RONNI APARECIDO FERNANDES DE OLIVEIRA

FORUM AVALIATIVO [AVA1]:

ARQUITETURA E ORGANIZAÇÃO DE COMPUTADORES

Apucarana
2022

RONNI APARECIDO FERNANDES DE OLIVEIRA

FORUM AVALIATIVO [AVA1]:

ARQUITETURA E ORGANIZAÇÃO DE COMPUTADORES

Forum avaliativo da disciplina Arquitetura e Organização de Computadores apresentada ao Curso de Análise e Desenvolvimento de Sistemas do Centro Universitário Filadélfia - UniFil.

Apucarana
2022

Caro(a) aluno(a), em nossos estudos, discutimos sobre a aritmética digital e falamos especificamente sobre adição e subtração.

Nesse fórum, quero pedir a você que pesquise como o computador trabalha com a multiplicação e divisão digital. Apresente a teoria e exemplos de cálculo dessas operações.

Multiplicação no Sistema Binário

Antes de adentrarmos na multiplicação e na divisão dentro da ULA, é necessário entendermos como é feito a multiplicação e a divisão de números binários, matematicamente falando.

A multiplicação, segue os mesmos moldes da já conhecida multiplicação de números decimais, ou seja, multiplicador versus multiplicando. Entretanto, considerando que só existem 02 números possíveis (0 e 1) o resultado das multiplicações serão conforme abaixo:

0 x 0 = 0
0 x 1 = 0
1 x 0 = 0
1 x 1 = 1

Isso significa dizer que, se o valor do multiplicador for 0, o resultado será 0. Se o valor do multiplicador for 1, o resultado será o valor do multiplicando.

Quando temos mais de um bit no multiplicador, à cada multiplicação realizada (considerando o bit do multiplicador, da direita para a esquerda), o valor do produto desta multiplicação é expresso em linhas separadas, deslocado uma casa à esquerda a medida em que se avança na multiplicação, assim como na matemática convencional hexadecimal. Por fim, os produtos das multiplicações são somados.

Por isso, também é comum dizer que a multiplicação binária pode ser feita através da técnica de deslocamento e soma.

Vamos aos exemplos:

$\mathbf{1010}_2 \times \mathbf{10}_2 \Rightarrow \quad \mathbf{10}_{10} \times \mathbf{2}_{10} \Rightarrow \quad \mathbf{20}_{10}$

	1	0	1	0																
x			1	0																
	0	0	0	0		Convertendo o resultado binário teremos:														
1	0	1	0	+		2^4	2^3	2^2	2^1	2^0	=>	2^4+2^2	=>	16	+	4	=	20		
1	**0**	**1**	**0**	**0**	**=>**	**1**	**0**	**1**	**0**	**0**										

Agora vamos a outro exemplo:

101100_2 x **1101_2** => **44_{10}** x **13_{10}** => **572_{10}**

				1	0	1	1	0	0	
				x		1	1	0	1	
				1	0	1	1	0	0	1ª Soma
			0	0	0	0	0	0		2ª Soma
	1	1	0	1	1	0	0		+	3ª Soma
	1	0	1	1	0	0				
1	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**0**	**0**	

Convertendo o resultado para decimal:

2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0	=>	2^9	+	2^5	+	2^4	+ 2^3	+	2^2		
1	0	0	0	1	1	1	1	0	0		512		32		16	8		4		
																			572_{10}	

Os exemplos apresentados aqui estão considerando apenas números positivos. Ainda é aplicado o princípio da potência do sinal, atribuindo o valor de 0 (se o número for positivo) ou 1 (se o número for negativo) ao primeiro elemento do número binário. E ainda é aplicado o complemento 2 para os números negativos.

Do ponto de vista do processamento do computador, vamos considerar o exemplo a seguir em que a ULA realiza o seguinte processo para calcular 2 x 3.

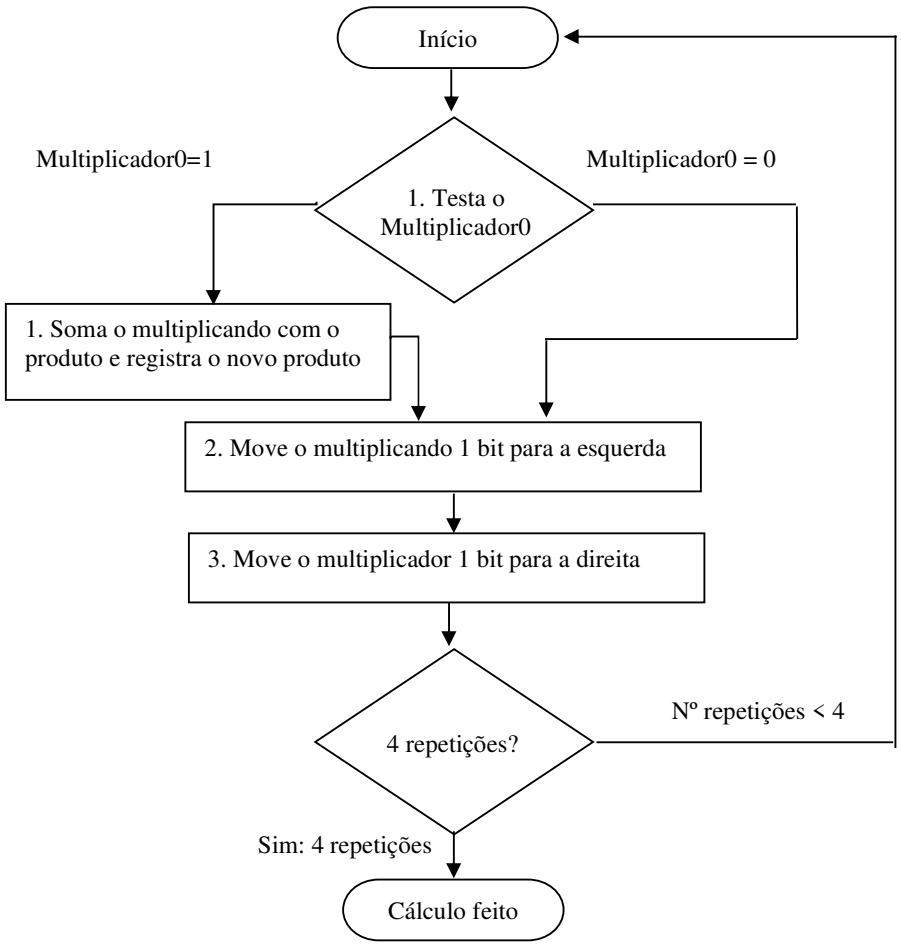
Assim, o cálculo em binário de 2x3 seria de 0010 x 0011:

			0	0	1	0	
	x		0	0	1	1	
			0	0	1	0	
		0	0	1	0		
	0	0	0	0		+	
0	0	0	0				
0	**0**	**0**	**0**	**1**	**1**	**0**	

Convertendo o resultado para decimal:

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0	=>	2^2	+	2^1	=>	$\mathbf{6}_{10}$
0	**0**	**0**	**0**	**0**	**1**	**1**	**0**		4		2		

De forma simplista, o algoritmo do computador iria realizar os seguintes passos para o cálculo:



Abaixo transcrevemos como estes dados seriam registrados na ULA:

Iteração	Passo	Multiplicador	Multiplicando	Produto
0	Valores iniciais	0011	0000 0010	0000 0000
1	a: 1 => Prod = Prod + Mcand	0011	0000 0010	0000 0010
	b: Muda para esquerda o Multiplicando	0011	0000 0100	0000 0010
	c: Muda para direita o Multiplicador	0001	0000 0100	0000 0010
2	a1: 1 => Prod = Prod + Mcand	0001	0000 0100	0000 0110
	b: Muda para esquerda o Multiplicando	0001	0000 1000	0000 0110
	c: Muda para direita o Multiplicador	0000	0000 1000	0000 0110
3	a: 0 => Sem operação	0000	0000 1000	0000 0110
	b: Muda para esquerda o Multiplicando	0000	0001 0000	0000 0110
	c: Muda para direita o Multiplicador	0000	0001 0000	0000 0110
4	a: 0 => Sem operação	0000	0001 0000	0000 0110
	b: Muda para esquerda o Multiplicando	0000	0010 0000	0000 0110
	c: Muda para direita o Multiplicador	0000	0010 0000	0000 0110

Divisão no sistema binário

Considerando o modelo matemático hexadecimal, na divisão de números binários, o princípio da divisão segue o mesmo esquema.

Consideremos dividir 80 por 4, em que dividendo (80) é maior que o divisor (4).

Primeiro comparamos estas grandezas entre dividendo e o divisor.

Neste caso, podemos fracionar o dividendo, da esquerda para a direita, e iniciarmos o cálculo quando tivermos o dividendo maior ou igual o divisor em pelo menos 1 vez (quociente).

Subtrai-se o divisor do dividendo e obtém-se o resto.

Baixa-se o próximo número, formando o próximo dividendo.

Quando o dividendo não é divisível pelo divisor, e ainda restam números a serem divididos no dividendo, o zero é apropriado ao quociente e o próximo número é baixado ao resto.

Se não houver mais números a dividir e o resto for zero, o cálculo está encerrado.

Vamos exemplificar o mesmo cálculo no esquema binário (80 / 40):

80 / 40 = 20 => **1010000_2** / **101000_2**

Transformar 80 em binário:
80 / 2 = 40 Resto 0
40 / 2 = 20 Resto 0
20 / 2 = 10 Resto 0
10 / 2 = 5 Resto 0
5 / 2 = 2 Resto 1
2 / 2 = 1 Resto 0
1 / 2 = 0 Resto 1

1010000

Transformar 40 em binário:
40 / 2 = 20 Resto 0
20 / 2 = 10 Resto 0
10 / 2 = 5 Resto 0
5 / 2 = 2 Resto 1
2 / 2 = 1 Resto 0
1 / 2 = 0 Resto 1

101000

Fazer a divisão de **1010000** / **101000**:

Faremos como na matemática convencional:

	1	**0**	**1**	**0**	**0’**	0’	0’	1	0	1	0	0
-	1	0	1	0	0			1	0			
	0	**0**	**0**	**0**	**0**	**0’**	Abaixa o próximo número (0)					
	-	0	0	0	0	0	Não dá para dividir, insere 0 no quociente					
	0	**0**	**0**	**0**	**0**	**0**	0	Não dá para dividir, resto 0				

Vamos fazer mais um cálculo:

11001$_2$ / **10**$_2$

	1	**1'**	**0**	**0**	**1**	**1**	**0**		
-	1	0				1	1	0	0
	0	**1**	**0**						
	-	1	0						
	0	**0**	**0**	**0**	**1**				
			-	0	0				
					1				

Em resumo, na divisão binária:

1. Verifica-se quantas vezes o divisor cabe no dividendo por tentativa
2. Busca o maior valor do quociente cuja sua multiplicação com o divisor não seja maior que o dividendo
3. Subtrai-se o produto do quociente versos o divisor, do dividendo
4. O resto da divisão deve ser um valor igual, no máximo, ao divisor menos 1

Referências:

3.2 - Arquitetura de Computadores - Multiplicação e Divisão de Inteiros Disponível em: https://www.youtube.com/watch?v=v4YQBWMStnc Acesso em: 26 mar. 2022

34. Arquitetura de Computadores: Unidade Lógico-Aritmética de 1 bit Disponível em: https://www.youtube.com/watch?v=5dE2tfL3orA Acesso em: 26 mar. 2022.

Arquitetura e Organização de Computadores Aula: Conversão de Bases e Aritimética Computacional Disponível em: https://www.cin.ufpe.br/~lfsc/cursos/arquiteturadecomputadores/unidade%201/aula%202%20-%20conversao%20de%20bases%20e%20aritmetica%20computacional.pdf Acesso em: 26 mar. 2022.