

O Processador: Caminho de Dados e Controle

- *Durante este semestre vimos que o desempenho de uma máquina é determinado por três fatores principais:*
- *Números de instruções executadas.*
- *Período de clock (freqüência).*
- *Número de ciclos gastos por instrução (CPI).*

- Vimos também que o compilador e a arquitetura do conjunto de instruções afetam o funcionamento da máquina porque determinam o número de instruções executadas.
- Por outro lado, tanto o tempo do freqüência de clock, quanto o número de ciclos por instrução são determinados pelo processador.
- Por este motivo, vamos estudar o conceito de **caminho de dados** e verificaremos que a escolha da estratégia de implementação afeta a freqüência do clock e a CPI da máquina.

- *O estudo aqui apresentado inclui o projeto de implementação de um pequeno subconjunto das instruções do processador MIPS.*
- *Considera-se este subconjunto de instruções representativo dos princípios fundamentais de um projeto de um caminho de dados e de uma unidade de controle.*
- *Isto porque qualquer conjunto de instruções possui pelo menos estas três classes de instruções:*
 - *as de acesso à memória, as lógicas e aritméticas, e as de transferência de controle.*

- *Lembre-se sempre que descrever uma arquitetura implica em definir quais são os seus componentes, a funcionalidade de cada um deles e como eles interagem entre si.*

Visão Geral de Implementação

- *Para qualquer tipo de instrução a ser executada, o processador passa inicialmente por dois passos:*
- *Enviar o valor armazenado no PC (program counter) para a memória que contém o código, trazendo para o caminho de dados a instrução armazenada nesta memória.*
- *Ler um ou dois registradores, usando os campos de instrução para selecionar os registradores a serem lidos.*

- *Após a execução desses passos, as ações necessárias para completar a execução de uma instrução dependem da classe de instrução em curso, mas deve-se lembrar que mesmo entre classes diferentes existem algumas semelhanças. Por exemplo, todas as classes usam a ULA após a leitura dos registradores:*
- *As instruções de referência à memória usam a ULA para calcular o endereço.*
- *As aritméticas e lógicas para execução da própria operação.*
- *As de desvio para efetuar comparações.*

- *Após usar a ALU, as ações necessárias para completar a execução das instruções de cada classe são diferentes.*
- *Uma instrução que referencia a memória precisa realizar o acesso à memória.*

- Nas instruções lógicas e aritméticas é preciso armazenar o resultado produzido pela ALU num registrador.
- E, numa instrução de desvio pode ser necessário modificar o endereço da próxima instrução a ser buscada da memória.

- Uma primeira visão do caminho de dados de um computador nos levaria ao seguinte modelo simplificado (Figura 1).

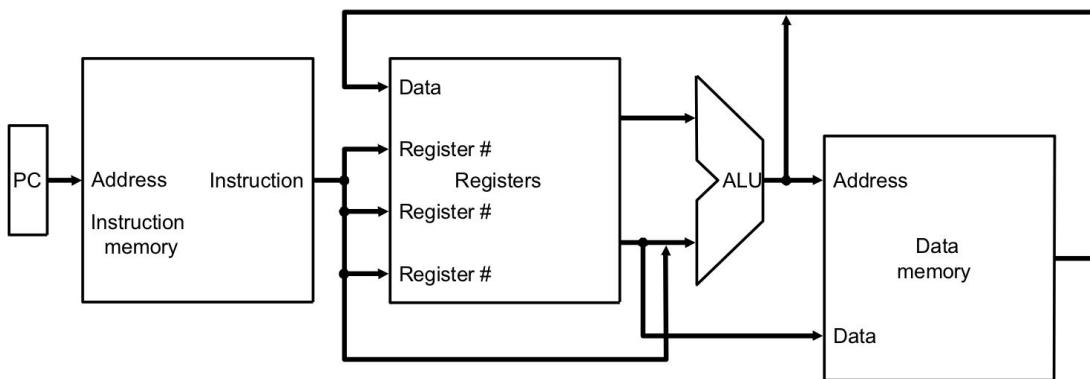


Figura 1 – Caminho de Dados Básico.

- As unidades funcionais componentes do MIPS são construídas a partir de:
 - Elementos que operam sobre dados (combinacionais).
 - Elementos de estado (seqüenciais e de memorização).
 - O sinal de clock determina quando um elemento de estado é lido ou escrito, determinando a cadência de funcionamento do hardware.

Metodologia de Temporização

- A *metodologia de temporização define quando os dados podem ser lidos ou escritos em cada um desses componentes.*
- *O mais importante dos dispositivos de memória é o flip-flop, que pode ser construído com a ajuda de um conjunto de portas lógicas básicas.*
- *Embora, individualmente, tais portas não tenham capacidade de memória, a conexão adequada entre elas permite o armazenamento de informações.*

- A partir do flip-flop básico apresentado na figura abaixo, pode-se compreender os princípios que norteiam o projeto e funcionamento de outros tipos de flip-flop, como o JK, o mestre-escravo, o T e o D.

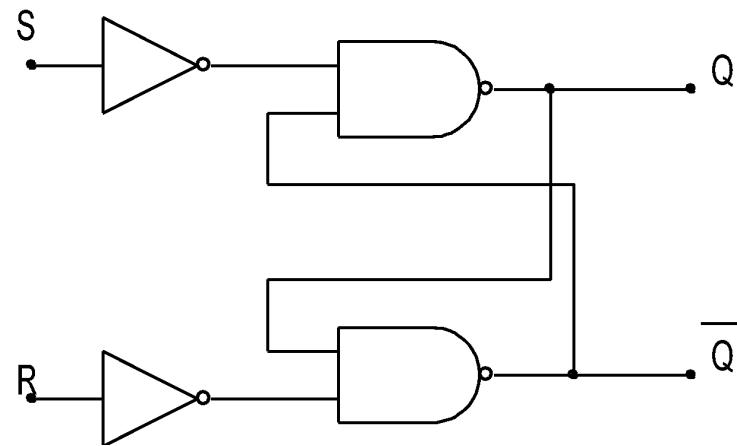


Figura 2 – Flip-Flop RS.

- A metodologia mais simples é àquela sensível às transições do sinal de clock, que é demonstrada abaixo.
- A atualização do estado armazenado somente nas transições não cria condições de corrida, o que evita a indeterminação dos valores dos dados.

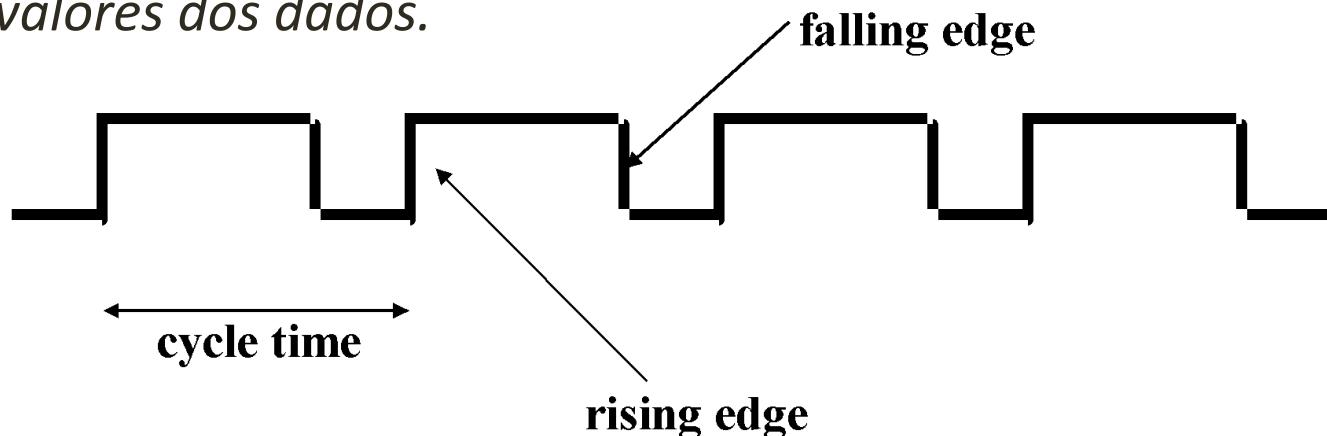


Figura 3 – Metodologia de Temporização.

- Todos os sinais precisam se propagar do elemento de estado 1 para o 2 através da lógica combinacional durante o tempo de um ciclo de clock, sem que haja realimentação e é isto que determina a freqüência de clock de uma máquina.

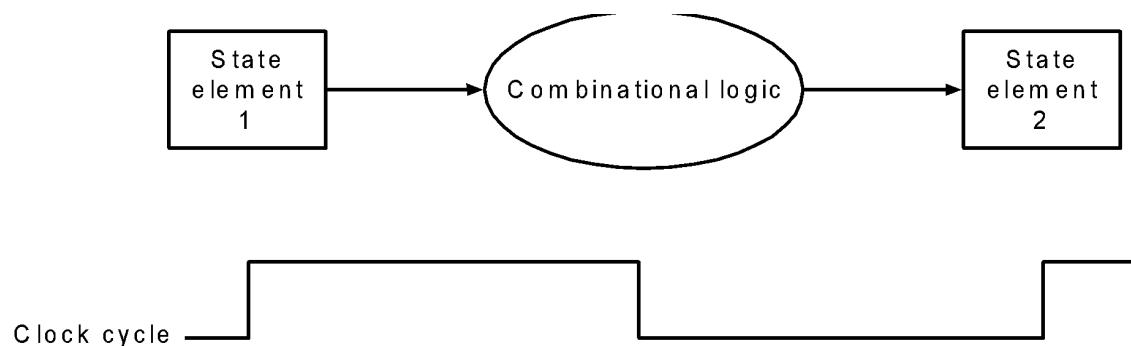


Figura 4 – Propagação dos Dados.

Implementação

- *Inicialmente mostraremos uma implementação bem simples do subconjunto de instruções.*
- *Ela é baseada num único ciclo de clock que é suficiente longo para atender todas as instruções consideradas (projeto monociclo).*

- Assim, toda instrução começa sua execução em uma transição ativa do clock e termina na próxima transição ativa do clock.
Dessa forma, todas as instruções gastam o mesmo tempo para serem executadas.
- Utilizando esta metodologia, é necessário que o tempo escolhido para determinar a freqüência de clock seja o da instrução mais demorada.
 - Esta implementação é apresentada por ser mais simples de entender, mas não é implementada na realidade porque é muito ineficiente.

Busca da Instrução

- *O primeiro passo de qualquer instrução é a busca.*
- *Para tanto são necessários três componentes: dois elementos de estado e um somador.*
- *Um dos elementos de estado armazena instruções (memória de instruções) e o outro armazena o endereço da próxima instrução a ser executada (Program Counter - PC).*
- *O terceiro componente calcula o endereço da próxima instrução a ser executada (somador).*
- *A tarefa do somador é incrementar o valor do PC.*
- *Ele pode ser visto como uma ULA muito simples que só realiza operação de soma.*
- *O valor 4 numa das entradas do somador indica que a próxima instrução está armazenada 4 bytes depois do valor corrente do PC.*

Instruções Aritméticas

- *Já examinamos os principais elementos básicos de um caminho de dados, em que foi possível perceber que toda e qualquer instrução começa com a busca da instrução na memória.*
- *Para preparar a próxima instrução, incrementa-se o PC, fazendo-o apontar para a próxima instrução (4 bytes depois do endereço corrente).*

- Agora vamos analisar as instruções do tipo R que precisam ler 2 registradores, realizar uma operação na ULA e escrever o resultado em um registrador.
- Deve-se considerar que estas instruções possuem 3 operandos em registradores, dessa forma conclui-se que é necessário ler 2 palavras de dados do banco de registradores.
- Para realizar tal leitura, é necessária uma entrada para o banco de registradores que especifique o número do registrador a ser lido, além de uma saída do banco de registradores para o valor lido.

- *Para escrever uma palavra no banco de registradores, são necessárias 2 entradas.*
- *A primeira especifica o número do registrador a ser escrito.*
- *A segunda fornece os dados a serem escritos no registrador.*

- *O banco de registradores sempre coloca na saída os conteúdos dos registradores solicitados.*
- *O processo de escrita, no entanto, é controlado pelo sinal de controle de escrita, que é habilitado na transição positiva do clock.*
- *Pelo exposto, conclui-se que o banco de registradores necessita de quatro entradas e duas saídas.*

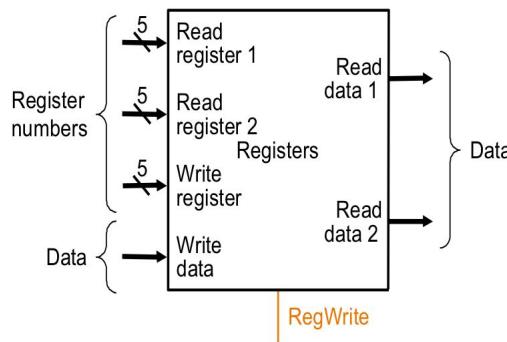


Figura 5 – Detalhamento do Banco de Registradores.

- Além disso, o caminho de dados para uma instrução tipo R também precisa de uma ULA, sendo então composto por:

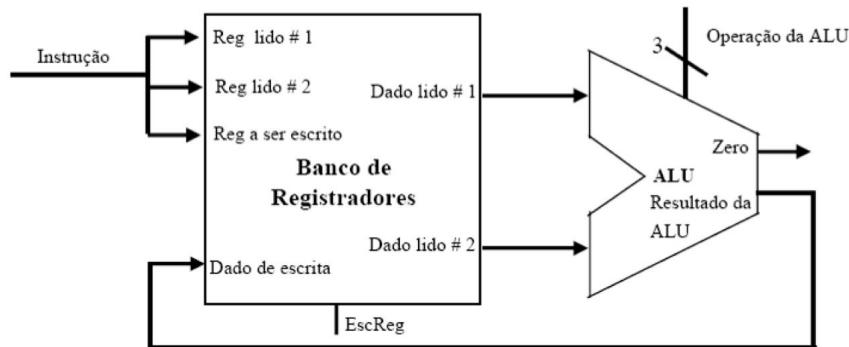


Figura 6 – Detalhamento do Banco de Registradores e ULA.

- *Estas últimas figuras demonstram que o “tamanho” do barramento que interliga duas unidades funcionais da máquina digital varia de acordo com sua função, embora todas as palavras tenham 32 bits.*
- *A escrita em um registrador precisa ser indicada explicitamente.*
- *Isto é feito através do sinal de controle de escrita (EscReg).*

- *O número do registrador que vai ser escrito e o sinal de controle de escrita precisam estar válidos nas transições ativas do clock.*
- *A operação realizada pela ALU é controlada por um sinal que contém 3 bits.*
- *A saída da ALU para detecção do valor zero é usada para implementação de desvios condicionais.*

Instruções de Acesso à Memória

- Agora consideremos as instruções *load word* e *store word*, que têm as formas:
 - *lw \$t1, deslocamento (\$t2)*
 - *sw \$t1, deslocamento (\$t2)*

- *Tais instruções calculam um endereço de memória somando o conteúdo do registrador base, ao número de 16 bits (sem sinal) correspondente ao deslocamento a partir do endereço base.*
- *Se a instrução for store word, o valor a ser armazenado na memória precisa ser lido do banco de registradores.*
- *Se a instrução for load Word, o valor lido da memória deve ser escrito no registrador especificado.*
- *Portanto, para este tipo de instrução são necessários o banco de registradores e a ULA.*
- *Além disso, precisamos de uma memória de dados e uma unidade de extensão de sinal para estender o campo de 16 bits do deslocamento para um valor de 32 bits com sinal.*

- A Figura 7 mostra estes dois novos elementos combinados com os elementos anteriormente estudados.

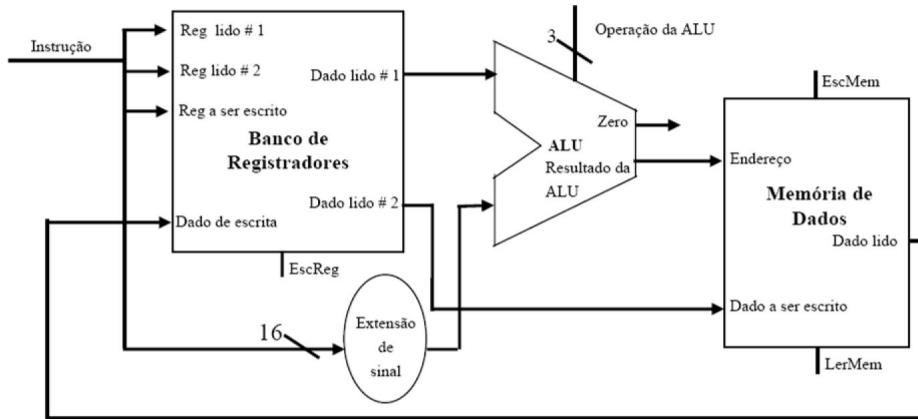


Figura 7 – Detalhamento do Banco de Registradores e ULA.

- *O número dos registradores de entrada do banco de registradores vem dos campos da própria instrução, assim com o deslocamento.*
- *O valor especificado no campo de deslocamento após ter seu sinal estendido torna-se a segunda entrada da ALU.*

Instruções de Desvio

- *Uma instrução de desvio condicional como, por exemplo, beq, possui três operandos.*
- *Dois registradores cujos conteúdos são comparados, e um deslocamento de 16 bits usado no cálculo do endereço alvo de desvio.*
- *Para implementação desta instrução, então, precisamos calcular o endereço alvo do desvio.*
- *Esse endereço é obtido somando o campo de deslocamento (com sinal estendido) da instrução ao valor armazenado no PC.*

- *Existem dois detalhes que precisam ser lembrados:*
 - *a arquitetura do conjunto de instruções estabelece que a base para o cálculo do endereço alvo do desvio é igual ao valor do PC atualizado ($PC + 4$, passo de busca); e*
 - *a arquitetura também define que o campo de deslocamento deve ser deslocado de 2 bits à esquerda. Isto significa um deslocamento relativo à palavra do processador.*

- A Figura 8 mostra o caminho de dados para uma instrução de desvio condicional.

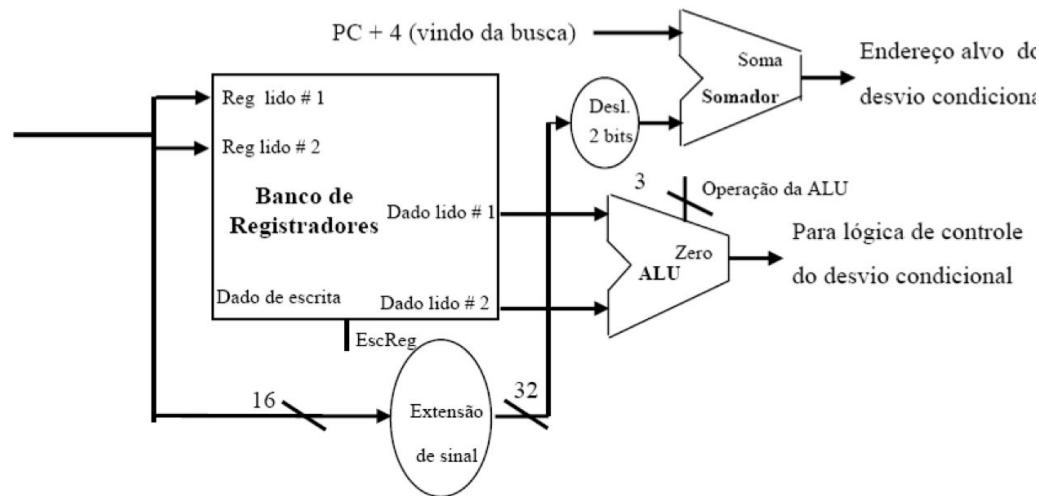


Figura 8 – Detalhamento do Banco de Registradores e ULA.

- Além de calcular o endereço alvo do desvio condicional, é necessário determinar se a próxima instrução a ser executada segue imediatamente a instrução atual ou se é a instrução armazenada no endereço alvo do desvio.
- Quando a condição é verdadeira o endereço alvo do desvio calculado deve ser armazenado no PC.
- Caso contrário, o valor do PC, incrementado no passo de busca, não deve ser substituído.

Controle da ULA

- Dependendo da classe de instrução, a ULA da máquina vai precisar executar uma das cinco funções básicas:

000	AND
001	OR
010	Soma
110	Subtracao
111	Set Less Than

- Para as instruções “lw” e “sw”, usa-se a ULA para calcular o endereço de memória por adição.
- No caso das instruções tipo R, a ULA pode executar qualquer uma das funções aritméticas ou lógicas, dependendo do valor correspondente dos 6 bits do campo “funct”, situado na parte menos significativa de cada instrução.

- *O funcionamento da ULA é controlado por 3 bits gerados por uma pequena unidade de controle, que recebe como entrada o campo de função da instrução e um campo de controle de 2 bits conhecido como ULAop.*
- *O uso de pequenas unidades de controle contribui para aumentar a velocidade da unidade de controle principal.*

- Existem muitas maneiras de se implementar o mapeamento dos bits de controle da ULA.
- Observando o fato de que somente algumas das 64 combinações possíveis dos bits de função são de interesse, pode-se utilizar uma lógica combinacional relativamente simples.
- Para tanto, o passo mais importante é a utilização de uma tabela da verdade e, a partir dela, obter o circuito desejado.

Projeto da Unidade de Controle

- *Para completar o projeto da unidade de controle é importante considerar o formato das instruções do processador (relembrar número de bit).*

Figura 9 – Formatos de Instruções.

- *Considerando estes diferentes formatos podemos adicionar ao caminho de dados a identificação da instrução e um multiplexador extra.*
- *Este multiplexador deve ser colocado na entrada do registrador a ser escrito no banco de registradores.*

- A entrada do multiplexador vem do campo rd [15-11], se a instrução for lógica ou aritmética, e do campo rt [20-16], se a instrução for de acesso à memória.
- Dessa forma, selecionamos qual dos campos da instrução é usado para indicar o número do registrador a ser escrito. O sinal de controle deste multiplexador é denominado *RegDest*.

CRIAÇÃO DE UM CAMINHO DE DADOS ÚNICO

- *Vimos até o momento os componentes necessários para a construção do caminho de dados para dois diferentes tipos de instrução. Agora veremos a idéia básica de um caminho de dados global.*
- *O mais simples caminho de dados possível deve se propor a executar todas as instruções dentro de um único período de clock, o que significa de nenhum dos recursos pode ser utilizado mais de uma vez por instrução, de modo que todo elemento que for necessário mais de uma vez dentro do mesmo período, deverá ser replicado.*

- *Esta é a razão de precisarmos de uma memória para instruções e outra para dados.*
- *Quando tal técnica é aplicada, algumas unidades funcionais podem precisar ser duplicadas, mas alguns elementos podem ser compartilhados em razão dos diferentes fluxos de execução das instruções.*
- *Para compartilhar um elemento do caminho de dados entre diferentes classes de instrução, apela-se para o multiplexador, que permite múltiplas conexões para o mesmo dispositivo.*

- Um exemplo de composição de caminho de dados pode ser feito com as instruções tipo R e de acesso à memória, que são bastante semelhantes. As principais diferenças são:
- A segunda entrada para a ULA é um registrador (se a instrução for do tipo R) ou extensão da instrução (se a instrução for de referência à memória).
- O valor armazenado no registrador de destino vem da ULA (se a instrução for tipo R) ou da memória (no caso de um load word).

- *Para combinar esses dois caminhos de dados usando somente um banco de registradores e uma ULA, há necessidade de se ter duas fontes diferentes para a segunda entrada da ULA, bem como duas fontes diferentes para o dado a ser armazenado no banco de registradores.*

- A solução para este impasse é a utilização de multiplexadores

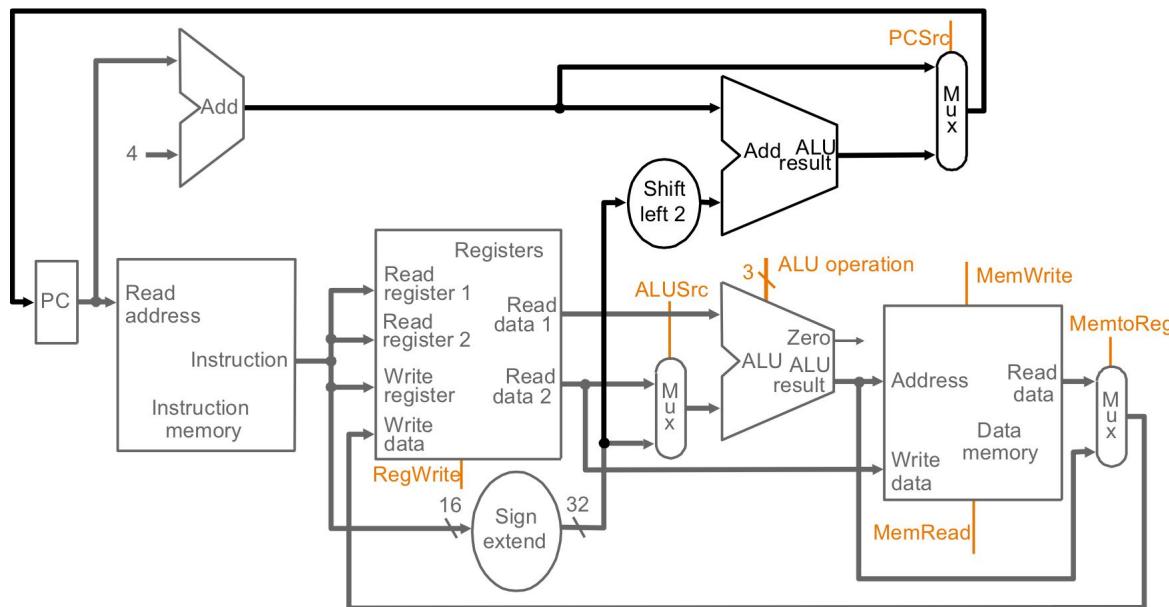


Figura 10 – Caminho de Dados Para Máquina MONOCICLO.

- *A este caminho simplificado, ora apresentado, ainda é necessário acrescentar o PC e o somador.*
- *Apesar do modo de construção do caminho de dados mostrado parecer interessante para o entendimento do seu projeto, ele não é adotado nas máquinas modernas por ser é ineficiente.*

- *Para entender o porquê da ineficiência, observe que o ciclo de clock deve ter o mesmo tamanho para todas as instruções. Por conta disso, a CPI da máquina será 1.*
- *Nesse contexto, o ciclo de clock será determinado levando-se em conta a instrução cujo tempo de execução for o maior de todos. Em geral, tal instrução trata de acesso à memória (lw e sw).*
- *Apesar de sua CPI ser igual a 1, a performance global das implementações monociclo não é muito boa porque a maioria das outras instruções poderiam ser executadas em um período de clock menor.*

A IMPLEMENTAÇÃO MULTICICLO

- *No início da análise do caminho de dados, dividimos cada instrução em uma série de passos que correspondiam às operações das unidades funcionais.*
 - *Tais passos podem ser utilizados para criar uma implementação MULTICICLO.*
- *Neste tipo de implementação, cada passo de execução gasta 1 período do clock. A implementação MULTICICLO permite que uma unidade funcional seja utilizada mais de uma vez por instrução, uma vez que ela está sendo usada em ciclos diferentes do clock.*

- *Em resumo as principais vantagens da implementação MULTICICLO são a possibilidade de fazer com que as instruções sejam executadas em quantidades diferentes de períodos de clock, e a capacidade de compartilhar unidades funcionais dentro do espaço de tempo necessário à execução de uma única instrução.*

- A figura 11 abaixo mostra uma versão abstrata do caminho de dados multiciclo:

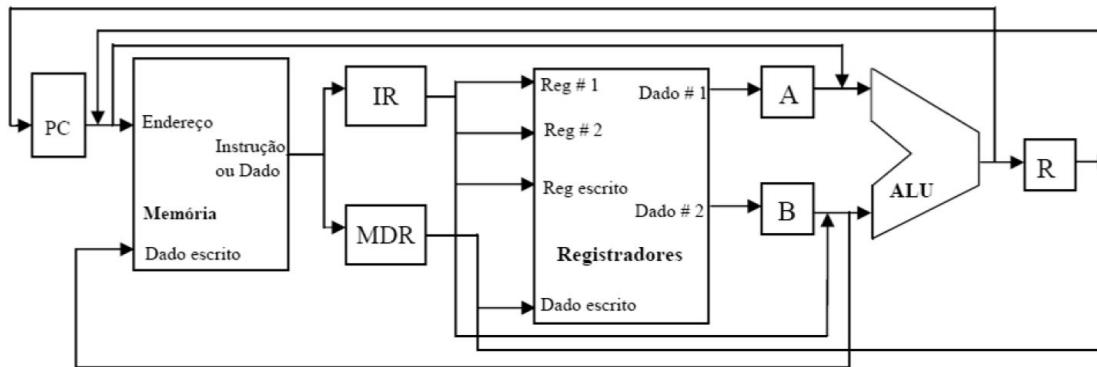


Figura 11 – Visão do Caminho de Dados Para Máquina MULTICICLO.

- Comparando esta figura com o caminho de dados monociclo podemos observar as seguintes diferenças:
 - uso de uma única memória tanto para instruções quanto para dados;
 - referência a uma única ALU e colocação de um ou mais registradores depois de cada unidade funcional para armazenar temporariamente o resultado calculado até que seja utilizado em um ciclo de clock subsequente.
 - No final de um ciclo de clock todos os dados que precisam ser usados em ciclos subsequentes devem ser armazenados em um elemento de estado.
- ◆ Os dados a serem usados em outras instruções devem ser armazenados em elementos de estado visíveis aos programadores (banco de registradores, PC ou memória).

EXCEÇÕES

- *O controle é um dos maiores desafios do projeto de um processador. Uma das partes mais difíceis do controle é a implementação das exceções e das interrupções, que são eventos que mudam o fluxo normal de execução de instruções.*
- *Uma exceção é um evento inesperado que vem de dentro do processador, como o overflow aritmético.*
 - Já uma interrupção é um evento semelhante à exceção, mas que tem origem fora do processador.
 - São normalmente utilizadas pelos dispositivos de E/S para se comunicar com o processador.

- A ação básica que a máquina precisa tomar na ocorrência de uma exceção é o salvamento dos endereços da instrução envolvida num registrador especial (EPC), então transferir o controle para o sistema operacional, que por sua vez executa ações apropriadas, reportando o erro.

- *Após a execução das ações necessárias, para que o sistema operacional possa tratar a exceção, é necessário que ele conheça sua origem. Isto é feito através de:*
- *utilização de registrador de STATUS (caso do MIPS), que indica o motivo.*
- *utilização de interrupções vetoradas, em que o endereço para o qual o controle é transferido é determinado pela causa da exceção.*

CONCLUSÕES

- *As técnicas descritas para a construção de caminho de dados e unidades de controle são o cerne do projeto de qualquer computador. No entanto, todos os computadores atuais vão pouco a pouco além dessas técnicas, e usam pipeline em seus projetos.*
- *O último processador da Intel na arquitetura 80x86, sem pipeline, foi o 80386, lançado em 1985. O primeiro processador MIPS R200, também lançado em 1995 já usava pipeline.*