

“Soluções temporárias geralmente se tornam problemas permanentes”
(Craig S. Bruce).

CPU Monociclo - Construção

Paulo Ricardo Lisboa de Almeida



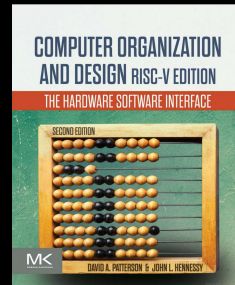
Implementação Básica

Construir um processador RISC-V básico, que implementa o seguinte:

- Instruções de referência a memória: `sw` e `lw`
- Instruções lógicas e aritméticas: `add`, `sub`, `and`, e `or`.
- Instrução de desvio: `beq`

Essa implementação é a descrita no livro base da disciplina.

Patterson, Hennessy.
Computer Organization and
Design RISC-V Edition: The
Hardware Software
Interface. 2020.



O Caminho de Dados

Analisando os principais componentes necessários para executar um programa são:

Uma **memória**, para armazenar as **instruções** do programa.

Entrada: O endereço da instrução desejada.

Saída: A instrução no endereço apontado.

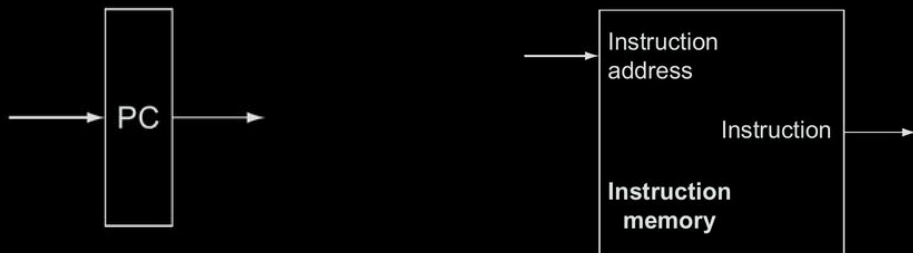
Um registrador que vai armazenar o endereço da próxima instrução a ser executada.

Registrador **PC** no RISC-V.

Entrada: O próximo endereço a ser armazenado no registrador.

Saída: O endereço corrente.

O Caminho de Dados



pc = 0x00400000	→ 0x00400000	instrução 1
	0x00400004	instrução 2
	0x00400008	instrução 3
	0x0040000C	instrução 4
	0x00400010	instrução 5
	0x00400014	instrução 6
	...	...

Program Counter

Caso não hajam desvios, após a execução da instrução atual, executar a próxima instrução.

O que fazer com o PC? Como?

0x00400000	instrução 1
0x00400004	instrução 2
0x00400008	instrução 3
0x0040000C	instrução 4
0x00400010	instrução 5
0x00400014	instrução 6
...	...

Program Counter

Caso não hajam desvios, após a execução da instrução atual, executar a próxima instrução.

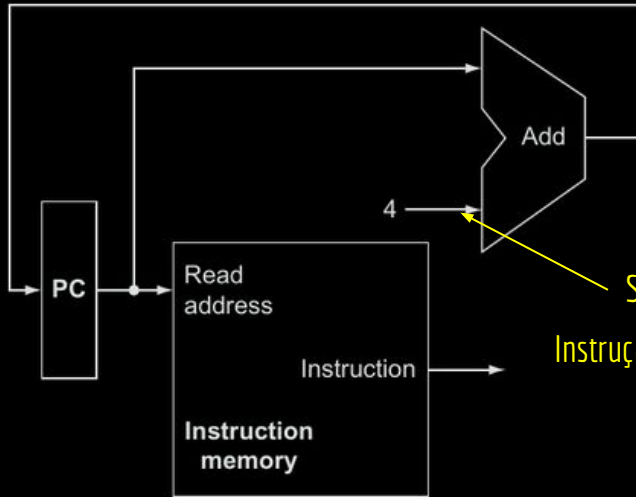
O que fazer com o PC? Como?

Memória endereçada em bytes, logo “saltamos” 4 bytes = 32 bits.

Somador.

0x00400000	instrução 1
0x00400004	instrução 2
0x00400008	instrução 3
0x0040000C	instrução 4
0x00400010	instrução 5
0x00400014	instrução 6
...	...

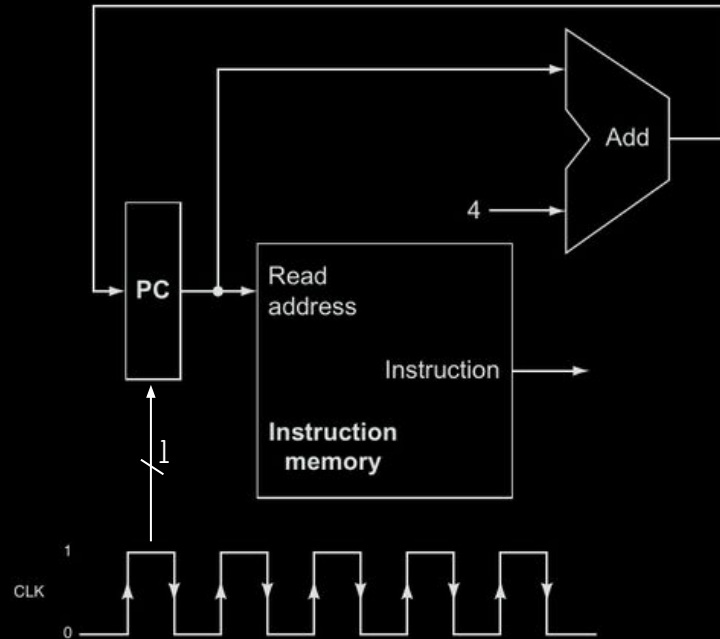
Ligando



Some 4_{10} no endereço atual de PC.

Instrução (32 bits) armazenada no endereço apontado por PC.

Sincronização



Sinais de **clock** são adicionados nos elementos de estado (sequenciais).

Ex.: O registrador PC só vai ler a entrada na transição de clock.

Enquanto não há transição, o valor antigo de PC é enviado para a saída, e lido pelo adder.

O sinal de clock vai ser **omitido para simplificar o raciocínio.**

Assuma que existe sincronismo, para que o sinal anterior não seja “atropelado” pelo próximo.

Ligando

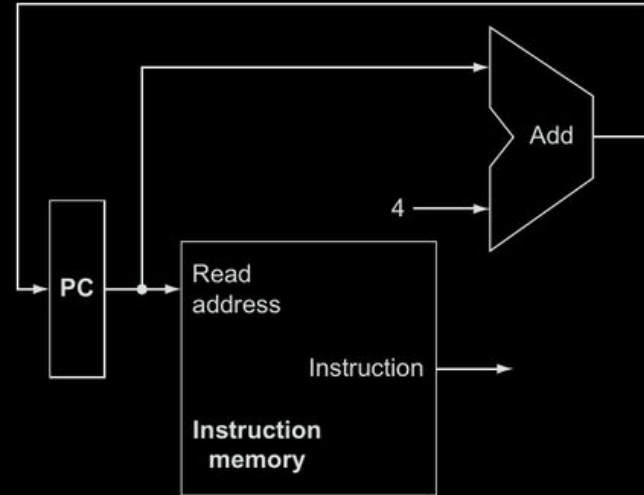
O “loop” principal está pronto.

A instrução é enviada para execução, e o PC é incrementado em 4 para apontar para a próxima.

O que é feito com a instrução agora depende do seu formato.

Começando com instruções básicas do tipo-R.

Como são?



Tipo-R

Exemplo: `add x9, x20, x21`

Ler dois registradores, executar uma operação aritmética em uma ALU (soma, subtração, deslocamento,...), e armazenar o resultado em um terceiro registrador.



Banco de Registradores

Contém os 32 registradores do RISC-V.

Entradas: Endereço do registrador de leitura 1;

Endereço do registrador de leitura 2;

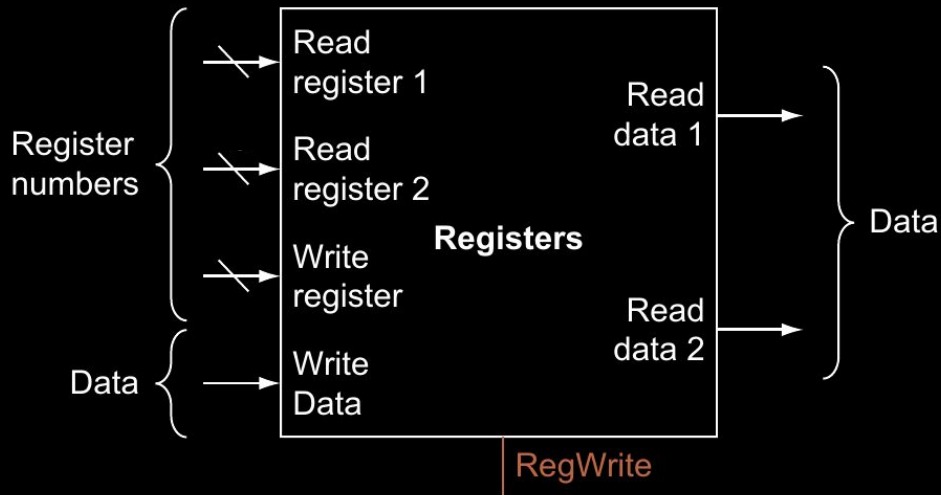
Endereço do registrador de escrita;

Dados a serem escritos no registrador de escrita.

Saídas: Dados do registrador de leitura 1;

Dados do registrador de leitura 2.

Pergunta: Qual o tamanho de cada um dos barramentos de entrada e saída no banco de registradores?



Banco de Registradores

Contém os 32 registradores do RISC-V.

Entradas: Endereço do registrador de leitura 1;

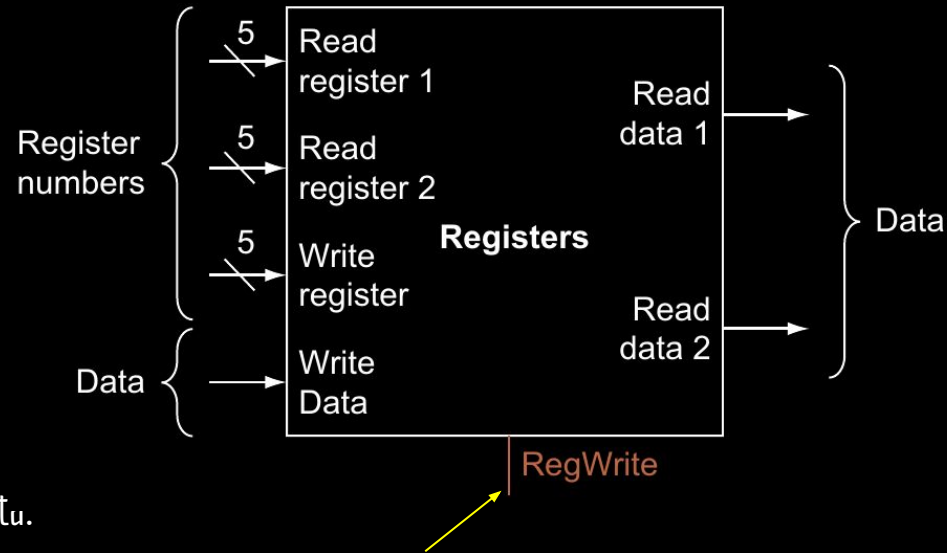
Endereço do registrador de leitura 2;

Endereço do registrador de escrita;

Dados a serem escritos no registrador de escrita.

Saídas: Dados do registrador de leitura 1;

Dados do registrador de leitura 2.



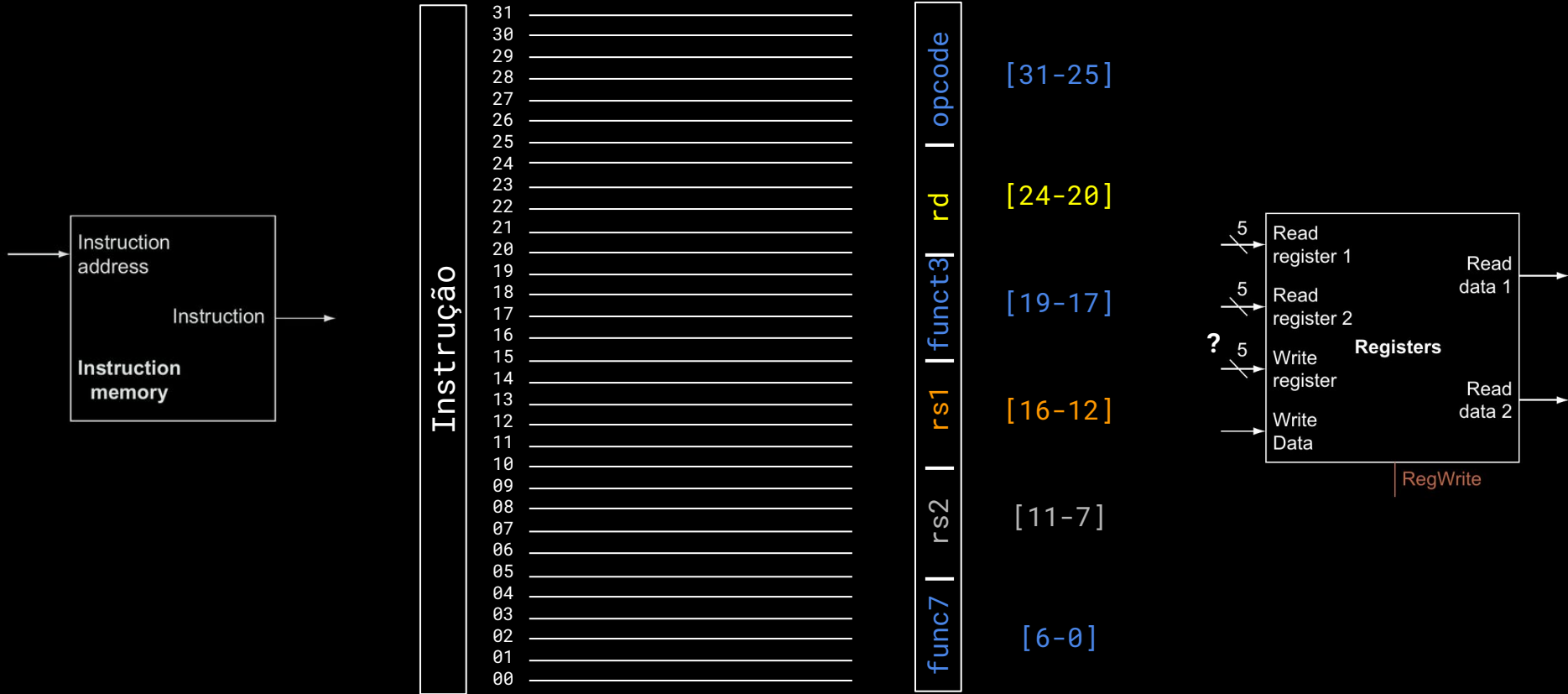
Veremos adiante o uso desse sinal de 1 bit.

Fonte dos bits

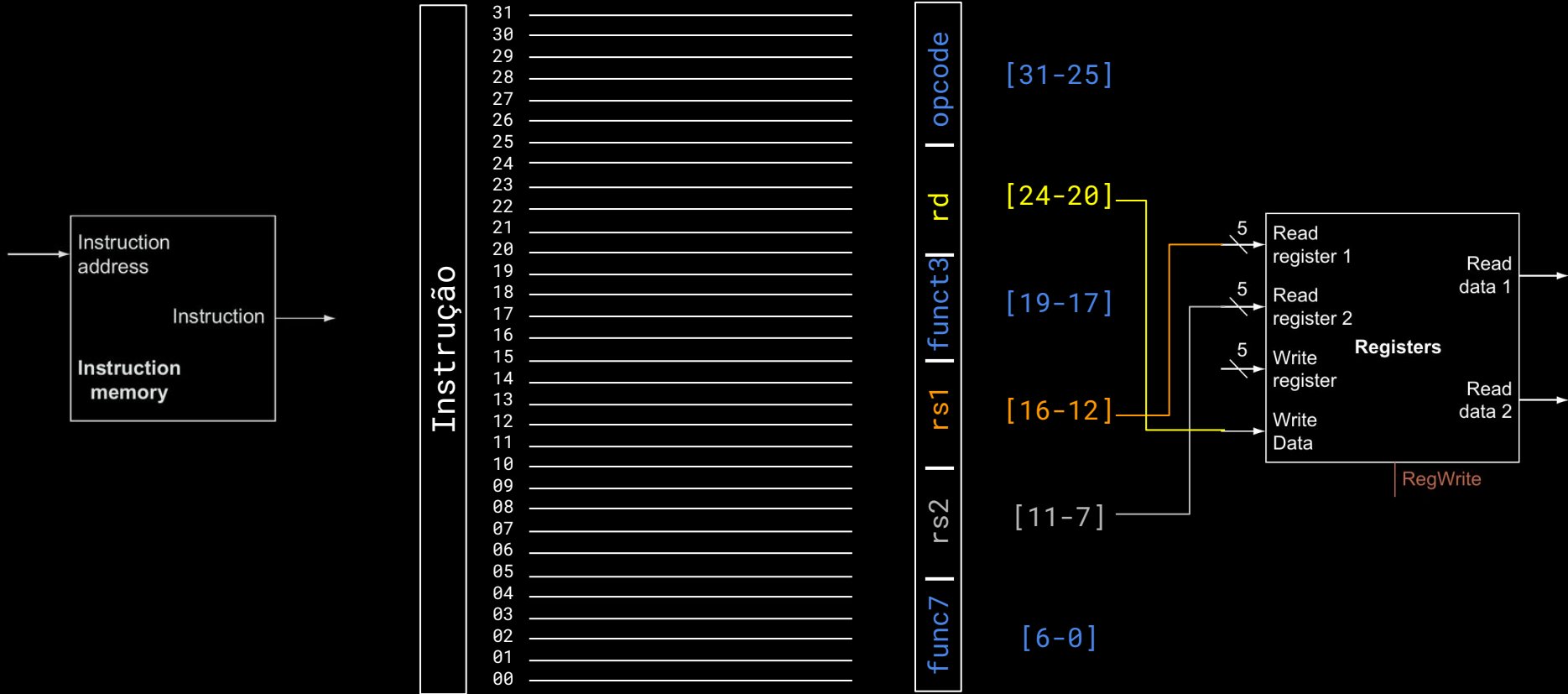
Como definido, uma linha sem marcação tem largura de 32 bits.

Então na verdade são 32 linhas (barramento), endereçadas de 0 a 31.

Fonte dos bits



Fonte dos bits



ALU

É necessário executar a operação especificada pela instrução.

Soma, subtração, deslocamento, ...

Arithmetic Logic Unit - **ALU**.

Entradas: Dois valores de 32 bits.

Entrada *ALU Operation* de 4 bits, que define qual a operação deve ser realizada com os valores.

Saídas: Uma saída de 32 bits com o resultado da operação.

Uma saída de 1 bit indicando se o resultado da operação foi zero.

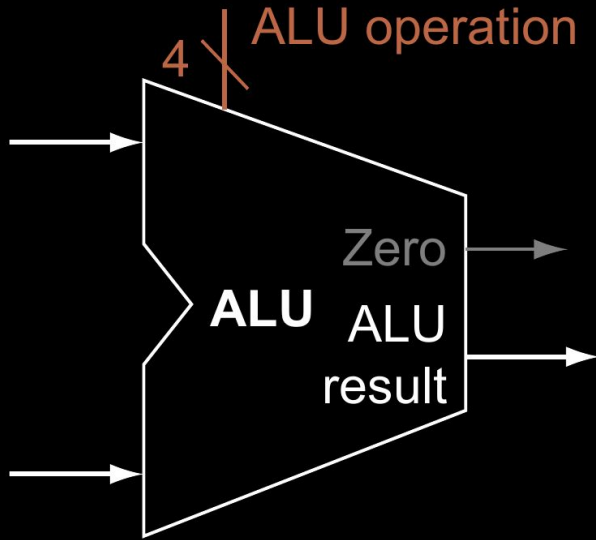
Simplificará a construção de outros trechos do circuito.

ALU

Dentro da ALU existem circuitos especialistas.

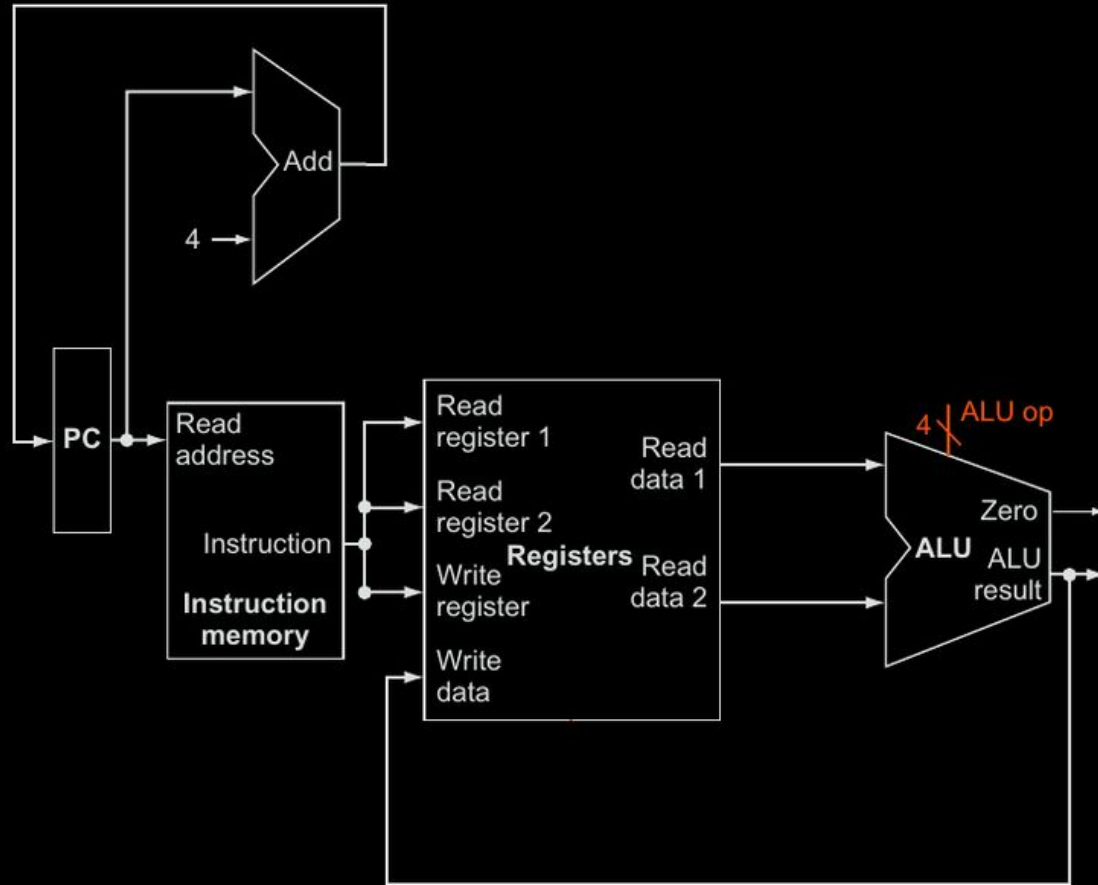
- Somadores.
- Operadores Lógicos.
- ...

Qual desses circuitos será ativado depende de **ALU Operation**.



Sinal em ALU Operation	Função
0000	AND
0001	OR
0010	somar
0110	subtrair

Ligando



Loads e Stores

Vamos adicionar instruções para *loads* (Tipo-I) e *stores* (Tipo-S).

Qual a diferença dessas instruções para o tipo-R?

Loads e Stores

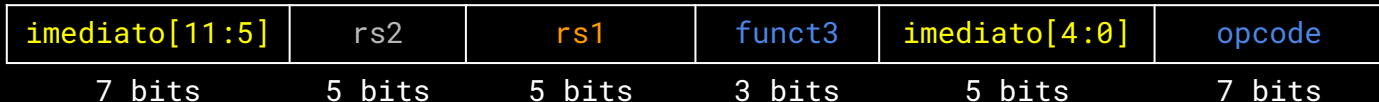
`lw x8, valor_offset(x22)`

Carrega para x8 o conteúdo apontado pelo endereço de memória $[x22 + \text{valor_offset}]$.



`sw x8, valor_offset(x22)`

Armazena no endereço apontado por $[x22 + \text{valor_offset}]$ o conteúdo de x8.



O Offset é representado em complemento de 2.

Pode ser positivo ou negativo.

Loads e Stores

Necessária uma **memória de dados**.

Entradas: Endereço de memória.

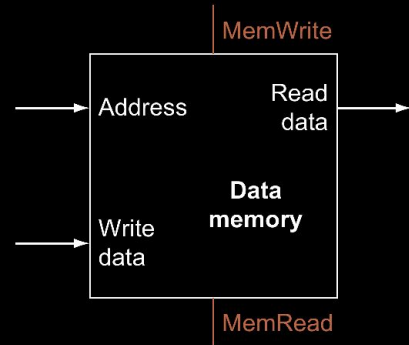
Dados a ser escrito.

Sinais de Controle MemWrite e MemRead.

Indica se a memória deve escrever na posição, ou;

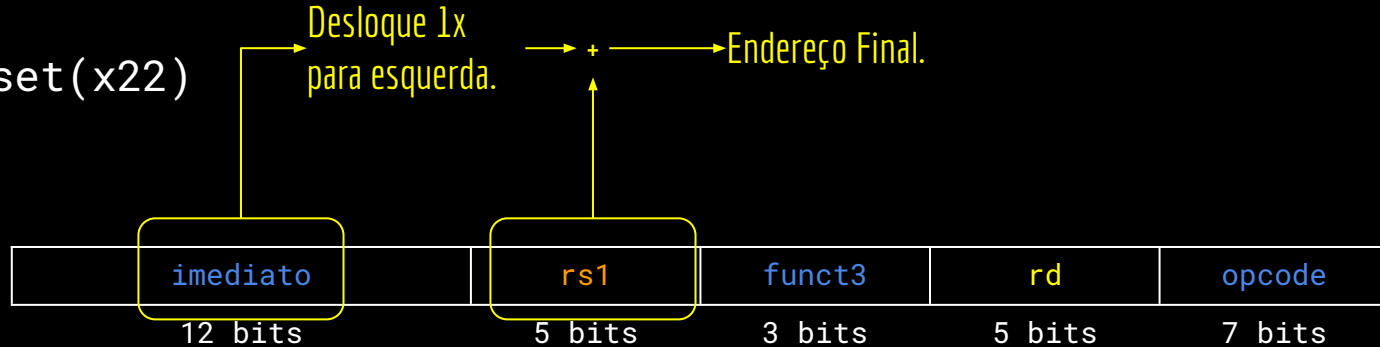
Se deve ler o dado especificado na posição e direcioná-lo para a saída.

Saída: Dado lido pela memória.

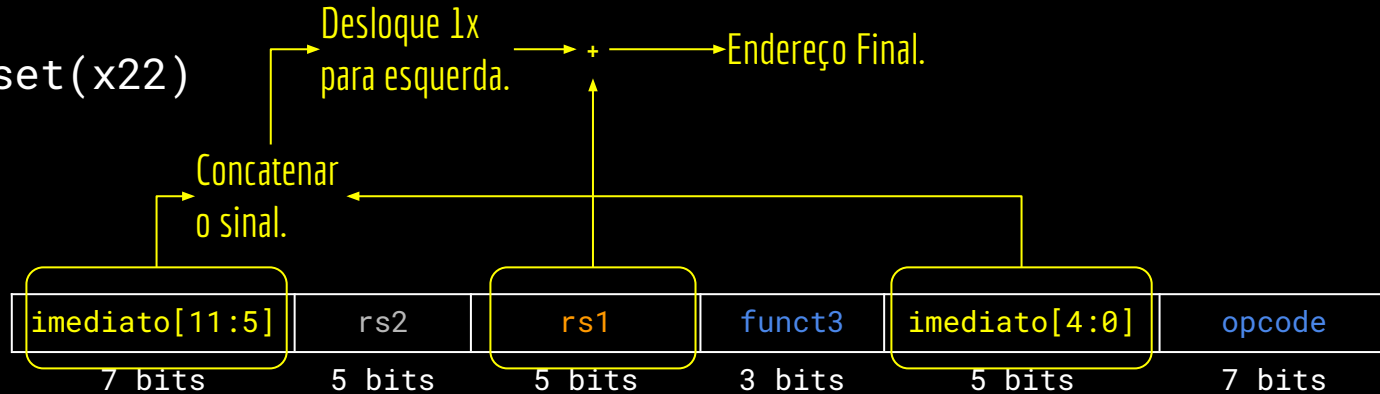


Imediato

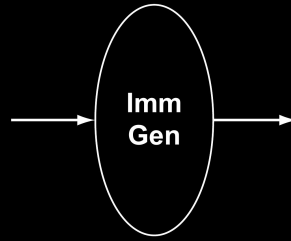
lw x8, valor_offset(x22)



sw x8, valor_offset(x22)

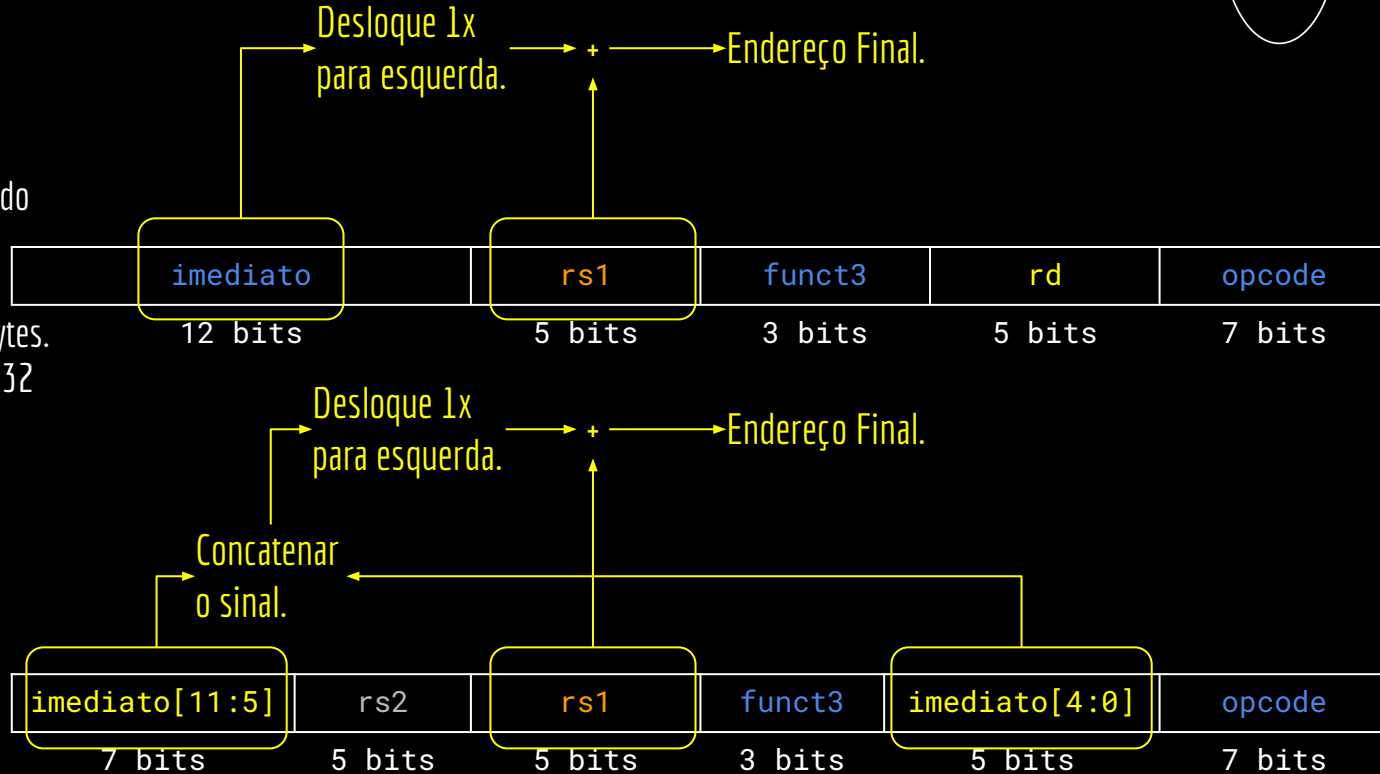


Imediato

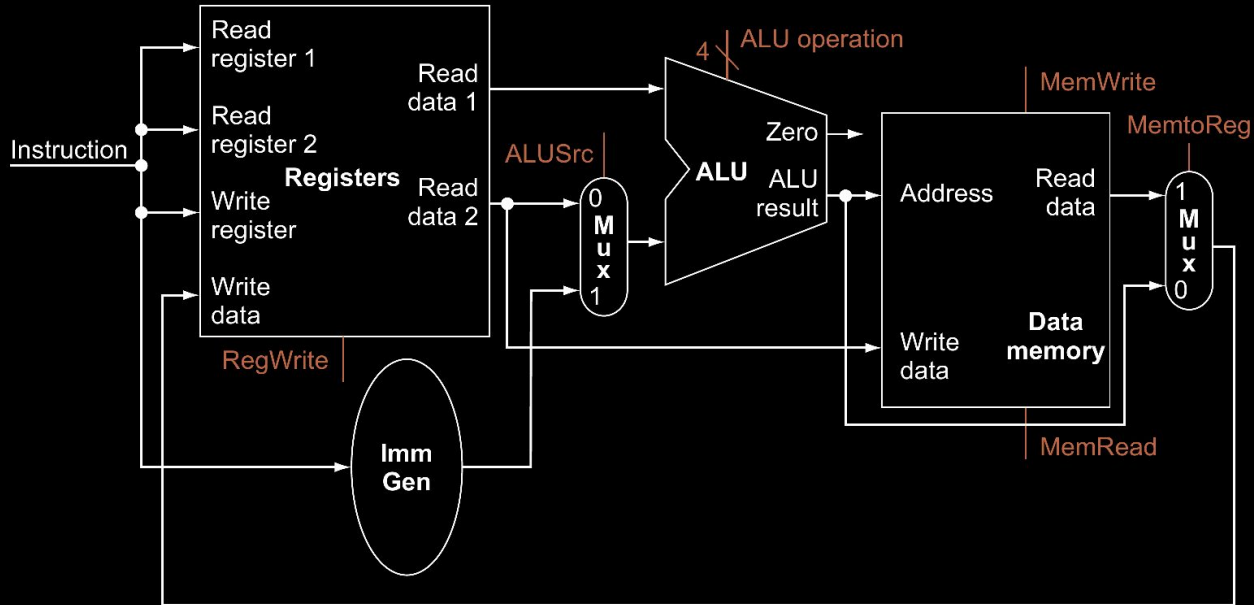


Unidade de Geração de Imediato.

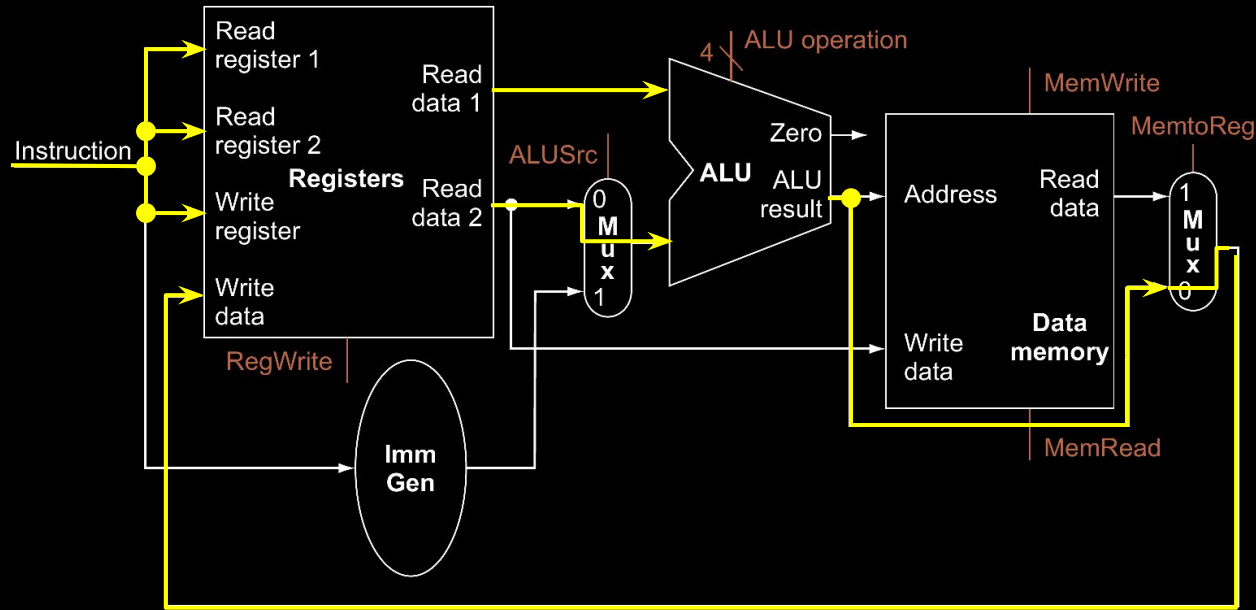
- Toma a instrução completa.
- Determina de onde os dados do imediato precisam vir.
- Desloca os dados 1x para transformar *half words* em bytes.
- Estende o imediato final para 32 bits.



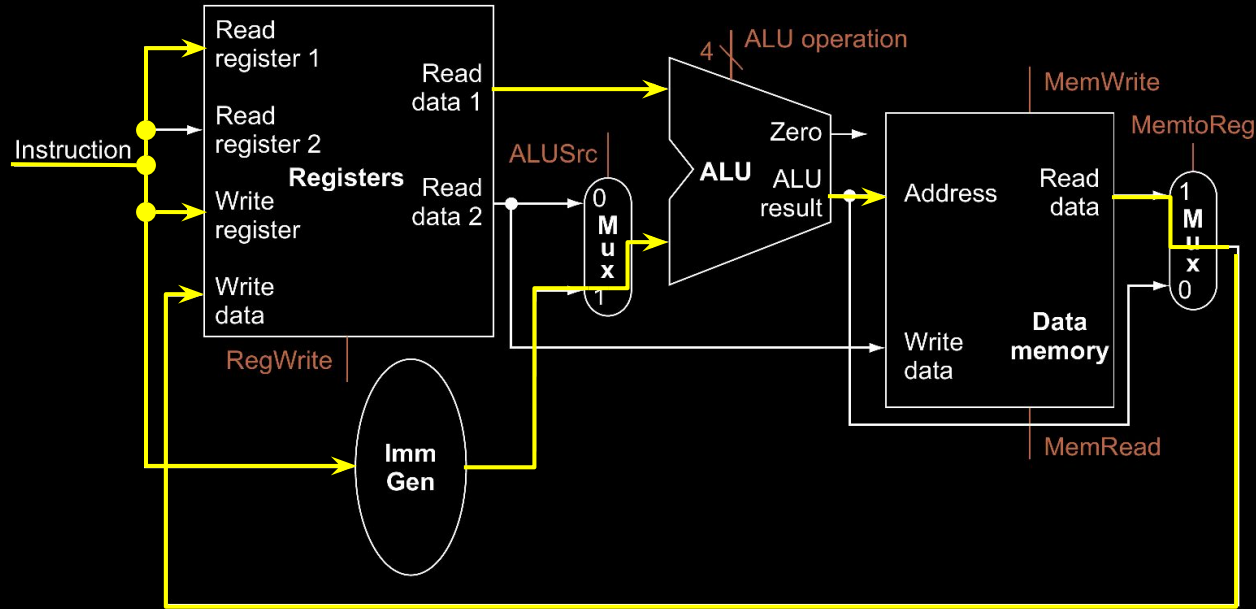
Ligando



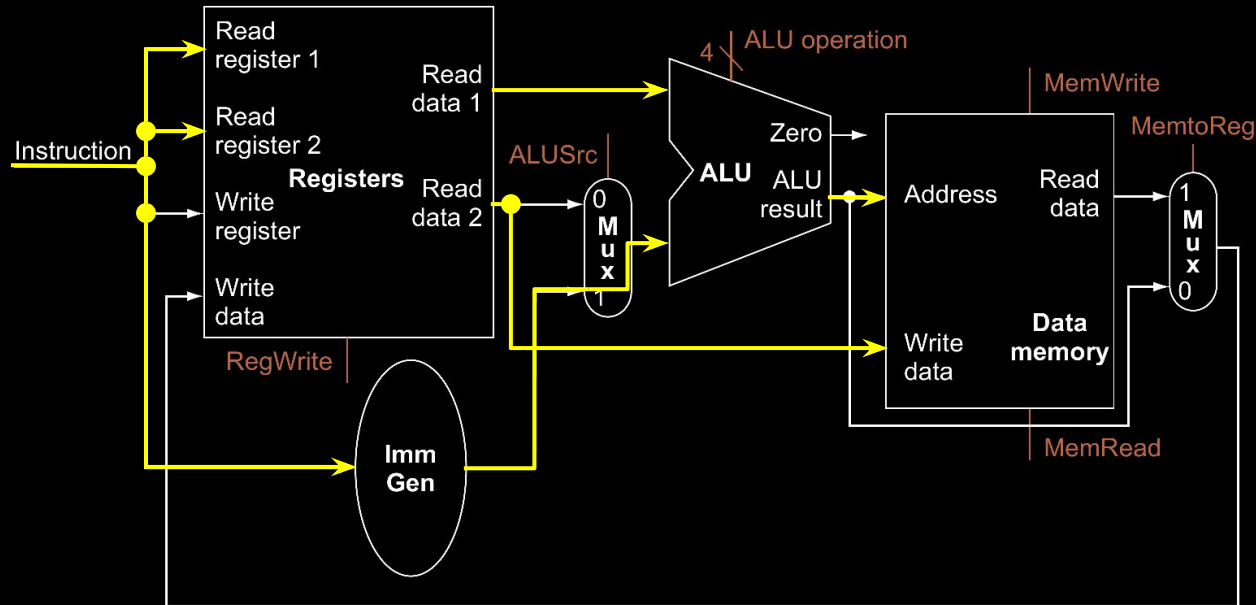
Instrução do Tipo-R



Instrução do Tipo-I com Load



Instrução do Tipo-S





Branches

Um **branch** soma o valor de deslocamento (que está no campo imediato) ao PC caso os registradores rs e rt sejam iguais.

```
beq rs1, rs2, DESLOCAMENTO
```

O endereço de deslocamento (offset) está em **half words**.

Deslocar 1x à esquerda para multiplicar por 2, para então obtermos o deslocamento em bytes.

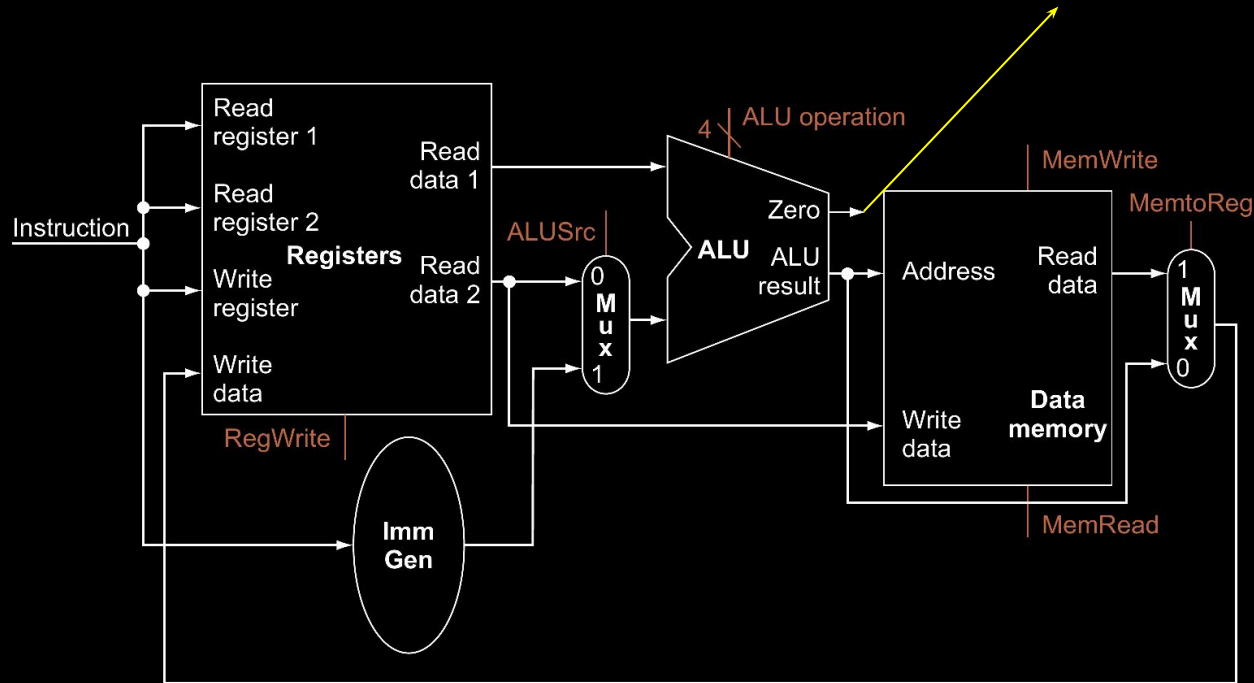
Unidade de Geração de Imediato.

Como comparar rs1 com rs2?

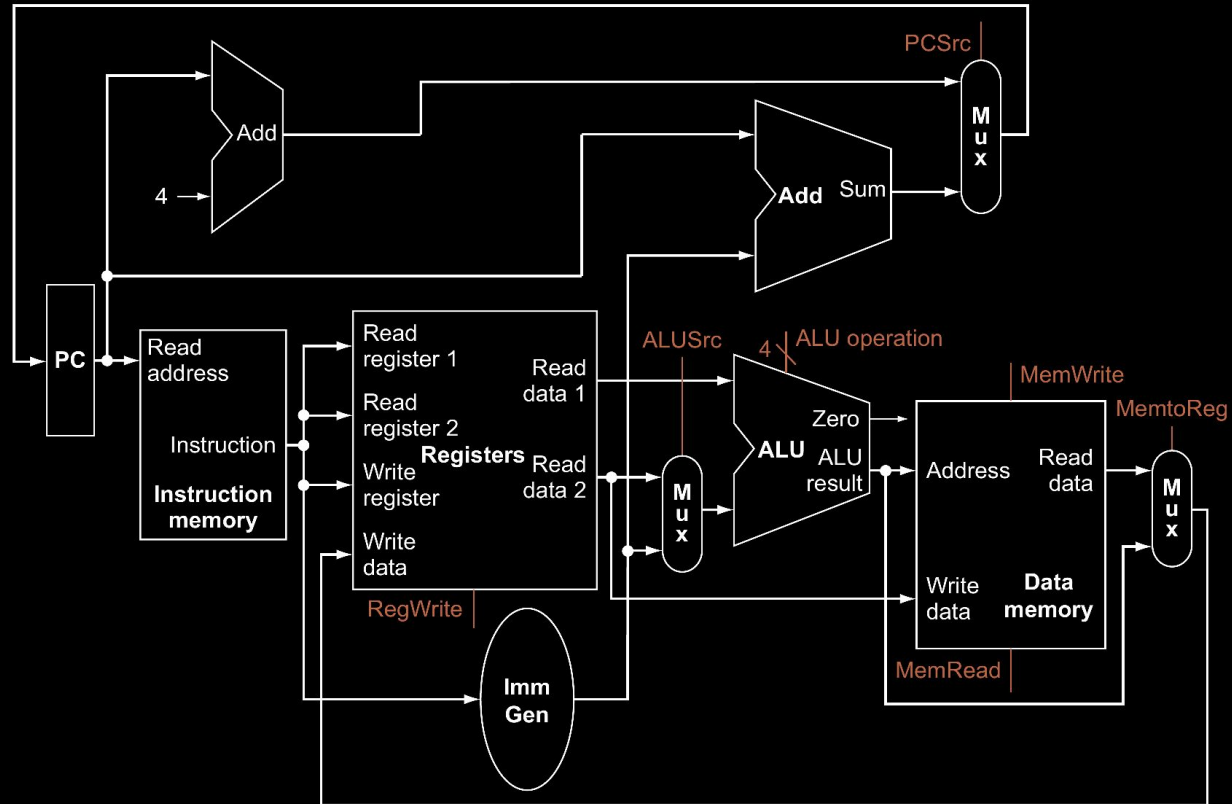
imediato[12 10:5]	rs2	rs1	funct3	imediato[4:1 11]	opcode
7 bits	5 bits	5 bits	3 bits	5 bits	7 bits

Ligando

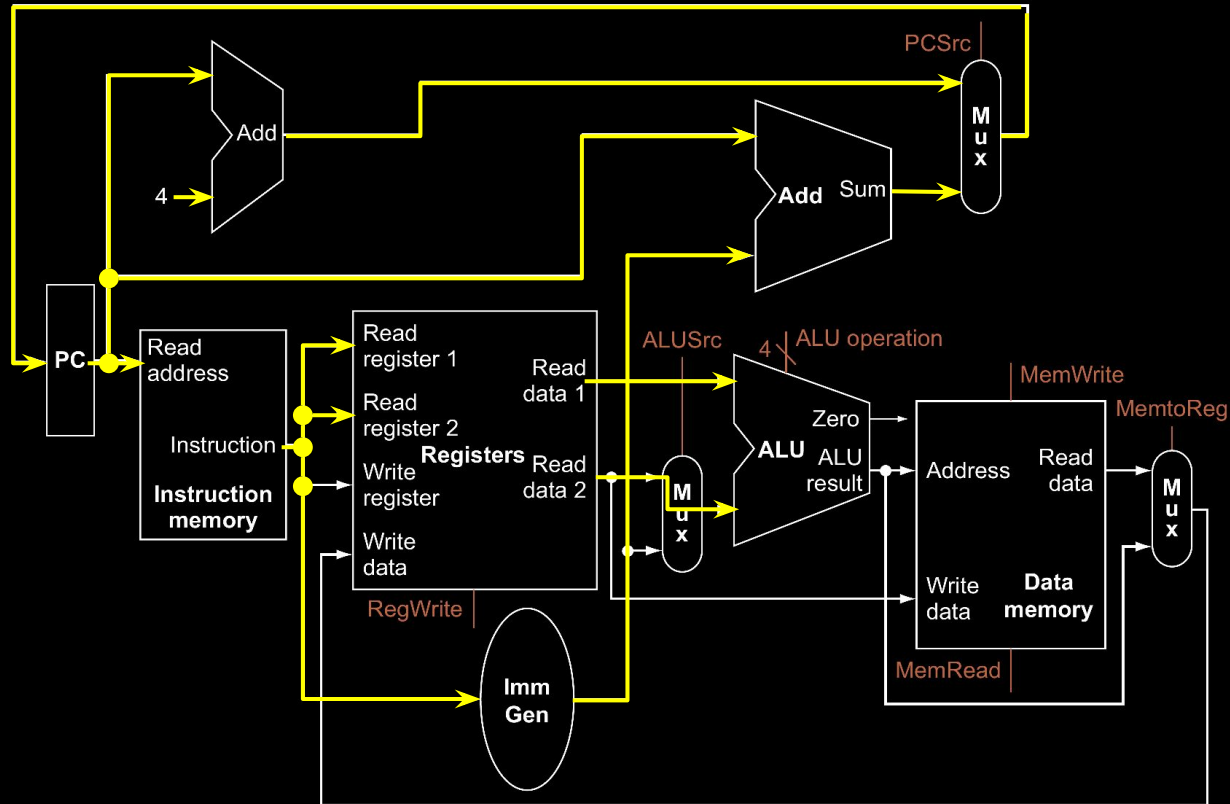
Resultado da subtração dos valores. Se for zero, o bit é setado, indicando que os valores são iguais.



Ligando

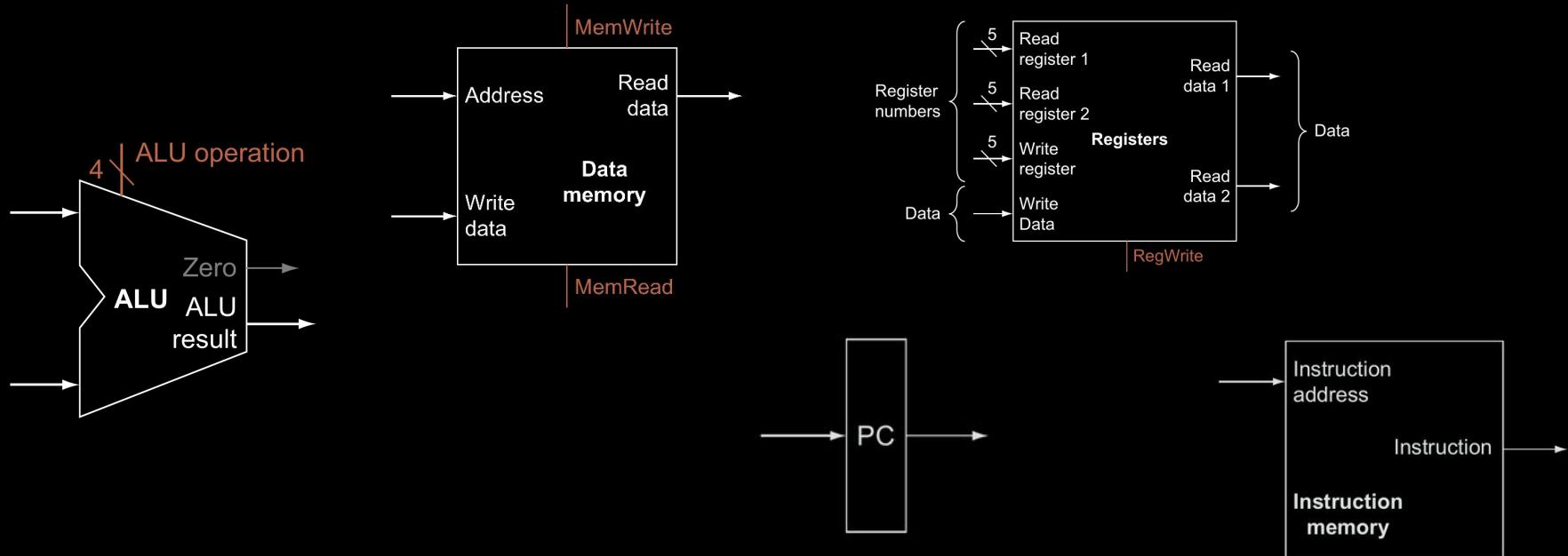


Instrução do Tipo-B



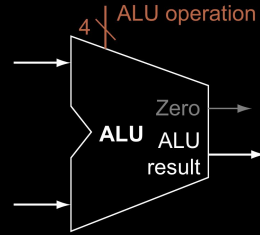
Exercícios

1. Sem olhar os slides anteriores, utilizando os seguintes componentes, monte novamente o processador RISC-V.

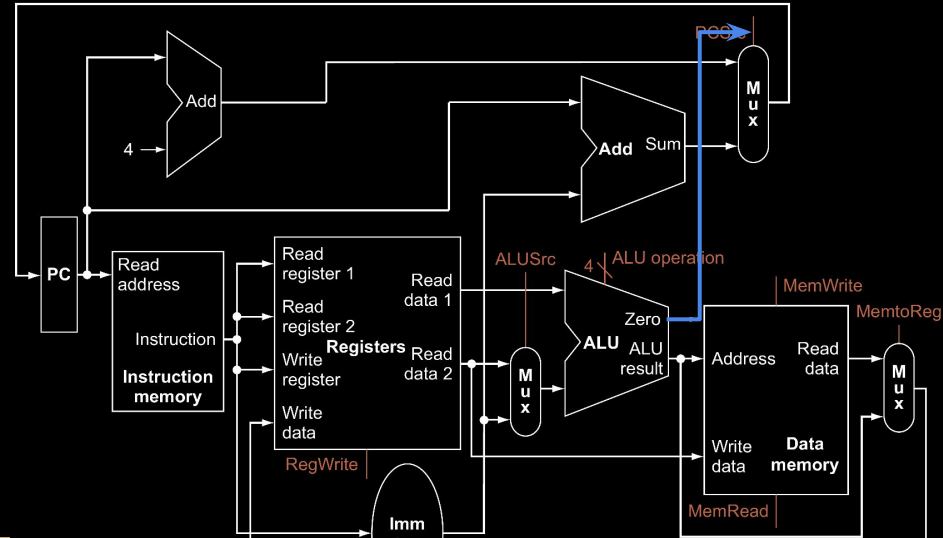


Exercícios

2. Os 4 bits que informam a ALU sobre qual operação deve ser executada podem vir de onde? Não precisa ligar no circuito, mas dê suas ideias sobre como poderíamos definir isso.



3. Considerando os branches, podemos ligar o seletor do multiplexador diretamente na saída zero da ALU, como feito em azul no circuito (a saída tem apenas 1 bit, indicando se o resultado da operação foi zero)? Sim? Não? Por quê?

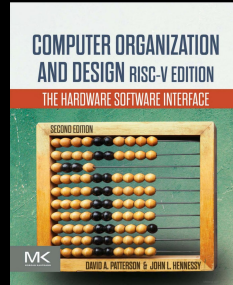


Exercícios

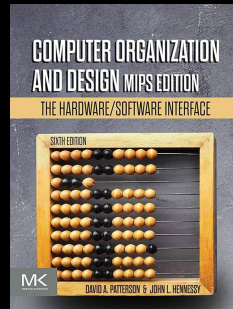
4. Esboce os circuitos dos componentes utilizados na construção do processador. Por exemplo, o PC pode ser um conjunto de 32 Flip-Flops do tipo D (para simplificar, considere cada flip-flop como uma caixa-preta, sem as pontas lógicas internas).
5. Defina a unidade Imm Gen. Para quais bits essa unidade precisa olhar para decidir que valores utilizar de imediato? Como fica a lógica para estender um dado de 12 bits que possui sinal para 32 bits? Para auxiliar, utilize a folha de instruções RISC-V disponibilizada na UFPR Virtual.

Referências

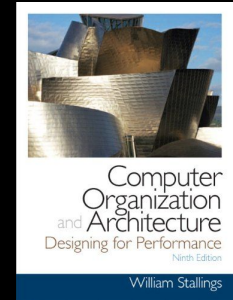
Patterson, Hennessy.
Computer Organization and
Design RISC-V Edition: The
Hardware Software
Interface. 2020.



Patterson, Hennessy. Computer
Organization and Design MIPS
Edition: The Hardware/Software
Interface. 2020.



Stallings, W. Organização de
Arquitetura de Computadores.
10a Ed. 2016.



Licença

Esta obra está licenciada com uma Licença [Creative Commons Atribuição 4.0 Internacional](https://creativecommons.org/licenses/by/4.0/).