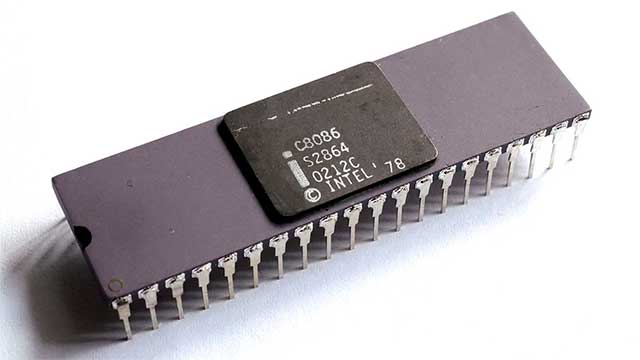
**Nomes:** Daniela Stohr, Jonathan Rodrigues, Luis Augusto Simon e Matheus Willms

**PROCESSADOR INTEL 8086**

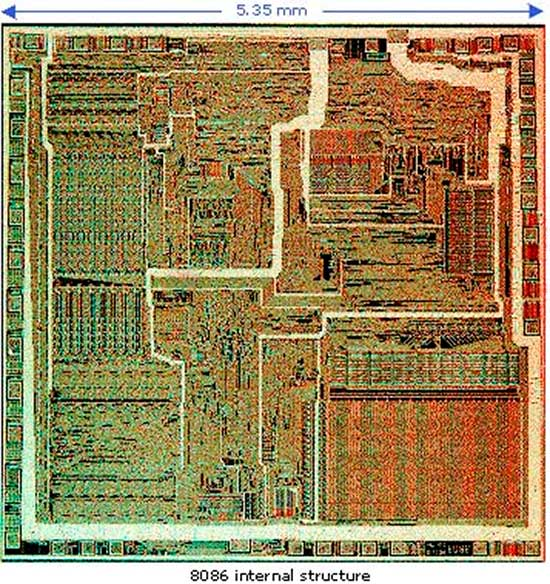
**DEFINIÇÃO**

O Intel 8086 era um processador de 16 bits desenvolvido pela Intel a partir de 1976 e lançado em 9 de junho de 1978. Ele deu origem à arquitetura x86 e iniciou a longa fila da arquitetura de CPU mais bem-sucedida do mundo. Ele tinha um barramento de dados de 16 bits, portas de E / S de 64 KB, um barramento externo de 20 bits e rodava tão rápido quanto 10MHz, pela primeira vez, a velocidade do clock alcançou 5MHz.



Intel 8086, o primeiro microprocessador X86

Tecnicamente falando, o Intel 8086 era um microprocessador com uma arquitetura completa de 16 bits que possuía registradores de 16 bits, um barramento de dados de 16 bits e um barramento de endereço de 20 bits que podiam fazer referência a 1 MB de memória física. Porém, por causa de seus registros de 16 bits, ele só conseguiu endereçar efetivamente 64 KB de memória. No entanto, o que tornou esse processador especial foram seus registros de segmento, que permitiram endereçar além de 64 KB de memória, o que poderia especificar os locais de memória para o código, dados, pilha e também um extra de 64 KB de segmento de dados.



O processador Intel 8086 foi utilizado por sistemas como semáforos e calculadoras, assim como no PC-1512, que foi um modelo de computador pessoal fabricado pela Amstrad, que dispunha de um processador de 16 bits, o Intel 8086, arquitetura XT. O computador era vendido com o GEM Desktop, um ambiente gráfico dos anos 1980.



O Amstrad PC-1512

**MODELO**

A arquitetura x86 segue o modelo da arquitetura de Von Neumann onde esse, trabalha principalmente usando uma unidade central de processamento (CPU) e uma memória principal.

As instruções podem trabalhar manipulando/lendo dados em registradores que são pequenas áreas de memória internas à CPU. E também pode manipular dados na memória principal, que no caso é a memória RAM. Bem como também usar o sistema de entrada e saída de dados, feito pelas portas físicas.

O registrador *Program Counter* no diagrama abaixo armazena o endereço da próxima instrução que será executada na memória principal. Na arquitetura x86 esse registrador é chamado de “Instruction Pointer”.

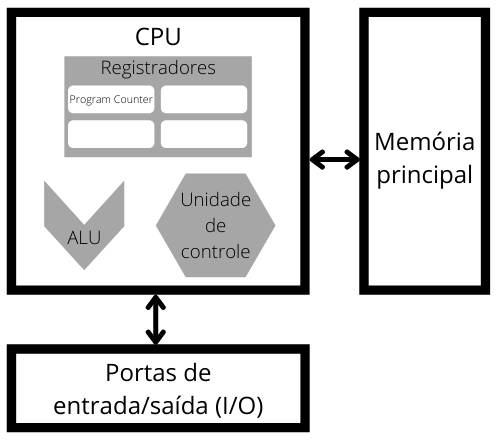


Diagrama da arquitetura de Von Neumann

**CONFIGURAÇÃO INTERNA DO 8086**

Divide-se internamente em duas unidades:

Execution Unit (EU) - unidade de execução:

* UAL - realiza operações aritméticas de +, -, X, / e operações lógicas AND, OR, NOT, XOR;
* Contém 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.

**REGISTRADORES**

Registradores 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 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 Cap2.5 uma porta de E/S.

Registradores de segmento:

O 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: endereçado por ES

Registrador apontador de instrução:

Utilizado em conjunto com CS para localizar a posição, dentro do segmento de código corrente, da próxima instrução a ser executada. O IP (instruction pointer) é automaticamente incrementado em função do número de bytes da instrução executada.

Registradores apontador de pilha e de índice:

Armazenam valores de deslocamento de endereços (offset), a fim de acessar regiões da memória muito utilizadas, como pilha, blocos de dados, arrays e strings. Podem ser utilizados em operações aritméticas e lógicas, possibilitando que os valores de deslocamento sejam resultados de computações anteriores:

* 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.

O SI e DI facilitam a movimentação de dados sequenciados entre posições fonte (indicado por SI) e posições destino (indicado por DI).

Registrador de sinalizadores (FLAGS):

Conjunto de bits individuais, cada qual indicando alguma propriedade e o estado do microprocessador durante a execução de cada instrução. Subdividem-se em: FLAGS de 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).

Flags de estado:

CF - Flag de Carry

CF = 1 -> após instruções de soma que geram "vai um" após instruções de subtração que não geram "empréstimo" ("empresta um");

CF = 0 -> caso contrário.

PF - Flag de paridade

PF = 1 -> caso o byte inferior do resultado de alguma operação aritmética ou lógica apresentar um número par de "1's";

PF = 0 -> caso contrário (número ímpar).

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.

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.

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;

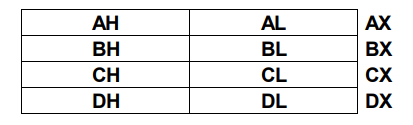
IF = 0 -> inibe interrupções tipo INT externas.

DF - Flag de Direção: Usado para indicar a direção em que as operações com strings são realizadas.

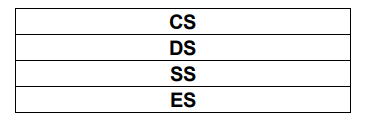
DF = 1 -> decremento do endereço de memória (DOWN);

DF = 0 -> incremento do endereço de memória (UP).

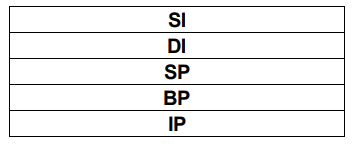
Os registradores do Intel 8086 (visão geral):



Registradores de dados



Registradores de segmentos



Registradores índices e apontadores



Registrador de sinalizadores

* O registrador IP corresponde ao Contador de Programa - PC.
* Todos os registradores são de 16 bits.
* AH -> byte alto de AX; AL byte baixo de AX; ambos de 8 bits.

**CONJUNTO DE INSTRUÇÕES**

Ao todo são 117 instruções e o código objeto de cada instrução ocupa de 1 a 6 bytes. Lista das classes de instrução e exemplos:

* Data transfer instructions: MOV, LDA, LDS…
* Arithmetic instructions: ADD, ADC, INC, SUB, DIV, MUL, IMUL, IDIV
* Logic instructions: AND, OR, XOR, NOT…
* Shift instructions: SAL, SHL, SHR, SAR…
* Rotate instructions: ROL, ROR, RDL, RCR…
* Flag control instructions: CLC, STC, LAHF, SAHF…
* Compare instructions: CMP
* Jump instructions: JMP, JA, JB, JGE, JNE…
* Subroutines and subroutine handling instructions: CALL, PUSH, RET...
* Loop and loop handling instructions: LOOP, LOOPE…
* Strings and string handling instructions: MOVS, CMPS...

Conjunto de instruções completas:

http://www.telecom.uff.br/orgarqcomp/Slides/OrgComp\_pt5\_Conjunto\_de\_Instrucoes\_8086-88.pdf

http://www.gabrielececchetti.it/Teaching/CalcolatoriElettronici/Docs/i8086\_instruction\_set.pdf

**LINK DO EMULADOR**

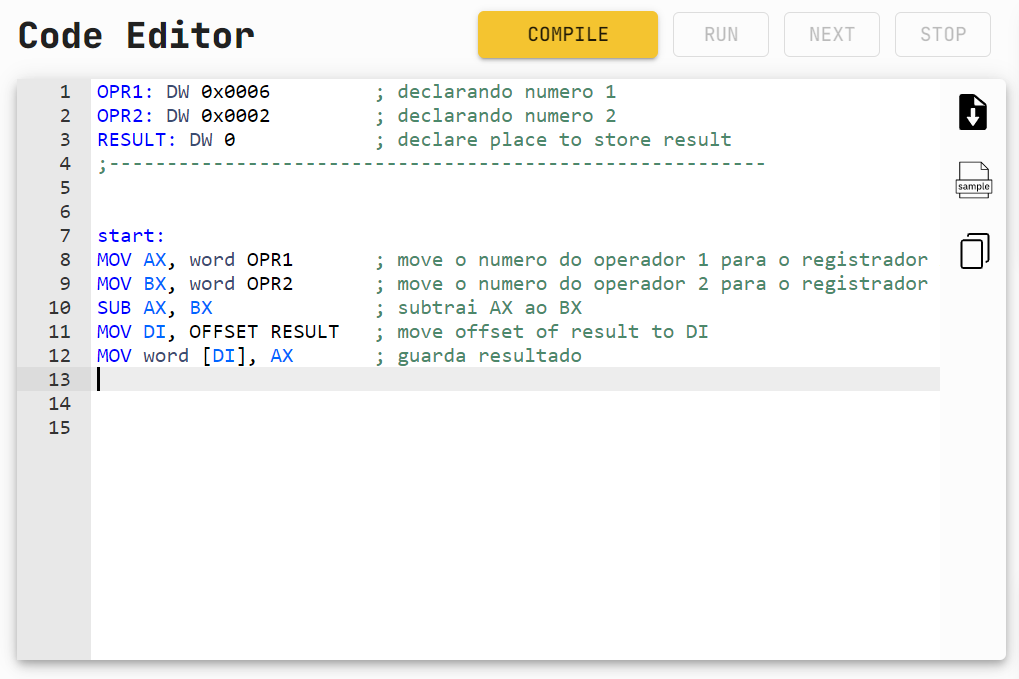
https://yjdoc2.github.io/8086-emulator-web/compile

**PROGRAMAS DESENVOLVIDOS**



Soma

:



Subtração

**REFERÊNCIAS**

<https://pt.theastrologypage.com/intel-8086>

<https://mentebinaria.gitbook.io/assembly/a-base/arquitetura-x86>

<https://www.ic.unicamp.br/~pannain/mc404/aulas/pdfs/OBCLM2.pdf>