FELIPE GIUNTE YOSHIDA
MARIANA RAMOS FRANCO
VINICIUS TOSTA RIBEIRO

MICROKERNEL PARA PROCESSADOR ARM7

São Paulo
2009

FELIPE GIUNTE YOSHIDA
MARIANA RAMOS FRANCO
VINICIUS TOSTA RIBEIRO

MICROKERNEL PARA PROCESSADOR ARM7

Monografia apresentada à Escola Politécnica da Universidade de São Paulo para a Conclusão do Curso de Engenharia da Computação.

São Paulo
2009

FELIPE GIUNTE YOSHIDA
MARIANA RAMOS FRANCO
VINICIUS TOSTA RIBEIRO

MICROKERNEL PARA PROCESSADOR ARM7

Monografia apresentada à Escola Politécnica da Universidade de São Paulo para a Conclusão do Curso de Engenharia da Computação.

Orientador:
Prof. Dr. Jorge Kinoshita

São Paulo
2009

FICHA CATALOGRÁFICA

Yoshida, Felipe Giunte
MICROKERNEL PARA PROCESSADOR ARM7 / F.G. Yoshida, M.R. Franco, V.T. Ribeiro. – São Paulo, 2009. 25 p.

Trabalho de Formatura — Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia de Computação e Sistemas Digitais.

1. Microkernel. 2. ARM. I. Yoshida, Felipe Giunte II. Franco, Mariana Ramos III. Ribeiro, Vinicius Tosta IV. Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia de Computação e Sistemas Digitais. II. t.

DEDICATÓRIA

a ...

AGRADECIMENTOS

Agradeço a

RESUMO

Exemplo de modelo de teses e dissertações da poli utilizando LaTeX. O estilo foi baseado no modelo da ABNT e "adaptado" para particularidades da Poli.

ABSTRACT

This document is an example of the Poli's thesis format using LaTeX. The document class is based on the ABNT class with little changes to fit some Poli singularities.

LISTA DE FIGURAS

2.1 Pipeline de 3 estágios 18

2.2 Pipeline do ARM7TDMI 19

LISTA DE TABELAS

LISTA DE ABREVIATURAS

B2B Business to Business

ED Especificação Deôntica

EE Especificação Estrutural

EF Especificação Funcional

EnO Entidade Organizacional

EO Especificação Organizacional

ES Esquema Social

IA Inteligência Artificial

IAD Inteligência Artificial Distribuída

KQML Knowledge Query and Manipulation Language

MOISE Model of Organization for multI-agent SystEms

OO Orientação a Objetos

RDP Resolução Distribuída de Problemas

SMA Sistemas Multiagentes

TAEMS Task Analysis, Environment Modeling, and Simulation

LISTA DE SÍMBOLOS

$\sim$ - indiferente a

$\succ$ - melhor que

$\succeq$ - melhor ou indiferente a

$\langle \mathbf{x}, \mathbf{y} \rangle$ - produto escalar entre os vetores $\mathbf{x}$ e $\mathbf{y}$

$\mathcal{A}$ - um conjunto de ações disponíveis

a - uma ação

SUMÁRIO

1 Introdução **13**

1.1 Objetivo 13

1.2 Motivação 13

1.3 Justificativa 14

1.4 Metodologia de Trabalho 14

1.5 Organização do Documento 15

2 Conceitos e Tecnologias Envolvidas **17**

2.1 O Processador ARM7TDMI 17

2.1.1 Arquitetura RISC 17

2.1.2 Pipeline 18

2.1.3 Estados de Operação 20

2.1.4 Modos de Operação 20

2.1.5 Registradores 21

2.1.6 Registradores de Estado 21

2.1.7 Interrupções 21

2.2 A Placa Experimental Evaluator-7T 21

3 O Sistema Operacional KinOS **23**

3.1 Tipos de kernel 23

4 Exemplo **24**

4.1 Motivação: Os Sistemas Multiagentes 24

1 INTRODUÇÃO

1.1 Objetivo

O objetivo deste projeto de formatura é desenvolver um microkernel para a placa experimental Evalutator-7T, contituída de um processador ARM7 e de alguns periféricos simples.

O microkernel implementa os mecanismos básicos de um sistema operacional, como o chaveamento de processos, as chamadas de sistema e utiliza alguns drivers para a comunicação com os periféricos da placa.

Além disso, foram criados alguns programas para testar e exemplificar o funcionamento do microkernel. Entre esses programas, um simples terminal foi desenvolvido para a interação do usuário com o sistema.

1.2 Motivação

As disciplinas de Laboratório de Processadores e de Sistemas Operacionais do curso de Engenharia da Computação na Escola Politécnica da USP, atualmente, estão muito distantes entre si, no entanto o conteúdo das mesmas é muito próximo.

Pensando então em como aproximar essas duas disciplinas, surgiu a idéia de desenvolver uma ferramenta didática que unisse um hardware de estudo simples a um sistema operacional igualmente simples, e que pudesse ser utilizada nas experiências do Laboratório de Processadores.

Para criação desta ferramenta, foi escolhida a placa experimental ARM Evaluator-7T, que possui uma arquitetura ARM e um poder de processamento bastante superior aos sistemas didáticos utilizados atualmente (baseados nos processadores Intel 8051 e o Motorola 68000). Assim sendo, pretende-se atualizar o material didático da disciplina de processadores, trazendo um sistema mais moderno e mais próximo da realidade atual, além de poder se relacionar com o conteúdo da disciplina de Sistemas Operacionais.

Outra motivação do projeto foi aprofundar nossos conhecimentos sobre sistemas operacionais e sobre a arquitetura dos processadores ARM, visto que este processador é, hoje em dia, largamente utilizado em sistemas embarcados e aparelhos celulares.

1.3 Justificativa

O objetivo inicial do projeto era portar um sistema operacional Unix já existente para a placa didática Evaluator-7T.

Inicialmente pensamos em utilizar os sistemas Android e Minix 3, mas ao estudar o kernel dos dois sistemas, vimos que os recursos de memória necessários para executá-los era muito maior que os 512KB disponíveis na placa. Além disso, no caso do Minix 3, teríamos que reescrever o assembly do kernel que atualmente só tem versão para i386, para assembly ARM, o que seria impossível com o tempo disponível para o projeto.

Assim surgiu a idéia de desenvolver um microkernel próprio, com as funcionalidades básicas de um sistema operacional, e que fosse de fácil entendimento; pois como mencionado anteriormente, espera-se que o material desenvolvido seja destinado a melhorar e aproximar o ensino de Sistemas Operacionais com as experiências do Laboratório de Processadores.

1.4 Metodologia de Trabalho

Para a realização desse projeto de formatura procurou-se seguir uma metodologia de trabalho cuja as etapas são descritas a seguir:

- Estudo da Arquitetura ARM e da Placa Didática Evaluator-7T:

  Antes de especificar as funcionalidades que seriam desenvolvidas, um estudo aprofundado da arquitetura ARM foi realizado para compreender o funcionamento do processador para o qual o microkernel foi desenvolvido, o ARM7TDMI.

  Além disso, rodamos alguns exemplos na placa didática Evaluator-7T para adquirir conhecimentos sobre o seu funcionamento e limitações.

- Montagem do Ambiente de Trabalho:

  Paralelamente ao estudo descrito no item anterior, montamos um ambiente de trabalho utilizando a IDE CodeWarrior para o desenvolvimento do código-fonte e o AXD Debugger para depurar o funcionamento do microkernel com ou sem a utilização da placa didática.

Um repositório de controle de versão também foi montado para estocar o material produzido durante do projeto (documentação e código-fonte) e para sincronizar o trabalho dos integrantes do grupo.

- Especificação Funcional do Microkernel:

  O microkernel desenvolvido foi especificado nessa etapa, onde levantamos as funcionalidades básicas de um sistema operacional que deveriam ser implementadas, como o chaveamento de processos e as chamadas de sistema.

- Desenvolvimento do Microkernel:

  Nessa fase, foi desenvolvido o microkernel utilizando como base a especificação definida no item anterior.

- Análise do Microkernel e Conclusões:

  Ao final do desenvolvimento, com base nas dificuldades e soluções encontradas, foi feita uma análise e conclusão sobre o microkernel desenvolvido e sua possível utilização no Laboratório de Processadores para exemplificar os conceitos visto na disciplina de Sistemas Operacionais.

1.5 Organização do Documento

Este documento foi estruturado da seguinte maneira:

- Capítulo 1 (Introdução):

  Apresenta objetivo, motivações, justificativas e a metodologia do trabalho.

- Capítulo 2 (Conceitos e Tecnologias Envolvidas):

  Contextualiza o leitor em aspectos técnicos específicos utilizados no desenvolvimento do trabalho.

- Capítulo 3 (O Sistema Operacional KinOS):

  Descreve como o microkernel foi desenvolvido, quais as suas funcionalidades e como funciona a sua integração com os periféricos da placa didática, com o terminal e com os outros programas implementados.

- Capítulo 4 (Considerações Finais):

Analisa os resultados obtidos em relação ao objetivo do projeto, as conclusões, as contribuições deste trabalho e indica possíveis trabalhos futuros com base neste.

2 CONCEITOS E TECNOLOGIAS ENVOLVIDAS

2.1 O Processador ARM7TDMI

O ARM7TDMI faz parte da família de processadores ARM7 32 bits conhecida por oferecer bom desempenho aliado a um baixo consumo de energia. Essas características fazem com que o ARM7TDMI seja bastante utilizado em media players, vídeo games e, principalmente, em sistemas embarcados e num grande número de aparelhos celulares.

2.1.1 Arquitetura RISC

Os processadores ARM, incluindo o ARM7TDMI, foram projetados com a arquitetura RISC.

RISC (Reduced Instruction Set Computer) é uma arquitetura de computadores baseada em um conjunto simples e pequeno de instruções capazes de serem executadas em um único ou poucos ciclos de relógio.

A idéia por trás da arquitetura RISC é de reduzir a complexidade das instruções executadas pelo hardware e deixar as tarefas mais complexas para o software. Como resultado, o RISC demanda mais do compilador do que os tradicionais computadores CISC (Complex Instruction Set Computer) que, por sua vez, dependem mais do processador já que suas instruções são mais complicadas.

As principais características da arquitetura RISC são:

1. Conjunto reduzido e simples de instruções capazes de serem executadas em único ciclo de máquina.
2. Uso de pipeline, ou seja, o processamento das instruções é quebrado em pequenas unidades que podem ser executadas em paralelo.
3. Presença de um conjunto de registradores.

4. Arquitetura Load-Store: o processador opera somente sobre os dados contidos nos registradores e instruções de load/store transferem dados entre a memória e os registradores.

5. Modos simples de endereçamento à memória.

2.1.2 Pipeline

A arquitetura de pipeline aumenta a velocidade do fluxo de instruções para o processador, pois permite que várias operações ocorram simultaneamente, fazendo o processador e a memória operarem continuamente.

O ARM7 possui uma arquitetura de pipeline de três estágios. Durante operação normal, o processador estará sempre ocupado em executar três instruções em diferentes estágios. Enquanto executa a primeira, decodifica a segunda e busca a terceira.

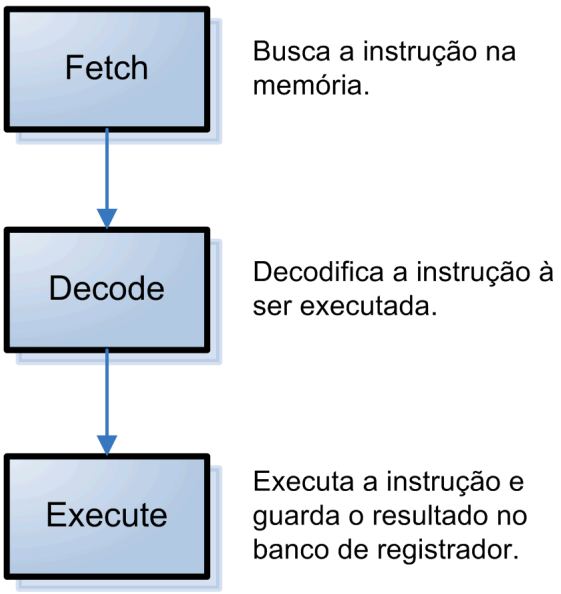

Figura 2.1: Pipeline de 3 estágios

O primeiro estágio de pipeline lê a instrução da memória e incrementa o valor do registrador de endereços, que guarda o valor da próxima instrução a ser buscada. O próximo estágio decodifica a instrução e prepara os sinais de controle necessários para executá-la. O terceiro lê os operandos do banco de registradores, executa as operações através da ALU (Arithmetic Logic Unit), lê ou escreve na memória, se necessário, e guarda o resultado das instruções no banco de registradores.

Algumas características importantes do pipeline do ARM7:

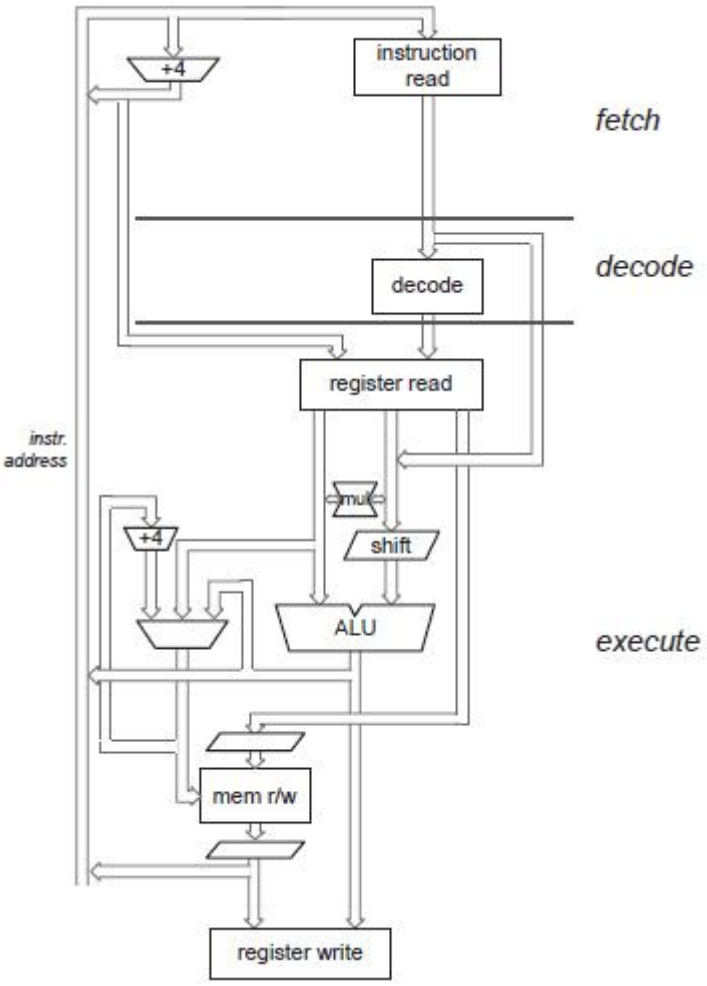

Figura 2.2: Pipeline do ARM7TDMI

- O Program Counter (PC) ao invés de apontar para a instrução que esta sendo executada, aponta para a instrução que esta sendo buscada na memória.
- O processador só processa a instrução quando essa passa completamente pelo estágio execute. Ou seja, somente quando a quarta instrução é buscada (fetched).
- A execução de uma instrução de branch através da modificação do PC provoca a descarga de todas as outras instruções do pipeline.
- Uma instrução no estágio execute será completada mesmo se acontecer uma interrupção. As outras instruções no pipeline serão abandonadas e o processador começará a encher o pipeline a partir da entrada apropriada no vetor de interrupção.

2.1.3 Estados de Operação

O processador ARM7TDMI possui dois estados de operação:

- ARM: modo normal, onde o processador executa instruções de 32 bits (cada instrução corresponde a uma palavra);
- Thumb: modo especial, onde o processador executa instruções de 16 bits que correspondem à meia palavra.

Instruções Thumbs são um conjunto de instruções de 16 bits equivalentes as instruções 32 bits ARM. A vantagem em tal esquema, é que a densidade de código aumenta, já que o espaço necessário para um mesmo número de instruções é menor. Em compensação, nem todas as instruções ARM tem um equivalente Thumb.

Neste projeto, decidimos pela utilização do processador no modo ARM que facilita o desenvolvimento por possuir um número maior de instruções.

2.1.4 Modos de Operação

Os processadores ARM possuem 7 modos de operação:

Modo	**Identificador**	**Descrição**
Usuário	usr	Execução normal de programas.
FIQ(Fast Interrupt)	fiq	Tratamento de interrupções rápidas.
IRQ (Interrupt)	irq	Tratamento de interrupções comuns.
Supervisor	svc	Modo protegido para o sistema operacional.
Abort	abt	Usado para implementar memória virtual ou manipular violações na memória.
Sistema	sys	Executa rotinas privilegiadas do sistema operacional.
Indefinido	und	Modo usado quando uma instrução desconhecida é executada.

Mudanças no modo de operação podem ser feitas através de programas, ou podem ser causadas por interrupções externas ou exceções (interrupções de software).

A maioria dos programas roda no modo Usuário. Quando o processador esta no modo Usuário, o programa que esta sendo executado não pode acessar alguns recursos protegidos do sistema ou mudar de modo sem ser através de uma interrupção.

Os outros modos são conhecidos como modos privilegiados. Eles têm total acesso aos recursos do sistema e podem mudar livremente de modo de operação. Cinco desses modos são conhecidos como modos de interrupção: FIQ, IRQ, Supervisor, Abort e Indefinido.

Entra-se nesses modos quando uma interrupção ocorre. Cada um deles possui registradores adicionais que permitem salvar o modo Usuário quando uma interrupção ocorre.

O modo remanescente é o modo Sistema, que não é acessível por interrupção e usa os mesmos registradores disponíveis para o modo Usuário. No entanto, este é um modo privilegiado e, assim, não possui as restrições do modo Usuário. Este modo destina-se as operações que necessitam de acesso aos recursos do sistema, mas que querem evitar o uso adicional dos registradores associados aos modos de interrupção.

2.1.5 Registradores

O processador ARM7TDMI tem um total de 37 registradores:

- 31 registradores de 32 bits de uso geral
- 6 registradores de estado

Esses registradores não são todos acessíveis ao mesmo tempo. O modo de operação do processador determina quais registradores são disponíveis ao programador.

2.1.6 Registradores de Estado

2.1.7 Interrupções

2.2 A Placa Experimental Evaluator-7T

O principal elemento de hardware deste projeto é a placa experimental ARM Evaluator-7T, baseada no processador ARM7TDMI, um processador RISC de 32 bits capaz de executar o conjunto de instruções denominado Thumb.

Os elementos presentes na arquitetura da placa Evaluator-7T são os seguintes:

- Microcontrolador Samsung KS32C50100
- 512kB EPROM flash
- 512kB RAM estática (SRAM)
- Dois conectores RS232 de 9 pinos tipo D
- Botões de reset e de interrupção
- Quatro LEDs programáveis pelo usuário e um display de 7 segmentos
- Entrada de usuário por um interruptor DIP com 4 elementos
- Conector Multi-ICE
- Clock de 10MHz (o processador usa-o para gerar um clock de 50MHz)
- Regulador de tensão de 3.3V

Com relação à memória flash da placa, ela vem de fábrica com o bootstrap loader da placa e programa monitor de debug. O restante dela pode ser usado para os programas de usuário.

Já em relação às duas portas seriais presentes na placa, cada uma tem usos específicos. A primeira, chamada DEBUG, é usada pelo monitor de debug ou pelo programa bootstrap presente na placa. Ela está conectada ao UART1 do microcontrolador. A segunda, chamada USER, é de uso genérico e está disponível para uso em programas. Ela está conectada ao UART0 do microcontrolador.

3 O SISTEMA OPERACIONAL KINOS

Quando um sistema operacional é projetado, geralmente é adotada uma arquitetura em módulos, a fim de se dividir esta tarefa complexa em várias partes com funções bem definidas, como o gerenciamento de E/S, drivers e aplicações. Nesta arquitetura, o módulo que é utilizado por todos os outros e que interage diretamente com o hardware é chamado de kernel. Por seu contato direto com o hardware, este código é escrito em assembly, e por ser utilizado por todos os outros módulos do sistema operacional, deve ser rápido para não comprometer a performance do sistema. Ele roda quando o hardware é ligado, a fim de se inicializar o sistema, e após isso apenas quando há algum tipo de interrupção.

3.1 Tipos de kernel

Há basicamente dois tipos de kernel: monolíticos e microkernels.