Curso de Ciência da computação 2024.1,
Matéria de Estrutura de Dados Básicas 1
Natal, Rio Grande do Norte

Technical Report: Processadores

Data de Submissão:

13/10/2024

Autores:

ARTHUR JOSÉ ARRUDA SKEETE,
CAROLINA NUNES DE CARVALHO
e
OSVALDO HEITOR DE ANDRADE TAVARES DE SOUZA

Docente:

MÁRCIO KREUTZ

Conteúdos

Sumário Executivo

Este documento tem como intenção apresentar os resultados, análises e desafios relacionados à modelagem de processadores didáticos inspirados na arquitetura de von Neumann.
Através da aplicação de algoritmos simples nos modelos implementados, é possível avaliar o desempenho e as limitações dessas arquiteturas com objetivo de complementar o aprendizado.

1. Introdução

Este relatório tem como temática a modelagem dos processadores Neander, Ramsés e César e a aplicação e a experimentação de diferentes algoritmos simples em cada.

1.1 Contexto

Durante o estudo de processadores, foi indicado a necessidade da implementação de modelos de computadores programáveis arcaicos - tudo com o objetivo de ampliar os conhecimentos discentes. Com base nisso, foram desenvolvidos, na linguagem de programação C + +, modelos de processadores hipotéticos Neander, Ramsés e César.

1.2 Objetivo

O relatório tem como objetivos a exposição dos programas desenvolvidos, que buscam modelar os processadores citados anteriormente. Dessa forma, encoraja-se o engajamento e a interação do leitor com o código desenvolvido, com o objetivo da expansão do conhecimento dos envolvidos.

2. Processadores

Os processadores implementados a seguir têm um nível crescente de complexidade, sendo Neander o mais simples e arcaico e o César o mais complexo e refinado. No entanto, todos eles possuem interseções para a caracterização básica de um processador - como a existência de uma Parte de Controle (PC) e Parte Operativa (PO).

Nesse sentido, antes de desenvolver cada um dos processadores, foi idealizado um diagrama de blocos apresentado em sua seção respectiva. Em seguida, cada bloco de cada diagrama apresentado foi transformado em uma classe equivalente no C + +, munida de métodos característicos a cada processador (métodos). Dessa forma, sinais transmitidos pela PC - abstraídos como estados da classe “uni_cont” nessa tarefa - são usados para sinalizar a ocorrência da próxima operação;

A seguir, mostraremos, em suas respectivas seções, como foram desenvolvidos os processadores Neander, Ramsés e César. Assim, com o auxílio de materiais

2.1 Neander

O mais arcaico dos processadores implementados, o neander é caracterizado por sua simplicidade; Munida apenas de operações básicas, - como a adição e a negação - um único modo de endereçamento e apenas um registrador geral, a arquitetura é conhecida por ser meramente didática.

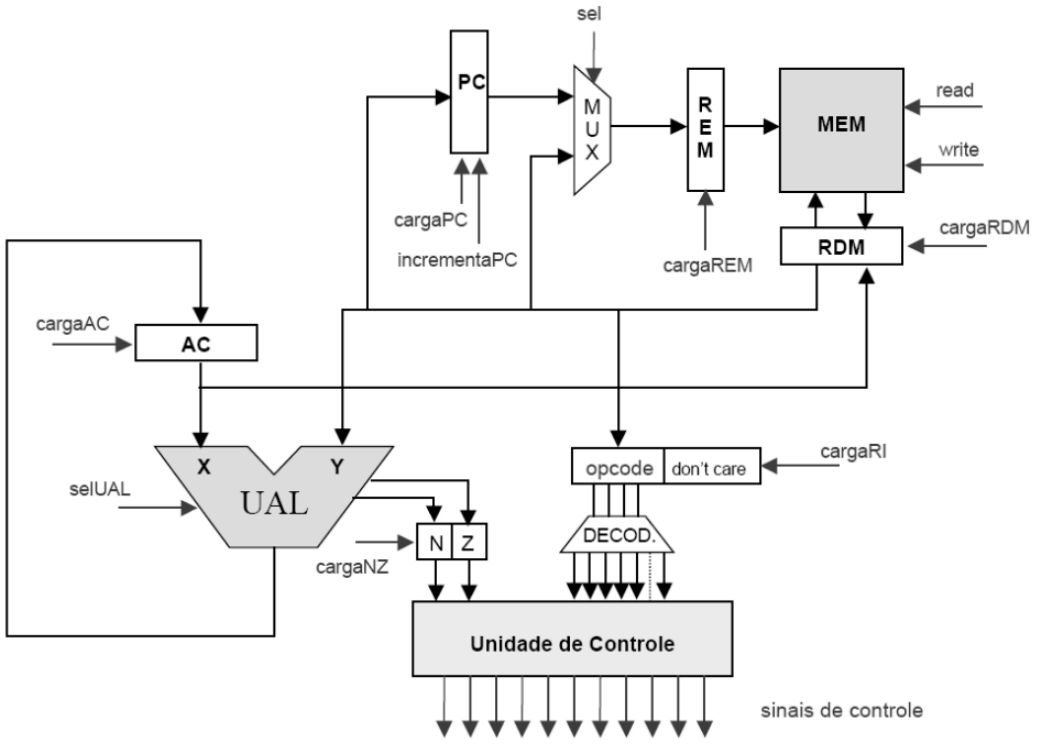

Figura 1. [Diagrama de blocos do processador Neander, acompanhado de sinais de controle equivalentes]

cargaAC: habilita carga no Acumulador
s: seleciona entradas do MUX
selUAL: seleciona operação a ser executada na UAL
cargaNZ: carrega ags N e Z
cargaRI: habilita carga no Registrador RI
cargaPC: habilita carga no Contador de Programas (PC)
incrementaPC: habilita o incremento do Contador de Programas (PC)
cargaREM: habilita carga no Registrador REM (Registrador de endereçamento de memória)
read: habilita leitura na memória • write: habilita escrita na memória
cargaRDM: habilita carga no RegistradorRDM (Registrador de Dados de Memória)

O Neander, por ser altamente simples, possui, em sua parte operativa, apenas um registrador de uso geral (AC), que recebe as informações. Ademais, possui outros componentes básicos, como uma memória de dados e instruções (MEM), um registrador encarregado da contagem

de programa (PC), um multiplexador (MUX), uma Unidade Lógico Aritmética (ULA), encarregada de realizar operações lógico-aritméticas, um Registrador de Endereço de Memória (REM) e um Registrador de Memória (RDM). Assim, os sinais de controle (simulados, no programa construído, por estados da unidade de controle da Unit_Cont) resultam no "acionamento" de suas respectivas microinstruções.

Dessa forma, uma classe foi implementada para cada bloco apresentado na Figura 1

Por fim, nota-se que o programa pode ser compilado, quando na pasta "src", com o comando:

g++ neander.cpp usefulfuncs.cpp -o neander.exe -g

E executado com o comando:
./neander.exe *path_to_file*
onde *path_to_file* representa o caminho até a file com o código assembly que deve ser lido.
ex:
./neander.exe ..\algoritmos\neander\subtracao.txt

2.2 Ramsés

O processador Ramsés representa um avanço técnico quando comparado ao Neander, em grande parte é parecido, mas possui algumas diferenças importantes, entre elas estão: 3 registradores de uso geral (RA, RB, RX); 4 modos diferentes de endereçamento (Direto, indireto, indexado e imediato); mais operações na ULA, facilitando a programação dos algoritmos; uma nova carga C para avisar quando uma operação ultrapassou o limite de 8 bits;

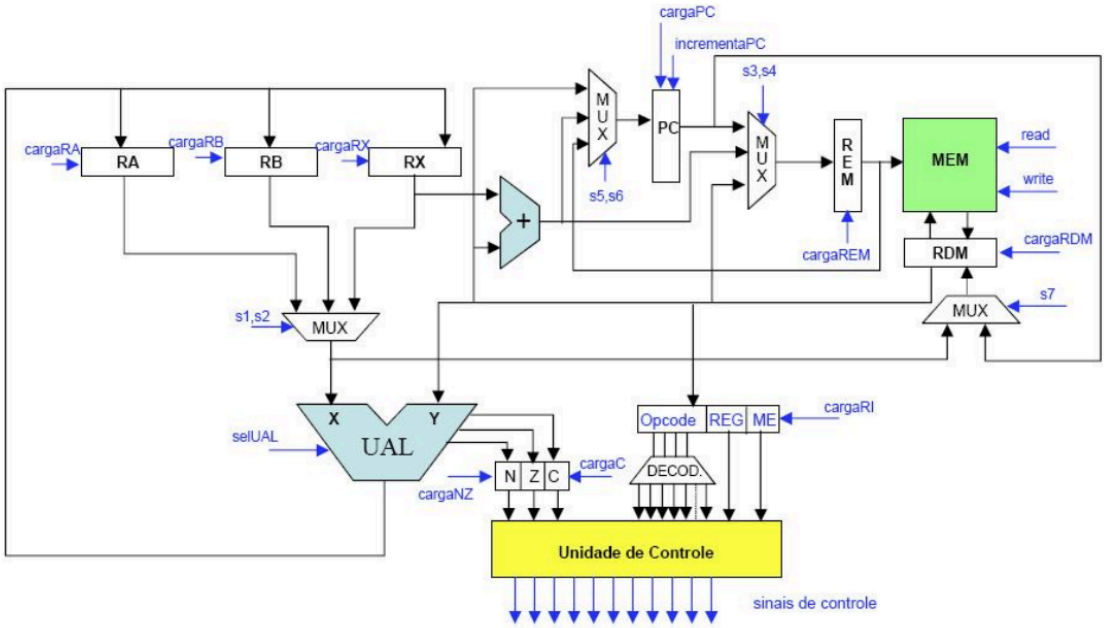

Figura 2. [Diagrama de blocos do processador Ramsés, acompanhado de sinais de controle equivalentes]

cargaRA: habilita carga no Registrador RA
cargaRB: habilita carga no Registrador RB
cargaRX: habilita carga no Registrador RX
s1, s2: seleciona entradas do MUX
selUAL: seleciona operação a ser executada na UAL
cargaNZ: carrega as flags N e Z
cargaC: carrega a flag C
cargaRI: habilita carga no Registrador RI
s5, s6: seleciona entradas do MUX
cargaPC: habilita carga no Contador de Programas (PC)
incrementaPC: habilita o incremento do Contador de Programas (PC)
s3, s4: seleciona entradas do MUX
cargaREM: habilita carga no Registrador REM (Registrador de endereçamento de memória)
read: habilita leitura na memória
write: habilita escrita na memória

cargaRDM: habilita carga no RegistradorRDM (Registrador de Dados de Memória)
s7: seleciona entradas do MUX
Por fim, nota-se que o programa pode ser compilado, quando na pasta “src”, com o comando:

g++ ramses.cpp usefulfuncs2.cpp -o ramses.exe -g

E executado com o comando:
./neander.exe *path_to_file*
onde *path_to_file* representa o caminho até a file com o código assembly que deve ser lido.
./ramses.exe ..\algoritmos\ramses\array.txt

2.3. César

O processador César é o mais completo entre os modelos didáticos, integrando conceitos de iteratividade e modularidade. Voltado para apresentar uma arquitetura mais próxima de processadores reais. Dessa forma, essa arquitetura possui 8 processadores de uso geral (R0-R7) e 8 modos de endereçamento (que incluem pré e pós-decremento). Ademais, verifica-se a existência da carga V (para o caso de overflow numérico) e um registrador auxiliar para a execução de instruções.
Além disso, ao aceitar operações com 2 operandos (no formato [dest <- dest * fonte]), a máquina otimiza o uso da memória e dos ciclos de relógio, resultando em um processo inteiramente mais rápido.

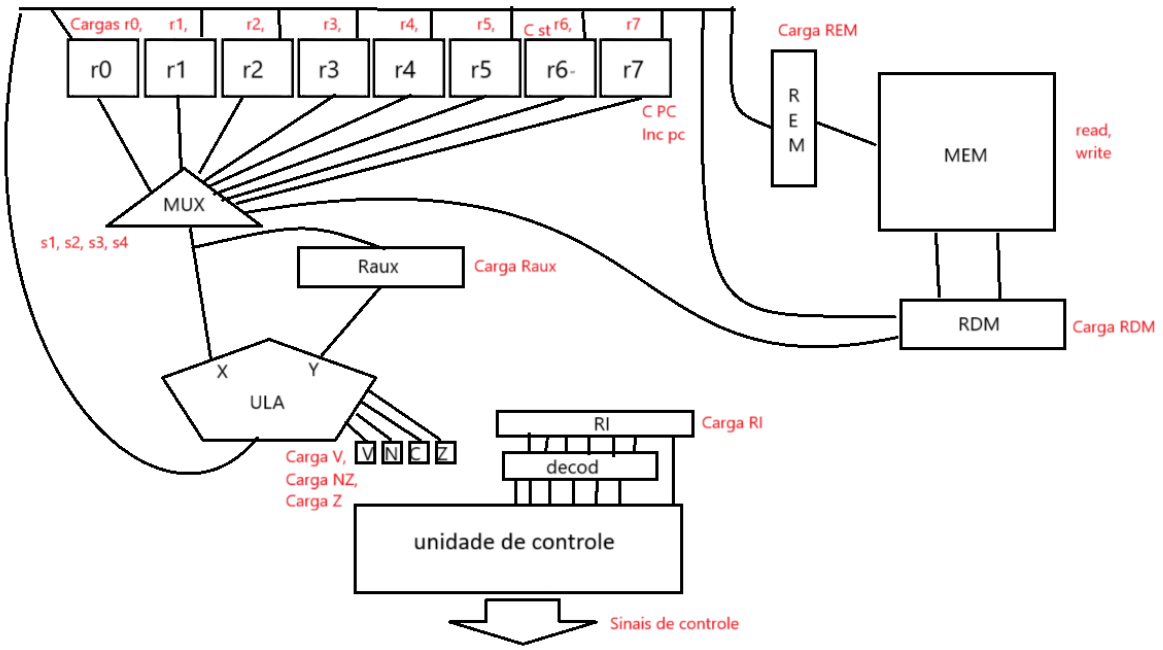

Figura 3. [Diagrama representando com blocos o processador César.]

cargaR0: habilita carga no Registrador R0
cargaR1: habilita carga no Registrador R1

cargaR2: habilita carga no Registrador R2
cargaR3: habilita carga no Registrador R3
cargaR4: habilita carga no Registrador R4
cargaR5: habilita carga no Registrador R5
cargaR6: habilita carga no Registrador R6
cargaR7: habilita carga no Registrador R7
cargaRaux: habilita carga no Registrador auxiliar
cargaPC (C PC): habilita carga no Contador de Programas (R7)
incrementaPC (Inc PC): habilita o incremento do Contador de Programas (R7)
CargaStackPointer (C st): habilita carga no apontador de stack (R6)
s1, s2, s3, s4: seleciona entradas do MUX
selUAL: seleciona operação a ser executada na UAL
cargaNZ: carrega as flags N e Z
cargaC: carrega a flag C
cargaC: carrega a flag V
cargaRI: habilita carga no Registrador RI
cargaREM: habilita carga no Registrador REM (Registrador de endereçamento de memória)
read: habilita leitura na memória
write: habilita escrita na memória
cargaRDM: habilita carga no RegistradorRDM (Registrador de Dados de Memória)

3. Resultados

3.1. Clock

Ao final da operação de subtração, os clocks marcaram:
Neander – 47
Ramsés – 33
César — ??

Os resultados encontrados indicam a baixa performance em operações avançadas do processador neander em comparação aos demais. Isso ocorre, claramente, devido à ausência de operações como a subtração na ULA do processador em questão - sendo, então substituída pelas operações “not” e “add”.
Dessa forma, tais mudanças vão repercutir diretamente no número final do clock, visto que a operação adicional resulta em mais buscas pela memória e mais uma operação lógico-aritmética.

3.2 Complexidade da programação

Como exemplificado no vídeo, o Neander, por possuir um conjunto de instruções mais básico, apresenta uma programação mais difícil, precisando implementar algoritmos para suprir a falta de instruções importantes. Em contrapartida, o Ramses facilita a programação, ao ver que é dá mais opções ao programador com seus diferentes modos de endereçamento, mais instruções, mais registradores e sub-rotinas.

4. Dificuldades encontradas

Ao longo do desenvolvimento do projeto, dificuldades foram indicadas.

4.1. Programação não algorítmica

Por serem acostumados a programar algoritmicamente, os integrantes do grupo apresentaram grandes dificuldades em captar a ideia e em desenvolver um programa “não algorítmico” ou “programável para um leitor externo”. Tal adversidade resultou na reestruturação dos processadores (sobretudo o Neander) múltiplas vezes.

4.1. Não conclusão do Processador César

Devido principalmente à perda de tempo com as dificuldades citadas na seção anterior (na reorganização do processador neander) e à agenda frenética do curso de Bacharelado em Ciência da Computação, evidenciada pela greve (e parcialmente devido à incompetência dos membros), o processador César não é executável; Dessa maneira, classes e métodos foram

criados para modelar o computador, mas o código não é completo ao ponto de simular corretamente tal arquitetura

5. Ilustrações

Figura 1. [Diagrama de blocos do processador Neander, acompanhado de sinais de controle equivalentes]

Figura 2. [Diagrama de blocos do processador Ramsés, acompanhado de sinais de controle equivalentes]

Figura 3. [Diagrama representando com blocos o processador César.]