

Desenvolvimento e Síntese de Processador RISC 32-bits

Disciplina EGM0018 – Projeto e Síntese de Sistemas Digitais

Objetivos do Projeto



Arquitetura

Projetar um processador baseado em arquitetura RISC de 32 bits, com implementação multiciclo para otimização de recursos.



Implementação

Desenvolver o código RTL (Register Transfer Level) utilizando a linguagem de descrição de hardware Verilog.



Síntese Física

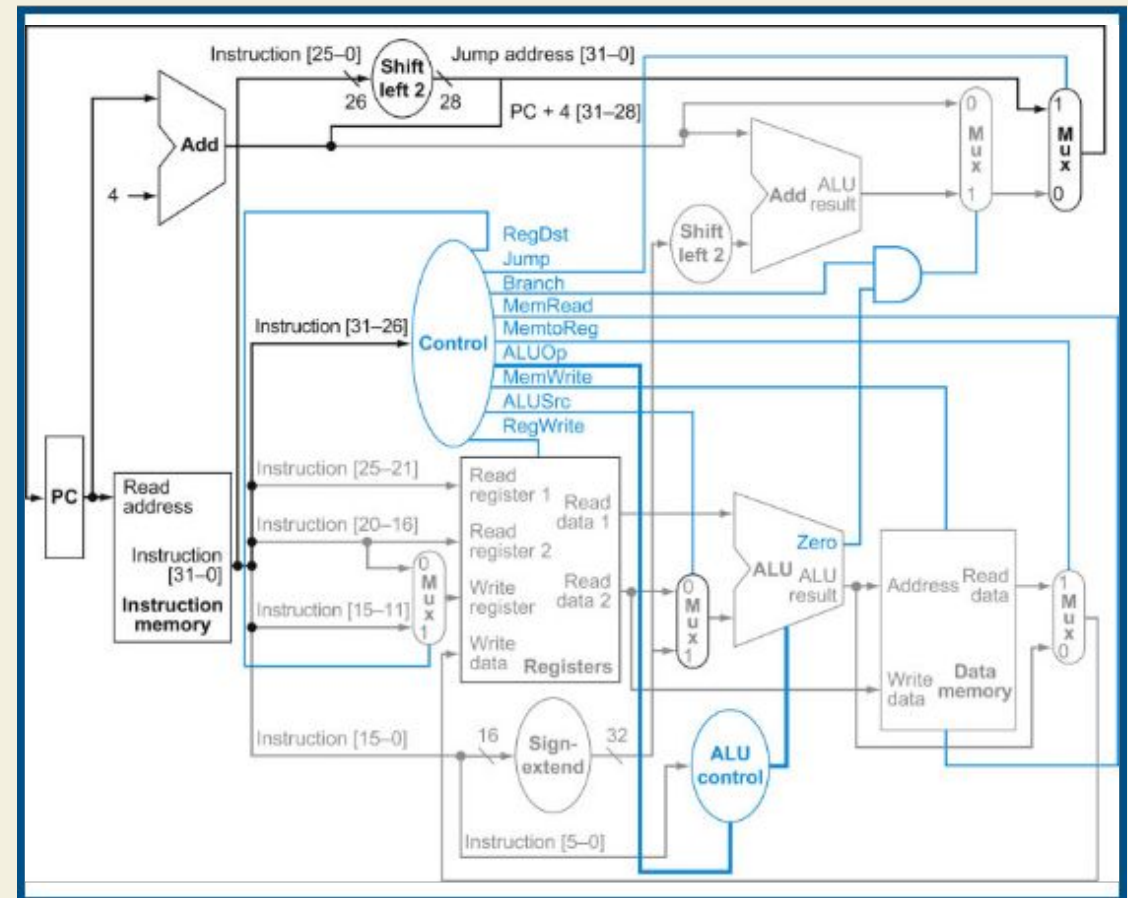
Validar e sintetizar o sistema na placa de desenvolvimento FPGA Altera DE2 (Cyclone II).

Visão Geral da ISA

Tipo	Instruções	Descrição
Aritmética / Lógica	ADD, SUB, AND, OR, SLT	Operações fundamentais entre registradores (R-Type).
Imediato	ADDI, ANDI	Operações com constantes de 16 bits.
Acesso à Memória	LW, SW	Arquitetura Load/Store. Acesso mapeado para I/O.
Controle de Fluxo	BEQ, BNE, J, JAL, JR	Desvios condicionais e incondicionais.

O Caminho de Dados (Datapath)

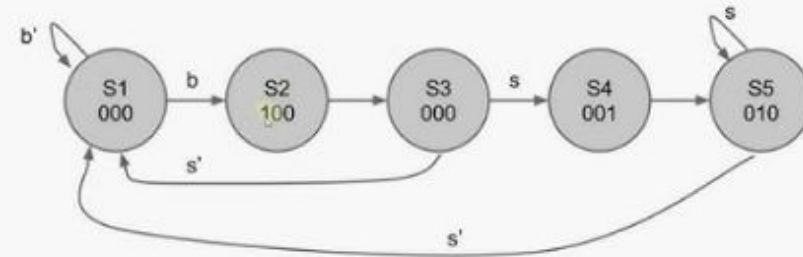
- **Multiciclo:** Execução dividida em 5 estágios (Fetch, Decode, Exec, Mem, WriteBack).
- **Recursos Compartilhados:** Uma única ALU e uma única memória para instruções e dados (Von Neumann).
- **Banco de Registradores:** 16 registradores de 32 bits, com R0 fixo em zero e R15 usado para Link (JAL).



Unidade de Controle (FSM)

- **Máquina de Estados Finita:** Controla o fluxo de dados em cada ciclo de clock.
- **Decodificação:** O estado inicial (Fetch) transita para Decode, onde o Opcode define o próximo passo.
- **Flexibilidade:** Permite ciclos de execução variáveis (ex: Jump leva 3 ciclos, LW leva 5 ciclos).

FSM Diagram Outputs = did21



Estratégia de Implementação



Modularização

O sistema foi dividido em três módulos principais: **Processor.v** (Core), **TopLevel.v** (Interface Física) e Memória RAM interna.



Memory Mapped I/O

Periféricos mapeados como endereços de memória. Endereço **60** lê as Chaves (SW) e endereço **61** escreve nos LEDs.

Desafio de Engenharia #1

O Bug da Leitura (LW)

O Problema

O processador executava a instrução LW, mas o registrador destino recebia sempre zero ou lixo, falhando na leitura das chaves.

A Solução

Identificamos que o dado vindo da memória não estava sendo capturado. Adicionamos a captura explícita no estado MEM:

```
if (state == MEM) MDR <= mem_rdata;
```

Desafio de Engenharia #2

O Bug do Endereçamento

O Problema

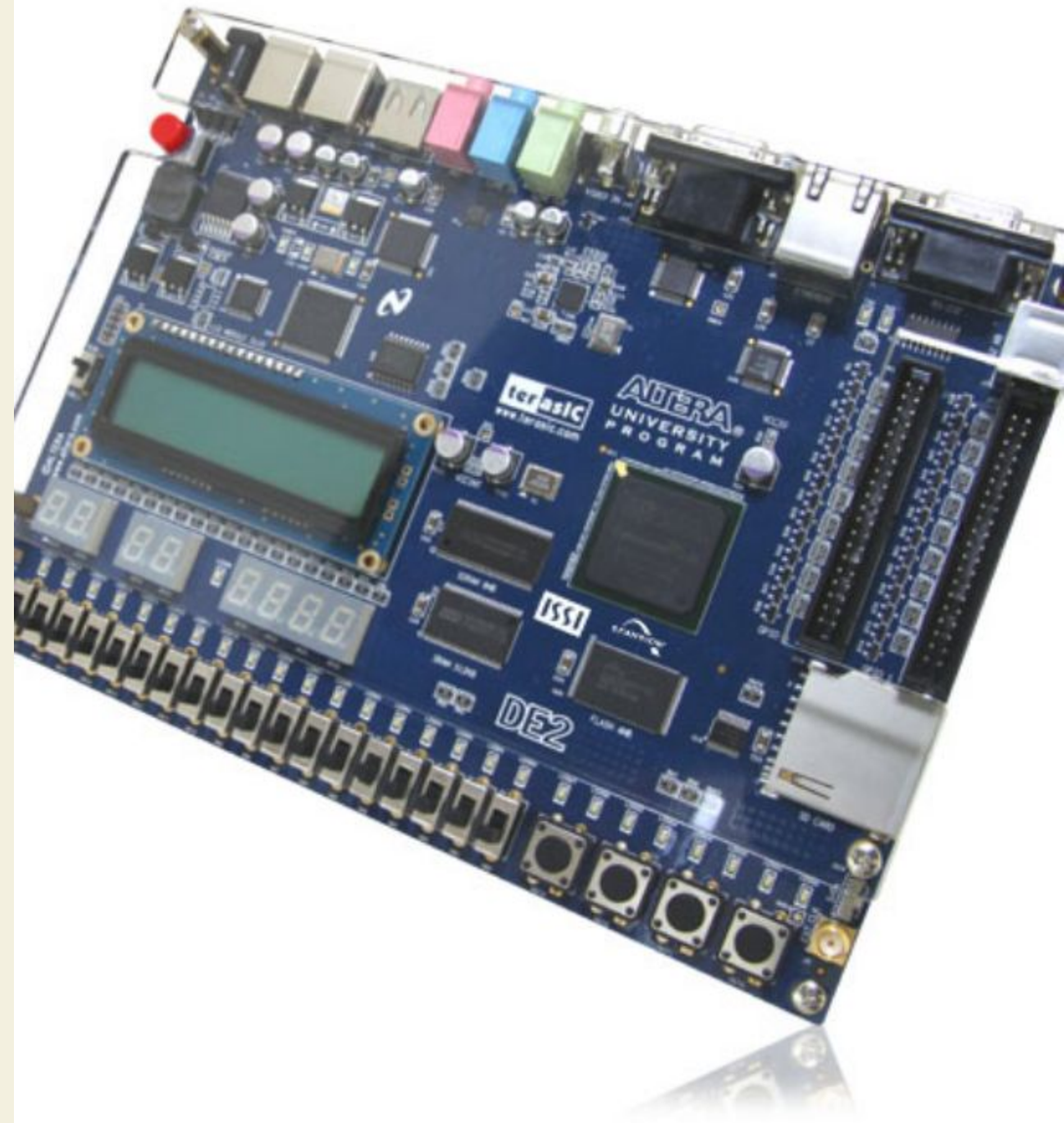
As instruções de Load e Store pulavam o cálculo de endereço na ALU, acessando posições incorretas de memória.

A Solução

Corrigimos a Máquina de Estados para forçar a transição para o estado **EXEC** antes de MEM, garantindo que $ALUResult = A + Imm$ seja calculado.

Síntese Física na DE2

- **Clock Divider:** Implementamos um divisor de frequência para reduzir o clock de 50MHz para ~3Hz, permitindo a visualização passo a passo.
- **Pin Planner:** Mapeamento manual dos pinos da FPGA Cyclone II para Switches (Input) e LEDs (Output).
- **Gravação:** Utilização da interface JTAG (USB-Blaster) e memória Flash (EPCS16) para persistência.



Resultados Finais



Visualização do PC

Os LEDs 0-7 piscam em contagem binária, indicando o fetch de instruções e o funcionamento do clock.



Processamento

O sistema lê o valor binário das chaves, soma 1 (usando a ALU) e exibe o resultado correto nos LEDs 8-15.



Integração Total

Sucesso na integração entre Software (Assembly), Arquitetura (Verilog) e Hardware (FPGA).

Conclusão

O projeto demonstrou a complexidade e a satisfação de construir um computador "do zero". Superamos desafios críticos de timing e lógica sequencial, resultando em um processador funcional e robusto operando em hardware real.

Dúvidas?

Obrigado pela atenção.

Link do Projeto:

<https://github.com/Jacsonrn/Processador-RISC-32-bits-Multiciclo-.git>