Arquitetura de Computadores
Reduced Instruction Set Computer: RISC-V

Rodolfo Labiapari Mansur Guimarães

rodolfolabiapari@decom.ufop.br
Lattes: `http://goo.gl/MZv4Dc`
Departamento de Computação – Universidade Federal de Ouro Preto
Ouro Preto - MG – Brasil

Última Atualização: 18 de julho de 2017

Sumário

Introdução ao RISC-V

RISC-V

Um moderno e de alta qualidade *Conjunto de Instruções para Computador de Propósito Geral*, o ISA RISC-V.

- RISC-V é um ISA[1] *open-source* desenvolvido por
  - **Profissionais**, **Professores** e **Voluntários Entusiastas** no ramo de Arquiteturas de Computadores;
  - Pessoas que possuem **experiência** em várias especialidades
    - Eletrônica Digital, Compiladores, Sistemas Operacionais, Microprocessadores, entre outras como simulação e validação dos design do projeto.
  - Foi desenvolvido com base em uma série de outros projetos acadêmicos de *design* de computadores.

[1] *Instruction-Set Archtecture.*

Introdução ao RISC-V

- Foi produzido pela *Computer Science Division* na Universidade da Califórnia, Berkeley.
- É um conjunto de instruções
  - **Limpo**;
  - **Modular**
  - Inteiros com bases em 32, 64 e 128 bits e;
  - Várias opções de **extensão de instruções** como ponto-flutuante, multiplicadores, etc.

Introdução ao RISC-V

- Cada empresa/usuário espera que os *designers* considerem a **performance** e também a **eficiência energética** ao projetar um processador novo processador. Mas isso é um *trade-off*
  - Em contrapartida, RISC-V é um projeto cujas instruções são de fácil implementação comparado a outras alternativas existentes no mercado;
  - E mesmo sendo um projeto novo, possui grande aceitação na indústria de semicondutores.

Meta

Criar um conjunto de instruções 'universal', que é livre e aberto para todos os usuários, provendo tudo que é necessário para suportar perfeitamente qualquer projeto comercial.

- Não é o primeiro projeto *open-souce*
  - É modelado para ser útil desde **projetos embarcados de pequeno porte** até **servidores**, **celulares *high-end***.
  - Incluindo fazer *design*, fabricar e vender os chips e *software* RISC-V.
- O ISA RISC-V tem sido desenvolvido com o intuito de ser
  - Pequeno, rápido e de gasto mínimo de energia em implementações em especificações RISC existente hoje
  - Sem sobrecarregar outras partes do seu sistema (*trade-off*).

Introdução ao RISC-V

Comparação com outros ISAs

- Embora alguns processadores disponíveis hoje no mercado sejam amplamente utilizados, eles **são complexos** e difícil de ser utilizado para experimentação e uso acadêmico.
- Difere da maioria dos ISA pois, chips da ARM e MIPS Technologies necessitam de licença para uso de suas patentes
  - Também requerem acordos de confidencialidade para uso de seus documentos que citam as vantagens de seus *design* e conjunto de instruções.
- Sua arquitetura é mais fácil de ser implementada do que um ARM, por exemplo:
  - Pela sua simplicidade em:
    - Decodificar códigos e modos de endereçamento.
  - Omite instruções complexas;
  - Tudo isso com **metade** do tamanho de projeto de um ARM *core*.

De Projeto Acadêmico à Oracle e Google

- Iniciado em 2010;
- Disponibilizado em 2015;
- E já é utilizado pela Mellanox, Oracle e Google, além de grandes centros acadêmicos.
- Por ser *open-source*, vários projetos funcionais online já estão disponíveis, usufruindo da permissão de licença BSD
  - Um exemplo é o escalar de 5 estágios chamado RISC-V Rocket[2] em Scala e para sintetização em chips[3].
- É considerado como a técnica mais **promissora** em flexibilidade de extensões customizadas combinado com arquitetura de propósito geral.

[2] Disponível em: `https://github.com/ucb-bar/rocket`

[3] Disponível em: `https://github.com/ucb-bar/rocket-chip`

Sumário

Exemplificação

Consideremos um *smartphone* moderno *high-end*.

- Possui dúzias de *cores* com diferentes pilhas de *softwares* como
  - Uma aplicação no processador ARM;
  - Uma GPU, 3 à 5 DSP[4], um *core* de gerenciamento de energia.
  - ...
- Em teoria, todos os *cores* usam uma variante de um ISA simples e muitos casos, **reutilizando** *hardware* e *software*
  - Por exemplo, todos os cores necessitam de itens básicos que são comuns em todos eles;
  - Redundância de equipamentos/código em sistemas embarcados tal como telefones modernos não é algo inteligente.

[4] *Digital Signal Processor*

Exemplificação

Consideremos um *smartphone* moderno *high-end*.

- Possui dúzias de *cores* com diferentes pilhas de *softwares* como
  - Uma aplicação no processador ARM;
  - Uma GPU, 3 à 5 DSP[4], um *core* de gerenciamento de energia.
  - ...
- Em teoria, todos os *cores* usam uma variante de um ISA simples e muitos casos, **reutilizando** *hardware* e *software*
  - Por exemplo, todos os cores necessitam de itens básicos que são comuns em todos eles;
  - Redundância de equipamentos/código em sistemas embarcados tal como telefones modernos não é algo inteligente.
- **Redundância de equipamentos/código? Reutilização????**

[4] *Digital Signal Processor*

Exemplificação

- Sobre este aspecto, o propósito do RISC-V:

Propósito

Oferecer uma opção prática para unificar todos esses *cores* num **único local**, por ser universal para qualquer aplicação comercial.

Sumário

Propósito

- Como já dito, seu propósito é amplo ao ponto de ser suportado para executar em:
  - Microcontroladores que processem imagens, gráficos ou até mesmo processadores de servidores.

Implementação

É **consistente em microarquiteturas** e por isso foi propositalmente direcionado para ser adequado para **quase todo tipo de implementação.**

- Similarmente, é projetado para ser adequado a quase todas implementações **sintetizáveis**, desde macros sintetizações em FPGA 😄, até projetos de processadores totalmente customizados.

Propósito

- RISC-V provê garantia de funcionamento para:
  - Instruções de inteiros tamanho 32, 64 e 128 bits;
  - Além de uma família de extensões opcionais e/ou predefinidas.

Sumário

Termos Gerais

Registradores

- RISC-V possui
  - 32 registradores inteiros e;
    - Também existe uma variante de pequeno porte do RISC-V com só 16 registradores inteiros.
  - 32 registradores **opcionais** para ponto flutuante.
    - Os registradores opcionais de ponto flutuante são incluídos junto com o pacote de extensão de operações de ponto flutuante.
- Sua memória possui endereçamento de 1 byte (8 bits).

Termos Gerais

Registradores

- O *assembler* usa o registrador *r0* (que possui o valor 0) como um espaço reservado para fazer manuseio de algumas operações.
- Também possui registradores de controle e status, mas somente o nível de privilégio mais alto que poderá acessá-los para medição de performance.

  a
- Como todos os outros *design* de RISCs, RISC-V é um máquina *load-store*
  - **Somente estas duas** instruções que possuem acesso à memória principal;
  - **Todas** as operações lógico-aritméticas ocorrem **entre** registradores.

Termos Gerais

Otimizações

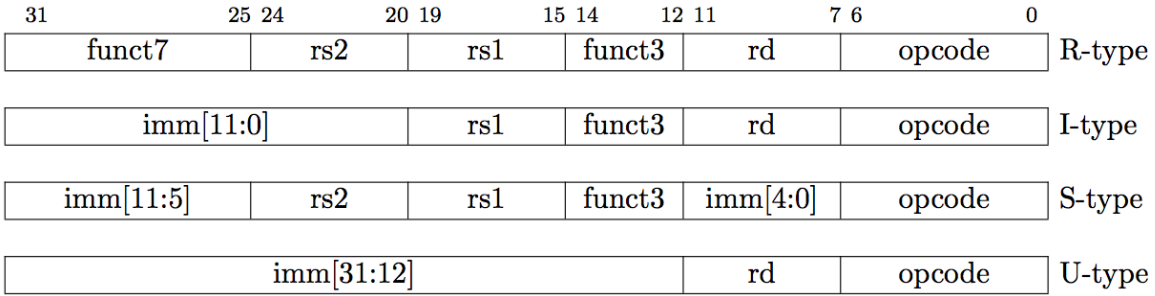
- Diferentemente de outros RISCs acadêmicos otimizados para simplicidade, o conjunto de instruções do projeto RISC-V foi desenvolvido com foco na **praticidade das implementações**
  - Com características que **aumentam a velocidade computacional** enquanto **reduz** seu **custo** e **energia**.

- Nele, várias otimizações foram feitas como
  - Colocar os bits mais significantes numa posição fixa;
  - A fixação de bits de operações para reduzir o número de multiplexadores no chip em sua implementação.

Termos Gerais

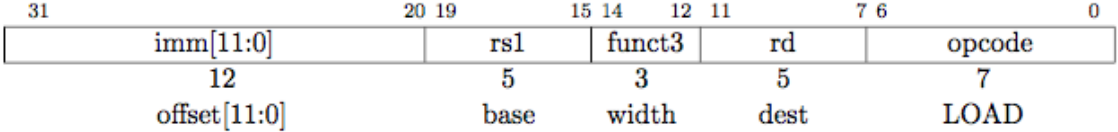
Instruções

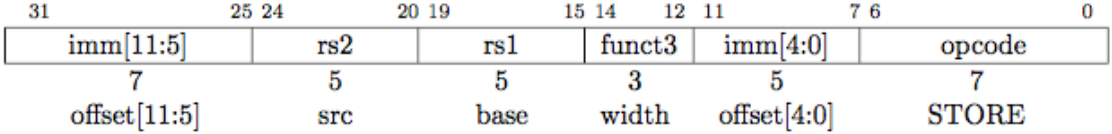
- Todas suas instruções possuem 32 bits.



Termos Gerais

Load e Store





Termos Gerais

Exemplo de um Register-Register

31 25	24 20	19 15	14 12	11 7	6 0
funct7	rs2	rs1	funct3	rd	opcode
7	5	5	3	5	7
0000000	src2	src1	ADD/SLT/SLTU	dest	OP
0000000	src2	src1	AND/OR/XOR	dest	OP
0000000	src2	src1	SLL/SRL	dest	OP
0100000	src2	src1	SUB/SRA	dest	OP

- **Opcode:** Local onde fica a operação principal;
- **Funct3:** Especificação da operação;
- **Funct7:** Especificação da operação;

Termos Gerais

Demais Instruções *Load* e *Store*

Pseudoinstruction	Base Instruction(s)	Meaning
`la rd, symbol`	`auipc rd, symbol[31:12]` `addi rd, rd, symbol[11:0]`	Load address
`l{b|h|w|d} rd, symbol`	`auipc rd, symbol[31:12]` `l{b|h|w|d} rd, symbol[11:0](rd)`	Load global
`s{b|h|w|d} rd, symbol, rt`	`auipc rt, symbol[31:12]` `s{b|h|w|d} rd, symbol[11:0](rt)`	Store global
`fl{w|d} rd, symbol, rt`	`auipc rt, symbol[31:12]` `fl{w|d} rd, symbol[11:0](rt)`	Floating-point load global
`fs{w|d} rd, symbol, rt`	`auipc rt, symbol[31:12]` `fs{w|d} rd, symbol[11:0](rt)`	Floating-point store global
`nop`	`addi x0, x0, 0`	No operation
`li rd, immediate`	*Myriad sequences*	Load immediate
`mv rd, rs`	`addi rd, rs, 0`	Copy register
`not rd, rs`	`xori rd, rs, -1`	One's complement
`neg rd, rs`	`sub rd, x0, rs`	Two's complement
`negw rd, rs`	`subw rd, x0, rs`	Two's complement word

Termos Gerais

Diferenças

- RISC-V, **intencionalmente**, **não possui**:
  - **Bit de *carry*** em algumas operações aritméticas
    - Não possui *carry* de operações aritméticas complicadas (multiplicação e divisão);
    - Não tem detector ou *flag* para erros aritméticos, incluindo *overflow*, *underflow* e divisão por 0.
  - Vários Condicionais
    - utiliza o `jal` (*jump and link*) que armazena o endereço em um registrador para o retorno.
- Os projetistas dizem afirmam que isso pode simplificar o desenvolvimento do CPU, minimizando interações entre instruções;

Sumário

Modulação

- Por causa do crescente interesse em aceleração em processamento, a **extensibilidade** do RISC-V é parte essencial da sua **universalidade**
  - **Pode-se adicionar módulos**, criando uma arquitetura que atende a todos os projetos de *hardware*.

- Para ativar as extensões, porções do espaço de codificação de instruções já foram reservados para uso futuro
  - Ou seja, é só adicionar a nova extensão e usar.

Base ISA	Instructions	Description
RV32I	47	32-bit address space and integer instructions
RV32E	47	Subset of RV32I, restricted to 16 registers
RV64I	59	64-bit address space and integer instructions, along with several 32-bit integer instructions
RV128I	71	128-bit address space and integer instructions, along with several 64- and 32-bit instructions

Extension	Instructions	Description
M	8	Integer multiply and divide
A	11	Atomic memory operations, load-reserve/store conditional
F	26	Single-precision (32 bit) floating point
D	26	Double-precision (64 bit) floating point; requires F extension
Q	26	Quad-precision (128 bit) floating point; requires F and D extensions
C	46	Compressed integer instructions; reduces size to 16 bits

Base ISA	Instructions	Description
RV32I	47	32-bit address space and integer instructions
RV32E	47	Subset of RV32I, restricted to 16 registers
RV64I	59	64-bit address space and integer instructions, along with several 32-bit integer instructions
RV128I	71	128-bit address space and integer instructions, along with several 64- and 32-bit instructions

- O modelo de programação **RV32I** é escasso.
  - *Program Counter* e 32 registradores inteiros (`x0` - `x31`).
  - Não contém registrador de retorno, sendo este o `x1`.
- Várias combinações de imediatos, operandos e outra especificações
  - O `opcode` e operando ficam em locais fixos, facilitando o processo de decodificação.
- As outras bases não alteram nada.

Modelo de Memória

- Sua memória é endereçada por *byte* e utiliza-se *little-endian*[5].
- Enquanto outros processadores utilizam complexas instruções e vários modos de endereçamento, RISC-V usa somente *base+offset*.
- Podem operar em dados de tamanho 8 (*byte*), 16 (*half-word*), 32 (*word*) ou 64 (*double*) bits além das opções de sinalização.

[5] *You store the **least** significant byte in the smallest address.*

Sumário

Variantes e Extensões do RISC-V

Variantes: RV32I, RV32E, RV64I e RV128I

- Todas as variantes do RISC-V descritas a seguir são baseadas do **RV32I** (já explicado).

- **RV32E**:
  - Satisfatório para implementações de **pequeno porte**.
  - Sendo um projeto menor, ocupa-se uma pequena porção de área *die* e com isso
    - Aumenta o custo/benefício de produção de chips;
    - Melhora a dissipação de calor por se tratar de um chip com menos área de circuito;

Resumindo

Redução de 32 registradores para somente o *Program Counter* e `x0` - `x15`.

RV64I e **RV128I**:

- Simples extensões de 64 e 128 bits.
- Diferença
  - Aumentam o espaço de endereço e;
  - Estende os registradores de 32 para o tamanho apropriado.
- Essa extensão introduz novas instruções que operam com dados menores como 32 e 64 bits, de acordo com a base.

Extension	Instructions	Description
M	8	Integer multiply and divide
A	11	Atomic memory operations, load-reserve/store conditional
F	26	Single-precision (32 bit) floating point
D	26	Double-precision (64 bit) floating point; requires F extension
Q	26	Quad-precision (128 bit) floating point; requires F and D extensions
C	46	Compressed integer instructions; reduces size to 16 bits

- **Multiplicação M**:
  - Adiciona 4 instruções de multiplicação, duas de divisão, e duas de manipulação de restos.
  - A base é indicada pela base do ISA.
  - A divisão por 0, por exemplo, não causa interrupção, mantendo a simplicidade.

Extension	Instructions	Description
M	8	Integer multiply and divide
A	11	Atomic memory operations, load-reserve/store conditional
F	26	Single-precision (32 bit) floating point
D	26	Double-precision (64 bit) floating point; requires F extension
Q	26	Quad-precision (128 bit) floating point; requires F and D extensions
C	46	Compressed integer instructions; reduces size to 16 bits

- **Sincronização A**:
  - Adiciona 11 instruções de sincronização visando consistência e atomicidade da operação.
  - São divididos em dois grupos: *Load* Reserved e *Store* Conditional para operações atômicas na memória.

Extension	Instructions	Description
M	8	Integer multiply and divide
A	11	Atomic memory operations, load-reserve/store conditional
F	26	Single-precision (32 bit) floating point
D	26	Double-precision (64 bit) floating point; requires F extension
Q	26	Quad-precision (128 bit) floating point; requires F and D extensions
C	46	Compressed integer instructions; reduces size to 16 bits

- Precisão simples (**F**), Precisão Dupla (**D**) e quádrupla precisão (**Q**)
  - Uma é pré-requisito de outra.
- Introduz também 32 regs. de ponto flutuante (`f0` - `f31`) de 32 bits.
- Registrador de 5 bits para exceções
  - Exceções não geram interrupções.
- As instruções *load-store* usam o mesmo endereçamento *base+offset*.

Extension	Instructions	Description
M	8	Integer multiply and divide
A	11	Atomic memory operations, load-reserve/store conditional
F	26	Single-precision (32 bit) floating point
D	26	Double-precision (64 bit) floating point; requires F extension
Q	26	Quad-precision (128 bit) floating point; requires F and D extensions
C	46	Compressed integer instructions; reduces size to 16 bits

- Não adiciona nenhum outra função. Ao invés disso, **codifica** as instruções inteiras para **salvar espaço** e com isso reduzir o tamanho do *footprint*.
- É disponível para bases inteiras.
- Cria-se instruções compressadas em 16 bit
  - Basicamente, cada função compactada é mapeada diretamente à instrução real.
- Visa sistemas embarcados.

Vantagens

- Resumindo:
  - Sua área *die* é menor;
  - Permite a adição de extensões customizadas.
  - É um projeto *open-source*.

Sumário

Hardware + Software

- Um **Linux** 4.1 foi portado para o projeto RISC-V
  - E um Linux Embarcado (**Yocto**) também foi disponibilizado.

- Várias ferramentas do RISC-V incluem **compiladores** GCC, LLVM e Clang, GDB, uma suite de **verificação** e **simuladores**.

Sumário

Adotantes do Projeto

- Um número de organizações comerciais planejam suportar o RISC-V Foundation
  - Bluespec, Inc.;
  - **Google**;
  - Hewlett Packard Enterprise (**HP**);
  - **Lattice** Semiconductor;
  - Mellanox Technologies;
  - Microsemi;
  - **Oracle**; e
  - Rambus Cryptography Research.

Adotantes do Projeto

- O Instituto Indiano de Tecnologia Madras estão **desenvolvendo 6 RISC-V** *open-source* para 6 tipos de usuários diferentes.
  - De um pequeno CPU 32 bit para IoT até largos computadores de 64 bit concebido para computadores em escala de de armazenamento.

- lowRISC é um projeto sem fins lucrativos que visa **implementar um sistema *open-source* num *System on Chip* (SoC)** baseado num RISC-V de 64 bits.

- O Laboratório de Computação, na Universidade de Cambridge, em colaboração com o FreeBSD Project, tem suportado o sistema operacional **FreeBSD para o RISC-V** 64-bit.

Adotantes do Projeto

- ETH Zurich e a Universidade de Bologna têm desenvolvido em cooperação um **System on Chip (SoC) de baixa energia** usando RISC-V.
- Um dos fundadores do Adapteva planeja usar **RISC-V como sucessor para o seu produto acelerador multicore**.
- Nvidia planeja **usá-lo** para substituir a linha de processadores Falcon **em suas placas gráficas GeForce**.

Electronics Weekly.com

NEWS | BUSINESS | MARKETS | DESIGN | PRODUCTS | BLOGS | EW COMP

Home » Blogs » Mannerisms » Dilemmas

Add to Bookmarks

By David Manners 28th November 2016

Mannerisms

Samsung Defection From ARM to RISC-V.

Could Samsung be the first big defection from ARM since the SoftBank takeover?

RECOMMENDED ARTICLES

It was always thought that, when ARM relinquished its independence, its customers would look around for other alternatives

- Porque continuar utilizando uma produto de uma empresa fechada como a ARM sendo que RISC-V é **independente**, ***open-source*** e de **direitos livres**?

- Porque continuar utilizando uma produto de uma empresa fechada como a ARM sendo que RISC-V é **independente**, ***open-source*** e de **direitos livres**?
- **Problema:** É quase impossível substituir uma arquitetura de processador em uma grande área de produto.

- Porque continuar utilizando uma produto de uma empresa fechada como a ARM sendo que RISC-V é **independente**, ***open-source*** e de **direitos livres**?
- **Problema:** É quase impossível substituir uma arquitetura de processador em uma grande área de produto.
  - No momento não há arquitetura de processador em IoT.
- Nvidia e Qualcomm já estão usando RISC-V no desenvolvimento de controladores de memória GPU e processadores IoT.

Dúvidas, Sugestões ou Reclamações?

- rodolfolabiapari@decom.ufop.br
- https://www.guerrillamail.com/

Arquitetura de Computadores
Reduced Instruction Set Computer: RISC-V

Rodolfo Labiapari Mansur Guimarães

rodolfolabiapari@decom.ufop.br
Lattes: `http://goo.gl/MZv4Dc`
Departamento de Computação – Universidade Federal de Ouro Preto
Ouro Preto - MG – Brasil

Última Atualização: 18 de julho de 2017