**Prova 1 – 6887 Arquitetura**

**RA: 107115**

**Aluno: Matheus Augusto Schiavon Parise**

**1)a)** 27,75

27 em binário é:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 16 | 8 | 4 | 2 | 1 |
| 1 | 1 | 0 | 1 | 1 |

0,75 em binário é:

0,75 x 2 = 1,5 (1)

0,5 x 2 = 1 critério de parada

11011,11

1,101111 x 24

Bias = 2(n-1) -1

Bias = 24-1 = 15

Bias + Expoente = 15 + 4

Sinal = 0

Expoente = 10011

Mantissa = 10111100

0 10100 10111100

**1)b)** 31,875

31 em binário é

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 16 | 8 | 4 | 2 | 1 |
| 1 | 1 | 1 | 1 | 0 |

0,875 em binário é

0,875 x 2 = 1,75 (1)

0,75 x 2 = 1,5 (1)

0,5 = 1 (critério de parada)

11110,111

1,1110111 x 24

Bias = 2(n-1) -1

Bias = 24-1 = 15

Bias + Expoente = 15 + 4

Sinal = 0

Expoente = 10011

Mantissa = 11101110

0 10011 11101110

**1)c)** 1-10110-10101100

Sinal = 1

Expoente = 10110

Mantissa = 10101100

10110 em decimal é 16+4+2 = 22

22 – Bias = Valor do expoente em decimal

22-15 = 7

Como o expoente é 1 então o número é negativo

-1,10101100 x 27

-11010110,0 x 20

-11010110 x 1

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 |
| 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 |

128+64+16+4+2 = 214

Então o numero é -214 por causa do expoente

**1)d)** 0-01011-00011111

Sinal = 0

Expoente = 01011

Mantissa = 00011111

01011 em decimal é 8+2+1 = 11

11-15 = -4

1,00011111 x 2-4

100011111 x 2-12

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 |
| 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |

128+8+4+2+1 = 143

Então o número é 143 x 2-12

e) Qual é o maior e qual é o menor valor que pode ser representado utilizando esse padrão IEEE6887?

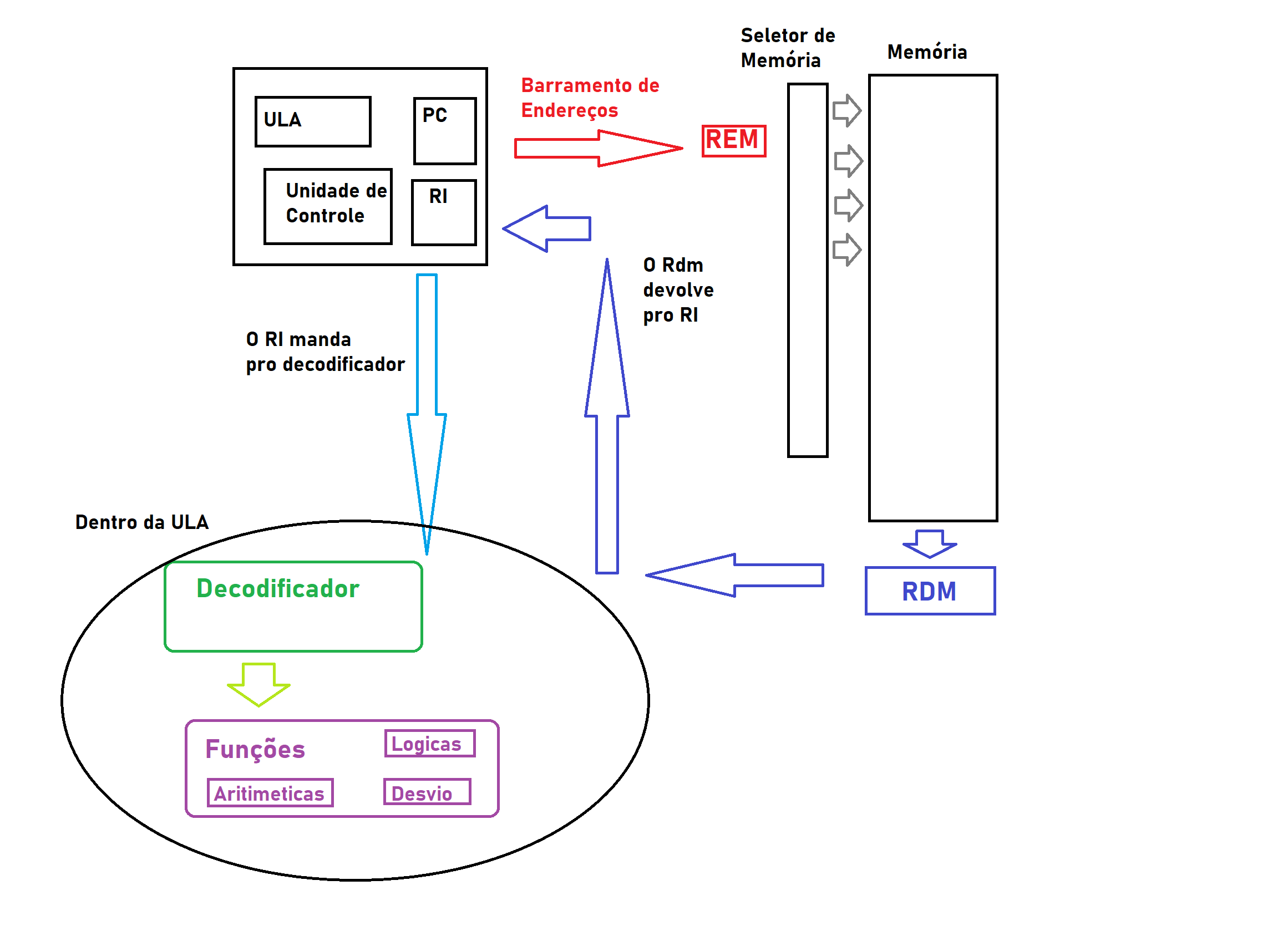
Bias = 2(n-1) -1

Bias = 2(5-1) -1 = 24-1 = 15

Então o maior valor é 15 e o menor é -14 por que o ainda existe o 0 para ser representado

**2)** A resposta está em um arquivo do circuit maker

**3)**



Inicialmente o PC que envia sua informação através do barramento de dados para REM, o REM passa sua informação para o seletor de memoria este libera as informações contidas na memoria que está na posição especificada pelo REM em um registrador chamado RDM que devolve essas informações para outro registrador chamado RI, após isso esse informações são decodificadas por um decodificador, essas informações significam um comando fazer uma função ou conjunto de funções dependendo da combinação delas, como tudo é em nível binário vai ser uma combinação de zeros e um’s elas são chamadas de OPCODE’s, cada OPCODE é único e pode corresponder a um MNEMÔNICO, porém um MNEMÔNICO pode ter vários OPCODE. Os MNEMÔNICOS representam para nós desenvolvedores o que a função vai fazer, existem vários tipos de função como por exemplo:

AND – Essa função verifica se todos os valores recebidos são iguais a 1, se forem ela retorna 1

MOV A, B – Dado os Registradores A e B, o MOV Copia o valor de A e Coloca ele em B

LOAD em A – A função load é especial pois necessita de ler 2 endereços da memoria então ela requer mais processos que as funções logicas comuns, Dado um registrador padrão previamente escolhido A, A partir do momento em que a função load é executada o valor da próxima palavra é lido e copiado para o registrador A.

Existem vários tipos de funções dentre elas logicas aritméticas e de desvio AND é logica, LOAD e MOV são de desvio e um exemplo de função aritmética seria uma soma entre duas palavras.

**4) Foi feito um circuito**

**5)** 8 linhas 4 bytes por linha, memoria principal 4096, 1024 blocos

128 blocos por linha

**a)**

4 bytes então precisam 2 bits

8 linhas precisam de 3 bits

e 128 precisa de 7 bits

TAG = 7

LINHA = 3

BYTE = 2

**b)**

4 bytes então precisam 2 bits

1024 precisa de 10 bits

TAG = 10

BYTE = 2

**c)**

4 bytes então precisam 2 bits

2 vias por conjuntos em 8 linhas = 2 bits

precisa de 10 bits

TAG = 5

CONJUNTO = 2

BYTE = 2