Nível da Arquitetura do Conjunto das Instruções

(Aula 11)

Visão Geral do Nível ISA

Antes de deixarmos o hardware (1)

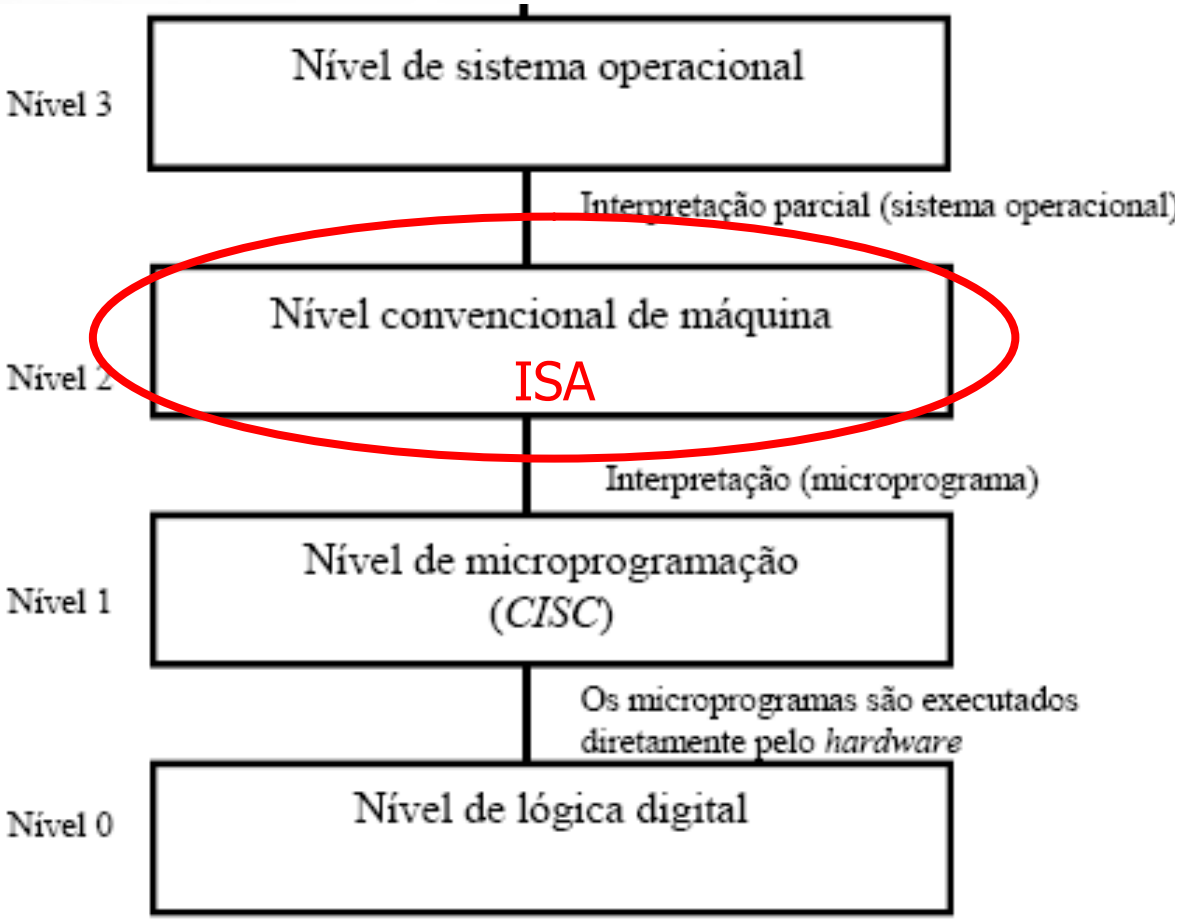
Ano	Chip	Largura do barramento	Velocidade do clock	Transistores
1971	4004	4 bits	740KHz	2300
1974	8080	8 bits	2 MHz	6.000
1979	8088	16 bits	Até 8 MHz	29.000
1982	80286	16 bits	Até 12 MHz	134.000
1985	80386	32 bits	Até 33 MHz	275.000
1989	Intel 486	32 bits	Até 100 MHz	1.600.000
1993	Pentium (original)	64 bits	Até 200 MHz	3,3 milhões
1998	Pentium II	64 bits	233 MHz	7,5 milhões
	Pentium III	64 bits	Até 1 GHz	9,5 milhões
	Pentium IV	64 bits	Até 3,8 GHz	55,0 milhões
2002	Intel® Itanium® processor			220,000,000
2003	Intel® Itanium® 2 processor			410,000,000

Antes de deixarmos o hardware (2)

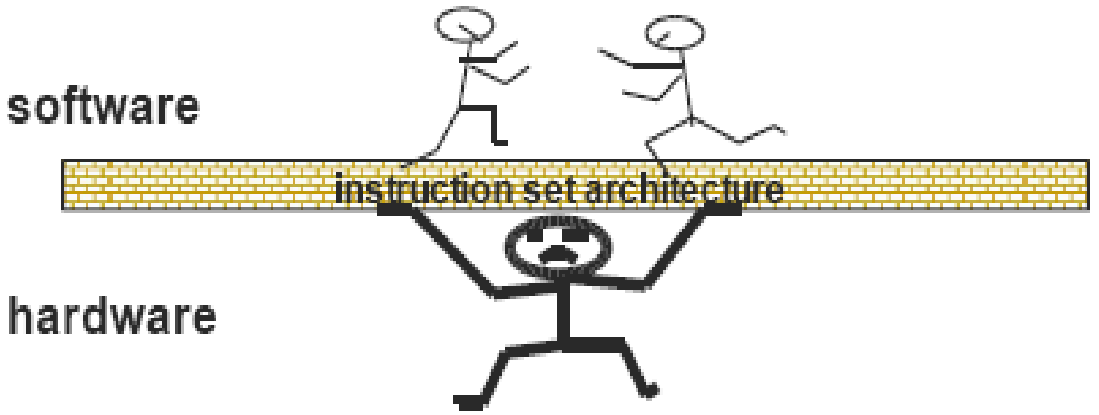
Chip	ALU	reg	dado	end.	Cache	Características
Pentium III	32	32	64	36	16K instr. 16K dado cache nív.2	Pentium II com instruções extras para paralelismo de ponto flutuante
Pentium 4	32	32	64	36	12K instr. 8K dado cache nív.2	Pentium III com instruções extras para paralelismo
Pentium4 HT	32	32	64	36	12K instr 16K dado cache niv.2	Pentium 4 com unidades de execução duplicadas

ISA (*Instruction Set Architecture*)

- Arquitetura do Conjunto de Instruções (*Instruction Set Architecture*)
- Nível Convencional de Máquina



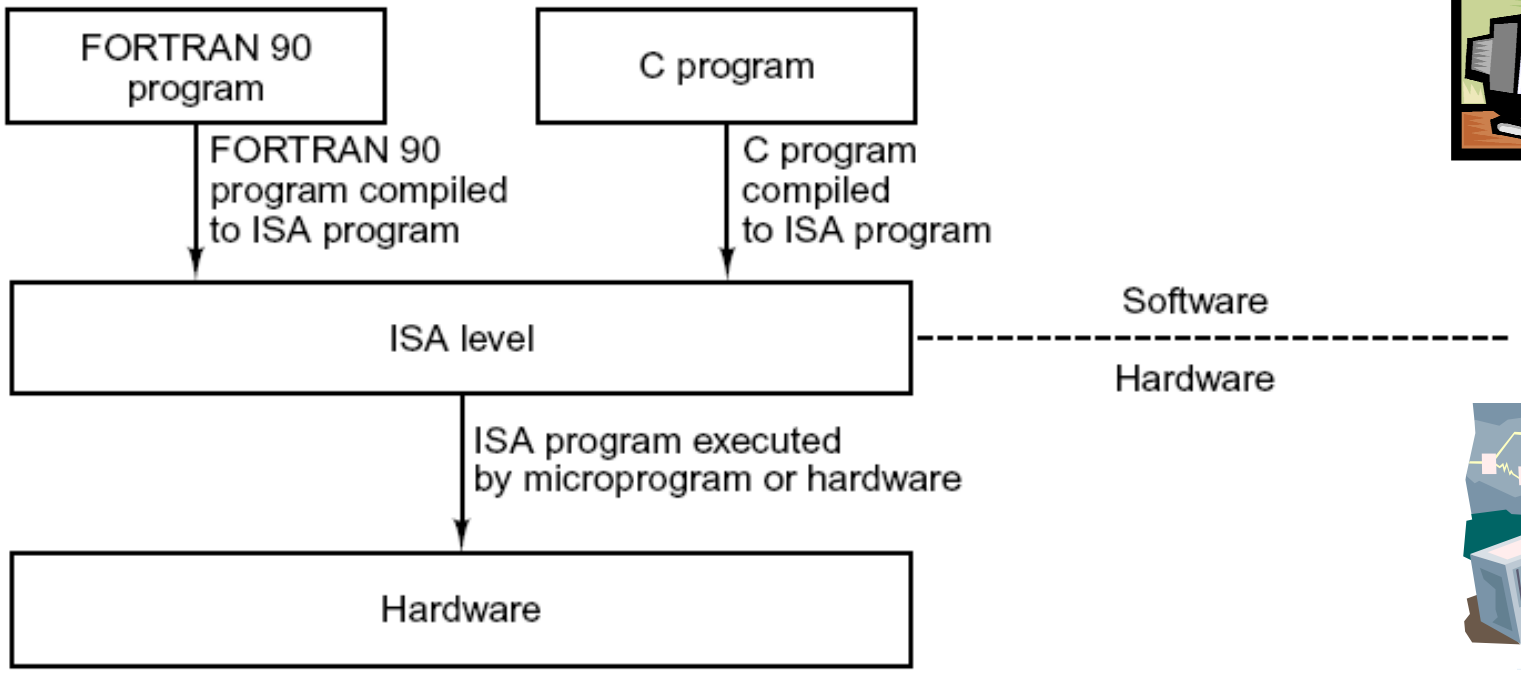
ISA Definição



- A parte do computador visível ao programador ou ao implementador de compiladores
- Interface entre o Hardware e o Software

ISA (*Instruction Set Architecture*) (1)

- Metodologia empregada nos sistemas computacionais:
  - Programas em diversas linguagens de alto nível são traduzidos para uma mesma linguagem intermediária (nível ISA) para serem então executados em um hardware construído para executar diretamente instruções do nível ISA.



ISA (*Instruction Set Architecture*) (2)

- O nível ISA define o aspecto da máquina para um programador da linguagem de máquina.
  - Quantas pessoas programam diretamente em linguagem de máquina atualmente?
- As instruções do nível ISA são aquelas para as quais o compilador deve gerar código
- Para tal, o projetista de compiladores deve conhecer um conjunto de informações que definem o nível ISA.
- Grande preocupação com compatibilidade
  - Novas máquinas devem ser compatíveis com as anteriores
  - Suportar os mesmos sistemas operacionais e programas de aplicação
- Em algumas arquiteturas, o nível ISA é especificado por meio de um documento formal (com seções normativas e informativas)

Se duas máquinas têm a mesma ISA, qual das seguintes quantidades é sempre idêntica?

freqüência de clock, CPI, tempo de execução, número de instruções por programa, MIPS

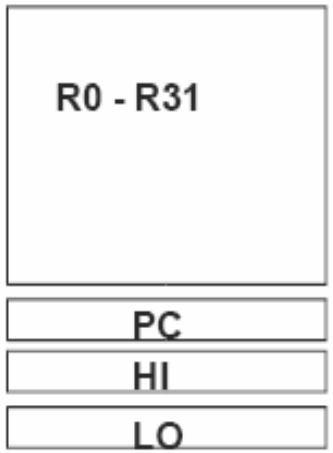
Exemplo: ISA do MIPS (R3000)

- *Microprocessor without Interlocked Pipeline Stages*
  - Não confundir com *Million Instructions per Second*
- Arquitetura RISC
- Nos anos 90, 1 em cada 3 microprocessadores de arquitetura RISC eram MIPS

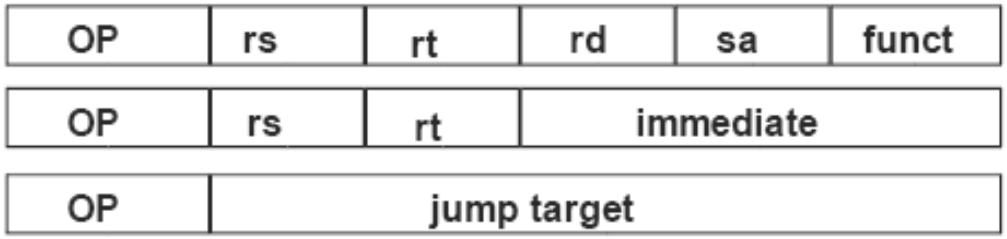
Categorias de instruções

- Load/Store
- Computacionais
- Desvio
- Ponto flutuante
  - co-processador
- Gerenciamento de memória
- Especiais

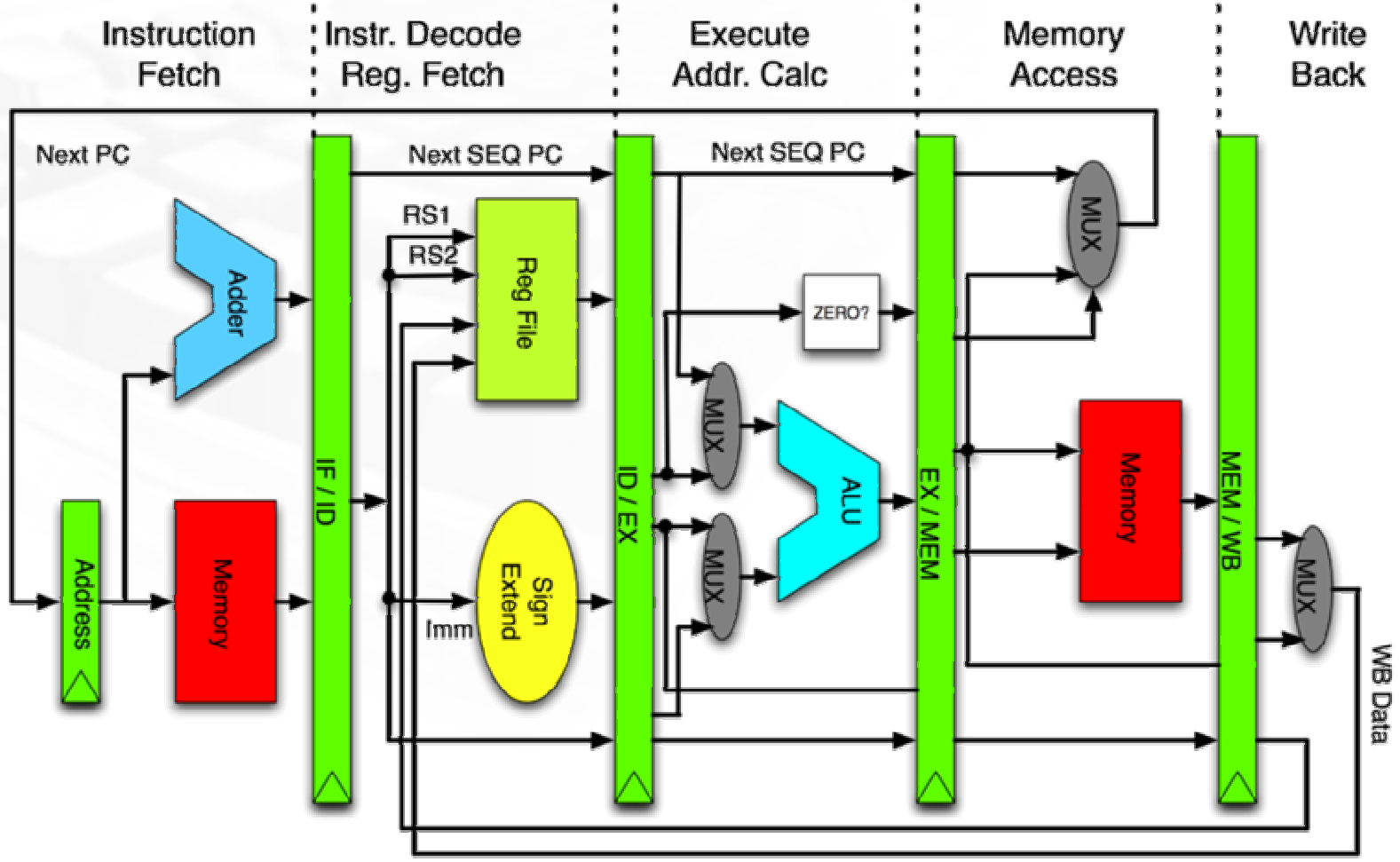
Registradores



3 Formatos de Instruções: todos com largura de 32 bits



Exemplo: ISA do MIPS (R3000): Descendo o Nível



Modos de Execução

- Na maioria das máquinas há, no mínimo, dois modos.
- Pode-se dizer que são *níveis de visibilidade do ISA*.

- **Modo *Kernel* (Modo Supervisor)**
  - Usado para executar o sistema operacional e permite que qualquer instrução da máquina seja executada nele.
- **Modo Usuário**
  - Usado para executar programas de aplicação e não permite que certas instruções mais sensíveis executem nele.
    - Instruções de manipulação de *cache* não podem ser executadas no modo usuário.

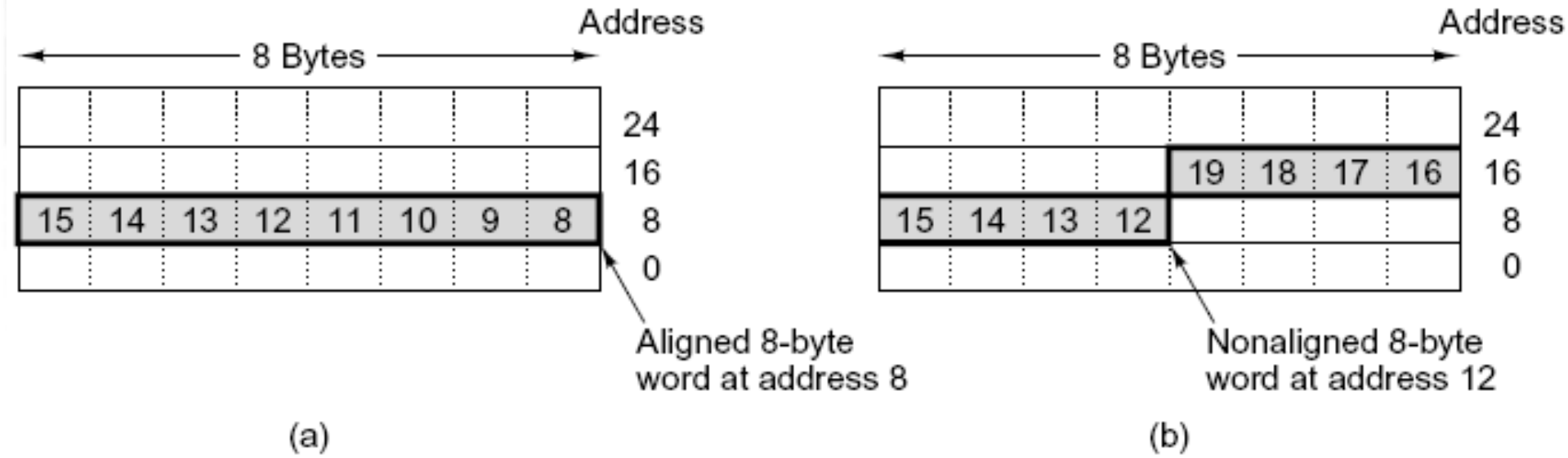
Componentes o nível ISA

- Modelo de Memória da Máquina;
- Conjunto de registradores
- Quais instruções são disponíveis e o que cada uma executa;
- Tipos de Dados utilizados;
- etc.

Modelos de Memória (1)

- Todos os computadores dividem suas memórias em um conjunto de células consecutivas.
  - Em geral, as células são definidas em 8 bits (1 byte).
  - Os bytes são agrupados em palavras de 4 bytes (32 bits) ou de 8 bytes (64 bits).
- Muitas arquiteturas exigem que as palavras estejam sempre alinhadas em suas fronteiras naturais.
  - Por exemplo, uma palavra de 4 bytes só pode começar nos endereços 0, 4, 8, ..., e não em endereços como 1 ou 2 (não múltiplos de 4).
  - Em geral, as memórias operam com mais eficiência quando as informações estão alinhadas.

Modelos de Memória (2)



"Em geral, as memórias operam com mais eficiência quando as informações estão alinhadas"... Por quê?

O acesso a memórias alinhadas são mais eficientes.
A leitura de um endereço arbitrário requer um hardware mais sofisticado e é mais caro

Modelos de Memória (3)

- A maioria das máquinas tem um único espaço de **endereçamento linear** no nível ISA
  - Do endereço 0 até um valor máximo (ex: 2^{32} bytes)
- Apesar disso, algumas máquinas implementam o conceito de espaço de endereçamento separados para **instruções** e **dados**.
  - A busca de uma instrução no endereço 8 acessa um espaço de endereços diferente daquele que é acessado quando ocorre a busca de um dado no endereço 8.
    - Com isso é possível endereçar 2^{32} bytes de código mais 2^{32} bytes de dados
  - Elimina uma fonte de *bugs*, quando uma escrita em um dado fica impedida de alterar acidentalmente uma instrução de programa.

Componentes o nível ISA

- Modelo de Memória da Máquina;
- Conjunto de registradores
- Quais instruções são disponíveis e o que cada uma executa;
- Tipos de Dados utilizados;
- etc.

Registradores (1)

- Todos os processadores têm um conjunto de registradores **visíveis** no nível ISA.
  - TOS e MAR são visíveis na microarquitetura, mas NÃO no ISA.
  - PC e SP são visíveis em ambos os níveis.
  - Os registradores visíveis no ISA são sempre visíveis na microarquitetura, pois é lá que eles são **implementados**.
- Os registradores existem para controlar a execução do programa, para guardar resultados temporários e para vários outros propósitos.

Registradores (2)

- **Registradores de Propósito Específico**
  - Exemplo: **PC** e **SP**. O que eles fazem?
  - Cumprem funções específicas
  - Alguns só são usados pelo processador
    - Isto é, o programador não consegue diretamente alterar seus valores
- **Registradores de Propósito Geral**
  - Armazenam variáveis locais e resultados intermediários de cálculos.
  - Devem permitir que os dados que estejam sendo usados com mais freqüência sejam acessados rapidamente (sem acessos à memória).
  - Em algumas máquinas, os registradores de propósito geral são intercambiáveis e equivalentes
    - A escolha do registrador R1 ou do R25 para executar uma tarefa é irrelevante.

Um exemplo: Registradores do MIPS

Name	Number	Use
$zero	$0	constant 0
$at	$1	assembler temporary
$v0–$v1	$2–$3	Values for function returns and expression evaluation
$a0–$a3	$4–$7	function arguments
$t0–$t7	$8–$15	temporaries
$s0–$s7	$16–$23	saved temporaries
$t8–$t9	$24–$25	temporaries
$k0–$k1	$26–$27	reserved for OS kernel
$gp	$28	global pointer
$sp	$29	stack pointer
$fp	$30	frame pointer
$ra	$31	return address

Registradores (4)

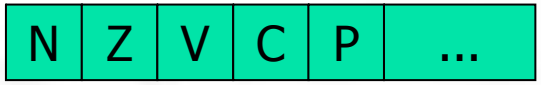
- No exemplo anterior, apesar de existirem diversos registradores de propósito geral, os sistemas operacionais e os compiladores adotam convenções de como usá-los.

- Os registradores do modo *kernel* não são visíveis aos usuários, pois controlam *caches*, memórias, dispositivos de E/S e outras características do hardware.
  - Somente o sistema operacional tem acesso a esses registradores.

Registradores (5)

- **PSW (*Program Status Word*) – Palavra de Status de Programa**
  - Registrador Híbrido
  - Guarda um conjunto diversificado de bits que são necessários à operação do processador. Por exemplo, os bits de **Códigos de Condição**.
  - Usos típicos: definir modo de execução da máquina (kernel ou usuário); bit para especificar o *trace* (depuração de programas); campo para especificar nível de prioridade da CPU; campo para habilitar interrupções; etc.
- **Códigos de Condição**
  - Esses bits são modificados pelo processador a cada ciclo da ULA, refletindo o estado do resultado da operação mais recente realizada
  - São importantes pois as instruções de **comparação** e de **desvio condicional** utilizam esses códigos nas suas execuções.

Registradores (6)



Exemplo de Flags definidas no PSW

- N – Igual a 1 quando o último resultado da ULA foi negativo
- Z – Igual a 1 quando o último resultado da ULA foi zero
- V – Igual a 1 quando a última operação da ULA gerou um *oVerflow*
- C – Igual a 1 quando a última operação da ULA gerou um Vai-um para o bit mais à esquerda
- P – Igual a 1 quando o último resultado da ULA apresentar Paridade par/impar

- Usos como Códigos de Condição (bits), exemplo:
  - Instrução **CMP** (comparação)
    - Subtrai dois operandos e ajusta os códigos de condição com base no resultado da operação. Se os operandos forem iguais, o resultado será zero e o bit referente ao código de condição Z da PSW recebe 1.
  - Instrução **BEQ** (Desvio se Igual – *Branch on EQual*)
    - Executada após a instrução CMP testa o bit Z e desvia se ele for igual a 1.

Visão Geral do Nível ISA do Pentium II (1)

- **ISA IA-32 da Intel ... Um pouquinho de história**
- Baseado e com suporte completo para execução de programas escritos nos processadores **8086** e **8088** (mesma arquitetura de 16 bits, o **8088** se difere por ter um barramento de dados de 8 bits).
- Contém resquícios da arquitetura do **8080** (processador de 8 bits da década de 70).
- O 8080 foi muito influenciado pelo **8008**, que por sua vez foi baseado no 4004 (chip de 4 bits).
- O sucessor do **8086/8088** foi o **80286** (também de 16 bits). Ele possuía um espaço de endereçamento maior que seus antecessores.
- O **80386** foi a primeira máquina 32 bits da Intel. As máquinas que a seguiram (**80486**, Pentium, Pentium Pro, Pentium II, Pentium III, Celeron e Xeon) possuem a arquitetura de 32 bits do 80386, a IA-32.
- A maior mudança foi a introdução das instruções MMX.
- Finalmente chegamos aos 64 bits
  - Pentium 4 : Extensão de 64bits para acesso a memória
  - ISA IA-64: Originalmente implementada no Itanium(2001)
  - x86-64 (AMD64 / EM64T)

Visão Geral do Nível ISA do Pentium II (2)

- Essa arquitetura possui três **Modos de Operação**.
- **Modo Real**
  - O processador funciona como fosse um simples 8088/8086.
  - Se qualquer programa fizer algo errado, toda a máquina pára!
- **Modo Virtual 8086**
  - Possibilita rodar programas antigos do 8088/8086 de forma protegida
  - Um sistema operacional real está controlando a máquina.
  - O SO cria um ambiente especial isolado que funciona como se fosse um processador 8088.
  - Ex: execução do MS-DOS sob Windows

Visão Geral do Nível ISA do Pentium II (3)

- **Modo Protegido**
  - O processador funciona como um Pentium II realmente.
  - Nesse modo de operação, existem 4 **Níveis de Privilégio** (controlados por bits da PSW). Geralmente, só são usados são os níveis 0 e 3.
  - **Nível 0:** corresponde ao modo Kernel de outros processadores. Os programas que rodam nesse nível têm acesso total a todos os recursos da máquina. O SO roda nesse nível.
  - **Nível 3:** utilizado para programas de usuário. Bloqueia acesso a determinadas instruções críticas e a certos registradores de controle para evitar que um programa de usuário possa causar algum dano à máquina.

Visão Geral do Nível ISA do Pentium II (4)

- Modos de operação x86-64

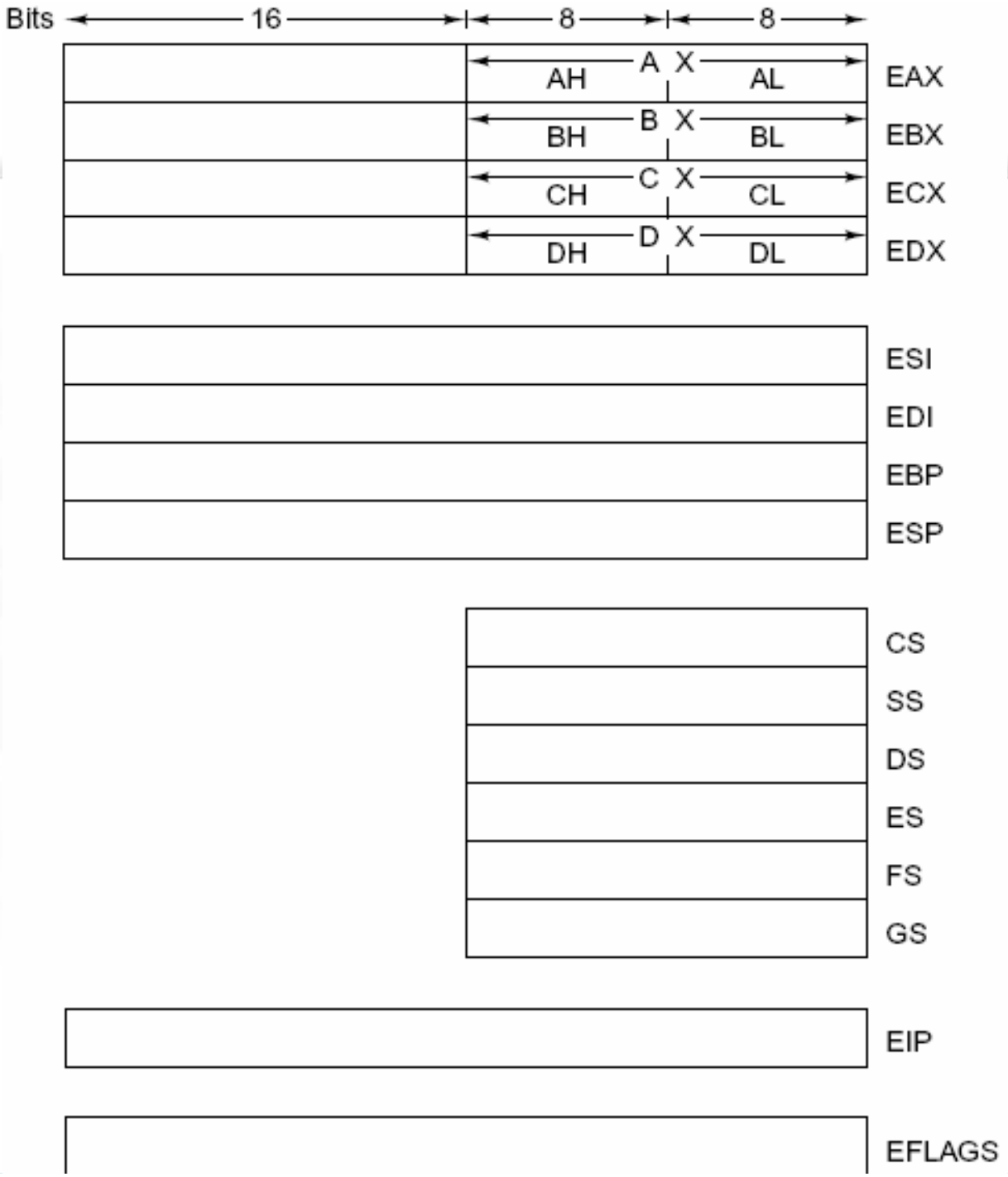
Operating mode		Operating system required	Application rebuild required	Default address size	Default operand size	Register extensions	Typical GPR width
Long mode	64-bit mode	New OS with 64-bit support	Yes	64	32	Yes	64
	Compatibility mode		No	32	32	No	32
				16	16		16
Legacy mode	Protected mode	Legacy 16-bit or 32-bit OS	No	32	32	No	32
				16	16		16
	Virtual 8086 mode			16	16		16
	Real mode	Legacy 16-bit OS					

Visão Geral do Nível ISA do Pentium II (4)

- **Espaço de Endereçamento**
  - É composto por 16.384 segmentos, cada um com endereços de 0 a $2^{32} - 1$.
  - No entanto, a maioria dos Sistemas Operacionais (UNIX e Windows entre eles) só suporta um segmento, de maneira que a maioria das aplicações enxerga um único espaço de endereçamento linear de 2^{32} bytes.
  - Cada byte tem seu próprio endereço
  - Palavras de 32 bits armazenadas no formato *little endian* (o byte de mais baixa ordem tem o menor endereço)

Visão Geral do Nível ISA do Pentium II (5)

- Principais Registradores do Pentium II



Visão Geral do Nível ISA do Pentium II (6)

- Registradores genéricos: **EAX, EBX, ECX e EDX**
  - São de 32 bits e funcionam (mais ou menos) de propósito geral.
  - EAX é o principal registrador aritmético;
  - EBX serve para armazenar ponteiros (endereços de memória);
  - ECX tem seu papel principal quando da execução de loops;
  - EDX é utilizado em operações de multiplicação e de divisão.
  - O "E" indica a Extensão dos registradores de 8 e de 16 bits dos processadores antecessores.

Visão Geral do Nível ISA do Pentium II (7)

- Registradores de índices: **ESI e EDI**
  - São de 32 bits e funcionam (mais ou menos) de propósito geral.
  - Objetivo principal de armazenamento de ponteiros para a memória, os quais são usados na execução de instruções de manipulação de *strings*.
  - ESI aponta para o string-fonte;
  - EDI aponta para o string-destino.
- Registradores de índices: **EBP e ESP**
  - ESP é o apontador de pilha.
  - O EBP é Apontador de Quadro: usado para apontar para a base do quadro da pilha local

Visão Geral do Nível ISA do Pentium II (8)

- Registradores de Segmento: **CS, SS, DS, ES, FS e GS**
  - São os registradores de **Segmento**.
  - Compatibilidade com 8088/8086
- **EIP (*Extended Instruction Pointer*)**
  - Program Counter !!!
- **EFLAGS**
  - É o PSW da arquitetura IA-32.

Componentes o nível ISA

- Modelo de Memória da Máquina;
- Conjunto de registradores
- Quais instruções são disponíveis e o que cada uma executa;
- Tipos de Dados utilizados;
- etc.

Instruções

- Principal característica de um nível ISA: seu conjunto de instruções de máquina.
- As instruções controlam tudo aquilo que a máquina pode fazer.
- De uma forma geral, podem sempre existir:
  - Instruções LOAD e STORE (em diversos formatos): permitir o movimento de dados entre a memória e os registradores.
  - Instruções MOVE: copiar valores entre registradores.
  - Instruções aritméticas, lógicas, de comparação de valores.
  - Instruções de desvio.

Tipos de Dados (1)

- **Tipos de Dados Numéricos**
  - Inteiros
    - Com e sem sinal
    - Diferentes tamanhos (8 ...64 bits)
  - Ponto flutuante
    - Considera-se o sinal, a parte inteira e a parte decimal
    - Podem existir registradores e instruções específicos para esses tipos de dados
- **Tipos de Dados Não-Numéricos**
  - Tipicamente usados em processadores de texto, planilhas e banco de dados
    - Instruções de cópia, busca, edição, manipulação de caracteres, etc.
  - Códigos mais usados atualmente de caracteres: ASCII e UNICODE
  - Valores Booleanos (true e false)
    - **Mapa de Bits:** contagem de blocos livres em um disco
- Tipo de Dado **Ponteiro**: um endereço de memória de máquina

Tipos de Dados (2)

- **Tipos de Dados no Pentium II**
  - Inteiros em complemento de 2
  - Inteiros sem sinal
  - Números decimais codificados em binário (BCD)
  - Ponto flutuante no padrão IEEE 754
  - ASCII de 8 bits

Type	8 Bits	16 Bits	32 Bits	64 Bits	128 Bits
Signed integer	×	×	×		
Unsigned integer	×	×	×		
Binary coded decimal integer	×				
Floating point			×	×	

Referências

- Andrew S. Tanenbaum, ***Organização Estruturada de Computadores***, 7ª edição, Prentice-Hall do Brasil, 2007.
- John L. Hennessy and David A. Patterson, **Arquitetura de Computadores: Uma Abordagem Quantitativa.** 3ª edição. Editora Campus, 2003.