VEŽBA 3

Mikrokontroleri su sistemi sa izrazito ograničenim resursima. Zbog toga je postojanje mehanizama za precizno manipulisanje memorijskim resursima od ključnog značaja. Među ove mehanizme spadaju strukture i unije.

Strukture i unije

Strukture

Definicija strukture se postiže na sledeći način:

```
struct ime_strukture {
  tip1 ime_polja1;    // navedena polja su članovi strukture
  tip2 ime_polja2;
  tip3 ime_polja3[];  // polje moze biti i niz
};
```

Ovim se uvodi novi tip podatka `struct ime_strukture`. Ključnu reč `struct` je neophodno navoditi jer prevodilac bez toga ne prepoznaje ime tipa kao strukturu.

Promenljiva tipa struktura definiše se na sledeći način:

```
struct ime_strukture ime_promenljive;
```

Karakteristika strukture je da se pod zajedničkim imenom, zbijeno u blok **susednih** memorijskih polja skladište **svi** elementi navedeni kao članovi strukture. Pristupa im se operatorom `.` (tačka) koja se navodi iza imena promenljive:

```
stp.cx = 100;  // pristup članu (polju cx) strukture stp
```

Ukoliko se pristupa preko pokazivača, koristi se operator `->`:

```
struct test* pstp = &stp;
...
rezultat = pstp->bx * 3;
```

Unije

Unija se formalno definiše na potpuno isti način kao i struktura. Suštinska razlika je što polja nisu susedna nego se skladište sva na potpuno istom prostoru u memoriji (jedno preko drugog). Operator pristupa (`.`) praktično definiše interpretaciju podataka, a ne selekciju izabranog člana grupe.

```
union abc {
  unsigned long num;
  unsigned char b[10];
};

union abc bafer;
...
bafer.num = 12345;
printf("Najnizi bajt broja: %u\n", (unsigned)bafer.b[0]);
```

Podaci kojima se pristupa u primeru se interpretiraju različito, ali se fizički nalaze na istom mestu u memoriji. Pristup prvom elementu niza karaktera efektivno će pristupiti grupi od osam najnižih bita 32-bitnog neoznačenog broja `num`.

Veličina objekta u memoriji

Svaki objekat u memoriji – promenljiva, pokazivač, niz, struktura, unija, … – zauzima određeni broj memorijskih ćelija. Kod savremenih računara i mikrokontrolera ustanovilo se da osnovna memorijska ćelija sadrži 8 binarnih cifara – bita – i naziva se **bajt** (*engl. byte*).

Ugrađeni tip jezika C koji ima dužinu od jednog bajta je `char` ilil `unsigned char`. Ugrađeni operator koji za rezultat ima dužinu proizvoljnog objekta merenu u dužinama tipa `char` (ili bajtovima) je operator `sizeof`. Npr.

```
unsigned d1, d2;
d1 = sizeof(int);  // operand moze biti ime tipa,
d2 = sizeof(d1);   // a moze biti i promenljiva
```

Zadaci

1. Definisati u programu uniju `abc` koja je data u jednom od prethodnih primera. Napisati program koji ispisuje dužine pojedinih elemenata unije i ukupnu dužinu unije. Šta se može zaključiti?
2. Uneti preko terminala hex broj (do 8 cifara), a potom ga štampati na terminalu bajt po bajt. Otkriti kojim redosledom su u memoriji raspoređeni biti (bajtovi) `unsigned long` podatka. Koji deo je na najnižoj adresi, bajt najveće ili najmanje težine?
3. Znajući redosled bajtova na osnovu prethodnog zadatka, tražiti unos 32-bitnog hex broja i rasporediti njegove delove (bajtove) u niz `b` tako da u njemu bude sadržan ispravan 32-bitni broj. Proveriti ispisom hex broja direktno putem ispisa elementa unije `num`.

Dodatni neobavezni zadatak:

4. Koristeći simulator LED svetala, napraviti sistem od 4 ulazna stupca (proizvoljne boje). Sa periodom od 500 ms očitavati ta četiri stupca i ispisivati 32-bitni broj koji čine u heksadecimalnom obliku, pri čemu je krajnji levi stubac čini najznačajnije bite, a krajnje desni najmanje značajne bite. Stubac prirodno ima najznačajniji bit na vrhu.

Pomoć za rešavanje zadataka

Ispis i unos heksadecimalnih brojeva

Kontrolni karakter `%x` ili `%X` omogućava ispis ili unos heksadecimalnog broja. Prva varijanta će pri ispisu koristiti mala slova (od a do f), dok će druga koristiti velika (od A do F). Pri unosu se ne pravi razlika između ovih varijanti.

Konverzija dugačkog broja u bajtove

Pri deljenju celog broja sa 256 dobija se vrednost najnižeg bajta kao ostatak, a količnik sadrži sve više bajtove broja (koliko god da ih ima). Ponavljanjem ovog postupka sve dok se ne dobije količnik jednak nuli dobijaju se ostali bajtovi po rastućem redosledu težine.

```
// razbijanje dugackog (32-bitnog) broja na bajtove
unsigned i = 0;   // brojac bajtova
unsigned long x;  // broj koji se konvertuje

char b[4];  // niz u kome se cuvaju bajtovi broja

while(i<4)
{
  b[i] = x % 256;
  x /= 256;
  i++;
}
```

Zaokruživanje dužine objekata

Mnogi procesori efikasnije pristupaju podacima ako se oni nalaze na adresama koje su deljive sa 4 (ponekad i 2, ili 8) pa se vrši podrazumevano dopunjavanje veličine objekta (*poravnanje, engl. alignment*) na tu dužinu deljivu sa 4. Unije, strukture i nizovi su naročito izloženi ovom efektu.