**Disciplina: Programação Estruturada**
**Turmas: A1 e A2 – Noturno**

Prof. Dr. Jesús P. Mena-Chalco
Assistente Docente: Rafael J. P. Damaceno

Universidade Federal do ABC

## Operadores Binários

A linguagem C é composta por operadores aritméticos (+, −, / e ∗), lógicos (>, >=, <, <=, ==, !=, && e ||), de incremento/decremento (++ e −−) e binários (<<, >>, &, |, ∧ e ∼). Os binários são utilizados para manipular dados a nível de bit, úteis para aplicações que envolvem microcontroladores. Note que dentre os operadores estão & e |. Estes, grafados com apenas um caractere representam entidades diferentes daqueles usados nos condicionais compostos (quando são grafados de forma duplicada). Veja um exemplo do caso em que representam agrupamento de condicionais:

**Exemplo 1**. Representação dos operadores de agrupamento de condicionais && e ||.

```
1  #include <stdio.h>
2  int main(){
3      int a = 10;
4      int b = 20;
5      if (a < b && b <= 20)
6          printf("ambas as condições são verdadeiras.\n");
7      if (b > a || b == 0)
8          printf("pelo menos uma das condições é verdadeira.\n");
9      return 0;
10 }
```

Neste tutorial focaremos apenas nos operadores para manipulação de bits. Começaremos vendo um pouco sobre como representar números na base decimal em números na base binária. A Tabela 1 mostra a correspondência entre números nestas bases.

Para converter um número qualquer da base decimal para outra base qualquer, basta dividi-lo sucessivamente pela base desejada, até obter-se o dividindo 0; e por fim fazer a leitura dos restos em ordem inversa. Por exemplo: vamos converter 10 na base decimal para 10 na base binária:

- $10/2 = 5$ (resto 0), **dividendo = 5**
- $5/2 = 2$ (resto 1), **dividendo = 2**
- $2/2 = 1$ (resto 0), **dividendo = 1**
- $1/2 = 0$ (resto 1), **dividendo = 0**

Lendo os restos no sentido inverso temos 1, 0, 1 e 0. Portanto, 10 na base decimal é 1010 na base binária. O número 4 na base decimal pode ser escrito com três bits, 100 (um seguido de dois zeros), na base binária. O decimal 15 pode ser escrito com quatro bits, isto é, 1111. Veja a Tabela 1 para consultar alguns valores decimais e suas representações em base binária.

| Decimal | Binário | Decimal | Binário | Decimal | Binário |
|---|---|---|---|---|---|
| 0 | 0000 | 8 | 1000 | 16 | |
| 1 | | 9 | 1001 | 17 | |
| 2 | | 10 | | 18 | 10010 |
| 3 | | 11 | 1011 | 19 | |
| 4 | 0100 | 12 | | 20 | 10100 |
| 5 | | 13 | 1101 | 21 | 10101 |
| 6 | | 14 | | 22 | |
| 7 | 0111 | 15 | | 23 | 10111 |

Tabela 1: Valores decimais e suas representações com o menor número possível de bits.

Exercício 1. Complete a Tabela 1.

Na linguagem C pode-se utilizar uma máscara de bits para fazer atribuições ou modificações em variáveis. Por exemplo, pode-se atribuir a uma variável `a` do tipo inteiro o valor 2 por meio de `0b10`, ou seja, um inicializar `0b` (zero seguido de b), e uma sequência de zeros e uns, no caso, `10`, que no escopo de dois bits, representa o decimal 2. Veja e rode o Exemplo 2:

**Exemplo 2**. Uso de máscara de bits para modificar ou atribuir valores inteiros.

```c
#include <stdio.h>
int main(){
    int a = 0b01; // máscara de bits
    int b = 0b10; // máscara de bits
    int c = a - b;
    printf("%d, %d, %d\n", a, b, c);
}
```

Exercício 2. 0b01 representa que valor inteiro?
Exercício 3. 0b10 representa que valor inteiro?
Exercício 4. Qual é o valor de c, em decimal e em binário?
Exercício 5. O que o `printf` imprime?
Exercício 6. Modifique o programa de forma que $a$ seja o decimal 30 e $b$ o decimal 50. Quantos bits são o mínimo necessário para representar os valores 8 e 15? Qual é o valor da variável c, em decimal e em binário?
Exercício 7. Quantos $k$ bits quantos valores distintos são possíveis de representar?

A Tabela 2 apresenta os operadores binários disponíveis na Linguagem C e a ação que exercem nas variáveis. Os operadores $<<$, $>>$ e $\sim$ agem modificando uma variável e os operadores &, | e $\wedge$ agem em pares ou em conjunto com uma máscara de bits.

| Representação | Função |
|---|---|
| $<<$ | Deslocamento para a esquerda |
| $>>$ | Deslocamento para a direita |
| & | AND |
| \| | OR |
| $\wedge$ | XOR |
| $\sim$ | Complemento |

Tabela 2: Operadores para manipulação de bits na Linguagem C.

A Tabela 3 apresenta a saída esperada ao se aplicar os operadores binários em uma ou duas variáveis. Para cada valor de entrada A, B, C o operador aplicado acarreta uma determinada saída. No escopo deste tutorial, considere que 0 representa um atributo `falso` e 1 um atributo `verdadeiro`. Por exemplo, para as entradas A = 1 e B = 0, obtém-se 0 para o operador & (AND), já que a sua lógica determina que apenas se ambas as entradas forem `verdadeira`, a saída também o será. Considerando a entrada A = B = 1, a saída para o operador | é 1, já que a saída é `verdadeira`, se alguma das entradas for `verdadeira`.

| A | B | & |
|---|---|---|
| 0 | 0 | **0** |
| 0 | 1 | **0** |
| 1 | 0 | **0** |
| 1 | 1 | **1** |

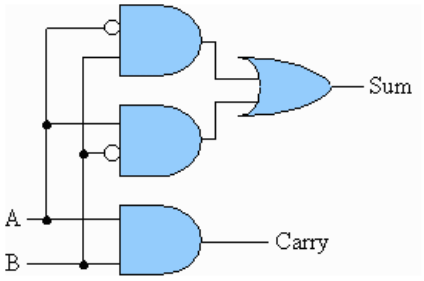
| A | B | \| |
|---|---|---|
| 0 | 0 | **0** |
| 0 | 1 | **1** |
| 1 | 0 | **1** |
| 1 | 1 | **1** |

| A | B | $\wedge$ |
|---|---|---|
| 0 | 0 | **0** |
| 0 | 1 | **1** |
| 1 | 0 | **1** |
| 1 | 1 | **0** |

| C | $\sim$ |
|---|---|
| 0 | **1** |
| 1 | **0** |

Tabela 3: Tabela verdade para os operadores &, |, $\wedge$ e $\sim$.

Na Figura 1 há um trecho do problema 1026 do URI Online. No Exemplo 3 há a resolução parcial do problema. Que operador binário resolve o problema? Substitua o termo OPERADOR por um dos operadores binários para obter a solução correta.




6+9=15 parece ok. Mas como pode estar certo 4+6=2?

Veja só. Mofiz trabalhou duro durante seu curso de Eletrônica Digital, mas quando lhe foi solicitado que implementasse um somador de 32 bits como exame no laboratório, ele acabou fazendo algum erro na parte de projeto. Depois de vasculhar seu projeto por uma hora e meia, ele encontrou seu erro. Ele estava fazendo soma de bits, mas seu carregador de bit (carry) sempre apresentava como saída o valor zero. Portanto,

```
4  = 00000000 00000000 00000000 00000100
+6 = 00000000 00000000 00000000 00000110
---------------------------------------
2  = 00000000 00000000 00000000 00000010
```

Figura 1: Trecho do problema 1026 do URI (Carrega ou não Carrega?) que usa os conceitos de operadores binários. Acesse o endereço: https://www.urionlinejudge.com.br/judge/pt/problems/view/1026 para visualizar o problema completo e testar a solução.

**Exemplo 3**. Resolução parcial do problema 1026 do URI (Carrega ou Não Carrega).

```c
#include <stdio.h>
int main() {
    unsigned long int n1, n2;
    while(scanf("%lu %lu", &n1, &n2) != EOF)
        printf("%lu\n", n1 OPERADOR n2); // qual é o OPERADOR?
    return 0;
}
```

O Exemplo 4 a seguir ilustra o uso dos operadores binários &, | e ∧. Rode o programa e verifique quais são os valores, na base decimal, das variáveis `xe`, `xo` e `xoe`. E na base binária, como é a representação destas variáveis?

**Exemplo 4**. Uso de operadores binários AND (&), OR (|) e XOR (∧).

```c
#include <stdio.h>
int main(){
    int x0 = 0b01;
    int x1 = 0b10;
    int xe = x0 & x1;
    int xo = x0 | x1;
    int xoe = x0 ^ x1;
    printf("x0=%d, x1=%d, xe=%d, xo=%d, xor=%d\n", x0, x1, xe, xo, xoe);
    return 0;
}
```

No caso da aplicação dos operadores de deslocamento à esquerda (<<) à uma variável, o seu último bit é movido para a próxima posição à esquerda (caso o deslocamento seja de 1). No caso da aplicação dos operadores de deslocamento à direita (<<) à uma variável, o seu primeiro bit é movido para a próxima posição à direita (caso o deslocamento seja de 1). Por exemplo, se x0 = 0b0001, aplicando-se x0 << 1, obtém-se 0b0010. Aplicando-se x0 << 2, obtém-se 0b0100. O mesmo raciocínio é aplicado para o operador >>. Veja o Exemplo 5.

**Exemplo 5**. Uso de operadores binários de deslocamento (<< e >>).

```c
#include <stdio.h>
int main(){
    int x0 = 0b001;
    int x1 = x0 << 1;
    int x2 = x1 << 1;
    int x3 = x0 << 2;
    printf("x0=%d, x1=%d, x2=%d, x3=%d\n", x0, x1, x2, x3);
    return 0;
}
```

Exercício 9. Qual é o valor de x1, x2, x3 e x4?
Exercício 10. Modifique o Exemplo 5 trocando o operador << por >>. Quais são os novos valores de x1, x2, x3 e x4?
Exercício 11. Modifique o Exemplo 5 trocando x0 por 0b10000. Que número na base decimal 0b10000 representa? Quais são os novos valores de x1, x2, x3 e x4?