

Mikroprocesorski softver, januar 2014.

1. Inicijalizovati VIC periferiju tako da prekid sa WDT periferije bude klasifikovan kao FIQ, a prekid **PLL Lock** ima najviši IRQ prioritet. Rutina za opsluživanje PLL ima početnu adresu **PLL_zakljucan**. Svi ostali prekidi treba da budu zabranjeni.....(6)

```
VICIntSelect = 0x0000 0001; /* prekid br. 0 klasifikovan kao FIQ */
VICIntEnable = 0x0000 1001; /* dozvoljeni prekidi br.0 i 12 */
VICVectCntl0 = 0x2C;          /* slot 0 uključen i ubačen prekid br. 12 */
VICVectAddr0 = PLL_zakljucan;
```

Opisati čemu bi mogao da služi prekid od PLL periferije i šta bi prekidna rutina mogla da radi.....(4)

2. Za mikrokontroler čija učestanost perifernog takta PCLK iznosi 10 MHz, generisati tri simetrično postavljena PWM signala (impulsi na sredini periode) na izlazima PWM2,4 i 6, sa periodom širinske modulacije 0,1 ms i sa impulsima 20%, 40% i 80% periode, respektivno.....(10)

Pošto se u drugom delu zadatke zahteva da se širina impulsa menja svakih 5ms, jedno od mogućih rešenja je da se napravi tabela sa svim širinama impulsa iz koje bi se redom čitale vrednosti širine koju u tom vremenskom intervalu treba ostvariti. U ponuđenom rešenju je napravljena tabela trenutaka rastuće ivice u odnosu na početak periode. Trenuci opadajuće ivice su 60, 70, 90 i 70.

```
const unsigned int ivica [4] = {40, 30, 10, 30};
// trenuci rastuće ivice za impulse širina 20%, 40%, 80% i 40%
unsigned int indeks2=0, indeks4=1, indeks6=2;
const unsigned int perioda = 100;

PWMTCCR = 0xA; /* zaustavljen PWM brojač */
PWMPR = 9;      /* deljenje PCLK takta sa 10. Rezolucija 1us je sasvim dovoljna */
PWMPCR = 0x5454; /* PWM2,4i6 su u režimu kontrole 2 ivice, izlazi uključeni */
PWMMCR = 0x2; /* poklapanje 0 resetuje brojač */

PWMMR0 = perioda;
// početne vrednosti indeksa za PWM2, PWM4 i PWM6 su 0,1 i 2
PWMMR1 = ivica[indeks2];          /* rastuća ivica */
PWMMR2 = perioda - ivica[indeks2]; /* opadajuća ivica */
PWMMR3 = ivica[indeks4];
PWMMR4 = perioda - ivica[indeks4];
PWMMR5 = ivica[indeks6];
PWMMR6 = perioda - ivica[indeks6];

PWMTCCR = 0x9; /* pokretanje PWM brojača u PWM načinu rada */
```

Izlaz PWM2 posle svakih 5 ms promeniti na 40%, pa na 80%, zatim na 40%, pa opet na 20% i tako dalje, periodično. Izlazi PWM4 i PWM6 treba da se menjaju periodično na isti način, samo sa početnom vrednošću 40% (PWM4) i 80% (PWM6). Dakle, PWM6 treba da bude 80%, 40%, 20%, 40%, 80%, 40% ...

Pretpostaviti da funkcija `void pauza_5ms(void)` postoji. Zanemariti vreme izvršavanja ostalih instrukcija.(10)

```
while (1)
{
    pauza_5ms ();
    indeks2++; indeks2&=0x03; /* samo poslednja dva bita indeksa ... */
    indeks4++; indeks4&=0x03; /* ... tako indeks broji 0 do 3 pa se pri ... */
    indeks6++; indeks6&=0x03; /* ... sledećem uvećanju premotava na 0 */
    PWMMR1 = ivica[indeks2]; /* rastuća ivica */
    PWMMR2 = perioda - ivica[indeks2]; /* opadajuća ivica */
    PWMMR3 = ivica[indeks4];
    PWMMR4 = perioda - ivica[indeks4];
    PWMMR5 = ivica[indeks6];
    PWMMR6 = perioda - ivica[indeks6];
    PWMLER = 07E; /* dozvola za upis novih vrednosi u PWMMR registre */
}
```

U čemu se razlikuje načina rada PWM periferije ako je bit PWMTCCR.3 = 1 i ako je PWMTCCR.3 = 0? Kada treba koristiti jedan, a kada drugi način rada?.....(4)

3. Napisati deo C koda kojim bi se uradile četiri uzastopne AD konverzije sa kanala AD0.0 i srednja vrednost rezulta dodelila promenljivoj `AD_rezultat`. Konverzije treba da budu jedna za drugom, čim se jedna završi, počinje druga. Pretpostaviti da primena dozvoljava čekanje na kraj konverzije u petlji. PCLK je 10 MHz. Po potrebi, deklarirati neophodne lokalne promenljive.....(6)

```
unsigned int i, rezultat=0, AD_rezultat;

for (i=0; i<4; i++)
{
    AD0CR = 0x01200201;
    // AD0CR_bit.SEL=1; AD0CR_bit.CLKDIV=2; AD0CR_bit.PDN=1;
    // AD0CR_bit.START=0x01; /* softverski start konverzije odmah */
    while (!(AD0DR&0x80000000)); /* čekanje kraja konverzije */
    rezultat += (AD0DR&0x0000FFFF)>>6;
}
AD_rezultat = rezultat>>2; /* deljenje sa 4 */
```

Kada nastaje greška prepisivanja (*overflow*) kako se proverava, a kako izbegava.....(4)

4. U promenljivoj `Napon` nalazi se 8-bitni podatak o naponu. Podatak o naponu izbaciti na izlaz DA konvertora. Ako je `Napon=0`, izlaz iz DA konvertora treba da bude 0, ako je `Napon=128`, izlaz treba da bude $V_{ref}/2$ i najzad, ako je `Napon=255`, izlaz treba da bude približno V_{ref} , tačnije $(V_{ref}-V_{ref}/256)$. Primena nije kritična što se tiče potrošnje. V_{ref} je referentni napon DA konvertora(6)

```
DACR = Napon<<8; /* Pomeranje za dva mesta više da bi se 8-bitni podatak ... */
/* ... dopunio sa još dve nule sa lsb strane. BIAS=0 */
```