Sistemas Digitais

Operações aritméticas

Prof. Manoel Eusebio de Lima
Centro de Informática
Universidade Federal de Pernambuco

Adição e subtração de números binários

- **Utilizando operações básicas para adição e subtração é também possível efetuarmos multiplicação e divisão.**
  - **Multiplicação pode ser feita pela repetição de adições**
  - **Divisão poder ser feita essencialmente pela repetição de subtrações**
- **De fato o que queremos mostrar ainda é que é inteiramente possível construir um computador no qual um "adicionador" é a única Unidade Aritmética presente.**

Adição de números binários

- **Se dois números binários de r-bits são adicionados o resultado poderá possuir r+1 bits por causa do carry.**

```
     1  1  1
     1  0  1  1  0
  1  0  1  1  1  0
  ----------------
  1  0  0  1  0  0
```

- **Assim uma soma de dois números de 5 bits resultou em um número de 6 bits.**
- **Regra Geral:**
  - **Se a soma de dois números de r bits cai em um valor menor ou igual a 2^r-1 então é possível se ter o resultado da soma registrado também em um registrador de r-bits. Caso contrário, é necessário um registrador de r+1 bits para armazenar o resultado da adição.**

Subtração binária

- **Assim como a adição, a subtração obedece o mesmo caminho que subtração decimal**

```
   0     1     1  1 0   (borrowing)
 - 0   - 0   - 1   - 1
 ---   ---   ---   ---
   0     1     0     1
```

- **Exemplo:**

```
      1  1  0  1  1
   -  0  1  1  0  1
      -------------
      0  1  1  1  0
```

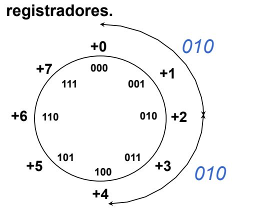
- **Observamos que existe uma diferença no processo para operar adição e subtração.**
- **A princípio portanto precisaríamos de dois circuitos diferentes para operar as duas funções.**
- **Existe no entanto, mecanismos que podem minimizar diferenças na implementação de tais funções lógicas baseados no sistema modular de números.**

Subtração binária

- Como desenvolver um circuito adicionador/subtrator otimizado?
- **Questionamentos**
  - **Como o método para codificação influência na realização das operações?**
  - **Como números negativos são representados?**
  - **Quantos bits são necessários para representar a informação?**

Sistema Numérico Modular

- **Em computação nós temos limitações no tamanho de registradores para representar números e a aritmética modular obedece tais limitações.**
- **Os números são finitos e devem ser representados no intervalo entre 0 e 2^r-1, onde r é o número de bits dos registradores.**

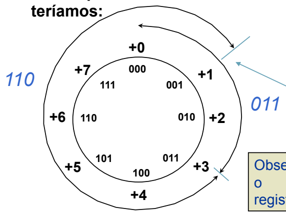


Representação circular de números armazenados em registradores de 3 bits sem sinal.
A= 010
B=010
A+B=100

Sistema Numérico Modular

- **Se agora A= 011 e B= 110. Neste caso observamos que A+B é maior que 2^r para r=3.**
- **Se a soma é igual ou maior que 2^r, o número resultante é o resto que nós obtemos subtraindo 2^r da soma. Assim teríamos:**

A+B=011+110=1<u>001</u>



Ou seja:
1001 - 2^r =1001-1000
$\boxed{= 001}$

Observe que estamos considerando o limite de armazenamento do registrador do circuito.

Sistema Numérico Modular

- **Se A e B são de módulos equivalentes N, o resto obtido quando dividimos A por N é o mesmo que o resto obtido no divisão de B por N.**
  N=2^r A $\equiv_N$ B (representação formal)
- Exemplo:
  A=10 B=18 dizemos que: resto

$A \equiv_8 B$

A $\equiv_{2^3}$ B desde que A= 10=1*8+<u>2</u> e B=18=2*8+<u>2</u>

No sistema de número modular

$0 \equiv_N N$
$1 \equiv_N N+1$
$2 \equiv_N N+2$

$N-1 \equiv_N N+N-1 = 2N-1$

Adição módulo N= 2^2 =4; r=2

+	0	1	2	3
0	0	1	2	3
1	1	2	3	$4 \equiv_4 0$
2	2	3	$4 \equiv_4 0$	$5 \equiv_4 1$
3	3	$4 \equiv_4 0$	$5 \equiv_N 1$	$6 \equiv_4 2$

Complemento a 2

- **Representação de nos. negativos complementados a 2.**
  - **Um número B é negativo de A se A+B=0 => B=-A**
  - **O que acontece se nós trabalharmos com um conjunto de números no módulo N, onde A+B $\equiv_N$ 0?**
  - **Isto indica que A é negativo de B. Mas B neste caso não é único.**
  - **B é tal que B=KN-A, K=0, 1, 2, ... satisfaz a condição que B é negativo de A no módulo N**

Exemplo: k=0, B= - A
k=1, B=N-A

Consideremos módulo N= 4 e A = 3. Assim teríamos:
B=kN-A => B=kN-(3), com k=1, r=2, N=2^r= 4,
teremos B=N-A => B=4-3=$_4$1

Assim, B+A=0 => (<u>01</u>+<u>11</u>) = <u>1</u>00 (0 no módulo 4)
ignorado

Complemento a 2

- **Nós podemos usar N-A em qualquer cálculo chamando -A, contanto que usemos operações no módulo N.**
- **De uma forma geral C=D-A é equivalente (módulo N) à C=D+(N-A)**
- <u>Se nós podemos encontrar N-A não envolvendo subtração nós vemos que a operação de subtração se transforma em uma operação de adição.</u>
  - Considere a operação usando-se registradores de r bits e módulo N=2^r. Representemos N em binário por

$$\underbrace{N=1000.....00}_{r+1\ \text{termos}} = \underbrace{111...11}_{r\ \text{termos}} + \underbrace{0000..1}_{r\ \text{termos}}$$

façamos

A = a_{r-1}, a_{r-2}.......,a_0 um número de r bits

$$N-A = \underbrace{(1-a_{r-1}),(1-a_{r-2}), ...(1-a_0)+000...01}_{\textbf{complemento a 2 de A}}$$

, onde $\begin{cases} a_j=0 \text{ ou } 1 \text{ e} \\ 1-a_j= 1 \text{ ou } 0 \end{cases}$

Complemento a 2

- **O valor (N-A) é chamado complemento a 2 de A. De uma maneira menos formal, o complemento a 2 de um número binário de r bits é encontrado pela expressão:**

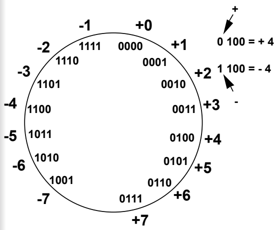
  (N-A):= $\overline{A}$+[1]
  - **Exemplo: Complemento a 2 de A:=[01000]**
    $\overline{A}$ + [1] = 10111+00001 = 11000
- **Exemplo:**
  Qual o complemento a 2 de A = +2
  r=3 A=010 N=2^r => N = 2^3 = 1000
  N=1000 = 111+001
  então N-A = 1000-010 = (111+001) - 010= (111- 010) + 001
  = (1-0),(1-1),(1-0) + 001
  = 1 0 1 + 001
  = <u>110</u>

Complemento a 2

- **Método simples de conversão**
  Complemento a 2 = Complemento a 1 + 1

  Ex: Comp'2 de 7 => 0111 -> 1000 + 1 -> 1001 (representação of -7)

  Comp'2 de -7 => 1001 -> 0110 + 1 -> 0111 (representação of 7)



þ **Características:**
- Similar a complemento a 1 exceto que deslocamos uma posição no sentido do relógio
- Apenas uma representação para o zero (0)
- Faixa numérica para r bits **= +/-2^{r-1} -1**

Exemplos

1. $A = 0111_2 \quad = +7_{10} \quad B = 0110_2 \quad = +6_{10}$
   $A\text{-}B = 0111 + 1010 = 10001 \equiv_{16} \; 0001$

2. $A = 0011_2 \quad =+3_{10} \quad B = 1101_2 \quad = -3_{10}$
   $A\text{-}B = 0011 + 0011 = 0110$

3. $A = 1101_2 \quad = -3_{10} \quad B = 0100_2 \quad =+4_{10}$
   $A\text{-}B = 1101 + 1100 = 11001 \equiv_{16} \; 1001$

Sempre que estivermos trabalhando com complemento a 2, o carry gerado no bit mais significativo é ignorado.
Esta generalização assume que o resultado real do cálculo cai sempre dentro dos limites ***+/- (2^{r-1}-1)***

Overflow

- Quando efetuamos operações aritméticas usando complemento a 2 devemos considerar a possibilidade de obtermos resultados que extrapolam os limites de representação dos números num dado módulo N, ou seja, fora dos limites de +/- (2^r-1)
  Quando isto ocorre dizemos que temos uma condição de aritmética de overflow.
  Overflow poderá ocorrer quando:
  1. **Os dois operandos têm o mesmo sinal e a adição complemento a 2 dos operandos produzir um resultado com um sinal oposto aos mesmos.**

  Exemplo: (N=16, r=4)

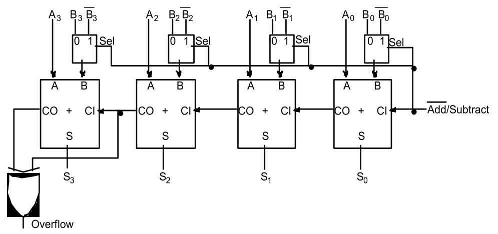
  $A = 0110_2 = 6_{10}$ $B = 0011_2 = 3_{10}$

  Então $A+B = 0110_2 + 0011_2 = 1001_2$, mas 1001_2 não é nove na aritmética complemento a 2 desde que o sinal mais significativo é o de sinal. Assim o resultado é -7, o que provoca o Overflow.

  Overflow = $\boxed{c_4 \oplus c_3}$

Somador/Subtrator

- Circuito somador e subtrator integrados



Números fracionários

- **As partes separadas por vírgula em números fracionários devem ser tratadas como dois números inteiros. Depois de completada a operação envolvendo os dois números, recolocamos a vírgula no lugar adequado.**
- **Exemplo:**

  A-B com A=1101.10 e B=10.111

  Os dois são números positivos a priori. Se nós trabalharmos com aritmética complemento a 2 nós devemos primeiro decidir o valor de r, ou seja, o número de bits necessários para representá-los corretamente.

  Para r = 8 , considerando bit de sinal teríamos A = 01101.100, adicione 0 como LSB

  <u>O complemento a 2 de B será</u>:

  Adicionemos 0 extras para r = 8 bits, assim B= - 00010.111

  Complemento a 2 de B = N-B = 11101.000 + 0000.001 = 11101.001

  Assim A-B = 01101.100
  11101.001

  Resultado A - B = 101010.101

  carry descartado

Exercício - Adição em BCD

- Dígitos em BCD são representados entre 0 e 9 e possuem representação de 0000 a 1001 no sistema binário.
- Exemplo:

5+3 = 8
0101 (5)
+ <u>0011 (3)</u>
1000 = 8

5+8 = 13
0101 (5)
+<u>1000 (8)</u>
1101 = 13

Problema: Como resolver números acima de 9 se só posso representar digitos até 9?

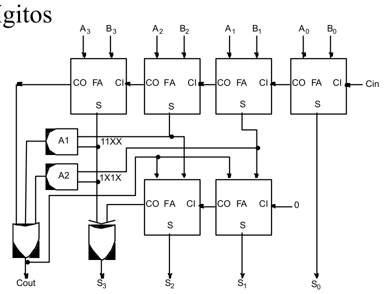
Solução: some 6 (0110) se o digito excede o número 9

13 = 0001 0011 (binário composto por dois digitos BCDs)
Para atingirmos esta solução soma-se 6 ao digito BDC que excede 9.
Assim 5 (0101) + 8 (1000) = 13 (1101) -> convertendo para BDC
teremos: 1101
<u>+0110</u>
1 0011 (13 em BCD)

digito mais significativo
digito menos significativo

Projeto - Adição em BCD

Implementar um somador BCD de dois dígitos



Soma BCD