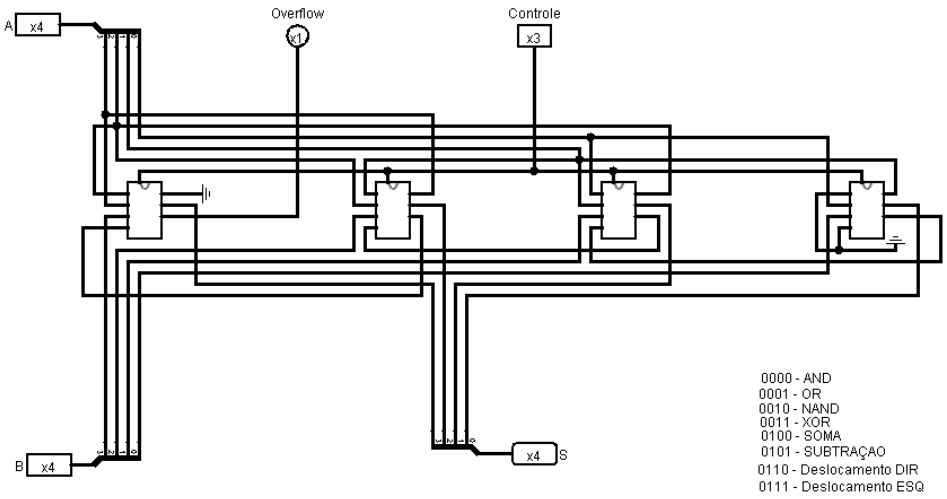
ULA desenvolvida para o trabalho da disciplina Arquitetura de Computadores.

Aluno: Gabriel Rocha de Souza.
Matricula: 474021.

PARTE 1: Visão geral.

Imagem 1: ULA



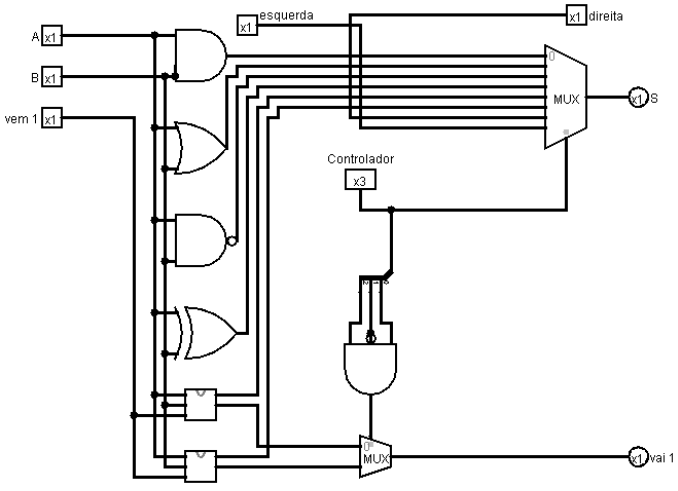
Na primeira imagem, é apresentado a parte mais “acima” do circuito, uma ULA básica de 4 bits que pode realizar 8 operações, 4 lógicas e 4 aritméticas (as mesmas citadas na própria imagem).

O circuito recebe 4 entradas, A e B, ambas de 4 bits e uma entrada de controle de 3 bits e exibe uma saída e uma flag de overflow.

O sistema é dividido em 4 **células** e cada uma delas trabalha com um bit A e um bit B, fazendo todas as operações.

PARTE 2: A célula.

Imagem 2: CÉLULA



A célula é basicamente onde tudo acontece, ela que recebe os bits de entrada, e ela que faz todas as operações. Como é possível notar, essa célula só trabalha com dois bits, por isso a necessidade de colocar 4 células na ula, cada uma trabalhando separadamente(Não é bem separadamente, pois a entrada **vem 1** e a saída **vai 1**, influenciam nos resultados uma das outras).

PARTE 3: 4 operações lógicas.

A primeira operação lógica da célula é a AND**(000)**, ela compara dois bits, caso eles sejam o digito 1 e iguais a saída será 1, em qualquer outro caso ela será zero.

A segunda operação lógica da célula é a OR**(001)**, ela compara dois bits, caso um deles seja verdadeiro(1) a saída será também verdadeira(1).

Já a terceira operação lógica é a NAND**(010),** ela nega a saída da AND e tudo que seria 0 se torna 1, no caso ela faz uma ação contraria.

No quarto e ultimo caso, temos a porta XOR**(011),** a porta XOR é um ou exclusivo, ela só e igual a 1 quando os bits A e B são diferentes.

Obs: Segue todos os exemplos na **Tabela 1**:

Teste/Exemplos:

		(000)	**(001)**	**(010)**	**(011)**
A	**B**	**A ^ B**	**A v B**	**~(A ^ B)**	**A (xor) B**
0	0	0	0	1	0
1	0	0	1	1	1
0	1	0	1	1	1
1	1	1	1	0	0

Tabela 1:

PARTE 4: As 4 operações aritméticas.

4.1: Soma**(100)**.

Imagem 3: SOMADOR

A imagem 3 representa o somador completo utilizado no circuito da ULA. O processo de criação deu inicio com a criação de um somador simples que somava dois bits, mas não considerava a terceira entrada(o vem 1), com o somador(a imagem 4 ilustra o somador simples) simples feito, foi só fazer a tabela verdade para chegar no mapa de Karnaugh e construir o circuito completo.

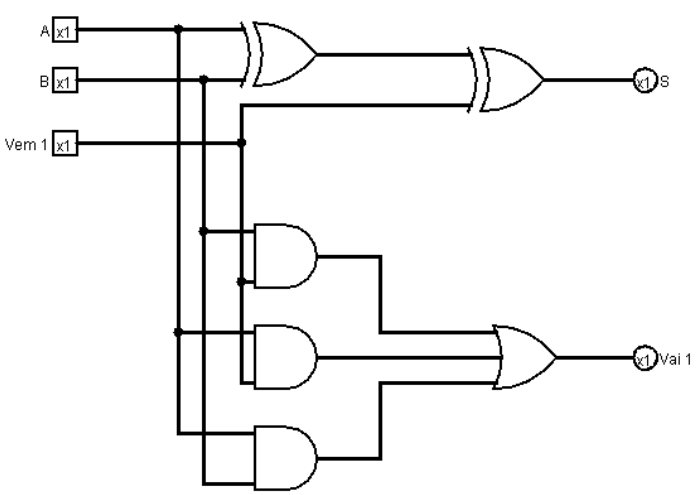
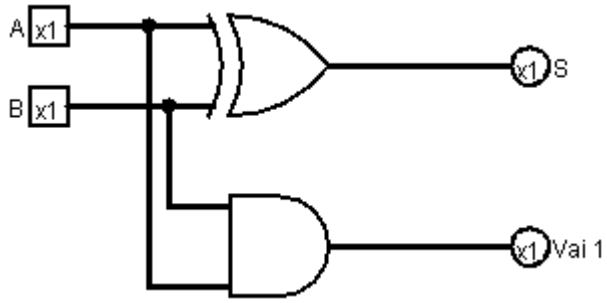


Imagem 4: SOMADOR SIMPLES



A	B	C in	A + B + C in	C out
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Tabela 2: “+” operador de soma.

	B/C in	00	01	11	10
A					
	0	0	0	1	0
	1	0	1	1	1

Tabela 3: mapa de Karnaugh.

k1 = A = 1, B = 1, C in= 0/1.
k2 = A = 1, B = 0/1, C in= 1.
k3 = A = 0/1, B = 1, C in= 1.

C out = A.B + A.C in + B.C in;

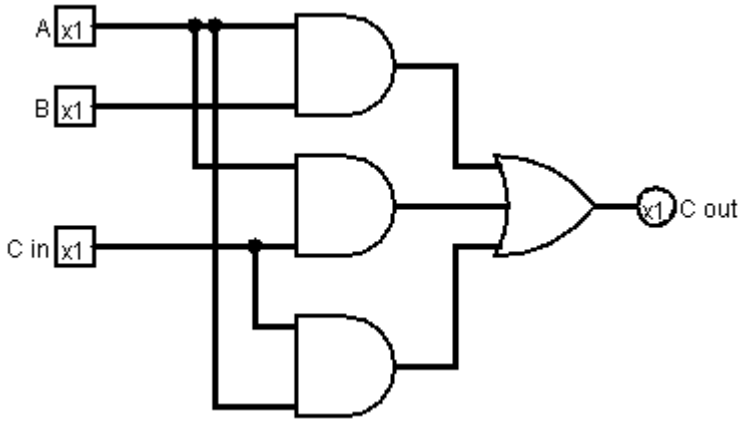


IMAGEM 5: circuito C out.

Imagem 6: circuito Subtrator completo

4.2: Subtração**(101)**.

O circuito de subtração se deu da mesma origem do da soma, o primeiro passo foi criar um circuito simples e depois fazer o mapa de Karnaugh para chegar ao circuito completo(para subtração funcionar a entrada A tem que ser maior que B).

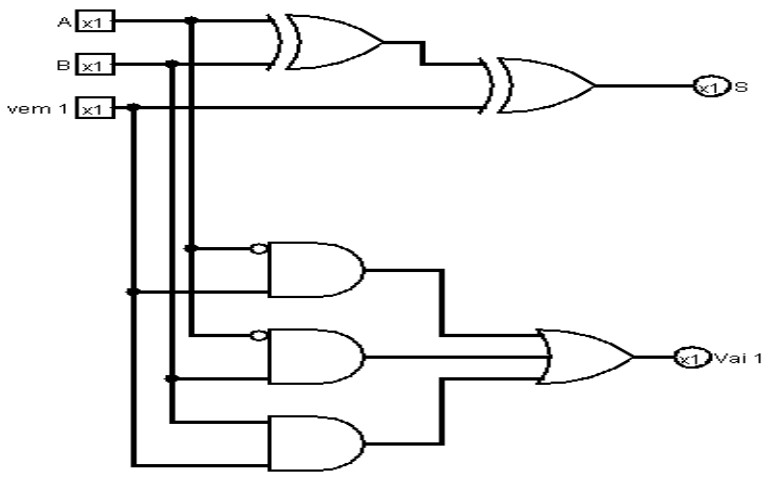
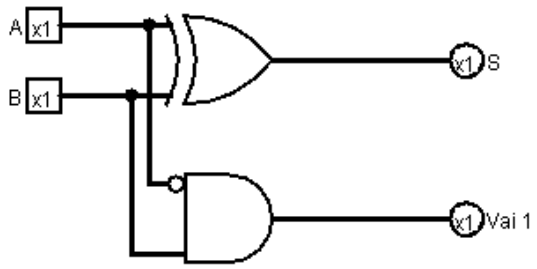


Imagem 7: circuito subtrator simples.



A	B	B in	def	B out
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

Tabela 4:

	B/B in	00	01	11	10
A					
	0	0	1	1	1
	1	0	0	1	0

Tabela 5: mapa de Karnaugh[2].

K1 = A = 0, B = 0/1, B in = 1.
k2 = A = 0, B = 1, B in= 0/1.
k3 = A = 0/1, B = 1, B in= 1.
B out = ~A.B in + ~A.B + B.B in;

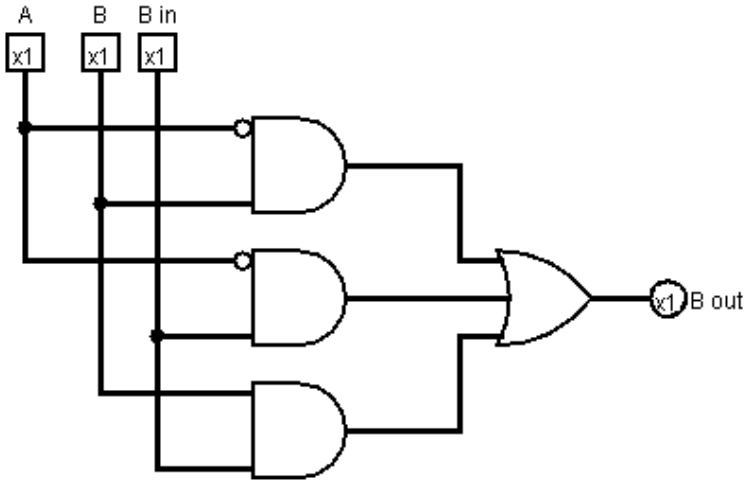


Imagem 8: circuito B out

Parte 4.3: Deslocamento Esq e Deslocamento Dir/ Funcionamento.

Os deslocamentos de bits de bits é uma coisa bem simples de se fazer, só precisa que a primeira célula(célula 0) tenha a sua entrada direita conectada na entrada **A** da célula 1 e da entrada esq da célula 2 e assim repetidamente até não ter mais como se repetir e você precisar atribuir o terra aos conectores.

Esses deslocamentos servem para “mover” os bits da entrada A. caso o deslocamento DIR seja ativado os bits serão movidos um bit a direita e caso o deslocamento **ESQ** seja ativado ocorre o caso contrario.

Funciona parecido ao vai e vem 1, que o vai 1 é ligado na saída do vem 1 da célula a esquerda.

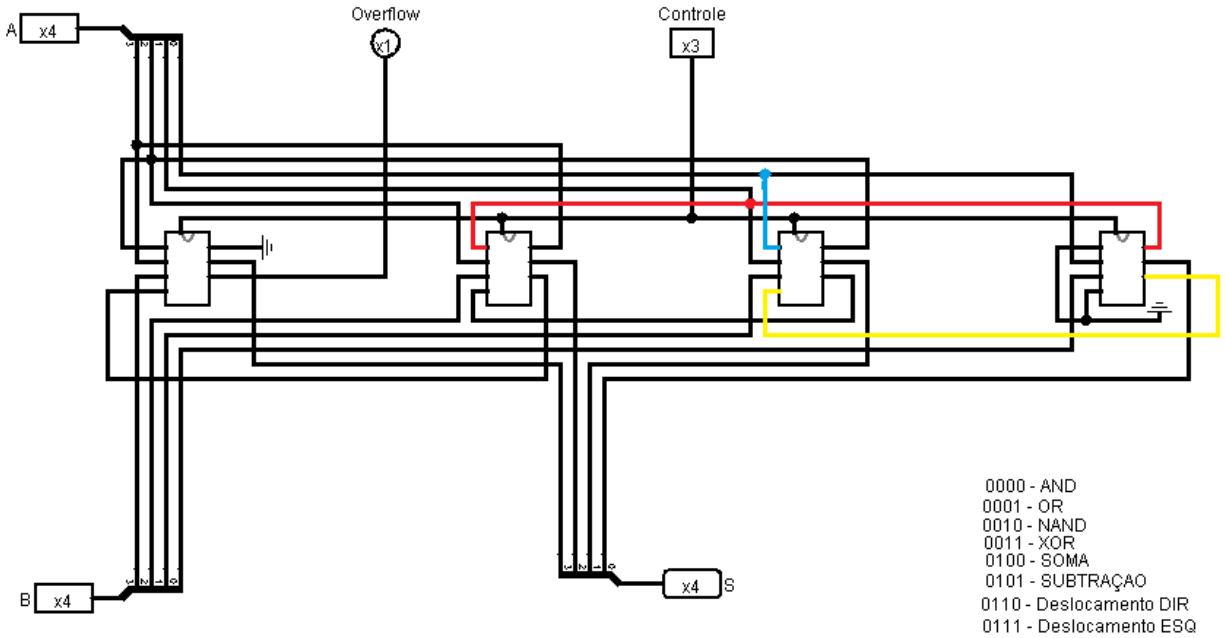


Imagem 9: conexões.

= Vai 1 conectado no vem 1.
= Deslocamento Dir conectado no A da célula 1 e na Esq da célula 2.
= Deslocamento Esq ligado no A da célula 0.

Fontes:

- Módulos Blocos: meio somador e somador completo - http://eaulas.usp.br/portal/video.action%3Bjsessionid=07ABB6CCF944623A1B8E3C8655E4C36C?idPlaylist=7725 - e-Aulas USP.

- Mapa de Karnaugh - https://www.youtube.com/watch?v=xB99jX9QMOE – Nivaldo JR.

- Criação da ULA - https://www.youtube.com/watch?v=4-UlzYdFQaY&t=8s – TADS – IFRN Natal Central.

- https://www.youtube.com/watch?v=zRX3sOtjS10&t=385s – Luciano Tavares.