Aula 3: Aritmética, Representação de dados

Circuitos Digitais

Rodrigo Hausen

CMCC – UFABC

28 e 30 de janeiro de 2013

http://compscinet.org/circuitos

Multiplicação binária

- Algoritmo da multiplicação: mesma idéia usada na base decimal.

```
      11011
x       101
   --------
      11011
     00000
+   11011
   --------
   10000111
```

- Algoritmo da multiplicação: mesma idéia usada na base decimal.

```
      11011
x       101
   --------
      11011
     00000
+   11011
   --------
   10000111
```

- Note que a tabuada da multiplicação na base 2 é muito mais fácil.

$\times$	0	1
0	0	0
1	0	1

Multiplicação binária

- Algoritmo da multiplicação: mesma idéia usada na base decimal.

```
     11011
x      101
  --------
     11011
    00000
+  11011
  --------
  10000111
```

- Note que a tabuada da multiplicação na base 2 é muito mais fácil.

$\times$	0	1
0	0	0
1	0	1

- Se A tem n algarismos e B tem m algarismos, então o produto $A \times B$ terá, no máximo,

Multiplicação binária

- Algoritmo da multiplicação: mesma idéia usada na base decimal.

```
      11011
x       101
   --------
      11011
     00000
+   11011
   --------
   10000111
```

- Note que a tabuada da multiplicação na base 2 é muito mais fácil.

$\times$	0	1
0	0	0
1	0	1

- Se A tem n algarismos e B tem m algarismos, então o produto $A \times B$ terá, no máximo, $n + m$ algarismos.

Multiplicação binária

Para casa: escrever o algoritmo de multiplicação binária para números naturais.

- Note que não é necessário armazenar todas as parcelas da soma ao mesmo tempo.

```
      11011
x       101
  ---------
```

Multiplicação binária

Para casa: escrever o algoritmo de multiplicação binária para números naturais.

- Note que não é necessário armazenar todas as parcelas da soma ao mesmo tempo.

```
       11011
x   ________101
       11011
+     000000 <- desloca 1
```

Para casa: escrever o algoritmo de multiplicação binária para números naturais.

- Note que não é necessário armazenar todas as parcelas da soma ao mesmo tempo.

```
      11011
x       101
     ------
     011011
```

Multiplicação binária

Para casa: escrever o algoritmo de multiplicação binária para números naturais.

- Note que não é necessário armazenar todas as parcelas da soma ao mesmo tempo.

```
       11011
x        101
   ---------
      011011
+   1101100<- desloca 2
```

Para casa: escrever o algoritmo de multiplicação binária para números naturais.

- Note que não é necessário armazenar todas as parcelas da soma ao mesmo tempo.

```
      11011
x       101
  ---------
   10000111
```

Divisão binária

- Algoritmo da divisão longa: de novo, emprestamos a idéia da base decimal.

Divisão binária

- Algoritmo da divisão longa: de novo, emprestamos a idéia da base decimal.
- Novamente, a tabuada binária facilita as contas. Os algarismos só podem ser 1 ou 0.

```
10000111   |__101___
```

Divisão binária

- Algoritmo da divisão longa: de novo, emprestamos a idéia da base decimal.
- Novamente, a tabuada binária facilita as contas. Os algarismos só podem ser 1 ou 0.

```
  10000111   | 101
- 101
   -1
```

Divisão binária

- Algoritmo da divisão longa: de novo, emprestamos a idéia da base decimal.
- Novamente, a tabuada binária facilita as contas. Os algarismos só podem ser 1 ou 0.

```
   10000111    |  101
                ------
 - 101          0
   ---
    -1
```

Divisão binária

- Algoritmo da divisão longa: de novo, emprestamos a idéia da base decimal.
- Novamente, a tabuada binária facilita as contas. Os algarismos só podem ser 1 ou 0.

```
  10000111   | 101
- 101          0
```

- Algoritmo da divisão longa: de novo, emprestamos a idéia da base decimal.
- Novamente, a tabuada binária facilita as contas. Os algarismos só podem ser 1 ou 0.

```
  10000111    |  101
- 101            0
   11
```

Divisão binária

- Algoritmo da divisão longa: de novo, emprestamos a idéia da base decimal.
- Novamente, a tabuada binária facilita as contas. Os algarismos só podem ser 1 ou 0.

```
  10000111   | 101
                -------
- 101          01
  ---
    11
```

Divisão binária

- Algoritmo da divisão longa: de novo, emprestamos a idéia da base decimal.
- Novamente, a tabuada binária facilita as contas. Os algarismos só podem ser 1 ou 0.

```
110111    |_101___
           01
```

- Algoritmo da divisão longa: de novo, emprestamos a idéia da base decimal.
- Novamente, a tabuada binária facilita as contas. Os algarismos só podem ser 1 ou 0.

```
   110111   | 101
 - 101      -------
     1        01
```

Divisão binária

- Algoritmo da divisão longa: de novo, emprestamos a idéia da base decimal.
- Novamente, a tabuada binária facilita as contas. Os algarismos só podem ser 1 ou 0.

```
    110111    | 101
  - 101       -------
      1        011
```

Divisão binária

- Algoritmo da divisão longa: de novo, emprestamos a idéia da base decimal.
- Novamente, a tabuada binária facilita as contas. Os algarismos só podem ser 1 ou 0.

```
   1111   | 101
 - 101     011
  -10
```

Divisão binária

- Algoritmo da divisão longa: de novo, emprestamos a idéia da base decimal.
- Novamente, a tabuada binária facilita as contas. Os algarismos só podem ser 1 ou 0.

```
  1111   | 101
- 101     0110
  -10
```

Divisão binária

- Algoritmo da divisão longa: de novo, emprestamos a idéia da base decimal.
- Novamente, a tabuada binária facilita as contas. Os algarismos só podem ser 1 ou 0.

```
  1111   | 101
- 101      01101
   10
```

Divisão binária

- Algoritmo da divisão longa: de novo, emprestamos a idéia da base decimal.
- Novamente, a tabuada binária facilita as contas. Os algarismos só podem ser 1 ou 0.

$$
\begin{array}{rl}
 101 & \underline{|\ \ 101\ \ } \\
- \underline{101} & \ \ 011011 \\
 0 &
\end{array}
$$

- Algoritmo da divisão longa: de novo, emprestamos a idéia da base decimal.
- Novamente, a tabuada binária facilita as contas. Os algarismos só podem ser 1 ou 0.

$$
\begin{array}{rl}
101 & \underline{|\ \ 101\ \ \ } \\
-\ \underline{101} & \ \ 011011 = \text{quociente} \\
0 = \text{resto} &
\end{array}
$$

Para casa: escrever, em pseudocódigo, o algoritmo da divisão binária para números naturais.

- Calcular $A \div B$, onde $A = a_{n-1} \ldots a_0$, $B = b_{m-1} \ldots b_0$ e $m \leq n$
- Note que as subtrações "da esquerda para a direita", são, na verdade, subtrações do dividendo pelo divisor multiplicado por 2^i, para $i = n - m \ldots 0$

```
  10000111   |  101
- 10100000
   -100111
```

Divisão binária

Para casa: escrever, em pseudocódigo, o algoritmo da divisão binária para números naturais.

- Calcular $A \div B$, onde $A = a_{n-1} \dots a_0$, $B = b_{m-1} \dots b_0$ e $m \leq n$
- Note que as subtrações "da esquerda para a direita", são, na verdade, subtrações do dividendo pelo divisor multiplicado por 2^i, para $i = n - m \dots 0$

```
  10000111   |  101
- 10100000     0
   -100111
```

Divisão binária

Para casa: escrever, em pseudocódigo, o algoritmo da divisão binária para números naturais.

- Calcular $A \div B$, onde $A = a_{n-1} \ldots a_0$, $B = b_{m-1} \ldots b_0$ e $m \leq n$
- Note que as subtrações "da esquerda para a direita", são, na verdade, subtrações do dividendo pelo divisor multiplicado por 2^i, para $i = n - m \ldots 0$

```
  10000111   |  101
- 1010000     0
   110111
```

Divisão binária

Para casa: escrever, em pseudocódigo, o algoritmo da divisão binária para números naturais.

- Calcular $A \div B$, onde $A = a_{n-1} \dots a_0$, $B = b_{m-1} \dots b_0$ e $m \leq n$
- Note que as subtrações "da esquerda para a direita", são, na verdade, subtrações do dividendo pelo divisor multiplicado por 2^i, para $i = n - m \dots 0$

```
  10000111    | 101
- 1010000       01
   110111
```

Divisão binária

Para casa: escrever, em pseudocódigo, o algoritmo da divisão binária para números naturais.

- Calcular $A \div B$, onde $A = a_{n-1} \ldots a_0$, $B = b_{m-1} \ldots b_0$ e $m \leq n$
- Note que as subtrações "da esquerda para a direita", são, na verdade, subtrações do dividendo pelo divisor multiplicado por 2^i, para $i = n - m \ldots 0$

```
   110111   | 101
 - 101000     01
    01111
```

Divisão binária

Para casa: escrever, em pseudocódigo, o algoritmo da divisão binária para números naturais.

- Calcular $A \div B$, onde $A = a_{n-1} \dots a_0$, $B = b_{m-1} \dots b_0$ e $m \leq n$
- Note que as subtrações "da esquerda para a direita", são, na verdade, subtrações do dividendo pelo divisor multiplicado por 2^i, para $i = n - m \dots 0$

```
  110111   | 101
- 101000    011
   01111
```

Divisão binária

Para casa: escrever, em pseudocódigo, o algoritmo da divisão binária para números naturais.

- Calcular $A \div B$, onde $A = a_{n-1} \dots a_0$, $B = b_{m-1} \dots b_0$ e $m \leq n$
- Note que as subtrações "da esquerda para a direita", são, na verdade, subtrações do dividendo pelo divisor multiplicado por 2^i, para $i = n - m \dots 0$

```
  01111   | 101
- 10100    0110...
  -...
```

Para casa: escrever, em pseudocódigo, o algoritmo da divisão binária para números naturais.

- Calcular $A \div B$, onde $A = a_{n-1} \ldots a_0$, $B = b_{m-1} \ldots b_0$ e $m \leq n$
- Note que as subtrações "da esquerda para a direita", são, na verdade, subtrações do dividendo pelo divisor multiplicado por 2^i, para $i = n - m \ldots 0$

```
  01111   | 101
- 10100     0110...
  -...
```

- Se a diferença é positiva, ela passa a ser o próximo dividendo.
- Pare quando $i = 0$.

Divisão binária: divisão não-inteira

- Note que se A é múltiplo de B, o resultado da última subtração será 0

Divisão binária: divisão não-inteira

- Note que se A é múltiplo de B, o resultado da última subtração será 0
- E se A não for múltiplo de B?

Divisão binária: divisão não-inteira

- Note que se A é múltiplo de B, o resultado da última subtração será 0
- E se A não for múltiplo de B?
- Podemos continuar a divisão, adicionando a vírgula (ponto, em inglês).

Divisão binária: divisão não-inteira

- Note que se A é múltiplo de B, o resultado da última subtração será 0
- E se A não for múltiplo de B?
- Podemos continuar a divisão, adicionando a vírgula (ponto, em inglês).
- para cada algarismo adicionado depois da vírgula, multiplique o dividendo por 2 (ou seja, adicione um 0 à direita)

Divisão binária: divisão não-inteira

- Note que se A é múltiplo de B, o resultado da última subtração será 0
- E se A não for múltiplo de B?
- Podemos continuar a divisão, adicionando a vírgula (ponto, em inglês).
- para cada algarismo adicionado depois da vírgula, multiplique o dividendo por 2 (ou seja, adicione um 0 à direita)
- Pare quando o resultado tiver k algarismos depois da vírgula.

Ex.: $110 \div 101$

Divisão binária: divisão não-inteira

- Note que se A é múltiplo de B, o resultado da última subtração será 0
- E se A não for múltiplo de B?
- Podemos continuar a divisão, adicionando a vírgula (ponto, em inglês).
- para cada algarismo adicionado depois da vírgula, multiplique o dividendo por 2 (ou seja, adicione um 0 à direita)
- Pare quando o resultado tiver k algarismos depois da vírgula.

Ex.: $110 \div 101$

Para casa: escrever esse algoritmo.

Números racionais

- O que acontece com os algoritmos da soma, subtração, multiplicação e divisão quando os números sendo operados não são inteiros?

Números racionais

- O que acontece com os algoritmos da soma, subtração, multiplicação e divisão quando os números sendo operados não são inteiros?
- Sem perder a generalidade, suporemos que A e B possuem k algarismos depois da vírgula. (Se eles não tiverem a mesma quantidade de algarismos após a vírgula?)

$$
\begin{aligned}
A &= a_{n-1}a_{n-2}\dots a_1a_0 \quad , \quad a_{-1}\dots a_{-k} \\
B &= b_{m-1}a_{m-2}\dots b_0 \quad , \quad b_{-1}\dots b_{-k}
\end{aligned}
$$

- O que acontece com os algoritmos da soma, subtração, multiplicação e divisão quando os números sendo operados não são inteiros?
- Sem perder a generalidade, suporemos que A e B possuem k algarismos depois da vírgula. (Se eles não tiverem a mesma quantidade de algarismos após a vírgula?)

$$A = a_{n-1}a_{n-2}\ldots a_1a_0 \quad , \quad a_{-1}\ldots a_{-k}$$
$$B = b_{m-1}a_{m-2}\ldots b_0 \quad , \quad b_{-1}\ldots b_{-k}$$

- Caso mais fácil: divisão. Note que:

$$\frac{A}{B} = \frac{A\times 2^k}{B\times 2^k} = \frac{a_{n-1}a_{n-2}\ldots a_1a_0a_{-1}\ldots a_{-k}}{b_{m-1}a_{m-2}\ldots b_0b_{-1}\ldots b_{-k}}$$

- O que acontece com os algoritmos da soma, subtração, multiplicação e divisão quando os números sendo operados não são inteiros?
- Sem perder a generalidade, suporemos que A e B possuem k algarismos depois da vírgula. (Se eles não tiverem a mesma quantidade de algarismos após a vírgula?)

$$A = a_{n-1}a_{n-2}\ldots a_1a_0 \quad , \quad a_{-1}\ldots a_{-k}$$
$$B = b_{m-1}a_{m-2}\ldots b_0 \quad , \quad b_{-1}\ldots b_{-k}$$

- Caso mais fácil: divisão. Note que:

$$\frac{A}{B} = \frac{A \times 2^k}{B \times 2^k} = \frac{a_{n-1}a_{n-2}\ldots a_1a_0a_{-1}\ldots a_{-k}}{b_{m-1}a_{m-2}\ldots b_0b_{-1}\ldots b_{-k}}$$

Simplesmente ignore a vírgula!

Números racionais

- Para a soma e a subtração: como os algoritmos são "copiados" da versão para números na base 10, a solução é simples: ignore, inicialmente a vírgula. Após a soma, recoloque a vírgula no seu lugar (conte k algarismos à direita).

- Para a soma e a subtração: como os algoritmos são "copiados" da versão para números na base 10, a solução é simples: ignore, inicialmente a vírgula. Após a soma, recoloque a vírgula no seu lugar (conte k algarismos à direita).
- Para a multiplicação: de novo, a inspiração vem da base decimal. Ignore, inicialmente a vírgula e, após a multiplicação, recoloque a vírgula no seu lugar (conte **2k** algarismos à direita).

Para casa:

(1) escreva as versões dos algoritmos da soma, subtração, multiplicação e divisão para números racionais sem sinal (positivos) com k algarismos após a vírgula;

(2) altere os algoritmos para permitir números racionais com sinal.

Representação numérica

- Representação de números no papel: usamos tantos dígitos forem necessários.
- Limitado apenas pela quantidade de papel, tempo disponível para escrever os dígitos, paciência...

- Representação de números no papel: usamos tantos dígitos forem necessários.
- Limitado apenas pela quantidade de papel, tempo disponível para escrever os dígitos, paciência...

Número π:

```
3.14159265358979323846264338327950288419716939937510582097
4944592307816406286208998628034825342117067982148086513282
3066470938446095505822317253594081284811174502841027019385
2110555964462294895493038196442881097566593344612847564823
3786783165271201909145648566923460348610454326648213393607
2602491412737245870066063155881748815209209628292540917153
6436789259036001133053054882046652138414695194151160943305
7270365759591953092186117381932611793105118548074462379...
```

Representação numérica num computador digital

- Recordando: em um computador digital qualquer informação, em última instância, é representada por um número.

Representação numérica num computador digital

- Recordando: em um computador digital qualquer informação, em última instância, é representada por um número.
- Atualmente, os números são representados internamente em binário (por vários motivos, entre eles facilidade de fazer contas na base 2).

Representação numérica num computador digital

- Recordando: em um computador digital qualquer informação, em última instância, é representada por um número.
- Atualmente, os números são representados internamente em binário (por vários motivos, entre eles facilidade de fazer contas na base 2).
- Um computador digital possui espaço finito para guardar informações.

Representação numérica num computador digital

- Recordando: em um computador digital qualquer informação, em última instância, é representada por um número.
- Atualmente, os números são representados internamente em binário (por vários motivos, entre eles facilidade de fazer contas na base 2).
- Um computador digital possui espaço finito para guardar informações.
- Por questões de eficiência, geralmente o processamento de dados (ou seja, números) não é feito algarismo binário por algarismo binário, e sim **por grupos de algarismos binários** de uma só vez.

Representação numérica num computador digital

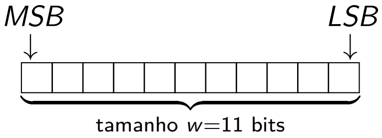
- Recordando: em um computador digital qualquer informação, em última instância, é representada por um número.
- Atualmente, os números são representados internamente em binário (por vários motivos, entre eles facilidade de fazer contas na base 2).
- Um computador digital possui espaço finito para guardar informações.
- Por questões de eficiência, geralmente o processamento de dados (ou seja, números) não é feito algarismo binário por algarismo binário, e sim **por grupos de algarismos binários** de uma só vez.

Bits e palavras

- Abreviação: algarismo binário = **bit** (do inglês **b**inary dig**it**)

Bits e palavras

- Abreviação: algarismo binário = **bit** (do inglês **b**inary dig**it**)
- A unidade natural de processamento de um determinado sistema é chamada **palavra de dado**; é, basicamente, uma sequência de bits com tamanho fixo que é processada em conjunto.

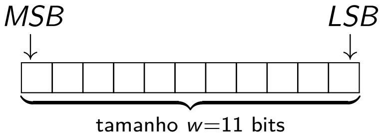


MSB = Most Significant Bit = bit mais significativo,

LSB = Least Significant Bit = bit menos significativo

Bits e palavras

- Abreviação: algarismo binário = **bit** (do inglês **b**inary dig**it**)
- A unidade natural de processamento de um determinado sistema é chamada **palavra de dado**; é, basicamente, uma sequência de bits com tamanho fixo que é processada em conjunto.



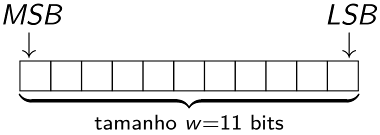
MSB = Most Significant Bit = bit mais significativo,

LSB = Least Significant Bit = bit menos significativo

- Tamanhos de palavras comuns são: 4, 8, 16, 32 e 64 bits.

Bits e palavras

- Abreviação: algarismo binário = **bit** (do inglês **b**inary dig**it**)
- A unidade natural de processamento de um determinado sistema é chamada **palavra de dado**; é, basicamente, uma sequência de bits com tamanho fixo que é processada em conjunto.



MSB = Most Significant Bit = bit mais significativo,

LSB = Least Significant Bit = bit menos significativo

- Tamanhos de palavras comuns são: 4, 8, 16, 32 e 64 bits.
- Nomes comuns para palavras:
  - 8 bits = **byte** (**b**inar**y** **te**rm) ou **octeto**
  - 4 bits = **nibble**
    (curiosidade: nibble, em inglês, significa "mordidinha" = "small bite")

Representando números em palavras binárias

Primeiro caso: número inteiro sem sinal (≥ 0).

Como representar um número inteiro $A = (a_{n-1}a_{n-2}\ldots a_0)_2$ numa palavra de comprimento w?

$$\underbrace{\square\square\square\square\square\square\cdots\square\square\square\square\square}_{w\text{ bits}}$$

Representando números em palavras binárias

Primeiro caso: número inteiro sem sinal (≥ 0).

Como representar um número inteiro $A = (a_{n-1}a_{n-2}\ldots a_0)_2$ numa palavra de comprimento w?

$$\underbrace{\boxed{0}\boxed{0}\boxed{a_{n-1}}\boxed{a_{n-2}}\ldots\boxed{a_2}\boxed{a_1}\boxed{a_0}}_{w \text{ bits}}$$

Qual é o maior inteiro sem sinal que podemos representar?

Representando números em palavras binárias

Primeiro caso: número inteiro sem sinal (≥ 0).

Como representar um número inteiro $A = (a_{n-1}a_{n-2}\ldots a_0)_2$ numa palavra de comprimento w?

$$\underbrace{\boxed{0}\boxed{0}\boxed{a_{n-1}}\boxed{a_{n-2}}\ldots\boxed{a_2}\boxed{a_1}\boxed{a_0}}_{w \text{ bits}}$$

Qual é o maior inteiro sem sinal que podemos representar?

Exemplo: quais inteiros sem sinal podemos representar com 3 bits?

Representando números em palavras binárias

Primeiro caso: número inteiro sem sinal (≥ 0).

Como representar um número inteiro $A = (a_{n-1}a_{n-2}\dots a_0)_2$ numa palavra de comprimento w?

$$\underbrace{\boxed{0}\boxed{0}\boxed{a_{n-1}}\boxed{a_{n-2}}\dots\boxed{a_2}\boxed{a_1}\boxed{a_0}}_{w\text{ bits}}$$

Qual é o maior inteiro sem sinal que podemos representar?

Exemplo: quais inteiros sem sinal podemos representar com 3 bits?

000	001	010	011	100	101	110	111
0	1	2	3	4	5	6	7

Representando números em palavras binárias

Primeiro caso: número inteiro sem sinal (≥ 0).

Como representar um número inteiro $A = (a_{n-1}a_{n-2}\ldots a_0)_2$ numa palavra de comprimento w?

$$\underbrace{\boxed{0}\boxed{0}\boxed{a_{n-1}}\boxed{a_{n-2}}\ldots\boxed{a_2}\boxed{a_1}\boxed{a_0}}_{w \text{ bits}}$$

Qual é o maior inteiro sem sinal que podemos representar?

Exemplo: quais inteiros sem sinal podemos representar com 3 bits?

000	001	010	011	100	101	110	111
0	1	2	3	4	5	6	7

De 0 até $7 = 2^3 - 1$.

Inteiros sem sinal em palavras binárias com w bits.

$$
\begin{array}{ccl}
\text{Palavra} & & \text{Decimal} \\
00\ldots000 & = & 0 \\
00\ldots001 & = & 1 \\
00\ldots010 & = & 2 \\
 & \vdots & \\
11\ldots110 & = & ? \\
11\ldots111 & = & ? = \text{maior inteiro sem sinal com } w \text{ bits}
\end{array}
$$

Inteiros sem sinal em palavras binárias com w bits.

$$
\begin{array}{rcl}
\text{Palavra} & & \text{Decimal} \\
00\ldots000 & = & 0 \\
00\ldots001 & = & 1 \\
00\ldots010 & = & 2 \\
 & \vdots & \\
11\ldots110 & = & ? \\
11\ldots111 & = & ? = \text{maior inteiro sem sinal com } w \text{ bits}
\end{array}
$$

O próximo número na sequência, que não cabe em w bits, é

$$(1\underbrace{00\ldots000}_{w\text{ bits}})_2 = 0 \cdot 2^0 + 0 \cdot 2^1 + \ldots + 0 \cdot 2^{w-1} + 1 \cdot 2^w = 2^w$$

Inteiros sem sinal em palavras binárias com w bits.

$$\begin{array}{rcl} \text{Palavra} & & \text{Decimal} \\ 00\ldots000 & = & 0 \\ 00\ldots001 & = & 1 \\ 00\ldots010 & = & 2 \\ & \vdots & \\ 11\ldots110 & = & ? \\ 11\ldots111 & = & ? = \text{maior inteiro sem sinal com } w \text{ bits} \\ 100\ldots000 & = & 2^w \end{array}$$

O próximo número na sequência, que não cabe em w bits, é

$$(1\underbrace{00\ldots000}_{w \text{ bits}})_2 = 0 \cdot 2^0 + 0 \cdot 2^1 + \ldots + 0 \cdot 2^{w-1} + 1 \cdot 2^w = 2^w$$

Representando números: inteiros sem sinal

Inteiros sem sinal em palavras binárias com w bits.

Palavra		Decimal
$00\ldots000$	$=$	0
$00\ldots001$	$=$	1
$00\ldots010$	$=$	2
	$\vdots$	
$11\ldots110$	$=$	$2^w - 2$
$11\ldots111$	$=$	$2^w - 1 =$ maior inteiro sem sinal com w bits
$100\ldots000$	$=$	2^w

O próximo número na sequência, que não cabe em w bits, é

$$(1\underbrace{00\ldots000}_{w \text{ bits}})_2 = 0 \cdot 2^0 + 0 \cdot 2^1 + \ldots + 0 \cdot 2^{w-1} + 1 \cdot 2^w = 2^w$$

Representando números: inteiros com sinal

- Precisamos reservar espaço na palavra para representar, além dos algarismos do número, alguma informação sobre o sinal.

Representando números: inteiros com sinal

- Precisamos reservar espaço na palavra para representar, além dos algarismos do número, alguma informação sobre o sinal.
- Como só existem duas possibilidades para o sinal, podemos usar um dos bits para representar o sinal.

Representando números: inteiros com sinal

- Precisamos reservar espaço na palavra para representar, além dos algarismos do número, alguma informação sobre o sinal.
- Como só existem duas possibilidades para o sinal, podemos usar um dos bits para representar o sinal. Sugestão:
  - sinal $+$: bit de sinal 0
  - sinal $-$: bit de sinal 1

$$\underbrace{\overbrace{\boxed{\;s\;}}^{\textit{sinal}}\overbrace{\boxed{a_{w-2}}\boxed{a_{w-1}}\cdots\boxed{a_2}\boxed{a_1}\boxed{a_0}}^{\textit{magnitude}}}_{w\ \text{bits}}$$

- Esta representação é conhecida como **sinal-magnitude**.

Representando números: inteiros com sinal

- Precisamos reservar espaço na palavra para representar, além dos algarismos do número, alguma informação sobre o sinal.
- Como só existem duas possibilidades para o sinal, podemos usar um dos bits para representar o sinal. Sugestão:
  - sinal $+$: bit de sinal 0
  - sinal $-$: bit de sinal 1

$$\underbrace{\overbrace{\boxed{\;s\;\;\;}}^{\textit{sinal}}\overbrace{\boxed{a_{w-2}}\boxed{a_{w-1}}\cdots\boxed{a_2}\boxed{a_1}\boxed{a_0}}^{\textit{magnitude}}}_{w\ \text{bits}}$$

- Esta representação é conhecida como **sinal-magnitude**.
- **Ex.:** inteiros representados em sinal-magnitude com 3 bits

000	001	010	011	100	101	110	111
+0	+1	+2	+3	−0	−1	−2	−3

Menor número: $\boxed{1}\boxed{1}\boxed{1}\ldots\boxed{1}\boxed{1} =$

Representação sinal-magnitude

Menor número: $\boxed{1}\boxed{1}\boxed{1}\ldots\boxed{1}\boxed{1} = -(\underbrace{11\ldots11}_{w-1 \text{ uns}})_2 =$

Representação sinal-magnitude

Menor número: $\boxed{1}\boxed{1}\boxed{1}\ldots\boxed{1}\boxed{1} = -(\underbrace{11\ldots11}_{w-1 \text{ uns}})_2 = -(1\underbrace{00\ldots00}_{w-1 \text{ zeros}}-1)_2$
$= -(2^{w-1} - 1) = -2^{w-1} + 1$

Representação sinal-magnitude

Menor número: $\boxed{1}\boxed{1}\boxed{1}\ldots\boxed{1}\boxed{1} = -(\underbrace{11\ldots11}_{w-1 \text{ uns}})_2 = -(1\underbrace{00\ldots00}_{w-1 \text{ zeros}}-1)_2$
$= -(2^{w-1} - 1) = -2^{w-1} + 1$

Maior número: $\boxed{0}\boxed{1}\boxed{1}\ldots\boxed{1}\boxed{1} = +2^{w-1} - 1$

Representação sinal-magnitude

Menor número: $\boxed{1}\boxed{1}\boxed{1}\ldots\boxed{1}\boxed{1} = -(\underbrace{11\ldots11}_{w-1 \text{ uns}})_2 = -(1\underbrace{00\ldots00}_{w-1 \text{ zeros}}-1)_2$
$= -(2^{w-1} - 1) = -2^{w-1} + 1$

Maior número: $\boxed{0}\boxed{1}\boxed{1}\ldots\boxed{1}\boxed{1} = +2^{w-1} - 1$

Vantagens

- simples de entender
- simples de implementar

Desvantagens

- zero tem duas representações: $\boxed{0}\boxed{0}\ldots\boxed{0} = +0$ e $\boxed{1}\boxed{0}\ldots\boxed{0} = -0$
- complica a aritmética: é necessário tratar o sinal separadamente na hora de fazer as contas de soma e subtração.

Representação em complemento de 2

- vimos que, uma maneira de fazer subtrações na forma $A - B$ era tomar o complemento a dois $\overline{B} + 1$ e fazer a soma $A + (\overline{B} + 1)$
- note que se $-B$ é um número negativo, então $-B = 0 - B$
- suponha que estamos representando todos os números positivos em palavras binárias de tamanho w na forma:

$$\boxed{0}\boxed{a_{w-2}}\boxed{a_{w-3}}\cdots\boxed{a_1}\boxed{a_0}$$

Representação em complemento de 2

- vimos que, uma maneira de fazer subtrações na forma $A - B$ era tomar o complemento a dois $\overline{B} + 1$ e fazer a soma $A + (\overline{B} + 1)$
- note que se $-B$ é um número negativo, então $-B = 0 - B$
- suponha que estamos representando todos os números positivos em palavras binárias de tamanho w na forma:

$$\boxed{0}\boxed{a_{w-2}}\boxed{a_{w-3}}\cdots\boxed{a_1}\boxed{a_0}$$

- **Ex.:** Calcule $0 - (11)_{10}$ usando complemento de 2 em palavras com 5 bits, sendo que o primeiro bit 0 representa sinal positivo.

$$0 = \boxed{0}\underbrace{\boxed{0}\boxed{0}\boxed{0}\boxed{0}}_{0000_2=0} \qquad 11_{10} = \boxed{0}\underbrace{\boxed{1}\boxed{0}\boxed{1}\boxed{1}}_{1011_2=11_{10}}$$

Representação em complemento de 2

- vimos que, uma maneira de fazer subtrações na forma $A - B$ era tomar o complemento a dois $\overline{B} + 1$ e fazer a soma $A + (\overline{B} + 1)$
- note que se $-B$ é um número negativo, então $-B = 0 - B$
- suponha que estamos representando todos os números positivos em palavras binárias de tamanho w na forma:

$$\boxed{0}\boxed{a_{w-2}}\boxed{a_{w-3}}\cdots\boxed{a_1}\boxed{a_0}$$

- **Ex.:** Calcule $0 - (11)_{10}$ usando complemento de 2 em palavras com 5 bits, sendo que o primeiro bit 0 representa sinal positivo.

$$0 = \boxed{0}\underbrace{\boxed{0}\boxed{0}\boxed{0}\boxed{0}}_{0000_2=0} \qquad 11_{10} = \boxed{0}\underbrace{\boxed{1}\boxed{0}\boxed{1}\boxed{1}}_{1011_2=11_{10}}$$

Complemento a dois de $01011 = 10100 + 1 = 10101$

Representação em complemento de 2

- vimos que, uma maneira de fazer subtrações na forma $A - B$ era tomar o complemento a dois $\overline{B} + 1$ e fazer a soma $A + (\overline{B} + 1)$
- note que se $-B$ é um número negativo, então $-B = 0 - B$
- suponha que estamos representando todos os números positivos em palavras binárias de tamanho w na forma:

$$\boxed{0}\boxed{a_{w-2}}\boxed{a_{w-3}}\cdots\boxed{a_1}\boxed{a_0}$$

- **Ex.:** Calcule $0 - (11)_{10}$ usando complemento de 2 em palavras com 5 bits, sendo que o primeiro bit 0 representa sinal positivo.

$$0 = \boxed{0}\underbrace{\boxed{0}\boxed{0}\boxed{0}\boxed{0}}_{0000_2=0} \qquad 11_{10} = \boxed{0}\underbrace{\boxed{1}\boxed{0}\boxed{1}\boxed{1}}_{1011_2=11_{10}}$$

Complemento a dois de $01011 = 10100 + 1 = 10101$

$-1011_2 = 0 - 1011_2 = 0 + 10101_2 = \boxed{1}\boxed{0}\boxed{1}\boxed{0}\boxed{1}$, bit de sinal 1

Representação em complemento de 2

Representação de inteiros com sinal em complemento de 2

- Números positivos

$$\boxed{0}\underbrace{\boxed{a_{w-2}}\boxed{a_{w-3}}\dots\boxed{a_1}\boxed{a_0}}_{w-1\text{ bits}} = +(a_{w-2}a_{w-3}\dots a_1a_0)_2$$

- Números negativos

$$\boxed{1}\underbrace{\boxed{a_{w-2}}\boxed{a_{w-3}}\dots\boxed{a_1}\boxed{a_0}}_{w-1\text{ bits}} = -(\overline{a_{w-2}a_{w-3}\dots a_1a_0} + 1)_2$$

- **Ex.:** a que número corresponde a palavra $\boxed{1}\boxed{0}\boxed{1}\boxed{1}\boxed{1}\boxed{0}\boxed{0}$?

Representação em complemento de 2

Representação de inteiros com sinal em complemento de 2

- Números positivos

$$\boxed{0}\underbrace{\boxed{a_{w-2}}\boxed{a_{w-3}}\dots\boxed{a_1}\boxed{a_0}}_{w-1 \text{ bits}} = +(a_{w-2}a_{w-3}\dots a_1 a_0)_2$$

- Números negativos

$$\boxed{1}\underbrace{\boxed{a_{w-2}}\boxed{a_{w-3}}\dots\boxed{a_1}\boxed{a_0}}_{w-1 \text{ bits}} = -(\overline{a_{w-2}a_{w-3}\dots a_1 a_0} + 1)_2$$

- **Ex.:** a que número corresponde a palavra | 1 | 0 | 1 | 1 | 1 | 0 | 0 | ?
  Bit de sinal 1 = número negativo.

Representação em complemento de 2

Representação de inteiros com sinal em complemento de 2

- Números positivos

$$\boxed{0}\underbrace{\boxed{a_{w-2}}\boxed{a_{w-3}}\ldots\boxed{a_1}\boxed{a_0}}_{w-1 \text{ bits}} = +(a_{w-2}a_{w-3}\ldots a_1a_0)_2$$

- Números negativos

$$\boxed{1}\underbrace{\boxed{a_{w-2}}\boxed{a_{w-3}}\ldots\boxed{a_1}\boxed{a_0}}_{w-1 \text{ bits}} = -(\overline{a_{w-2}a_{w-3}\ldots a_1a_0} + 1)_2$$

- **Ex.:** a que número corresponde a palavra $\boxed{1}\boxed{0}\boxed{1}\boxed{1}\boxed{1}\boxed{0}\boxed{0}$?
  Bit de sinal 1 = número negativo.
  $\overline{011100} + 1 = 100011 + 1 = 100100 = 36_{10}$

Representação de inteiros com sinal em complemento de 2

- Números positivos

$$\underbrace{\boxed{0}\,\boxed{a_{w-2}}\boxed{a_{w-3}}\ldots\boxed{a_1}\boxed{a_0}}_{w-1 \text{ bits}} = +(a_{w-2}a_{w-3}\ldots a_1 a_0)_2$$

- Números negativos

$$\boxed{1}\,\underbrace{\boxed{a_{w-2}}\boxed{a_{w-3}}\ldots\boxed{a_1}\boxed{a_0}}_{w-1 \text{ bits}} = -(\overline{a_{w-2}a_{w-3}\ldots a_1 a_0} + 1)_2$$

- **Ex.:** a que número corresponde a palavra $\boxed{1}\boxed{0}\boxed{1}\boxed{1}\boxed{1}\boxed{0}\boxed{0}$?
  Bit de sinal 1 = número negativo.
  $\overline{011100} + 1 = 100011 + 1 = 100100 = 36_{10}$
  $\boxed{1}\boxed{0}\boxed{1}\boxed{1}\boxed{1}\boxed{0}\boxed{0} = -36_{10}$

Representação em complemento de 2

Inteiros representados em complemento de dois em palavras de 3 bits:

$011 = +3_{10}$
$010 = +2_{10}$
$001 = +1_{10}$
$000 = 0_{10}$

Representação em complemento de 2

Inteiros representados em complemento de dois em palavras de 3 bits:

$011 = +3_{10}$

$010 = +2_{10}$

$001 = +1_{10}$

$000 = 0_{10}$

$111 = \qquad -(\overline{11} + 1)_2 = -(00 + 1)_2$

Inteiros representados em complemento de dois em palavras de 3 bits:

$\texttt{011} = +3_{10}$

$\texttt{010} = +2_{10}$

$\texttt{001} = +1_{10}$

$\texttt{000} = 0_{10}$

$\texttt{111} = -1_{10} \quad = -(\overline{11}+1)_2 = -(00+1)_2$

Representação em complemento de 2

Inteiros representados em complemento de dois em palavras de 3 bits:

$\texttt{011} = +3_{10}$

$\texttt{010} = +2_{10}$

$\texttt{001} = +1_{10}$

$\texttt{000} = 0_{10}$

$\texttt{111} = -1_{10} \quad = -(\overline{11}+1)_2 = -(00+1)_2$

$\texttt{110} = \quad -(\overline{10}+1)_2 = -(01+1)_2$

Inteiros representados em complemento de dois em palavras de 3 bits:

$011 = +3_{10}$
$010 = +2_{10}$
$001 = +1_{10}$
$000 = 0_{10}$
$111 = -1_{10} \quad = -(\overline{11} + 1)_2 = -(00 + 1)_2$
$110 = -2_{10} \quad = -(\overline{10} + 1)_2 = -(01 + 1)_2$

Inteiros representados em complemento de dois em palavras de 3 bits:

$011 = +3_{10}$

$010 = +2_{10}$

$001 = +1_{10}$

$000 = 0_{10}$

$111 = -1_{10} \quad = -(\overline{11} + 1)_2 = -(00 + 1)_2$

$110 = -2_{10} \quad = -(\overline{10} + 1)_2 = -(01 + 1)_2$

$101 = -3_{10}$

Representação em complemento de 2

Inteiros representados em complemento de dois em palavras de 3 bits:

$011 = +3_{10}$
$010 = +2_{10}$
$001 = +1_{10}$
$000 = 0_{10}$
$111 = -1_{10} \quad = -(\overline{11} + 1)_2 = -(00 + 1)_2$
$110 = -2_{10} \quad = -(\overline{10} + 1)_2 = -(01 + 1)_2$
$101 = -3_{10}$
$100 = -4_{10}$

Representação em complemento de 2

Inteiros representados em complemento de dois em palavras de 3 bits:

$011 = +3_{10}$
$010 = +2_{10}$
$001 = +1_{10}$
$000 = 0_{10}$
$111 = -1_{10} \quad = -(\overline{11}+1)_2 = -(00+1)_2$
$110 = -2_{10} \quad = -(\overline{10}+1)_2 = -(01+1)_2$
$101 = -3_{10}$
$100 = -4_{10}$

- note que o intervalo de representação não é simétrico
- como só há uma representação para 0, é possível representar um inteiro negativo a mais
- somas/subtrações com esta representação são simples!
  $1 + (-3) = \boxed{0}\boxed{0}\boxed{1} + \boxed{1}\boxed{0}\boxed{1} = \boxed{1}\boxed{1}\boxed{0} = -2$

Representação em complemento de 2

Maior número: $\boxed{0}\boxed{1}\boxed{1}\ldots\boxed{1}\boxed{1} = +2^{w-1} - 1$ (como sinal-magnitude)

Representação em complemento de 2

Maior número: $\boxed{0}\boxed{1}\boxed{1}\ldots\boxed{1}\boxed{1} = +2^{w-1} - 1$ (como sinal-magnitude)

Menor número: $\boxed{1}\boxed{0}\boxed{0}\ldots\boxed{0}\boxed{0} = -(\overline{00\ldots00} + 1)_2 = -(11\ldots11 + 1)$

Representação em complemento de 2

Maior número: $\boxed{0}\boxed{1}\boxed{1}\ldots\boxed{1}\boxed{1} = +2^{w-1} - 1$ (como sinal-magnitude)

Menor número: $\boxed{1}\boxed{0}\boxed{0}\ldots\boxed{0}\boxed{0} = -(\overline{00\ldots00} + 1)_2 = -(11\ldots11 + 1)$

$$= -(1\underbrace{00\ldots00}_{w-1 \text{ zeros}} - 1 + 1)$$

Representação em complemento de 2

Maior número: $\boxed{0}\boxed{1}\boxed{1}\ldots\boxed{1}\boxed{1} = +2^{w-1} - 1$ (como sinal-magnitude)

Menor número: $\boxed{1}\boxed{0}\boxed{0}\ldots\boxed{0}\boxed{0} = -(\overline{00\ldots00} + 1)_2 = -(11\ldots11 + 1)$

$$= -(1\underbrace{00\ldots00}_{w-1 \text{ zeros}} - 1 + 1) = -2^{w-1}$$

(uma unidade menor que sinal-magnitude)

Maior número: $\boxed{0}\boxed{1}\boxed{1}\ldots\boxed{1}\boxed{1} = +2^{w-1} - 1$ (como sinal-magnitude)

Menor número: $\boxed{1}\boxed{0}\boxed{0}\ldots\boxed{0}\boxed{0} = -(\overline{00\ldots00} + 1)_2 = -(11\ldots11 + 1)$

$$= -(1\underbrace{00\ldots00}_{w-1 \text{ zeros}} - 1 + 1) = -2^{w-1}$$

(uma unidade menor que sinal-magnitude)

Vantagens

- representação única para o zero
- somas e subtrações são feitas da mesma forma que para números sem sinal

Desvantagens

- não é tão intuitivo para nós (indiferente para computador)
- comparação não é tão simples: $1 = \boxed{0}\boxed{0}\boxed{1} > \boxed{1}\boxed{0}\boxed{1} - 3$

Para casa

- Pensar:
  - como converter palavras de dados de tamanhos diferentes? Ex.: de 8 para 16 bits?

Para casa

- Pensar:
  - como converter palavras de dados de tamanhos diferentes? Ex.: de 8 para 16 bits?
  - O que acontece se o resultado da soma/subtração/multiplicação de dois inteiros representados em palavras de w bits não couber em w bits? (Overflow)

Para casa

- Pensar:
  - como converter palavras de dados de tamanhos diferentes? Ex.: de 8 para 16 bits?
  - O que acontece se o resultado da soma/subtração/multiplicação de dois inteiros representados em palavras de w bits não couber em w bits? (Overflow)
  - Comportamentos distintos para representação sem sinal, sinal-magnitude e complemento de dois

Para casa

- Pensar:
  - como converter palavras de dados de tamanhos diferentes? Ex.: de 8 para 16 bits?
  - O que acontece se o resultado da soma/subtração/multiplicação de dois inteiros representados em palavras de w bits não couber em w bits? (Overflow)
  - Comportamentos distintos para representação sem sinal, sinal-magnitude e complemento de dois
- Seções do livro: 2-4, 2-5, 2-6 e 2-7