INTRODUÇÃO: O QUE É RISC?

Durante grande parte da história dos computadores pessoais, o modelo predominante dos microprocessadores tem sido da Intel Corporation. O primeiro processador no IBM PC foi o Intel 8088. As gerações seguintes de processadores Intel foram da família 80X86 – 8086, 80286, 80386, 80486. Todos eram versões elaboradas do 8088 original, mas com desempenho melhorado de duas maneiras – operando mais rapidamente ou manejando mais dados simultaneamente. O 8088, por exemplo, operava a 4,7 Mhz – ou 4,7 milhões de oscilações por segundo – e alguns chips 80486 vão tão rápido quanto 133 MHz. O 8088 conseguia manejar 8 bits de dados por vez, enquanto o 80486 maneja 32 bits internamente.

Mas apesar das mudanças, os processadores Intel até o 80486 eram baseados em uma filosofia de projeto denominada CISC, do Inglês Complex Instruction Set Computing. O padrão CISC usa comandos que incorporam muitas instruções mínimas para executar uma simples operação. É como se ele fosse uma faca que corta dados e códigos. Ou ainda, por comparação, como um bisturi que corta pedaços cada vez menores e mais delicados de dados e códigos. Esse bisturi é chamado RISC, do Inglês Reduced Instruction Set Computing. Projetos RISC são encontrados em processadores mais novos, entre eles o DEC Alpha, o RISC 6000 da IBM, o processador Power PC e, embora a Intel não chame os processadores Pentium de chips RISC, eles têm muito em comum com o padrão RISC de executar códigos.

O RISC é um projeto menos complicado que usa muitas instruções mais simples para executar uma operação comparável e em menos tempo que um processador CISC que executa uma comando maior e mais complicado. Os chips RISC podem ser fisicamente menores que os chips CISC. E pelo fato de usarem menos transistores, geralmente sua fabricação é mais barata e produzem menos calor.

Muitas previsões vêm afirmando que o futuro dos processadores caminha para um projeto RISC, e provavelmente eles estejam corretas. Mas não tem havido um movimento de venda em massa do RISC, por duas razões. A mais importante delas é manter a compatibilidade com o vasto número de software aplicativo, desenvolvido para trabalhar com os processadores Intel CISC mais antigos. A Segunda razão é que você não recebe todos os benefícios da arquitetura RISC, a não ser que esteja usando um sistema operacional e programas que tenham sido criados e compilados especificamente para tirar vantagens das operações RISC. É a clássica situação do ovo e da galinha. Alguns fabricantes de computadores estão oferecendo processadores RISC para se projetarem como líderes em tecnologia. Ele executam os antigos programas CISC somente através da emulação de um processador CISC, o que acaba negando as vantagens do RISC. Ao mesmo tempo, os criadores de software relutam em converter seus programas para versões compiladas RISC quando ainda não há muitas pessoas usando PCs baseados em RISC.

Mas o Pentium mudou isso. Os técnicos podem argumentar se o Pentium pode ser um verdadeiro chip RISC. É um cômodo comprometimento. O Pentium executa antigos aplicativos e sistemas operacionais e ao mesmo tempo oferece vantagens na velocidade de programas criados especialmente para usar a capacidade do padrão Pentium. E com um sistema operacional avançado como o Windows NT, você pode colocar mais de um microprocessador Pentium no computador para dobrar o poder de processamento.

CARACTERÍSTICAS E FUNCIONAMENTO DA ARQUITETURA RISC

O que é RISC?

Processadores de Conjunto Reduzido de Instruções Computacionais ('Reduced Instruction Set Computing' ou, abreviadamente RISC) são muito mais velozes do que os processadores comuns (CISC). O termo 'Conjunto Reduzido de Instruções' refere-se ao número de ciclos de clock que o processador leva para selecionar uma instrução. Processadores comuns levam vários ciclos de clock para selecionar uma única instrução. Um chip RISC, por outro lado, pode selecionar e executar uma instrução a cada ciclo de clock.

RISC: Princípio de Processamento

As funções de comando criadas em um processador RISC consistem de muitas instruções pequenas e individuais, que realizam apenas uma única tarefa. O software aplicativo, que precisa ser recompilado especialmente para um processador RISC, realiza a tarefa de informar ao processador qual a combinação de seus comandos menores deve ser executada para completar uma operação maior.

Todos os comandos RISC são do mesmo tamanho, e há somente uma maneira de carregá-los e armazená-los. Além disto, uma vez que cada comando já está na forma de microcódigo, os processadores RISC não precisam de um passo extra para passar as instruções que eles recebem através de uma unidade que transforme os comandos complexos em microcódigos mais simples. Como resultado destas diferenças, os comandos RISC são carregados muito mais rapidamente que os comandos CISC.

Durante a compilação de software especificamente para um chip RISC, o compilador determina quais comandos não vão depender dos resultados de outros. Como estes comandos não têm de esperar por outros comandos, o processador pode executar simultaneamente até 10 comandos em paralelo.

Como o processador RISC está trabalhando com comandos mais simples, seu conjunto de circuitos também pode ser mais simples, Os comandos RISC passam por um número menor de transistores em circuitos mais curtos, desta forma os comandos são executados mais rapidamente. O resultado é que os processadores RISC geralmente necessitam apenas de um ciclo do relógio da CPU por instrução. O

número de ciclos necessários para completar uma operação inteira depende do número de pequenos comandos que constituem essa operação. Em uma operação comparável, o tempo necessário para interpretar e executar as instruções RISC é bem menor que o tempo para se carregar e decifrar um comando CISC complexo e então executar cada um de seus componentes.

Resumo das Características

Conjunto de Instruções Simples: a arquitetura RISC, é constituída por um conjunto de instruções simples, instruções básicas, que compõem instruções mais complexas.

Instruções do Mesmo Tamanho: as instruções têm sempre um único tamanho, e uma única maneira de executá-las.

Uma Instrução por Ciclo de Máquina: todas as instruções são executadas em um único ciclo, fazendo com que o processador execute várias instruções ao mesmo tempo, tornando o processamento muito mais rápido Isto é possível devido a um tipo de tecnologia chamada de Pipelined, que abordaremos no próximo tópico.

Pipelined: Uma Técnica Fundamental

Os projetistas de RISC se preocupam principalmente, em criar técnicas e dispositivos que acelerem o processamento da informação. Uma dessas técnicas é o Pipelined. O Pipelined é uma técnica em que o hardware processa mais de uma instrução de cada vez. Ele não espera o término de uma instrução para executar outra. Em uma máquina CISC típica, temos, basicamente, quatro fases para a execução: busca, decodificação, execução e escrita. Em uma máquina RISC, temos as mesmas fases, só que executadas em paralelo. Uma fase não precisa esperar a outra terminar, para que ela se inicie. Este procedimento, não diminui o tempo de execução da tarefa, mas melhora o processamento global.

VANTAGENS E DESVANTAGENS DO RISC

Vantagens

Velocidade

Devido a tecnologia pipelined os processadores RISC alcançam duas a quatro vezes a performance dos processadores CISC usando tecnologia de semicondutor equivalente e os mesmos valores de clock.

Simplicidade do Hardware

Pelo fato de um processador RISC trabalhar com instruções simples, o processador utiliza de menos espaço no chip, funções extras como circuito de gerenciamento de memória e unidade aritmética armazenada num mesmo chip. Chips menores permitem que o fabricante armazenem mais dispositivos em uma única pastilha, o que pode baixar consideravelmente o custo.

Instrução de máquina simples

As instruções construídas para um processador RISC são simples e pequenas o que aumenta a sua performance.

Desvantagens

A transição da arquitetura CISC para arquitetura RISC pode apresentar alguns problemas devido ao fato que os engenheiros de software podem ter problemas para fazer a transição do código de memória de maneira correta.

Qualidade do Código

A performance de um processador RISC depende diretamente do código gerado pelo programador. No caso de um código mal desenvolvido o processador pode gastar um tempo demasiado na execução das instruções, isto faz com que a performance de uma máquina RISC dependa em grande parte da qualidade do código, gerado pelo programador.

Expansão do Código

O fato da arquitetura CISC trabalhar com instrução única com ações complexas e as máquinas RISC trabalharem com instrução simples a transição do código pode acarretar problemas. O termo "expansão do código" refere-se ao aumento de tamanho que se obtém de um programa originalmente compilado para uma máquina CISC, ter sido recompilado para uma máquina RISC. A expansão vai depender da capacidade do programador e a natureza do conjunto de instruções de máquina.

Projeto de Sistema

Outra desvantagem que a arquitetura RISC apresenta é o fato de requerer sistema de memória rápida para alimentar suas instruções. Tipicamente sistemas baseados nesta arquitetura costumam apresentar grande quantidade de memória cache interna, conhecida como "first-level cache", o que encarece o projeto

DIFERENÇAS ENTRE ARQUITETURAS RISC E CISC

A transição de CISC para RISC foi uma mudança radical na arquitetura. Os conjuntos de instruções foram mudados, sacrificando a compatibilidade binária para o desempenho. Reduzindo o conjunto de instruções, o processador cabia em um chip menor que permitia que os desenvolvedores aumentassem a velocidade do clock. Além disso, o processador poderia ser pipelineado. RISC, "resolve os casos comuns mais rápidos", era o princípio que conduziu a um aumento impressionante da performance comparado com os processadores CISC. Os processadores mais adiantados tais como IBM RT, SPARCv7 e MIPS R2000 aderiram completamente ao princípio fundamental RISC. Entretanto, enquanto a tecnologia avançava para aumentar os tamanhos dos dados e aumentar a densidade do transistor, os desenvolvedores do processador RISC começaram a considerar maneiras de usar esse espaço novo do chip. Alguns dos usos desse espaço incluíram:

Registros adicionais;

- On-chip chaces que são cronometrados tão rapidamente quanto o processador;
- Unidades funcionais adicionais para execução superescalar;
- Instruções adicionais não-RISC (mais rápidas);
- On-chip aceitando operações de ponto flutuante;
- Profundidade aumentada no Pipelined.

Assim, a geração atual de processadores de desempenho elevado carrega poucas características dos processadores que iniciaram a revolução RISC.

Arquiteturas RISC

Nas arquiteturas RISC a grande maioria dos processadores possuem 32 registros genéricos de 32 bits – para representar valores inteiros e/ou endereços de memória – para além de 32 registros de 32 bits para representação de valores em vírgula flutuante. Estes registros são considerados de uso genérico e podem ser usados explicitamente por qualquer instrução que aceda operandos de registros, com exceção de um registro que contenha o valor ZERO, que não pode ser alterado (só de leitura). Ainda visível ao programador está sempre também o apontador para a próxima instrução, o instruction pointer ou program counter.

Arquiteturas CISC

A existência de um grande número de registros nas arquiteturas RISC, aliado à evolução da tecnologia dos compiladores dos últimos anos (em especial, na geração de código), vem permitindo representar a maioria das variáveis escalares diretamente em registro, não havendo necessidade de recorrer com tanta freqüência à memória. Esta organização não foi economicamente viável nas gerações anteriores de microprocessadores, com destaque para a família da Motorola (M680x0) e, ainda mais antiga, a família da Intel (ix86). Estes processadores dispunham de um menor nº de registros e, conseqüentemente, uma diferente organização que suportasse eficientemente diversos mecanismos de acesso à memória.

No caso da família M680x0, o programador tinha disponível dois bancos de 8 registros genéricos de 32 bits: um para dados (D) e outro para apontadores para a memória (A), suportando este último banco um variado leque de modos de endereçamento à memória. Apenas um dos registros (A7) é usado implicitamente em certas operações de manuseamento da stack.

A família Intel é mais complicada, por não Ter variadamente registros de uso genérico. A arquitetura de base dispõe efetivamente de 4 registros para conter operandos aritméticos (A, B, C e D), mais 4 para trabalhar com apontadores para a memória (BP, SP, DI e SI) e outros 4 para lidar com uma memória segmentada (CS, DS, SS e ES; a única maneira de uma arquitetura de 16 bits poder aceder a mais de 64K células de memória). Cada um destes registros não pode ser considerado de uso genérico, pois quase

todos eles são usados implicitamente (isto é, sem o programador explicitar o seu uso) em várias instruções (por exemplo, os registros A e D funcionam de acumuladores em operações de multiplicação e divisão, enquanto o registro C é usado implicitamente como variável de contagem em instruções de controle de ciclos). A situação complica-se ainda mais com a variação da dimensão dos registros na mesma família (de registros de 16 bits no i286 para registros de 32 bits no i386), pois o formato de instrução foi concebido para distinguir apenas operandos de 8 e de 16 bits, e um bit bastava; para garantir compatibilidade ao longo de toda a arquitetura, os novos processadores têm de distinguir operandos de 8, 16 e 32 bits, usando o mesmo formato de instrução.

Diferenças entre RISC e CISC

Todos os processadores dispõem de instruções de salto "de ida e volta", normalmente designados de instruções de chamada de sub-rotinas: nestas, para além de se alterar o conteúdo do registro PC como qualquer instrução de salto, primeiro guarda-se o endereço de instrução que segue a instrução de salto (e que se encontra no PC); nas arquiteturas CISC este valor é normalmente guardado na stack; nas arquiteturas RISC esse valor é normalmente guardado num registro.

Conjunto de instruções de um processador RISC

O conjunto de instruções que cada processador suporta é bastante variado. Contudo é possível identificar e caracterizar grupos de instruções que se encontram presentes em qualquer arquitetura.

Para transferência de informação: integram este grupo as instruções que transferem informação entre registros e a memória (load/store), entre registros (simulado no Assembler do MIPS, e implementando com uma soma com o registro 0), diretamente entre posições de memória (suportado por exemplo, no M680x0, mas não disponível em qualquer arquitetura RISC), ou entre registros e a stack, com incremento/decremento automático do sp (disponível em qualquer arquitetura CISC, mas incomum em arquiteturas RISC);

Operações aritméticas, lógicas, ...: soma, subtração e multiplicação com inteiros e fp, e operações lógicas AND, OR, NOT, ShiftLeft/Right são as mais comuns; alguns processadores suportam ainda a divisão, quer diretamente por hardware, quer por microprogramação.

Acesso a operandos em memória em CISC e RISC

Uma das conseqüências do fato das arquiteturas CISC disporem de um menor número de registros é a alocação das variáveis escalares, em regra, a posições de memória, enquanto que nas arquiteturas RISC, a regra era a alocação a registros. Atendendo ao modelo de programação estruturada tão em voga nos anos 70, ao fato da maioria das variáveis escalares serem locais a um dado procedimento, e à necessidade do modelo de programação ter de suportar o alinhamento e recursividade de procedimentos, as arquiteturas CISC necessitavam de um leque rico de modos de endereçamento à memória, para reduzir o gap semântico entre uma HLL e a linguagem máquina.

Resume-se aqui, as principais diferenças entre as arquiteturas CISC e RISC, nas facilidades de acesso a operandos que se encontram em memória:

CISC: grande riqueza na especificação de modos de endereçamento; exemplo do i86: modo absoluto; por registro indireto –(R), - (SP), (SP)+; por registro base –(Rb)+desloc8,16,32, (Rb)+(R), (Rb)+desloc8,16,32; com acessos indiretos à memória, isto é, com apontadores para as variáveis sem memória armazenados em células de memória.

RISC: apenas em operações load/store e apenas 1 ou 2 modos; exemplo do MIPS: apenas (R)+desloc16.

Operações lógicas e aritméticas em CISC e RISC

Duas grandes diferenças se fazem notar entre estes 2 modelos: na localização dos operandos neste tipo de operações, e o nº de operandos que é possível especificar em cada instrução. Resumindo o que foi visto anteriormente (complementando com exemplos na aula):

CISC: 1 ou mais operandos em memória (máx 1 no i86 e M68K); nem sempre se especificam 3 operandos (máx 2 no i86 eM68K).

RISC: operandos sempre em registros; 3 operandos especificados (1 dest, 2 fontes).

CONCLUSÃO

A arquitetura RISC caracteriza-se por sua estrutura dinâmica, simples, objetiva e de grande velocidade. Trabalhando com um conjunto de instruções curtas e simples, consegue alcançar grande performance, que faz com que suas aplicações tenham grande potencial em Sistemas de Rede, Internet e Bancos de Dados.

Mas toda arquitetura tem suas vantagens e desvantagens. Se de um lado, esta arquitetura apresenta tantas qualidades, ela pode também se transformar num sistema de baixa performance se não for projetado de forma correta. Seus códigos têm de ser bem construídos e bem codificados para que se tenha alto desempenho. No geral, as máquinas baseadas nessa arquitetura, conquistam cada vez mais o mercado de alto nível, sendo responsável por grande parte dos sistemas de grande porte mundiais.