RISC-V

MO601 - Arquitetura de Computadores II

http://www.ic.unicamp.br/~rodolfo/mo601

Rodolfo Azevedo - rodolfo@ic.unicamp.br

Conjuntos de Instruções do Processador - ISA

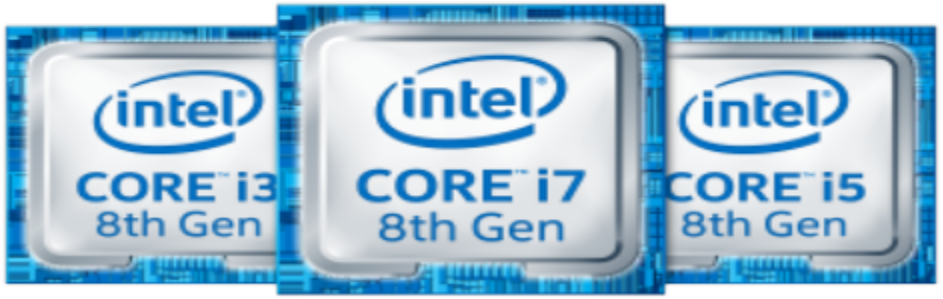
- *"The portion of the computer that is visible to the programmer or the compiler writer."* - Computer Architecture: A quantitative approach
- *"An instruction set architecture (ISA) is an abstract model of a computer. It is also referred to as architecture or computer arquiteture."* - Wikipedia
- *"A contract HW and SW designers agreed to obey"* - Minha definição de uma linha
- **"Um contrato em que os projetistas de hardware e software concordaram em obedecer"** - Minha definição de uma linha

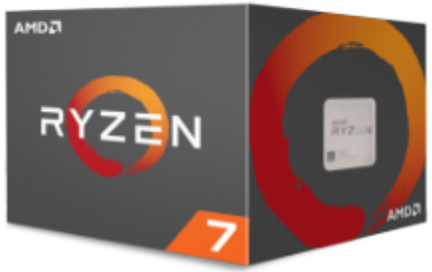
Arquitetura vs Microarquitetura

- Arquitetura é o modelo
  - x86, ARM, RISC-V, Power
- Microarquitetura é a implementação
  - Intel i7 geração 11, AMD Ryzen 3, ARM Cortex-A53, RISC-V RV32IMAC, PowerPC 970
- Conjunto de instruções pode ser visto como a borda
  - Pode facilitar ou dificultar a implementação em cada um dos lados

ISA é importante?

Instruction Set Architecture





O que é?

- Conjunto de instruções aberto
  - Sem proteção de patentes
  - Permite implementações independentes
- ISA modularizado
  - Apenas os sub-conjuntos de instruções necessários precisam ser implementados
  - Pacote mínimo de 47 instruções
- Espaços de endereçamento de 32, 64 e 128 bits
- Registradores de 32, 64 e 128 bits
- ISA totalmente virtualizável

Características

- 32 registradores de inteiros
  - Registrador 0 tem valor fixo em 0
  - Registrador 1 é o endereço de retorno
- Opcionalmente 32 registradores de ponto flutuante conforme IEEE 754-2008
- Memória endereçada em bytes e instruções alinhadas em 32 bits de memória (4 bytes)
- Arquitetura load-store
- Poucos formatos de instruções

Extensões

Extensão	Característica
RV32I	Conjunto base de instruções de inteiros de 32 bits
RV64I	Conjunto base de instruções de inteiros de 64 bits
M	Instruções de multiplicação e divisão de inteiros
A	Instruções de operações atômicas
F e D	Instruções de ponto flutuante de precisão simples e dupla
G	Equivalente a IMAFD
C	Instruções compactas

Ambientes de Execução

Tipo	Sistema Operacional	Acesso aos Periféricos	Ambiente	Exemplo
Bare metal	Não	Direto	Memória	Arduino
Sistema Operacional	Sim	Indireto	Processo	Windows, Linux, iOS, Android
Hypervisor	Sim	Indireto	Máquina Virtual	VirtualBox, VMware, QEMU
Emulador	Sim	Indireto	Processo	MARS, QEMU

Codificação das Instruções

Formato	Bits																													
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	func7							rs2					rs1					func3			rd					opcode						
I	imm[11:0]												rs1					func3			rd					opcode						
S	imm[11:5]							rs2					rs1					func3			imm[4:0]					opcode						
SB	[12]	imm[10:5]						rs2					rs1					func3			Imm[4:1]				[11]	opcode						
U	imm[31:12]																				rd					opcode						
UJ	[20]	imm[10:1]										[11]	imm[19:12]								rd					opcode						

Instruções RV32I - parte 1

Categoria	Instrução	Formato	RV32I	
Loads	Load Byte	I	LB	rd, rs1, imm
	Load Halfword	I	LH	rd, rs1, imm
	Load Word	I	LW	rd, rs1, imm
	Load Byte Unsigned	I	LBU	rd, rs1, imm
	Load Half Unsigned	I	LHU	rd, rs1, imm
Stores	Store Byte	S	SB	rs1, rs2, imm
	Store Halfword	S	SH	rs1, rs2, imm
	Store Word	S	SW	rs1, rs2, imm
Shifts	Shift Left	R	SLL	rd, rs1, rs2
	Shift Left Immediate	I	SLLI	rd, rs1, shamt
	Shift Right	R	SRL	rd, rs1, rs2
	Shift Right Immediate	I	SRLI	rd, rs1, shamt
	Shift Right Arithmetic	R	SRA	rd, rs1, rs2
	Shift Right Arith Imm	I	SRAI	rd, rs1, shamt
Arithmetic	ADD	R	ADD	rd, rs1, rs2
	ADD Immediate	I	ADDI	rd, rs1, imm
	SUBtract	R	SUB	rd, rs1, rs2
	Load Upper Imm	U	LUI	rd, imm
	Add Upper Imm to PC	U	AUIPC	rd, imm
Logical	XOR	R	XOR	rd, rs1, rs2
	XOR Immediate	I	XORI	rd, rs1, imm
	OR	R	OR	rd, rs1, rs2
	OR Immediate	I	ORI	rd, rs1, imm
	AND	R	AND	rd, rs1, rs2
	AND Immediate	I	ANDI	rd, rs1, imm

Instruções RV32I - parte 2

Categoria	Instrução	Formato	RV32I	
Compare	Set <	R	SLT	rd, rs1, rs2
	Set < Immediate	I	SLTI	rd, rs1, imm
	Set < Unsigned	R	SLTU	rd, rs1, rs2
	Set < Imm Unsigned	I	SLTIU	rd, rs1, imm
Branches	Branch =	SB	BEQ	rs1, rs2, imm
	Branch !=	SB	BNE	rs1, rs2, imm
	Branch <	SB	BLT	rs1, rs2, imm
	Branch >=	SB	BGE	rs1, rs2, imm
	Branch < Unsigned	SB	BLTU	rs1, rs2, imm
	Branch >= Unsigned	SB	BGEU	rs1, rs2, imm
Jump & Link	J & L	UJ	JAL	rd, imm
	J & Link Register	UJ	JALR	rd, rs1, imm
Synch	Synch thread	I	FENCE	
	Sync Instr & Data	I	FENCE.I	
System	System CALL	I	SCALL	
	System BREAK	I	SBREAK	
Counters	ReaD CYCLE	I	RDCYCLE	rd
	ReaD CYCLE upper Half	I	RDCYCLEH	rd
	ReaD TIME	I	RDTIME	rd
	ReaD TIME upper Half	I	RDTIMEH	rd
	ReaD INSTR RETired	I	RDINSTRRET	rd
	ReaD INSTR upper Half	I	RDINSTRETH	rd

Instruções RV32M

Categoria	Instrução	Formato	RV32M	
Multiply	MULtiply	R	MUL	rd, rs1, rs2
	MULtiply upper Half	R	MULH	rd, rs1, rs2
	MULtiply Half Sign/Uns	R	MULHSU	rd, rs1, rs2
	MULtiply upper Half Uns	R	MULHU	rd, rs1, rs2
Divide	DIVide	R	DIV	rd, rs1, rs2
	DIVide Unsigned	R	DIVU	rd, rs1, rs2
Remainder	REMainder	R	REM	rd, rs1, rs2
	REMainder Unsigned	R	REMU	rd, rs1, rs2

Registradores

- O processador RISC-V tem 32 registradores conforme tabela abaixo

Register	ABI Name	Description	Saver
x0	Zero	Always zero	
x1	ra	Return addres	Caller
x2	sp	Stack pointer	Callee
x3	gp	Global pointer	
x4	tp	Thread pointer	
x5	t0	Temporary / alternate return address	Caller
x6–7	t1–2	Temporary	Caller
x8	s0/fp	Saved register / frame pointer	Callee
x9	s1	Saved register	Callee
x10–11	a0–1	Function argument / return value	Caller
x12–17	a2–7	Function argument	Caller
x18–27	s2–11	Saved register	Callee
x28–31	t3–6	Temporary	Caller