Arquitetura do x86

ORGANIZAÇÃO BÁSICA DE COMPUTADORES E LINGUAGEM DE MONTAGEM

Processador INTEL 80X86

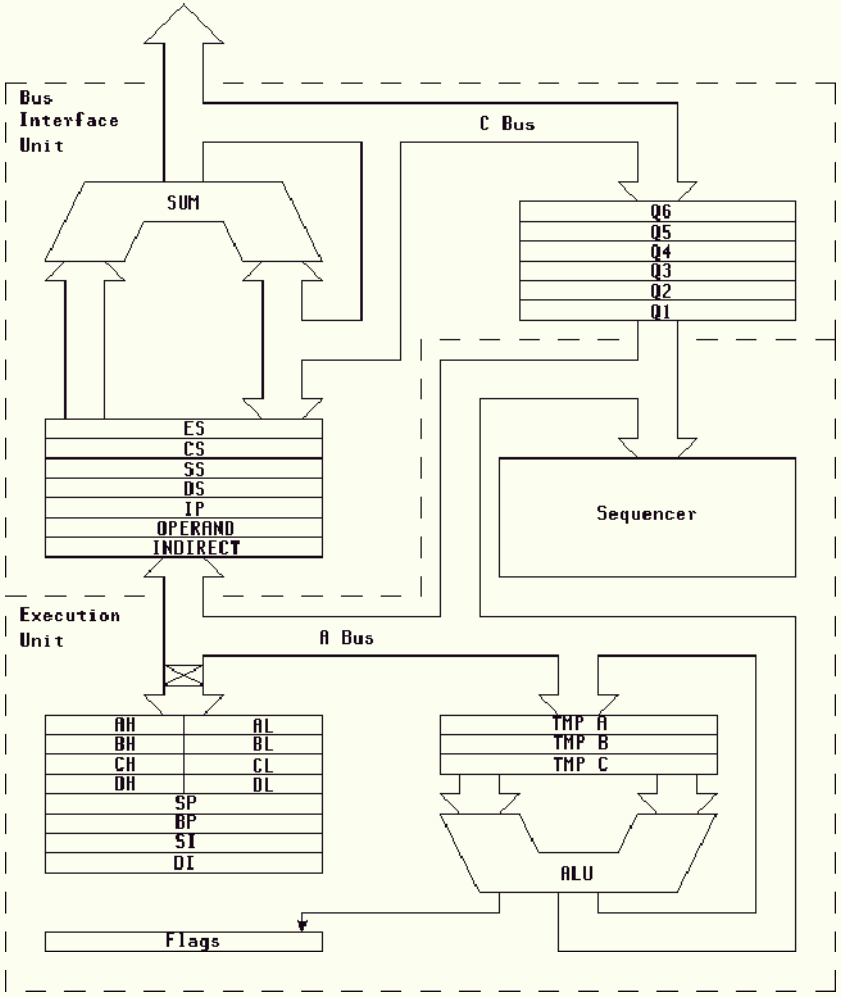
- **O 8086 divide-se internamente em duas unidades:**

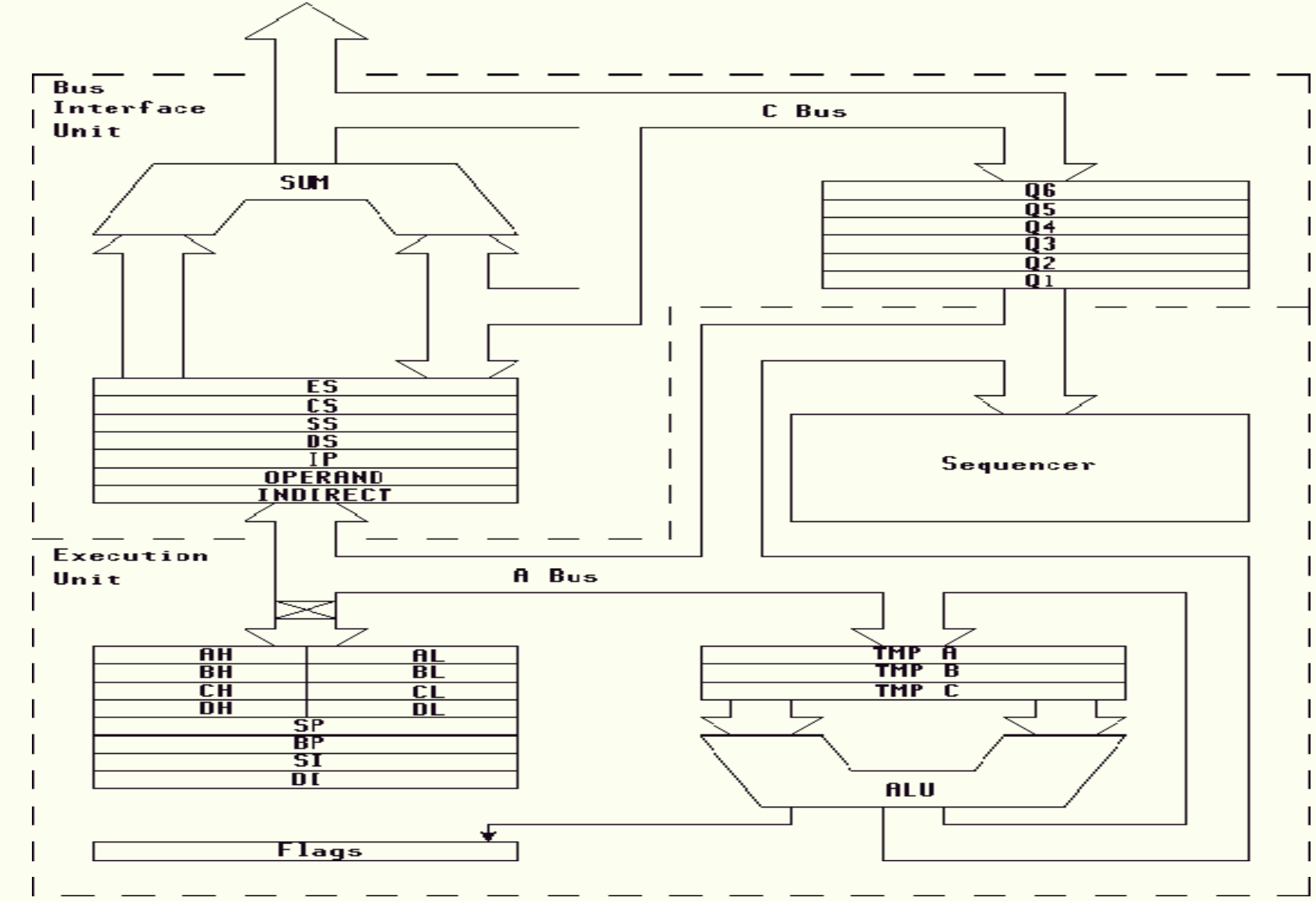
  - **Execution Unit (EU) - unidade de execução:**
    - **ULA - realiza operações aritméticas de +, -, X, / e operações lógicas AND, OR, NOT, XOR;**
    - **Registradores para armazenamento temporário durante as operações, que são endereçados por nome.**

  - **BUS Interface Unit (BIU) - unidade de interface de barramento:**
    - **Faz a comunicação de dados entre a EU e o meio externo (memória, E/S);**
    - **Controla a transmissão de sinais de endereços, dados e controle;**
    - **Controla a sequência de busca e execução de instruções;**
    - **Mecanismo de *pre-fetch*: busca até 6 instruções futuras deixando-as na fila de instruções (*instruction queue*) → aumento de velocidade.**

Processador INTEL 80X86

- **Organização**




Bus
Interface
Unit
C Bus
SUM
Q6
Q5
Q4
Q3
Q2
Q1
ES
CS
SS
DS
IP
OPERAND
INDIRECT
Sequencer
Execution
Unit
A Bus
AH
AL
BH
BL
CH
CL
DH
DL
SP
BP
SI
DI
TMP A
TMP B
TMP C
ALU
Flags

ORGANIZAÇÃO BÁSICA DE COMPUTADORES E LINGUAGEM DE MONTAGEM

Processador INTEL 80X86

- **Registradores:**
  - **de propósito geral ou de dados;**
  - **de endereços (segmentos, apontadores e índices);**
  - **sinalizadores de estado e controle (FLAGS);**

- **Registradores de propósito geral (de dados):**
  - **AX, BX, CX e DX**
    - **são todos registradores de 16 bits**
    - **utilizados nas operações aritméticas e lógicas**
    - **podem ser usados como registradores de 16 ou 8 bits:**
      - **AH e AL**
      - **BH e BL**
      - **CH e CL**
      - **DH e DL**

      8 registradores de 8 bits cada
      "H" → byte alto ou superior
      "L" → byte baixo ou inferior

Processador INTEL 80X86

- **Registradores de propósito geral (de dados):**
  - **AX (acumulador) → utilizado como acumulador em operações aritméticas e lógicas; em instruções de E/S, ajuste decimal, conversão, etc**
  - **BX (base) → usado como registrador de BASE para referenciar posições de memória; BX armazena o endereço BASE de uma tabela ou vetor de dados, a partir do qual outras posições são obtidas adicionando-se um valor de deslocamento (*offset*).**
  - **CX (contador) → utilizado em operações iterativas e repetitivas para contar bits, bytes ou palavras, podendo ser incrementado ou decrementado; CL funciona como um contador de 8 bits.**
  - **DX (dados) → utilizado em operações de multiplicação para armazenar parte de um produto de 32 bits, ou em operações de divisão, para armazenar o resto; utilizado em operações de E/S para especificar o endereço de uma porta de E/S**.

Processador INTEL 80X86

- **Registradores de segmentos:**
  - **CS, DS, SS e ES**
    - **são todos registradores de 16 bits**
    - **o endereçamento no 8086 é diferenciado para:**
      - **código de programa (instruções)**
      - **dados**
      - **Pilhas**
  - **segmento: é um bloco de memória de 64 KBytes, endereçável.**
  - **durante a execução de um programa no 8086, há 4 segmentos ativos:**
    - **segmento de código endereçado por CS**
    - **segmento de dados endereçado por DS**
    - **segmento de pilha endereçado por SS (stack segment)**
    - **segmento extra (de dados) endereçado por ES**

Processador INTEL 80X86

- **Registrador apontador de instrução:**
  - **IP (*instruction pointer*): utilizado em conjunto com CS para localizar a posição, dentro do segmento de código corrente, da próxima instrução a ser executada;**
  - **IP é automaticamente incrementado em função do número de bytes da instrução executada.**
  - **Observação: o IP é o já mencioanado PC (program counter) do INTEL 80x86**

Processador INTEL 80X86

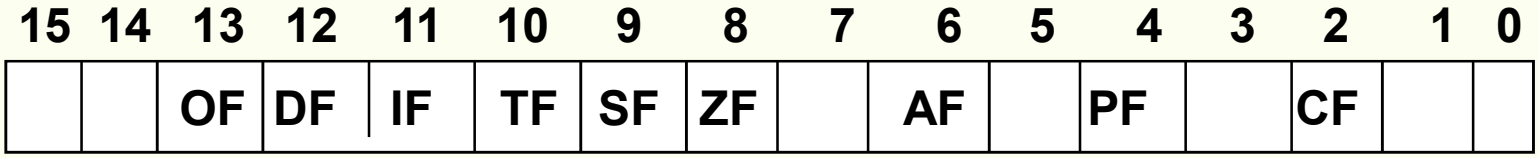
- **Registradores apontador de pilha e de índice:**
  - **Armazenam valores de deslocamento de endereços (*offset*), a fim de acessar regiões da memória muito utilizadas:**
    - **pilha,**
    - **blocos de dados,**
    - ***arrays* e *strings*.**
  - **SP (*stack pointer* - apontador de pilha) é utilizado em conjunto com SS, para acessar a área de pilha na memória; aponta para o topo da pilha.**
  - **BP (*base pointer* - apontador de base) é o ponteiro que, em conjunto com SS, permite acesso de dados dentro do segmento de pilha.**
  - **SI (*source index* - índice fonte) usado como registrador índice em alguns modos de endereçamento indireto, em conjunto com DS.**
  - **DI (*destination index* - índice destino) similar ao SI, atuando em conjunto com ES.**
- **Obs: SI e DI facilitam a movimentação de dados sequenciados entre posições fonte (indicado por SI) e posições destino (indicado por DI).**

Processador INTEL 80X86

- **Registrador de sinalizadores (FLAGS):**
  - **indica o estado do processador durante a execução de cada instrução;**
  - **conjunto de bits individuais, cada qual indicando alguma propriedade;**
  - **subdividem-se em: FLAGS da estado (*status*) e FLAGS de controle.**
  - **organização:**
    - **1 registrador de 16 bits;**
    - **6 FLAGS de estado;**
    - **3 FLAGS de controle;**
    - **7 bits não utilizados (sem função);**

Processador INTEL 80X86

- **Registrador de sinalizadores (FLAGS):**



- **Flags de estados:**
  - **CF - Flag de Carry**
    - **CF = 1 → após instruções de soma que geram "vai um"após instruções de subtração que não geram "emprestimo" ("empresta um");**
    - **CF = 0 → caso contrário.**
  - **PF - Flag de paridade**
    - **PF = 1 → caso o resultado de alguma operação aritmética ou lógica apresentar um número par de "1's";**
    - **PF = 0 → caso contrário (número impar de 1s).**

Processador INTEL 80X86

- **Flags de estados (continuação):**
  - **AF - Flag de Carry Auxiliar: utilizado em instruções com números BCD**
    - **AF = 1 → caso exista o "vai um" do bit 3 para o bit 4 de uma adição caso não exista "empréstimo" do bit 4 para o bit 3 numa subtração;**
    - **AF = 0 → caso contrário.**
  - **ZF - Flag de Zero**
    - **ZF = 1 → caso o resultado da última operação aritmética ou lógica seja igual a zero;**
    - **ZF = 0 → caso contrário.**

```
SUB AX,BX
JZ END

CMP AX,BX
JE END
```

SE AX = BX ZF = 1
CASO CONTRÁRIO ZF = 0

Processador INTEL 80X86

- **Flags de estados (continuação):**
  - **SF - Flag de Sinal: utilizado para indicar se o número resultado é positivo ou negativo em termos da aritmética em Complemento de 2 (se não ocorrer erro de transbordamento - *overflow*).**
    - **SF = 1 → número negativo;**
    - **SF = 0 → número positivo.**
  - **OF - Flag de Overflow (erro de transbordamento).**
    - **OF = 1 → qualquer operação que produza *overflow*;**
    - **OF = 0 → caso contrário.**

OVERFLOW

NÚMERO SEM SINAL

```
  1011      11
+ 0111      07
_______
1  0010     18
```

NÚMERO SINALIZADO

```
  1010      -6          0101 +1 = 0110  = 6 DECIMAL
 +1010      -6
______
1 0100
```

Processador INTEL 80X86

- **Flags de controle:**
  - **TF - Flag de Trap (armadilha)**
    - **TF = 1→ após a execução da próxima instrução, ocorrerá uma interrupção; a própria interrupção faz TF = 0;**
    - **TF = 0 → caso contrário**
  - **IF - Flag de Interrupção**
    - **IF = 1 → habilita a ocorrência de interrupções;**
    - **IF = 0 → inibe interrupções tipo INT externas.**
  - **DF - Flag de Direção: usado para indicar a direção em que as operações com *strings* são realizadas.**
    - **DF = 1 → decremento do endereço de memória (DOWN);**
    - **DF = 0 → incremento do endereço de memória (UP).**

ORGANIZAÇÃO BÁSICA DE COMPUTADORES E LINGUAGEM DE MONTAGEM

Processador INTEL 80X86

- **Os registradores do 8086 (resumo):**

Registradores de dados			
AH	AL	→	AX
BH	BL	→	BX
CH	CL	→	CX
DH	DL	→	DX
Registradores de segmentos			
CS			
DS			
SS			
ES			
Registradores índices e apontadores			
SI			
DI			
SP			
BP			
IP			
Registrador de sinalizadores			
FLAGS			

Processador INTEL 80X86

- **Barramentos**
  - **barramento de endereços é comum com o de dados: 20 bits**
  - **endereços: 20 bits, 2^{20} = 1.048.576 combinações = 1 MByte (1 MB)**
  - **dados: utiliza somente 16 bits do barramento comum**
  - **barramento de controle: 16 bits independentes do barramento comum**

Processador INTEL 80X86

- **Gerenciamento de Memória**
  - **O 8086 possui 20 bits para acessar posições de memória física**
  - **2^{20} = 1.048.576 bytes (1 Mbyte) posições endereçáveis**
  - **Exemplos de endereços:**

$0000\ 0000\ 0000\ 0000\ 0000_2 \rightarrow 00000_{16}$
$0000\ 0000\ 0000\ 0000\ 0001_2 \rightarrow 00001_{16}$
$0000\ 0000\ 0000\ 0000\ 0010_2 \rightarrow 00002_{16}$
$0000\ 0000\ 0000\ 0000\ 0011_2 \rightarrow 00003_{16}$
....
$1111\ 1111\ 1111\ 1111\ 1111_2 \rightarrow FFFFF_{16}$

ORGANIZAÇÃO BÁSICA DE COMPUTADORES E LINGUAGEM DE MONTAGEM

Processador INTEL 80X86

- **Gerenciamento de Memória**

  - **O 8086 opera internamente com 16 bits**
    - **Problema: Como gerar endereços com 20 bits?**
    - **Solução: Utilizar a idéia de segmentação de memória!**

  - **Segmento de memória: bloco de 64 Kbytes de posições de memória consecutivas**
  - **2^{16} = 65.536 bytes (64 Kbytes)**
  - **Segmento de memória é identificado por um número de segmento**
  - **Uma posição de memória é especificada pelo número de segmento e por um deslocamento (*offset*) em relação ao início do segmento**

  - **Formato de endereço lógico → segmento:offset**
  - **Exemplo de endereçamento:**

    Dado o endereço lógico: 8350:0420h

    reconhece-se:

    segmento no. 8350_{16} } endereço

    deslocamento 0420_{16} } lógico

    o endereço físico vale:

    83500_{16} -> desloca-se 1 casa hexa (4 casas binárias)

    $+\ 0420_{16}$ -> soma-se o deslocamento

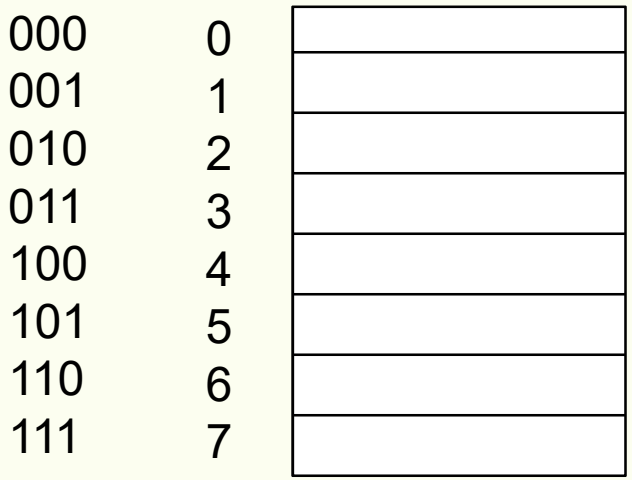
    83920_{16} -> endereço físico resultante (20 bits)

ORGANIZAÇÃO BÁSICA DE COMPUTADORES E LINGUAGEM DE MONTAGEM

		5 bits	8bits		
	SEG	**END**	**CONTEÚDO**		barramento - 5 bits
	0	0			palavras - 3 bits
		1			
00010		2			
		3			
		4			
		5			
		6			
		7			
	1	8			
		9			
		10			
		11			
		12			
01101		13			
		14			
		15			
	2	16			
		17			
		18			
		19			
		20			
		21			
		22			
		23			
	3	24			
		25			
		26			
		27			
		28			
		29			
11110		30			
		31			

ORGANIZAÇÃO BÁSICA DE COMPUTADORES E LINGUAGEM DE MONTAGEM

ENDEREÇO FÍSICO = 3



8 BITS

OFFSET TENHA 2 BITS
E SEGMENTOS

OFFSET
00
01
10
11

0:01 -> 000 + 01 = 001

Processador INTEL 80X86

- **Gerenciamento de Memória**
  - **O identificador de segmento (base) aponta para uma região da memória;**
  - **O offset aponta para um local dentro deste segmento;**
  - **O offset é aquele que aparece nos programas como o endereço dos dados, rótulos e endereços de instruções;**
  - **Segmentação é um esquema muito útil para gerar códigos relocáveis;**
  - **Endereços lógicos diferentes podem representar o mesmo endereço físico;**

Exemplo:

base →	**$028C_{16}$**
offset →	**0003_{16}**
endereço físico →	**$028C3_{16}$**
base →	**0287_{16}**
offset →	**0053_{16}**
endereço físico →	**$028C3_{16}$.**

Processador INTEL 80X86

- **Organização de memória**
  - **A ocupação de memória feita pelo DOS (1024 Kbytes = 1 Mbyte) considera: 640 Kbytes para programas (10 segmentos disjuntos de 64 Kbytes) e 384 Kbytes reservados para memória de vídeo, BIOS, etc.**

BIOS	**F0000h**
Reservado	**E0000h**
Reservado	**D0000h**
Reservado	**C0000h**
Vídeo	**B0000h**
Vídeo	**A0000h**
Área de programas de aplicação	
Sistema Operacional (DOS)	
BIOS e dados da BIOS	**00400h**
Vetores de interrupção	**00000h**

Processador INTEL 80X86

- **Interrupção**
  - **Ocorrência eventual, durante a execução de um processamento pelo computador, que deve ser prontamente atendida, causando a suspensão do processamento em curso para o atendimento da "*chamada*".**
- **Tipos de interrupções do 8086:**
  - **Causadas pela ocorrência de eventos "catastróficos": falta de energia, erro de memória, erro de paridade em comunicações, etc. (este tipo de interrupção não pode ser inibida).**
  - **Causadas pela ação de dispositivos externos (periféricos): podem ser habilitadas ou inibidas.**
  - **Causadas pelo próprio programa em curso: erro de divisão, erro de transbordamento (*overflow*).**
  - **TRAP - útil para depuração de um programa**
  - **BREAKPOINT - colocado em pontos estratégicos do programa para permitir processamento especial**
- **Interrupção RESET: permite a inicialização do microprocessador, via *hardware*.**