CILJ VEŽBE

- Upoznavanje sa radom i načinom korišćenja ulazno-izlaznim portom opšte namene (***G**eneral **P**urpose **I**nput **O**utput* - GPIO) mikrokontrolera ARM_LPC2148
- Upoznavanje sa periferijom tajmer/brojač (*timer*)
- Rešavanje praktičnih problema korišćenjem ulazno-izlaznog porta opšte namene.
- Samostalna izrada jednostavnih programa od ideje do testiranja.

GPIO periferija se sastoji od dva porta (PORT0 i PORT1) pri čemu svaki od njih sadrži do 32 ulazno-izlazne linije. Svakoj ulazno-izlaznoj liniji je pridružena po jedna nožica mikrokontrolera i po jedan bit u odgovarajućem registru podataka (IO0PIN i IO1PIN). Stanje svakog bita u IOPIN registrima je povezano sa logičkim stanjem odgovarajuće nožice mikrokontrolera. Na primer, logičko stanje bita 15 porta 0 (P0.15) odgovara logičkom stanju nožice 45 na mikrokontroleru.

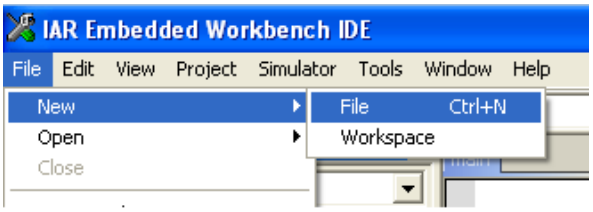
ZADATAK 1: Izraditi program koji bi po startovanju naizmenično palio i gasio neku od svetlećih dioda, tako da dioda svetli tačno sekundu i isto toliko je ugašena..

REŠENJE:

Korišćenjem uputstva sa standardnom uvodnom procedurom za rešavanje zadataka, sa početka praktikuma, pokrenuti i podesiti razvojno okruženje, a zatim uraditi specifična podešavanja za ovaj zadatak koja slede u nastavku.

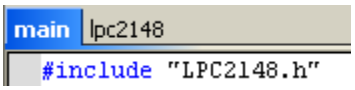
1. **Kreiranje datoteke zaglavlja (*header file*)**
   Obzirom da je C programski jezik višeg nivoa, u naredbama nije podržano direktno korišćenje adresa memorijskih lokacija, registara i periferija, već je potrebno za takve svrhe koristiti pokazivače. Kako bismo olakšali korišćenje registara periferija i učinili kôd preglednijim, pre početka razvoja kôda glavnog programa deklarisaćemo simbolička imena za periferije koje ćemo koristiti u vežbi. To ćemo uraditi u posebnoj datoteci zaglavlja (***header***) koju ćemo nazvati **LPC2148.h** i uključiti je (pomoću direktive #***include***) u glavni program.

   Kreirati novu datoteku **LPC2148.h** i snimiti je. Na slici je prikazano kako se projektu dodaju nove datoteke. Nakon kreiranja, datoteku treba snimiti odabirom **File→Save** pod imenom „**LPC2148.h**“ u radni direktorijum vežbe.



   Nakon snimanja, datoteku treba dodati u projekat odabirom opcije nad projektom: **Desni klik→Add→Add „LPC2148.h“**.

   U **main** programu dodati na početku direktivu:



2. **Deklarisanje simboličkih imena periferija**
   Za potrebe prvog zadatka od periferija je potrebno da koristimo samo jednu liniju jednog GPIO porta. Stoga ćemo deklarisati samo simbolička imena za pokazivače na registre posredstvom kojih se upravlja portom IOPORT0, čije adrese treba da pronađemo u user manualu za procesor LPC2148:

```
IOPIN0        0xE0028000
IOSET0        0xE0028004
IODIR0        0xE0028008
IOCLR0        0xE002800C
```

Deklarisanje simboličkog imena u C programskom jeziku se radi pomoću direktive #***define***, a tip pokazivača na memorijski lokaciju se, u zavisnosti od veličine podatka kojem se pristupa, definiše kao:
`(*((volatile unsigned char *)` – pokazivač na memorijsku lokaciju jednobajtnog podatka;
`(*((volatile unsigned short *)` – pokazivač na memorijsku lokaciju dvobajtnog podatka;
`(*((volatile unsigned long *)` – pokazivač na memorijsku lokaciju četvorobajtnog podatka.
Obzirom da su svi GPIO registri 32-bitni, koristićemo četvorobajtni tip pokazivača. Prema tome, deklaracija registara IOPORT0 izgleda kao Ilustracija 1:

main | lpc2148

```
#define IOPIN0        (*((volatile unsigned long *) 0xE0028000))
#define IOSET0        (*((volatile unsigned long *) 0xE0028004))
#define IODIR0        (*((volatile unsigned long *) 0xE0028008))
#define IOCLR0        (*((volatile unsigned long *) 0xE002800C))
```

Ilustracija 1 Deklarisanje IOPORT0 registara

U ovoj vežbi nam je takođe potrebno da koristimo TIMER periferiju, pa ćemo deklarisati simbolička imena za pokazivače na registre posredstvom kojih se njom upravlja. Njihove adrese možemo pronaći u *user manual*-u za procesor LPC2148:

```
T0TCR         0xE0004004
T0TC          0xE0004008
```

3. **Deklaracija tipova**
   Na sličan način (kao za tipove pokazivača na memorijske lokacije u prethodnom odeljku) deklarisaćemo u datoteci i takva skraćena simbolička imena za osnovne C tipove, koja će u nazivu govoriti koliko bita nosi podatak tog tipa, kao i da li predstavlja označeni ili neoznačeni broj:

```
/* Skracena notacija za tipove podataka */
#define C8      char
#define U8      unsigned char
#define U16     unsigned short
#define U32     unsigned long
#define S8      signed char
#define S16     signed short
#define S32     signed long
```

4. **Deklaracija konstanti za vreme**
   Deklarisaćemo konstante koje predstavljaju vrednost tajmerskog brojača za 1 sekundu i za 10 milisekundi:

```
#define SEK (U32)(14745600/4)
#define MILISEK10 (U32)(SEK/100)
```

5. **Inicijalizacija GPIO periferije**
   Na početku glavnog programa deklarisati funkciju

```
void Init_GPIO(void);
```

   Na kraju glavnog programa definisati funkciju Init_GPIO, u kojoj jedino što treba inicijalizovati je da jedna linija porta bude izlazna, na primer P0.7, setovanjem odgovarajućeg pina u IODIR0 registru (*default* smer za sve linije GPIO je ulazni):

```
void Init_GPIO(void)
{
  IODIR0 |=1<<7;
}
```

6. **Pomoćne funkcije Timer periferije**
   Na početku glavnog programa deklarisati funkciju za resetovanje brojača Timer0 periferije:
   ```
   void Start_TMR0(void);
   ```

   Na kraju glavnog programa definisati funkciju Start_TMR0, koja resetuje brojač Timer0 periferije:
   ```
   void Start_TMR0(void)
   {
      T0TCR=3;
      T0TCR=1;
   }
   ```

7. **Glavna programska funkcija**

   Na početku glavne programske funkcije najpre ćemo inicijalizovati GPIO periferiju pozivom funkcije `Init_GPIO()` i inicijalno startujemo tajmer pozivom funkcije `StartTMR0()`:
```
void main( void )
{
   Init_GPIO();
   Start_TMR0();
   while(1)
   {
        while(T0TC<SEK);
        Start_TMR0();
        IOPIN0 ^=1<<7;
   }
}
```

   Nakon inicijalizacije izvršava se beskonačna petlja u kojoj se naizmenično upisuje logička 1 i logička 0 na mesto bita 7 registra IO0PIN (IO0PIN.7), što uzrokuje odgovarajuće promene naponskih nivoa nožice pridružene ovom bitu GPIO porta. Između ovih promena stanja pravi se pauza od približno 1s čekanjem da tajmerski brojač odbroji do 14745600/4.

Na simulatoru povezati nožicu odabrane izlazne linije GPIO-a mikrokontrolera sa nekom od svetlećih dioda. Podesiti u simlatoru putanju do kreirane .elf datoteke i pokrenuti njegovo izvršavanje. Posmatrati promene stanja u registrima periferija.

ZADATAK 2: Izraditi program koji bi na pritisak tastera pokretao treperenje neke od svetlećih dioda.

PROCEDURA:

- Slično kao u prvom zadatku, podesiti registre mikrokontrolera tako da se jedna od linija ulazno-izlaznog porta GPIO (npr. P0.6) koristi kao izlazna, a neka druga kao ulazna (npr P0.8).

- Projekat kreiran u okruženju ***IAR Embedded Workbench*** za potrebe prvog zadatka, izmeniti tako da se paljenje i gašenje svetlećih dioda pokrene tek nakon što se sa odabrane ulazne linije ulazno-izlaznog porta (GPIO) pročita stanje pritisnutog tastera. Prevesti program.
- Nožicu odabrane ulazne linije GPIO-a mikrokontrolera u simulatoru povezati sa nekim od tastera.
- Nožicu odabrane izlazne linije GPIO-a mikrokontrolera u simulatoru povezati sa nekom od svetlećih dioda.
- Podesiti u simlatoru putanju do kreirane .elf datoteke i pokrenuti njegovo izvršavanje.

ZADATAK 3: Izraditi program koji bi samo dok je pritisnut taster pokretao „trčeću“ svetleću diodu – palio i gasio nekoliko svetlećih dioda jednu za drugom, ali tako da program bude protočan, tj. da se ne „zaglavljuje“ čekajući da tajmer odbroji predviđeno vreme.

PROCEDURA:

- Slično kao u drugom zadatku, podesiti registre mikrokontrolera tako da se nekoliko linija ulazno-izlaznog porta GPIO (npr. P0.4, P0.5, P0.6 i P0.7) koriste kao izlazne, a jedna kao ulazna (npr P0.8).
- Projekat kreiran u okruženju ***IAR Embedded Workbench*** za potrebe drugog zadatka, izmeniti tako da se paljenje i gašenje svetlećih dioda pokrene tek nakon što se sa odabrane ulazne linije ulazno-izlaznog porta (GPIO) pročita stanje pritisnutog tastera. Prevesti program.
- Nožice odabrane ulazne linije GPIO-a mikrokontrolera u simulatoru povezati sa nekim od tastera.
- Nožice odabranih izlaznih linija GPIO-a mikrokontrolera u simulatoru povezati sa svetlećim diodama.
- Podesiti u simlatoru putanju do kreirane .elf datoteke i pokrenuti njegovo izvršavanje.