Operations on Bits

Discussion

- How does computer hardware work?
- What are bits?

Why Bits?

- Programming languages are byte oriented whereas hardware is bit oriented
- Bits important where program interacts directly with the hardware
- Gives better control

Bitwise operators

- ~ - One’s complement
- >> - Right shift
- << - Left shift
- & - Bitwise AND
- | - Bitwise OR
- ^ - Bitwise XOR

Summary

- To help manipulate hardware oriented data—individual bits rather than bytes a set of bitwise operators are used.
- The bitwise operators include operators like one's complement, right-shift, left-shift, bitwise AND, OR, and XOR.
- The one's complement converts all zeros in its operand to 1s and all 1s to 0s.
- The right-shift and left-shift operators are useful in eliminating bits from a number—either from the left or from the right.
- The bitwise AND operators is useful in testing whether a bit is on/off and in putting off a particular bit.
- The bitwise OR operator is used to turn on a particular bit.
- The XOR operator works almost same as the OR operator except one minor variation.

Question

- unsigned int m=32;

printf(“%x”,~m)

- unsigned int a,b,,c,d,e,f;

a=b=c=d=e=f=32;

a<<=2;

b>>=2;

c^=2;

d|=2;

e&=2;

f~=2; //print all in hex

- Left shift is equivalent to multiply by 2 and right shift is divide by 2? true/false
- printf(“%d>>%d %d>>%d\n”, 4>>1,8>>1);
- printf (“%d”,((64>>(2+1-2))&(~(1<<2)))

Answer

- fffffdf
- Error no such operation as ~=
- true
- 2>>4 Garbage value>> Garbage Value
- 32

Contact Info

- trainers@finaldesk.com
- rishabh@finaldesk.com
- nilesh@finaldesk.com
- jignesh@finaldesk.com
- yash@finaldesk.com
- anand@finaldesk.com