UNIVALI

Laboratório de Sistemas
Embarcados e Distribuídos

Organização do RISC-V:
Monociclo

Histórico de revisões

Revisão	Data	Responsável	Descrição
0.1	- X -	Prof. Cesar Zeferino	Primeira versão
0.2	03/2016	Prof. Cesar Zeferino	Revisão do modelo e atualização de conteúdo
0.3	05/2020	Prof. Cesar Zeferino	Revisão geral
0.4	10/2022	Felski	Revisão geral da Arquitetura

Introdução

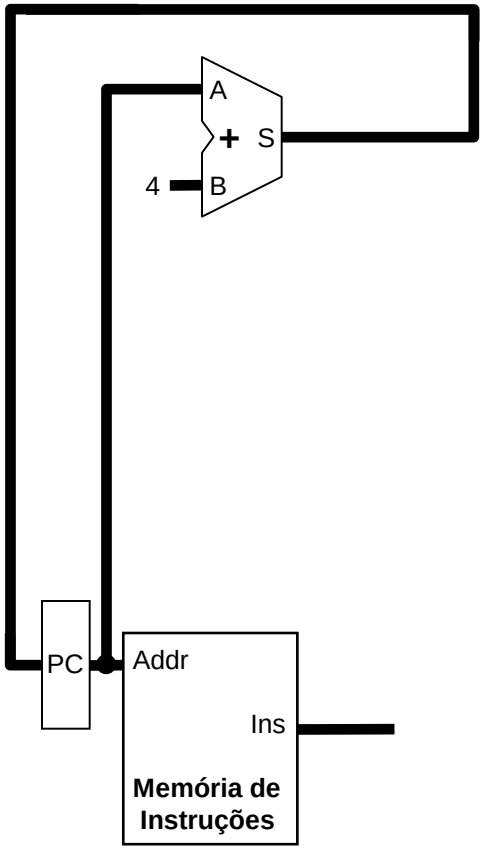
- **Objetivo**
  - Descrever o caminho de dados do RISC-V monociclo e projetar o controle desse processador
- **Conteúdo**
  - Projeto e funcionamento do caminho de dados do RISC-V monociclo
  - Projeto do controle do RISC-V monociclo

Introdução

Bibliografia

- PATTERSON, David A.; HENNESSY, John L. O Processador. *In*: _____. **Organização e projeto de computadores**: a interface hardware/software. 4. ed. Rio de Janeiro: Campus, 2014. cap. 4.

Caminho de dados



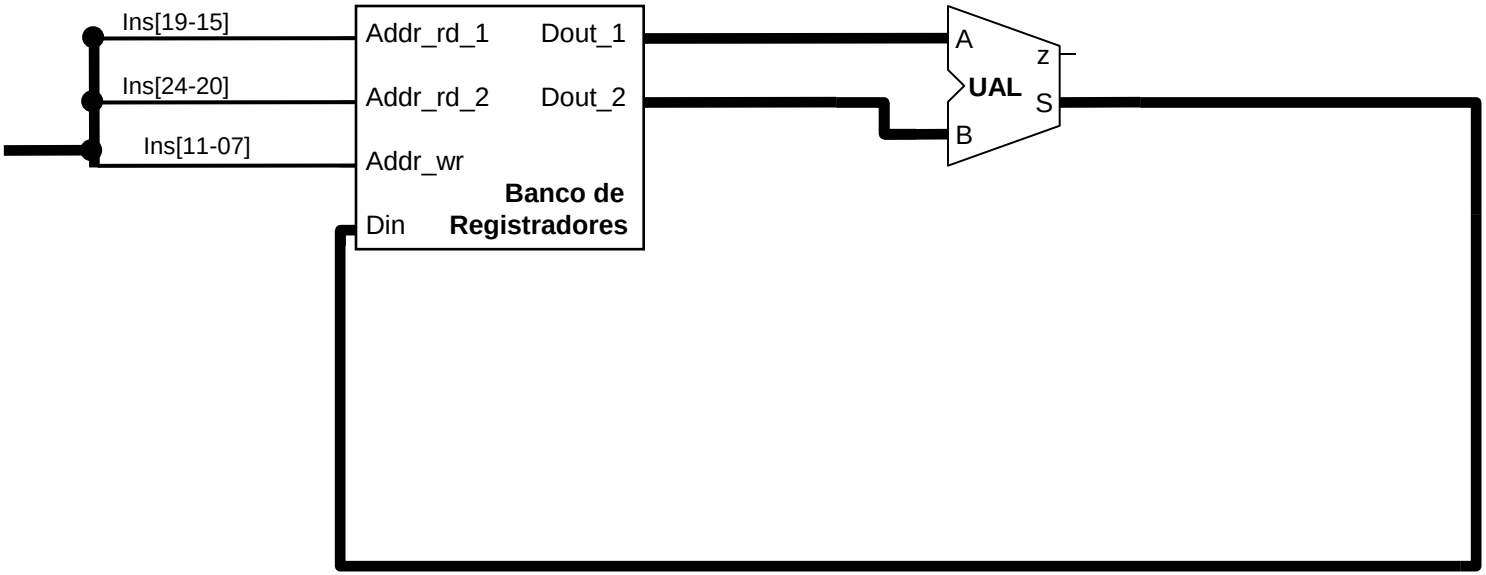
Caminho de dados para a busca de instrução
(Não considerando as operações de desvio)

A cada instrução executada, o PC é incrementado em 4 unidades para apontar para a instrução seguinte

Caminho de dados

Caminho de dados para instruções do tipo R

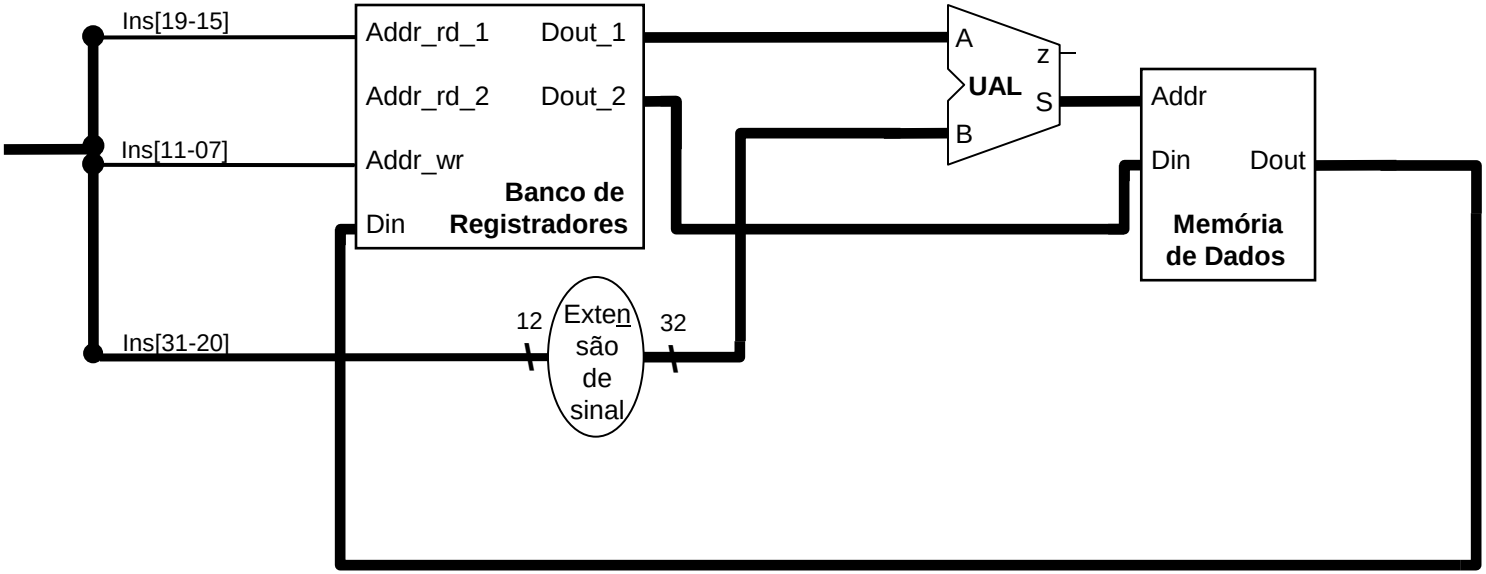
A UAL opera sobre dados lidos do Banco de Registradores e o resultado é escrito de volta no banco



Caminho de dados

Caminho de dados para instruções *lw*

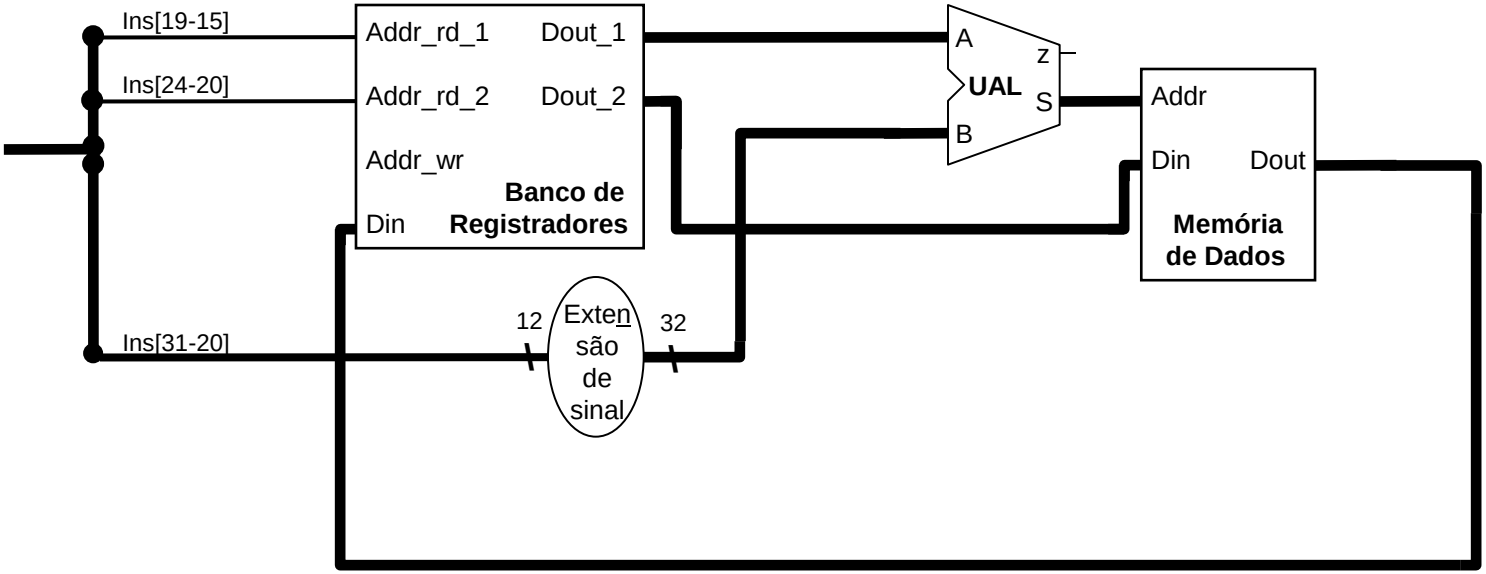
A UAL calcula o endereço a ser acessado na memória somando rs1 (*Addr_rd_1) com o imediato, após a extensão do sinal. A transferência ocorre entre a memória e o rd, onde:
:: rd = Addr_wr, para um *load*



Caminho de dados

Caminho de dados para instruções *sw*

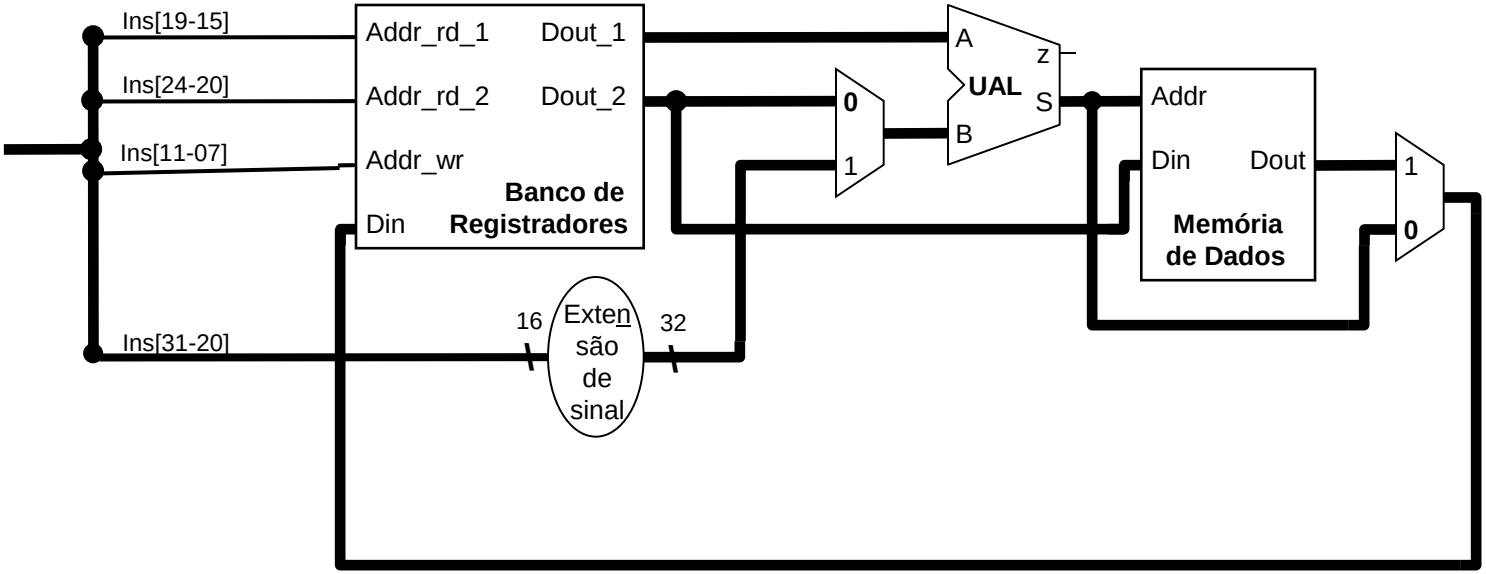
A UAL calcula o endereço a ser acessado na memória somando rs1 (*Addr_rd_1) com o imediato, após a extensão do sinal. A transferência ocorre entre a memória e o rs2, onde:
:: rs2 = Din, para um *store*



Caminho de dados

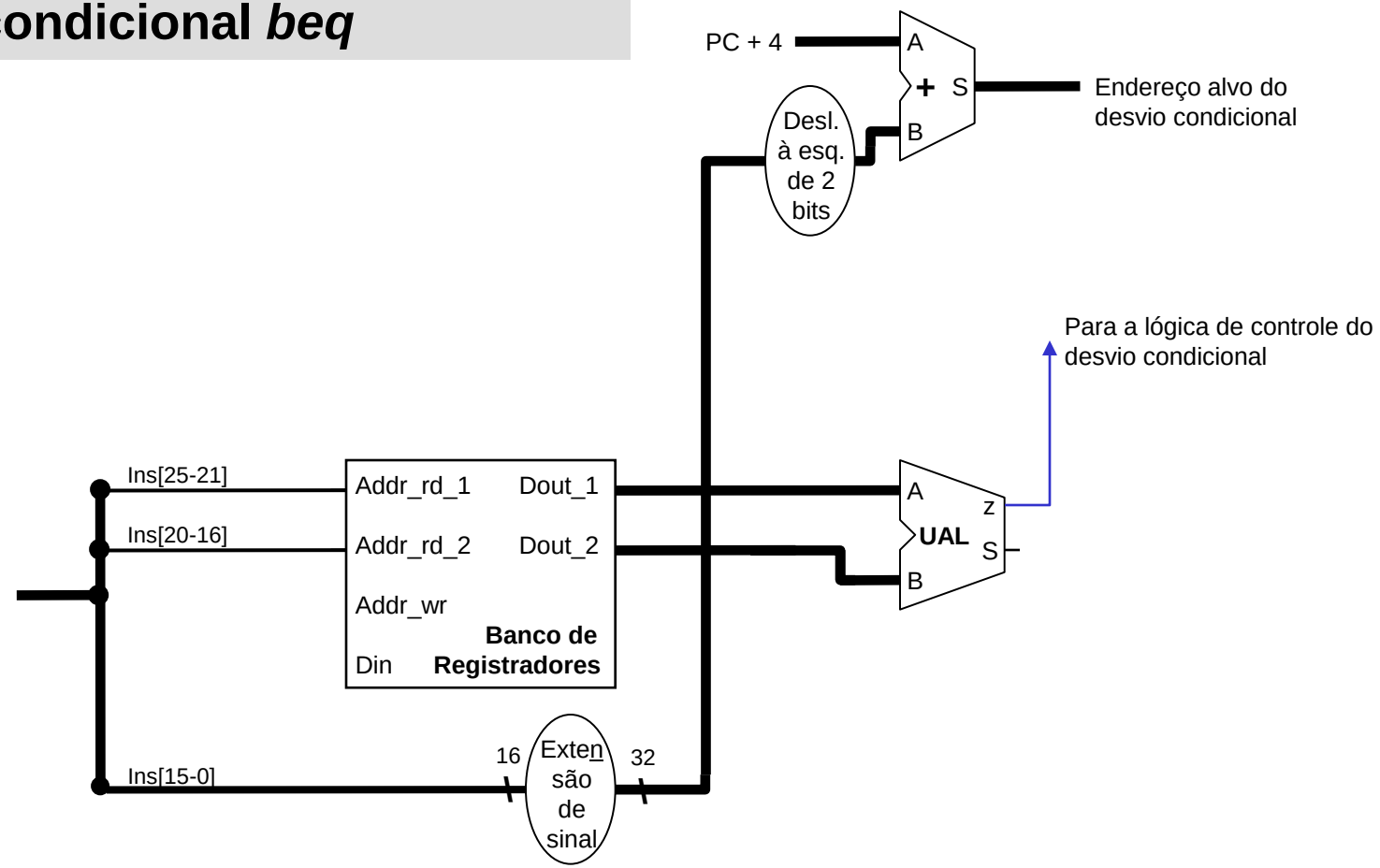
Caminho de dados para instruções do tipo R, *lw* e *sw*

São incluídos três multiplexadores para permitir a conexão de saídas de diferentes blocos a uma mesma entrada.

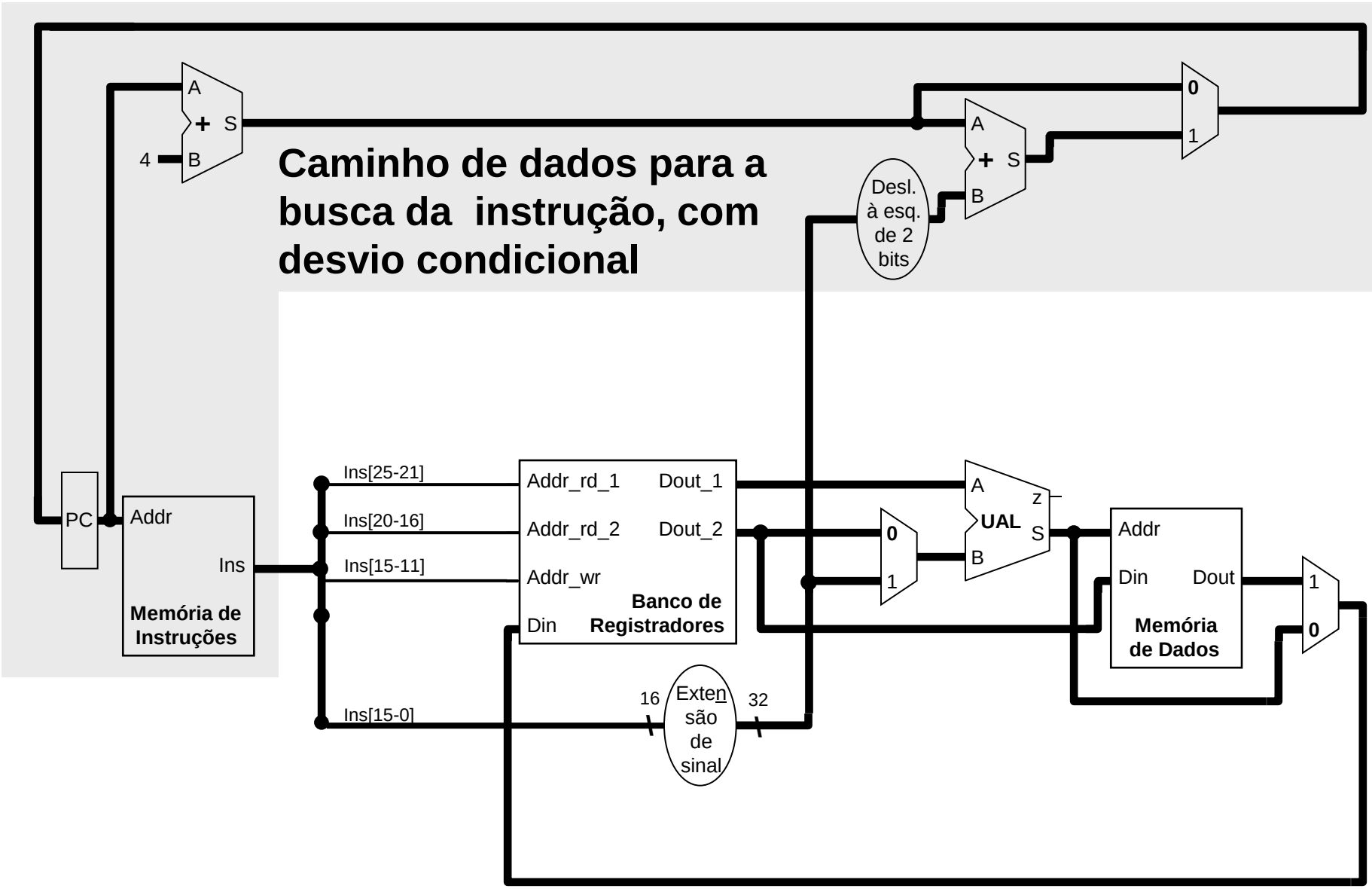


Caminho de dados

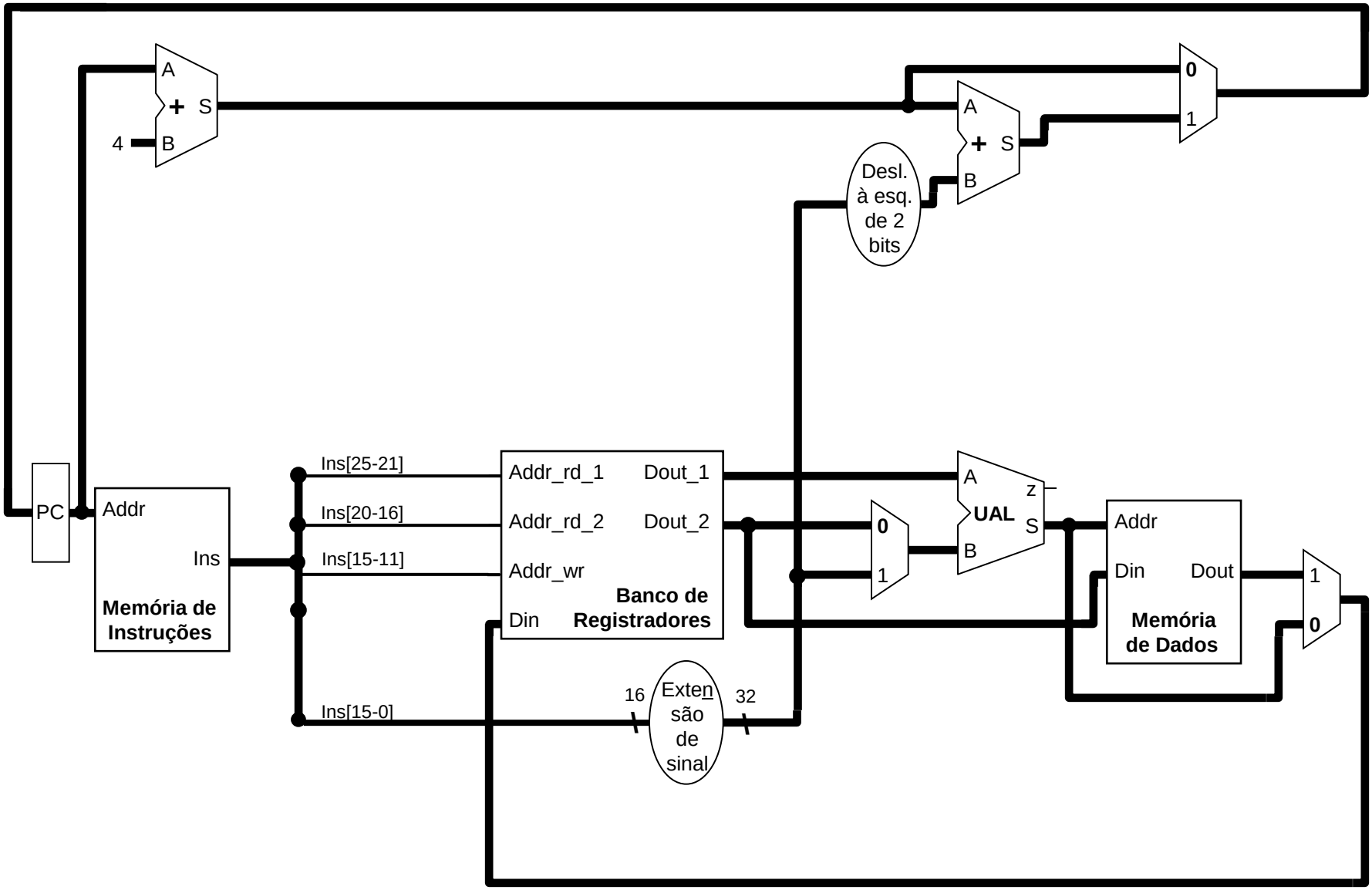
Caminho de dados para a instrução de desvio condicional *beq*



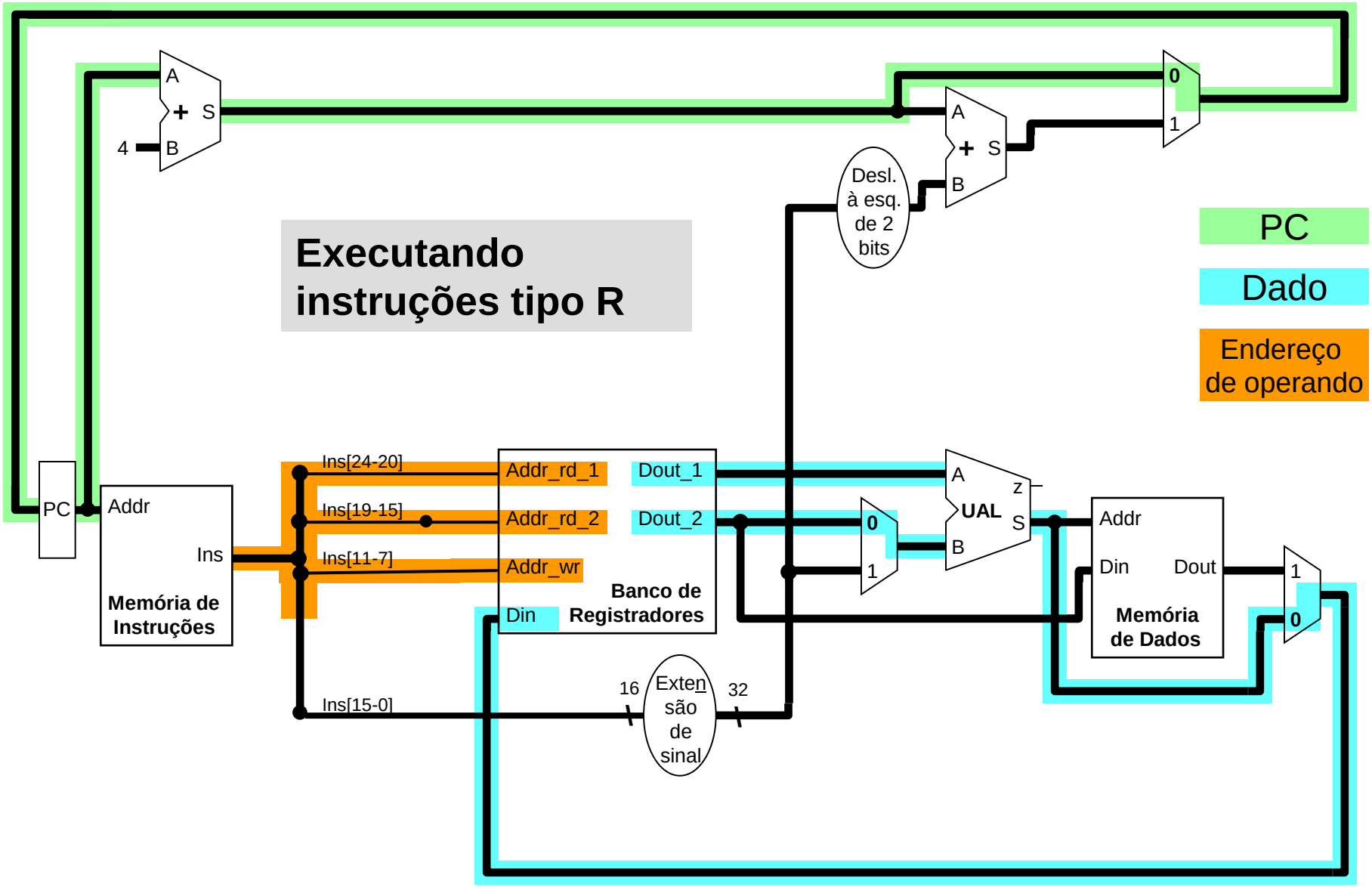
Caminho de dados



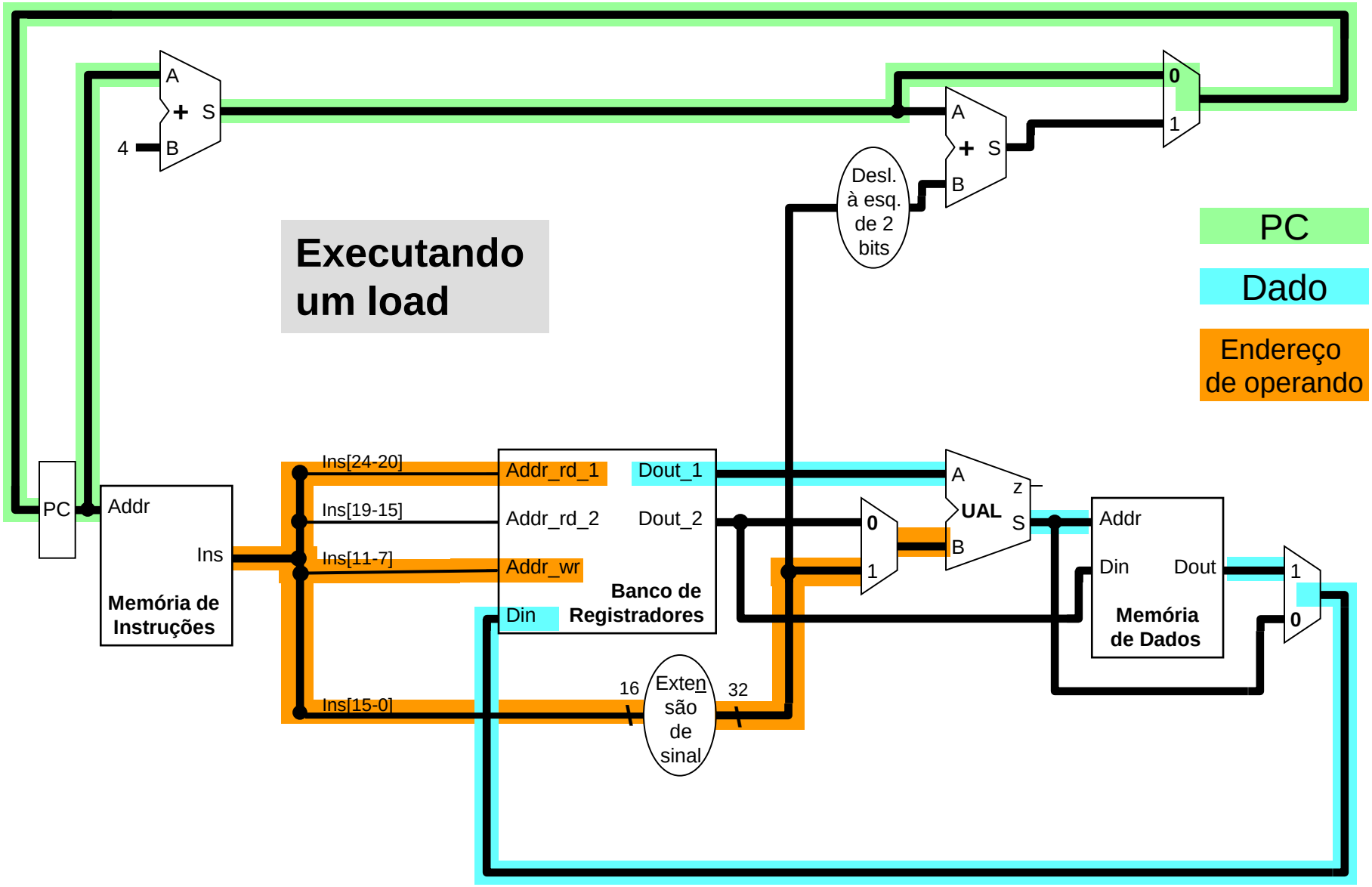
Caminho de dados



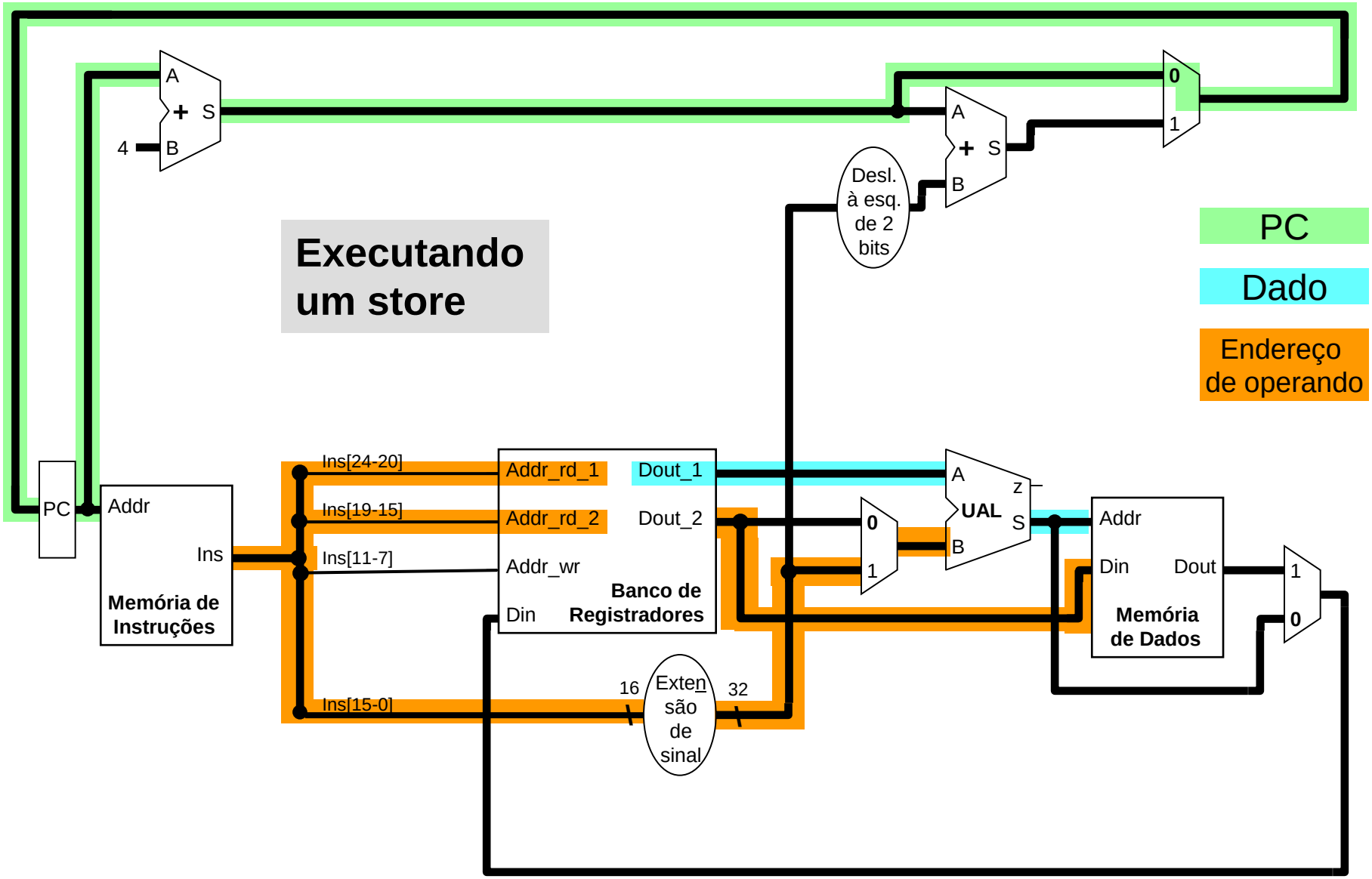
Caminho de dados



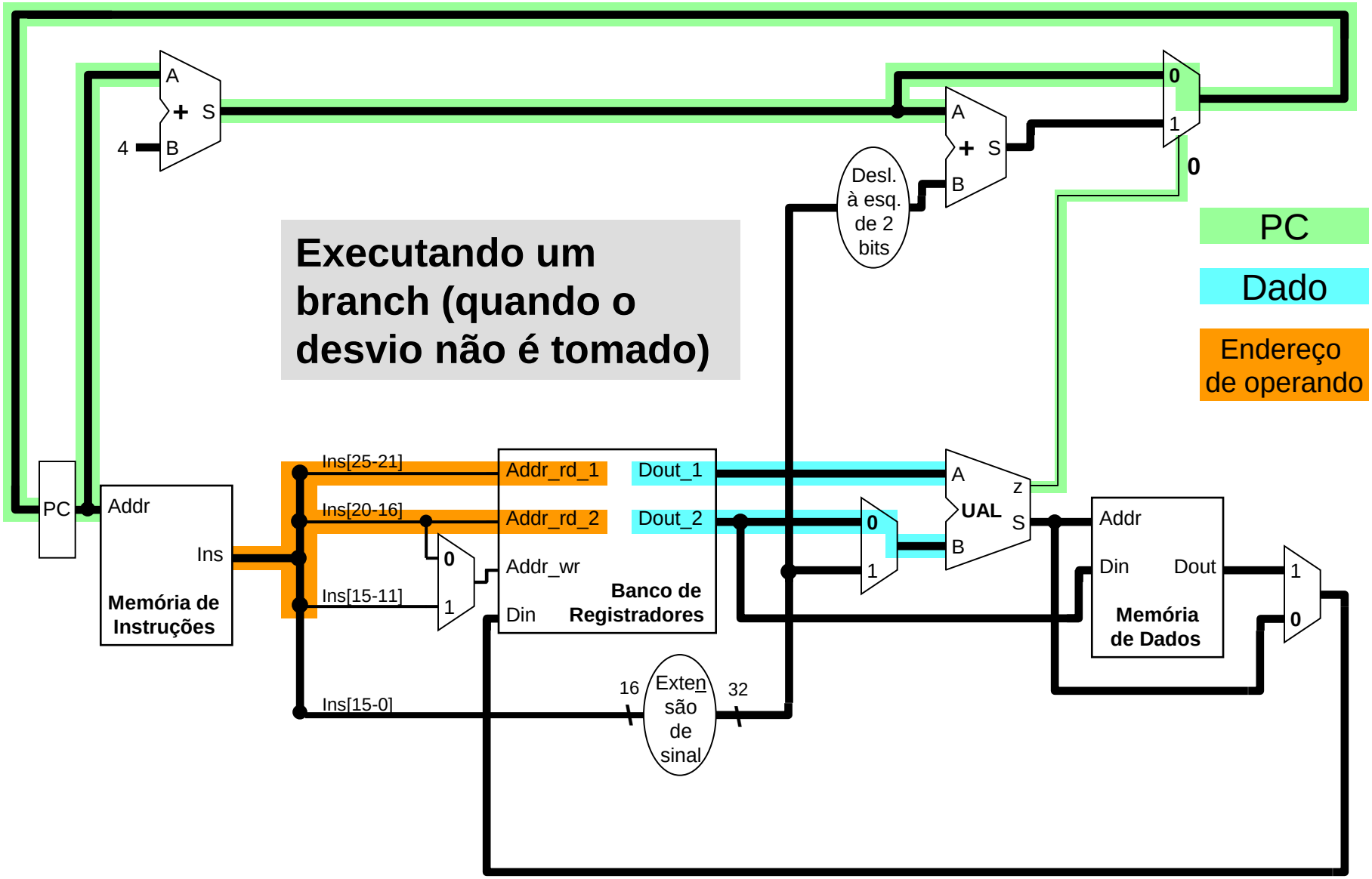
Caminho de dados



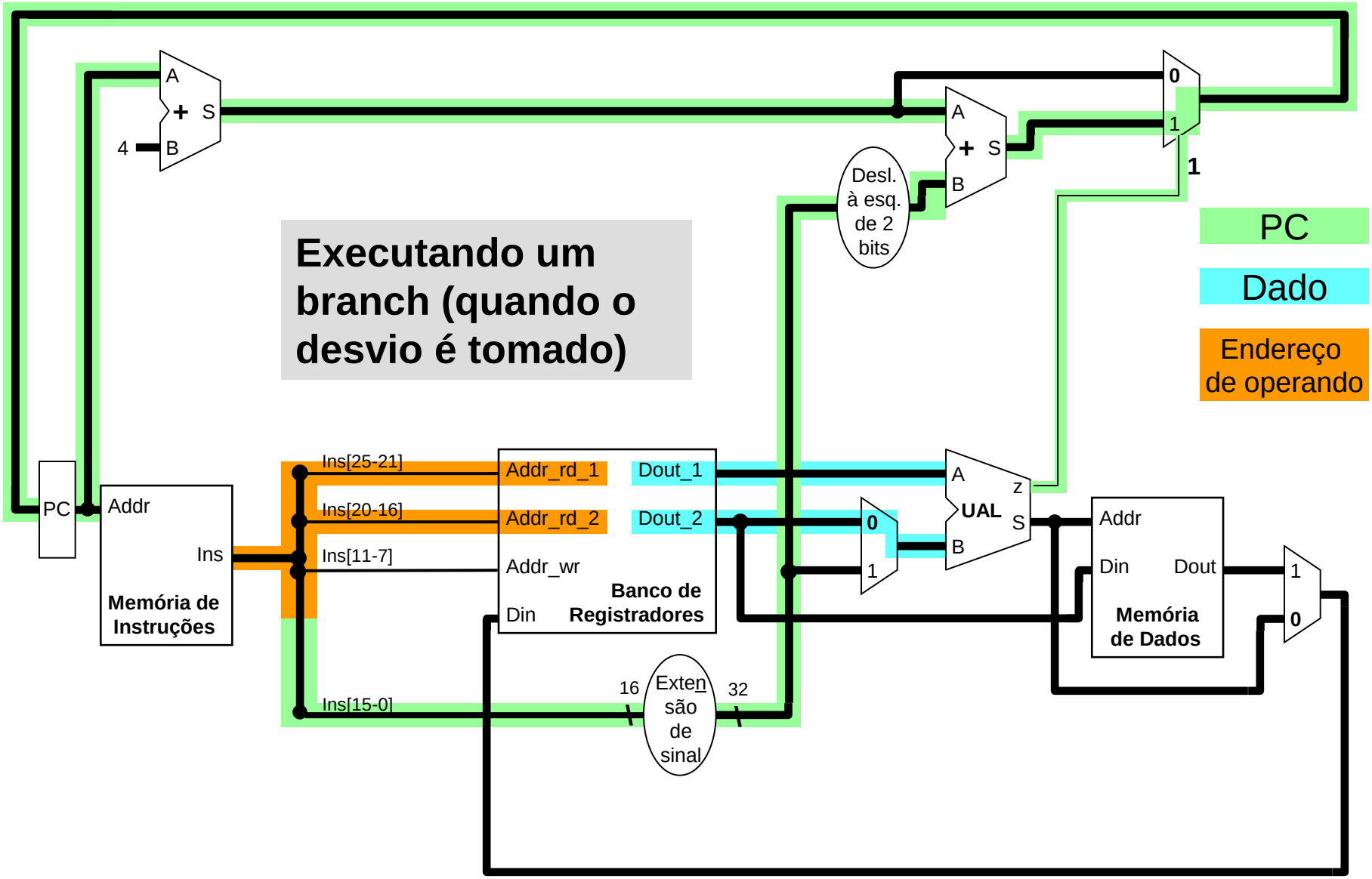
Caminho de dados



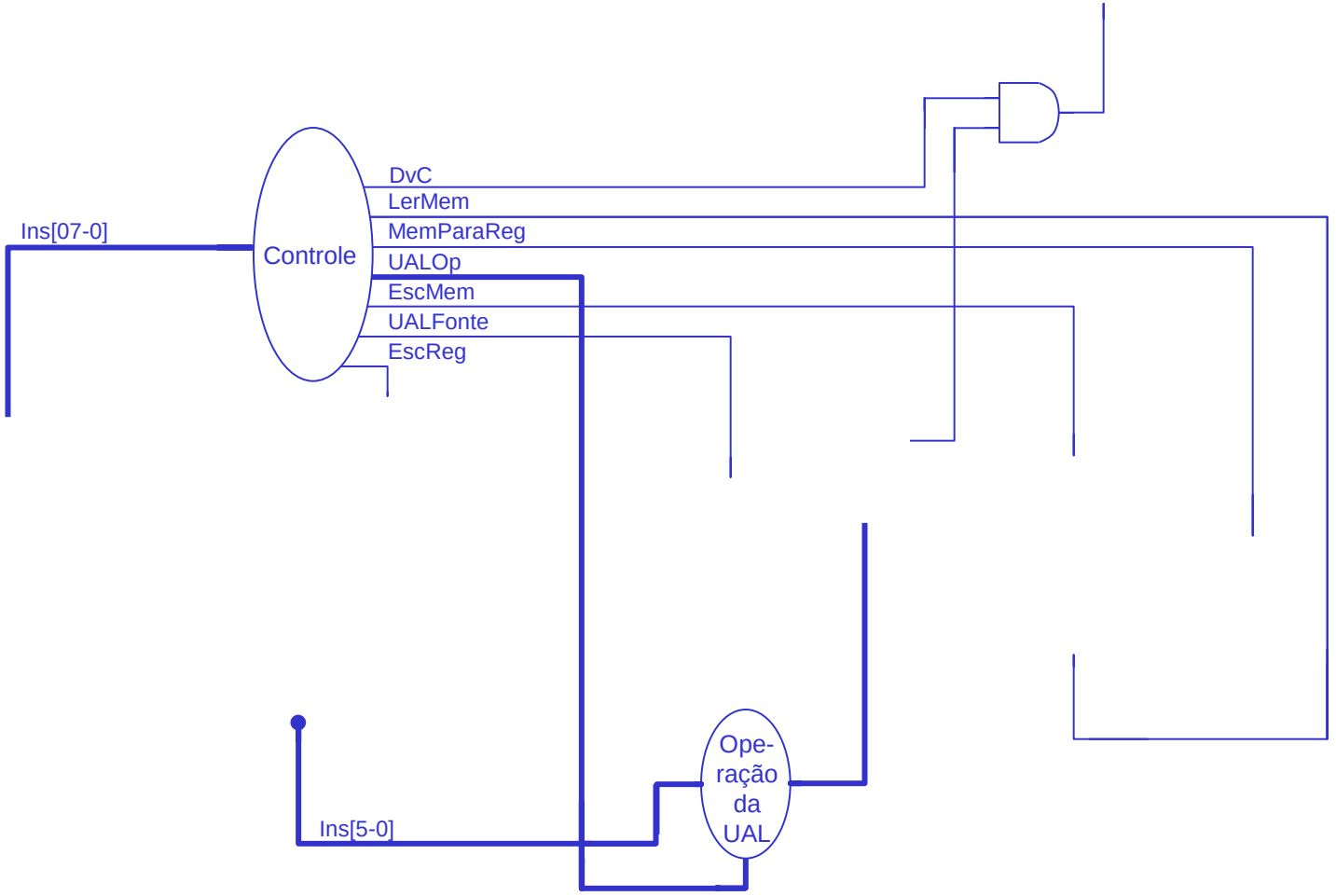
Caminho de dados



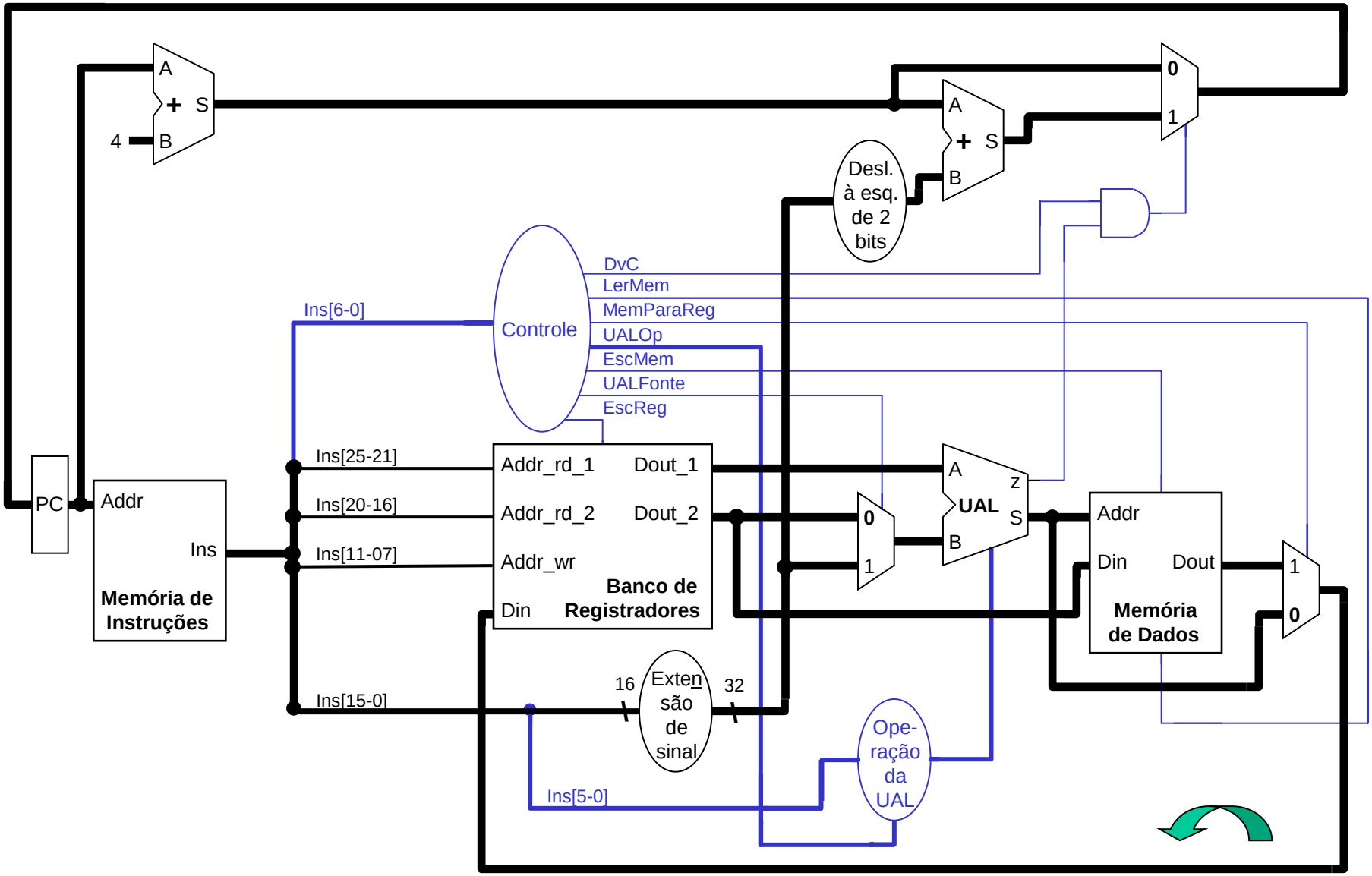
Caminho de dados



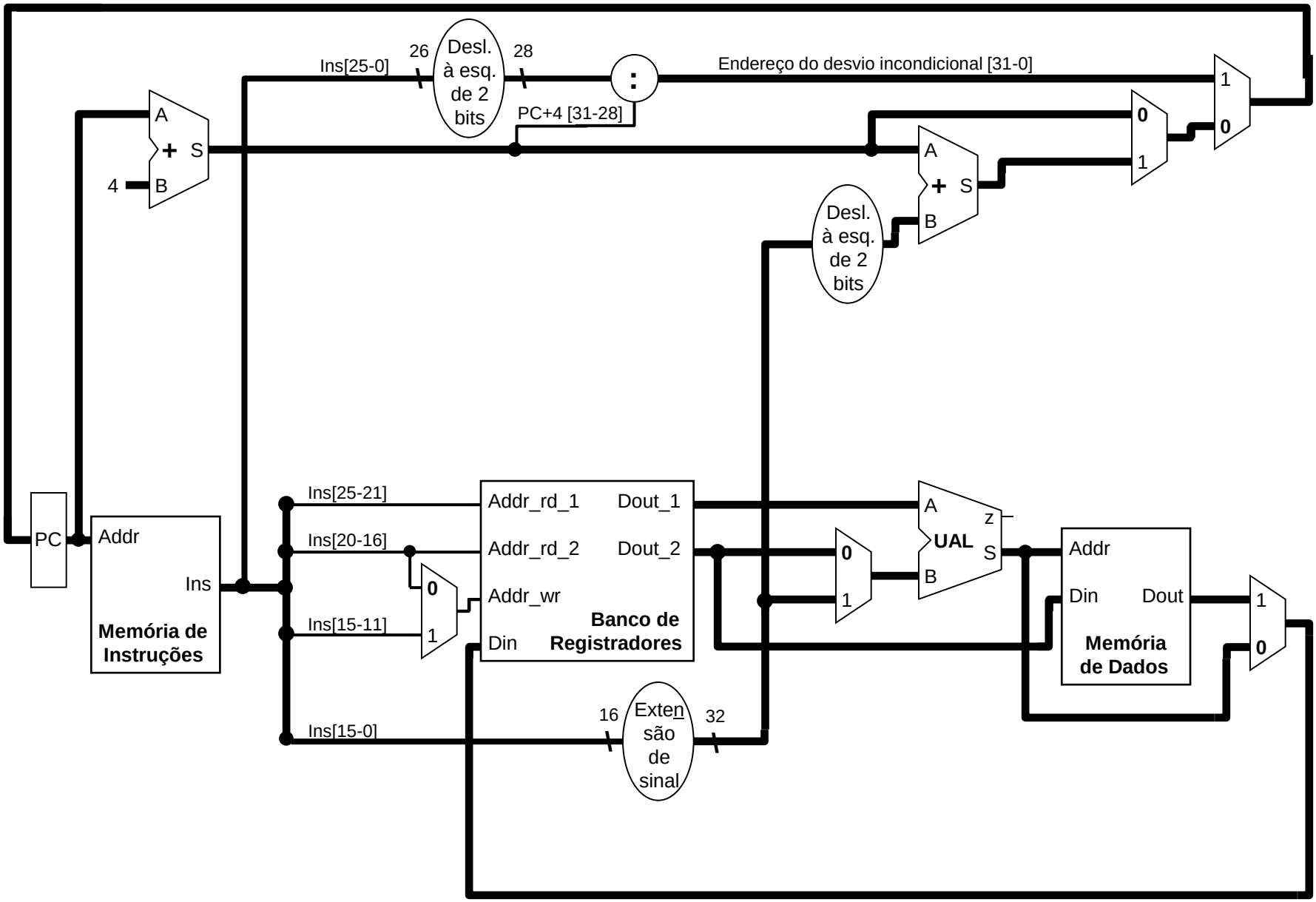
Sinais de controle



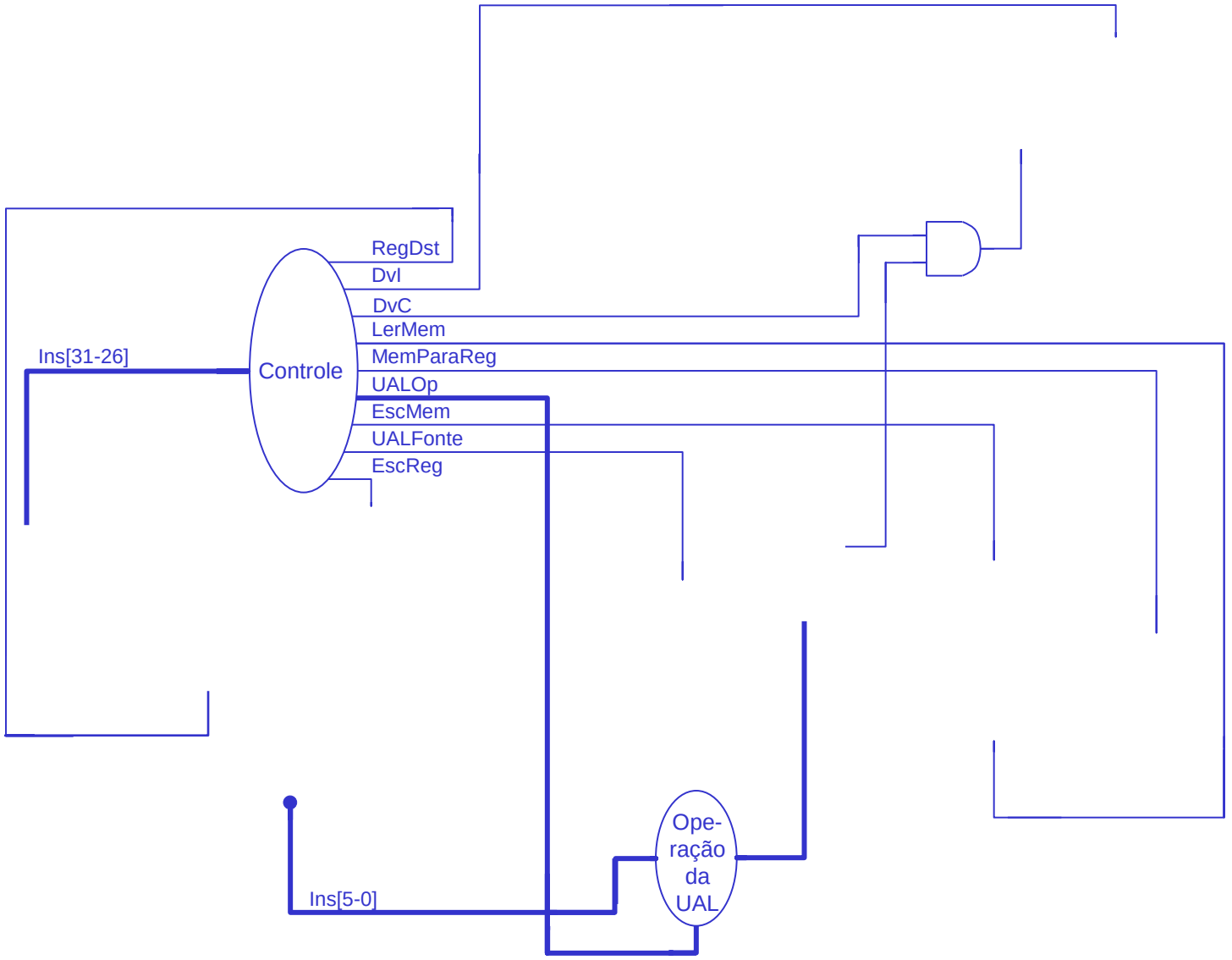
Sinais de controle



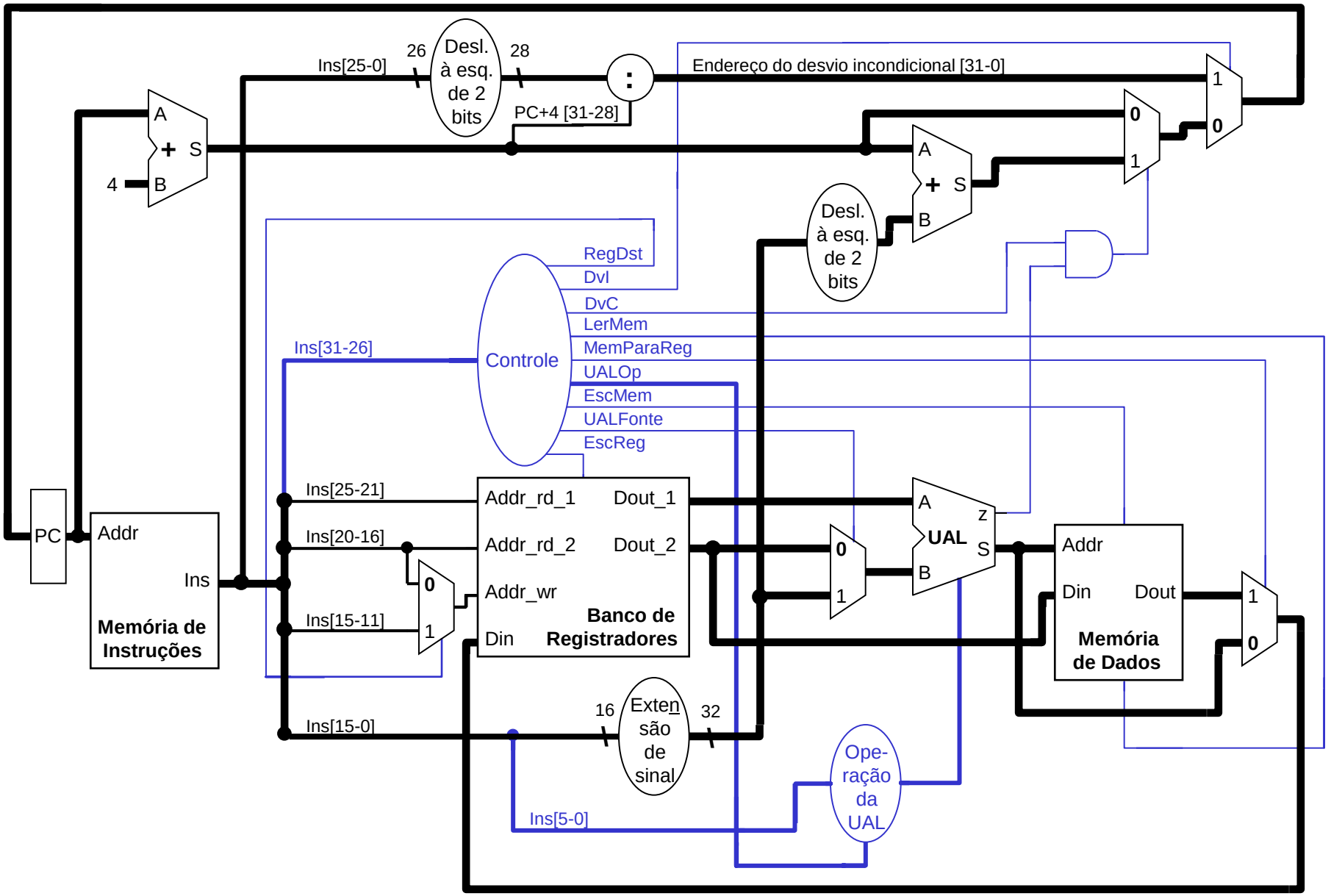
Caminho de dados com jump



Controle com jump

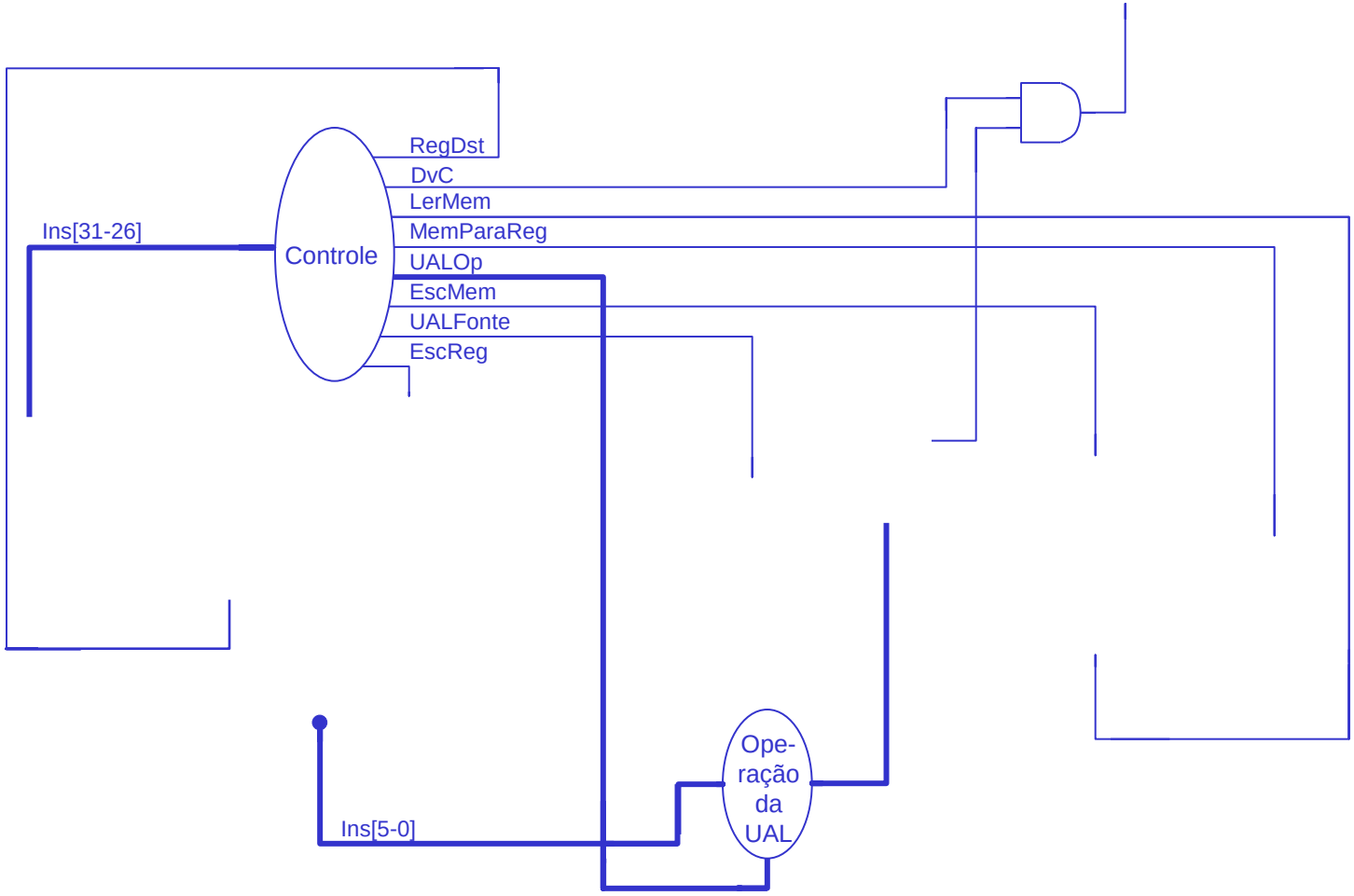


Controle com jump



Projeto do controle

- **Dois blocos de controle**
  - Controle
  - Operação da UAL



Projeto do controle de operação da UAL

- **O bloco de controle principal envia um comando de dois bits ao bloco de controle da UAL onde:**
  - UALOp = 00 => instrução lw ou sw
  - UALOp = 01 => instrução beq
  - UALOp = 10 => instrução no formato R
- **No caso de instrução no formato R, o controle da UAL tem que analisar o valor do campo Funct da instrução**

Projeto do controle de operação da UAL

- **Entradas**
  - Funct: campo da instrução
  - UALOp (ou OpALU): gerado pelo controle principal

- **Saídas**
  - Operation: vai para a UAL
  - Tabela verdade

OpALU		Campo Funct						Operation
OpALU1	OpALU2	F5	F4	F3	F2	F1	F0	
0	0	X	X	X	X	X	X	010
X	1	X	X	X	X	X	X	110
1	X	X	X	0	0	0	0	010
1	X	X	X	0	0	1	0	110
1	X	X	X	0	1	0	0	000
1	X	X	X	0	1	0	1	001
1	X	X	X	1	0	1	0	111

Projeto do controle de operação da UAL

Equações

OpALU		Campos de código Function					
OpALU1	OpALU0	F5	F4	F3	F2	F1	F0
X	1	X	X	X	X	X	X
1	X	X	X	X	X	1	X

a. A tabela verdade para Operation2 = 1 (essa tabela corresponde ao bit mais à esquerda do campo Operation na Figura C.2.1)

OpALU		Campos de código Function					
OpALU1	OpALU0	F5	F4	F3	F2	F1	F0
0	X	X	X	X	X	X	X
X	X	X	X	X	0	X	X

b. A tabela verdade para Operation1 = 1

OpALU		Campos de código Function					
OpALU1	OpALU0	F5	F4	F3	F2	F1	F0
1	X	X	X	X	X	X	1
1	X	X	X	1	X	X	X

c. A tabela verdade para Operation0 = 1

Projeto do controle de operação da UAL

Equações

OpALU		Campos de código Function					
OpALU1	OpALU0	F5	F4	F3	F2	F1	F0
X	1	X	X	X	X	X	X
1	X	X	X	X	X	1	X

a. A tabela verdade para Operation2 = 1 (essa tabela corresponde ao bit mais à esquerda do campo Operation na Figura C.2.1)

OpALU		Campos de código Function					
OpALU1	OpALU0	F5	F4	F3	F2	F1	F0
0	X	X	X	X	X	X	X
X	X	X	X	X	0	X	X

b. A tabela verdade para Operation1 = 1

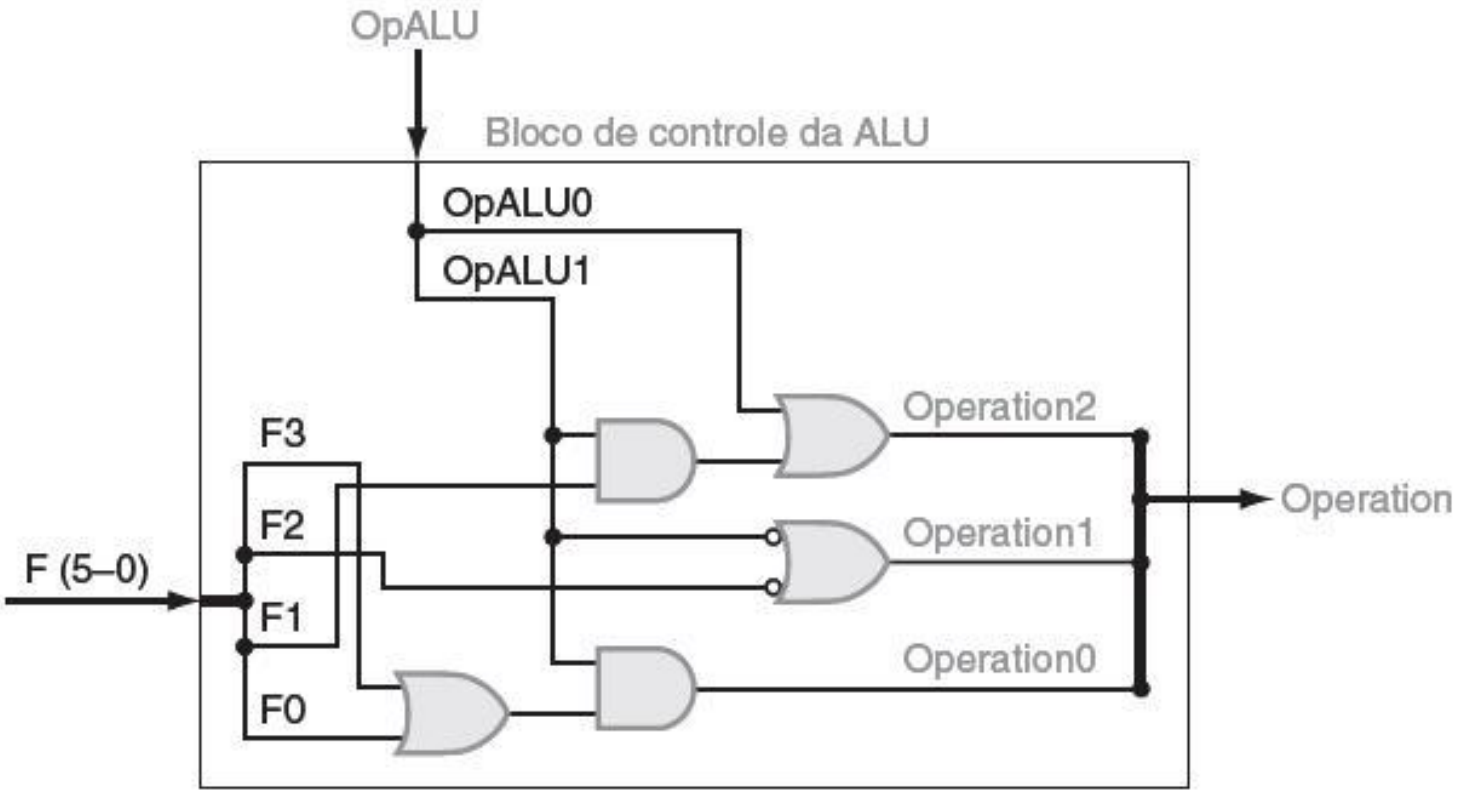
OpALU		Campos de código Function					
OpALU1	OpALU0	F5	F4	F3	F2	F1	F0
1	X	X	X	X	X	X	1
1	X	X	X	1	X	X	X

c. A tabela verdade para Operation0 = 1

- Operation2 = OpALU0 + (OpALU1 . F1)
- Operation1 = OpALU1'+ F2'
- Operation0 = OpALU1 . (F3 + F0)

Projeto do controle de operação da UAL

Equações



- Operation2 = OpALU0 + (OpALU1 . F1)
- Operation1 = OpALU1'+ F2'
- Operation0 = OpALU1 . (F3 + F0)

Projeto do controle principal

Entradas e saídas

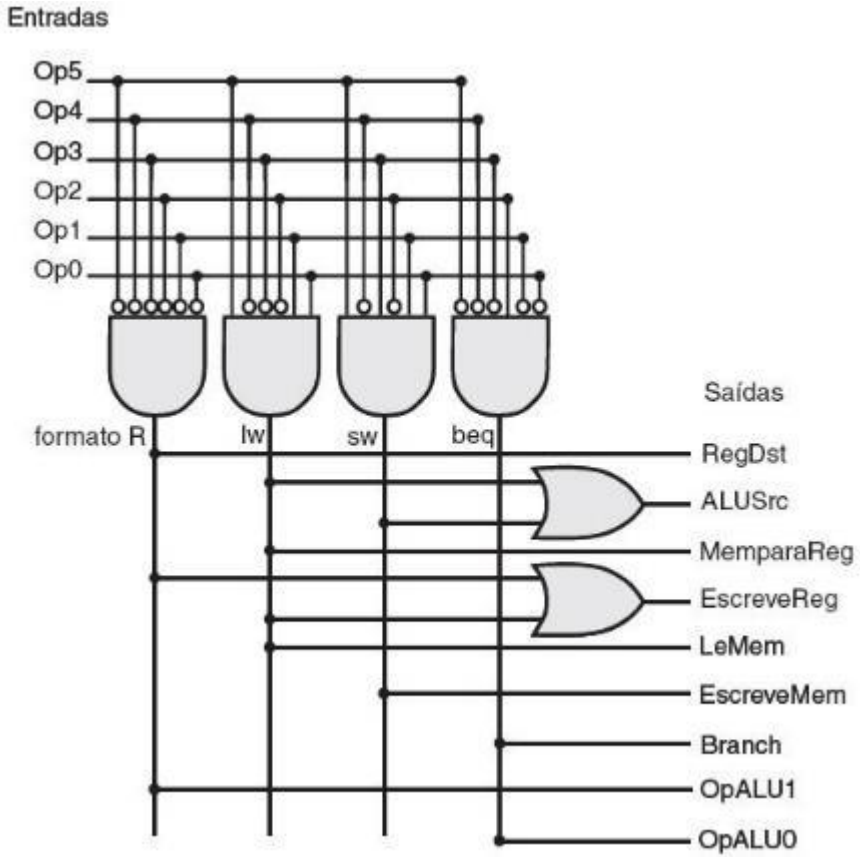
Controle	Nome do sinal	formato R	lw	sw	beq
Entradas	Op5	0	1	1	0
	Op4	0	0	0	0
	Op3	0	0	1	0
	Op2	0	0	0	1
	Op1	0	1	1	0
	Op0	0	1	1	0
Saídas	RegDst	1	0	X	X
	ALUSrc	0	1	1	0
	MemparaReg	0	1	X	X
	EscreveReg	1	1	0	0
	LeMem	0	1	0	0
	EscreveMem	0	0	1	0
	Branch	0	0	0	1
	OpALU1	1	0	0	0
	OpALU0	0	0	0	1

Projeto do controle principal

- **Controle principal** **(2a Ed.)**
  - RegDst = formato_R
  - ALUSrc = lw + sw (UALFonte)
  - MemparaReg = lw
  - EscreveReg = formato_R + lw (EscReg)
  - LeMem = lw (LerMem)
  - EscreveMem = sw (EscMem)
  - Branch = beq (DvC)
  - OpALU1 = formato_R
  - OpALU0 = beq

Projeto do controle principal

- **Controle principal**
  - RegDst = formato_R
  - ALUSrc = lw + sw
  - MemparaReg = lw
  - EscreveReg = formato_R + lw
  - LeMem = lw
  - EscreveMem = sw
  - Branch = beq
  - OpALU1 = formato_R
  - OpALU0 = beq



Projeto do controle principal

- Controle principal