Decodificador & Memórias

Kaike Carvalho, Felipe Kenzo Suguimoto, Eduardo Knabben Tiyo

Universidade Tecnológica Federal do Paraná – UTFPR

COCIC – Coordenação do Curso de Bacharelado em Ciência da Computação

Campo Mourão, Paraná, Brasil

kaikecarvalho@alunos.utfpr.edu.br
felipekenzo@alunos.utfpr.edu.br
tiyo@alunos.utfpr.edu.br

Resumo

Neste relatório, irá se apresentar 4 partes de um datapath (Decodificador de Instruções, Unidade de Extensão de Sinal, Memória de Instruções e memória de dados), juntamente com suas imagens e especificações.

1. Introdução

Os elementos descritos abaixo são essenciais para um datapath monociclo, que é um tipo de implementação de unidade de processamento em um processador, onde cada instrução é executada em um único ciclo de clock. Ou seja, significa que todas as etapas do ciclo de instrução, como busca, decodificação, execução e gravação de resultados, são concluídas em um único pulso de clock.

2. Especificação

2.1 Decodificador de Instruções

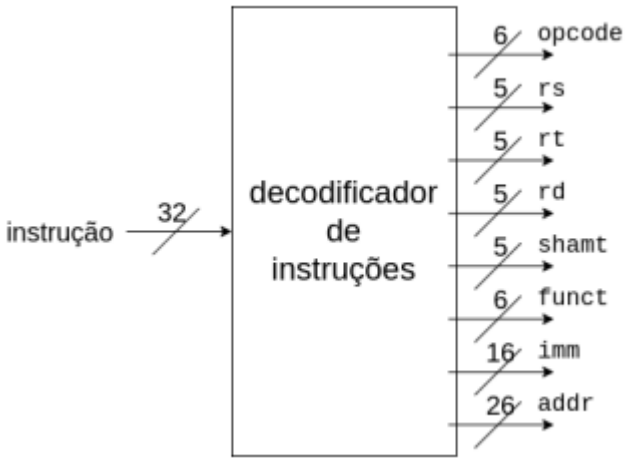

Figura 1 – Entradas e saídas do decodificador de instruções.

O decodificador de instruções é um componente que recebe os bits da instrução e decodifica em (opcode, rs, rt, rd, shamt, imm, addr), sendo o opcode comum para as 3 instruções existentes no MIPS (R, I e J).

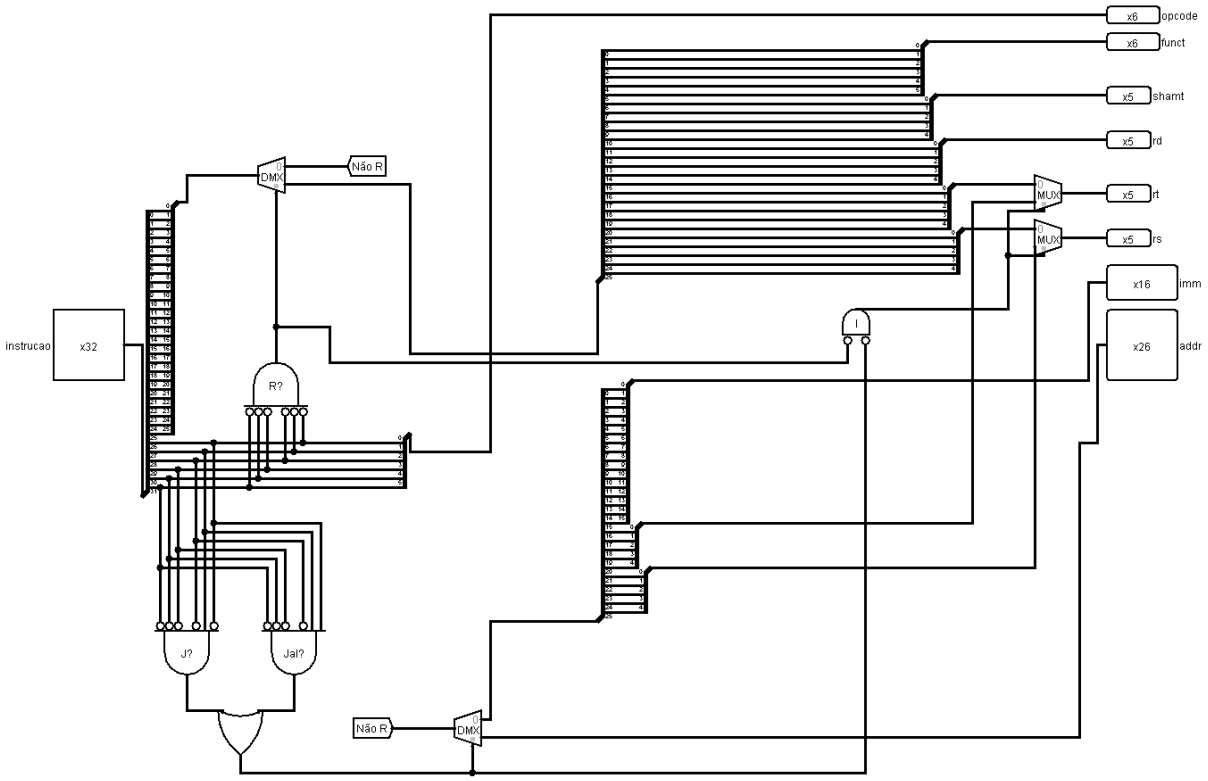

Figura 1.1 - Decodificador

Conforme a Figura 1.1, os 32 bits da instrução passados do PC são distribuídos em uma estrutura de dados. Os seis bits mais significativos correspondem ao opcode, que é comum para três tipos de instruções. Esses bits são diretamente conectados à entrada do opcode. No conjunto dos seis bits mais significativos, uma porta AND inteira negada é empregada para detectar a instrução do tipo R (opcode 000000). Se o opcode for 000000 (indicando uma instrução do tipo R), o sinal na saída da porta AND "R?" será ativado. Esse sinal é então direcionado para um DEMUX, que seleciona entre instruções não-R ou R.

Além disso, um bloco lógico OR é utilizado, composto por duas portas AND, para identificar os opcodes 2 e 3 (J e JAL). A saída deste bloco OR é conectada a outro DEMUX. Esse DEMUX, quando ativado, diferencia entre operações não-R, identificando se a instrução é do tipo I ou J com base na saída do bloco OR. Se a saída do bloco OR for 1, a instrução é do tipo J; caso contrário, é do tipo I.

É importante notar que as entradas "rs" e "rt" possuem multiplexadores (MUX) para determinar se os bits devem ser utilizados para uma instrução do tipo I ou R. Para essa seleção, uma porta AND de duas entradas, ambas negadas, é empregada. Se a instrução não for do tipo R (indicado pelo sinal na saída da AND "R?") nem do tipo J (indicado pelo sinal na saída do bloco OR "J?" e "Jal?"), ela será uma instrução do tipo I. Por fim, as instruções do tipo J utilizam os bits restantes para formar o endereço de destino (addr).

A tabela abaixo resume a distribuição dos bits para cada tipo de instrução:

Tipos de Instrução	Bits do OPCODE	Bits da Instrução
R	000000	[0-5] funct, [6-10] shamt, [11-15] rd, [16-20] rt, [21-25] rs
I	Diferentes de 000000 e 110000	[0-15] imm, [16-20] rt, [21-25] rs
J	000010(J) ou 000011(JAL)	Restantes para o endereço(addr)

2.2 Extensor de Sinal

No MIPS, um extensor de sinal é usado para estender números imediatos de 16 bits para 32 bits, mantendo o valor original. Se o bit mais significativo do número de 16 bits for 1, os bits extras na extensão (bits 16 a 31) também serão 1. Se o bit for 0, os bits extras na extensão serão 0.

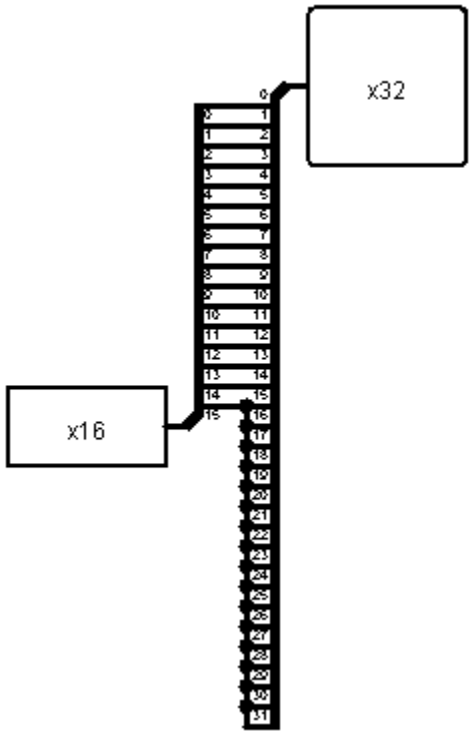

Figura 1.2: Extensor de Sinal

2.3 Memória de Instruções

A memória de instruções (Figura 1.3) é estruturada em três partes distintas: o campo de endereço, a memória ROM e a instrução. O campo de endereço possui 32 bits, enquanto a ROM suporta até 24 bits. Devido a essa diferença de tamanho, é necessário empregar dois distribuidores: um de 32 bits, que recebe o endereço completo, e outro de 24 bits, responsável por enviar dados para a entrada da ROM.

É importante notar que o distribuidor conectado à ROM é vinculado do bit 2 ao 24 do distribuidor principal. Esse arranjo é implementado para permitir que o ROM avance de 4 em 4, garantindo uma leitura sequencial e correta das instruções. Essa organização da memória de instruções é essencial para o funcionamento eficiente do sistema.

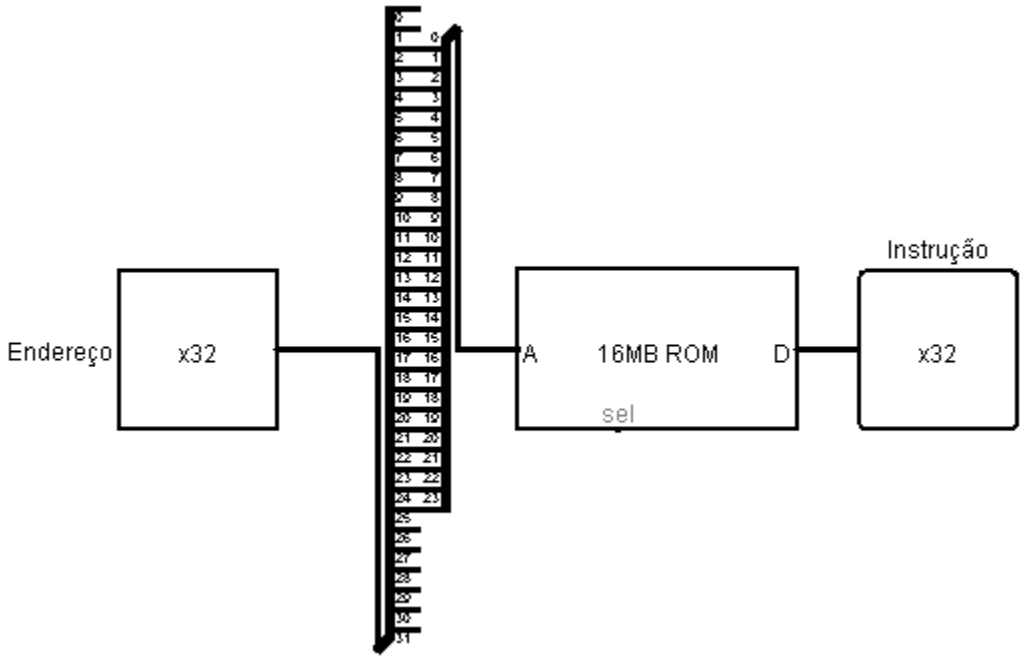

Figura 1.3: Memória de Instruções

2.4 Memória de Dados

A memória de dados (Figura 1.4) é um componente essencial projetado para armazenar e manipular dados, permitindo a leitura e modificação de informações em diferentes endereços de memória. É composta por três elementos principais: o registrador EL, que contém 32 bits e atua como índice para os endereços de memória; o registrador WD, que também possui 32 bits e representa os dados a serem escritos na memória no endereço especificado durante operações de escrita; e a memória RAM, que possui duas entradas, uma para o endereço (com 24 bits) e outra para os dados (com 32 bits).

Para garantir compatibilidade com a entrada de endereço da memória RAM, é necessário utilizar dois distribuidores para ajustar o formato do registrador EL. Além disso, a memória RAM possui três entradas adicionais: "memory write" para escrever os dados no endereço especificado, "memory read" para ativar ou desativar a saída e um sinal de clock para sincronização. Essa configuração cuidadosamente elaborada permite operações eficientes de leitura e escrita, garantindo que o sistema possa acessar e manipular dados de maneira confiável e precisa.

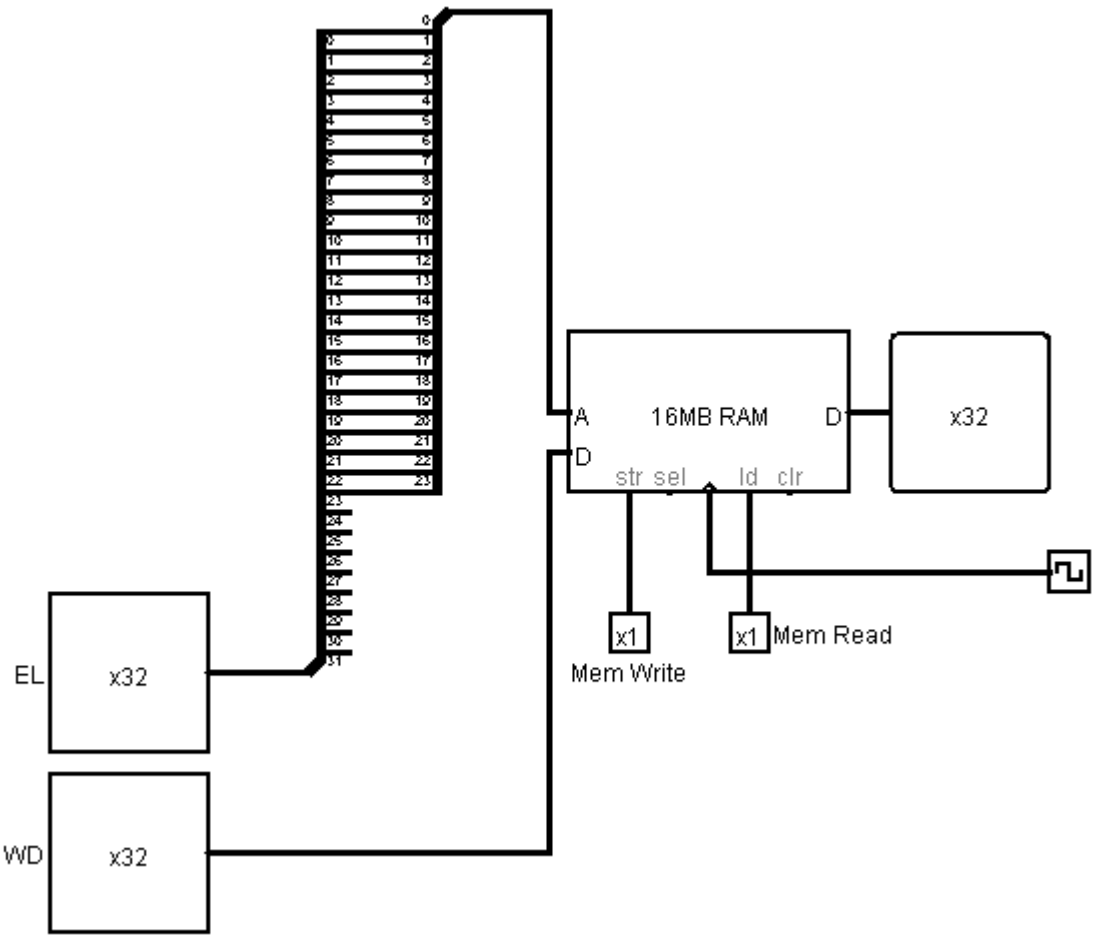

Figura 1.4: Memória de Dados

3 Conclusão

Neste relatório, exploramos quatro componentes essenciais de um datapath: o Decodificador de Instruções, a Unidade de Extensão de Sinal, a Memória de Instruções e a Memória de Dados. Cada componente desempenha um papel importante na execução eficiente de operações em um processador monociclo. O Decodificador de Instruções é responsável por interpretar os códigos de operação e direcionar as instruções para o caminho apropriado. A Unidade de Extensão de Sinal garante que números imediatos sejam estendidos corretamente para operações aritméticas e lógicas. A Memória de Instruções armazena sequências de instruções, enquanto a Memória de Dados gerencia a leitura e escrita de dados.

4 Referências

[1] Patterson, David A. Hennessy, John L. Organização e Projeto de Computadores.