Exercício Prático 2 Parte 1
Laboratório de ac2

Objetivo:
Cosntruir uma Unidade Lógica e Aritmética (ULA) de 1 bit, 4 bits e implementar no Logisim e Arduino.

Parte 1 (estudo da ALU usando Logisim):

1. Considere a Unidade Lógica e Aritmética de 1 bit ilustrada na Figura 1 a seguir:

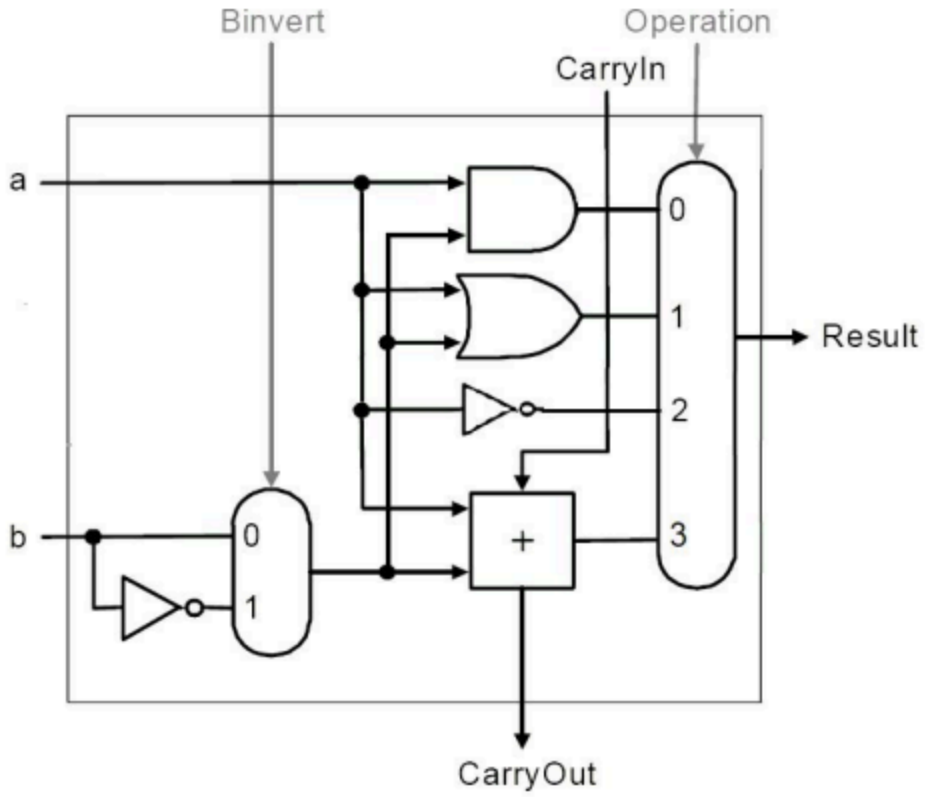
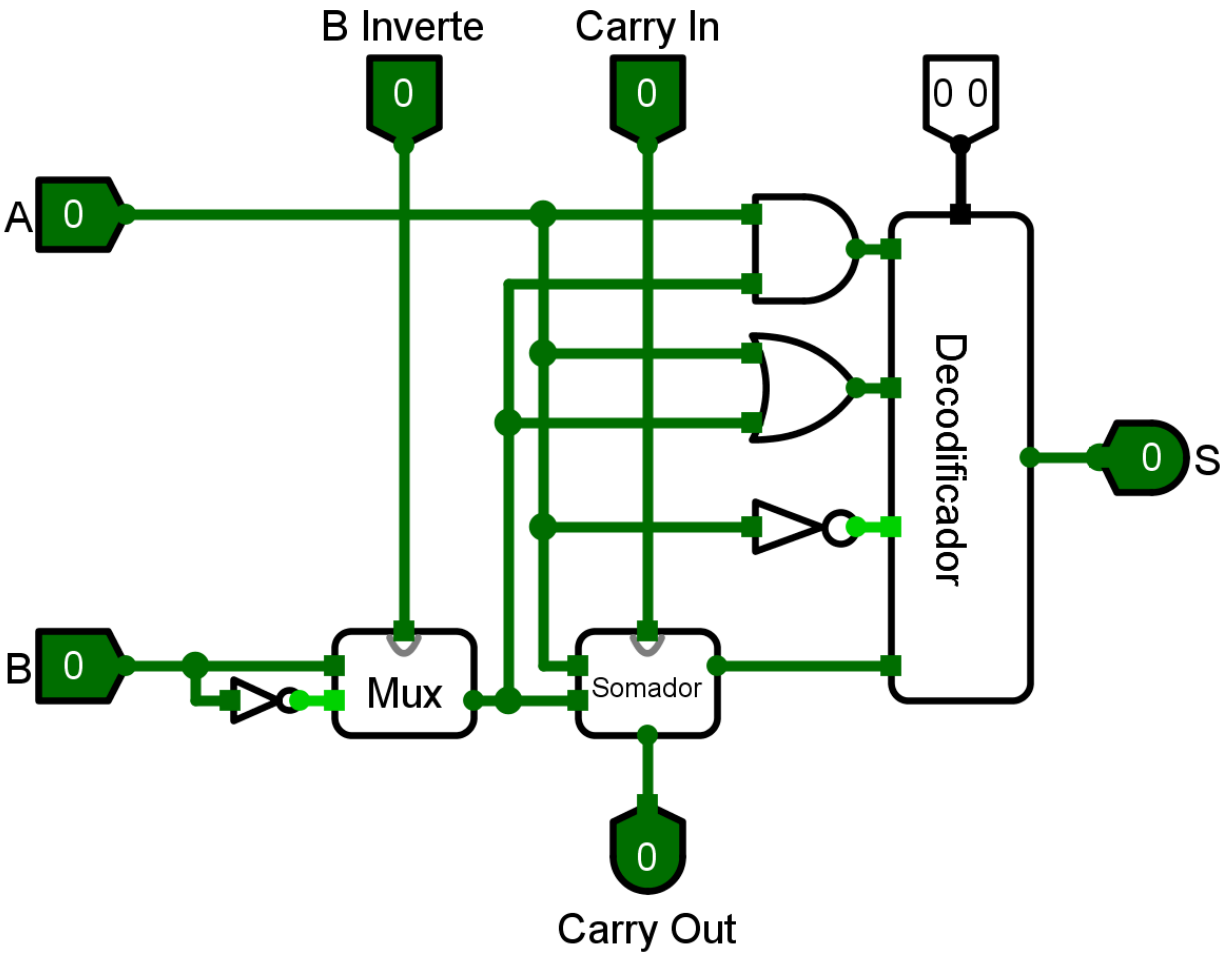


Figura 1

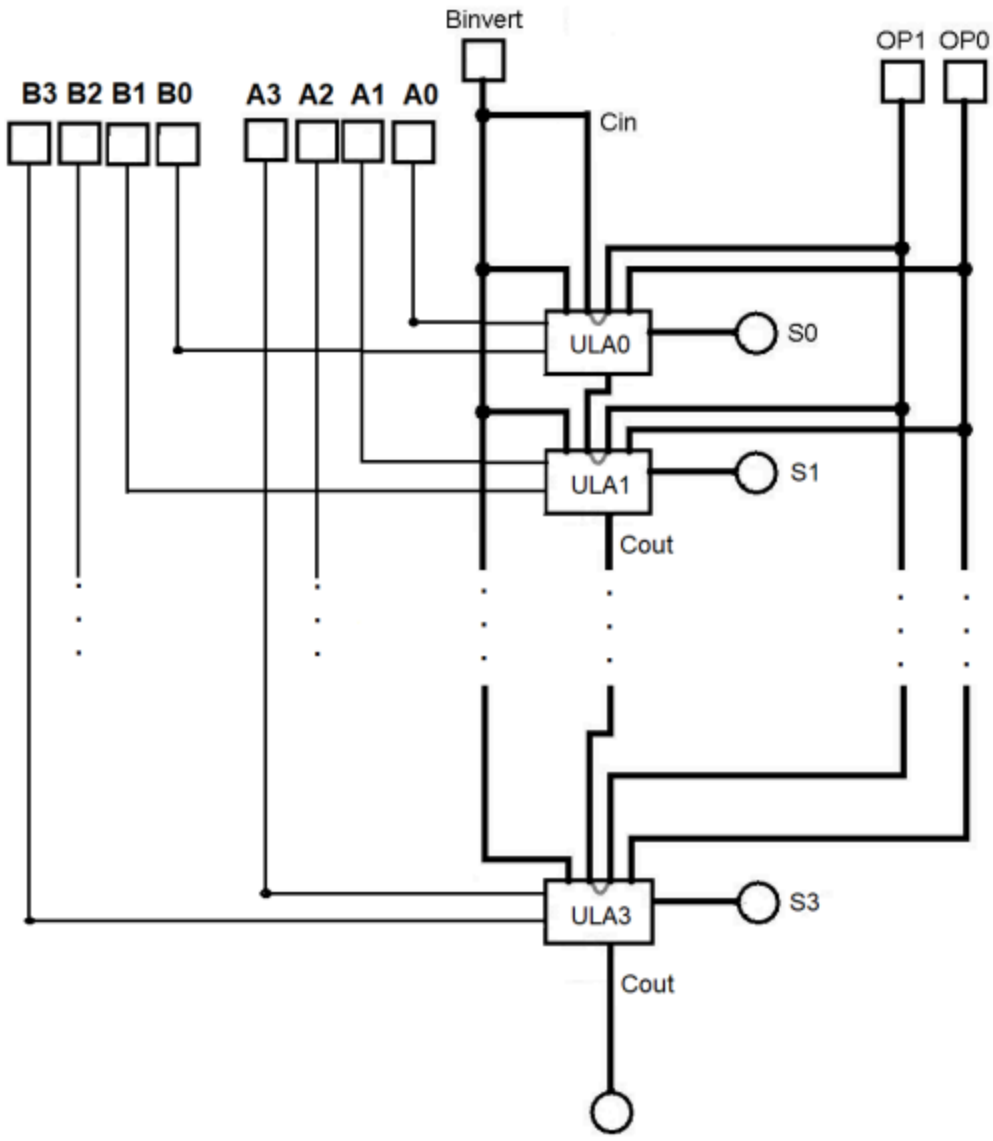
2. Procure entender o esquema, principalmente a subtração.
3. Sua ULA possui a seguinte tabela de opcodes:

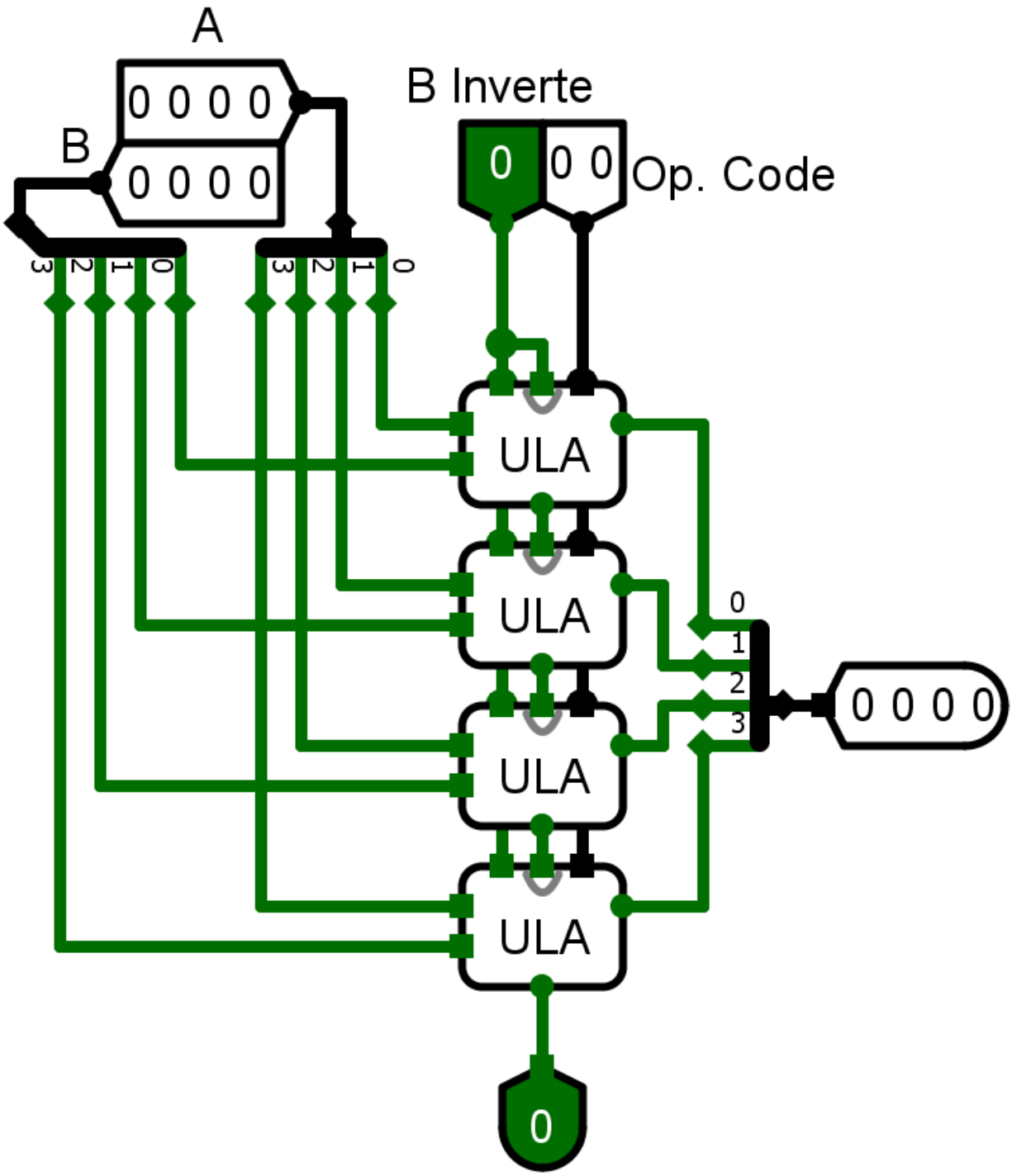
Op. Code (Operation)	Instrução (Result)
0	AND (a,b)
1	OR (a,b)
2	NOT (a)
3	SOMA(a,b)


B Inverte
Carry In
0
0
0 0
A
0
Decodificador
0
S
B
0
Mux
Somador
0
Carry Out

4. Teste a sua ULA de acordo com o seguinte roteiro:
   Inicio:
   A=0;
   B=1;
   AND(A,B);
   A=1;
   B=1;
   OR(A,B);
   SOMA(A,B);
   NOT(A);
   SOMA (A,-B);
   Fim.

5. Usando essa ula de 1 bit, construa essa ULA para **4 bits** no Logisim e verifique o seu funcionamento. Veja como funciona o barramento de instruções (operation) e o barramento de dados (a e b). Observe a ligação do Binvert ao Carry_in da primeira ULA. Procure usar subcircuitos, seu circuito deverá estar como a figura a seguir:




A
0 0 0 0
B
0 0 0 0
B Inverte
0
0 0
Op. Code
3
2
1
0
3
2
1
0
ULA
ULA
ULA
ULA
0
1
2
3
0 0 0 0
0

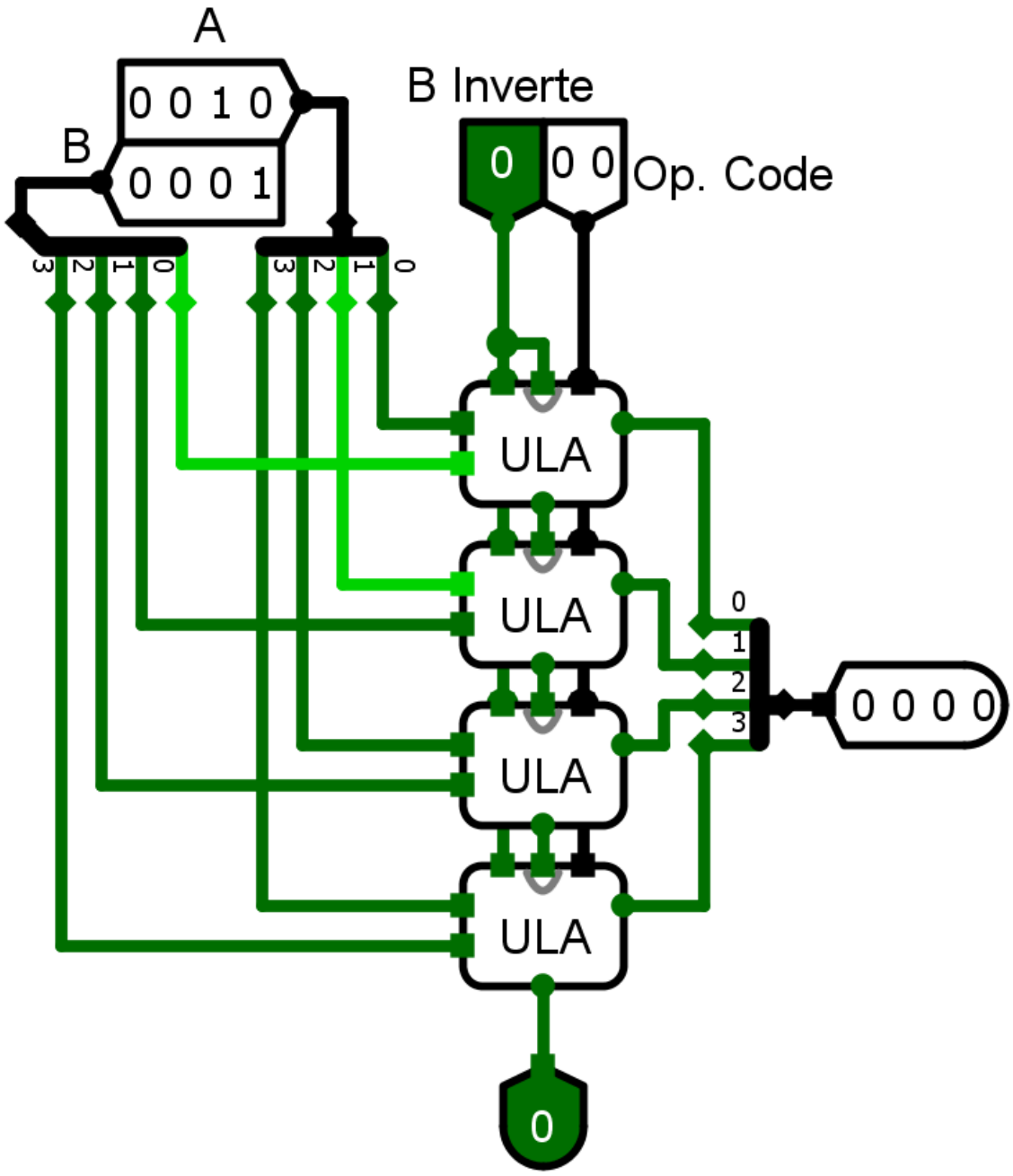
6. Teste a sua ULA de acordo com o seguinte roteiro (considerando os números de 4 bits):

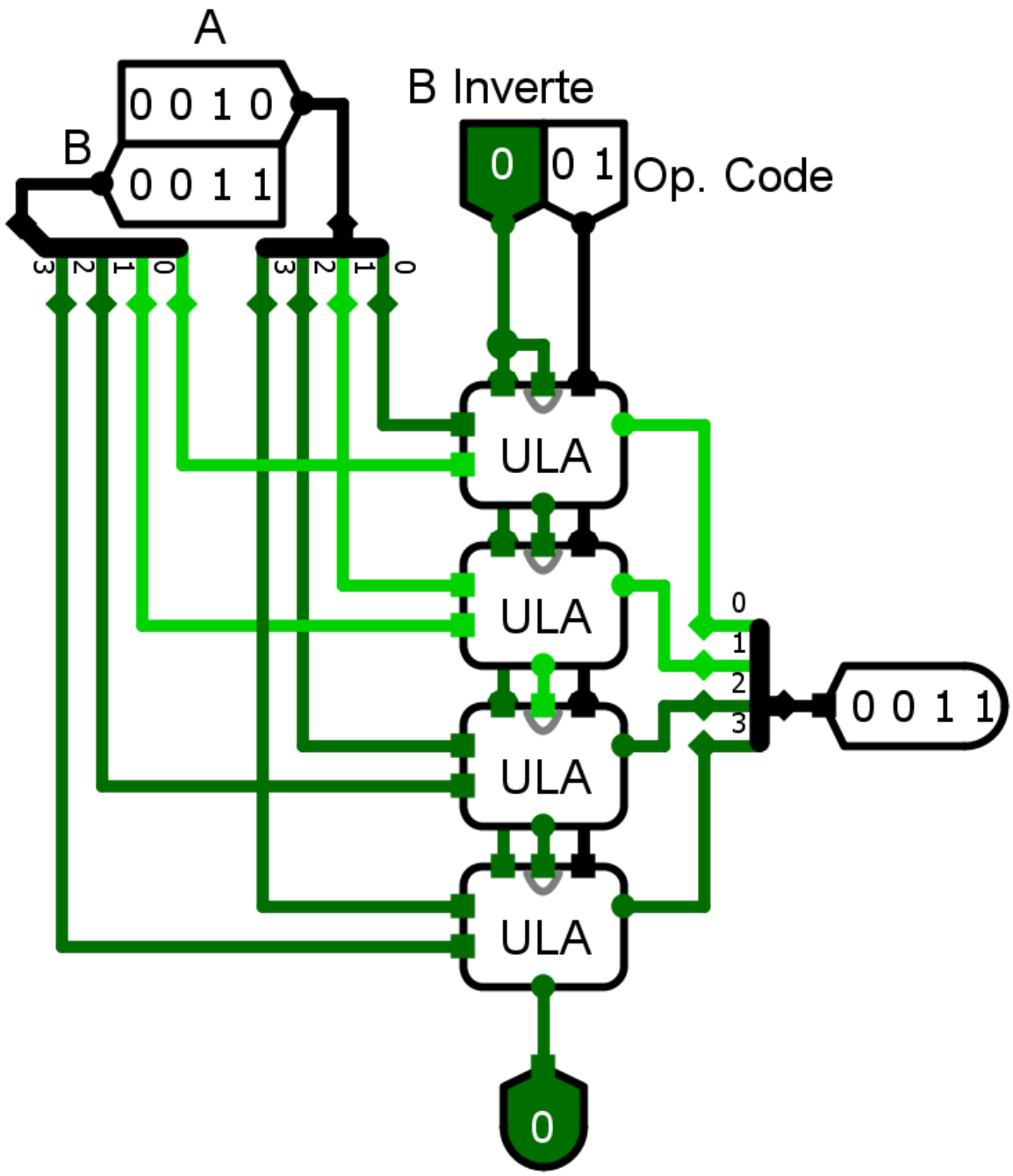
Inicio:
A=2; (ou A=0010)
B=1; (ou B=0001)
AND(A,B);
B=3; (ou B=0011)
OR(A,B);
SOMA(A,B);
A=12; (ou A=1100)
NOT(A);
B=13; (ou B=1101)
AND(B,A);
Fim.

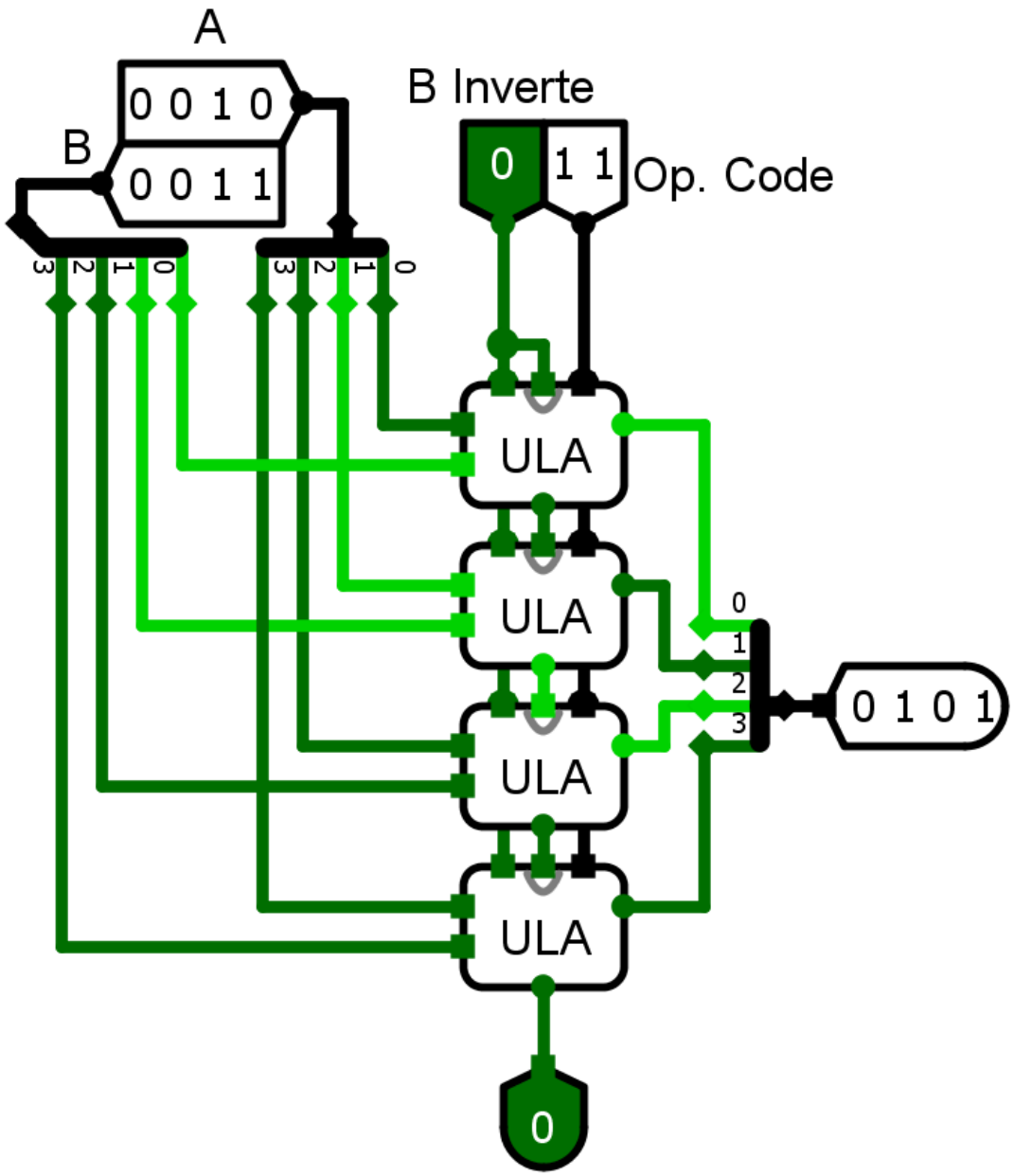
Para o programa de teste acima, preencher a tabela a seguir considerando que cada linha corresponderá à execução de uma instrução (a primeira linha já foi realizada, observe que a palavra deverá conter 10 bits, para escrevermos em hexa completamos os dois bits à esquerda com zero):

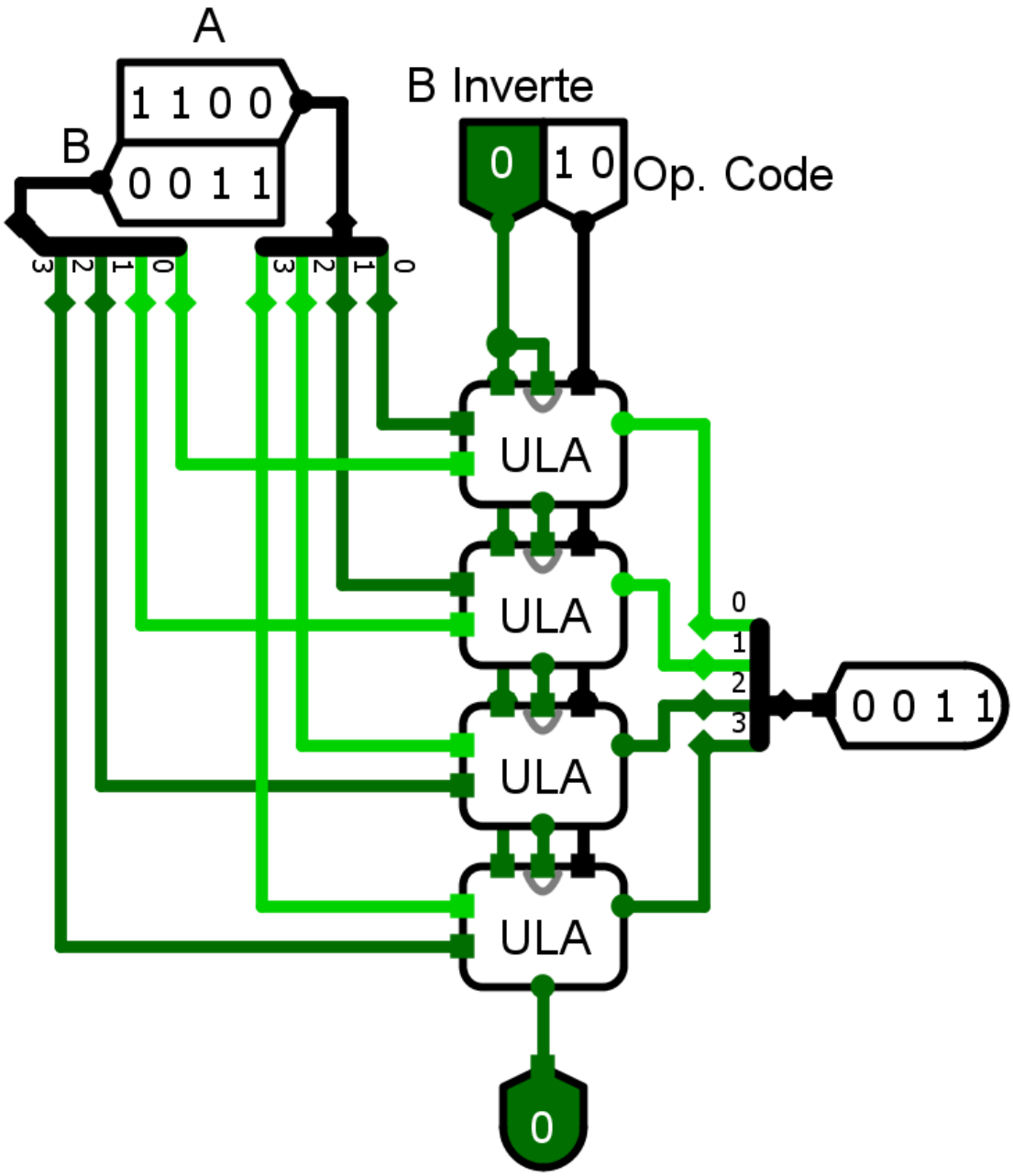
Para o programa de teste acima, preencher a tabela a seguir considerando que cada linha corresponderá à execução de uma instrução (a primeira linha já foi realizada, observe que a palavra deverá conter 10 bits, para escrevermos em hexa completamos os dois bits à esquerda com zero):

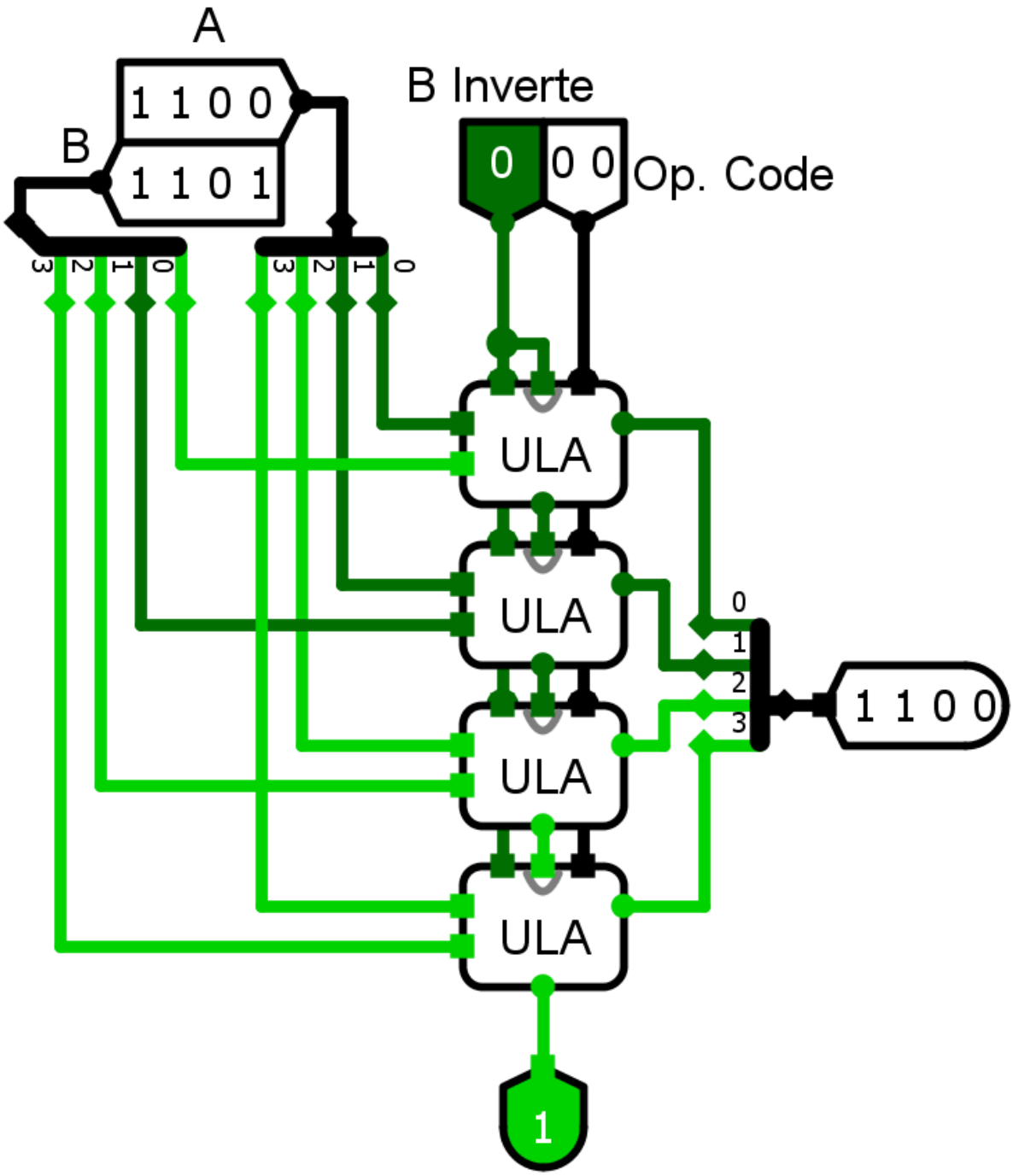
Instrução realizada	Binário (A,B,Op.code)	Valor em Hexa (0x …)	Resultado em binário
AND(A,B)	0010 0001 00	(0000 1000 0100) = 0x084	0000
OR(A,B)	0010001101	(000010001101) = 0x08D	0011
SOMA(A,B)	0010001111	(000010001111) = 0x08F	0101
NOT(A)	1100001110	(001100001110) = 0x30E	0011
AND(B,A)	1100110100	(001100110100) = 0x334	1100


A
0 0 1 0
B
0 0 0 1
B Inverte
0
0 0
Op. Code
3
2
1
0
3
2
1
0
ULA
ULA
ULA
ULA
0
1
2
3
0 0 0 0
0


A
0 0 1 0
B
0 0 1 1
B Inverte
0
0 1
Op. Code
3
2
1
0
3
2
1
0
ULA
ULA
ULA
ULA
0
1
2
3
0 0 1 1
0

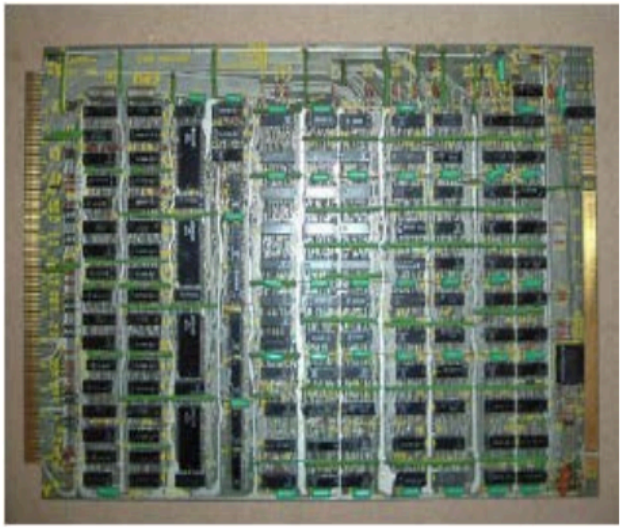

A
0 0 1 0
B
0 0 1 1
B Inverte
0
1 1
Op. Code
3
2
1
0
3
2
1
0
ULA
ULA
ULA
ULA
0
1
2
3
0 1 0 1
0

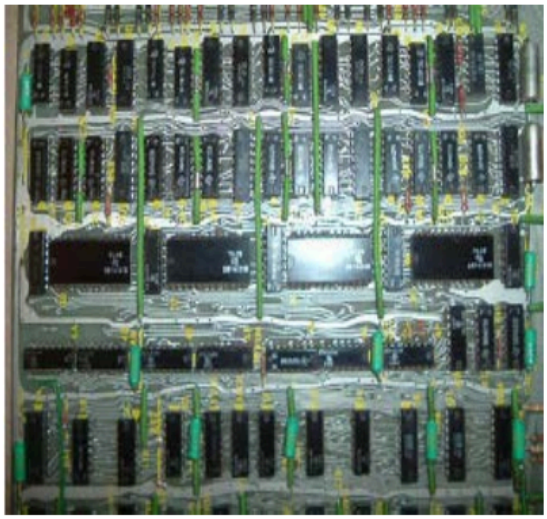

A
1 1 0 0
B
0 0 1 1
B Inverte
0
1 0
Op. Code
3 2 1 0
3 2 1 0
ULA
ULA
ULA
ULA
0
1
2
3
0 0 1 1
0

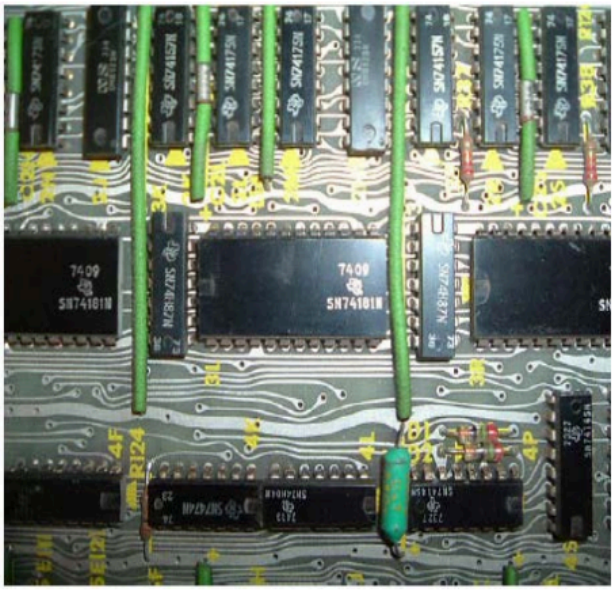

A
1 1 0 0
B
1 1 0 1
B Inverte
0
0 0
Op. Code
3
2
1
0
3
2
1
0
ULA
ULA
ULA
ULA
0
1
2
3
1 1 0 0
1

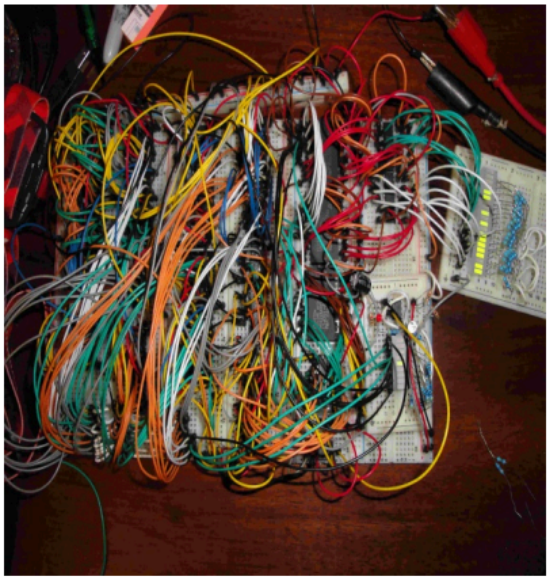
Parte 2

Nesta experiência você irá projetar no logisim o circuito 74181, que foi inicialmente utilizado para a construção de computadores de 8 e 16 bits (conforme as figuras abaixo). Posteriormente iremos implementar uma ULA semelhante dentro do Arduino, por isso é importante conhecê-la.







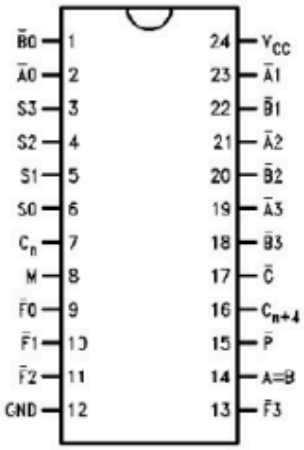


Como a ULA funciona.

A ULA a ser utilizada é a 74LS181, que possui 4 bits de controle e é uma ULA de 4 bits (saída). Portanto, opera sobre duas entradas de 4 bits. A distribuição dos pinos pode ser vista a seguir:

SELECTION				ACTIVE-HIGH DATA		
				M = H	M = L; ARITHMETIC OPERATIONS	
S3	S2	S1	S0	LOGIC FUNCTIONS	$\overline{C_n}$ = H (no carry)	$\overline{C_n}$ = L (with carry)
L	L	L	L	$F = \overline{A}$	F = A	F = A PLUS 1
L	L	L	H	$F = \overline{A + B}$	F = A + B	F = (A + B) PLUS 1
L	L	H	L	$F = \overline{A}B$	$F = A + \overline{B}$	$F = (A + \overline{B})$ PLUS 1
L	L	H	H	F = 0	F = MINUS 1 (2's COMPL)	F = ZERO
L	H	L	L	$F = \overline{AB}$	$F = A$ PLUS $A\overline{B}$	$F = A$ PLUS $A\overline{B}$ PLUS 1
L	H	L	H	$F = \overline{B}$	F = (A + B) PLUS $A\overline{B}$	F = (A + B) PLUS $A\overline{B}$ PLUS 1
L	H	H	L	$F = A \oplus B$	F = A MINUS B MINUS 1	F = A MINUS B
L	H	H	H	$F = A\overline{B}$	$F = A\overline{B}$ MINUS 1	$F = A\overline{B}$
H	L	L	L	$F = \overline{A} + B$	F = A PLUS AB	F = A PLUS AB PLUS 1
H	L	L	H	$F = \overline{A \oplus B}$	F = A PLUS B	F = A PLUS B PLUS 1
H	L	H	L	F = B	$F = (A + \overline{B})$ PLUS AB	$F = (A + \overline{B})$ PLUS AB PLUS 1
H	L	H	H	F = AB	F = AB MINUS 1	F = AB
H	H	L	L	F = 1	F = A PLUS A	F = A PLUS A PLUS 1
H	H	L	H	$F = A + \overline{B}$	F = (A + B) PLUS A	F = (A + B) PLUS A PLUS 1
H	H	H	L	F = A + B	$F = (A + \overline{B})$ PLUS A	$F = (A + \overline{B})$ PLUS A PLUS 1
H	H	H	H	F = A	F = A MINUS 1	F = A

Connection Diagram



Pin Descriptions

Pin Names	Description
$\overline{A0}$–$\overline{A3}$	Operand Inputs (Active LOW)
$\overline{B0}$–$\overline{B3}$	Operand Inputs (Active LOW)
S0–S3	Function Select Inputs
M	Mode Control Input
C_n	Carry Input
$\overline{F0}$–$\overline{F3}$	Function Outputs (Active LOW)
A = B	Comparator Output
G	Carry Generate Output (Active LOW)
$\overline{P}$	Carry Propagate Output (Active LOW)
C_{n+4}	Carry Output

Nessa primeira parte do experimento você deverá implementar toda a ULA no Logisim. Esta ULA permite a execução de instruções lógicas e aritméticas e permite que usemos entradas ativas em nível alto e em nível baixo. **Atenção:** Usaremos entradas em **nível alto**, conforme a tabela de funções ilustrada e iremos utilizar **apenas as instruções lógicas.**

O próximo passo será um projeto no LOGISIM dessa mesma ULA.

Agora você poderá utilizar os componentes presentes no Logisim (MUX, somadores, portas de múltiplas entradas, etc). Você também deverá utilizar o conceito de barramento para cada entrada e/ou saída, isso evitará um número muito grande de conexões.

Para um teste, você deverá dar um valor para A, um valor para B e executar todas as funções que a ALU permite através de S0, S1, S2 e S3. A saída da ALU deverá ser verificada nos pinos F0, F1, F2 e F3.

Iremos testar todas as funções lógicas da ULA da seguinte forma:
- criaremos uma palavra de 12 bits (os primeiros 4 bits para A0, A1, A2 e A3), os próximos 4 bits para B (B0, B1, B2 e B3) e os 4 bits finais para a operação desejada (S0, S2 e S3). O valor a ser preenchido da tabela será o resultado da operação.

Exemplo:

Instrução	Binário	Resultado da operação
4CB	010011001011	4

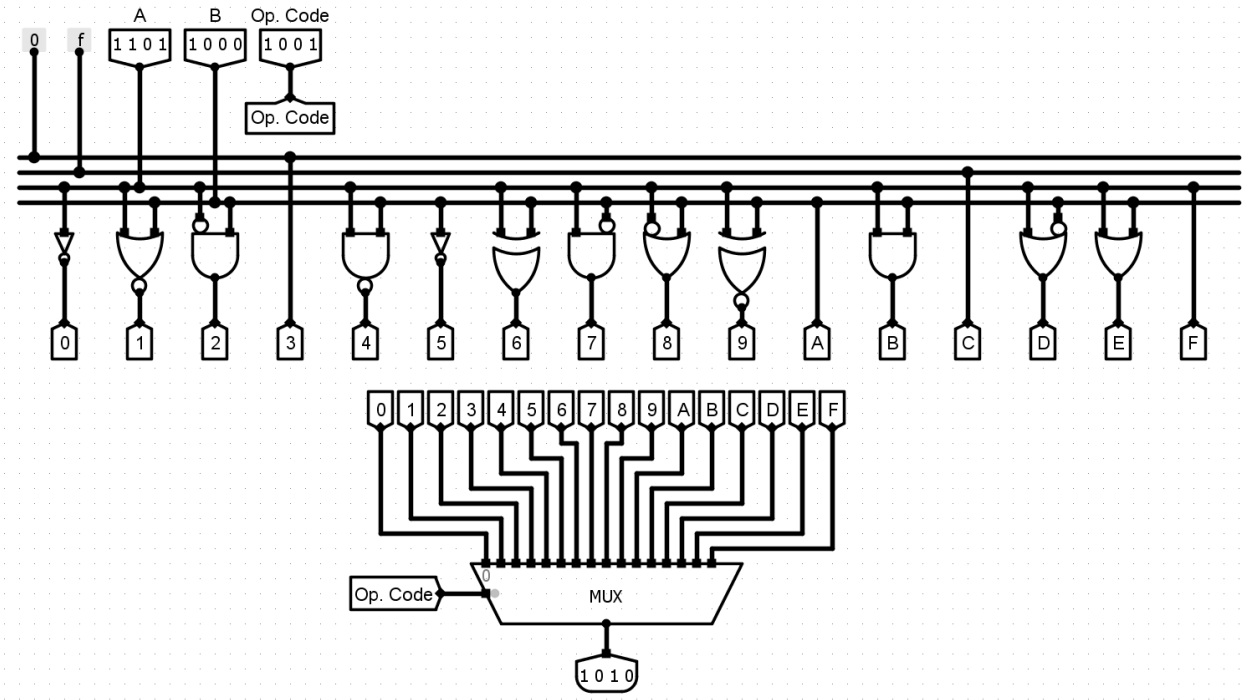
O significado da instrução é o seguinte (observe que escrevemos os valores em Hexadecimal para simplificar):
- O valor de A = 4 (ou **0100** em binário)
- O valor de B = C (ou **1100** em binário)
- A operação será B (ou **1011** em binário)

O que deveremos então fazer será a operação **1011** sobre os dados **0100** (que é o valor de A ou o primeiro operando) sobre **1100** (que é o valor de B ou o segundo operando).

Quando olhamos na tabela da ULA, a operação 1011 (ou H L H H) corresponde a F = AB, ou seja, a saída da ULA será o AND de A com B. Como A=0100 e B=1100, o AND de A e B será 0100, que é o resultado da operação e que deverá ser colocado na tabela (0100 = 4).

Complete agora a tabela a seguir onde todas as instruções que a ULA pode fazer serão testadas.

Instruções	Binário	Resultado da operação
450	010001010000	0xB
CB1	110010110001	0x2
A32	101000110010	0x1
C43	110001000011	0x0
124	000100100100	0xF
785	011110000101	0x7
9B6	100110110100	0x2
CD7	110011010111	0x0
FE8	111111101000	0xE
649	011001001001	0xD
D9A	110110011010	0x9
FCB	111111001011	0xC
63C	011000111100	0xF
98D	100110001101	0xF
76E	011101101110	0x7
23F	001000111111	0x2



Se o objetivo fosse realmente testar esta ULA teríamos que testar TODAS as possibilidades, como temos 4 Bits na entrada A, 4 Bits na Entrada B e 4 Bits de Seleção teríamos que testar 2^12 combinações, ou seja, 4096 possibilidades.