

ARQ1 _ Aula_02 - Revisão

Tema: Sistemas de Numeração e representações de dados

Sistemas de Numeração – Conversões entre bases (parte inteira e fracionária)

Parte inteira

Exemplos:

1.) Sistema decimal

$$163_{(10)} = 1 \times 10^2 + 6 \times 10^1 + 3 \times 10^0 \quad - \text{na forma canônica}$$

Para converter um valor decimal (base=10) para binário (base=2),
usar divisões sucessivas por 2 e tomar os restos na ordem inversa
em que forem calculados:

| operação | quociente | resto |
|----------|-----------|----------------|
| 163 / 2 | = 81 | + 1 (último) |
| 81 / 2 | = 40 | + 1 |
| 40 / 2 | = 20 | + 0 |
| 20 / 2 | = 10 | + 0 |
| 10 / 2 | = 5 | + 0 |
| 5 / 2 | = 2 | + 1 |
| 2 / 2 | = 1 | + 0 |
| 1 / 2 | = 0 | + 1 (primeiro) |

Sistema binário

$$1010\ 0011_{(2)} \quad - \text{número na base 2}$$

ou

| | | | | | | | | |
|-------|-------|-------|-------|-------|-------|-------|-------|--|
| 2^7 | 2^6 | 2^5 | 2^4 | 2^3 | 2^2 | 2^1 | 2^0 | - potências da base 2 |
| 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 | - valor decimal da potência na base 10 |
| 1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | - coeficientes |

- 2.) Para converter um valor binário (base=2) para decimal (base=10), usar a soma dos produtos de cada algarismo pela potência da base equivalente à posição:

Sistema binário

$1010\ 0011_{(2)}$ - número na base 2

Sistema decimal

$1 \times 2^7 + 0 \times 2^6 + 1 \times 2^5 + 0 \times 2^4 + 0 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$ - forma canônica

$128 + 0 + 32 + 0 + 0 + 0 + 2 + 1 = 163_{(10)}$

Os procedimentos semelhantes servirão para converter de decimal para outras bases.

Para converter um valor decimal para a base 4 (quaternário):

operação quociente resto

$163 / 4 = 40 + 3$ (último)

$40 / 4 = 10 + 0$

$10 / 4 = 2 + 2$

$2 / 4 = 0 + 2$ (primeiro)

Sistema quaternário

$2203_{(4)}$ - número na base 4

Para converter um valor decimal para a base 8 (octal):

operação quociente resto

$163 / 8 = 20 + 3$ (último)

$20 / 8 = 2 + 4$

$2 / 8 = 0 + 2$ (primeiro)

Sistema octal

$243_{(8)}$ - número na base 8

Para converter um valor decimal para a base 16 (hexadecimal):

operação quociente resto

$163 / 16 = 10 + 3$ (último)

$10 / 16 = 0 + 10$ (primeiro, substituindo pelo algarismo A=10)

Sistema hexadecimal

$A3_{(16)}$

- número na base 16

Os procedimentos semelhantes servirão para converter dessas bases para decimal.

Sistema quaternário

$$\begin{aligned} 2203_{(2)} &= 2 \times 4^3 + 2 \times 4^2 + 0 \times 4^1 + 3 \times 4^0 \\ &= 128 + 32 + 0 + 3 = 163_{(10)} \end{aligned}$$

- número na base 4 na forma canônica

Sistema octal

$$\begin{aligned} 243_{(8)} &= 2 \times 8^2 + 4 \times 8^1 + 3 \times 8^0 \\ &= 128 + 32 + 3 = 163_{(10)} \end{aligned}$$

- número na base 8 na forma canônica

Sistema hexadecimal

$$\begin{aligned} A3_{(16)} &= (A=10) \times 16^1 + 3 \times 16^0 \\ &= 160 + 3 = 163_{(10)} \end{aligned}$$

- número na base 16 forma canônica

- 3.) As bases que são potências múltiplas de outra (e apenas essas) compartilham propriedades especiais, como a possibilidade de conversões entre elas, sem passar pela base decimal:

Sistema binário (base=2) para quaternário (base=4=2²):

1010 0011₍₂₎ = [10][10] [00][11]₍₄₎ = 2203₍₄₎ agrupar de 2 em 2 para a esquerda

Sistema binário (base=2) para quaternário (base=8=2³):

1010 0011₍₂₎ = [010][100][011]₍₈₎ = 243₍₈₎ agrupar de 3 em 3 para a esquerda

OBS: Neste caso, completar com zeros para formar os grupos.

Sistema binário (base=2) para quaternário (base=16=2⁴):

1010 0011₍₂₎ = [1010] [0011]₍₁₆₎ = A3₍₁₆₎ e A=10 agrupar de 4 em 4 para a esquerda

ou usar uma tabela com as principais equivalências entre essas bases de numeração.

| 10 | 2 | 4 | 8 | 16 |
|----|-----------|-------|-----|----|
| 00 | 0000 0000 | 00 00 | 000 | 00 |
| 01 | 0000 0001 | 00 01 | 001 | 01 |
| 02 | 0000 0010 | 00 02 | 002 | 02 |
| 03 | 0000 0011 | 00 03 | 003 | 03 |
| 04 | 0000 0100 | 00 10 | 004 | 04 |
| 05 | 0000 0101 | 00 11 | 005 | 05 |
| 06 | 0000 0110 | 00 12 | 006 | 06 |
| 07 | 0000 0111 | 00 13 | 007 | 07 |
| 08 | 0000 1000 | 00 20 | 010 | 08 |
| 09 | 0000 1001 | 00 21 | 011 | 09 |
| 10 | 0000 1010 | 00 22 | 012 | 0A |
| 11 | 0000 1011 | 00 23 | 013 | 0B |
| 12 | 0000 1100 | 00 30 | 014 | 0C |
| 13 | 0000 1101 | 00 31 | 015 | 0D |
| 14 | 0000 1110 | 00 32 | 016 | 0E |
| 15 | 0000 1111 | 00 33 | 017 | 0F |

Parte fracionária

Exemplos:

1.) Sistema decimal

$$0,6875_{(10)} = 6 \times 10^{-1} + 8 \times 10^{-2} + 7 \times 10^{-3} + 5 \times 10^{-4} \quad - \text{na forma canônica}$$

Para converter a parte fracionária de um valor decimal (base=10) para binário (base=2), usar multiplicações sucessivas por 2 e tomar as partes inteiras na mesma ordem em que forem calculados, prosseguindo com a parte fracionária restante.

| operação | produto | parte inteira | parte fracionária | binário |
|-----------------------|---------|---------------|-------------------|-----------------|
| $0,6875 * 2 = 1,3750$ | $=$ | 1 | ,3750 | 0,1 (primeiro) |
| $0,3750 * 2 = 0,7500$ | $=$ | 0 | ,7500 | 0,10 |
| $0,7500 * 2 = 1,5000$ | $=$ | 1 | ,5000 | 0,101 |
| $0,5000 * 2 = 1,0000$ | $=$ | 1 | ,0000 | 0,1011 (último) |

Parar, se a parte fracionária se tornar igual a zero.

Sistema binário

$$0,1011_{(2)} \quad - \text{número na base 2}$$

ou

| 2^{-1} | 2^{-2} | 2^{-3} | 2^{-4} | |
|----------|----------|----------|----------|--|
| 0,5 | 0,25 | 0,125 | 0,0625 | - potências negativas da base 2 |
| 0, 1 | 0 | 1 | 1 | - valor decimal da potência na base 10 |
| | | | | - coeficientes |

Caso a parte fracionária não se tornar igual a zero dentro de certo número de operações, parar quando for alcançada a precisão desejada ou se esgotar a quantidade de casas disponíveis. Também podem surgir dízimas, periódicas ou não.

| operação | produto | parte inteira | parte fracionária | binário |
|-------------------|---------|---------------|-------------------|-------------------------|
| $0,69 * 2 = 1,38$ | $=$ | 1 | ,38 | 0,1 (primeiro) |
| $0,38 * 2 = 0,76$ | $=$ | 0 | ,76 | 0,10 |
| $0,76 * 2 = 1,52$ | $=$ | 1 | ,52 | 0,101 |
| $0,52 * 2 = 1,04$ | $=$ | 1 | ,04 | 0,1011 |
| $0,04 * 2 = 0,08$ | $=$ | 0 | ,08 | 0,10110 |
| $0,08 * 2 = 0,16$ | $=$ | 0 | ,16 | 0,101100 |
| $0,16 * 2 = 0,32$ | $=$ | 0 | ,32 | 0,1011000 |
| $0,32 * 2 = 0,64$ | $=$ | 0 | ,64 | 0,10110000 |
| $0,64 * 2 = 1,28$ | $=$ | 1 | ,28 | 0,101100001 |
| $0,28 * 2 = 0,56$ | $=$ | 0 | ,56 | 0,1011000010 |
| $0,56 * 2 = 1,02$ | $=$ | 1 | ,02 | 0,10110000101 |
| $0,02 * 2 = 0,04$ | $=$ | 0 | ,04 | 0,101100000010 (dízima) |

Para converter um valor decimal para a base 4 (quaternário):

| operação | produto | parte inteira | parte fracionária | quaternário |
|-----------------------|---------|---------------|-------------------|----------------|
| $0,6875 * 4 = 2,7500$ | $=$ | 2 | ,7500 | 0,2 (primeiro) |
| $0,7500 * 4 = 3,0000$ | $=$ | 3 | ,0000 | 0,23 (último) |

Sistema quaternário

$0,23_{(4)}$ - número na base 4

Por agrupamento do binário equivalente
e substituição do valor binário por dígitos dessa base:

$0,1011_{(2)} = 0, [10] [11]_{(4)} = 0,23_{(4)}$ - agrupar de 2 em 2 para a direita

Para converter um valor decimal para a base 8 (octal):

| operação | produto | parte inteira | parte fracionária | octal |
|-----------------------|---------|---------------|-------------------|----------------|
| $0,6875 * 8 = 2,7500$ | $=$ | 5 | ,7500 | 0,5 (primeiro) |
| $0,7500 * 8 = 4,0000$ | $=$ | 4 | ,0000 | 0,4 (último) |

Sistema octal

$0,54_{(8)}$ - número na base 8

Por agrupamento do binário equivalente e completando com zeros (0), se necessário,
e substituição do valor binário por dígitos dessa base:

$0,1011_{(2)} = 0, [101] [100]_{(8)} = 0,54_{(8)}$ - agrupar de 3 em 3 para a direita

Para converter um valor decimal para a base 16 (hexadecimal):

| operação | produto | parte inteira | parte fracionária | hexadecimal |
|------------------------|---------|---------------|-------------------|-------------------------|
| $0,6875 * 16 = 2,7500$ | $=$ | 11 | ,0000 | 0,B (primeiro e último) |

Sistema hexadecimal

$0,B_{(16)}$ - número na base 16

Por agrupamento do binário equivalente
e substituição do valor binário por dígitos dessa base:

$0,1011_{(2)} = 0, [1011]_{(16)} = 0,B_{(16)}$ - agrupar de 4 em 4 para a direita

- 2.) Para converter um valor fracionário em binário (base=2) para decimal (base=10), usar a soma dos produtos de cada algarismo pela potência negativa da base equivalente à posição:

Sistema binário

0,1011₍₂₎

- número na base 2

Sistema decimal

$1 \times 2^{-1} + 0 \times 2^{-2} + 1 \times 2^{-3} + 1 \times 2^{-4}$ - forma canônica

$1/2^1 + 0 + 1/2^3 + 1/2^4$

$1/2 + 0 + 1/8 + 1/16 = (8+2+1)/16$

$0,5 + 0 + 0,125 + 0,0625 = 0,6875_{(10)}$

Para converter um valor da base 4 (quaternário) para decimal:

Sistema quaternário

0,23₍₄₎

- número na base 4

Sistema decimal

$2 \times 4^{-1} + 3 \times 4^{-2} + 0 \times 4^{-3} + 0 \times 4^{-4}$ - forma canônica

$2/4^1 + 3/4^2 + 0/4^3 + 0/4^4$

$2/4 + 3/16 + 0/64 + 0/256 = (8+3)/16$

$0,5 + 0,1875 + 0 + 0 = 0,6875_{(10)}$

Para converter um valor da base 8 (octal) para decimal:

Sistema octal

0,54₍₈₎

- número na base 8

Sistema decimal

$5 \times 8^{-1} + 4 \times 8^{-2} + 0 \times 8^{-3} + 0 \times 8^{-4}$ - forma canônica

$5/8^1 + 4/8^2 + 0/8^3 + 0/8^4$

$5/8 + 4/64 + 0/512 + 0/4096 = (40+4)/64$

$0,625 + 0,0625 + 0 + 0 = 0,6875_{(10)}$

Para converter um valor da base 16 (hexadecimal) para decimal:

Sistema hexadecimal

0,B₍₁₆₎

- número na base 16

Sistema decimal

$11 \times 16^{-1} + 0 \times 16^{-2} + 0 \times 16^{-3} + 0 \times 16^{-4}$ - forma canônica

$11/16^1 + 0/16^2 + 0/16^3 + 0/16^4$

$11/16 + 0/256 + 0/4096 + 0/65536 = (11)/16$

$0,6875 + 0 + 0 + 0 = 0,6875_{(10)}$

Representações de potências de 2.

| x | 2^x | X₍₁₀₎ | X₍₂₎ | X₍₄₎ | X₍₈₎ | X₍₁₆₎ |
|----------|----------------------|-------------------------|------------------------|------------------------|------------------------|-------------------------|
| 0 | 2 ⁰ | 1 | 1 | 1 | 1 | 1 |
| 1 | 2 ¹ | 2 | 10 | 2 | 2 | 2 |
| 2 | 2 ² | 4 | 100 | 10 | 4 | 4 |
| 3 | 2 ³ | 8 | 1000 | 20 | 10 | 8 |
| 4 | 2 ⁴ | 16 | 1 0000 | 100 | 20 | 10 |
| 5 | 2 ⁵ | 32 | 10 0000 | 200 | 40 | 20 |
| 6 | 2 ⁶ | 64 | 100 0000 | 1000 | 100 | 40 |
| 7 | 2 ⁷ | 128 | 1000 0000 | 2000 | 200 | 80 |
| 8 | 2 ⁸ | 256 | 1 0000 0000 | 10000 | 400 | 100 |
| 9 | 2 ⁹ | 512 | 10 0000 0000 | 20000 | 1000 | 200 |
| 10 | 2 ¹⁰ | 1024 | 100 0000 0000 | 100000 | 2000 | 400 |
| | | | | | | |
| x | 2^x | X₍₁₀₎ | X₍₂₎ | X₍₄₎ | X₍₈₎ | X₍₁₆₎ |
| -10 | 2 ⁻¹⁰ | 0,0009765625 = 1/1024 | 0,0000000001 | 0,00001 | 0,0004 | 0,004 |
| -9 | 2 ⁻⁹ | 0,001953125 = 1/512 | 0,000000001 | 0,00002 | 0,001 | 0,008 |
| -8 | 2 ⁻⁸ | 0,00390625 = 1/256 | 0,00000001 | 0,0001 | 0,002 | 0,01 |
| -7 | 2 ⁻⁷ | 0,0078125 = 1/128 | 0,0000001 | 0,0002 | 0,004 | 0,02 |
| -6 | 2 ⁻⁶ | 0,015625 = 1/64 | 0,000001 | 0,001 | 0,01 | 0,04 |
| -5 | 2 ⁻⁵ | 0,03125 = 1/32 | 0,00001 | 0,002 | 0,02 | 0,08 |
| -4 | 2 ⁻⁴ | 0,0625 = 1/16 | 0,0001 | 0,01 | 0,04 | 0,1 |
| -3 | 2 ⁻³ | 0,125 = 1/8 | 0,001 | 0,02 | 0,1 | 0,2 |
| -2 | 2 ⁻² | 0,25 = 1/4 | 0,01 | 0,1 | 0,2 | 0,4 |
| -1 | 2 ⁻¹ | 0,5 = 1/2 | 0,1 | 0,2 | 0,4 | 0,8 |
| 0 | 2 ⁰ | 1 | 1 | 1 | 1 | 1 |

Apêndices

A1.) Equivalências entre sistemas de numeração (parte inteira com agrupamento):

| Decimal | Hexadecimal | Octal | Quaternário | Binário |
|---------|--------------------|------------------|----------------------------|---------|
| 0 | 00 = [0000] [0000] | 00 = [000] [000] | 00 = [00] [00] | 0 0000 |
| 1 | 01 = [0000] [0001] | 01 = [000] [001] | 01 = [00] [01] | 0 0001 |
| 2 | 02 = [0000] [0010] | 02 = [000] [010] | 02 = [00] [10] | 0 0010 |
| 3 | 03 = [0000] [0011] | 03 = [000] [011] | 03 = [00] [11] | 0 0011 |
| 4 | 04 = [0000] [0100] | 04 = [000] [100] | 10 = [01] [00] | 0 0100 |
| 5 | 05 = [0000] [0101] | 05 = [000] [101] | 11 = [01] [01] | 0 0101 |
| 6 | 06 = [0000] [0110] | 06 = [000] [110] | 12 = [01] [10] | 0 0110 |
| 7 | 07 = [0000] [0111] | 07 = [000] [111] | 13 = [01] [11] | 0 0111 |
| 8 | 08 = [0000] [1000] | 10 = [001] [000] | 20 = [10] [00] | 0 1000 |
| 9 | 09 = [0000] [1001] | 11 = [001] [001] | 21 = [10] [01] | 0 1001 |
| 10 | 0A = [0000] [1010] | 12 = [001] [010] | 22 = [10] [10] | 0 1010 |
| 11 | 0B = [0000] [1011] | 13 = [001] [011] | 23 = [10] [11] | 0 1011 |
| 12 | 0C = [0000] [1100] | 14 = [001] [100] | 30 = [11] [00] | 0 1100 |
| 13 | 0D = [0000] [1101] | 15 = [001] [101] | 31 = [11] [01] | 0 1101 |
| 14 | 0E = [0000] [1110] | 16 = [001] [110] | 32 = [11] [10] | 0 1110 |
| 15 | 0F = [0000] [1111] | 17 = [001] [111] | 33 = [11] [11] | 0 1111 |
| 16 | 10 = [0001] [0000] | 20 = [010] [000] | 40 = [[00][01]] [[00][00]] | 1 0000 |

A2.) Equivalências entre sistemas de numeração (parte fracionária com agrupamento):

| Decimal | Hexadecimal | Octal | Quaternário | Binário |
|--------------|-----------------|-----------------------|-------------------------------|---------|
| 00/16=0,0000 | 0,0 = 0, [0000] | 0,00 = 0, [000] [000] | 0,00 = 0, [00] [00] | 0, 0000 |
| 01/16=0,0625 | 0,1 = 0, [0001] | 0,04 = 0, [000] [001] | 0,01 = 0, [00] [01] | 0, 0001 |
| 02/16=0,1250 | 0,2 = 0, [0010] | 0,10 = 0, [000] [010] | 0,02 = 0, [00] [10] | 0, 0010 |
| 03/16=0,1875 | 0,3 = 0, [0011] | 0,14 = 0, [000] [011] | 0,03 = 0, [00] [11] | 0, 0011 |
| 04/16=0,2500 | 0,4 = 0, [0100] | 0,20 = 0, [000] [100] | 0,10 = 0, [01] [00] | 0, 0100 |
| 05/16=0,3125 | 0,5 = 0, [0101] | 0,24 = 0, [000] [101] | 0,11 = 0, [01] [01] | 0, 0101 |
| 06/16=0,3750 | 0,6 = 0, [0110] | 0,30 = 0, [000] [110] | 0,12 = 0, [01] [10] | 0, 0110 |
| 07/16=0,4375 | 0,7 = 0, [0111] | 0,34 = 0, [000] [111] | 0,13 = 0, [01] [11] | 0, 0111 |
| 08/16=0,5000 | 0,8 = 0, [1000] | 0,40 = 0, [001] [000] | 0,20 = 0, [10] [00] | 0, 1000 |
| 09/16=0,5625 | 0,9 = 0, [1001] | 0,44 = 0, [001] [001] | 0,21 = 0, [10] [01] | 0, 1001 |
| 10/16=0,6250 | 0,A = 0, [1010] | 0,50 = 0, [001] [010] | 0,22 = 0, [10] [10] | 0, 1010 |
| 11/16=0,6875 | 0,B = 0, [1011] | 0,54 = 0, [001] [011] | 0,23 = 0, [10] [11] | 0, 1011 |
| 12/16=0,7500 | 0,C = 0, [1100] | 0,60 = 0, [001] [100] | 0,30 = 0, [11] [00] | 0, 1100 |
| 13/16=0,8125 | 0,D = 0, [1101] | 0,64 = 0, [001] [101] | 0,31 = 0, [11] [01] | 0, 1101 |
| 14/16=0,8750 | 0,E = 0, [1110] | 0,70 = 0, [001] [110] | 0,32 = 0, [11] [10] | 0, 1110 |
| 15/16=0,9375 | 0,F = 0, [1111] | 0,74 = 0, [001] [111] | 0,33 = 0, [11] [11] | 0, 1111 |
| 16/16=1,0000 | 1,0 = 1, [0000] | 1,00 = 1, [000] [000] | 1,00 = [[00][01]], [[00][00]] | 1, 0000 |

Preparação

Como preparação para o início das atividades, recomendam-se

- a.) leitura prévia do resumo teórico, do detalhamento na apostila e referências recomendadas
- b.) estudo e testes dos exemplos
- c.) assistir aos seguintes vídeos:

http://www.youtube.com/watch?v=wP_bJGUOnMk

<http://www.youtube.com/watch?v=EJ8lqw67FgE>

http://www.youtube.com/watch?v=vjSKQPTkJ_o

Orientação geral:

Apresentar todas as soluções em apenas um arquivo com formato texto (.txt).

As implementações e testes dos exemplos em Verilog (.v) fornecidos como pontos de partida, também fazem parte da atividade e deverão ser entregues os códigos fontes separadamente.

As saídas de resultados, opcionalmente, poderão ser copiadas ao final do código, como comentários.

Outras formas de solução são opcionais; e, se entregues, contarão como atividades extras

(c, .java, py). Os programas com funções desenvolvidas em C, Java ou Python

(usar modelos para verificação automática de testes das respostas), se entregues, também deverão estar em arquivos separados, com o código fonte, para serem compilados e testados.

As execuções deverão, preferencialmente, serem testadas mediante uso de

entradas e saídas padrões e os dados/resultados usados para testes armazenados

em arquivos textos. Os resultados poderão ser anexados ao código, ao final, como comentários.

Planilhas, caso utilizadas, deverão ser programadas e/ou usar funções nativas.

Também deverão ser entregues em formato texto, com colunas separadas por tabulações ou no formato (.csv).

Arquivos em formato (.pdf), fotos, cópias de tela ou soluções manuscritas serão aceitas como recursos suplementares para visualização e não terão validade para fins de avaliação.

Atividades

01.) Fazer as conversões entre as bases indicadas:

- a.) $0,11001_{(2)} = X_{(10)}$
- b.) $0,01010_{(2)} = X_{(10)}$
- c.) $0,01101_{(2)} = X_{(10)}$
- d.) $1,11011_{(2)} = X_{(10)}$
- e.) $11,10110_{(2)} = X_{(10)}$

01a.) mediante uso de uma função bin2double(x)

01b.) mediante uso de um programa em Verilog

```
/*
Guia_0201
*/
module Guia_0201;
// define data
real      x = 0 ; // decimal
real power2 = 1.0; // power of 2
integer   y = 7 ; // counter
reg [7:0]  b = 8'b10100000; // binary (only fraction part, Big Endian)

// actions
initial
begin : main
$display ( "Guia_0201 - Tests" );
$display ( "x = %f" , x );
$display ( "b = 0.%8b", b );
while ( y >= 0 )
begin
power2 = power2 / 2.0;
if ( b[y] == 1 )
begin
x = x + power2;
end
$display ( "x = %f", x );
y=y-1;
end // end while
end // main

endmodule // Guia_0201
```

01c.) mediante uso de uma planilha

Exemplo:

[illegible]

02.) Fazer as conversões entre as bases indicadas:

- a.) $0,75000_{(10)} = X_{(2)}$
- b.) $0,87500_{(10)} = X_{(2)}$
- c.) $1,62500_{(10)} = X_{(2)}$
- d.) $3,12500_{(10)} = X_{(2)}$
- e.) $12,31250_{(10)} = X_{(2)}$

02a.) mediante uso de uma função double2bin(x)

02b.) mediante uso de um programa em Verilog

```
/*
  Guia_0202
*/
module Guia_0202;
// define data
  real    x = 0.75; // decimal
  integer y = 7 ;   // counter
  reg [7:0] b = 0 ; // binary

// actions
initial
begin : main
  $display ( "Guia_0202 - Tests" );
  $display ( "x = %f" , x );
  $display ( "b = 0.%8b", b );
  while ( x > 0 && y >= 0 )
  begin
    if ( x*2 >= 1 )
    begin
      b[y] = 1;
      x = x*2.0 - 1.0;
    end
    else
    begin
      b[y] = 0;
      x = x*2.0;
    end // end if
    $display ( "b = 0.%8b", b );
    y=y-1;
  end // end while
end // main

endmodule // Guia_0202
```

02c.) mediante uso de uma planilha

Exemplo:

[illegible]

03.) Fazer as conversões de base entre as bases indicadas:

DICAS: Para uma mesma base ou usar agrupamentos ou desagrupamentos.

Para conferir, compare os valores decimais equivalentes.

Completar com zeros, se necessário

a.) $0,110100_{(2)} = X_{(4)}$

b.) $0,100101_{(2)} = X_{(8)}$

c.) $0,101100_{(2)} = X_{(16)}$

d.) $1,110010_{(2)} = X_{(8)}$

e.) $1011,1101_{(2)} = X_{(16)}$

03a.) mediante uso de uma função dbin2base(base, x)

03b.) mediante uso de um programa em Verilog

```
/*
  Guia_0203
*/
module Guia_0203;
  // define data
  real    x  = 0.625;           // decimal
  reg [7:0] b = 8'b1010_0000 ; // binary

  // actions
  initial
  begin : main
    $display ( "Guia_0203 - Tests" );
    $display ( "x = %f" , x );
    $display ( "b = 0.%8b", b );
    $display ( "b = 0.%x%x (16)", b[7:4],b[3:0] );
    $display ( "b = 0.%o%o (8) ", b[7:5],b[4:2] ); // missing last group !!!
  end // main

endmodule // Guia_0203
```

03c.) mediante uso de uma planilha

Exemplo:

| $X_{(10)}$ | 2^{-1} 1/2 | 2^{-2} 1/4 | 2^{-3} 1/8 | 2^{-4} 1/16 | 2^{-5} 1/32 | 2^{-6} 1/64 | 2^{-7} 1/128 | 2^{-8} 1/256 | Σ | nova base |
|------------|----------------------|-----------------|------------------|------------------|-------------------|------------------|-------------------|-------------------|----------------------------------|--|
| 0,375 | ,0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | $1/4+1/8=0,25+0,125$ | $0,01100000_{(2)}$ |
| | ,0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | | $0, \underline{01 \ 10} \ 00 \ 00_{(2)}$ |
| | 2^1 | 2^0 | 2^1 | 2^0 | 2^1 | 2^0 | 2^1 | 2^0 | | |
| | 4^{-1} /4 | | 4^{-2} /16 | | 4^{-3} /64 | | 4^{-4} /256 | | | |
| 0,375 | (0+1) | /4 | (1*2+0) | /16 | (0+0) | /64 | (0+0) | /256 | $1/4+2/16+0+0=6/16$ | $0, \underline{1 \ 2} \ 0 \ 0_{(4)}$ |
| | ,0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | (completar com 0) | $0, \underline{011} \ 000 \ 00_{(2)}$ |
| | 2^2 | 2^1 | 2^0 | 2^2 | 2^1 | 2^0 | 2^2 | 2^1 | | |
| | 8^{-1} /8 | | | 8^{-2} /64 | | | 8^{-3} /512 | | | |
| 0,375 | (0+ $1*2+1$) | /8 | (0+ $0+0$) | /64 | (0+0) | /512 | $3/8+0+0 = 3/8$ | | $0, \underline{3} \ 0 \ 0_{(8)}$ | |
| | ,0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | | $0, \underline{0110} \ 0000_{(2)}$ |
| | 2^3 | 2^2 | 2^1 | 2^0 | 2^3 | 2^2 | 2^1 | 2^0 | | |
| | 16^{-1} /16 | | | | 16^{-2} /256 | | | | | |
| 0,375 | (0+ $1*4+$ $1*2+0$) | /16 | (0+ $0+$ $0+0$) | /256 | $6/16+0 = 6/16$ | | | | $0, \underline{6} \ 0_{(16)}$ | |

04.) Fazer as conversões de base entre as bases indicadas:

DICAS: Para uma mesma base ou usar agrupamentos ou desagrupamentos.
Para conferir, compare os valores decimais equivalentes.

a.) $0,132_{(4)} = X_{(2)}$

b.) $0,3C1_{(16)} = X_{(4)}$

c.) $0,715_{(8)} = X_{(2)}$

d.) $4,730_{(8)} = X_{(4)}$ DICA: Converter para binário primeiro, depois para a base 4.

e.) $A,D70_{(16)} = X_{(4)}$ DICA: Converter diretamente por desagrupamento.

04a.) mediante uso de uma função dbase2base(base1, base2, x)

04b.) mediante uso de um programa em Verilog

```
/*
  Guia_0204
*/
module Guia_0204;
// define data
  real    x = 0.625;          // decimal
  reg [7:0] b = 8'b1010_0000 ; // binary
  integer q [3:0];
// actions
  initial
  begin : main
    $display ( "Guia_0204 - Tests" );
    $display ( "x = %f" , x );
    $display ( "b = 0.%8b", b );
    $display ( "b = 0.%x%x (16)", b[7:4],b[3:0] );
    q[3] = b[7:6];
    q[2] = b[5:4];
    q[1] = b[3:2];
    q[0] = b[1:0];
    $display ( "b = 0.%2b %2b %2b %2b (2)", b[7:6],b[5:4],b[3:2],b[1:0] );
    $display ( "q = 0.%2d %2d %2d %2d (4)", q[3] ,q[2] ,q[1] ,q[0] );
  end // main

endmodule // Guia_0204
```

04c.) mediante uso de uma planilha

05.) Fazer as operações indicadas:

a.) $101,01_{(2)} + 11,101_{(2)} = X_{(2)}$

b.) $1001,001_{(2)} - 10,11_{(2)} = X_{(2)}$

(OBS.: Colocar operandos do mesmo tamanho)

c.) $100,111_{(2)} * 11,01_{(2)} = X_{(2)}$

(OBS.: Considerar as vírgulas, após operar)

d.) $10111,11_{(2)} / 10,101_{(2)} = X_{(2)}$

e.) $1101001_{(2)} \% 1001_{(2)} = X_{(2)}$

(OBS.: Considerar resto de divisão inteira (%))

DICA: Para conferir o resultado, converter para a base 10.

05a.) mediante uso de uma função dbinEval (bin1, "?", bin2)

05b.) mediante uso de um programa em Verilog

```
/*
  Guia_0205
*/
module Guia_0205;
// define data
  reg [7:0] a = 8'b000_1010 ; // binary
  reg [7:0] b = 8'b000_1100 ; // binary
  reg [7:0] c;
// actions
  initial
  begin : main
    $display ( "Guia_0205 - Tests" );
    $display ( "a = %8b", a );
    $display ( "b = %8b", b );
    c = a+b;
    $display ( "c = a+b = %8b", c );
    c = a-b;
    $display ( "c = a-b = %8b", c );
    c = b-a;
    $display ( "c = b-a = %8b", c );
    c = a*b;
    $display ( "c = a*b = %8b", c );
    c = b/a;
    $display ( "c = b/a = %8b", c );
  end // main

endmodule // Guia_0205
```

05c.) mediante uso de uma planilha

Modelo em Java

```
/**
    Arquitetura de Computadores I - Guia_02.
    Nome: _____ Matricula: _____

*/
public class Guia_02
{
    /**
        Contador de erros.
    */
    private static int errors = 0;

    /**
        Testar se dois valores sao iguais.
        @param x - primeiro valor
        @param y - segundo valor
    */
    public static void test_equals ( Object x, Object y )
    {
        if ( (""+x).compareTo(""+y) != 0 )
            errors = errors + 1;
    } // end test_equals ( )

    /**
        Exibir o total de erros.
        @return mensagem com o total de erros
    */
    public static String test_report ( )
    {
        return ( ""+errors );
    } // end test_report ( )

    /**
        Converter valor binario para decimal com parte fracionaria.
        @return decimal equivalente
        @param value - valor binario
    */
    public static double bin2double ( String value )
    {
        return ( -1.0 );
    } // end bin2double ( )
}
```

```

/*
    Converter valor decimal para binario com parte fracionaria.
    @return valor binario equivalente
    @param value - decimal
*/
public static String double2bin ( double value )
{
    return ( "0" );
} // end double2bin ( )

/*
    Converter valor binario com parte fracionaria para base indicada.
    @return base para a conversao
    @param value - valor binario
*/
public static String dbin2base ( String value, int base )
{
    return ( "0" );
} // end dbin2base ( )

/*
    Converter valor com parte fracionaria de uma base para outra base indicada.
    @return valor equivalente na segunda base
    @param value - valor na base1
    @param base1 - primeira base
    @param base2 - base para a conversao
*/
public static String dbase2base ( String value, int base1, int base2 )
{
    return ( "0" );
} // end dbase2base ( )

/*
    Operar valores em binartest_
    @return valor resultante da operacao, se valida
    @param value1 - primeiro valor binario
    @param op - operacao
    @param value2 - segundo valor binario
*/
public static String dbinEval ( String value1, String op, String value2 )
{
    return ( "0" );
} // end dbinEval ( )

```

```

/*
Acao principal.
*/
public static void main ( String [ ] args )
{
    System.out.println ( "Guia_02 - Java Tests" );
    System.out.println ( "Nome: _____ Matricula: _____ " );
    System.out.println ( );

    test_equals ( bin2double ( "0.11001" ), 0 );
    test_equals ( bin2double ( "0.01010" ), 0 );
    test_equals ( bin2double ( "0.01101" ), 0 );
    test_equals ( bin2double ( "1.11011" ), 0 );
    test_equals ( bin2double ( "11.10110" ), 0 );
    System.out.println ( "1. errorTotalReport = "+test_report ( ) );

    test_equals ( double2bin ( 0.75000 ), "0" );
    test_equals ( double2bin ( 0.87500 ), "0" );
    test_equals ( double2bin ( 1.62500 ), "0" );
    test_equals ( double2bin ( 3.12500 ), "0" );
    test_equals ( double2bin ( 12.31250 ), "0" );
    System.out.println ( "2. errorTotalReport = "+test_report ( ) );

    test_equals ( dbin2base ( "0.110100", 4 ), "0" );
    test_equals ( dbin2base ( "0.100101", 8 ), "0" );
    test_equals ( dbin2base ( "0.101100", 16 ), "0" );
    test_equals ( dbin2base ( "1.110010", 8 ), "0" );
    test_equals ( dbin2base ( "1011.1101", 16 ), "0" );
    System.out.println ( "3. errorTotalRepor = "+test_report ( ) );

    test_equals ( dbase2base ( "0.132", 4, 2 ), "0" );
    test_equals ( dbase2base ( "0.3C1", 4, 16 ), "0" );
    test_equals ( dbase2base ( "0.715", 8, 2 ), "0" );
    test_equals ( dbase2base ( "4.730", 8, 4 ), "0" );
    test_equals ( dbase2base ( "A.D70", 16, 4 ), "0" );
    System.out.println ( "4. errorTotalReport = "+test_report ( ) );

    test_equals ( dbinEval ( "101.01", "+", "11.101" ), "0" );
    test_equals ( dbinEval ( "1001.001", "-", "10.11" ), "0" );
    test_equals ( dbinEval ( "100,111", "*", "11.01" ), "0" );
    test_equals ( dbinEval ( "10111.11", "/", "10.101" ), "0" );
    test_equals ( dbinEval ( "1101001", "%", "1001" ), "0" );
    System.out.println ( "5. errorTotalReport = "+test_report ( ) );

    System.out.print ( "\n\nApertar ENTER para terminar." );
    System.console ( ).readLine ( );
} // end main ( )
} // end class

```

Modelo em Python

```
"""
    Arquitetura de Computadores I - Guia_02.
    Nome: _____ Matricula: _____
"""
"""
    Contador de erros.
"""
errors = 0;

"""
    Testar se dois valores sao iguais.
    @param x - primeiro valor
    @param y - segundo valor
"""
def test_equals ( x, y ):
    global errors;
    if ( str(x) != str(y) ):
        errors = errors + 1;
# end test_equals ( )

"""
    Exibir o total de erros.
    @return mensagem com o total de erros
"""
def test_report ( ):
    global errors;
    return ( ""+str(errors) );
# end test_report ( )

def bin2double ( value ):
    return ( -1.0 );
# end bin2double ( )

"""
    Converter valor decimal para binario com parte fracionaria.
    @return valor binario equivalente
    @param value - decimal
"""
def double2bin ( value ):
    return ( "0" );
# end double2bin ( )
```

```

'''
    Converter valor binario com parte fracionaria para base indicada.
    @return base para a conversao
    @param value - valor binario
'''

def dbin2base ( value, base ):
    return ( "0" );
# end dbin2base ( )

'''
    Converter valor com parte fracionaria de uma base para outra base indicada.
    @return valor equivalente na segunda base
    @param value - valor na base1
    @param base1 - primeira base
    @param base2 - base para a conversao
'''

def dbase2base ( value, base1, base2 ):
    return ( "0" );
# end dbase2base ( )

'''
    Operar valores em binartest_
    @return valor resultante da operacao, se valida
    @param value1 - primeiro valor binario
    @param op      - operacao
    @param value2 - segundo valor binario
'''

def dbinEval ( value1, op, value2 ):
    return ( "0" );
# end dbinEval ( )

```

```

'''
    Acao principal.
'''
def main ( ):
    print ( "Guia_02 - Python Tests" );
    print ( "Nome: _____ Matricula: _____ " );
    print ( );

    test_equals ( bin2double ( "0.11001" ), 0 );
    test_equals ( bin2double ( "0.01010" ), 0 );
    test_equals ( bin2double ( "0.01101" ), 0 );
    test_equals ( bin2double ( "1.11011" ), 0 );
    test_equals ( bin2double ( "11.10110" ), 0 );
    print ( "1. errorTotalReport = "+test_report ( ) );

    test_equals ( double2bin ( 0.75000 ), "0" );
    test_equals ( double2bin ( 0.87500 ), "0" );
    test_equals ( double2bin ( 1.62500 ), "0" );
    test_equals ( double2bin ( 3.12500 ), "0" );
    test_equals ( double2bin ( 12.31250 ), "0" );
    print ( "2. errorTotalReport = "+test_report ( ) );

    test_equals ( dbin2base ( "0.110100", 4 ), "0" );
    test_equals ( dbin2base ( "0.100101", 8 ), "0" );
    test_equals ( dbin2base ( "0.101100", 16 ), "0" );
    test_equals ( dbin2base ( "1.110010", 8 ), "0" );
    test_equals ( dbin2base ( "1011.1101", 16 ), "0" );
    print ( "3. errorTotalReport = "+test_report ( ) );

    test_equals ( dbase2base ( "0.132", 4, 2 ), "0" );
    test_equals ( dbase2base ( "0.3C1", 4, 16 ), "0" );
    test_equals ( dbase2base ( "0.715", 8, 2 ), "0" );
    test_equals ( dbase2base ( "4.730", 8, 4 ), "0" );
    test_equals ( dbase2base ( "A.D70", 16, 4 ), "0" );
    print ( "4. errorTotalReport = "+test_report ( ) );

    test_equals ( dbinEval ( "101.110", "+", "11.101" ), "0" );
    test_equals ( dbinEval ( "1001.001", "-", "10.11" ), "0" );
    test_equals ( dbinEval ( "100,111", "*", "11.01" ), "0" );
    test_equals ( dbinEval ( "10111.11", "/", "10.101" ), "0" );
    test_equals ( dbinEval ( "1101001", "%", "1001" ), "0" );
    print ( "5. errorTotalReport = "+test_report ( ) );

    print ( "\n\nApertar ENTER para terminar." );
    input ( );
# end main ( )

if __name__ == "__main__":
    main( );

```