UFCG CCT Departamento de Sistemas e Computação

Universidade Federal de
Campina Grande
Centro de Ciências e Tecnologia

DSC

Departamento de Sistemas
e Computação

Av. Aprígio Veloso, 882 – Bodocongó
Caixa Postal 10.106
58.109-970 – Campina Grande – PB – Brasil
Fone: 310-1119 — Fax: 310-1273
email: dsc@dsc.ufcg.edu.br

Relatório Técnico

Nº DSC/XXX/05

Processador ARM

Nádia Milena Barbosa

UFCG/CCT/DSC
nadia@dsc.ufcg.edu.br

15páginas

maio de 2005

Processador ARM

Nádia Milena Barbosa
E-mail: nadia@dsc.ufcg.edu.br

Departamento de Sistemas e Computação
Av. Aprígio Veloso, 882 — Bodocongó — Caixa Postal 10.106
CEP 58109-970 — Campina Grande — PB — Brasil
Fone: 310-1119 — Fax: 310-1273

Resumo

A ARM (Advanced Risc Machines Inc.) é uma empresa que oferece grande variedade de processadores baseados em uma arquitetura comum.

O processador ARM1020E, que será abordado em detalhes nesse trabalho, é uma arquitetura RISC (Reduced Instruction Set Computer) de 32 bits e foi desenvolvido para executar aplicacações embarcadas bastante complexas e sistemas operacionais sofisticados.

No diagrama de blocos deste processador existe um bloco chamado "Integer Core". É neste bloco que encontra-se o pipeline, suporte para MMU (Memory Management Unit) e a existência de caches de dados e de instruções. Outro bloco bastante importante para o escopo deste trabalho é o Prefetch Unit, que engloba a previsão de desvios e a busca antecipada de instruções.

Abstract

The ARM (Advanced Risc Machines Inc.) is an enterprise that offers great variety of microprocessors based in a common architecture.
The ARM1020E microprocessor, that will be saw detailed in this work, is an 32-bit RISC architeture (Reduced Instruction Set Computer) and was developed to execute complex embedded applications and sophisticated operation systems. In the block diagram of this microprocessor there is a block called "Integer Core". In that block there is the pipeline, support to MMU (Memory Management Unit) and the data cache existence and instructions. Another block very important for this work is the Prefetch Unit, that embodies the deviation prevision and antecipated search of instructions.

Palavras-chave: processor, ARM, mobile device

Sumário

1 Introdução **5**

2 Integer Core **6**
2.1 Unidade de Busca Antecipada 6
2.2 Predição de Desvios 7
2.3 Unidade de Inteiros 7
2.4 Unidade Load/Store 7

3 O Coprocessador **8**

4 O Pipeline **9**
4.1 Estágios do Pipeline 9

5 Registradores **9**

6 Unidade de Gerenciamento de Memória **10**

7 Conjunto de Instruções **11**

8 Facilidades para o SO **12**

9 Aplicações/Características **13**

10 Aspectos de Desempenho **13**

11 Conclusão **13**

Lista de Figuras

1 Diagrama de blocos do ARM1020E 6
2 Integer Core detalhado 8

1 Introdução

A ARM (Advanced Risc Machines Inc.), empresa fabricante do ARM1020E foi criada em 1990 por um consórcio de empresas, sendo elas a Apple Computer, Acorn e VLSI Technology. A ARM licencia o projeto dos microprocessadores para diversos fabricantes: Texas Instruments, VLSI Technology, Samsung Corporation, NEC Corporation, dentre outras. Hoje em dia é a líder mundial de microprocessadores embarcados, com cerca de 75A empresa oferece uma grande variedade de processadores baseados em uma arquitetura comum. Este trabalho, portanto, irá abordar com mais profundidade o processador ARM1020E da família ARM10E e baseado na arquitetura ARMv5TE.

O processador ARM1020E é uma arquitetura RISC (Reduced Instruction Set Computer) de 32 bits e foi desenvolvido para executar aplicacações embarcadas bastante complexas e sistemas operacionais sofisticados, como Linux, JavaOS, WindowsCE e NetBSD. Opera a uma frequência de 325MHz e possui alta performance, baixo consumo de energia, além de ter suporte a memória virtual. Ainda, é compatível com os conjuntos de instruções ARM e Thumb, que serão melhor explicados a seguir.

Dessa maneira, possui todas as principais características de uma típica arquitetura RISC:

- Um grande número de registradores (31) de propósito geral, sendo 16 visíveis ao usuário e os demais usados para o processamento de exceções
- É baseada em instruções load-store
- Modo de endereçamento simplificado
- Tamanho fixo de instruçõoes (32 bits)

Além disso, a arquitetura ARM ainda fornece as seguintes funcionalidades:

- Operações de load e store sobre multiplas instruções, a fim de melhorar a performance
- Execução condicional de todas instruções para maximizar performance

O diagrama de blocos deste processador pode ser visto na Figura 1. Como pode-se observar, existe um bloco chamado "Integer Core". É neste bloco que encontra-se o pipeline do processador e será melhor detalhado na seção 4. Além disso, é possível ver a existência de suporte para MMU (Memory Management Unit) - ver seção 7 e a existência de caches de dados e de instruções. Outro bloco bastante importante para o escopo deste trabalho é o Prefetch Unit, que engloba a previsão de desvios e a busca antecipada de instruções.

2 Integer Core

Essa unidade caracteriza-se por possuir múltiplas unidades de execução e, por isso, há a possibilidade de se ter várias instruções no mesmo estágio do pipeline, possibilitando assim a execução simultânea de algumas instruções. Como consequência, o pipeline consegue uma taxa máxima de uma instrução por ciclo.

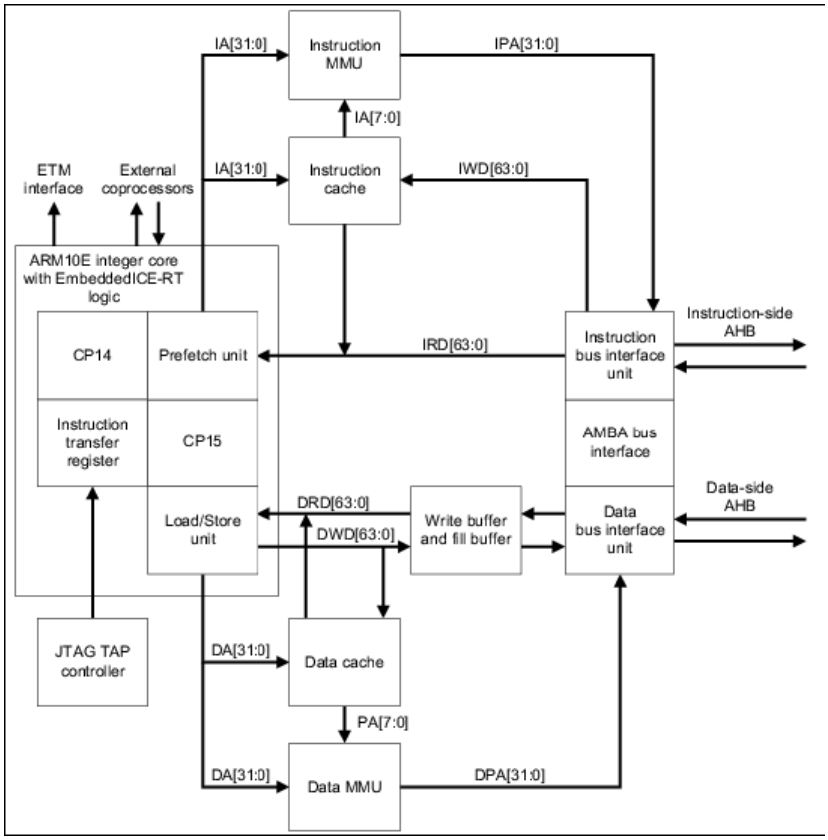


Figura 1: Diagrama de blocos do ARM1020E

2.1 Unidade de Busca Antecipada

A Unidade de Busca Antecipada atua no estágio de busca do pipeline (descrito na seção 4) e busca instruções da cache de instruções ou diretamente da memória. Além disso, para reduzir as perdas no pipeline, há também a predição de desvios, quando possível.

A UBA pode recuperar 64 bits a cada ciclo a partir da cache de instruções, sendo portanto, duas instruções. Além disso, pode enviar apenas uma instrução de 32 bits para a Unidade de Inteiros por ciclo. Pelo fato de ser capaz de buscar mais instruções do que pode repassar, as instruções pendentes são postas no Buffer de Busca Antecipada. Dessa maneira, quando uma instrução está no buffer, a lógica de predição de desvios pode decodificá-la e verificar se há a possibilidade de se prever o desvio. Além disso, a UBA deve ser esvaziada quando (i) uma exceção é gerada, (ii) o PC é atualizado, (iii) ocorre uma operação aritmética envolvendo o PC ou (iv) é detectado que um desvio foi erroneamente previsto.

Quando possível, a lógica de predição de desvios remove a instrução de desvio da fila de instruções a serem processadas. Por exemplo, se é previsto que o desvio deve ser tomado, então a próxima instrução a ser buscada será a instrução alvo do desvio. Entretanto, caso seja previsto que o desvio não deve ser tomado, então a próxima instrução a ser buscada é aquela imediatamente posterior a instrução de desvio, sendo que, mesmo assim, é feito o cálculo do endereço alvo para os casos em que a predição está errada. Nestes casos, se a instrução seguinte ao desvio já estiver na Unidade de Busca Antecipada, ela pode ser enviada para o próximo estágio do pipeline, de maneira que o desvio não leva nenhum ciclo.

A Unidade de Inteiros processa desvios não previstos ou desvios não previsíveis. Para obter os endereços rapidamente, ela usa um rápido e dedicado somador.

Existe, também, uma ligação da saída da ULA com a Unidade de Busca Antecipada. Isso é usado por instruções que escrevem no PC.

2.2 Predição de Desvios

O principal objetivo do mecanismo de predição de desvios é o aumento de performance, que é obtido através da redução do número de bolhas que devem ser inseridas no pipeline. A idéia da predição de desvios é antecipar a decisão sobre um salto 2 no programa, de maneira que o pipeline permance "cheio"durante todo o tempo.

Além disso, uma característica importante no ARM1020E é que as instruções de desvio não são executadas, pois já é determinado se o desvio será ou não tomado na Unidade de Busca Antecipada. Importante ressaltar que toda a predição de desvio feita pelo ARM1020E é para desvios em relação ao PC (Program Counter). A estratégia utilizada é a predição estática e, portanto, sem histórico, onde os desvios condicionais para frente são considerados como "não tomados"e os desvios condicionais para trás são considerados como "tomados".

No caso de ser previsto que um desvio será tomado e houver um cache hit do endereço alvo, a instrução do endereço correspondente é enviada diretamente para a Unidade de Busca Antecipada. Caso ocorra um cache miss, há uma espera até que o desvio seja confirmado.

Desvios mal previstos têm custo de três ciclos (com cache hit) extras, ou seja, considera-se que o estágio de Execução esteja parado.

2.3 Unidade de Inteiros

A unidade de inteiros decodifica as instruções enviadas pela unidade de busca antecipada. Ela contem ULA, multiplicadores e executa instruções do tipo MOV, ADD e MUL. Além disso, a unidade de inteiros auxilia a unidade Load/Store a executar loads e stores, bem como possui o "sequenciador de instruções"que lida com instruções de múltiplos ciclos.

2.4 Unidade Load/Store

A unidade Load/Store (LSU, em inglês) é a responsável por carregar da memória para registradores e enviar de registradores para a memória o conteúdo de variáveis. Essa unidade pode carregar ou armazenar dois registradores (64 bits) por ciclo. As instruções

de load/store simples (LDR, STR) não são beneficiadas com isso. Porém, as instruções de load/store múltiplas (LDM, STM) e as duplas (LDRD,STRD) têm vantagem devido a largura de banda disponível.

A LSU calcula o endereço a ser acessado através de um somador dedicado. Esse somador é avaliado em paralelo com o somador da ULA.

A LSU suporta a operação Hit-Under-Miss (HUM). Se um load resulta em um cache miss, esta solicitação é movida para o HUM. Dessa maneira, outras instruções, incluindo loads, podem ser executadas, desde que elas não sejam dependentes daquela instrução no HUM e desde que não aconteça outro cache miss. Portanto, com o uso do HUM aumenta-se o paralelismo e não há perda de ciclos com a ocorrência de um cache miss.

Estas unidades podem ser vistas na Figura 2 de maneira mais detalhada.

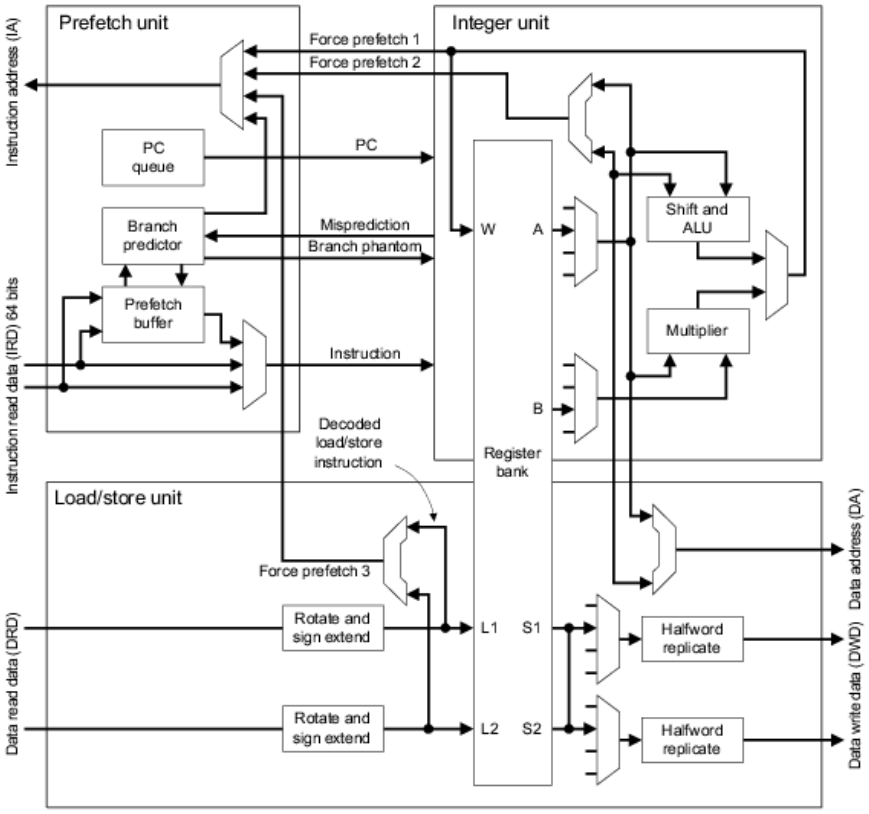


Figura 2: Integer Core detalhado

3 O Coprocessador

O coprocessador (CP), ou simplesmente CP15, é usado para controlar as MMUs, as caches e o write buffer. Além disso, fornece registradores adicionais (16, no total) para configurar e controlar os dispositivos acima citados e, também, o sistema de proteção e gerenciamento de energia (power down).

Além disso, o CP possui um pipeline e este roda um ciclo atrasado em relação ao pipeline do ARM1020E . Através desse mecanismo, garante-se que o estágio no ARM1020E

foi finalizado antes deste enviar dados ao CP.

4 O Pipeline

O pipeline do ARM1020E é composto por seis estágios. Através dele consegue-se uma taxa máxima de uma instrução por ciclo.

O estágio de busca usa um buffer do tipo FIFO que pode armazenar até três instruções. Os estágios de Decodificação 1 e 2 podem conter um desvio previsto em paralelo com uma outra instrução. Os estágios de Execução, Memória e Escrita podem conter simultaneamente:

- Um desvio previsto
- Uma instrução da ULA ou de multiplicação
- Múltiplas instruções de store

4.1 Estágios do Pipeline

A Figura 3(a) mostra os estágios do pipeline para uma operação típica.

A Figura 3(b) mostra os estágios do pipeline para uma operação usando a ULA. Já na Figura 3(c) é mostrada uma operação de load/store múltipla, onde o pipeline da LSU é utilizado para auxiliar na execução. Outras instruções podem usar o pipeline da ULA enquanto que a instrução de LDM/STM é finalizada no pipeline load/store.

E, na Figura 3(d), é mostrada uma operação de load onde há um cache miss.

5 Registradores

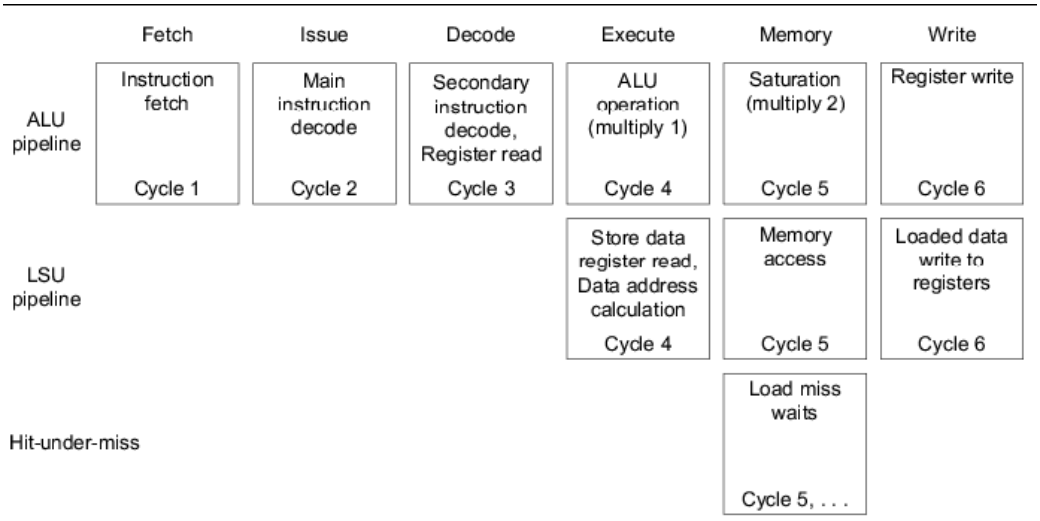
Como em qualquer processador RISC, os registradores do ARM1020E são de propósito geral. Entretanto, para algumas operações, alguns deles têm funções específicas e serão descritas a seguir.

O Contador de Programa (PC) ou R15, como já dito, é de propósito geral e pode ser livremente acessado. Permite boas possibilidades de desvios e aponta para a instrução que está duas posições a frente da que está sendo executada. Além disso, possui 32 bits, porém apenas 30 são usados, já que o endereçamento é por palavra.

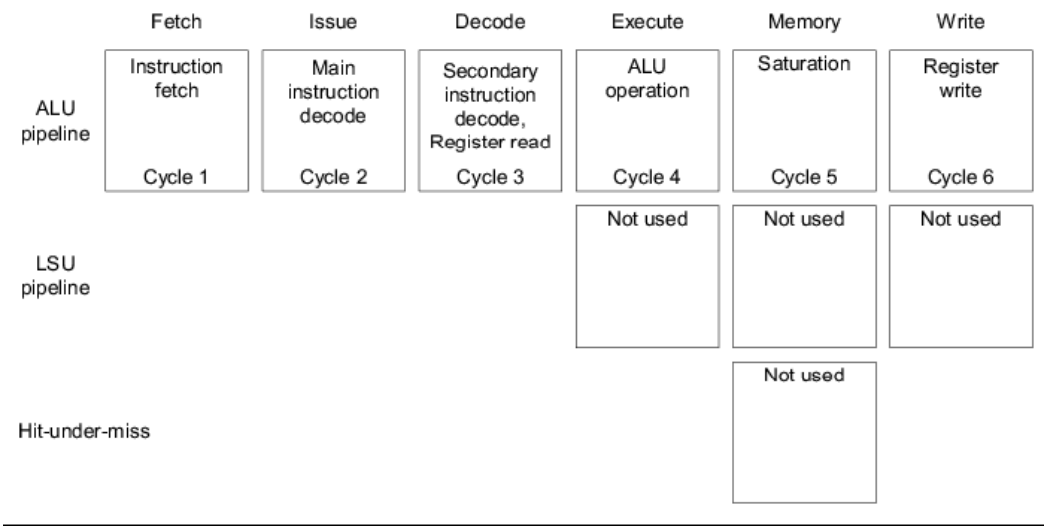
O Registrador de Link (LR) ou R14 armazena o endereço da instrução seguinte após chamada a instrução BL (Branch and Link) e é usado em chamadas de subrotina. Em qualquer outro momento, é de propósito geral.

O registrador CPSR (Current Program Status Register) armazena o estado atual do processador e está presente no coprocessador. Contem códigos de condição (negative, zero, carry e overflow), 2 bits para desabilitar interrupções e, ainda, 5 bits para descrever o modo do processador. Já o SPSR (Saved Program Status Register) preserva o conteúdo do CPSR durante a execução de uma exceção e também fica no coprocessador. Na figura 7 é mostrada a organização de tais registradores.

Todos os demais registradores têm sua função definida por software. Normalmente o R13 será usado como stack pointer.



(a) Estágios do pipeline para uma operação típica



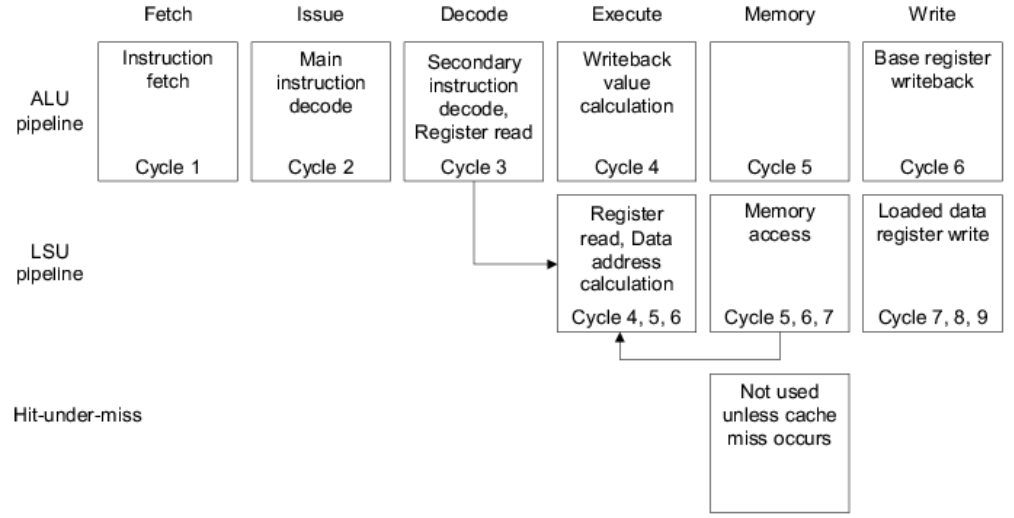
(b) Estágios do pipeline para uma operação usando a ULA

6 Unidade de Gerenciamento de Memória

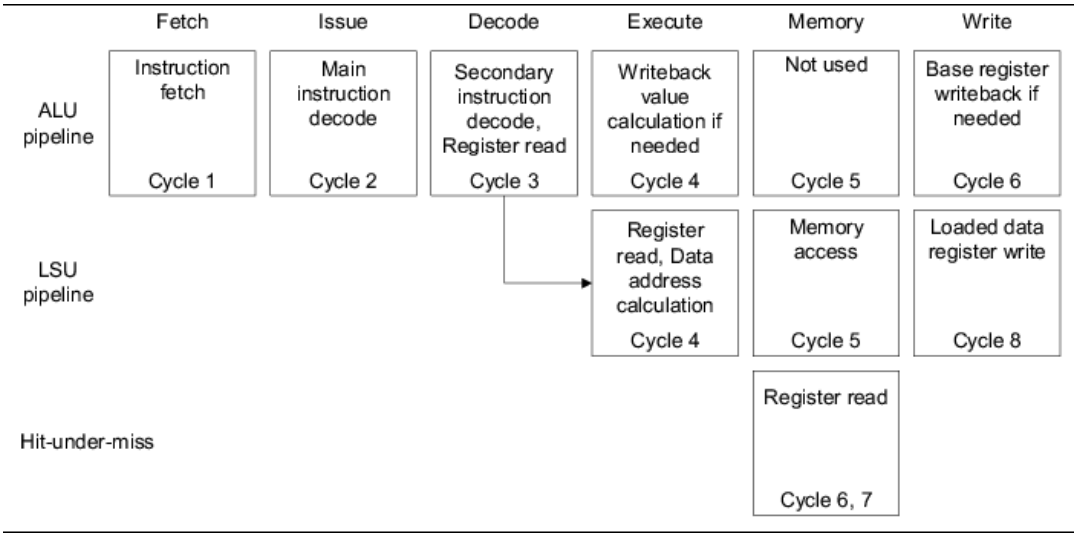
A Unidade de Gerenciamento de Memória MMU (*Memory Management Unit*) permite um alto controle do sistema de memória. Muitos dos controles detalhados são possíveis devido às *translation tables* existentes na memoria. Entradas nestas tabelas definem as propriedades de áreas de memória de vários tamanhos (1KB a 1MB). Estas propriedades incluem:

- mapeamento entre endereço físico e virtual
- permissões de acesso a memória
- bits de *cachability* e *bufferability*

O processo de olhar para toda a *translation table* é conhecido como *translation table walk* que é realizado automaticamente pelo *hardware* e tem custo de execução signifi-



(c) Estágios do pipeline para operação de load ou store múltiplos



(d) Estágios do pipeline para operação de load com cache miss

cante(pelo menos um acesso a memória principal e freqüentemente dois). Para reduzir o custo de acesso a memória, os resultados das *translation table walk* são *cached* em estruturas conhecidas como *Translation Lookaside Buffers* (TLBs).

7 Conjunto de Instruções

O processador ARM1020E possui suporte para dois conjuntos de instruções: (i) instruções ARM e (ii) instruções Thumb. O primeiro é o default. Engloba todas as instruções da arquitetura, sendo que cada uma possui 32 bits. Já o segundo conjunto de instruções é, na verdade, um subconjunto do primeiro e, portanto, não suporta todas as instruções da arquitetura. Sua principal vantagem é o aumento da densidade de código, já que cada instrução possui apenas 16 bits. O conjunto de instruções com o qual o processador está trabalhando é determinado pelo bit T do CPSR.

Neste texto o conjunto de instruções ARM será abordado com mais profundidade visto que é o padrão da arquitetura e, também, engloba todas as instruções do processador. Pode ser dividido em 4 tipos de instruções: desvios, operações com dados, instruções load/store e instruções do coprocessador.

Os desvios utilizam um offset de 24 bits, podendo ser positivo ou negativo. Duas instruções de desvio são consideradas especiais: (i) BL (Branch and Link) coloca o endereço atual em R14, pois, como já visto, é usada em chamadas de subrotina; (ii) SWI (Software Interrupt) faz uma chamada ao SO para realizar o desvio.

Operações com dados são as instruções lógicas e aritméticas. Tais instruções são num total de 16 e a maioria possui três registradores como operandos. Na instrução de multiplicação de dois registradores de 32 bits o resultado pode ser posto em um registrador de 32 bits (formato normal) ou em dois registradores de 32 bits (formato longo).

As instruções load/store podem ser classificadas em três tipos: (i) aquelas que operam sobre um registrador, (ii) aquelas que operam sobre múltiplos registradores e (iii) aquelas que fazem swapping, ou seja, trocam o conteúdo de um registrador com uma posição de memória. Pode-se ainda realizar operações apenas sobre word, halfword ou byte.

E, por último, as instruções ligadas ao coprocessador. Podem ser operações sobre dados, transferência de dados do coprocessador para registrador ou ainda transferência de dados do coprocessador para memória.

8 Facilidades para o SO

O ARM1020E apresenta alguns mecanismos que fornecem algumas facilidades para o Sistema Operacional. Neste texto serão destacados três deles: modos do processador, exceções e MMU.

O processador pode ser chaveado para operar em um dos diversos modos disponíveis. Cada modo oferece algum mecanismo de proteção e níveis hierárquicos, sendo que o único que não é considerado modo privilegiado é o USER. No modo USER a execução é normal e apresenta diversas restrições. No modo FIQ há a transferência de dados em alta velocidade. No IRQ ocorre o tratamento geral de interrupções. O modo SUPERVISOR é protegido e é o modo utilizado pelo SO. O modo ABORT implementa memória virtual, o UNDEFINED suporta emulação de software para coprocessador e o SYSTEM roda tarefas privilegiadas do SO.

O mecanismo de exceções é bastante útil para sistemas operacionais. No ARM1020E existem 5 tipos de exceções sendo que para cada um existe um modo privilegiado associado. Quando uma exceção é invocada, o registrador R14 armazena o endereço de retorno da exceção e, portanto, o endereço que a causou. Além disso, o PC é alterado para uma posição fixa de memória onde normalmente está o vetor de interrupções que possui uma posição para cada tipo de exceção. Importante ressaltar que os tratadores de exceção devem ser instalados pelo SO nessas posições fixas.

A MMU (Memory Management Unit) traduz endereços virtuais em físicos e, portanto, dá suporte a implementação de paginação no SO. No ARM1020E seu funcionamento é controlado por um registrador no coprocessador. Existe uma MMU e um TBL (Translation Lookaside Buffer) para instruções e outra MMU e TLB para dados. Os TLBs são as tabelas que armazenam a correspondência entre endereços físicos e virtuais e, no

ARM1020E , cada um possui 64 entradas.

9 Aplicações/Características

Existem vários ambientes/linguagens disponíveis para a programação sobre o ARM1020E . Alguns deles são o ARM C++, Java, Free-Pascal, Ada (com tradutor para C++) e Assembly.

Como já mencionado o ARM1020E possui suporte a vários SOs, sendo Linux, WindowsCE, JavaOS, NetBSD e EPOC- 32. Além disso, a última versão do ARMpossui suporte para execução nativa de bytecodes Java através da tecnologia Jazelle.

Um ponto bastante importante a ser mencionado é o uso extensivo de processadores ARM em aplicações embarcadas, tais como telefones celulares, videogames, sistemas de navegação veicular, impressoras laser, estações de trabalho de alta performance, dentre inúmeras outras.

10 Aspectos de Desempenho

Embora pouca pesquisa esteja disponível sobre este tema, o ARM1020E tem um desempenho de 400 MIPS (milhões de instruções por segundo). Além disso, quando utilizada uma unidade de ponto flutuante opcional (VFP10), ele fornece 650 MFLOPS (milhões de instruções de PF por segundo).

11 Conclusão

Através do que foi exposto aqui, pode-se concluir que o ARM1020E possui diversos mecanismos para maximizar a performance. Além disso, apresenta algumas estratégias para aumentar o paralelismo de certas operações. Por fim, pode-se constatar o seu bom desempenho na execução de aplicações embarcadas.

Referências

[ARM] ARM. Arm. Acessado em maio de 2005.

[Lau00] Luiz Sequeira Laurino. Um estudo sobre o arm1020e. 2000.

[SP00] A. SILBERSCHATZ and P.GALVIN. *Sistemas Operacionais: conceitos*. Prentice-Hall, 2000.