Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Título da monografia

Alexandre Silva Dantas
Matheus Costa de Sousa Carvalho Pimenta

Monografia apresentada como requisito parcial
para conclusão do Bacharelado em Ciência da Computação

Orientador
Prof. Dr. Marcus Vinicius Lamar]

Brasília
2012

Universidade de Brasília — UnB
Instituto de Ciências Exatas
Departamento de Ciência da Computação
Bacharelado em Ciência da Computação

Coordenador: Prof. Dr. Coordenador

Banca examinadora composta por:

Prof. Dr. Marcus Vinicius Lamar] (Orientador) — CIC/UnB
Prof. Dr. Professor I — CIC/UnB
Prof. Dr. Professor II — CIC/UnB

CIP — Catalogação Internacional na Publicação

Dantas, Alexandre Silva.

Título da monografia / Alexandre Silva Dantas, Matheus Costa de Sousa Carvalho Pimenta. Brasília : UnB, 2012.
25 p. : il. ; 29,5 cm.

Monografia (Graduação) — Universidade de Brasília, Brasília, 2012.

1. palvrachave1, 2. palvrachave2, 3. palvrachave3

CDU 004.4

Endereço: Universidade de Brasília
Campus Universitário Darcy Ribeiro — Asa Norte
CEP 70910-900
Brasília–DF — Brasil

Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Título da monografia

Alexandre Silva Dantas
Matheus Costa de Sousa Carvalho Pimenta

Monografia apresentada como requisito parcial
para conclusão do Bacharelado em Ciência da Computação

Prof. Dr. Marcus Vinicius Lamar] (Orientador)
CIC/UnB

Prof. Dr. Professor I
CIC/UnB

Prof. Dr. Professor II
CIC/UnB

Prof. Dr. Coordenador
Coordenador do Bacharelado em Ciência da Computação

Brasília, 12 de dezembro de 2012

Dedicatória

Dedico a....**mamãe**

Agradecimentos

Agradeço a....*papai*

Abstract

A ciência...

Palavras-chave: palvrachave1, palvrachave2, palvrachave3

Abstract

The science...

Keywords: keyword1, keyword2, keyword3

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

Com o aumento na capacidade de processamento dos computadores modernos, aumenta-se a complexidade de seus componentes internos. As arquiteturas de computadores modernos possuem mais instruções que as de computadores antigos, devido à refinação do *hardware*.

Em contrapartida, existem computadores com conjunto reduzido de instruções (*RISC*) e até máquinas com apenas uma única instrução (*OISC*). Esse trabalho se concentra no segundo, mais especificamente na máquina de instrução única *SUBLEQ*.

Esse trabalho visa desenvolver um sistema computacional completo (*hardware*, *software* básico e *software* de aplicação) baseado na arquitetura de instrução única *SUBLEQ*. Dentre as aplicações possíveis, esperamos que o gasto de energia no processador seja constante, em função do tempo.

Vamos explorar os conceitos básicos que levem ao objetivo final.

1.1 Conceitos Básicos

Computadores são sistemas incrivelmente complexos. Inúmeros componentes com papeis específicos necessitam se intercomunicar para executar a mais simples das tarefas. Dessa forma, para compreender seu funcionamento, se faz o uso de camadas de abstrações.

Essas camadas exercem funções diferentes e são visíveis de acordo com seu uso — um usuário final não precisa saber programar para usar um processador de texto; da mesma forma, um programador não necessita saber da estrutura dos circuitos internos. Cada camada possui seu domínio, sendo as mais próximas do usuário final denominadas de "alto-nível" e as mais próximas dos transistores e fios, "baixo-nível". Observe a figura 1.1, especificada de acordo com Murdocca [5].

Uma definição muito importante para o programador de sistema é a Arquitetura do Conjunto de Instruções (*Instruction Set Arquitecture*) — de agora em diante referida apenas como arquitetura, ou *ISA*. Hennessy a define como "o limite entre *software* e *hardware*" [2].

A *ISA* descreve vários componentes essenciais para a criação de programas de sistema. Seu *design* define a memória interna do processador, o endereçamento de memória interna e externa, quais instruções/operações são suportadas, tipos e tamanhos de operandos, dentre muitos outros [6].

Existem tipos diferentes de *ISA*, sendo comuns as arquiteturas *RISC* e *CISC*.

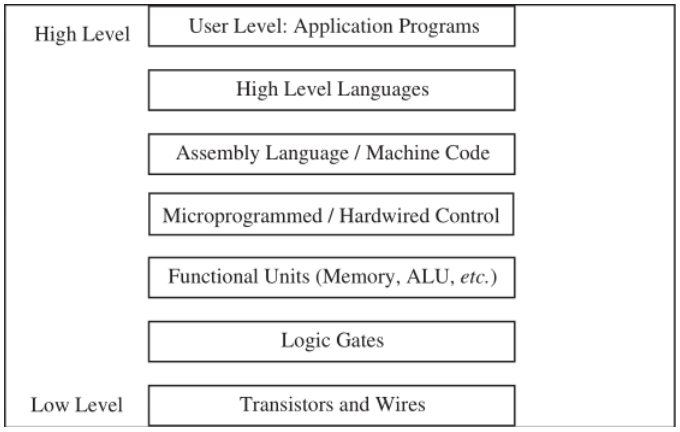

Figura 1.1: Camadas de abstração de um computador.

Arquiteturas *RISC* (*Reduced Instruction Set Computer*) possuem uma quantidade reduzida de instruções. Em geral são instruções simples e rápidas que têm de ser combinadas para ações mais complexas. Já arquiteturas *CISC* (*Complex Instruction Set Computer*) provêem uma quantidade maior de instruções, que nativamente executam ações mais complicadas e abrangentes.

Instruções da arquitetura *RISC* são mais simples; elas partem da filosofia de otimizar os casos frequentes, visando tornar o comportamento geral mais rápido. Murdocca argumenta que um conjunto de instruções mais simples resulta numa central de processamento simples e menor, liberando espaço no processador para outros componentes, como registradores [5].

Porém Mostafa argumenta que isso traz a desvantagem de que uma grande quantidade de instruções são necessárias para executar uma função simples [4], possivelmente reduzindo o desempenho geral. Por fim, isso tambem causa um problema cognitivo, já que programas *RISC* tendem a ser mais verbosos e depositarem a complexidade do programa nos ombros do programador.

Além de ambas as *ISA*s citadas acima, existe a arquitetura *OISC* (*One Instruction Set Computer*). Ela define computadores com apenas uma única instrução. De acordo com Gilreath, *OISC* é como um *CISC* em um nível mais alto de abstração, já que precisa-se combinar essa única instrução de diversas formas para sintetizar o que seriam as instruções mais complexas [1].

Pela própria definição, arquiteturas *OISC* possuem as desvantagens de *RISC* em escala muito maior. Qualquer função simples necessitará de várias combinações da única instrução, dificultando tanto a velocidade quanto compreensão do programa final.

Entretanto, existem vantagens na previsibilidade de máquinas *OISC*.

Em máquinas não-*OISC*, existe um conjunto bem-definido de instruções que podem ser executadas. Cada uma exige demandas específicas do processador, resultando em gastos de energia possivelmente diferentes.

Dessa forma, ao se monitorar o gasto de energia por um período suficiente de tempo, pode-se observar os padrões de gasto de energia do processador. Então, um observador externo poderá deduzir quais instruções foram executadas na máquina sem necessariamente ter acesso à mesma.

Considerando que arquiteturas *OISC* possuem apenas uma instrução, cuja demanda ao processador é única, assume-se que o gasto de energia será constante. Logo, não seria possível determinar quais ações essa máquina executou independentemente da quantidade de tempo de monitoramento.

O ponto abordado nesse trabalho é exatamente esse — determinar se o gasto de energia em função do tempo é constante numa máquina *OISC*. Se for o caso, pode-se determinar aplicações interessantes para esse tipo de computador na área de segurança de informação e criptografia.

Primeiramente, devemos determinar que instrução será usada na nossa máquina *OISC*.

1.2 Linguagem de Montagem Subleq

Existem várias máquinas de arquitetura *OISC*. Uma delas é a máquina que possui apenas a instrução *SUBLEQ* (*Subtract and Branch on Less or Equal* [3].

Capítulo 2

Protótipo do Segundo Capítulo

Não tem *nada* aqui.

Referências

[1] Laplante Phillip A. Gilreath William F. *Computer Architecture: A Minimalist Perspective: Dynamics and Sustainability*. Springer, 2003.

[2] Patterson David Hennessy John L. *Computer Architecture: A Quantitative Approach*. Morgan Kaufman Publishers, 1989. 1

[3] O. Mazonka and A. Kolodin. A Simple Multi-Processor Computer Based on Subleq. *ArXiv e-prints*, June 2011. 3

[4] Hesham El-Rewini Mostafa Abd-El-Barr. *Fundamentals of Computer Organization and Architecture*. Wiley, December 2004. 2

[5] Heuring Vincent P. Murdocca Miles. *Principles of Computer Architecture*. Prentice Hall, 1999. 1, 2

[6] Hennessy John L. Patterson David. *Computer Organization and Design: the Hardware/Software Interface*. The Morgan Kaufmann Series in Computer Architecture and Design. Morgan Kaufmann Publishers, 2011. 1