3.24 – Casos Reais

PowerPC da IBM e da Motorola:

Semelhanças com o MIPS:

- 32 registradores para inteiros.
- Instruções de 32 bits.
- Troca de dados com a memória a partir de instruções de `load` e `store`.

Diferença: possui dois outros modos de endereçamento.

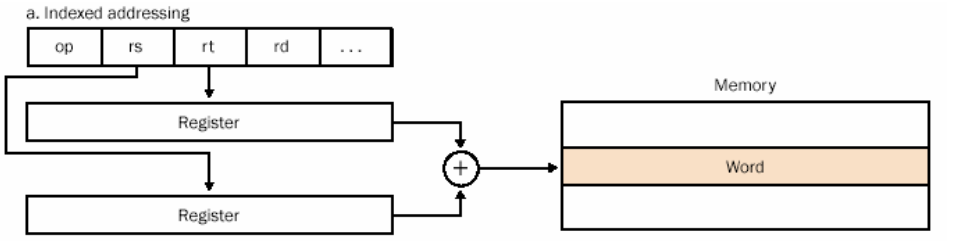
- Endereçamento indexado: permite que dois registradores sejam referenciados juntos.

No MIPS:

```
add $t0, $a0, $s3 # a0 tem a base de um vetor, $s3 é
                  # o índice
lw $t1, 0($t0)    # $t1 ← Memória[$a0 + $s3]
```

No PowerPC:

```
lw $t1, $a0 + $s3 # $t1 ← Memória[$a0 + $s3]
```



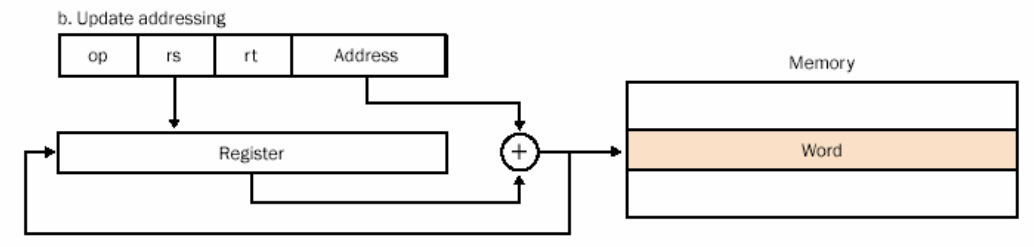
- Endereçamento atualizado: as instruções de transferência de dados que incrementam automaticamente o registrador-base e apontando para a próxima palavra toda vez que o dado é transferido.

No MIPS:

```
lw $t0, 4($s3)    # $t0 ← Memória[$s3 + 4]
addi $s3, $s3, 4  # $s3 ← $s3 + 4
```

No PowerPC:

```
lwu $t0, 4($s3) #$t0 ← Memória[$s3 + 4];$s3 ← $s3+4
```



3.25 - Intel 80x86:

Intel 8080 – processador de 8 bits.
1978 – Intel anuncia processador de 16 bits, o 8086, com registradores internos de 16 bits com uso específico para determinadas operações. Enfrentou problemas com o avanço tecnológico de 16 bits sendo que os programas desenvolvidos ainda eram para 8 bits. Intel mantém compatibilidade com o 8080.

1980: introduz o 8087, co-processador para operações de ponto-flutuante que estendeu a arquitetura do 8086 introduzindo 60 novas instruções de FP. O 8087 baseia-se em pilha.

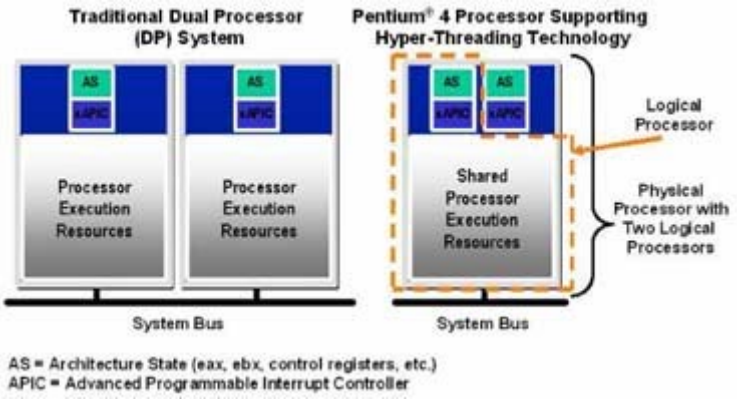
1982: Intel lança o 80286 aumentando o espaço de endereçamento para 24 bits. Mapeamento de memória diferenciado e com modo protegido, além de adicionar novas instruções ao ISA.

1985: o processador 80386 estende a arquitetura do 80286 para 32 bits, com 32 registradores e espaçamento de endereçamento de 32 bits. A instruções adicionais fazem com que o 80386 pareça uma máquina com registradores de propósito geral.

1989-1985: Intel lança o 80486 em 1989, o Pentium em 1992 e em 1995 o Pentium Pro. São adicionadas novas instruções em cada implementação.

1997: expansão do conjunto de instruções MMX (Multi Media Exchange).

2004: Hyper-Threading **–** provém um único processador físico a executar dois segmentos de código (chamadas de threads) concorrentemente, aumentando a utilização do processador, a vazão e melhorando a performance. Esta arquitetura Hyper-Threading consiste de dois processadores lógicos Pentium 4.



2005: Tecnologia Centrino para computadores móveis. Hardware de ultra-baixa potência para economia de energia.

3.26 - Operações inteiras no 80x86

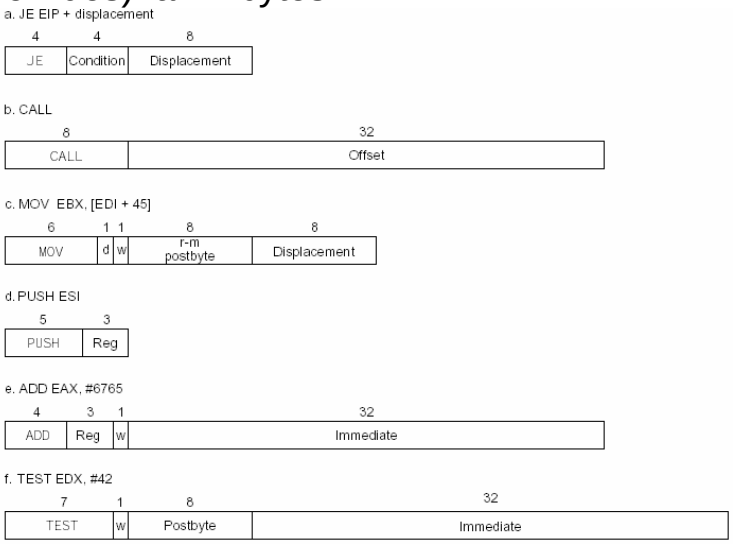
A arquitetura Intel deve suportar todos os modos de endereçamento desde o 8086.
O tamanho dos dados pode ser de 8, 16 e 32 bits.
Dessa forma, não existe um tamanho padrão, o tamanho é definido diretamente pela instrução.
As instruções aritméticas e lógicas têm um operando que atua tanto como fonte como destino.
Os operandos das instruções aritméticas e lógicas do 80x86 podem estar na memória.

Tipo de operando fonte/destino	Segundo operando fonte
Registrador	Registrador
Registrador	Imediato
Registrador	Memória
Memória	Memória
Memória	Imediato

As operações inteiras do 80x86 são dividas em quatro categorias:

- Instruções de movimento de dados: incluem move, push e pop.
- Instruções aritméticas inteiras e lógicas, operações de teste e operações aritméticas inteiras sobre binários e decimais.
- Instruções para controle de fluxo de execução: desvio condicional e incondicional, chamadas de procedimento e retorno de procedimento.
- Instruções de manipulação de strings: movimento e comparação de strings.

A codificação das instruções é complexa com muitos formatos diferentes de instruções.
O tamanho das instruções pode variar de 1 byte (quando não há operandos envolvidos) a 17 bytes



3.27 - Lendas e Falhas:

Lenda: Instruções mais poderosas significam melhor performance.

Lenda: Programar diretamente em linguagem de montagem para obter uma melhor performance.
A constante melhora dos compiladores tem mostrado que é cada vez menor a diferença entre o código gerado pelo computador e o código gerado diretamente pelo programador em linguagem de montagem.

Falha: Esquecer que os endereços de palavras consecutivas com endereçamento a byte não diferem de uma unidade.
No MIPS as instruções são endereçadas de 4 em 4 bytes.

3.28 - Considerações Finais:

O conjunto de instruções de uma máquina deve ter comprometimento entre a **quantidade de instruções** necessárias para a execução de um programa, **número de ciclos de clock** gasta por cada uma e a **freqüência do clock**.
O balanceamento deve seguir quatro princípios básicos:

1. A simplicidade é favorecida pela regularidade: instruções de mesmo tamanho, campos dos registradores na mesma posição em cada formato de instrução e operações aritméticas sempre com três operandos.
2. Quanto menor mais rápido: somente 32 registradores.
3. Um bom projeto demanda compromisso: permitir a representação de constantes e endereços maiores e a necessidade de manter as instruções de mesmo tamanho.
4. Torne o caso comum mais rápido: endereçamento relativo ao PC para desvios condicionais e endereçamento imediato para constantes.

As categorias de instruções do MIPS estão associadas às linguagens de alto nível:

- Instruções aritméticas associadas aos comandos de atribuição.
- Instruções de transferência de dados associadas a estruturas de dados como os vetores.
- Desvios condicionais aos comando if e loops.
- Desvios incondicionais associados a chamadas a procedimentos e retornos de procedimentos.

3.29 - Arquiteturas baseadas em Acumulador:

Os primeiros computadores tinham apenas um único registrador para realizar as operações aritméticas.
A falta de mais registradores permite que um operando seja referenciado à memória

Exemplo: (processador Z80)

A = B + C na arquitetura baseada em acumulador será:

```
load Address B  # Acc ← Memória[Endereço de B]
add Address C   # Acc ← B + Memória[Endereço de C]
store Address A # Memória[Endereço de A] ← B + C
```

3.30 - Arquiteturas com Registradores de Propósito Geral

O projeto de arquiteturas evoluiu e outros registradores de propósito específico foram adicionados.
A generalização de máquinas com registradores dedicados permitiu que os registradores pudessem ser usados com a finalidade de propósito específico.
Este projeto permite ter um operando armazenado em memória para execução de instruções aritméticas, como nas máquinas baseada em acumulador chamadas de arquitetura ***Registrador-Memória***.
Nas arquiteturas em que os operandos estejam sempre em registradores as máquinas são chamadas de ***load-store***, ou máquinas ***Registrador-Registrador***.
Exemplos dessa arquitetura: MIPS, CDC6600 e outros

A combinação de operandos em memória, é conhecida como uma arquitetura ***Memória-Memória***. São exemplos dessa arquitetura: VAX (DEC) e IBM 360.

Exemplo:
Código A = B + C usando a instruções Memória-Memória.

```
add EndereçoA, EndereçoB, EndereçoC
```

Problema: tamanho diferente de instruções!

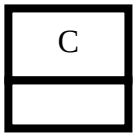
3.31 - Arquitetura de Pilha

O conjunto de instruções baseados no modelo de execução com base em uma pilha, a exemplo de instruções desenvolvidas nas calculadoras HP.
Os operandos são colocados em uma pilha vindos da memória e saem da pilha em direção a memória.

Exemplo:
Código A = B + C usando a instruções de pilha.

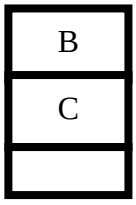
```
push AddressC  #topo ← topo +4;
               # pilha[topo] ← memória[AddressC]
push AddressB  #topo ← topo +4;
               # pilha[topo] ← memória[AddressB]
add            #pilha[topo – 4] ← pilha[topo] +
               #pilha[topo – 4]; topo ← topo -4
pop AddressA   #Memória{AddressA] ← pilha[topo];
               # topo ← topo -4
```

```
push AddressC
```



← Topo

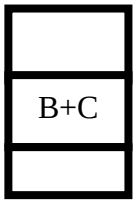
```
push AddressB
```



← Topo

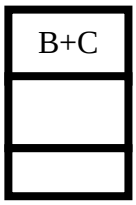
← Topo -4

```
add
```



← Topo

Topo -4 ← Topo + Topo -4



Topo ← Topo -4

Topo -4

```
pop AddressA  #Memória{AddressA] ← pilha[topo];
```