Manipulating Bits in C

By :-
Rachit Garg

First of all, why do we need to manipulate bits ?

- The best reason is sometimes you can do a lot of stuff easily and much more efficiently by directly manipulating the bits in binary.
- It gives you this ability to directly do a lot of useful stuff with the files that you already have.

Well, Here are some operations you can do on bits in c

- Bitwise AND (&)
- Bitwise OR (|)
- Bitwise NOT (~)
- Bitwise XOR (^)
- Bit Shifting (<< or >>)
- Bit Masking (we'll talk about this later)

Bitwise AND

- ‘&’ operator is used to do Bitwise AND in C.
- It is used to AND two binary numbers.

Bitwise OR

- ‘ | ‘ is used for doing Bitwise OR of two binary numbers in C.

Bitwise NOT

- Basically ‘~’ is used for inverting a binary value in C i.e simply flipping the respective 0s and 1s.

Bitwise XOR

- ‘ ^ ‘ is used to do XOR operation on two binary numbers.

OK, So Let’s Take an Example

```
#include <stdio.h>

int main()

{

int a,b;

a = 0b001001; // 9

b = 0b001110; // 14

printf(“The AND of a and b is %i”,(a & b));

printf(“\nThe OR of a and b is %i”, (a | b));

printf(“\nThe XOR of and b is %i\n”, (a ^ b));

printf(“The NOT of a is %i and the NOT of b is %i”, ~a, ~b);

}
```

Here is the Output we are going to get

```
The AND of a and b is 8
The OR of a and b is 15
The XOR of a and b is 7
The NOT of a is -10 and the NOT of b is -15
```

Bit Shifting

- We use ‘<<‘ for shifting bits in a binary number by n bits in left.
- We use ‘ >>’ for shifting bits in a binary number by n bits in right.

Well, Here’s an example to explain it

- The operation x << n shifts the value of x left by n bits.
- Let's look at an example. Suppose x is a char and contains the following 8 bits.

suppose we have a 8 bit binary value

x = 1 1 0 0 0 1 1 1

suppose we shift it by 2 bits on the left we will use

x << 2

this will give us

0 0 0 1 1 1 0 0

Similarly, the >> operator will shift bits on the right in the format x >> n.

Let’s have a code to see it working

```
#include<stdio.h>


int main()

{

int a = 60;


printf("\nNumber is Shifted By 1 Bit  : %d",a >> 1);

printf("\nNumber is Shifted By 2 Bits : %d",a >> 2);

printf("\nNumber is Shifted By 3 Bits : %d",a >> 3);


return(0);

}
```

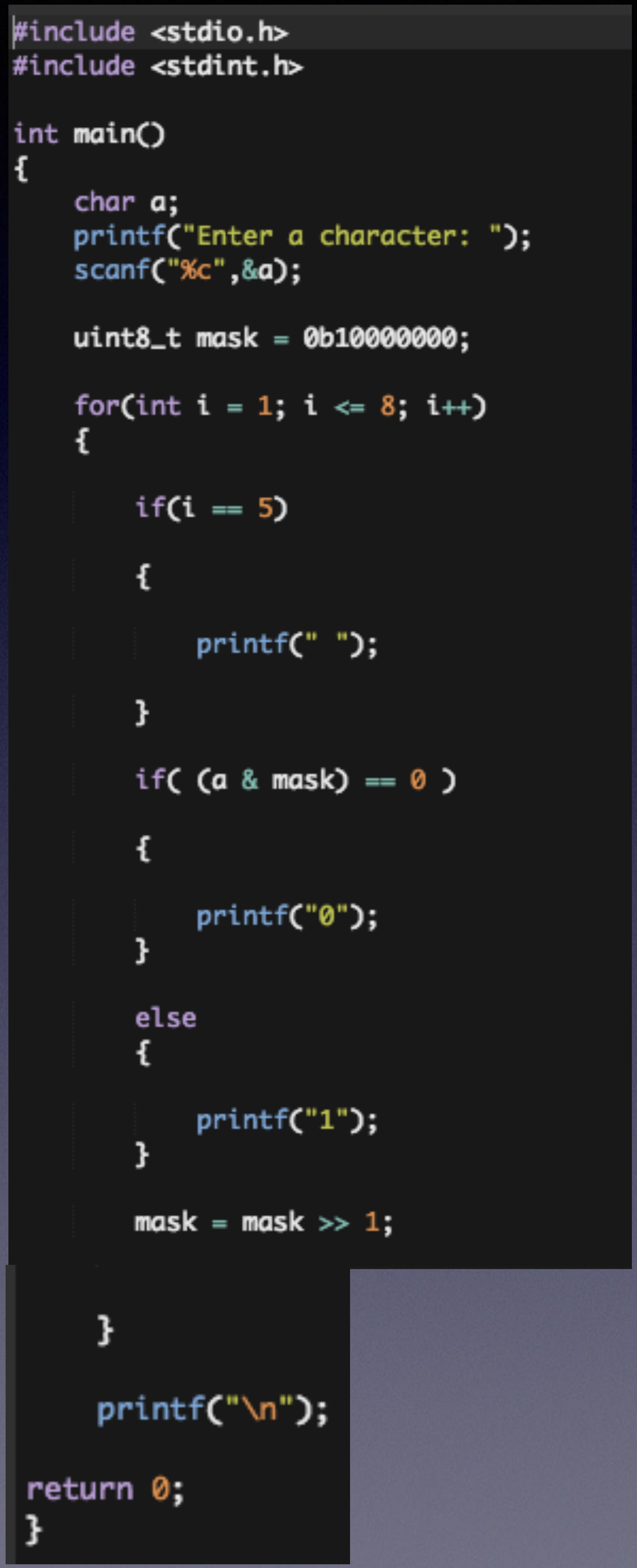
Here's the Output

```
Number is Shifted By 1 Bit  : 30

Number is Shifted By 2 Bits : 15

Number is Shifted By 3 Bits : 7
```

Finally, Bit Masking

- A mask defines which bits you want to keep, and which bits you want to clear.
- Masking is the act of applying a mask to a value. This is accomplished by doing:
- Bitwise ANDing in order to extract a subset of the bits in the value
- Bitwise ORing in order to set a subset of the bits in the value
- Bitwise XORing in order to toggle a subset of the bits in the value

Let's take the first example (character to binary conversion)

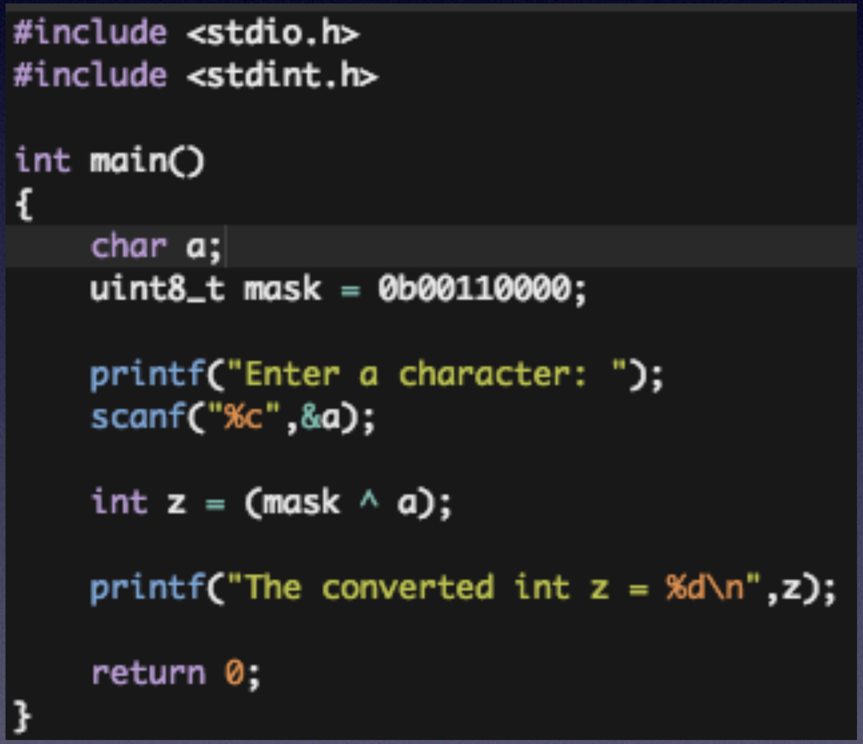


```c
#include <stdio.h>
#include <stdint.h>

int main()
{
    char a;
    printf("Enter a character: ");
    scanf("%c",&a);

    uint8_t mask = 0b10000000;

    for(int i = 1; i <= 8; i++)
    {

        if(i == 5)

        {

            printf(" ");

        }

        if( (a & mask) == 0 )

        {

            printf("0");
        }

        else
        {

            printf("1");
        }

        mask = mask >> 1;


    }

    printf("\n");

 return 0;
}
```

OUTPUT:

```
Enter a character: 3

0011 0011

Enter a character: b

0110 0010

Enter a character: A

0100 0001
```

Here comes the last Example (Character to integer conversion)



```c
#include <stdio.h>
#include <stdint.h>

int main()
{
    char a;
    uint8_t mask = 0b00110000;

    printf("Enter a character: ");
    scanf("%c",&a);

    int z = (mask ^ a);

    printf("The converted int z = %d\n",z);

    return 0;
}
```

OUTPUT:

```
Enter a character: 1
The converted int z = 1

Enter a character: 9
The converted int z = 9
```

And, There’s a Whole lot you can do with it

- Like if you turned on or off a bit in an alphabet called as a flag bit you can even convert a lower to uppercase or vice versa.
- There’s much more than that.

Thank you
Have a nice day :)