

Vežba 4

Povezivanje periferija sa 8051 mikrokontrolerom

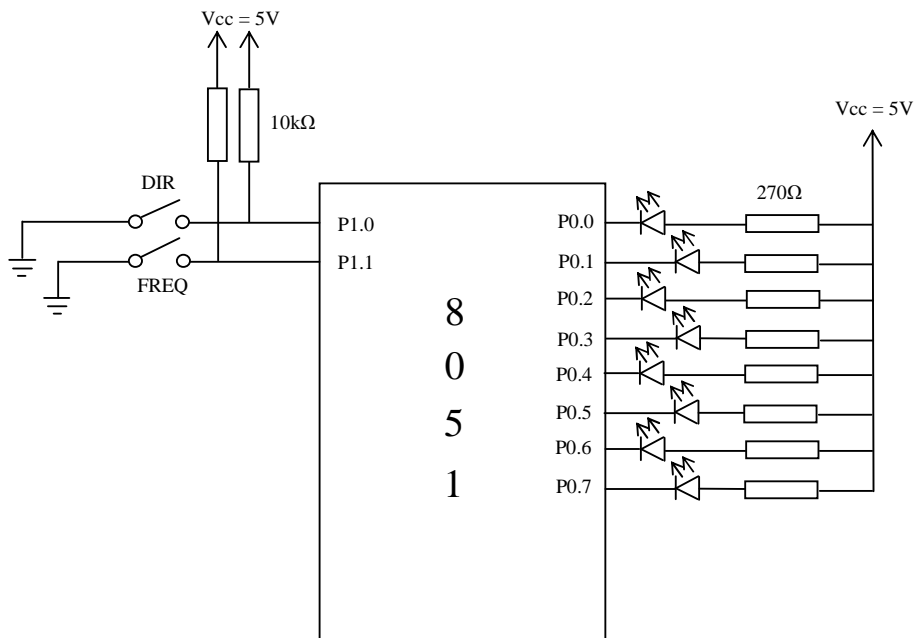
1. DIREKTNO POVEZIVANJE

Mikrokontroler 8051 ima dosta mogućnosti za povezivanje sa periferijama, ali je, u suštini, ograničen brojem fizičkih nožica za povezivanje tj. brojem nožica na portovima. Periferni uređaji mogu biti: svetleće diode (LED), sedmosegmentni LED displeji, prikidači, memorije, drajverska kola za motore, razni digitalni senzori itd.

Direktno povezivanje periferija sa mikrokontrolerom podrazumeva da se periferije vezuju na portove (P0, P1, P2 i P3). Otuda i naziv **direktno povezivanje**, jer nisu potrebne nikakve dodatne elektronske komponente da bi kompletan sistem funkcionisao u skladu sa željenom funkcionalnošću. Prednost ovakvog upravljanja periferijama je svakako jednostavnost celokupnog dizajna - kako sa stanovišta hardvera, tako i sa stanovišta softvera, jer se pokazuje da je softversko kontrolisanje periferija prilično jednostavnije kada su one povezane direktno na portove mikrokontrolera. Mana ovakvog dizajna je svakako u tome što samo ograničen broj periferija može biti povezan sa mikrokontrolerom, imajući u vidu da na raspolaganju imamo samo 4 osmootna porta. Ipak, vrlo često je broj periferija korišćenih u nekom mikrokontrolerskom sistemu mali, pa je sasvim opravdano koristiti upravo ovaj način povezivanja sa mikrokontrolerom.

Na sledećem primeru ćemo videti kako se direktno na mikrokontroler mogu povezati periferije i kako se može njima upravljati.

Primer 1: Na mikrokontroler 8051, čija je frekvencija spoljašnjeg oscilatora 12MHz, povezan je niz od 8 LED dioda na portu P0. Na portu P1 se nalaze dva prekidača-jedan od njih (DIR) se nalazi na P1.0, a drugi (FREQ) na P1.1. Prekidač DIR određuje smer u kome će se rotirati uključene diode tako da se one rotiraju u desno kada je DIR u otvorenom položaju, odnosno u levo kada je DIR zatvoren. Prekidač FREQ određuje frekvenciju kojom će diode da se rotiraju. Kada je FREQ otvoren frekvencija obrtanja je 20Hz, dok je 0.5Hz kada je FREQ zatvoren. Mikrokontroler isprogramirati u assembleru.



Rešenje:

Kada definišemo frekvenciju rotiranja dioda, možemo da je definišemo na dva načina. Možemo reći da je period koji odgovara frekvenciji od 20Hz (50ms) ustvari period jednog pomeraja dioda, ali možemo reći i da je to ustvari period za koji se desi 8 pomeraja. Ova druga definicija frekvencije rotiranja ima možda više smisla jer nakon 8 pomeraja ponovo ćemo imati istu sliku uključenih i isključenih dioda, pa možemo reći da je onda zapravo 50ms osnovna perioda rotiranja. Broj tajmerskih ciklusa koji treba da se odbroji kada je FREQ otvoren je tada $N1 = (50ms/8)/1\mu s = 6250$, a kada je zatvoren $N2 = (2s/8)/1\mu s = 250000$. Kako 16-bitni tajmer može

maksimalno da odbroji 65536 puta potrebno je uvesti pomoćni brojač u prekidnoj rutini za detekciju broja prekida. Pošto je $N2=40*6250$ vidi se da je potrebno da se desi 40 prekida pa da se onda izvrši rotiranje dioda u desno ili u levo, za frekvenciju od 0,5Hz. U Timer0 je potrebno uneti vrednost: $0-6250=0000h-186Ah=E796h$

Program:

```
F20    EQU    1                ; dodela vrednosti 1 simbolu F20
                                ; (označava da je na frekv.
                                ; 20Hz potreban 1 prekid pre
                                ; promene stanja LED dioda)
F05    EQU    40               ; dodela vrednosti 40 simbolu F05
                                ; (označava da je na frekv. 0.5Hz
                                ; potrebno 40 prekida pre promene
                                ; stanja LED dioda)
DIR     BIT    P1.0            ; ulazno stanje prekidača DIR
FREQ    BIT    P1.1            ; ulazno stanje prekidača FREQ

DSEG                                ; naredni segment se odnosi na
                                ; interni RAM
        ORG    20h              ; naredni podaci se smestaju od
                                ; adrese 20h unutar internog RAM-a
Diode:   DS      1              ; rezervisanje jednog bajta
                                ; (biće na adresi 20h)
Count:   DS      1              ; rezervisanje jos jednog bajta
                                ; (biće na adresi 21h)

CSEG                                ; nastavak predstavlja programski kod
                                ; i smeštaće se u programsku memoriju
        ORG    0000H            ; program se smešta u programsku memoriju
                                ; počev od adrese zadate sa ORG: 0h
        JMP    INICIJALIZACIJA

        ORG    000BH            ; kao odgovor na prekid tajmera 0,
                                ; skaće se na odresu 000Bh. Zato se
                                ; naredna instrukcija mora naći na
                                ; adresi 000Bh
        JMP    PREKID           ; skok na prekidnu rutinu

INICIJALIZACIJA:
        MOV    IE,#82H          ; dozvola interapta
                                ; prekoračenja Tajmera0
        MOV    TMOD,#01H        ; tajmer 0 je 16-bitni
        MOV    TH0,#E7H        ; početna vrednost
        MOV    TL0,#96H        ; tajmer 0 je 9E58h
        MOV    R4,#1            ; R4 <- 1, R4 je pomoćni brojač
                                ; za brojanje N2
        MOV    A,#3CH           ; postavljanje početnog stanja dioda
        MOV    P0, A
        MOV    Diode, A         ; sačuvaj sliku dioda u promenljivoj Diode
        SETB   TR0              ; start tajmera 0

MAIN:                                ; program se vrti u ovoj petlji
                                ; (proverava prekidač FREQ) i čeka
                                ; prekide
        MOV    A,#F20           ; Izbor frekvencije
        JB     FREQ,PostaviBrojac
        MOV    A,#F05
PostaviBrojac:
        MOV    Count,A
        SJMP   MAIN             ; nazad u petlju

PREKID:
        CLR    TR0              ; Tajmer0 je potrebno zaustaviti
```

```

MOV TH0,#E7H      ; početna vrednost
MOV TL0,#96H      ; Timera0 je 9E58h
SETB  TR0
DJNZ  R4,Int_Kraj
MOV   R4,Count    ;ponovo napuni brojač u skladu sa FREQ
MOV  A, Diode      ;uzmi trenutnu sliku porta 0
JB   DIR, Desno    ;ako je DIR = 1 treba rotirati u desno
RL   A             ;u suprotnom u levo
SJMP Ispis
Desno:
RR   A
Ispis:
MOV  P0, A         ;pošalji na port 0 novo stanje dioda
MOV  Diode, A      ;i sačuvaj to stanje kao trenutnu sliku
Int_Kraj:
RETI               ;za sledeći prekid
END

```

U ovom jednostavnom primeru smo videli kako upravljamo diodama koje su direktno povezane na portove mikrokontrolera. Očigledno je, takođe, da smo na sličan način mogli da spojimo još dva niza od po 8 dioda (npr. na port 2 i port 3), ali već sa četvrtim bismo imali problem, jer su neke linije porta 1 već zauzete (na njima se nalaze prekidači DIR, FREQ i prekidači koje bismo morali da dodamo za upravljanje diodama na portu 2 i na portu 3). Ukoliko rešavanje nekog problema zahteva više periferija, njih više ne možemo direktno povezivati na portove već moramo koristiti *memorijsko mapiranje periferijskih jedinica*!

2. MEMORIJSKO MAPIRANJE PERIFERIJSKIH JEDINICA

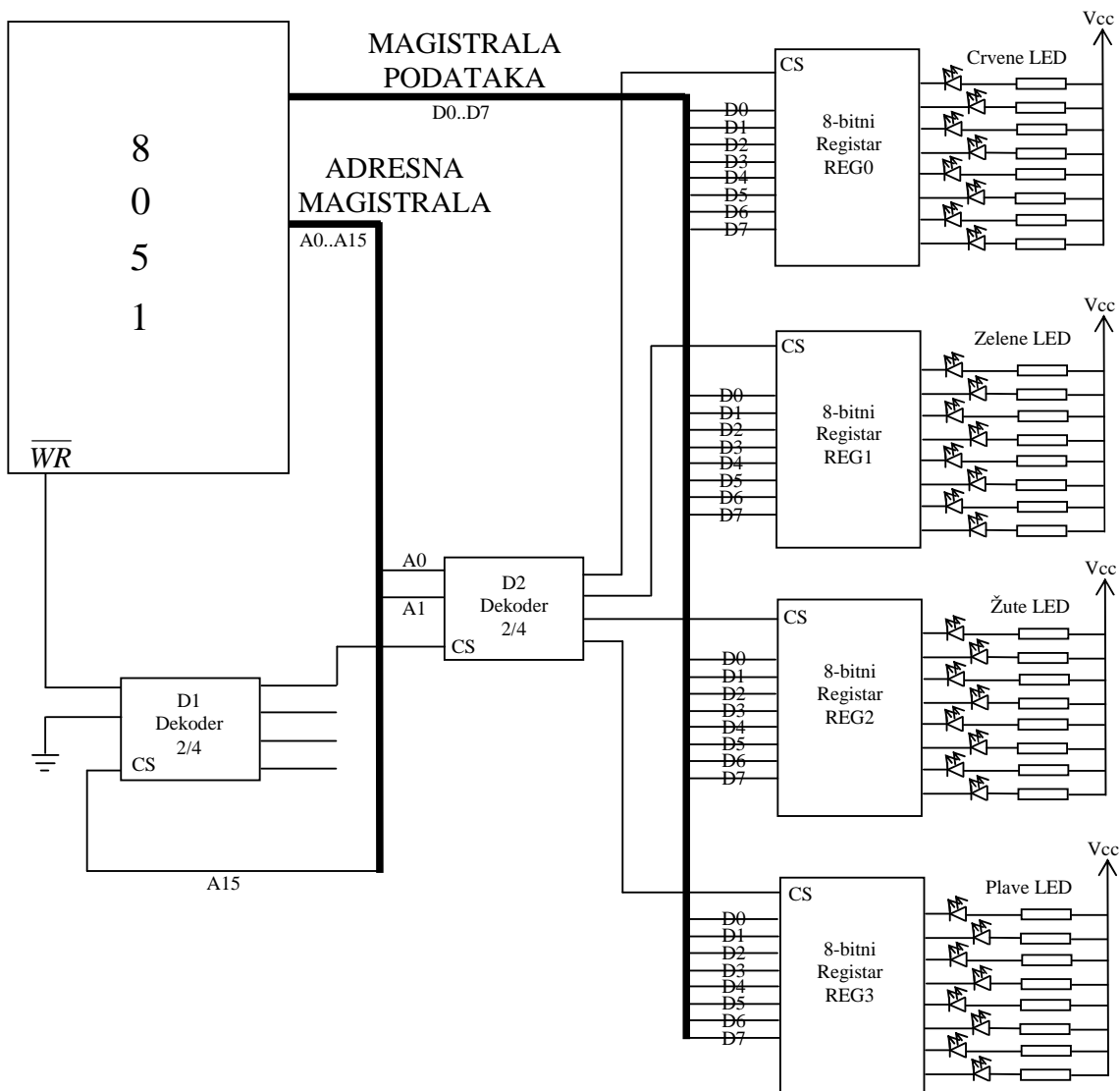
Ranije je objašnjeno da kućište mikrokontrolera ima 40 nožica i da je proizvođač u cilju obezbeđivanja veće funkcionalnosti čipa implementirao multipleksiranje određenih signala. Ako se koristi mikrokontroler sa internim ROM-om i nema spoljnog ROM-a ili RAM-a, portovi P0 i P2 mogu koristiti sve linije porta kao univerzalne ulaze ili izlaze. Međutim, ovi portovi imaju dodatnu ulogu ako se koristi spoljašnji ROM ili spoljašnji RAM. Tako se port P0 koristi kao izvor signala adresa A0-A7 i ulaz/izlaz podataka D0-D7. Pre svakog očitavanja programa iz spoljne memorije ili prozivanja RAM-a mikrokontroler na P0 prosleđuje niži bajt adresnog registra i aktivira izlaz ALE. Spoljni registar (najčešće LATCH registar tipa 373 ili 573 iz TTL 74xx familije) na visok nivo ALE memoriše stanje P0, a izlazi ovog registra se koriste kao A0-A7. U drugom delu mašinskog ciklusa mikrokontrolera P0 se koristi kao magistrala podataka (Data Bus).

Slično se ponaša i port P2, tj. ako se koristi spoljna memorija, onda su ovo visoki adresni izlazi, od A8 do A15. U tom slučaju, čak i ako se koriste samo neke adrese, preostale nožice ovog porta ne mogu da se koriste kao ulazi ili izlazi.

Dakle, mikrokontroler preko portova P0 i P2 (ukupno 16 adresa) može da adresira 64k memorije. Iako je memorija praktično periferni uređaj, postavlja se pitanje, šta je sa ostalim periferijama kao što su displeji, prekidači ili svetleće diode?

Kontrolni procesi koji se mogu implementirati 8051 mikrokontrolerom, najčešće ne zahtevaju pun opseg spoljašnje memorije, bilo da je u pitanju RAM, ROM ili kombinacija. To znači da veliki deo memorijskog prostora ostaje neiskorišćen. Ne samo da deo memorijskog prostora ostaje neiskorišćen, već su nožice portova koji nisu važne za formiranje adrese neupotrebljivi. Na primer, ako nam je potrebna eksterna memorija od 1k bajta, za adresiranje je potrebno 10 linija od ukupno 16 sa portova P0 i P2. Ostale nožice, iako ne učestvuju u formiranju adresa, ne mogu biti iskorišćene za vezivanje periferija, jer to ne dozvoljava mikrokontroler.

Ipak, moguće je vezati periferne uređaje ako ih posmatramo kao delove memorijskog adresnog prostora mikrokontrolera. Kombinacijom kola za pamćenje stanja i adresnog dekodera moguće je povezati veliki broj periferija. Na sledećem primeru ćemo videti kako povezati četiri niza od 8 dioda (npr. crvenih, zelenih, žutih i plavih LED).

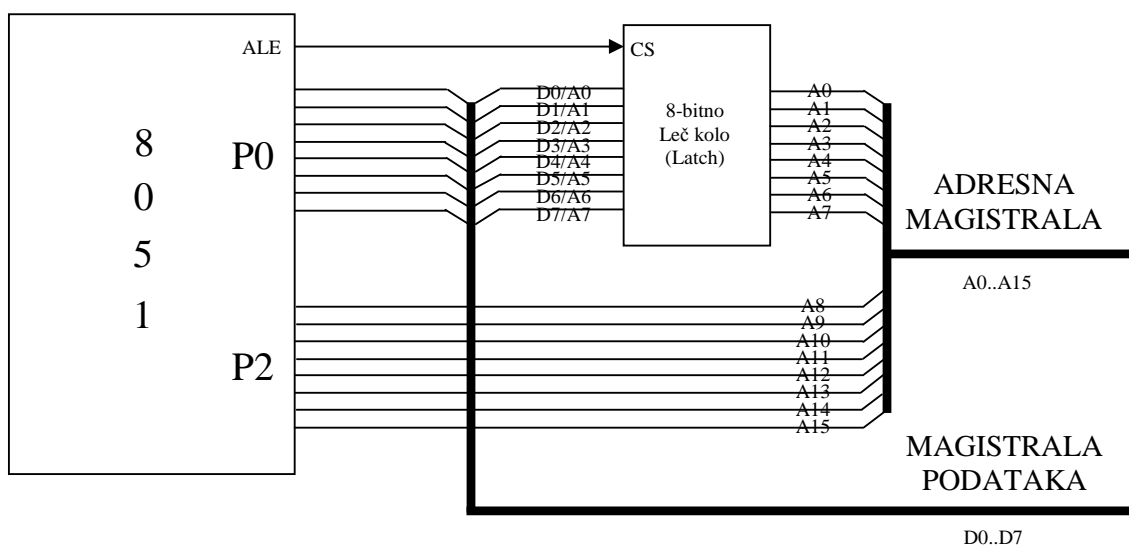


Slika 1: Memorijsko mapiranje 4 niza od 8 LED

Na slici (sl. 1) je dat uprošćen prikaz jednostavnog primera memorijskog mapiranja periferija. Za početak, nećemo ulaziti u detalje oko konkretnog formiranja adresne magistrale i magistrale podataka, jer ćemo akcenat u ovoj analizi staviti na samu ideju memorijskog mapiranja periferija. Primećuje se sa slike da su 4 niza od po 8 LED dioda povezane na izlaze registara REG0..REG3, umesto direktno na portove mikrokontrolera kao što je to bio slučaj kod „direktnog povezivanja periferija“. Sva četiri registra su svojim ulazima spojeni na magistralu podataka (ulaze D0..D7). Ovo ne znači da će se uvek istovremeno paliti i gasiti iste crvene, zelene, žute i plave diode. Koji niz od po 8 dioda će biti trenutno aktivan određuje adresni dekodera D2. Na njegove ulaze spojene su najniže adresne linije (A0 i A1) sa magistrale adresa. To znači da će u svakom trenutku samo jedan od izlaza dekodera biti aktiviran (postavljen na logičko "1"), a sve u skladu sa ulazima A0 i A1. Ako su A0 i A1 na nivou logičke nule, logička jedinica se pojavljuje na CS (Chip Select) ulazu registra koji upravlja crvenim diodama, ako su A1=0 i A0=1, CS=1 na ulazu registra koji upravlja zelenim LED diodama, itd. Onog momenta kada se pojavi logička jedinica na nekom od CS ulaza registara REG0..REG3, binarna vrednost koja se u tom trenutku nalazi na magistrali podataka (D0..D7), prosleđuje se na izlaz odgovarajućeg registra i na taj način se upravlja crvenim, zelenim, žutim ili plavim LED diodama. Međutim, postoji situacija u kojoj će svi izlazi dekodera D2 biti u stanju logičke nule! To se dešava ukoliko je na CS ulazu dekodera D2 nizak naponski nivo (logička nula). Dakle, adresni dekodera D2 je "isključen" kada je na njegovom CS ulazu logička "0", a to se opet dešava kada A15 adresna linija ima vrednost logičke "0", ili ako ima vrednost logičke "1", ali je vrednost signala $\overline{WR} = 1$. Iz

svega ovoga proizilazi da će se upravljati nekim od niza dioda, samo ako je $A_{15}=1$ i $\overline{WR}=0$. Zbog čega je ovo neophodno? Na ovaj način smo crvene diode mapirali na adresu 0x8000, zelene na adresu 0x8001, žute na 0x8002 i plave na 0x8003. Dakle, diode posmatramo kao da se radi o eksternoj RAM memoriji, pa kada mikrokontroler 8051 vrši instrukciju upisa, na primer, vrednosti 0x0F na adresu 0x8000, indirektno ćemo uključiti gornje, a isključiti donje 4 crvene LED diode (dioda se uključuje niskim naponskim nivoom). Ovo se dešava zbog toga što je $A_{15}=1$, $A_1=A_0=0$ (za adresu 0x8000 = 1000 0000 0000 0000 b), a signal \overline{WR} se automatski aktivira (postavlja na logičku nulu) kada se izvršava instrukcija UPISA u eksternu memoriju! Ovde vredi pomenuti da bi se povezivanjem \overline{RD} izlaza mikrokontrolera na drugi ulaz dekodera D1, moglo omogućiti dodatno priključenje periferija na isti adresni opseg (0x8000-0x8003) ali bi te periferije bile aktivirane samo kada bi se vršilo ČITANJE sa neke od adresa iz spomenutog opsega. Ovo je moguće zbog činjenice da se nikada istovremeno ne aktiviraju signali \overline{WR} i \overline{RD} , tako da se upisom na adresu, na primer, 0x8003 upravlja plavim LED diodama, dok bi se čitanjem sa adrese 0x8003 vršilo čitanje periferije (npr. prekidača koji su mapirani na tu adresu).

Još samo da zaokružimo priču o adresnoj magistrali i magistrali podataka. Dosta smo već spominjali P0 i P2, kao i njihovu ulogu u formiranju adresa prilikom pristupa spoljašnjem ROM-u ili spoljašnjem RAM-u. Spominjali smo takođe da P0 služi za generisanje donjih 8 bita adrese ($A_0..A_7$) tokom prve polovine mašinskog ciklusa u kome se izvršava pristup spoljašnjoj memoriji. Tokom druge polovine mašinskog ciklusa P0 služi za prijem podatka, u slučaju ČITANJA spoljašnje memorije, ili slanje podatka u slučaju UPISA u spoljašnju memoriju (u slučaju upisa možemo govoriti samo o upisu u spoljašnji RAM, naravno, a ne i o upisu u spoljašnji programski ROM). Kako je ovo ostvarivo i koje nam dodatne elektronske komponente trebaju za ovakvo multipleksiranje adresnih linija sa linijama podataka, vidi se na slici 2.

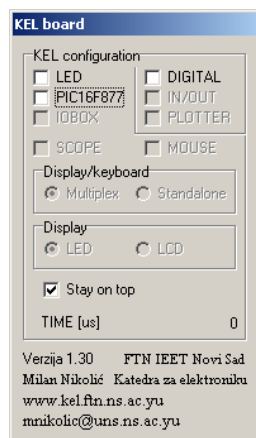


Slika 2: Formiranje adresne magistrale i magistrale podataka

Na slici 2 možemo videti kako se formiraju magistrala adresa i magistrala podataka. Leć kolo prikazano na slici je neophodno kako bi prihvatilo donjih osam bita adrese tokom druge polovine mašinskog ciklusa kada P0 služi za slanje ili prijem podataka. Signal ALE generiše se od strane mikrokontrolera da označi trenutak kada leć kolo treba da privati tih donjih osam bita adrese ($A_0..A_7$). U trenutku kada se pojavi opadajuća ivica (sa "1" na "0") na liniji ALE, leć kolo prosledjuje na svoj izlaz 8 bita koji su u tom trenutku prisutni na njegovom ulazu. Od tog trenutka pa do kraja mašinskog ciklusa, izlazi leć kolo zajedno sa linijama porta P2 sačinjavaju adresnu magistralu, dok port P0 sačinjava magistralu podataka.

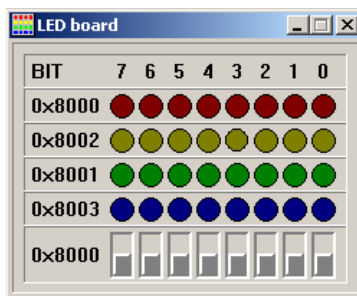
Zadaci

Slede zadaci u kojima će se vežbati memorijsko mapiranje periferija. Za simulaciju zadataka koji slede potrebno je u Keil simulatoru, nakon pokretanja Debug moda, uključiti opciju KEL maketa (Peripherals -> KEL maketa). Nakon toga će se pojaviti prozor prikazan na slici 3.



Slika 3: Izgled prozora za predstavljanje KEL makete u simulatoru

KEL maketa je maketa koja se koristi tokom izvođenja vežbi na višim godinama studija, a mi ćemo koristiti simulator KEL makete. Na prozoru koji se otvori nakon uključivanja opcije KEL maketa u meniju, treba odabrati polje LED unutar grupe KEL configuration (slika 3). Nakon toga se pojavljuje prozor prikazan na slici 4. Na slici se vide četiri niza od po 8 dioda (crvenih, žutih, zelenih i plavih). Na pravoj maketi ove diode su povezane sa mikrokontrolerom na način prikazan na slici 1 (praktično smo se upoznali sa idejom memorijskog mapiranja periferija na ovom primeru).



Slika 4: Prikaz prozora LED

Na slici se čak mogu videti i prekidači koji su takođe mapirani na adresu 0x8000. Spomenuli smo prilikom analize slike 1 da se mogu na istu adresu mapirati dve periferije ukoliko se u jednu od njih upisuje a iz druge čita. To je moguće zbog toga što se prilikom upisa koristi signal \overline{WR} mikrokontrolera, a prilikom čitanja signal \overline{RD} . Obratite pažnju da šema sa slike 1 to ne bi omogućila jer signal \overline{RD} nije povezan sa dekomerom D1. Na pravoj KEL maketi upravo to je urađeno kako bi se moglo čitati stanje prekidača sa adrese 0x8000.

Zadatak 1:

Ponoviti zadatak iz primera 1 (poglavlje 1: *Direktno povezivanje periferija*). Prekidači DIR i FREQ se nalaze direktno povezani na port mikrokontrolera kao u primeru 1, ali treba upravljati nizom žutih LED dioda (memorijski mapiranih na adresi 0x8002).

Zadatak 2:

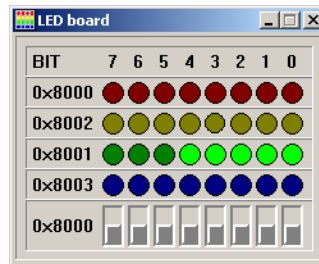
Na adresi 0x8001 se nalazi 8 memorijski mapiranih zelenih LED dioda, a na adresi 0x8001 8 žutih LED dioda. Na P1.0 se nalazi prekidač DIRG, a na P1.1 prekidač DIRY. Prekidač DIRG određuje smer rotiranja zelenih LED dioda, a prekidač DIRY smer rotiranja žutih LED dioda (kada je na portu logičko "1" rotiraju se u desno, a kada je "0" u levo). Dioda se rotiraju frekvencijom 10Hz.

Zadatak 3:

Ponoviti prethodni zadatak uz izmenu da su prekidači memorijski mapirani na adresu 0x8000.

Zadatak 4:

Elektronska kockica. Koristeći mikrokontroler Atmel 89C51 radne frekvencije 12MHz kod koga je na 0x8001 memorijski mapiran niz zelenih LED dioda i na port **P1.0** vezan jedan prekidač, napisati program koji realizuje elektronsku kockicu (brojanje od jedan do šest). Koristiti timer1, a frekvencija promene brojeva treba da bude 200Hz. Broj kockice (1 do 6) prikazan je tako što je uključen odgovarajući broj zelenih LED dioda. Prekidač priključen na port P2 služi da zaustavi kockicu (kada je njegova vrednost logička "0") odnosno da je ponovo pokrene (kada je njegova vrednost "1"). Na sledećoj slici je prikazan izgled LED dioda nakon dobijenog broja 5.



Zadatak 5:

Štoperica. Koristeći mikrokontroler Atmel 89C51 radne frekvencije 12MHz kod koga su na adresama 0x8000-0x8003 memorijski mapirani nizovi LED dioda (crvene, zelene, žute i plave) i na port **P3.0** vezan jedan prekidač, napisati program koji realizuje štopericu. Minuti su **binarno** predstavljeni crvenim (viša cifra 0..5) i žutim (niža cifra 0..9) diodama. Sekunde su predstavljene **binarno** zelenim i plavim diodama (zelenim viša cifra, a plavim niža cifra merenih sekundi). Prekidač služi da pokreće i zaustavlja štopericu ("0" – stani, "1" – kreni).