Uvod u programiranje

VJEŽBA 01 – bez rješenja
Grupa 01, Šimić, 2005.

Cjeline za vježbanje

- Cijeli brojevi
  - Prikaz pozitivnih cijelih brojeva
  - Raspon cijelih brojeva
  - Prikaz negativnih cijelih brojeva
  - Raspon brojeva različitih tipova podataka
- Realni brojevi
  - Prikaz komponenti realnih brojeva
  - Prikaz vrijednosti realnih brojeva
  - Preciznost realnog broja
  - Raspon realnih brojeva
- Brojevni sustavi (hexa, octal itd.)

Prikaz pozitivnih cijelih brojeva

- Odrediti dekadsku vrijednost u registru (memoriji) za broj bez korištenja i uz korištenje dvojnog komplementa
  - Recimo registar od 3 bita i broj 2+2:

    $2_{10} = 010_2$ -> $2_{10} + 2_{10} = 100_2$ -> $\mathbf{4_{10}}$

    $2_{10} = 010_2$ -> $2_{10} + 2_{10} = 100_2$ (dvojni komplement) ->
    komplement = 011_2 -> + 1 = 100_2 -> $\mathbf{-4_{10}}$
  - Za vježbu napraviti npr. $3_{10}+3_{10}$ (rješenje 6 $_{10}$ i -2 $_{10}$).
- Odrediti dekadsku vrijednost cijelog broja uz korištenje tehnike dvojnog komplementa
  - Primjer 01011_2 i 11011_2

    $01011_2 = 11_{10}$

    11011_2 = (dvojni komplement) ->
    -> komplement = 00100_2 -> + 1 = 00101_2 -> $\mathbf{-5_{10}}$

Prikaz pozitivnih cijelih brojeva

- Nakon zadane računske operacije odrediti sadržaj registra
  - Primjer 3+3+1 u 3 bitnom registru

    3+3+1= 7_{10} = 111_2
    - Ili isti broj u 2 bitnom registru

      $11_2 = 3_{10}$
- Odrediti maksimalni cijeli broj koji se može prikazati u registru s dvostrukim komplementom i bez njega
  - Primjer za 2 bitni i 4 bitni registar

    $\mathbf{2^n-1}$ => 2^2-1 = **3**, 2^4-1 = **15** $\mathbf{2^{n-1}-1}$ => $2^{2-1}-1$= **1**, $2^{4-1}-1$= **7**

    -> 2 puta više bita -> __ puta veći broj = $2^{\text{\_\_.puta.više.bita}}+1$
- Odrediti broj različitih vrijednosti koje se može prikazati
  - Prethodni primjer

    za 2 bita __ vrijednosti i za 4 bita __ vrijednosti = 2^{---}

Prikaz pozitivnih cijelih brojeva

- Kako izgleda zbroj dva binarna broja
  - 1101_2 + 1011_2 registru
    11000_2 $(24=13+11)_{10}$
- Koji broj je prikazan u prethodnom primjeru ukoliko se primjeni tehnika dvojnog komplementa?
  11000_2 -> komplement = 00111_2 (dvojni komplem.) +1 -> 01000_2 = **-8_{10}**
- Pretvoriti dekadski broj u binarni
  - Primjer 199_{10}
    11000111_2
- Množenje razlomljenog binarnog broja
  - Primjer množenje s 2_{10}
    binarni zarez (točka) se pomiče __ x u desno

Raspon cijelih brojeva

- Različiti pozitivni brojevi u binarnom prikazu
  - Recimo registar od 3 bita:
    000, 001, 011, 010 -> __ različita broja
    općenito vrijedi **2^{---} - 1**
- Različiti negativni brojevi u binarnom prikazu
  - Recimo registar od 3 bita:
    100, 101, 111, 110 -> __ različita broja
    općenito vrijedi **2^{---}**
- Raspon brojeva u binarnom prikazu
  - Recimo registar od 5 bita:

bez predznaka	s predznakom u tehnici dvojnog komplementa
[0 , 31]	[-16, 15]

Raspon cijelih brojeva

- Raspon brojeva u binarnom prikazu
  - char (oktet, byte):

s predznakom u tehnici dvojnog komplementa	unsigned (bez predznaka)
[-128, 127]	[0 , 255] [0 , 2^{--}-1]

  - short int (16 bita):

unsigned (bez predznaka)	s predznakom u tehnici dvojnog komplementa
[0 , 2^{--}-1] [0 , 65535]	[-2^{-1}, 2^{-1}-1]

Prikaz negativnih brojeva

- Potreban broj bita za prikaz nekog negativnog broja
  - Recimo broj -42_{10}:
    42_{10} = 101010_2 -> komplement = 010101_2
    (dvojni komplement) +1 -> 010110_2
    Potrebno je minimalno _ bitova.
- Zapis negativnog broja tehnikom dvojnog komplementa
  - Recimo broj -3 u 16 bitnom registru:
    3_{10} = $0000\ 0000\ 0000\ 0011_2$ -> komplement = $1111\ 1111\ 1111\ 1100_2$
    (dvojni komplement) +1 -> **$1111\ 1111\ 1111\ 1101_2$**

Prikaz negativnih brojeva

- Prikazati dekadski broj heksadecimalno primjenom tehnike dvojnog komplementa
  - Recimo **-21_{10}** u 8 bitnom registru :

    21_{10} = 0001 0101_2 -> komplement = 1110 1010_2

    (dvojni komplement) +1 -> **1110 1011_2 = EB_{16}**
  - Recimo **-121_{10}** u 9 bitnom registru :

    121_{10} = 0 0111 1001_2 -> komplement = 1 1000 0110_2

    (dvojni komplement) +1 -> **1 1000 0111_2 = 187_{16}**
- Sadržaj registra opisan hexadecimalno u dekadski
  - Recimo da je u 4 bitnom registru zapisano **B_{16}**

  B_{16} = **1011_2** komplement = 0100_2 (za dvojni kompl.) + 1 = 0101_2 = **-5_{10}**

Vrijednost u varijabli nakon pridruživanja

char	vrijednost$_{10}$	(bin. ili hex.)
`char c = -8;`	-8	1111 1000
`char c = 121;` `c = c + 7;`	-128	1000 0000
`char c = 125;` `c = c + 9;`	-122	1000 0110
`char c = -125;` `c = c - 6;`	125	0111 1101

- short

`short i =32767;` `i = i + 1;`	-32768	

Prikaz realnih brojeva

- Raspon binarnog eksponenta (BE)
  - [-126, 127] za standardnu preciznost
    - K=BE+127 (__ bita za karakteristiku)
    - K=255 posebno značenje (M!=0 -> ___, M=0 -> ____)
    - K=0 posebno značenje (M!=0 -> __, M=0 -> __________ broj)
  - [-1022, 1023] za dvostruku preciznost
    - K=BE+1023 (__ bita za karakteristiku)
    - K=2047 posebno značenje (M!=0 -> ___, M=0 -> ___)
- Duljina mantise – određuje preciznost
  - __ bita za standardnu preciznost (+ skriveni bit) - prec. __ dek. znam.
  - __ bita za dvostruku preciznost (+ skriveni bit) - prec. __ dek. znam.
- Dodavanje jako malog broja jako velikom broju
  - Na primjer 1,0e33 + 1,0e-33

    Ne mijenja veliki broj jer ___________________________!

Prikaz realnih brojeva

- Rezultat množenja binarnog broja
  - 101101 s 2^{-5}

    1.01101
  - 101.101 s 2^{2}

    10110.1
- Karakteristika za realni broj u standardnoj preciznosti
  - Realni broj 33,257

    Dovoljno je gledati samo cijelobrojni dio: 33_d = 100001_b

    $\Rightarrow$ treba množiti s 2^{--} da se dobije = 1.00001 => BE=5

    $\Rightarrow$ K=5+127=132 $_d$ => 1000100_b

Broj bita mantise i dekadskih znamenaka

- Broj potrebnih bita **nb** za dekadski broj s **nd** znamenaka
  - $2^{nb} = 10^{nd}$ -> nb = nd/log2 = nd/0,3
  - Recimo za dekadski broj s 6 znamenaka:
    nb = 5/0,3 = 16,7 = 17 bita -> 16 bita mantisa (uz skriveni bit)
- Broj dekadskih znamenaka **nd** uz zadani broj bita **nb**
  - $2^{nb} = 10^{nd}$ -> nd = nb*log2 = nb*0,3
  - Recimo za mantisu sa 23 bita:
    nd = 24*0,3 = 7,2 = 7 decimalnih znamenaka
- Imati na umu raspodjelu bita na **P + K + M**
  - Za preciznost imati na umu i skriveni bit **nb = M +1**

Određivanje mantise za realni broj

- Prikaz u binarnom i heksadecimalnom obliku
- Primjer -118.625_{10}

Binarno: $1110110.101_2 = 1.110110101 \cdot 2^6$

Binarno u standardnoj preciznosti:

BE=2 -> K=6+127=133

```
1      8              23          broj bita
+-+--------+-----------------------+
|S| Karakt.| Mantisa               |
|1|10000101|11011010100000000000000|
+-+--------+-----------------------+
31 30      23 22     redni broj bita (0 je desno)    0
   BE +127
```

1100 0010 1110 1101 0100 0000 0000 0000

$C2ED4000_{16}$

- Primjer 5.25_{10}

Binarno: $101.01_2 = 1.0101 \cdot 2^2$

Binarno u standardnoj preciznosti:

BE=2 -> K=2+127=129

0 100 0000 1 **010 1**000 0000 0000 0000 0000

$40A80000_{16}$

- Za vježbu:

Prikazati -9.125 u binarnoj standardnoj preciznosti te hexadecimalno i oktalno