Operacije nad bitovima

Slide 1

Operatori nad bitovima:

- Operatori nad bitovima mogu se primijeniti na cjelobrojne tipove podataka `char, short, int` i `long`.
- Ovi operatori djeluju na pojedine bitove unutar varijable.

Operator	Značenje
`&`	logičko I bit-po-bit
`|`	logičko ILI bit-po-bit
`^`	ekskluzivno logičko ILI bit-po-bit
`<<`	lijevi pomak
`>>`	desni pomak
~	1-komplement

Slide 2

- Operatori `&`, `|` i `^` uzimaju dva operanda i vrše operacije na bitovima koji se nalaze na odgovarajućim mjestima.

Definicije operacija:

`b1`	`b2`	`b1 & b2`	`b1 ^ b2`	`b1 | b2`
1	1	1	0	1
1	0	0	1	1
0	1	0	1	1
0	0	0	0	0

Primjeri:

Slide 3

Logičko I:

`a`	=	0100 0111 0101 0111
`b`	=	1101 0100 1010 1001
`a & b`	=	0100 0100 0000 0001

Logičko ILI:

`a`	=	0100 0111 0101 0111
`b`	=	1101 0100 1010 1001
`a | b`	=	1101 0111 1111 1111

Ekskluzivno logičko ILI:

`a`	=	0100 0111 0101 0111
`b`	=	1101 0100 1010 1001
`a ^ b`	=	1001 0011 1111 1110

Slide 4

Maskiranje (1):

Logički operatori najčešće služe *maskiranju* pojedinih bitova u operandu (transformaciji binarnog zapisa).

Primjer: treba šest najmanje značajnih bitova iz varijable `a` kopirati u varijablu `b`, a sve ostale bitove varijable `b` staviti na nulu.

```
mask = 0000 0000 0011 1111 (=0x3f)
```

`a`	=	0100 0111 0101 0111
`mask`	=	0000 0000 0011 1111
`a & mask`	=	0000 0000 0001 0111

odnosno

```
b=a & 0x3f;
```

Slide 5

Maskiranje (2):

Logičko I može poslužiti postavljanjem na nulu određenih bitova. Ako u varijabli `a` želimo postaviti na nulu neki bit, dovoljno je napraviti logičko I s konstantom koja na traženom mjestu ima nulu, a na svim ostalim jedinice.

Primjer: deseti bit postavljamo na nulu

```
mask = 1111 1101 1111 1111 (=0xfdff)
```

`a`	=	0100 0111 0101 0111
`mask`	=	1111 1101 1111 1111
`a & mask`	=	0100 0101 0101 0111

odnosno

```
b=a & 0xfdff;
```

Slide 6

Programiranje (C) Ak. god. 2003/04

Maskiranje (3):

Logičko ILI;

Primjer: treba šest najmanje značajnih bitova iz varijable `a` kopirati u varijablu `b`, a sve ostale bitove varijable `b` postaviti na jedan.

```
mask = 1111 1111 1100 0000 (=0xffc0)
```

`a`	=	0100 0111 0101 0111
`mask`	=	1111 1111 1100 0000
`a	mask`	=	1111 1111 1101 0111

odnosno

```
b=a | 0xffc0;
```

- Ova operacija ovisi o duljini tipa `int` (odn. onog tipa koji se koristi) no to se može izbjeći pomoću 1-komplementa.

Slide 7

Unarni operator 1-komplement (∼) :

djeluje tako da jedinice pretvara u nule u nule u jedinice:

```
~0101 1110 0001 0101 = 1010 0001 1110 1010
```

odnosno,

```
~0x5e15=0xa1ea
```

Umjesto

```
b=a | 0xffc0;
```

možemo pisati

```
b=a | ~0x3f;
```

(neovisno o duljini tipa s koji se radi).

Slide 8

M. Jurak pred12.tex

Maskiranje (4):

Ekskluzivno ILI možemo koristiti za postavljanje određenih bitova na 1 ako su bili 0 i obratno.

Primjer: peti i šesti bit treba invertirati

```
mask =   0000 0000 0011 0000 (=0x30)
```

`a`	=	0100 0111 0101 0111
`mask`	=	0000 0000 0011 0000
`a ^ mask`	=	0100 0111 0110 0111

odnosno

```
b=a ^ 0x30;
```

- Ako ponovimo operaciju dolazimo do polazne vrijednosti.

Slide 9

Operatori pomaka:

- Operatori pomaka `<<` i `>>` uzimaju dva operanda: prvi operand mora biti cjelobrojni tip nad kojim se operacija vrši, a drugi broj bitova za koji treba izvršiti pomak (`unsigned int`).
- Drugi operand ne smije premašiti broj bitova u prvom operandu.
- `<<` pomiće bitove ulijevo. Pri tome se najznačajniji bitovi gube, a sa desne strane mjesta se popunjavaju nulama.

Primjer: `b=a<<6` ima sljedeći efekt:

```
     a = 0110 0000 1010 1100
a << 6 = 0010 1011 0000 0000
```

Slide 10

- `>>` pomiće bitove nadesno. Pri tome se bitovi na desnoj strani (najmanje značajni) gube, dok se na lijevoj uvode novi.
- Ako se pomak vrši na varijabli tipa `unsigned`, onda se na lijevoj strani uvode nule.

Primjer:

```
     a = 0110 0000 1010 1100
a >> 6 = 0000 0001 1000 0010
```

- Ako je varjabla `a` cjelobrojni tip s predznakom onda rezultat može ovisiti o implementaciji. Većina prevodioca uvest će na lijevoj strani bit predznaka broja. S druge strane, neki prevodioci će ispražnjena mjesta uvijek popunjavati nulama.

Slide 11

Operatori pridruživanja:

Logički operatori formiraju operatore pridruživanaja

```
&=   ^=   |=   <<=   >>=
```

Primjer: Neka je `a=0x6db7`. Tada je

izraz	ekvivalentan izraz	vrijednost
`a &= 0x7f`	`a = a & 0x7f`	0x37
`a ^= 0x7f`	`a = a ^ 0x7f`	0x6dc8
`a |= 0x7f`	`a = a | 0x7f`	0x6dff
`a <<= 5`	`a = a << 5`	0xb6e0
`a >>= 5`	`a = a >> 5`	0x36d

Slide 12

Programiranje (C) Ak. god. 2003/04

M. Jurak pred12.tex

```
#include <stdio.h>
/* Ispis binarnog zapisa cijelog broja */
int main(void) {
    int a,b,i,nbits;
    unsigned mask;
    nbits=8*sizeof(int);  /* duljina tipa int */
    mask=0x1 << (nbits-1); /* 1 na najznacajnijem mjestu*/
    printf("\nUnesite cijeli broj: "); scanf("%d",&a);
    for(i=1;i<=nbits;++i) {
       b=(a & mask) ? 1 : 0;
       printf("%d",b);
       if(i % 4 == 0) printf(" ");
       mask >>= 1;
    }
    printf("\n");
}
```

Slide 13

Polja bitova

- Polja bitova nam omogućuju rad s pojedinim bitovima unutar jedne računalne riječi.
- Deklaracija polja bitova posve je slična deklaraciji strukture:

```
struct ime {
           clan_1;
           clan_2;
           .....
           clan_n;
          };
```

- Svaki član polja bitova predstavlja jedno polje bitova unutar računalne riječi.
- Sintaksa je takva da iz imena varijable dolazi dvotočka i broj bitova koji član zauzima.

Slide 14

Programiranje (C) Ak. god. 2003/04

Primjer:

```
#define STATIC 01
struct primjer {
           unsigned a : 1;
           unsigned b : 3;
           unsigned c : 2;
           unsigned d : 1;
          };

struct primjer v;
........
if(v.a == 1) ...
v.c=STATIC;
```

Slide 15

- Deklaracija definira strukturu razbijenu u četiri polja bitova, `a, b, c i d`, duljinu 1, 3, 2 i 1 bit.
- Poredak tih bitova unutar jedne kompjutorske riječi ovisi o implementaciji.
- Pojedine članove polja bitova možemo dohvatiti istom sintaksom kao i kod struktura, dakle `v.a`, `v.b` itd.
- Ako broj bitova deklariran u polju bitova nadmašuje jednu kompjutorsku riječ, za pamćenje polja bit će upotrebljeno više kompjutorskih riječi.

Slide 16

M. Jurak pred12.tex

Primjer:

```
#include <stdio.h>

int main(void)
{
    static struct{
        unsigned a : 5;
        unsigned b : 5;
        unsigned c : 5;
        unsigned d : 5;
        } v={1,2,3,4};

   printf("v.a=%d, v.b=%d, v.c=%d, v.d=%d\n",v.a,v.b,v.c,v.d);
   printf("v treba %d bajtova\n", sizeof(v));
   return 0;
}
```

Slide 17

- Poredak polja unutar riječi može se kotrolirati pomoću neimenovanih članova unutar polja.

```
struct  {
           unsigned a : 5;
           unsigned b : 5;
           unsigned   : 5;
           unsigned c : 5;
          };

struct primjer v;
```

Slide 18

- Neimenovani član čija je širina deklarirana kao 0 bitova tjera prevodilac da sljedeće polje smjesti u sljedeću računalnu riječ.

```
#include <stdio.h>

int main(void) {
    static struct{
        unsigned a : 5;
        unsigned b : 5;
        unsigned   : 0;
        unsigned c : 5;
        } v={1,2,3};

   printf("v.a=%d, v.b=%d, v.c=%d\n",v.a,v.b,v.c);
   printf("v treba %d bajtova\n", sizeof(v));
   return 0;
}
```

Slide 19