Relatório sobre o desenvolvimento da arquitetura do processador ARM7

Jonathan de Andrade Silva

[1]Universidade Católica Dom Bosco – UCDB,
Av. Tamandaré, 6000, Jardim Seminário,
Campo Grande, MS, Brasil, 79117–900

jsilva@acad.ucdb.br

1. Implementação

Neste trabalho foi implementada algumas instruções básicas para o desenvolvimento do processador arm versão 7. Essas implementações foram baseadas no datasheet disponível na página [1]. As instruções implementadas foram de três tipos, instruções de processamento de dados (*Data Processing*), desvio (*Branch*, *Exchange*), multiplicação (*Multiply* e *Multiply Long*) e acesso a memória (*Single Data Transfer*), esta última e as instruções de *Multiply Long* foram definida somente o formato. Na tabela 1, é apresentada as instruções implementada para tipo. Devido ao tempo de implementação, não foram implementadas as instruções dos tipos: *Swap* de dados, *Halfword Data Transfer*, *Undefined*, *Block Data Transfer*, *Software Interrupt*.

A implementação deste trabalho foi através da linguagem C, com a linguagem de programação Archc. Para a implementação do processador é necessário criar três arquivos. O primeiro arquivo definido da seguinte forma '(nome do arquivo).ac', por exemplo arm7.ac, é utilizado para poder definir as características da arquitetura do processador, por exemplo o número de registrador, tamanho da memória, o formato dos *pipelines*. O segundo arquivo com o seguinte formato, '(nome do arquivo)_isa.ac' é utilizado para definir os tipos das instruções e seus respectivos formatos. Por fim, o terceiro arquivo é criado seguindo a sintaxe '(nome do arquivo)−isa.cpp' contem a implementação do comportamento de cada instrução nos pipelines, como por exemplo a atualização do *program counter* após carregar uma instrução e a execução das operações para uma determinada instrução. Este processador possui três estágios, *Instruction Fetch* (IF), *Instruction Decode* (ID) e *Instruction Execute* (EX). Cada estágio do pipeline possui um formato para poder atribuir informações sobre os campos dos registradores que serão executados para manter a consistência de dados. No estágio IF a instrução é carregada, atualizado o *program counter* e verificada se a instrução poderá ser executada. No estágio seguinte (ID) a instrução é decodificada e em seguida no estágio EX a instrução é executada.

2. Sintaxe de execução

Para a execução das instruções neste processador é necessário antes que as instruções sejam codificadas em hexadecimal. O tamanho do registrador para as instruções é de 32 bits. Para instruções de desvio (*Branch*) é necessário inserir bolhas nos pipelines. A inserção de bolhas no pipeline é necessária para que a instrução seguinte a um desvio não

[1]http://www.gpec.ucdb.br/ricrs/Courses/AdvTopCompSys/ARM7-Microprocessor.pdf

Data Processing	Multiply	Multiply Long	Branch	Branch Exchange	Single Data Transfer
ADC	MUL	SMLAL	B	BLX	LDR
ADD	MLA	SMULL	BL		STR
AND		UMLAL			
BIC		UMULL			
CMN					
CMP					
EOR					
MOV					
MVN					
ORR					
RSB					
RSC					
SBC					
SUB					
TEQ					
TST					

Tabela 1. Tabela com as instruções definidas.

seja carrega no pipeline para a execução, pelo fato de não estar calculada ainda o endereço de desvio.

3. Bugs encontrados

Um problema encontrado é ao terminar a execução das instruções o *program counter* é incrementado e ao tentar buscar instruções naquela posição, que não existe, ocorre um aviso de que o *program counter* é inválido.

4. Dificuldades

Algumas dificuldades encontradas também foram para definir os formatos das instruções, que para o mesmo tipo dependo do tipo da operação, podendo ser com imediato e registrador, mudava o formato da instrução. Mas através da api do Archc, foi possível definir o mesmo formato para os dois tipos de operações através do uso do caractere '[' seguido dos campos para o primeiro tipo de operação, em seguida uma barra '|' e os campos para a segunda operação, por fim o caractere ']' para fechar os campos que podem alternar seu uso.

Outra dificuldade encontrada foi quando ao executar uma operção de **mul**, multiplicação, confundia-se com a operação de **and**, pois o valor do campo opcode eram semelhantes apesar de não ter o mesmo tamanho. Assim foi necessário definir primeiro as instruções de multiplicação e depois *Data processing*.

5. Conclusão

Concluímos que este trabalho teve uma grande contribuição para poder aprofundar os conhecimentos sobre o funcionamento da arquitetura de um processador com vários estágios e como são executadas as instruções.