

Relatório 09- Unidade Lógica Aritmética



UNIVERSIDADE
FEDERAL DO CEARÁ

Autores

João Mateus Dias do
Carmo

Matrícula: 390187

Marcelo Martins Da Silva

Matrícula: 385166

Professor: Elvis Miguel Galeas

Stancanelli

Disciplina: Circuitos Digitais

Introdução

A trabalho que será apresentado irá abordar a construção e os testes de uma Unidade Lógica e Aritmética de acordo como foi abordado em sala cada bloco funcional.

Desenvolvimento

O desenvolvimento do trabalho irá seguir a ordem da seguinte tabela verdade e seus respectivo tópicos:

Sinal de Controle	Funcionalidades
0000	AND bit a bit
0001	OR bit a bit
0010	XOR bit a bit
0011	NOT, inversão dos bits de X
0100	Deslocamento de um bit a esquerda de X
0101	Deslocamento de um bit a direita de X
0110	Deslocamento circular de um bit a esquerda de X
0111	Deslocamento circular de um bit a direita de X
1000	Soma de X com Y
1001	Subtração de X com Y
1010	Multiplicação de X com Y
1011	Teste lógico se $X < Y$
1100	Teste lógico se $X > Y$
1101	Teste lógico se $X = Y$

Ao saber a sequência podemos apresentar as características do sinal de controle e das organizações dos Bits de X e Y:

O Sinal de Controle tem a seguinte configuração:

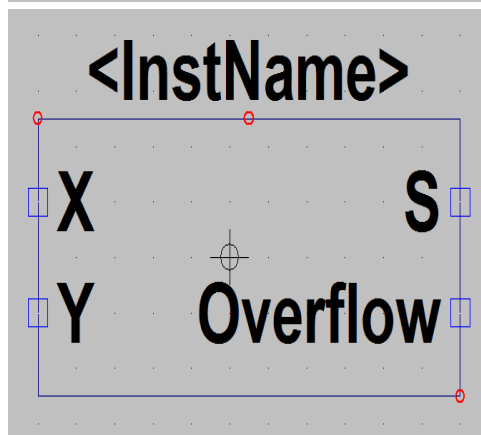
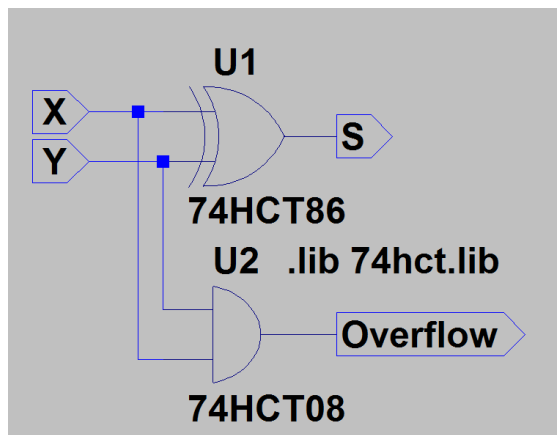
SC3	SC2	SC1	SC0
-----	-----	-----	-----

A organização dos bits de Entrada:

X3	X2	X1	X0
Y3	Y2	Y1	Y0

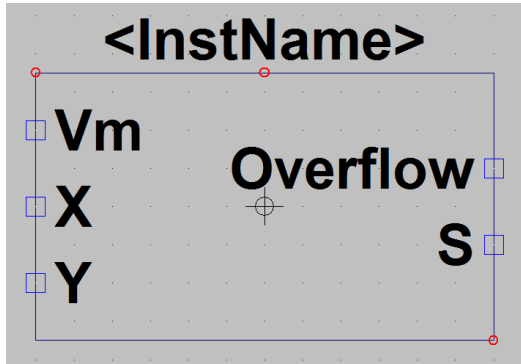
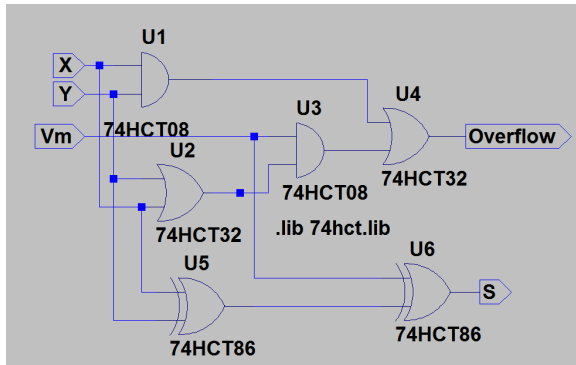
Antes de começar a abordar os tópicos acima é necessário a abordagem da criação de dois blocos adicionais para melhor apresentação e criação dos circuitos.

- **Meio Somador**



O Circuito acima é responsável pela função da soma de dois bits, porém sem levar em conta algum bit anterior. Ao lado dele é o bloco que foi criado a parti do circuito do meio somador.

- **Somador Completo**



O Circuito acima é responsável pela função da soma de dois bits levando em conta algum bit anterior.
Ao lado dele é o bloco que foi criado a parti do circuito do somador completo.

Após o conhecimentos dos dois circuitos adicionais podemos apresentar os casos pedido na tabela acima.

Caso: 0000 → AND Bit a Bit

Foi realizado uma operação AND de X_n por Y_n da seguinte forma:

X3	X2	X1	X0
AND	AND	AND	AND
Y3	Y2	Y1	Y0
↓	↓	↓	↓
S3	S2	S1	S0

O circuito abaixo é responsável pela operação demonstrado pela tabela acima:

Circuito	Bloco gerado
	

Caso: 0001 → OR Bit a Bit

Foi realizado uma operação OR de X_n por Y_n da seguinte forma:

X3	X2	X1	X0
OR	OR	OR	OR
Y3	Y2	Y1	Y0
↓	↓	↓	↓
S3	S2	S1	S0

O circuito abaixo é responsável pela operação demonstrado pela tabela acima:

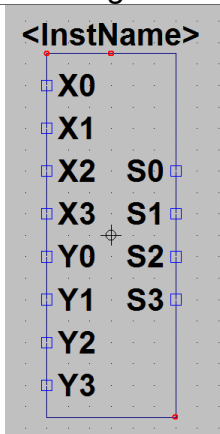
Circuito	Bloco gerado
	

Caso: 0010 → XOR Bit a Bit

Foi realizado uma operação XOR de X_n por Y_n da seguinte forma:

X3	X2	X1	X0
XOR	XOR	XOR	XOR
Y3	Y2	Y1	Y0
↓	↓	↓	↓
S3	S2	S1	S0

O circuito abaixo é responsável pela operação demonstrado pela tabela acima:

Circuito	Bloco gerado
	

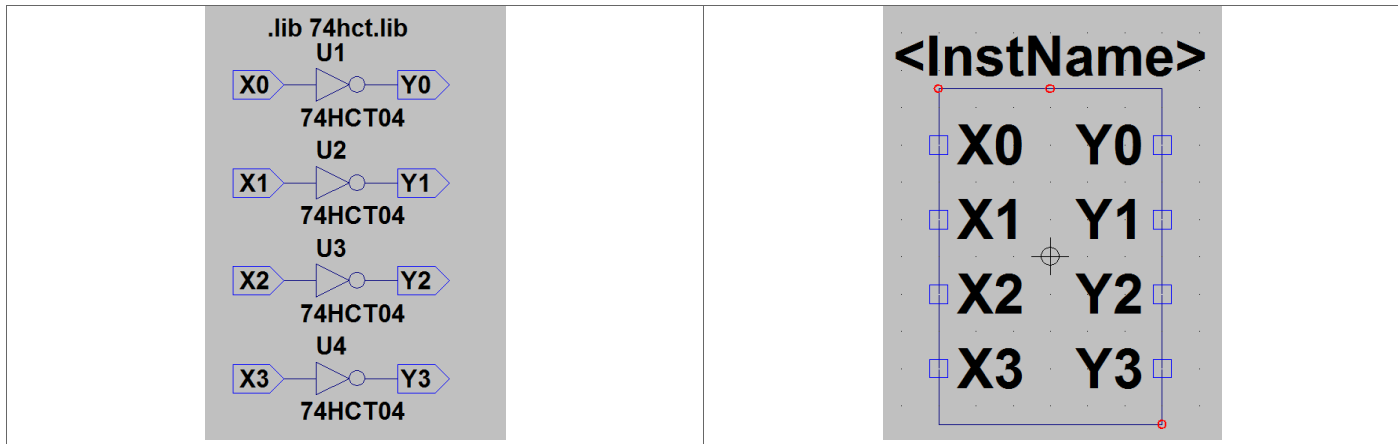
Caso: 0011 → NOT inversão de X Bit a Bit

Foi realizado uma operação NOT de X_n da seguinte forma:

X3	X2	X1	X0
NOT	NOT	NOT	NOT
↓	↓	↓	↓
S3	S2	S1	S0

O circuito abaixo é responsável pela operação demonstrado pela tabela acima:

Circuito	Bloco gerado
----------	--------------

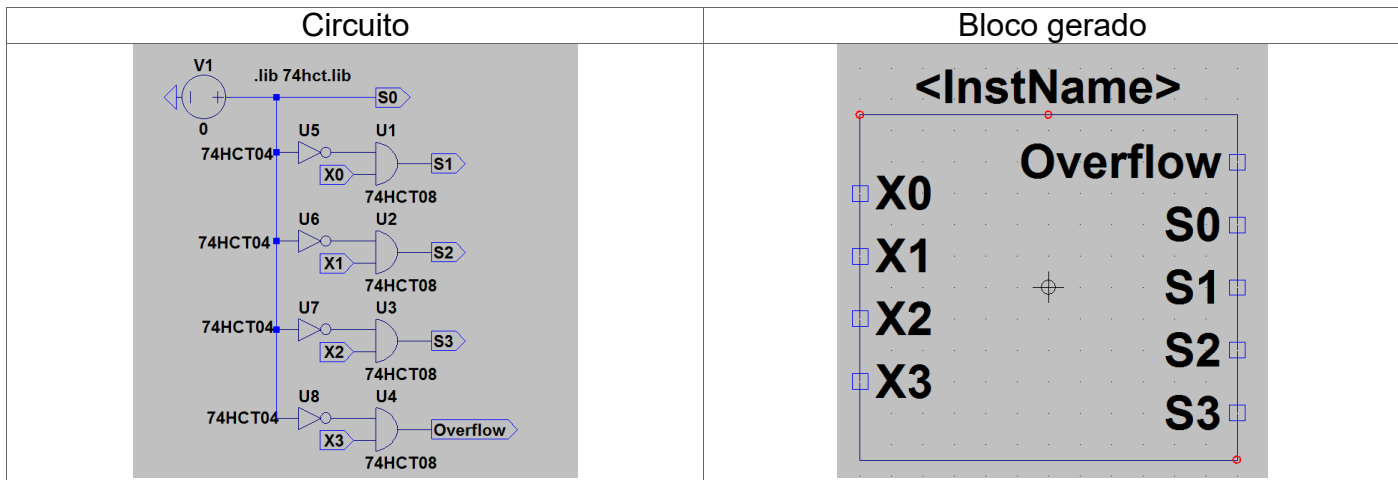


Caso: 0100 → Deslocamento um bit à esquerda de X

Foi realizado uma operação de deslocamento X_n da seguinte forma:

Overflow	X3	X2	X1	X0
	↓	↓	↓	↓
	←	←	←	←
Overflow= X3	S3=X2	S2=X1	S1=X0	S0=0

O circuito abaixo é responsável pela operação demonstrado pela tabela acima:

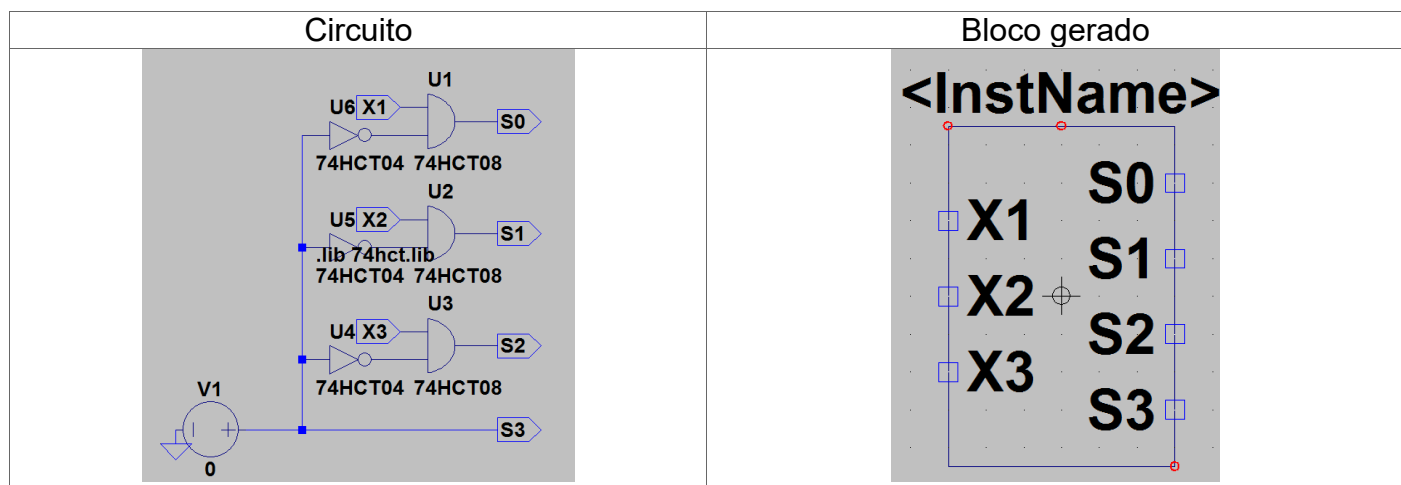


Caso: 0101 → Deslocamento um bit à direita de X

Foi realizado uma operação de deslocamento X_n da seguinte forma:

X3	X2	X1	X0
↓	↓	↓	↓
→	→	→	→
S3=0	S2=X3	S1=X2	S0=X1

O circuito abaixo é responsável pela operação demonstrado pela tabela acima:



Caso: 0110 → Deslocamento circular de um bit à esquerda de X

Foi realizado uma operação de deslocamento X_n da seguinte forma:

Diagram illustrating the shift operation for the addition of two 4-bit numbers. The top row shows the inputs X_3 , X_2 , X_1 , and X_0 . The second row shows the result of the addition, with carry bits indicated by arrows. The third row shows the shifted result, with the carry bit shifted left and the original result shifted right. The bottom row shows the final result $S_3=X_2$, $S_2=X_1$, $S_1=X_0$, and $S_0=X_3$.

O circuito abaixo é responsável pela operação demonstrado pela tabela acima:

Circuito	Bloco gerado

Caso: 0111 → Deslocamento circular de um bit à direita de X

Foi realizado uma operação de deslocamento Xn da seguinte forma:

X3	X2	X1	X0
↓	↓	↓	↓
→	→	→	→
↓	←	←	←
S3=X0	S2=X3	S1=X2	S0=X1

O circuito abaixo é responsável pela operação demonstrado pela tabela acima:

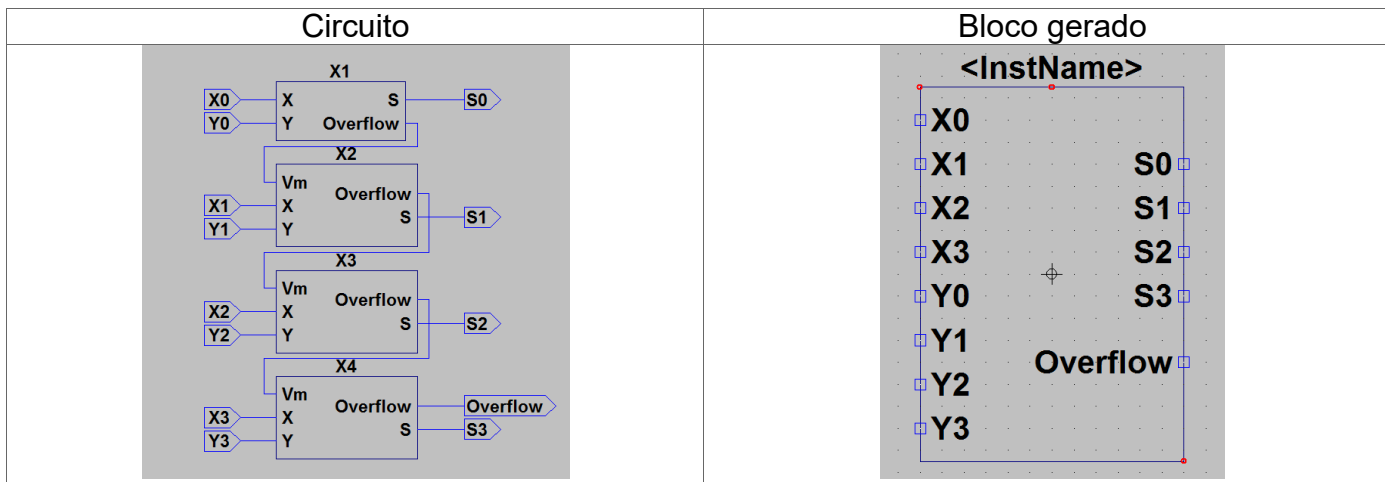
Circuito	Bloco gerado

Caso: 1000 → Soma Binária

Foi realizado a Operação de Soma da seguinte forma:

	V _{m-1}	V _{m-1}	V _{m-1}	
Overflow	X3	X2	X1	X0
+				
	Y3	X2	X1	
↓	↓	↓	↓	↓
Overflow	S3	S2	S1	S0

O circuito abaixo é responsável pela operação demonstrado pela tabela acima:



Na criação do Circuito somador foi utilizado um meio somador e três somadores completo como pode ser visto no circuito acima.

O meio somador foi usado para os dois primeiro bits pois são os únicos que não dependem de um bit vindo de outra operação antiga, já que essa é a primeira operação da soma dos bits.

Caso: 1001 → Subtração Binária

(^) → é correspondente ao Barramento do valor.

A subtração foi realizada seguido os seguintes procedimentos, primeiramente foi feito o complemento de dois, do segundo operando, Y nesse caso, Após saber seu valor negativo foi realizada a soma com o valor X:

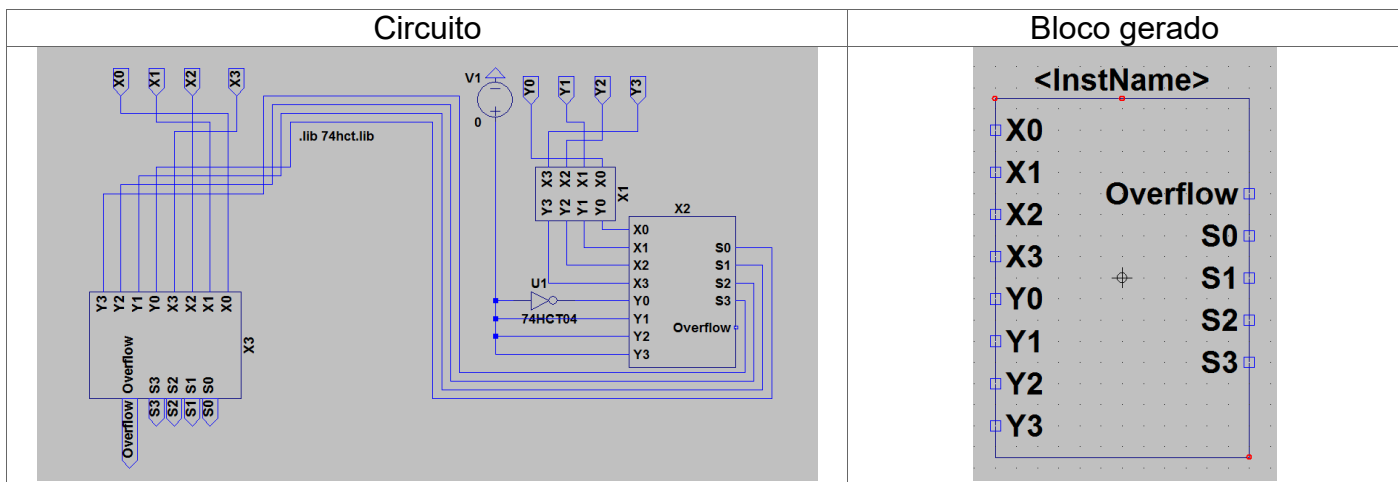
O circuito foi construído com dois blocos já mostrados anteriormente, uma inversora de 4 bits e dois somadores de 4 bits de cada operando.

Foi realizado a Operação de Soma da seguinte forma:

Y3	Y2	Y1	Y0
----	----	----	----

	NOT	NOT	NOT	NOT
	$\wedge Y3$	$\wedge Y3$	$\wedge Y3$	$\wedge Y3$
+	0	0	0	1
	↓	↓	↓	↓
	ST3	ST2	ST1	ST0
(Nesse ponto da Operação temos o valor negativo de Y)				
	ST3	ST2	ST1	ST0
+	X3	X2	X1	X0
	↓	↓	↓	↓
	S3	S2	S1	S0

O circuito abaixo é responsável pela operação demonstrado pela tabela acima:



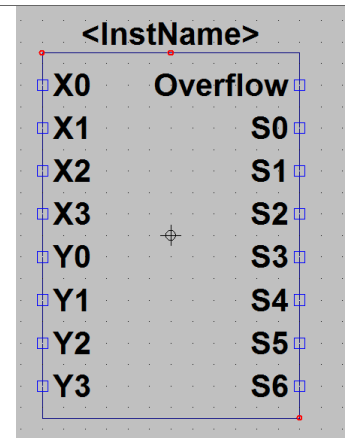
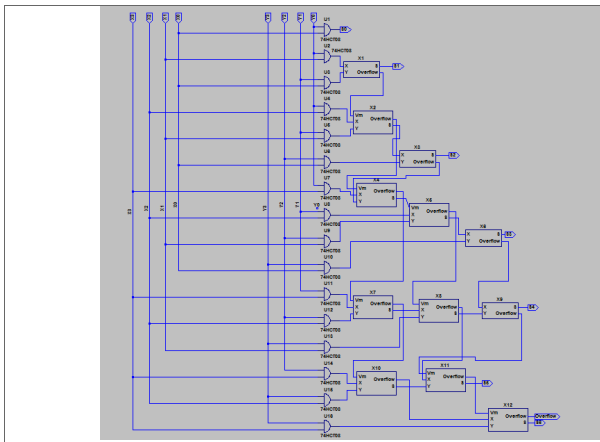
Caso: 1010 → Multiplicação Binária

O circuito multiplicador semelhante ao somador foi feito com base em Meio somadores e somadores completo, Ele foi planejado da seguinte forma:

Overflow					Y3	Y2	Y1	Y0
					X3	X2	X1	X0
*								
					X0*Y3	X0*Y2	X0*Y1	Y0*X0
				X1*Y3	X1*Y2	X1*Y1	X1*Y0	↓
		X2*Y3	X2*Y2	X2*Y1	X2*Y0	↓	↓	↓
	X3*Y3	X3*Y2	X3*Y1	X3*Y0	↓	↓	↓	↓
	↓	↓	↓	↓	↓	↓	↓	↓
Overflow	S6	S5	S4	S3	S2	S1	S0	

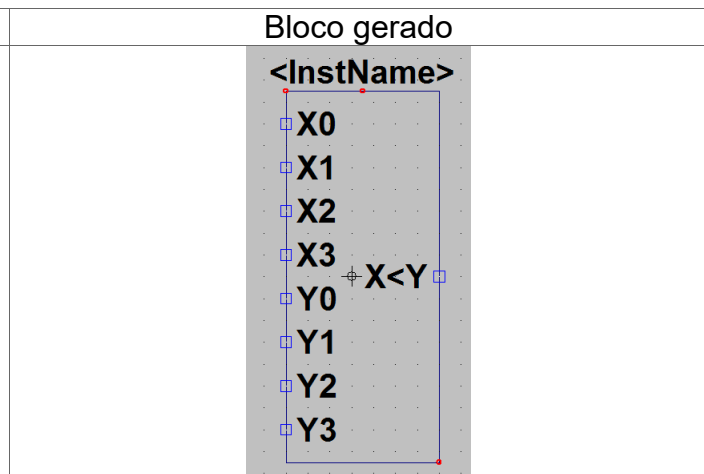
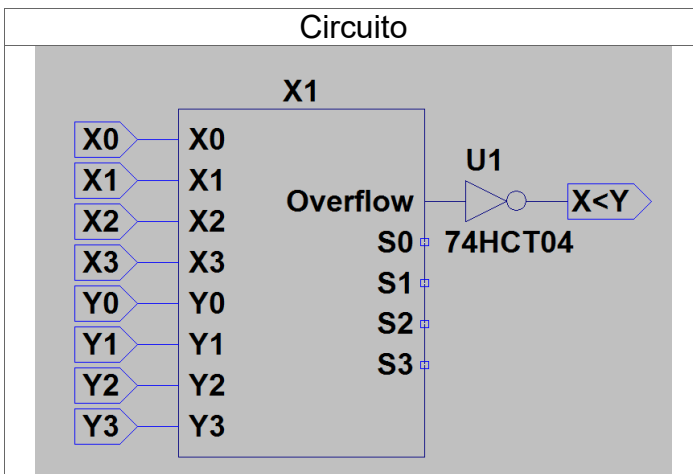
O circuito abaixo é responsável pela operação demonstrado pela tabela acima:





Caso: 1011 → Comparação de Valores, $X < Y$

Para a criação do Circuito comparador desse caso foi usado o bloco responsável pela subtração, pois foi notado uma característica quando é feita uma operação, que o Overflow é 1 Lógico (5v) quando o resultado é negativo e ele é 0 Lógico (0v) quando o resultado é positivo, então temos o seguinte circuito:

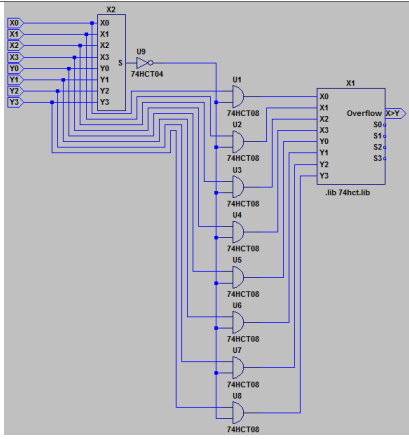
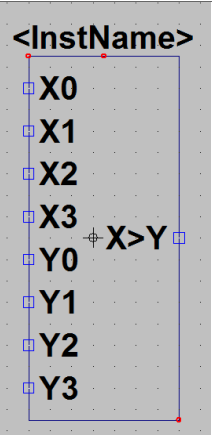


Caso: 1100 → Comparação de Valores, $X > Y$

Para a criação do Circuito comparador desse caso foi usado o bloco responsável pela subtração, pois foi notado uma característica quando é feita

uma operação, que o Overflow é 1 Lógico (5v) quando o resultado é negativo e ele é 0 Lógico (0v) quando o resultado é positivo, então temos o seguinte circuito:

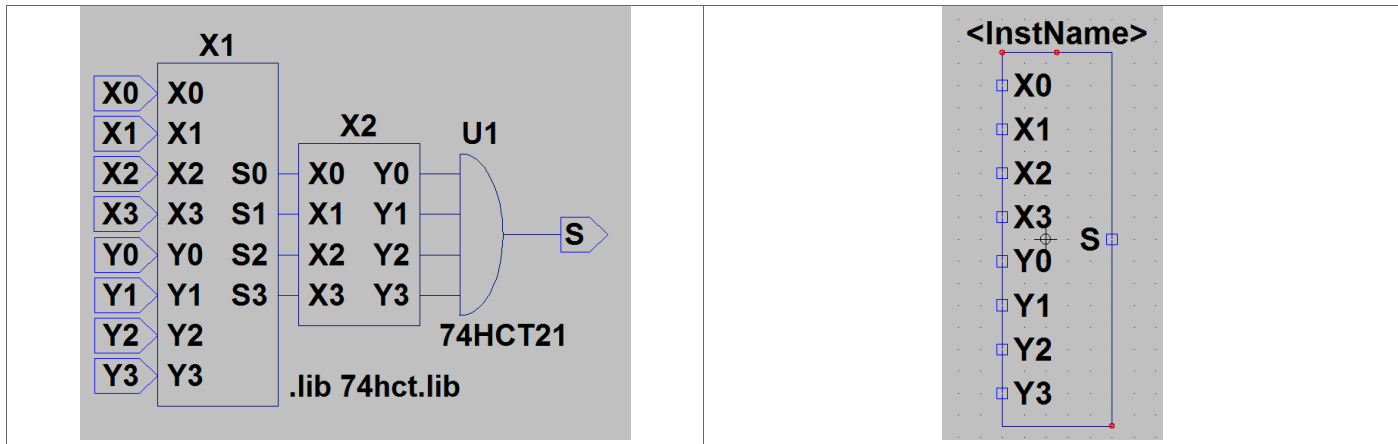
Nesse circuito houve uma peculiaridade que não houve no anterior, pois quando entrava o X e o Y iguais ele afirmava com a resposta em 1 lógico que o X era maior que o Y, então foi necessário testar se os dois eram diferentes, caso sejam, o teste é feito, caso contrário o bloco irá retornar 0.

Circuito	Bloco gerado
	

Caso: 1101 → Comparação de Valores, X=Y

Para a criação do circuito de teste de igualdade foi usado dois blocos já mostrados, o bloco das XORs e um bloco das inversoras bit a bit e a saída foram todas conectadas em uma AND para testar seus valores.

Circuito	Bloco gerado
----------	--------------

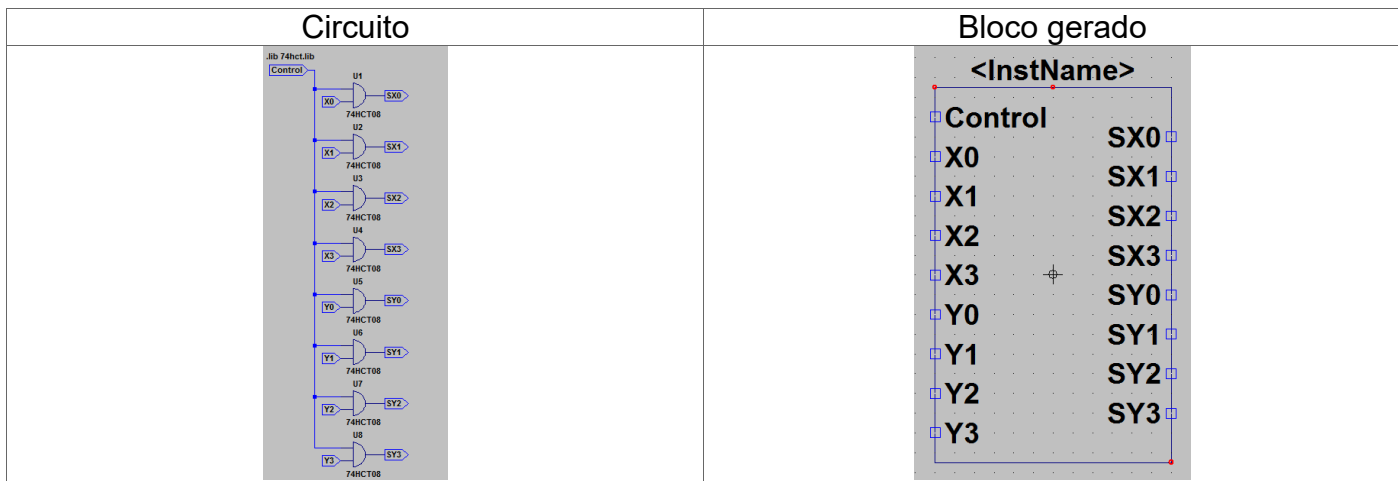


Circuitos Auxiliares

Foi necessário a criação de circuitos para auxiliar e organizar melhor as saídas finais.

Circuito Auxiliar de Seleção:

Esse circuito tem a característica de repassar as entradas para as saídas somente quando o sinal de controle tiver valor de 1 lógico:



Circuito Auxiliar Final Aritmética:

Esse circuito tem como característica organizar todas as saídas de operações aritméticas feitas no circuito em uma mesma saída.

Circuito	Bloco gerado

Circuito Auxiliar Final Deslocamento:

Esse circuito tem como característica organizar todas as saídas de operações de deslocamento feitas no circuito em uma mesma saída.

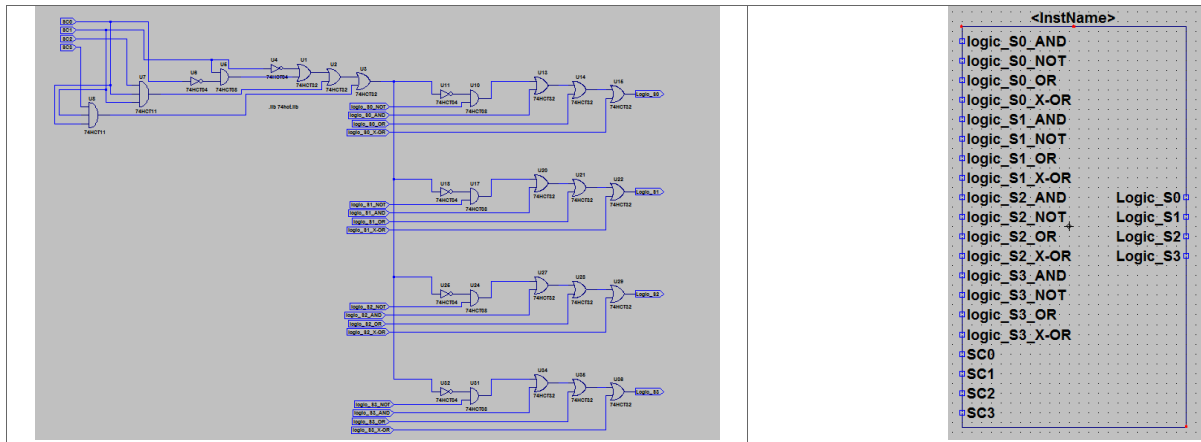
Circuito	Bloco gerado

Circuito Auxiliar Final Lógica:

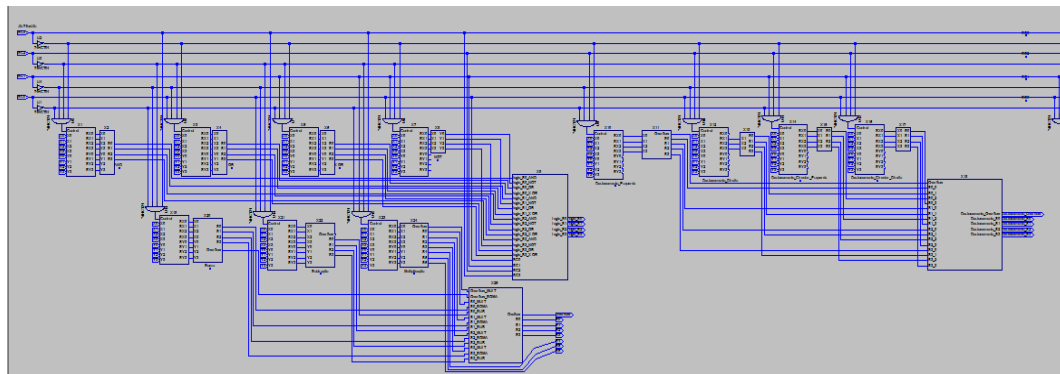
Esse circuito tem como característica organizar todas as saídas de operações Lógicas feitas no circuito em uma mesma saída.

Nesse circuito foi necessário a criação de um circuito adicional no seu interior para “cancelar” o caso que o sinal de controle zere as entradas na Porta NOT por iria sair todas como 1 lógico.

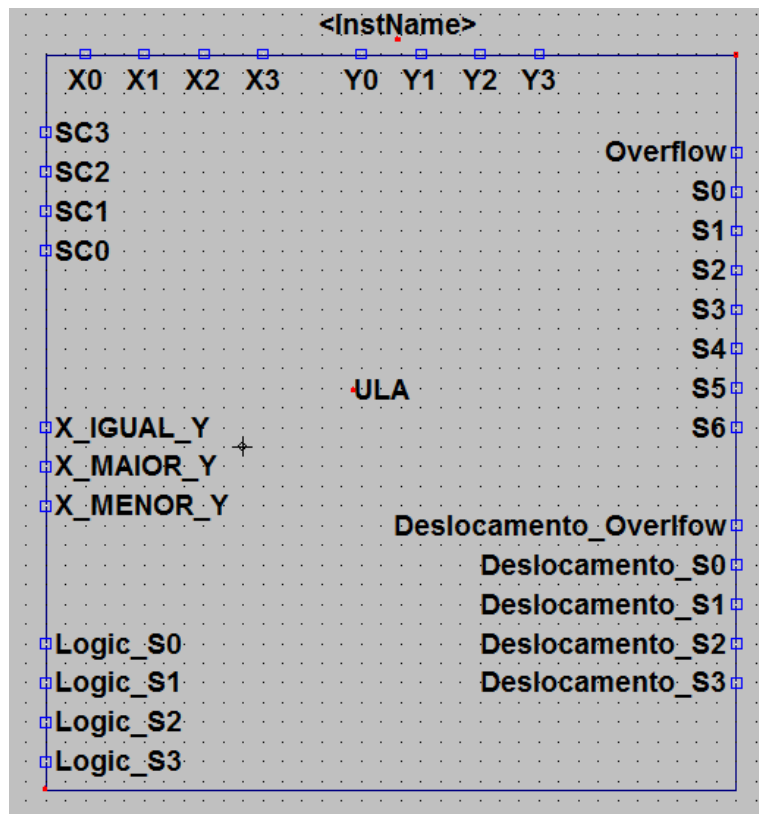
Circuito	Bloco gerado
----------	--------------



Após abordar todos os blocos de forma separadamente, já podemos demonstrar toda as ligações necessárias para ter uma ULA funcionando de acordo como o que foi pedido no relatório, então temos que após a construção da ULA temos o seguinte circuito:



Após ter montado toda a ULA é possível gerar e organizar o seu Bloco:



Testes da ULA

Para os seguintes testes iremos adotar os seguintes valores:

X=1100

Y=1010

(Os valores de X e Y estão sendo esses pois correspondem a todos os casos da tabela verdade)

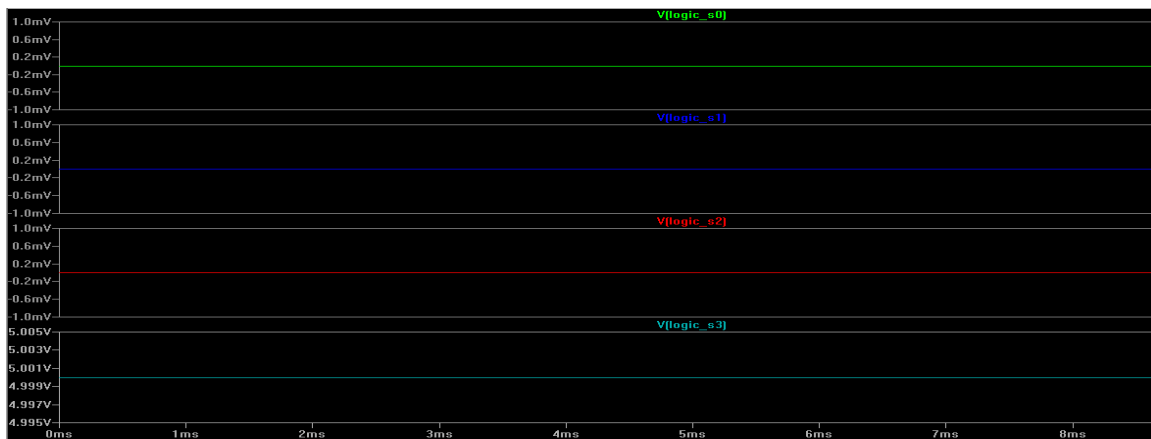
Caso(AND): 0000

$X_0=0 \text{ AND } Y_0=0 \Rightarrow \text{Logic_S0}=0$

$X_1=0 \text{ AND } Y_1=0 \Rightarrow \text{Logic_S1}=0$

$X_2=1 \text{ AND } Y_2=0 \Rightarrow \text{Logic_S2}=0$

$X_3=1 \text{ AND } Y_3=1 \Rightarrow \text{Logic_S3}=1$



Como foi esperado os valores correspondem as descritos acima da imagem.

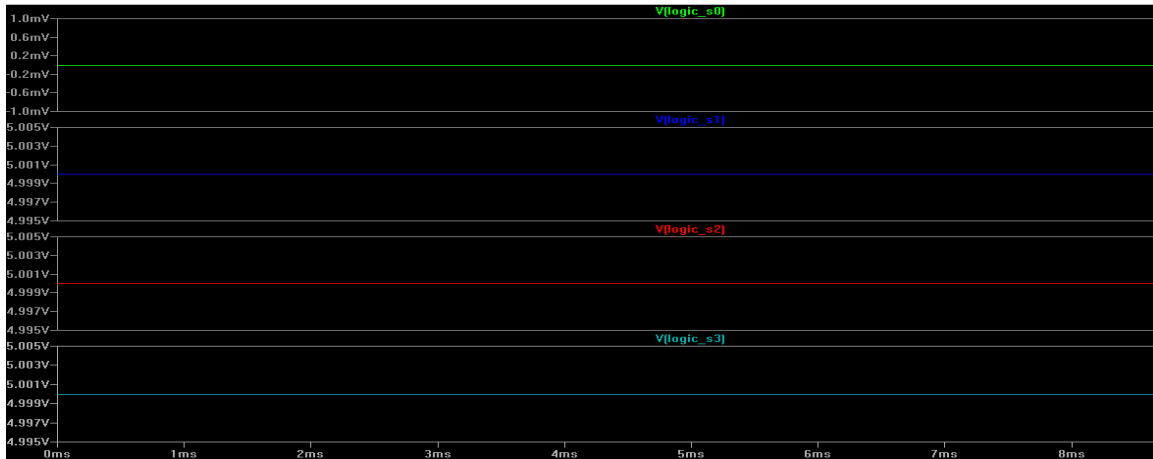
Caso(OR): 0001

$X_0=0 \text{ OR } Y_0=0 \Rightarrow \text{Logic_S0}=0$

$X_1=0 \text{ OR } Y_1=0 \Rightarrow \text{Logic_S1}=1$

$X_2=1 \text{ OR } Y_2=0 \Rightarrow \text{Logic_S2}=1$

$X_3=1 \text{ OR } Y_3=1 \Rightarrow \text{Logic_S3}=1$



Como foi esperado os valores correspondem as descritos acima da imagem.

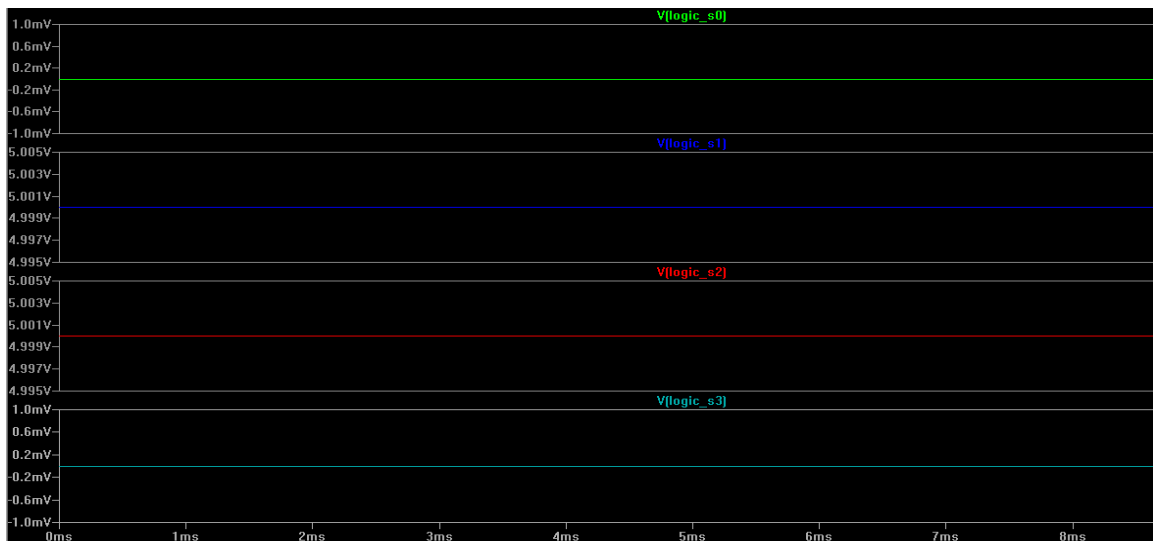
Caso(XOR): 0010

$X_0=0 \text{ XOR } Y_0=0 \Rightarrow \text{Logic_S0}=0$

$X_1=0 \text{ XOR } Y_1=0 \Rightarrow \text{Logic_S1}=1$

$X_2=1 \text{ XOR } Y_2=0 \Rightarrow \text{Logic_S2}=1$

$X_3=1 \text{ XOR } Y_3=1 \Rightarrow \text{Logic_S3}=0$



Como foi esperado os valores correspondem as descritos acima da imagem.

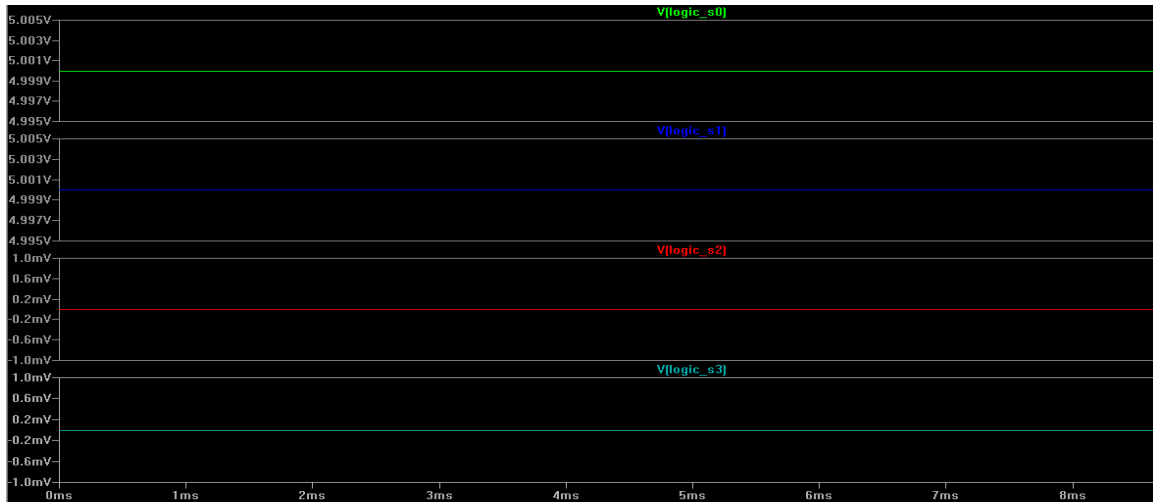
Caso(NOT): 0011

$X_0=0 \Rightarrow \text{Logic_S0}=1$

$X_1=0 \Rightarrow \text{Logic_S1}=1$

$X_2=1 \Rightarrow \text{Logic_S2}=0$

$X_3=1 \Rightarrow \text{Logic_S3}=0$



Como foi esperado os valores correspondem as descritos acima da imagem.

Caso (Deslocamento de um bit à esquerda): 0100

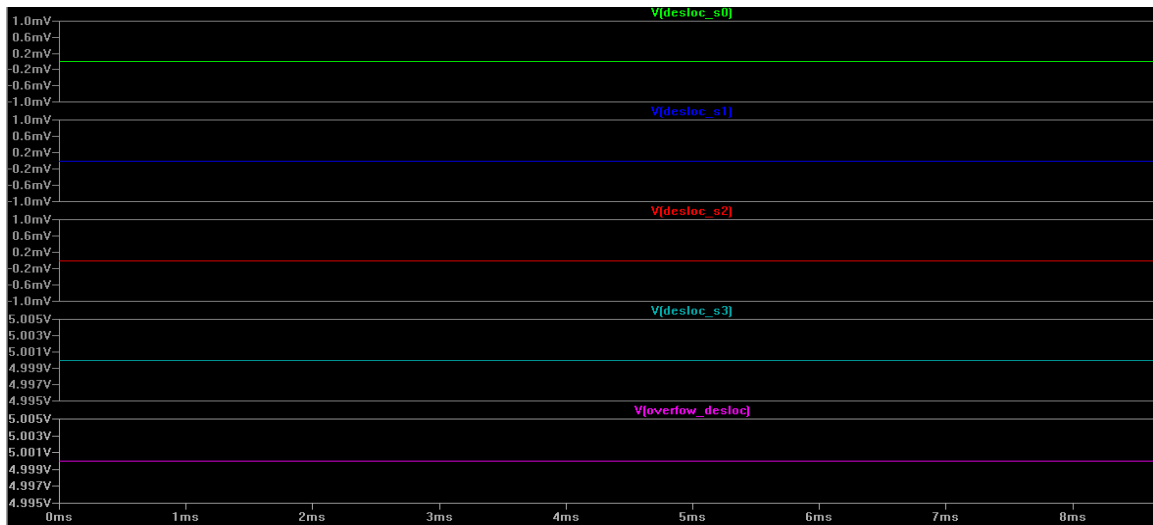
$X_0=0 \Rightarrow \text{Deslocamento_S0}=0$

$X_1=0 \Rightarrow \text{Deslocamento_S1}=X_0(0)$

$X_2=1 \Rightarrow \text{Deslocamento_S2}=X_1(0)$

$X_3=1 \Rightarrow \text{Deslocamento_S3}=X_2(1)$

$\text{Overflow}=0 \Rightarrow \text{Deslocamento_Overflow}=X_3(1)$



O resultado das saídas correspondem ao valores citados acima.

Caso (Deslocamento de um bit à direita): 0101

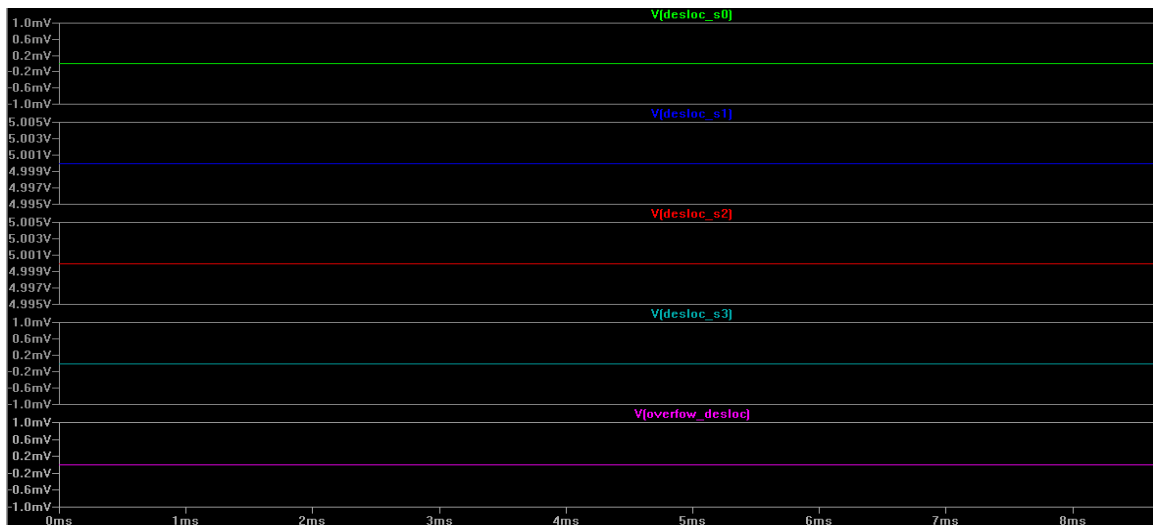
$X0=0 \Rightarrow \text{Deslocamento_S0}=X1$

$X1=0 \Rightarrow \text{Deslocamento_S1}=X2$

$X2=1 \Rightarrow \text{Deslocamento_S2}=X3$

$X3=1 \Rightarrow \text{Deslocamento_S3}=0$

$\text{Overflow}=0 \Rightarrow \text{Deslocamento_Overflow}=0$



O resultado das saídas correspondem ao valores citados acima.

Caso (Deslocamento circular de um bit à esquerda): 0110

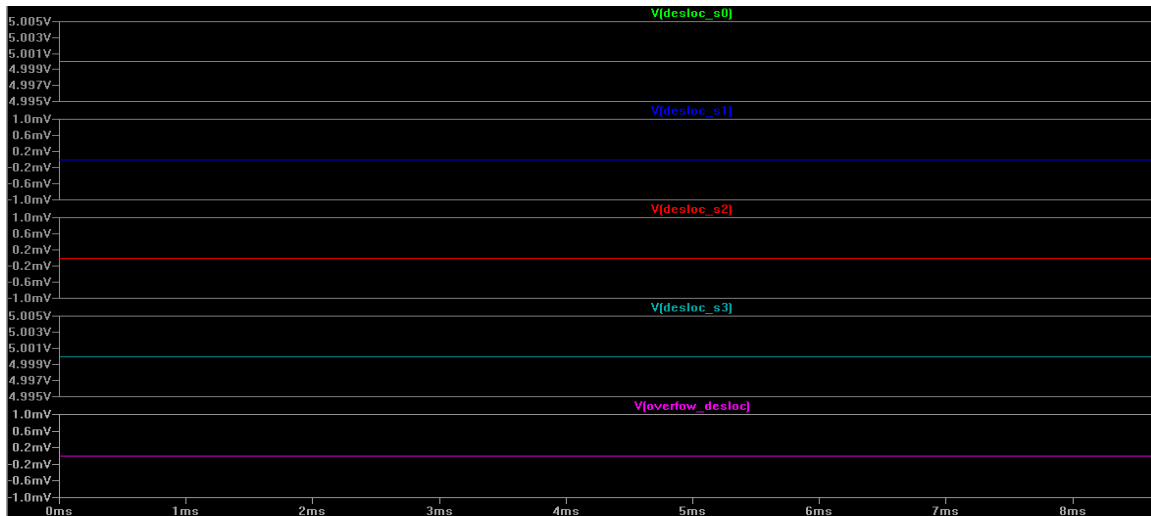
X0=0 => Deslocamento_S0=X3

X1=0 => Deslocamento_S1=X0

X2=1 => Deslocamento_S2=X1

X3=1 => Deslocamento_S3=X2

Overflow=0 => Deslocamento_Overflow=0



O resultado das saídas correspondem ao valores citados acima.

Caso (Deslocamento circular de um bit à direita): 0111

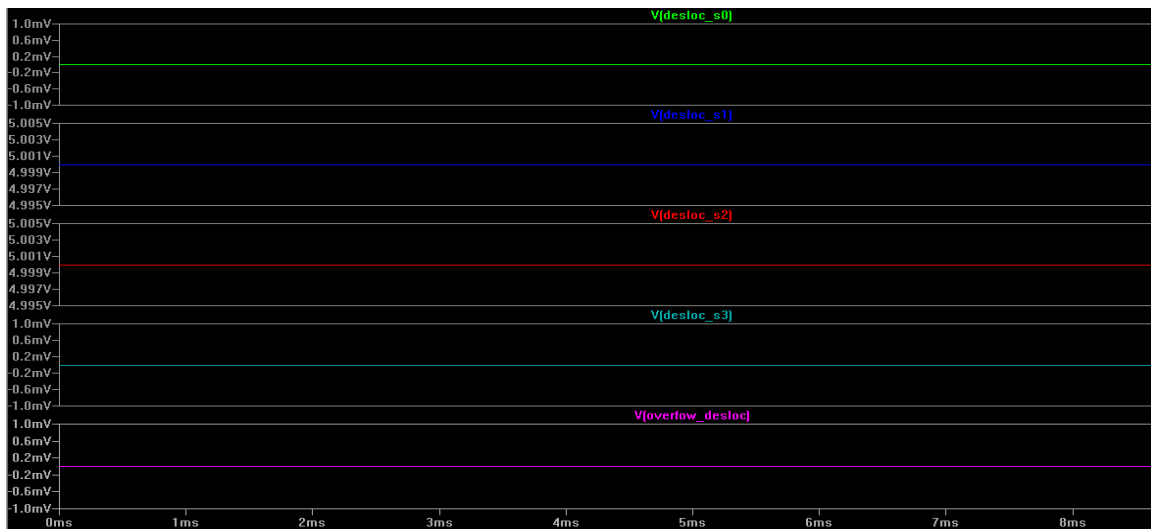
X0=0 => Deslocamento_S0=X1

X1=0 => Deslocamento_S1=X2

X2=1 => Deslocamento_S2=X3

X3=1 => Deslocamento_S3=X0

Overflow=0 => Deslocamento_Overflow=0



O resultado das saídas correspondem ao valores citados acima.

Caso(Soma):1000

X=1100 (12 Dec)

Y=1010 (10 Dec)

S= 10110 (22 Dec)

Saídas esperada na parte aritmética da ULA:

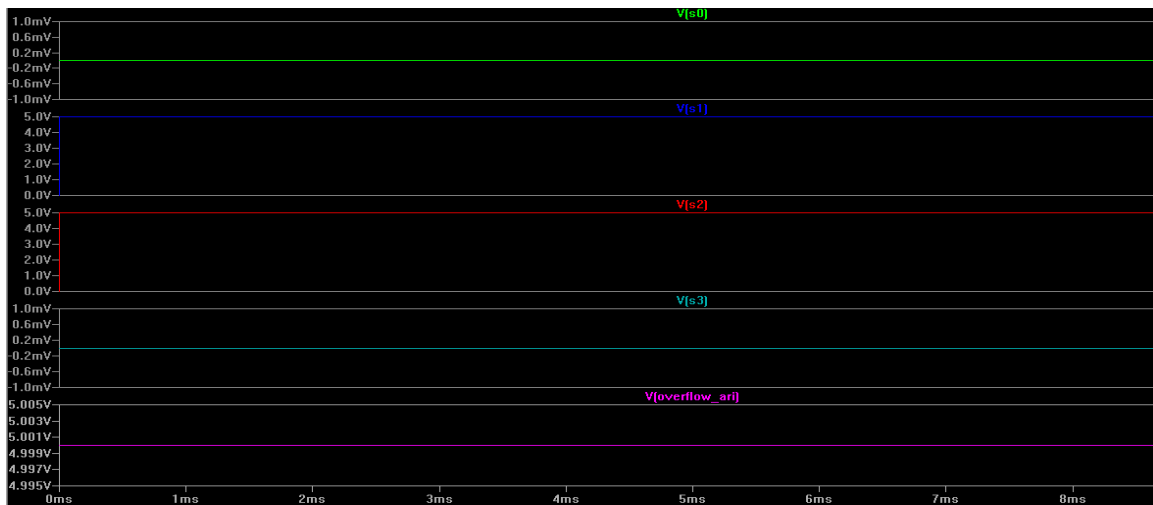
S0=0

S1=1

S2=1

S3=0

Overflow_ARI=1



Caso(Subtração):1001

X=1100 (12 Dec)

Y=1010 (10 Dec)

S= 00010 (22 Dec)

Saídas esperada na parte aritmética da ULA:

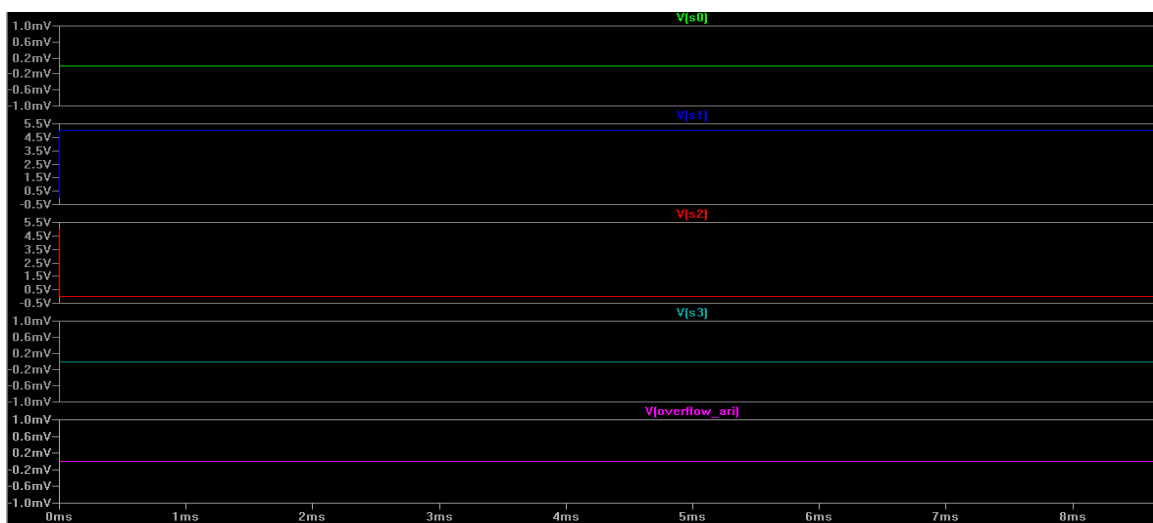
S0=0

S1=1

S2=0

S3=0

Overflow_ARI=0



Caso(Multiplicação):1010

X=1100 (12 Dec)

Y=1010 (10 Dec)

S= 01111000

S0=0

S1=0

S2=0

S3=1

S4=1

S5=1

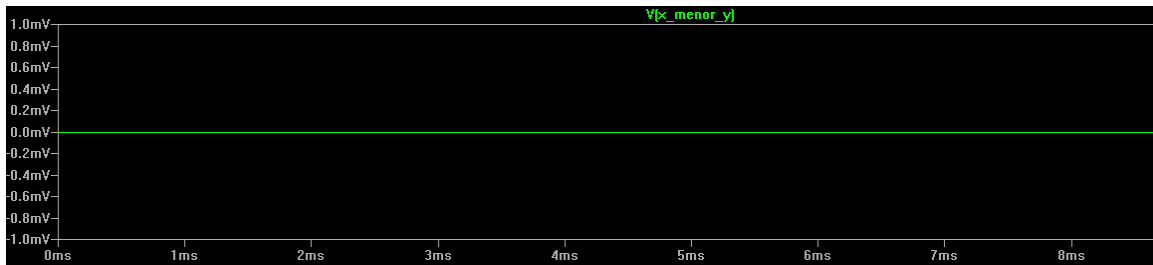
S6=1

Overflow_Ari=0



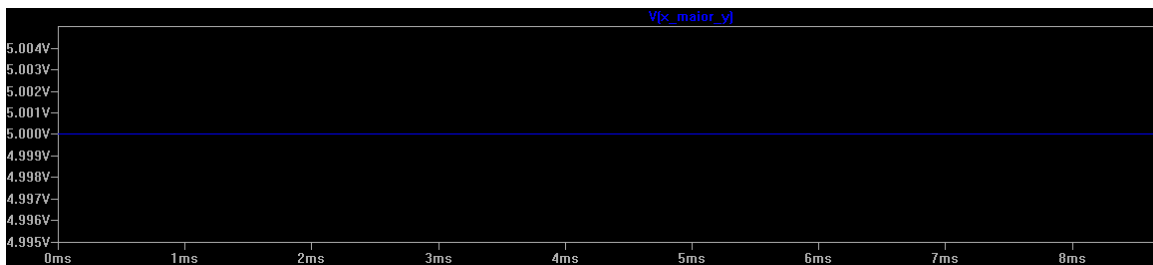
Caso(X<Y): 1011

X<Y=0



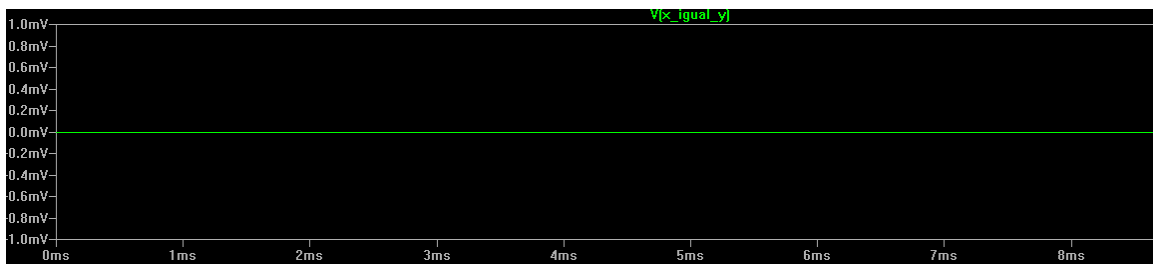
Caso($X>Y$):1100

$X>Y=1$



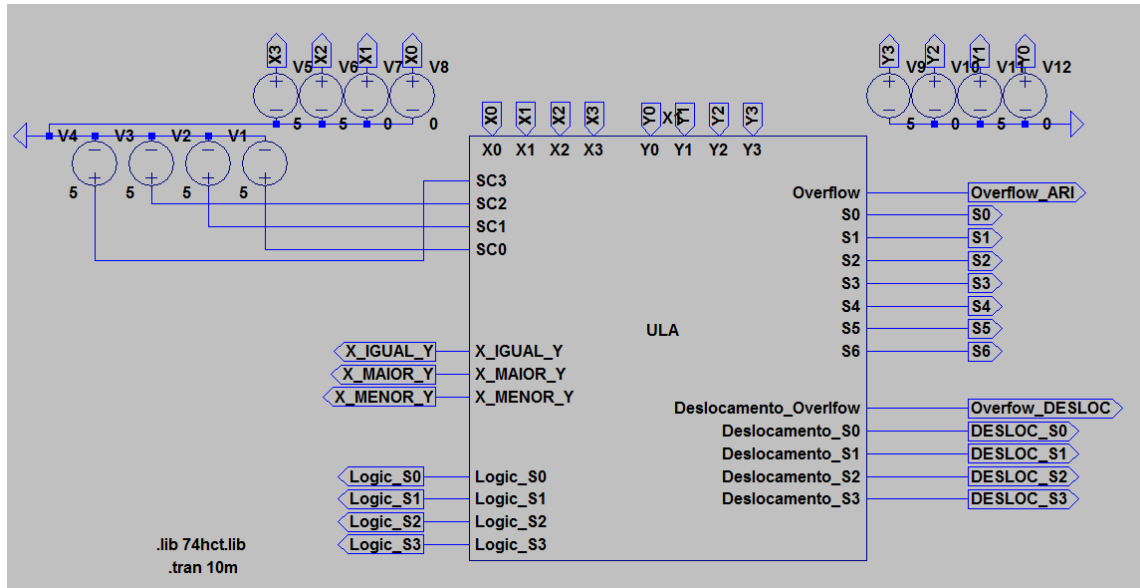
Caso($X=Y$):1101

$X=Y=0$

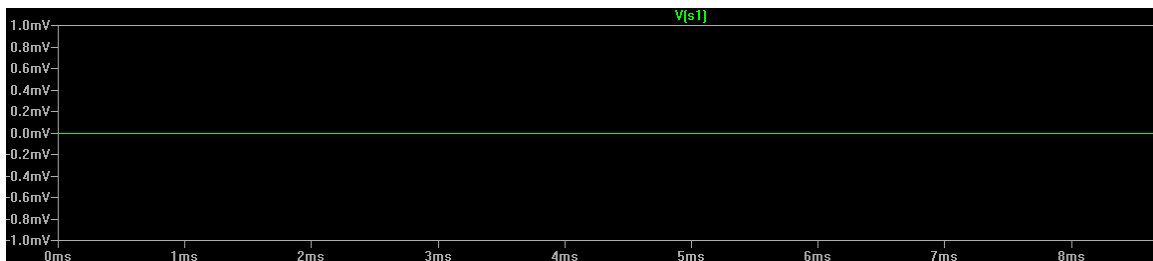
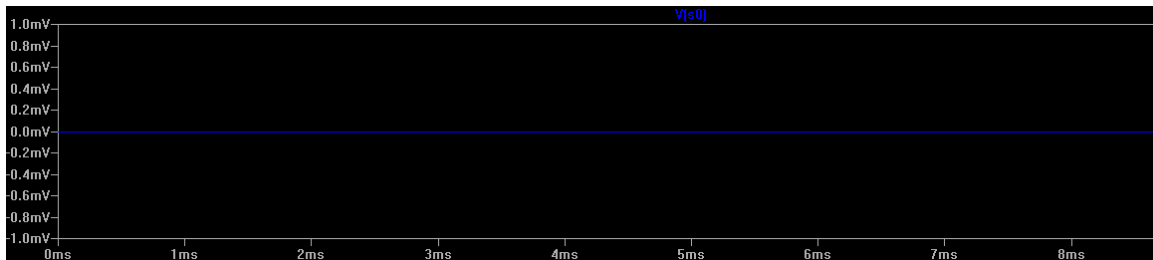


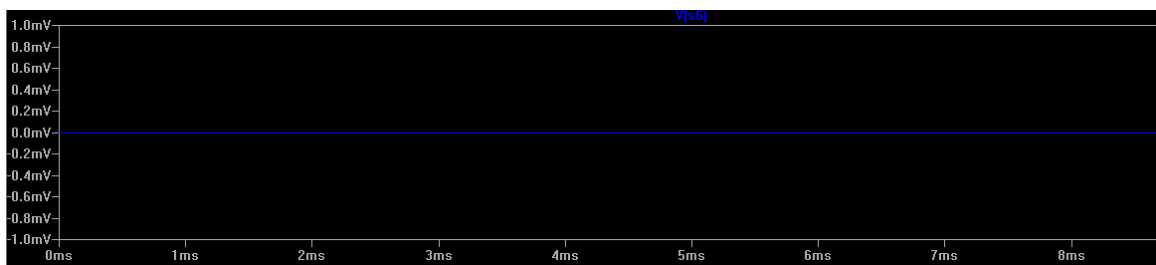
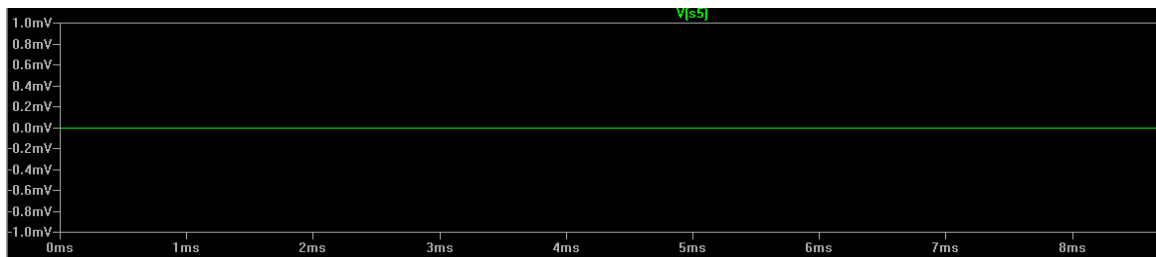
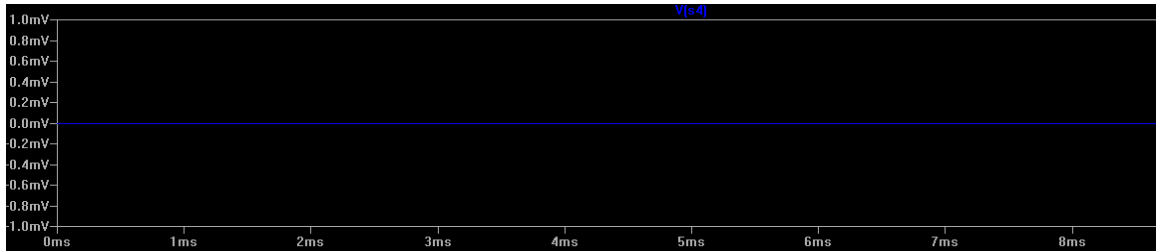
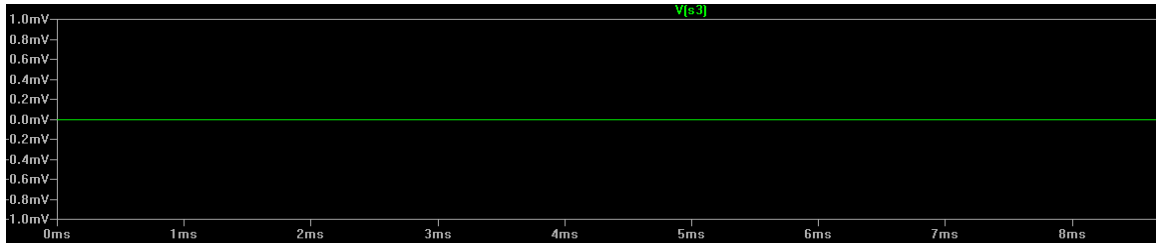
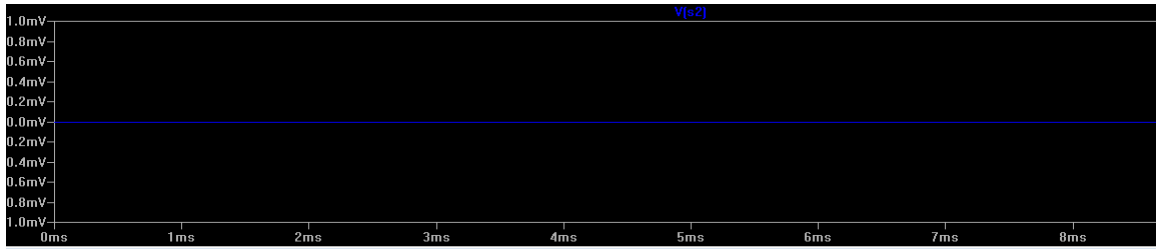
Verificação das saídas em 0 caso não sejam ativadas

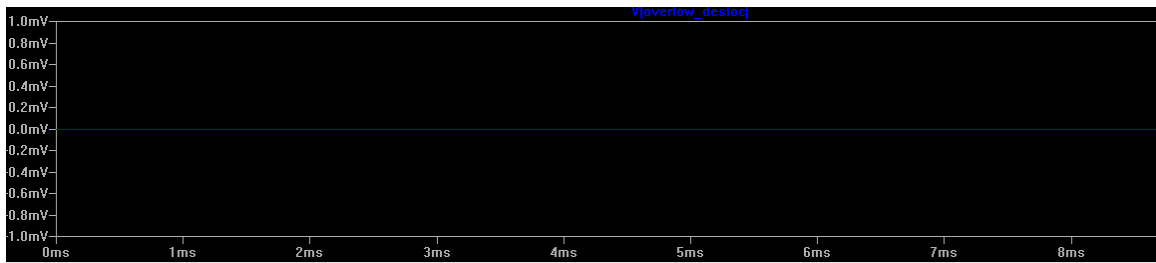
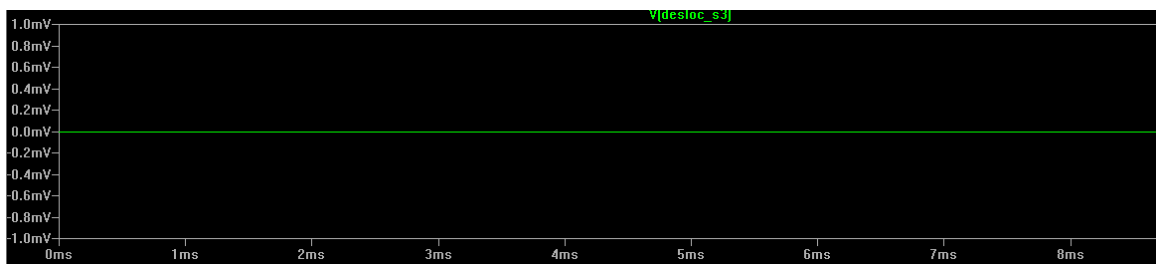
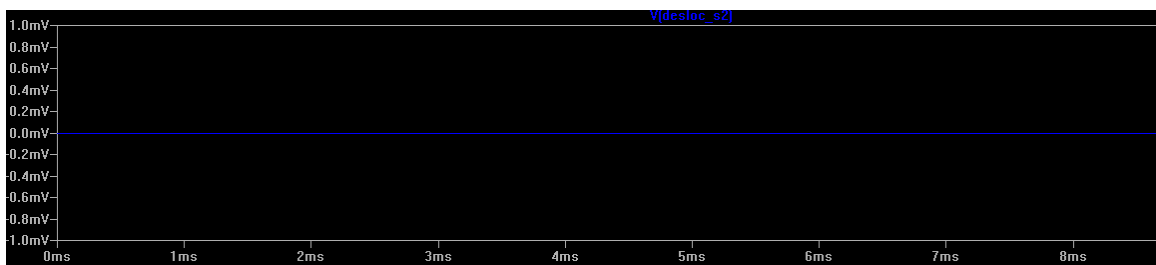
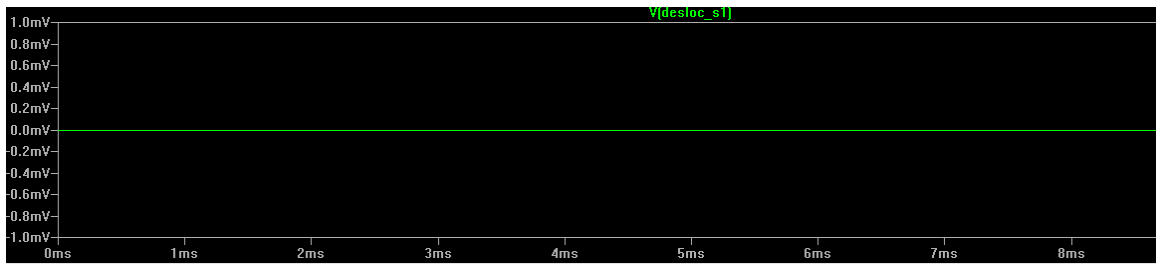
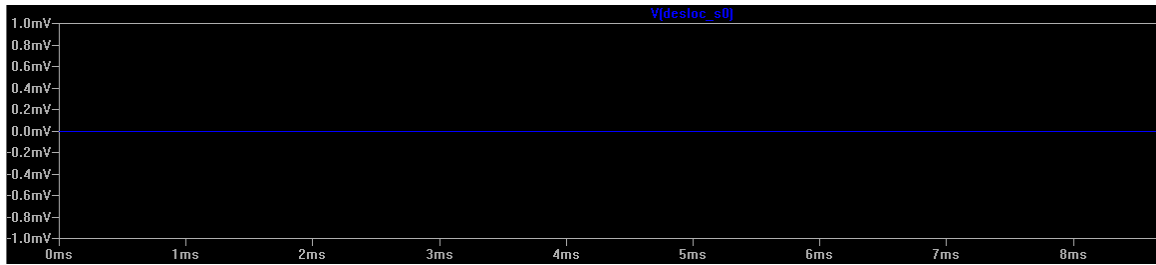
Para essa verificação o sinal de controle foi *setado* em 1111 pois é um dos casos que não há nenhuma operação a ser feita com o sinal de controle nesse valor:

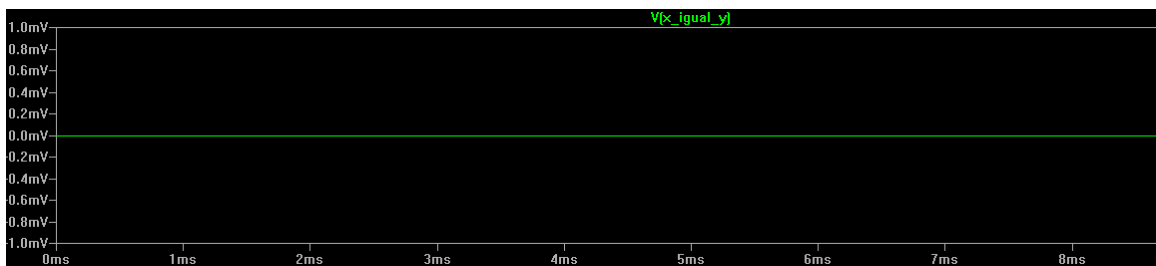
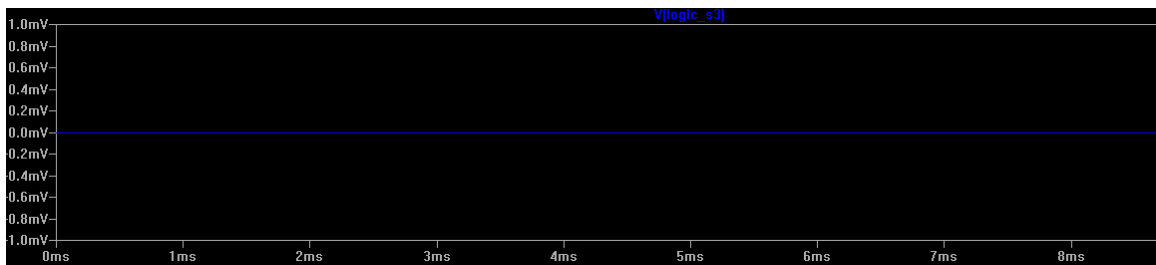
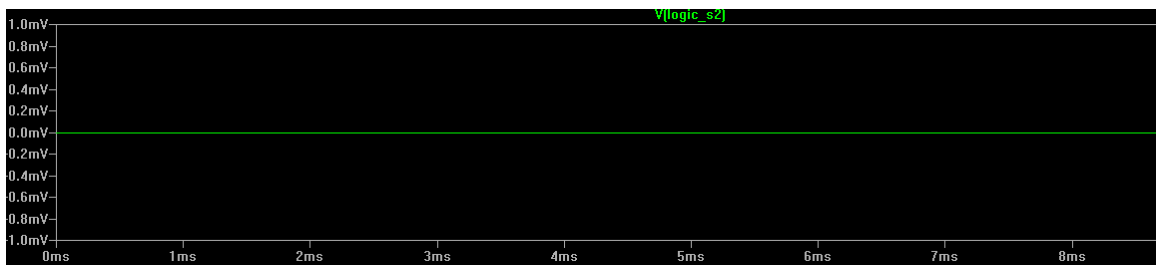
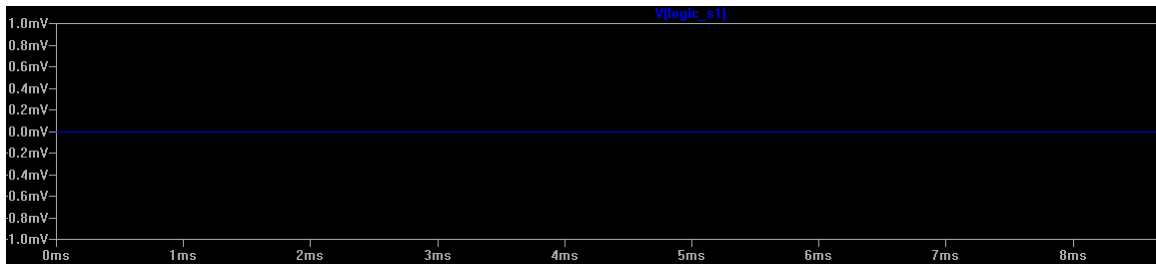
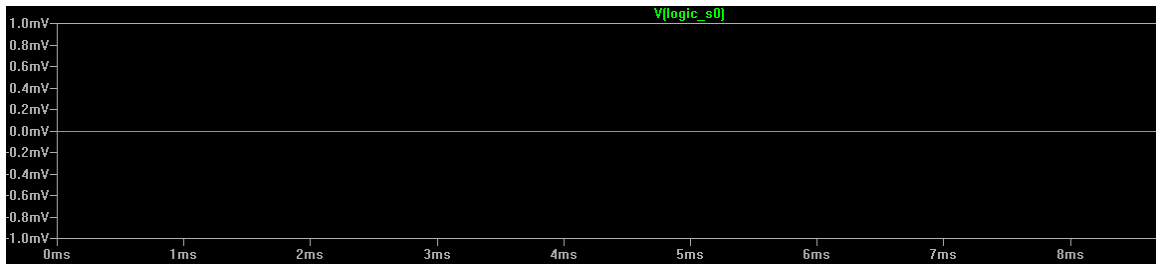


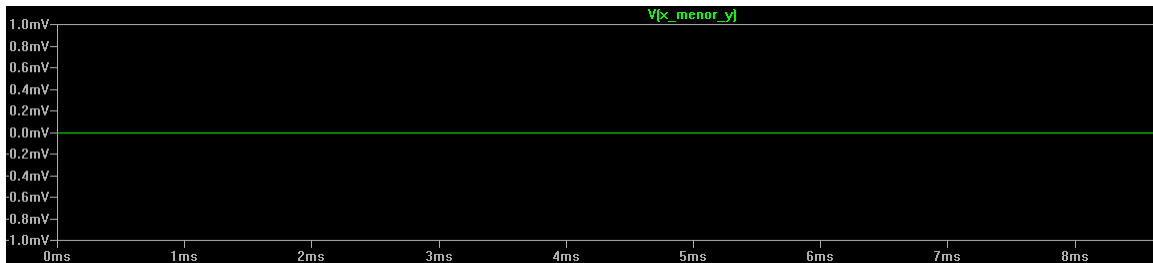
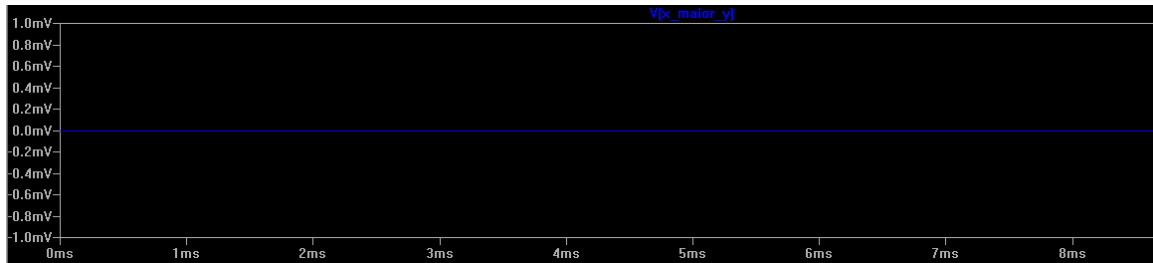
Agora o resultado de todas as saídas











Conclusão

Pode ser notado que foi feito o que foi proposto de uma forma mais complexa em comparação com as outras práticas, mas foi apenas usado os conhecimento já dado em sala de aula e em outras práticas, houve também o estímulo de atividades individuais para resolver problemas adicionais que apareceram no desenvolvimento da ULA.

A Unidade Lógica Aritmética final tem uma configuração, mas caso seja necessário como foi visto no teste das saídas é facilmente maleável para diminuir a quantidade de saídas pois todas sendo 0 quando não estão sendo usadas ajuda muito na reconstrução da saída da ULA.