Fabrício Mady **1215080268**

Arquitetura ARM

Características gerais da Arquitetura ARM

Processadores ARM possuem 37 registradores, dos quais 30 são de propósito geral. Cada modo de operação do processador utiliza diferentes registradores, fazendo com que apenas 15, dos 30 registradores gerais, estejam visíveis em cada modo. Além disso, dois deles são reservados para ponteiro de pilha e link register, que armazena endereços de retorno para quando ocorrem chamadas de função. Os 7 registradores de uso específico incluem contador de programa e registradores de estado, que são especialmente importantes na arquitetura ARM, devido a execução condicional de instruções.

A maioria das instruções apresenta um campo de 4 bits, em que é definida uma condição para que a operação seja executada, ou seja substituída por uma instrução NOP (No Operation), caso a condição não seja atendida. Há 4 flags de estado, N,Z,C e V (Negativo, Zero, Carry e Overflow), que são armazenadas no registrador de estado. Os valores assumidos por essas flags definem se as condições das operações foram atendidas. Há, nas instruções de processamento de dados, um bit, chamado bit S, cujo valor define se o resultado da operação altera os valores das flags. Essa abordagem influencia no pipeline, mas apresenta a vantagem de diminuir a quantidade instruções explícitas de desvio condicional, e contribui para desempenho e eficiência energética, uma vez que a instrução de desvio condicional não faz processamento útil de fato, apenas altera o fluxo do programa.

Os modos de execução do processador são 7. O primeiro deles é o modo usuário, que não oferece nenhum tipo de privilégio ao programa em execução. Os outros modos oferecem diferentes opções de privilégios e são usados pelo Sistema Operacional. Oferecer tantos modos de operação permite ao Sistema Operacional fácil adaptação a diferentes cenários e tipos de aplicações. Alguns registradores são de uso exclusivo de modos de execução, o que promove trocas de contexto mais rápidas, uma vez que os valores desses registradores não têm que ser salvos e recuperados para programas que executam em outros modos.

O mapeamento de endereços da memória cache é feito de forma associativa em conjuntos. Até a família de processadores ARM10, a memória cache usava endereços lógicos (cache lógica), mas hoje usa-se cache física. Um aspecto particular da organização de cache dos processadores ARM é a existência de um buffer de escrita entre a cache e a memória principal. Esse buffer armazena dados que devem ser escritos na memória, enquanto as escritas ocorrem, de forma que, se ele estiver cheio, instruções Store, que não sejam de blocos, são suspensas até que haja espaço. O tamanho do buffer é reduzido a fim de evitar ou reduzir o prejuízo imposto pela não disponibilidade dos dados que aguardam escrita. A menos que haja muitas escritas, o uso do buffer promove uma melhora de desempenho.

A memória principal é organizada em superseções de 16MB, seções de 1MB, páginas grandes de 64KB e páginas pequenas de 4KB. A fim de acelerar buscas na memória, são mantidas, na memória principal, duas tabelas, L1 e L2, para superseções/seções e páginas grandes/pequenas, respectivamente. Além das divisões em seção e página, existe a divisão em domínios de memória, que consistem em coleções de páginas e seções.

Para acessar um domínio, é necessária permissão, e os processos podem ser clientes ou gerentes de um domínio. Ativar ou desativar o acesso a um domínio requer apenas a alteração de dois bits nos registradores de controle de acesso a domínio. A ideia de cliente e gerente de domínio permite o compartilhamento de domínios entre processos sem comprometer totalmente as restrições, que podem ser definidas pelo processo gerente.

São suportados 7 tipos de exceção para tratar interrupções. Cada tipo de exceção tem uma rotina específica, cujo endereço é fixo. O conjunto de endereços dessas rotinas é chamado vetor de exceções. Sempre que ocorre uma interrupção, o contador de programa recebe o endereço da rotina correspondente.

1-Conjunto de instruções (Básicas) e suas codificações

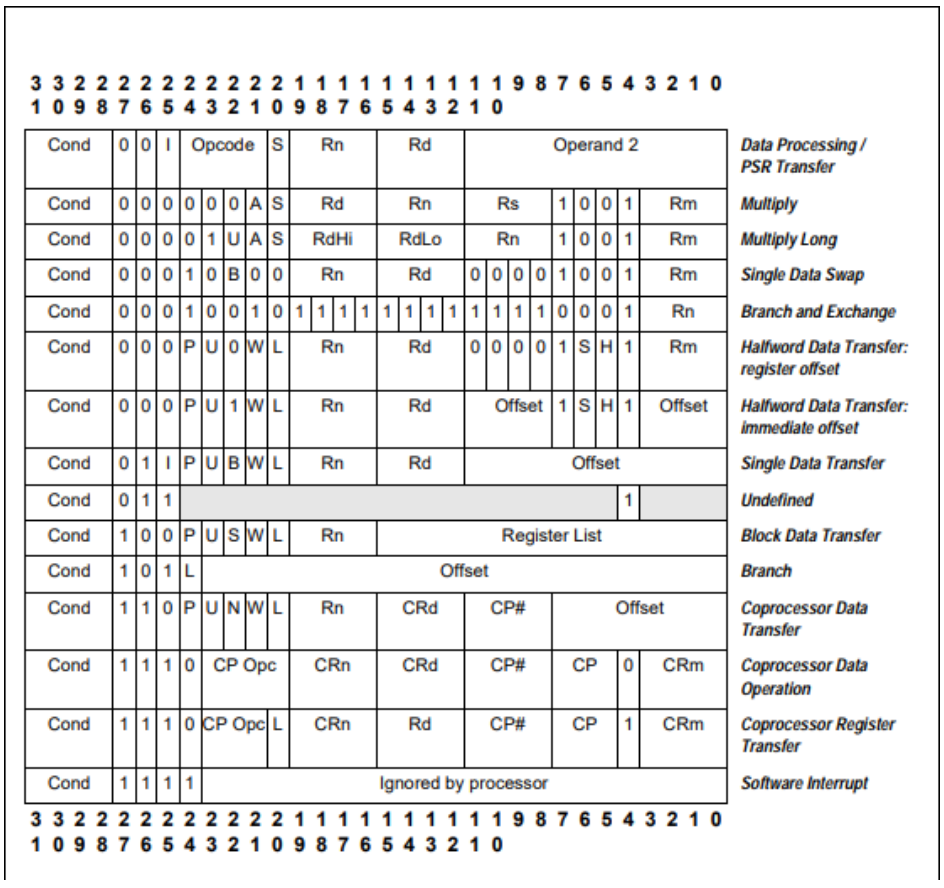
Load/Store: instruções de carregamento e escrita de dados na memória. Podem ler/escrever palavras inteiras (32 bits), meias-palavras (16 bits), bytes sem sinal, e podem ler e estender o sinal de meias-palavras.

Desvio: permitem desvio condicional de, no máximo 32MB, para frente ou para trás. A condição é dada por um campo de 4 bits.

Processamento de Dados: são operações de adição e subtração, operações lógicas AND, OR e XOR, e instruções de comparação e teste.

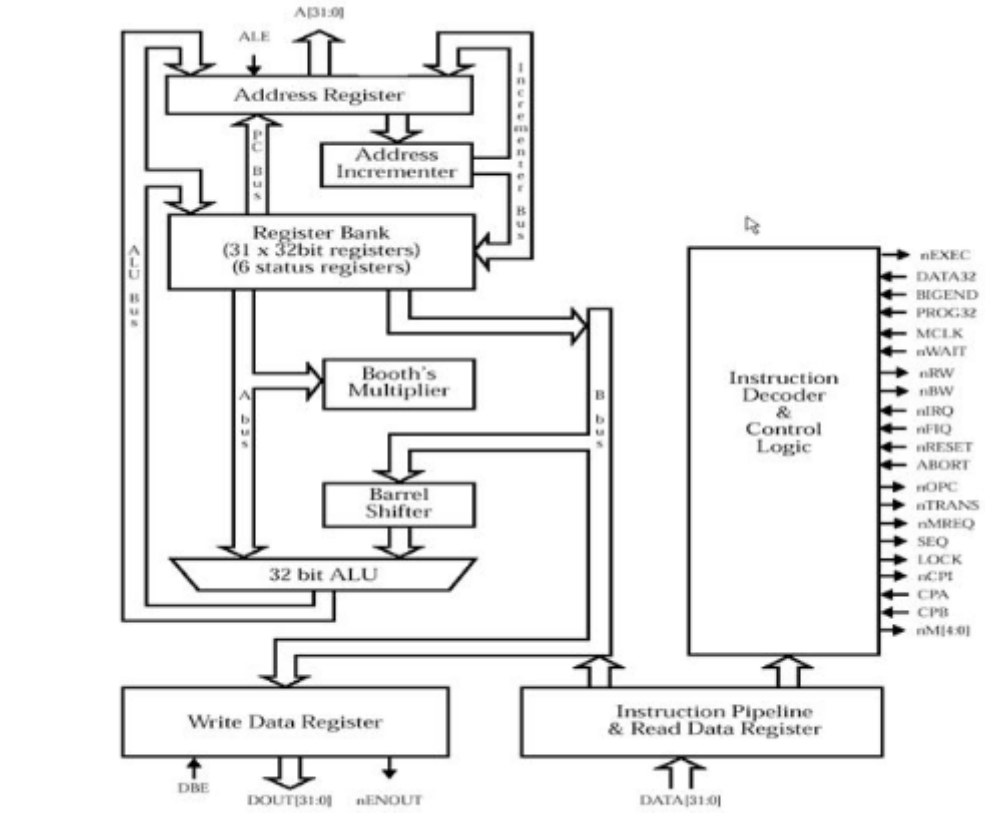
Multiplicação de inteiros: as multiplicações de inteiros podem ser feitas com operandos de 32 ou 16 bits, e o resultado pode ocupar 32 ou 64 bits.

Podemos observar a codificação dos conjuntos de instruções do ARM na figura abaixo:



2-Caminho de Dados e Controle e Pipeline ARM

Datapath:



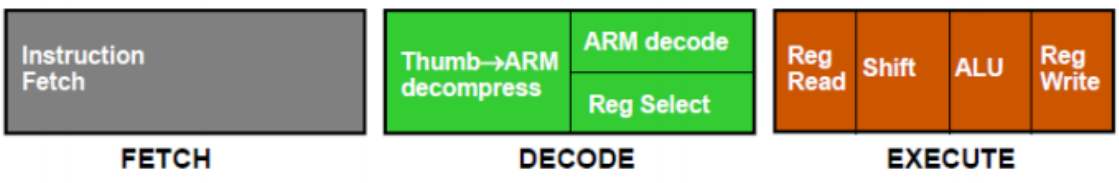
Pipeline:

A arquitetura ARM possui um pipeline de três estágios: Fetch, Decode e Execute

– **Fetch**: a instrução é trazida da memória e colocada no pipeline

– **Decode**: os registradores usados na instrução são decodificados

– **Execute**: o banco de registradores é lido, as operações lógico aritméticas são executadas sobre os operandos, o resultado da operação é gerado e escrito no registrador de destino



3-Principais diferenças ARM & MIPS, vantagens e desvantagens

-MIPS e ARM são duas arquiteturas com seus conjuntos de instruções diferenciados. Embora ambos os conjuntos de instruções tenham um tamanho de instrução fixo, o ARM possui apenas 16 registros enquanto o MIPS possui 32 registros.

-ARM tem maior rendimento e eficiência do que MIPS porque os processadores ARM suportam buses de dados de 64 bits entre o núcleo e os caches.

-Para permitir uma troca de contexto eficiente, a arquitetura MIPS suporta implementação de bancos múltiplos de registros. O ARM fornece apenas registros de uso geral para operações aritméticas e todas as outras funções, mas MIPS fornece dois registros separados para manter os resultados da operação de multiplicação.

-O MIPS não possui instrução equivalente à instrução ARM MOV.

-A instrução MIPS ADD normalmente gera exceção no overflow.

-Todas as instruções de processamento de dados ARM definem os códigos de condição ALU por padrão, mas MIPS fornece o SLT para comparação.

REFERÊNCIAS

<https://edisciplinas.usp.br/pluginfile.php/4480562/mod_resource/content/0/27aula%20-%20Arquiteturas%20ARM%20-%20POWER%20-%20XEON%20PHI.pdf> Acesso: 05/10/19

<https://slideplayer.com/slide/5199892/> Acesso: 05/10/19

<https://letsembed.wordpress.com/2013/12/15/arm-features_-code-data-path/>Acesso: 05/10/19

<http://embeddedreference.blogspot.com/2013/08/ARMCortexM0PlusLowEnergy.html> Acesso: 05/10/19

<https://www.dcce.ibilce.unesp.br/~aleardo/cursos/arqcomp/Semin_ARM.pdf>Acesso: 05/10/19

<http://home.ufam.edu.br/lucascordeiro/asd/slides/2_instrucoes_linguagem_do_computador.pdf> Acesso: 05/10/19