Arquitetura de Computadores

Instruction Set Architecture (ISA)

Bib: B – secções A.1 a A.6

João Pedro Patriarca (jpatri@cc.isel.ipl.pt), Gabinete F.0.23 do edifício F
ISEL, ADEETC, LEIC

ISA – Visão geral

- O que é o ISA?
  - Significa *Instruction Set Architecture* e corresponde a uma codificação estruturada das instruções
- Distinção entre arquiteturas RISC e CISC no que diz respeito ao ISA
  - CISC (*Complex Instruction Set Computer*): instruções com dimensões diferentes
  - RISC (*Reduced Intruction Set Computer*): instruções com a mesma dimensão
- Principais aspetos que caracterizam um ISA
  - Classes de instruções
  - Tipos de operandos
  - Modos de endereçamento

Arquiteturas RISC versus CISC

RISC	CISC
Instruções simples, tipicamente, uma instrução corresponde a uma única ação	Instruções complexas com o objetivo de reduzir o número de instruções na tradução de um algoritmo
O mesmo algoritmo é traduzido com menos instruções no âmbito de uma arquitetura CISC do que no âmbito de uma arquitetura RISC	
Instruções mais simples, menor número de ciclos de *clock* para executar uma instrução, tipicamente, com as mesmas temporizações para todas as instruções	Instruções mais complexas, maior número de ciclos de *clock* para executar uma instrução, tipicamente com temporizações diferentes de instrução para instrução
As arquiteturas RISC simplificam a introdução do conceito Pipeline na arquitetura do processador	

Classes de instruções

- Instruções de transferência de dados
  - Instruções que envolvem acessos a memória

IA32	Keil535	P16
`mov (8), ax` `mov al, (ebx)`	`mov 16, a` `mov a, @R0`	`str r0, [r3]` `ldr r5, [r2]`

- Instruções de processamento de dados
  - Instruções para realizar cálculo
  - A execução destas instruções envolve operações aritméticas e lógicas (ALU)

IA32	Keil535	P16
`add (8), ax` `and al, (ebx)`	`addc a, r4` `xrl a, (50)`	`sub r0, r1, r5` `lsl r3, r1, 5`

- Instruções de controlo de fluxo
  - Instruções de decisão e de ciclo implementadas com base nestas instruções
  - A maioria da execução destas instruções baseia-se nas *flags* produzidas pela ALU

IA32	Keil535	P16
jge label jne label	jnc label jz label	bge label bzs label

Tipos de operandos

- Registos
- Imediato
- Memória

	IA32	Keil535	P16
Registos	`inc ecx` `mov dx, ax`	`mov a, r5` `mov r3, a`	`mov r0, r1` `cmp r4, r6`
Imediato	`mov ax, 15`	`mov a, #0x30`	`mov r0, #15`
Memória	`dec (ebx)` `mov (16), ah`	`xrl a, 0x30` `mov 16, a`	`str r0, [r3]` `ldr r5, [r2]`

- Na mesma instrução podem aparecer operandos de diferentes tipos

	IA32	Keil535
Memória + imediato	`mov (eax), 0x100`	`mov 30, #10`

- Nas arquiteturas RISC, operandos do tipo Memória apenas podem aparecer em instruções da classe transferência de dados, ou seja, em instruções LDR e STR

Modos de endereçamento

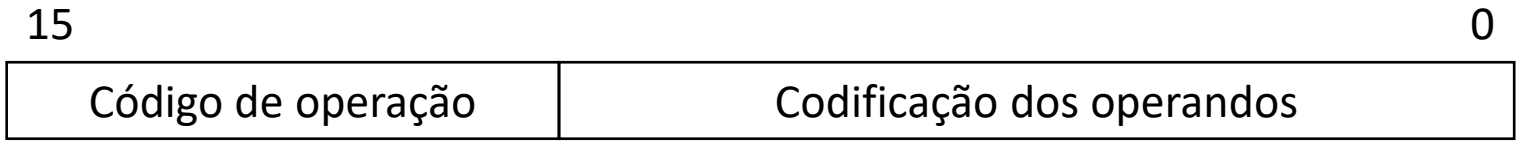
- Direto: o endereço é codificado diretamente na instrução
- Indireto: o endereço é especificado através do valor de um registo
- Baseado e indexado: o endereço é definido por um endereço base especificado por um registo e por um índice que pode corresponder ao valor de um registo ou a um valor imediato

	IA32	Keil535	P16
Direto	`mov ax, (100)`	`mov a, 100`	`--`
Indireto	`mov ax, (ecx)`	`mov a, @r1`	`ldr r0, [r1]`
Baseado e indexado	`mov ax, (ecx, 10)` `mov ax, (ecx, ebx)`	`movc a, @a+pc`	`str r0, [r3, r1]` `ldr r0, [r3, 6]`

- Na arquitetura ARM (e P16) não existe endereçamento direto

P16 – características principais

- Todas as instruções são codificadas com o mesmo número de bits
- O código de cada instrução ocupa uma única palavra de memória
- Cada instrução é codificada com 16 bits
- Um subconjunto dos bits servem para identificar univocamente a instrução (código de operação); os restantes bits servem para codificar operandos



- As classes de instruções dividem-se em Transferência de dados, Processamento de dados e Controlo de fluxo
- Não dispõe de endereçamento direto
- *Register File* constituído por 16 registos (R0 a R15) sendo cada registo constituído por 16 bits
- ALU realiza operações de 16 bits

Quis 2 – ISA

Access code: CZ3SH