

# ARQUITETURA E ORGANIZAÇÃO DE COMPUTADORES



Processador de 16bits  
**JUNIN**

# “*Introdução*”

Este trabalho aborda o processo de implementação de um processador de 16 bits. A arquitetura desse processador envolve a manipulação de dados e instruções em unidades de 16 bits, permitindo operações mais complexas do que processadores de 8 bits, embora com menor capacidade que arquiteturas de 32 ou 64 bits.

# Jornada de Produção

**Fase 1:**  
Escolha da  
plataforma

**Fase 2:**  
Montagem do  
Sistema

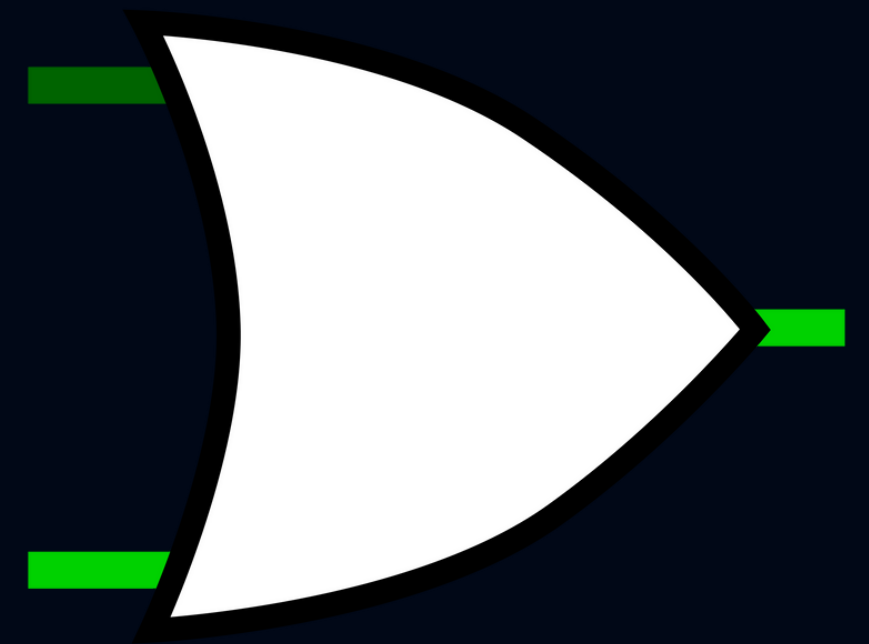
**Fase 3:**  
Testes do  
Sistema

**Fase Final:**  
Produção do relatório e  
apresentação

# Onde foi feito?

O PROCESSADOR FOI FEITO

NA IDE: LOGSIM



# Conjunto de Instruções

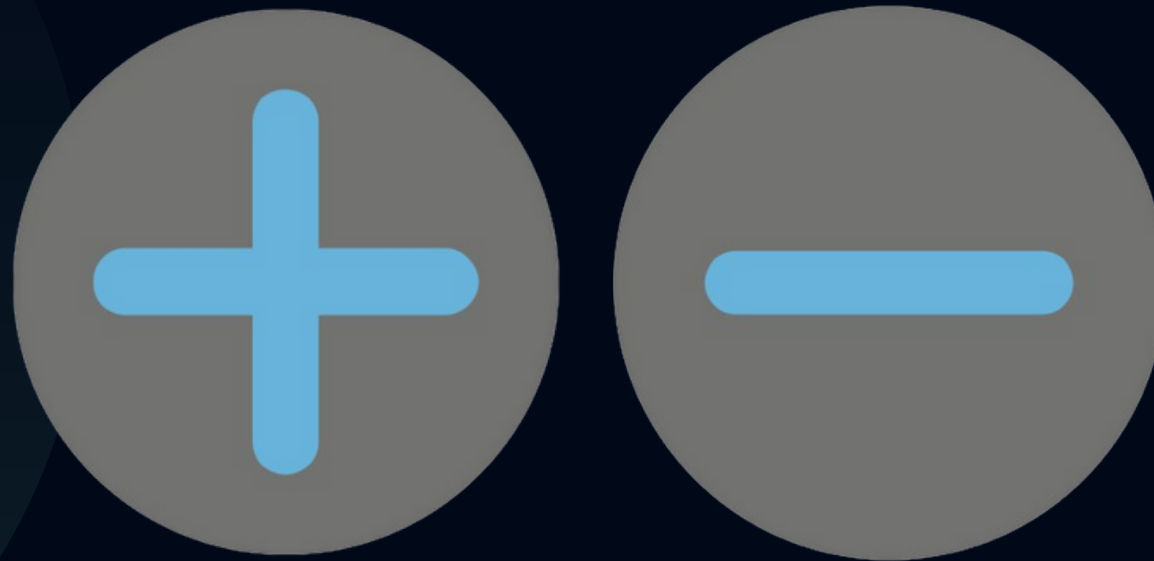
TIPO R				
OPCODE	REG1	REG2	REG3	SHAMT
15-13	12-9	8-5	4-1	0

TIPO I				
OPCODE	REG1	REG2	REG3	SHAMT
15-13	12-9	8-5	4-1	0

TIPO J	
OPCODE	ENDEREÇO
15-13	12-0

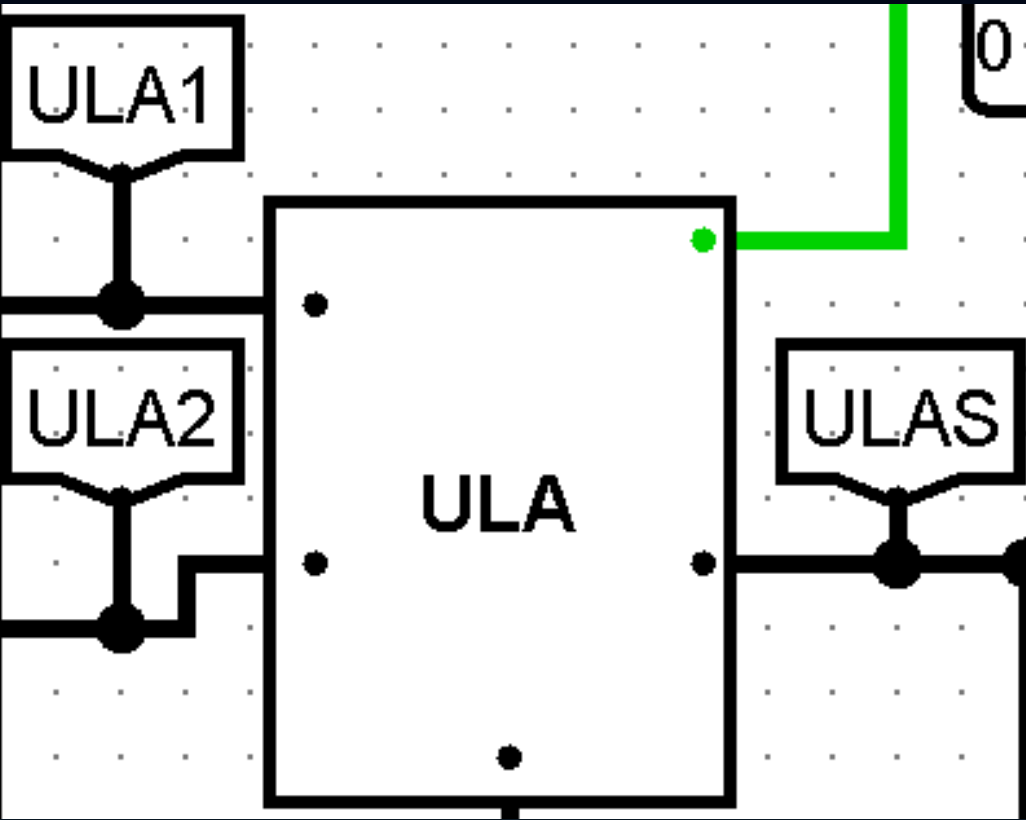
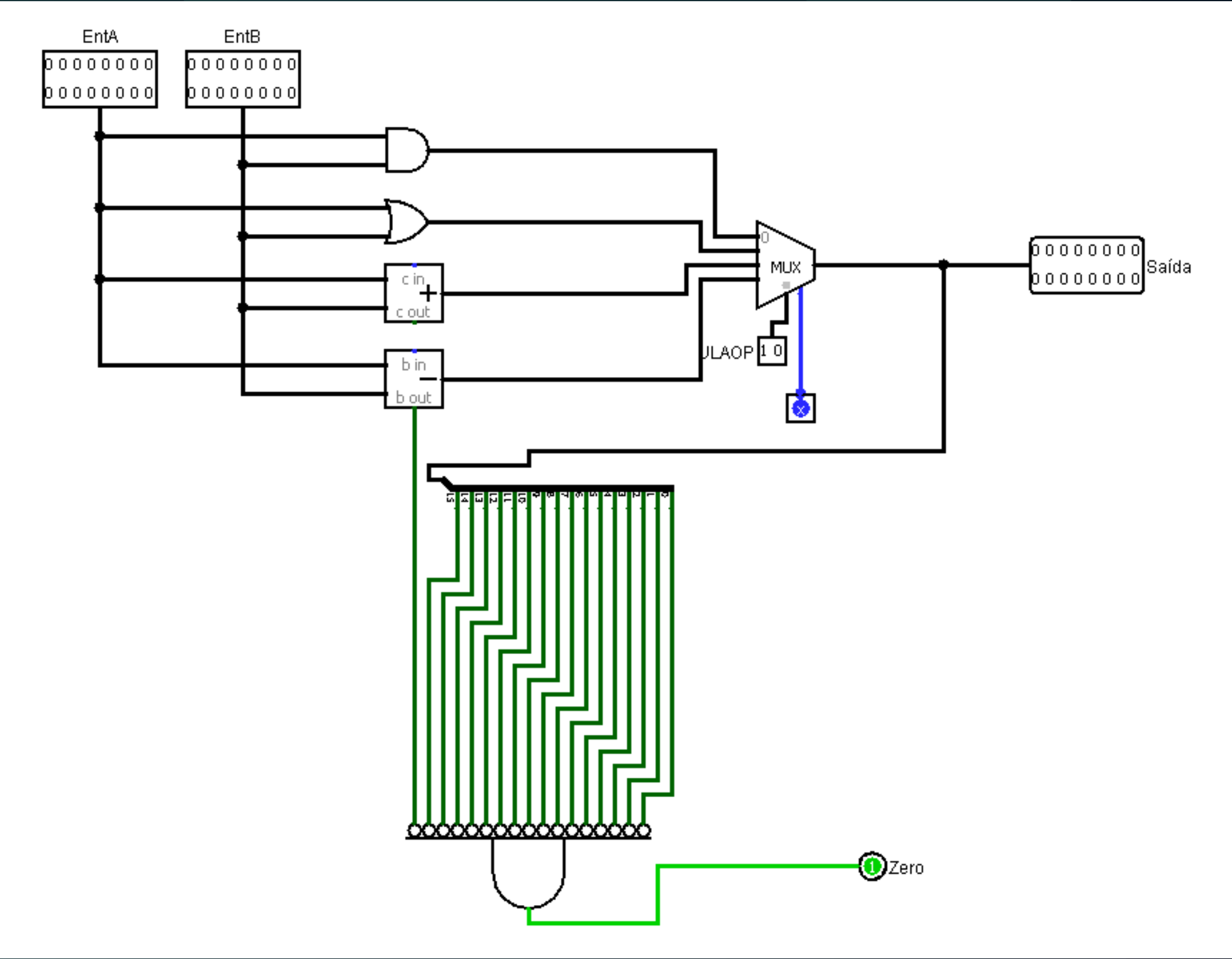
# Conjunto de Instruções

- **ADD**
- **SUB**
- **AND**
- **OR**



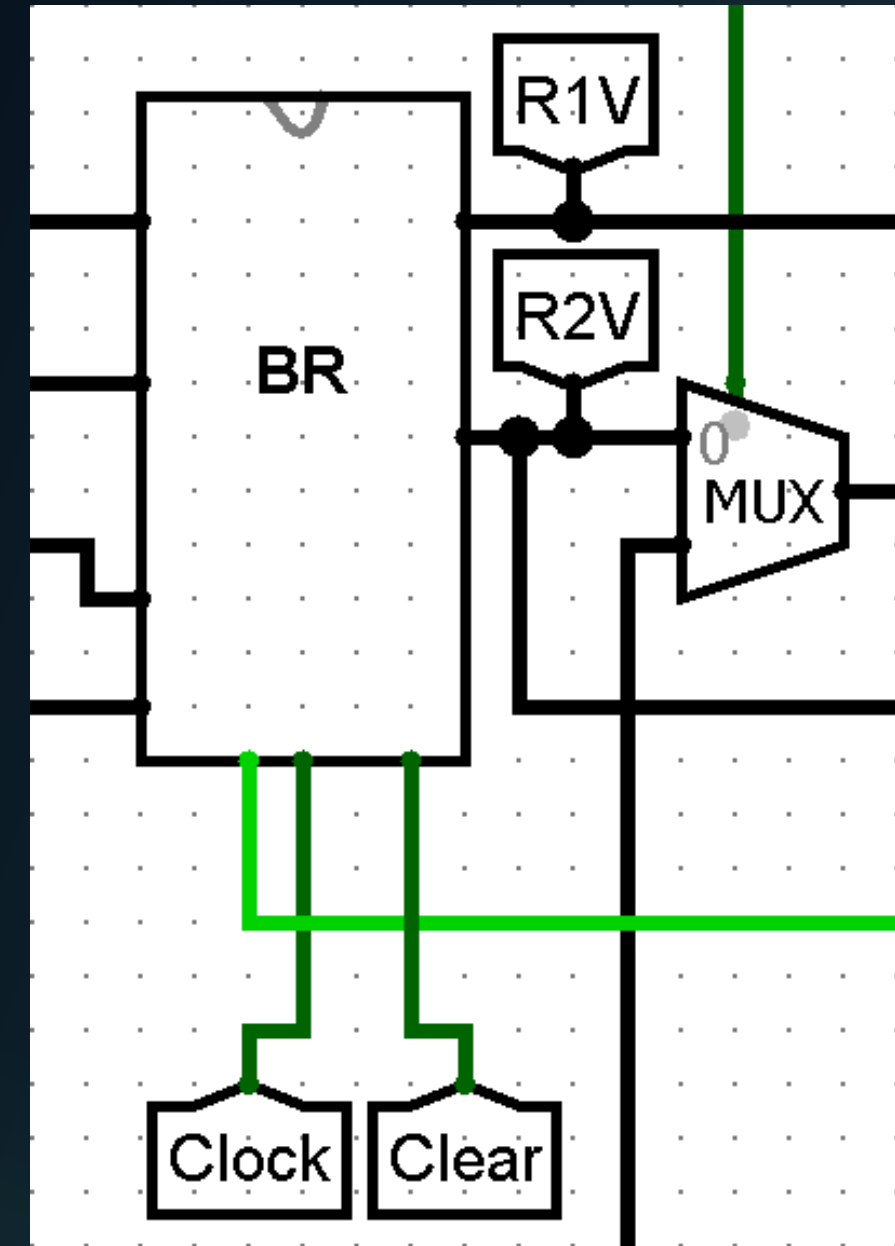
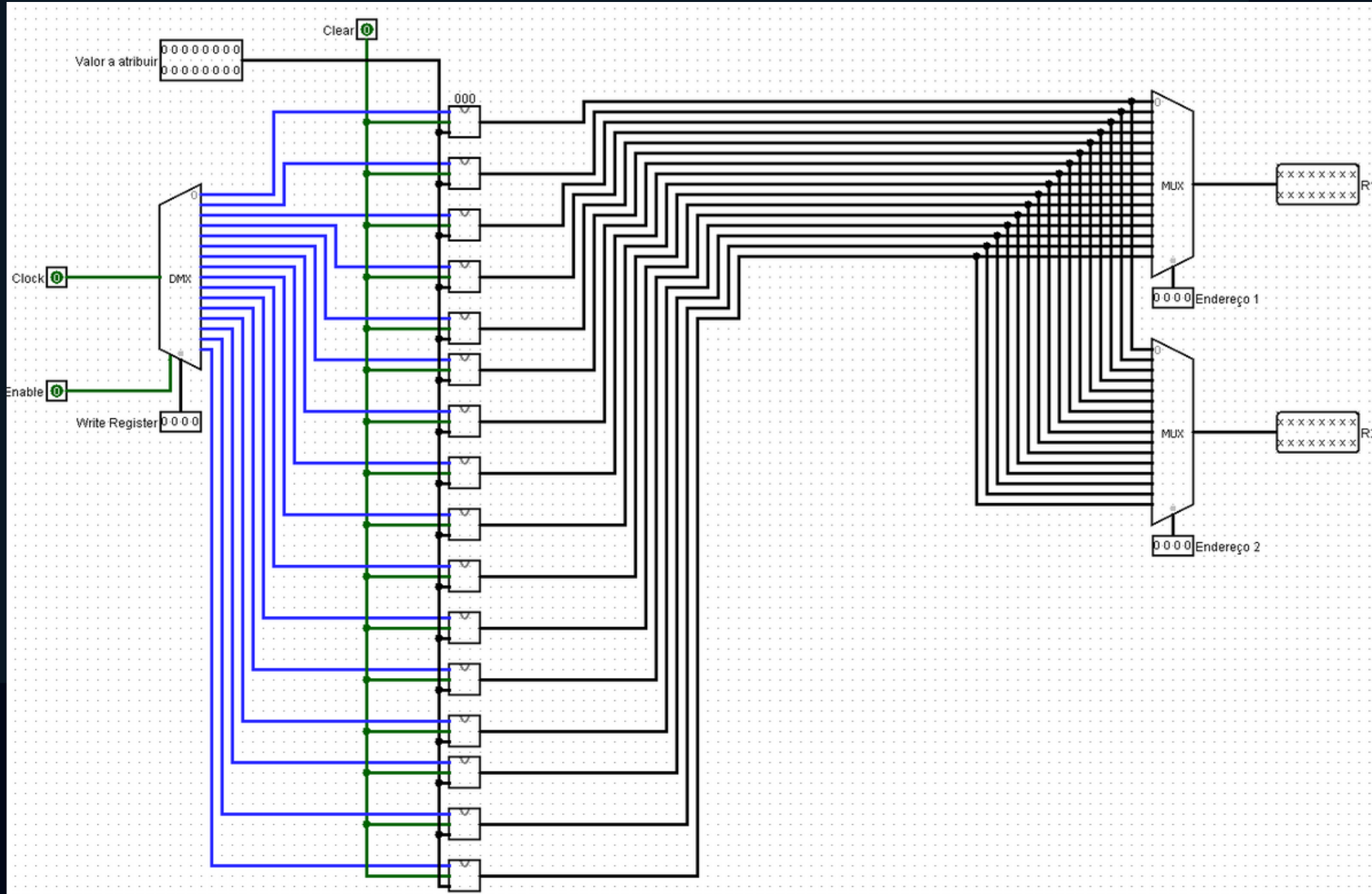
- **LOAD**
- **STORE**
- **JUMP**
- **BEQ**

# ULA



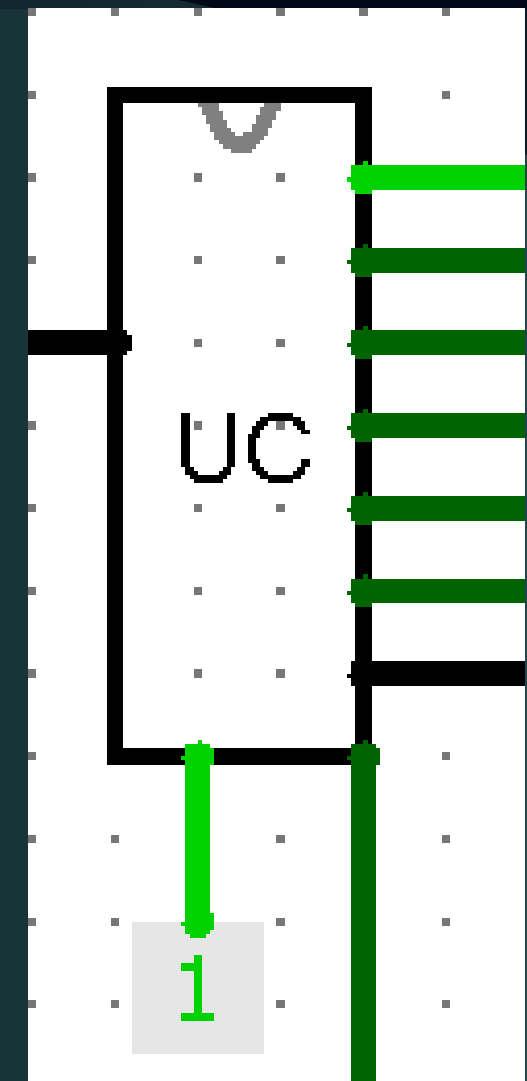
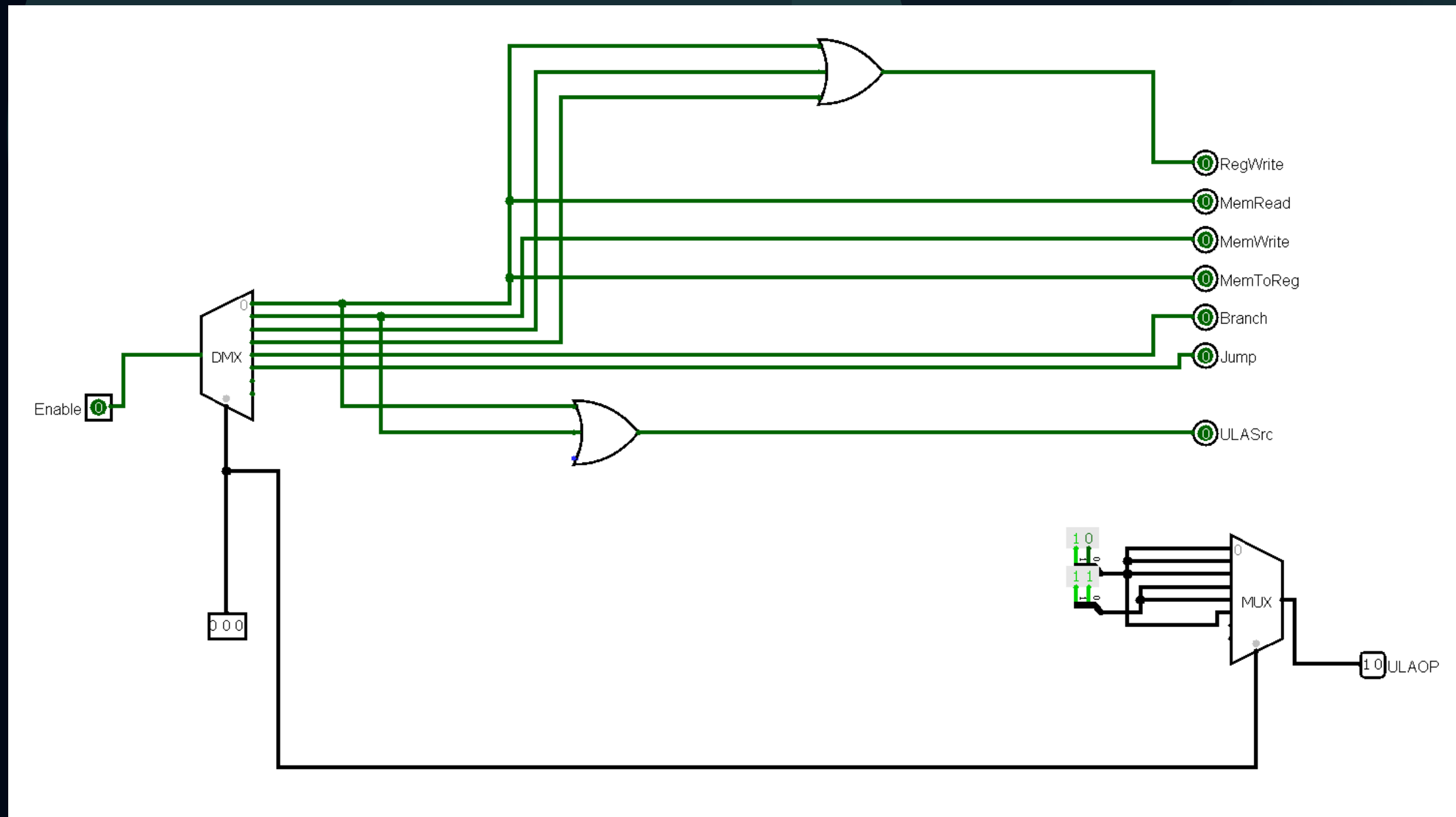
OP ULA	OPERAÇÃO
00	AND
01	OR
10	ADD
11	SUB

# Banco de Resgitradores



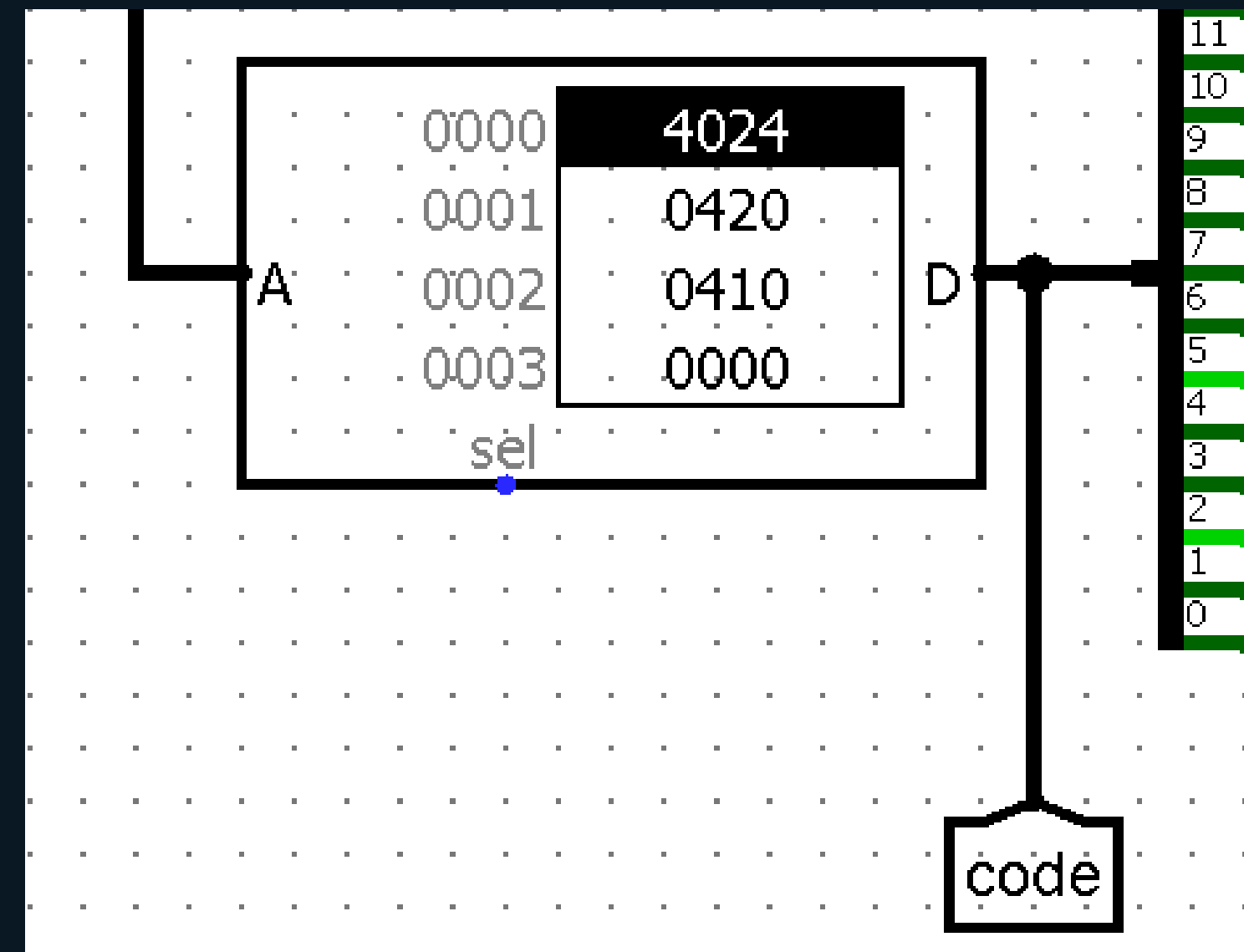
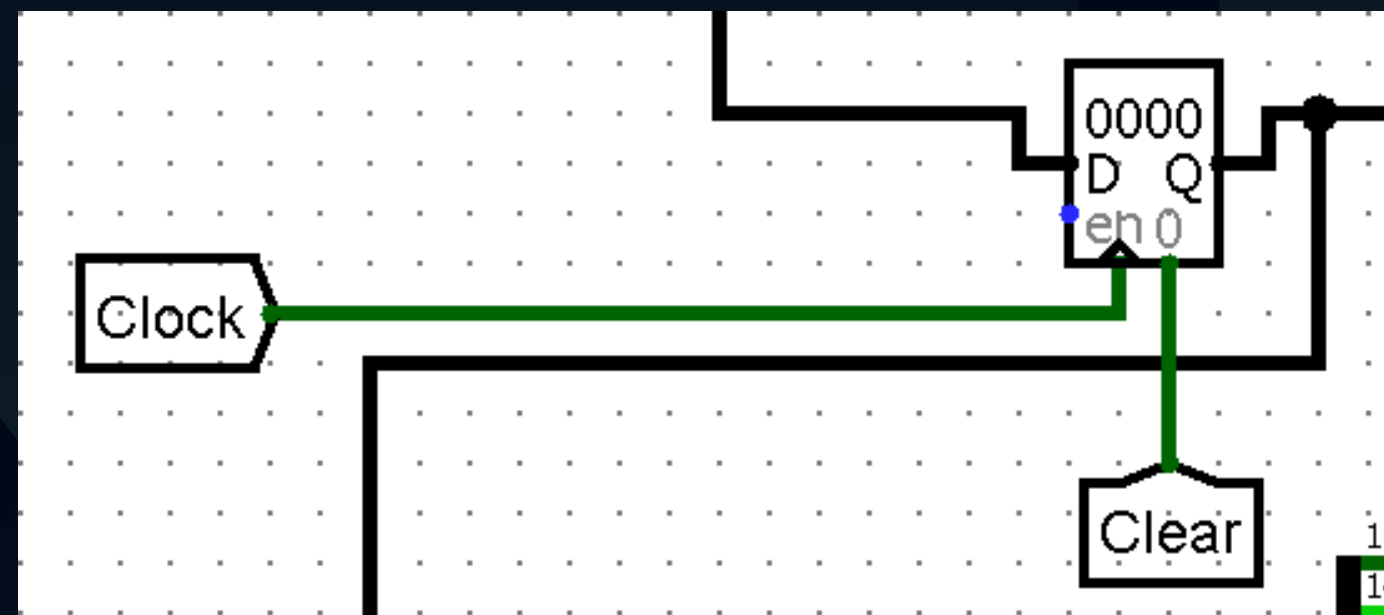


# Unidade de Controle

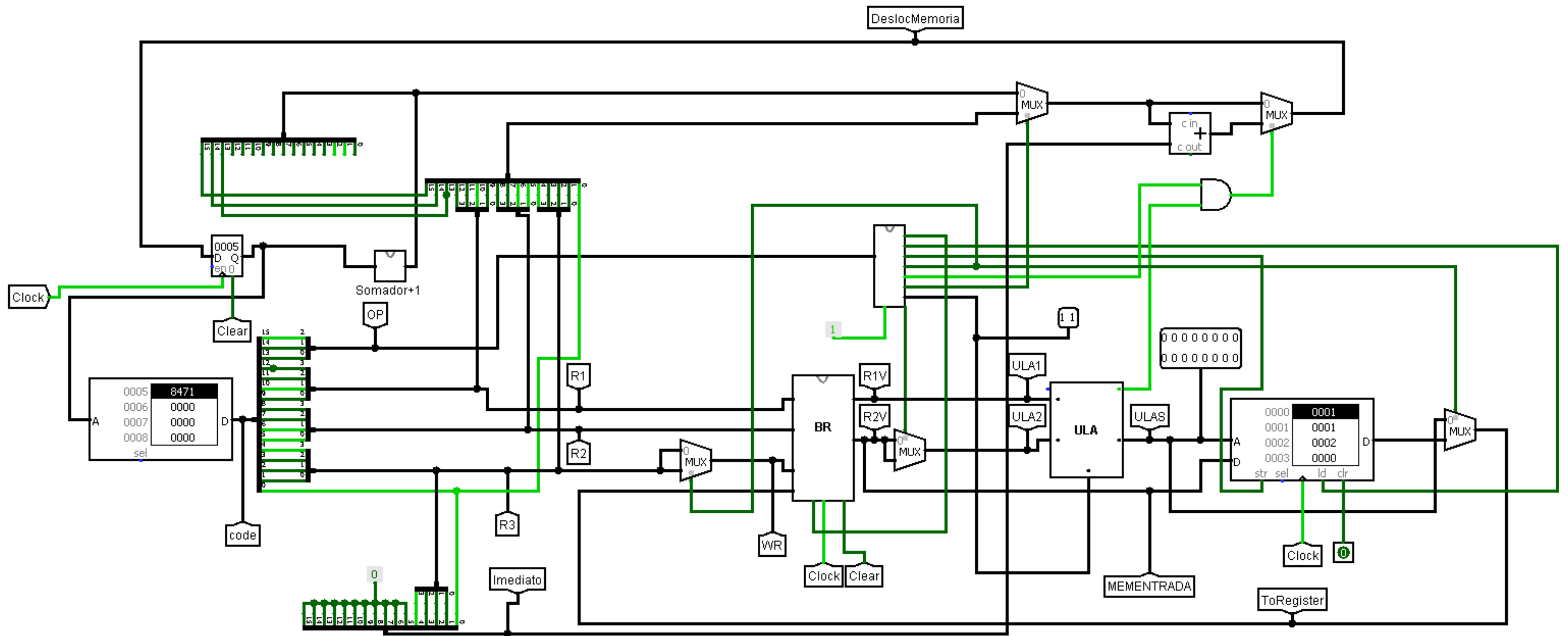


- 1 RegWrite**
- 2 MemRead**
- 3 MemWrite**
- 4 MemToReg**
- 5 Branch**
- 6 Jump**
- 7 ULASrc**
- 8 ULAOP**

# PC E Memória ROM



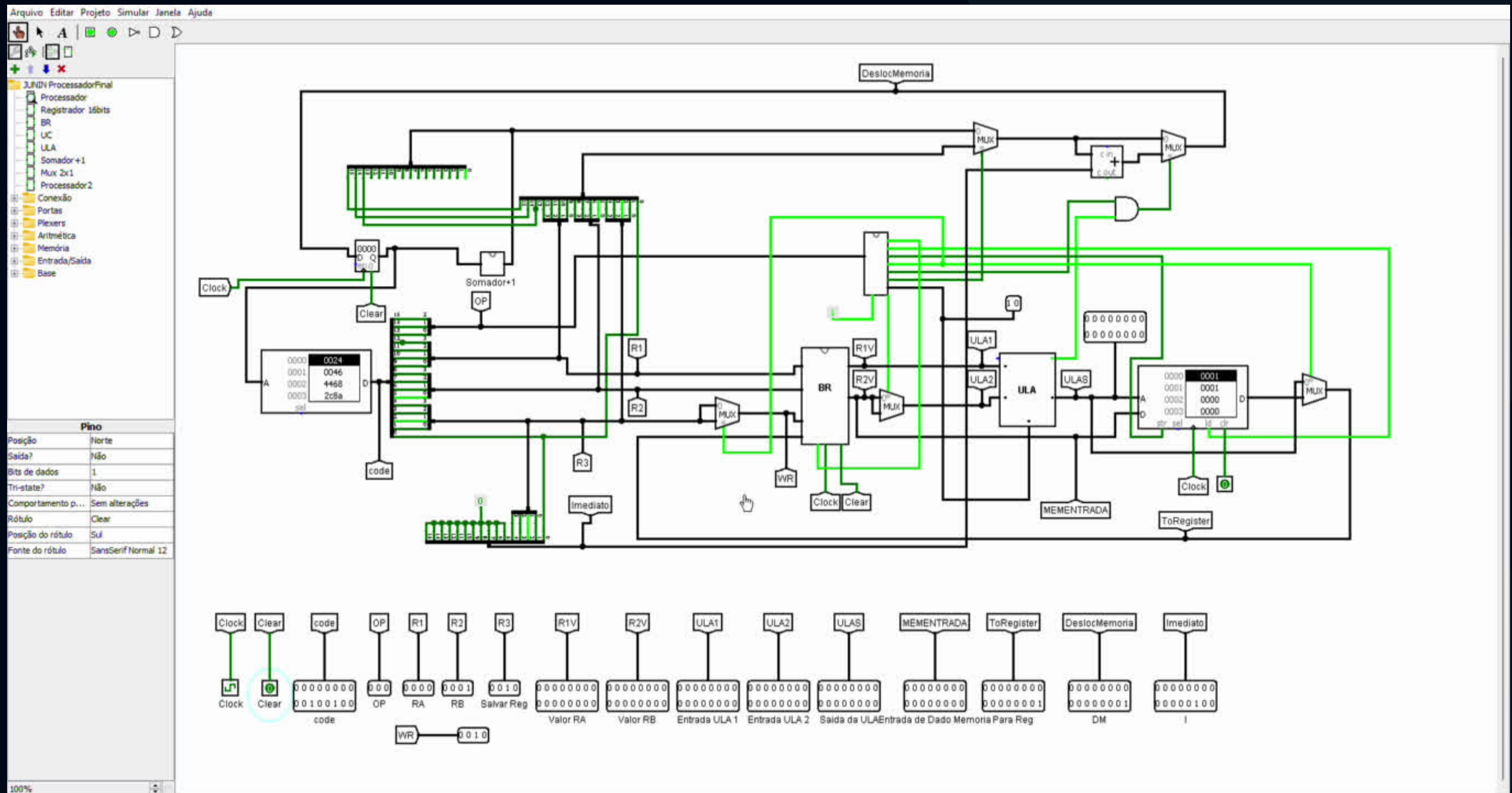
# Datapath



# Teste - Tradução

Endereço	Linguagem de Alto Nível	Binário				
		Opcode	Reg1	Reg2	Reg3	Shamt
			Endereço			
		Dado				
0000	lw \$R3, \$R0, \$R1	000	0000	0001	0010	0
0001	lw \$R4, \$R0, \$R3	000	0000	0010	0011	0
0002	add \$R5, \$R3, \$R4	010	0010	0011	0100	0
0003	sw \$R6, \$R7, \$R5	001	0110	0100	0101	0
0004	sub \$R8, \$R6, \$R3	011	0101	0010	0111	0
0005	beq \$R3, \$R4, \$R9	100	0010	0011	1000	1
0017	jump 0031	101	0000000110001			
0031	jump 0005	101	00000000000101			
0005	beq \$R3, \$R4, \$R9	100	0010	0011	1000	1
0017	jump 0031	101	0000000110001			
0031	jump 0005	101	00000000000101			

# Teste - Resultado



# Obrigado!

