Arquitetura de Computadores

Universidade Federal da Paraíba — UFPB

Centro de Informática — CI

Lista de Exercícios Avaliativa

Discente: Gustavo José Leite Cordeiro

Matrícula do Discente: 20220060522

Docente: Rômulo Calado Pantaleão Câmara

Exercícios

Exercícios sobre Arquitetura ARM

1 FAÇA UM ESTUDO DA ARQUITETURA ARM DESTACANDO OS SEGUINTES ASPECTOS:

a. Tamanho do dado a ser processado.
   A arquitetura ARM é baseada em registradores de 32 bits. Isso significa que a maioria das operações aritméticas e lógicas é realizada em dados de 32 bits. No entanto, a arquitetura ARMv8-A introduziu suporte para registros de 64 bits, permitindo operações em dados de 64 bits.

b. Espaço de endereçamento de memória.
   A arquitetura ARM possui um espaço de endereçamento de memória de 32 bits, o que significa que ela pode acessar até 4 GB de memória diretamente. Com a introdução do ARMv8-A, o espaço de endereçamento de memória foi expandido para 64 bits, permitindo acessar uma quantidade ainda maior de memória.

c. Número de registradores.
   A arquitetura ARM possui um conjunto de 16 registradores de uso geral (R0-R15), cada um deles com 32 bits de largura. Além disso, existem registradores especiais, como o Program Counter (PC), o registrador de status (CPSR ou NZCV no ARMv8-A) e os registradores de controle de interrupção.

d. Formato e tamanho das instruções.

As instruções na arquitetura ARM têm tamanhos variados, dependendo do tipo de instrução. As instruções podem ter 32 ou até mesmo 16 bits de largura (Thumb). O formato das instruções também varia, mas geralmente segue um padrão de opcode seguido por operandos e modificadores. O ARMv8-A introduziu o conceito de instruções de comprimento variável, permitindo que algumas instruções sejam codificadas em 16 bits, enquanto outras podem precisar de 32 bits ou mais.

Segue um exemplo da estrutura das instruções MUL e MLA em ARM:

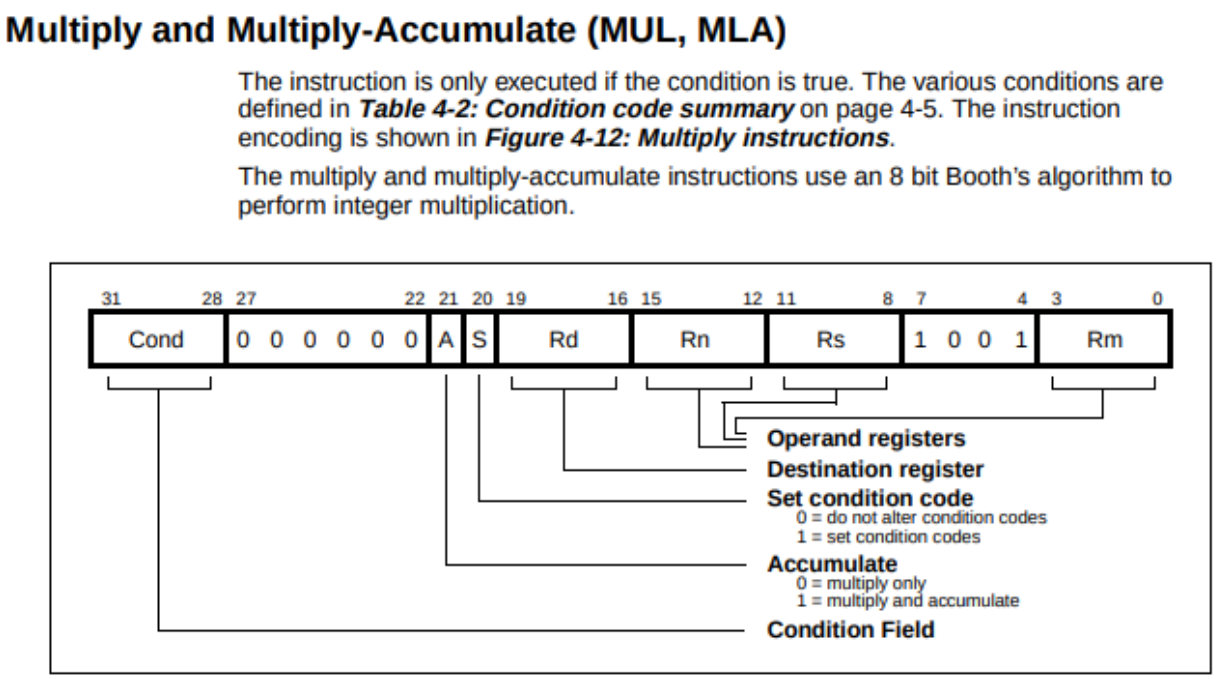

Figura 1 – Fonte: https://iitd-plos.github.io/col718/ref/arm-instructionset.pdf

2 TANTO NA ARQUITETURA PROJETADA EM SALA DE AULA COMO NA DESCRITA NO LIVRO TODAS AS INSTRUÇÕES ARITMÉTICAS ENVOLVEM TRÊS OPERANDOS (REGISTRADORES) E POSSUEM O MESMO FORMATO. NA SUA OPINIÃO DE PROJETISTA, QUAIS AS VANTAGENS E DESVANTAGENS DESTA DECISÃO DE PROJETO?

A escolha de uma arquitetura com instruções aritméticas de três operandos e formato único oferece vantagens em termos de simplicidade, flexibilidade e eficiência de código. No entanto, é importante considerar as desvantagens em relação ao desempenho, otimização e consumo de energia, especialmente em cenários específicos, em que, uma única instrução pode realizar o que seria necessário em duas instruções separadas em arquiteturas com três operandos. A decisão final deve ser tomada com base em uma análise criteriosa das necessidades e restrições do projeto.

3 PORQUE O PROCESSADOR ARM SE CARACTERIZA COMO UMA ARQUITETURA LOAD/STORE? QUAIS AS VANTAGENS E DESVANTAGENS DESTA CARACTERÍSTICA?

Arquiteturas Load-Store se baseiam em instruções que somente processarão (soma, subtração, etc) valores que estiverem nos registradores e sempre armazenarão os resultados em algum registrador. O processador ARM se caracteriza como uma arquitetura LOAD/STORE devido à sua abordagem de acesso à memória. Nesse tipo de arquitetura, as únicas instruções que acessam a memória são LOAD (carregar) e STORE (armazenar). Todas as operações aritméticas e lógicas são realizadas apenas entre registradores.

4 O QUE É MODO DE ENDEREÇAMENTO E QUAIS OS MODOS DE ENDEREÇAMENTO DAS INSTRUÇÕES DO ARM?

- Registrador-Direto: Os operandos estão em registradores especificados diretamente na instrução.
- Imediato: Um valor imediato é especificado diretamente na instrução.
- Registrador-Indireto: O operando é um endereço contido em um registrador.
- Base-Deslocamento: O endereço do operando é calculado somando-se um registrador base com um deslocamento especificado na instrução.
- Base-Indexado: Similar ao modo de base-deslocamento, mas o deslocamento é um valor contido em outro registrador.
- Base-Indexado e Escalonado: Similar ao modo de base-indexado, mas o valor do registrador de índice é escalonado antes de ser adicionado ao registrador base.

5 QUAL O SUPORTE DO ARM PARA FUNÇÕES? EXISTE SUPORTE PARA RECURSIVIDADE? COMO?

O ARM oferece suporte completo para funções, incluindo chamadas de função e retorno de valores. Para implementar a recursividade em ARM, você pode seguir a abordagem padrão de chamadas de função recursivas. Ao chamar uma função recursiva, é importante salvar o estado atual (como registradores importantes) na pilha para garantir que as chamadas recursivas não sobrescrevam dados importantes. Após a chamada recursiva, o estado salvo deve ser restaurado antes de retornar. Isso geralmente é feito usando o registrador de link (LR) para salvar o endereço de retorno e manipulando a pilha adequadamente.

6 CONSIDERE A SEGUINTE PARTE DE PROGRAMA EM LINGUAGEM ASSEMBLY DO ARM:

Listing 1 – Código Assembly do ARM

```
.data 0x10010000 # segmento de dados
palavra1: .word 13
palavra2: .word 0x15
```

Indique, em hexadecimal, quais os valores dos seguintes itens:

Palavra1: **0xD**

Palavra2: **0x15**

6.1 Faça um programa assembly que imprima “hello world” na saída do simulador ARM.

Listing 2 – Código Assembly do ARM - "Hello World"

```
.section .data
    message:
        .asciz "Hello, World!\n"
        message_len = .-message

.section .text
.global _start

_start:
    MOV r0, #1
    LDR r1, =message
    LDR r2, =message_len
    MOV r7, #4
    SVC 0

exit:
    MOV r7, #1
    MOV r0, #0
    SVC 0
```

7 PRETENDE-SE CODIFICAR UM PROCEDIMENTO SUBSTITUI(STRING,X,Y), EM ASSEMBLY DO ARM, QUE DADA UMA STRING E DOIS CARACTERES, X E Y, SUBSTITUI NESSA STRING TODAS AS OCORRÊNCIAS DO CARACTER X PELO CARACTER Y. O PROCEDIMENTO TERÁ COMO ARGUMENTOS (1,5): • STRING: O ENDEREÇO DA STRING • X: O CARACTERE A PROCURAR • Y: O CARACTERE PARA A SUBSTITUIÇÃO POR EXEMPLO, PARA A STRING “SOBSTITOI”, SE O CARACTER A PROCURAR FOR O CARACTER ‘O’, E O CARACTER PARA A SUBSTITUIÇÃO FOR O CARACTER ‘U’, A STRING DEVE SER ALTERADA PARA “SUBSTITUI”.

7.1 a. Desenhe um fluxograma para este procedimento.

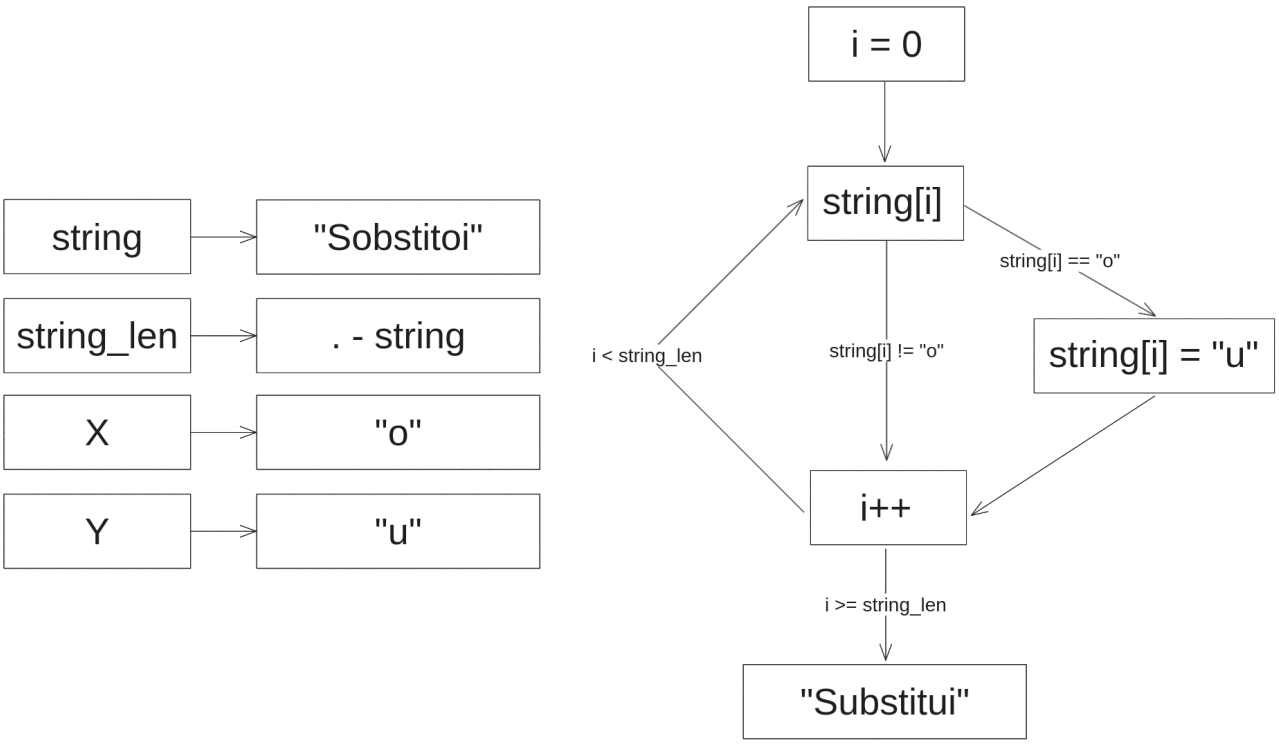

Figura 2 – Fluxograma do programa