	-	36	N.C.	-	-
37	N.C.	-	-	37	N.C.	-	-
38	N.C.	-	-	38	N.C.	-	-
39	*GCLK1	p80	-	39	N.C.	-	-
40	GND	-	-	40	N.C.	-	-

* Ulazi za takt signal.

Tabela 2. Veza FPGA pinova sa pinovima ekspanzione ploče KPGS-007

Pin na ekspanzionoj kartici (DIO1 i DIO4)	FPGA pin na D2E kartici	FPGA pin na D2-SB kartici	Opis funkcije
BTN1	P149	p47	Tasteri
BTN2	P150	p46	
BTN3	P151	p86	
BTN4	P152	p84	
BTN5	P178	p83	
SW1	P126	p71	Prekidači
SW2	P129	p69	
SW3	P133	p64	
SW4	P135	p62	
SW5	P138	p60	
SW6	P140	p58	
SW7	P145	p56	
SW8	P147	p49	
LED1	P154	p111	Ledovke
LED2	P161	p109	
LED3	P163	p102	
LED4	P165	p100	
LED5	P167	p98	
LED6	P169	p96	
LED7	P174	p94	
LED8	P176	p89	
LEDG	P179	p88	Dozvola ledovki
SDP	P148	p48	Decimalna tačka
S0	p127	p70	Segmenti
S1	P132	p68	
S2	P134	p63	
S3	P136	p61	
S4	P139	p59	
S5	P141	p57	
S6	P146	p55	
AN1	P160	p82	Selekcija cifre na 7. segmentnom displeju
AN2	P162	p81	
AN3	P164	p75	
AN4	P166	p74	

Tabela 3. Veza FPGA pinova sa pinovima ekspanzionih ploča DIO1 i DIO4 kada su povezane na ekspanzione portove C i D ploče D2E, odnosno B1 i B2 ploče D2-SB