UNIVERSIDADE CATÓLICA DE PERNAMBUCO

ESCOLA UNICAP ICAM TECH

CURSO DE COMPUTAÇÃO

ANDRÉ LUIZ TÔRRES LINS DE CARVALHO

RAYSSA RACHEL D'ÁVILA MUNIZ

RICARDO ANDRÉ OSCAR DA SILVA JÚNIOR

VICTOR GABRIEL ARAÚJO DE LIMA

PESQUISA - ATIVIDADE 1- ARQUITETURA E ORGANIZAÇÃO DE COMPUTADORES II

RECIFE

2023

Arquitetura de processadores RISC-V e ARM e arquitetura de Multiprocessadores AMP (Multiprocessamento Assimétrico) e SMP (Multiprocessamento Simétrico):

RISC-V e ARM são duas arquiteturas RISC que oferecem processamento eficiente e consumo de energia reduzido devido ao seu conjunto de instruções simplificadas. Embora ambos tenham conjuntos de instruções semelhantes que incluem operações aritméticas e lógicas, acesso à memória, transferência de controle e instruções especializadas, o RISC-V se destaca por seu conjunto de instruções modular e extensível que permite a personalização para atender a requisitos específicos. Em contraste, o ARM usa um conjunto fixo de instruções que são otimizadas para desempenho e eficiência de energia.

A RISC-V usa formatos instruções de comprimento variável, já o ARM, usa comprimento fixo que são fáceis de decodificar, permitindo uma execução rápida e eficiente. Ambos suportam vários tipos de dados, como números inteiros, números de ponto flutuante e dados vetoriais, com o RISC-V tendo instruções específicas para cada tipo de dados e o ARM usando um conjunto unificado de instruções que podem lidar com diferentes tipos de dados. Eles também suportam diferentes tipos de instrução, incluindo instruções aritméticas, lógicas, shift, load/store e branch. O RISC-V também suporta instruções vetoriais e operações de memória atômica, enquanto o ARM possui instruções especializadas para multimídia, criptografia e outras aplicações.

Ambas as arquiteturas possuem unidades de controle e caminhos de dados simples e eficientes. Ambos usam uma arquitetura Harvard com instruções separadas e memória de dados, permitindo a busca simultânea de instruções e acesso a dados. Além disso, ambos usam um arquivo de registros para armazenar e acessar dados. O RISC-V possui 32 registradores de uso geral, enquanto o ARM possui 16, incluindo o contador de programa, ponteiro de pilha e registradores de status.

Os multiprocessadores utilizam vários processadores para executar tarefas computacionais. Dois tipos de arquiteturas de multiprocessador são AMP e SMP . Essas, utilizam o repertório de instruções e o formato de seus processadores subjacentes. As instruções são executadas independentemente em cada processador e são armazenadas na memória como seriam em um sistema de processador único.

Os multiprocessadores suportam vários tipos de dados, incluindo números inteiros, números de ponto flutuante e dados vetoriais. Os tipos de dados suportados dependem dos processadores subjacentes. Os tipos de instrução são consistentes em AMP, SMP e seus processadores subjacentes.Cada processador pode executar independentemente seu próprio conjunto de instruções.

As arquiteturas AMP consistem em vários processadores, cada um com sua própria unidade de controle, caminho de dados e conjunto de registradores. Isso permite a execução independente de instruções sem coordenação com outros processadores. As arquiteturas SMP compartilham uma memória comum e uma unidade de controle em vários processadores, permitindo a execução coordenada de instruções. No entanto, apenas um processador pode escrever em um registrador por vez.

TAMANHO DA PALAVRA DE CÓDIGO E DE DADOS

RISC-V: Palavras num intervalo de 8 a 64 bits. A depender da implementação realizada nesse tipo de processador, esse tamanho pode variar dentro desse intervalo, suportando dados de 8, 16, 32 (sendo esse formato de instrução, o mais comumente utilizado), e 64 bits.

ARM: Isso varia de acordo com o modo do processador ARM que está sendo utilizado. Adotando o ARMv6 como exemplo, o tamanho da palavra código é de 32 bits. As quais podem ser divididas em 4 bytes, e o tamanho da palavra de dados pode ser de 32 ou 16 bits.

DESEMPENHO

RISC-V: Mínimo gasto de energia sem gerar sobrecarga em outras partes do seu sistema. Outrossim, pelo fato de ser uma arquitetura a qual tem o número de instruções reduzidas, ao compararmos com a CISC(Complex Instruction Set Computing), por exemplo, ela consegue operar numa velocidade mais alta, aumentando o número de execuções executadas num menor tempo, logo, aumentando o nível de desempenho.

ARM: Assim como o processador RISC-V, também consome menos energia, e no caso do processador ARM(Advanced RISC Machine), é bastante eficiente principalmente quando se trata de desempenho por watt(eficiência no consumo de potência). Além disso, possui uma boa dissipação de calor o que ajuda a não comprometer o seu desempenho.

MODOS DE ENDEREÇAMENTO

RISC-V: Há 4 modos de endereçamento. Dentre eles temos: **1->** o endereçamento por imediato - quando o valor que vamos usar na instrução já está dentro da instrução, pois não precisamos realizar a busca desse valor em algum lugar específico. **2->** endereçamento por registrador, que ocorre quando desejamos ler um valor que está dentro de um registrador, logo, a instrução possui um campo de referência (um identificador) para esse registrador que vai ser lido, então a partir dessa leitura desse dado que vai estar dentro do banco de registradores, iremos conseguir acessar esse dado para conseguir executar a operação da instrução e armazenar o resultado em outro registrador. **3->** endereçamento por deslocamento por base, onde o valor vai estar alocado na memória principal e para acessar esse valor calcula-se o valor efetivo da memória, então temos o registrador, que vai conter qual é o endereço base da memória e o valor imediático com o deslocamento desse endereço base. E o cálculo feito é a soma do deslocamento + o endereço base. Após isso, temos o endereço efetivo em que vamos acessar o dado na memória principal, que pode ser de qualquer tamanho suportado pelo RISC-V(ex: byte, halfword,word). **4->** endereçamento relativo ao PC (Program Counter), esse tipo é bastante usado pelas instruções de branch, ao passo que, quando realiza-se um desvio, coloca-se o deslocamento com base no PC. Aqui o valor que está no campo de imediato corresponde ao deslocamento enquanto o endereço base é o endereço que está no PC.

ARM: 1-> Endereçamento base-mais-offset ex: LDR r0,[r1,#8](nesse tipo, é carregado em um registrador - r0 - uma palavra de código a partir da posição de memória r1+8 - oito é o deslocamento). **2->** Endereçamento por auto-incremento: modifica a referência do registrador base, pois passa-se à apontar para o início do vetor, e a cada passada vai incrementando o tamanho do deslocamento escolhido. **3->** Mesma lógica de endereçamento por auto-incremento, contudo, ao contrário. **4->** Endereçamento por pós-incremento ex: LDR r0,[r1],#16 (insere no registrador r0, o conteúdo da posição de memória r1, e a partir disso, soma 16 a r1. OBS: também há pseudo-operações de endereçamento em que, não há referência direta para um endereço numa instrução, mas sim o endereço é tido através do resultado de operação aritmética sobre o PC. **5->** Endereçamento por deslocamento. **6->** Endereçamento por registrador. **7->** Endereçamento imediato.

PIPELINE:

RISC-V: Com a CPU Pipeline têm-se um benefício notável, visto que, dinamizamos o processo ao executar várias instruções ao mesmo tempo. Diante disso, na Risc-V esse processo acontece em 5 etapas, e são elas: Etapa 1 - IFetch(Instruction Fetch): Busca a instrução na memória e copia para dentro do registrador. Etapa 2 - Reg/Dec: É feita a leitura do banco de registradores(lê os registradores referenciados pela instrução), enquanto a instrução é decodificada. Etapa 3 - Exec: Executa a operação ou calcula um endereço(usando a ALU). Etapa 4 - Acesso à memória de dados(as instruções de load e store vão estar acessando a MD nessa etapa). Etapa 5 - Escrita do resultado disso no banco de registradores

ARM: Por ser um processador pertencente à classificação RISC, nele é possível o uso do pipeline(diferentemente de máquinas do tipo CISC, nas quais é extremamente dificil o uso do pipeline, visto que, por exemplo, cada instrução ser executada à sua maneira ao invés de ser uma execução regular e as instruções possuem tamanhos variáveis, ao contrário da RISC na qual as instruções têm um tamanho fixo). Possui 3 estágios de pipeline de execução, que são: busca da informação, decodificação e execução.

CÓDIGO ABERTO OU FECHADO:

RISC-V: Código aberto, com desenvolvimento colaborativo, e graças a isso, vem surgindo múltiplas implementações distintas de RISC-V.

ARM: Código fechado, entretanto, recém aberto apenas para empresas desenvolvedoras parceiras.

Arquiteturas de multiprocessadores AMP e SMP:

Taxonomia de Flynn:

A taxonomia de Flynn é um modelo de classificação de arquitetura de computadores baseado no fluxo de instrução e dados. Este modelo é dividido em quatro categorias: SISD, SIMD, MISD e MIMD.

SISD (Single Instruction Single Data): Fluxo único de instruções sobre um único conjunto de dados;
SIMD (Single Instruction Multiple Data): Fluxo único de instruções em múltiplos conjuntos de dados;
MISD(Multiple Instruction Single Data): Fluxo múltiplo de instruções em um único conjunto de dados;

MIMD(Multiple Instruction Multiple Data): Fluxo múltiplo de instruções sobre múltiplos conjuntos de dados.

SISD - É uma máquina com um único processador capaz de executar uma única instrução, operando em um único caminho de dados. As instruções desta máquina são processadas de maneira sequencial e os computadores que adotam esse modelo são conhecidos como computadores sequenciais. Todas as instruções e dados a serem processados devem ser armazenados na memória primária. As arquiteturas SISD caracterizam-se por ter uma única unidade de controle. A velocidade de processamento é limitada. Exemplo disso são computadores pessoais com um processador convencional.

SIMD - É uma máquina multiprocessada capaz de executar a mesma instrução em todas as CPUs, mas trabalhando em diferentes fluxos de dados. Essas máquinas são bem adequadas à computação científica, pois envolvem muitas operações de vetor e array. Para que a informação possa ser passada para todos os elementos de processamento, os elementos de dados organizados dos vetores podem ser divididos em vários conjuntos e cada elemento de processamento pode processar um conjunto de dados. Exemplo de uso do SIMD são as máquinas ILLIAC IV, etc.

MISD - É uma máquina com multiprocessador capaz de executar instruções em diferentes elementos de processamento, mas todos operando no mesmo conjunto de dados. O sistema executa diferentes operações no mesmo conjunto de dados. Essas máquinas não são muito úteis na maioria das aplicações. Alguns representantes desta categoria são os servidores multiprocessadores, as redes de estações e as arquiteturas massivamente paralelas.

MIMD - É uma máquina com multiprocessadores que é capaz de executar instruções múltiplas em conjuntos de dados múltiplos. Cada elemento de processamento tem instruções e fluxos de dados separados. Portanto, as máquinas construídas com esse modelo são capazes de fazer qualquer tipo de aplicação. Ela funciona de forma assíncrona.

Essa máquina é dividida em duas categorias:

Memória Compartilhada (Sistemas Multiprocessadores fortemente acoplados) - Todos os elementos de processamento são conectados a uma única memória global e todos têm acesso a ela.

Memória Distribuída (Sistemas Multiprocessadores fracamente acoplados) - Todos os elementos de processamento tem uma memória local. A conexão entre eles neste modelo ocorre através de uma rede de interconexão.

As arquiteturas de processamento paralelo desempenham um papel fundamental na computação moderna, permitindo que os sistemas aproveitem vários processadores ou núcleos para executar tarefas de forma mais eficiente. Duas abordagens comuns para processamento paralelo são o Asymmetric Multiprocessing (AMP) e o Symmetric Multiprocessing (SMP). Este texto explora as diferenças entre essas duas arquiteturas, analisando seu desempenho, sincronismo, concorrência por recursos, comunicação entre processos e processadores e gerenciamento de recursos.

Em termos de desempenho, a arquitetura SMP geralmente supera a arquitetura AMP. A SMP permite que vários processadores acessem uma única memória compartilhada e barramento do sistema, proporcionando melhor coordenação e sincronização entre os processadores. Em contraste, a arquitetura AMP opera com elementos de processamento separados que operam independentemente, o que pode resultar em um desempenho menos otimizado. Quanto ao sincronismo, a arquitetura SMP oferece maior previsibilidade de tempo, já que é projetada para sincronizar vários processadores ou núcleos, permitindo melhor controle sobre o tempo e sincronização. A arquitetura AMP, por outro lado, opera de forma independente, o que pode resultar em comportamento imprevisível do tempo.

Em relação à concorrência por recursos, na arquitetura SMP, todos os processadores competem por recursos compartilhados, como memória e interfaces de E/S, o que pode levar à contenção de recursos e degradação do desempenho. Em contrapartida, na arquitetura AMP, cada elemento de processamento possui recursos próprios, reduzindo a competição por recursos. A comunicação entre processos e processadores na arquitetura AMP geralmente é realizada por meio de passagem de mensagens, o que pode ser menos eficiente do que a comunicação de memória compartilhada usada na arquitetura SMP. A passagem de mensagens requer sobrecarga adicional para criação, transmissão e recepção de mensagens, enquanto a memória compartilhada na arquitetura SMP fornece comunicação mais rápida e eficiente.

No que diz respeito ao gerenciamento de recursos, a arquitetura SMP requer hardware e software mais complexos do que a arquitetura AMP. A SMP exige um sistema de memória compartilhada, que requer componentes adicionais de hardware e software, como controladores de memória, caches de memória e mecanismos de sincronização. Em contraste, a arquitetura AMP é mais simples e requer componentes de hardware e software menos complexos. Em resumo, a arquitetura SMP oferece melhor desempenho e previsibilidade de tempo, mas pode resultar em contenção de recursos e requer hardware e software mais complexos. A arquitetura AMP é mais simples e tem menos concorrência por recursos, mas pode resultar em comunicação menos eficiente e menos otimização de desempenho. A escolha entre as arquiteturas AMP e SMP depende dos requisitos e restrições específicas do sistema em desenvolvimento.

REFERÊNCIAS

https://dl.acm.org/doi/pdf/10.1145/3386377?casa_token=Osy8VnsgE94AAAAA%3ACALEII_u1-hDGEGT7YMd2Y1T8KaVZjDAshkOs87pA4QNc8hpGl0drMpify-DZDOR_dWog1ML5jBL5w

https://www.cin.ufpe.br/~if674/arquivos/2019.1/Aulas/superscalar-riscV.pdf

https://www.cin.ufpe.br/~ensb/courses/public_html/slides/design-tech-ensb-arm-06-1-2.pdf

https://www.professores.uff.br/lbertini/wp-content/uploads/sites/108/2017/08/Cap-3-Conjunto-de-Instrucoes.pdf

Referências: El-Rewini, H., & Abd-El-Barr, M. (2005). Advanced Computer Architecture and Parallel Processing. John Wiley & Sons.Patterson, D. A., & Hennessy, J. L. (2017). Computer Organization and Design: The Hardware/Software Interface. Morgan Kaufmann.Tannenbaum, A. S., & Bos, H. (2015). Modern Operating Systems. Prentice Hall.

https://www.maxwell.vrac.puc-rio.br/16578/16578_4.PDF

https://acervolima.com/arquitetura-do-computador-taxonomia-de-flynn/

https://www.cin.ufpe.br/~joa/menu_options/school/cursos/ppd/aulas/flynn.pdf

https://docplayer.com.br/11804674-Introducao-as-arquiteturas-paralelas-e-taxonomia-de-flynn.html