



**CPS**  
Centro  
Paula Souza

 **GOVERNO DO ESTADO  
SÃO PAULO**

# Conjunto de instruções



# Conjunto de instruções

Processador	Número de Instruções
-------------	----------------------

8080	78
------	----

8088	117
------	-----

80486	286
-------	-----

Pentium II	217
------------	-----

Conjunto de instruções são as operações que um processador, ou outros periféricos programáveis suportam, fornecem ou disponibilizam para o programador, ou seja, é a representação do código de máquina, com a finalidade de facilitar o acesso ao componente.

$$A = ((B + C) \times D + E - F)$$

A – F – denotam posições da memória – endereços

As arquiteturas possuem as seguintes operações aritméticas:

ADD – para soma

SUB – para subtração

MUL – para multiplicação

DIV - para divisão

# Arquitetura de 4 endereços

$$A = ((B + C) \times D + E - F)$$

Endereço	Instrução	Comentário
e1	ADD B C A e2	Soma B com C, salva resultado em A
e2	MUL A D A e3	Multiplica A por D, salva resultado em A
e3	ADD A E A e4	Soma A com E, salva resultado em A
e4	SUB A F A e5	Subtrai F de A, salva resultado em A
e5	HALT	Fim do programa

# Arquitetura de 3 endereços

$$A = ((B + C) \times D + E - F)$$

## PC – program counter

Endereço	Instrução	Comentário
e1	ADD B C A	Soma B com C, salva resultado em A, incrementa PC
e2	MUL A D A	Multiplica A por D, salva resultado em A, incrementa PC
e3	ADD A E A	Soma A com E, salva resultado em A, incrementa PC
e4	SUB A F A	Subtrai F de A, salva resultado em A, incrementa PC
e5	HALT	Fim do programa

Qual a vantagem da arquitetura de 3 endereços para a arquitetura de 4 endereços?

# Arquitetura de 2 endereços

A economia de memória compensa o grau de liberdade as instruções;  
No entanto, instruções de 3 endereços ainda consomem muita memória;

Simplificando essa arquitetura, tem-se arquiteturas de 2 endereços.

$$A = ((B + C) \times D + E - F)$$

Endereço	Instrução	Comentário
e1	MOV A B	Move B para A
e2	ADD A C	Soma A com C, resultado em A
e3	MUL A D	Multiplica A por D, resultado em A
e4	ADD A E	Soma A com E, resultado em A
e5	SUB A F	Subtrai F de A, resultado em A
e6	HALT	Fim do programa

# Arquitetura de 1 endereços

Uso de registradores especiais, para reduzir o número de endereços.

Da memória para o acumulador (**LDA – Load Accumulator**)

Do acumulador para a memória (**STA – Store Accumulator**)

$$A = ((B + C) \times D + E - F)$$

Endereço	Instrução	Comentário
e1	LDA B	Move B para o acumulador
e2	ADD C	Soma Acumulador com C, resultado no acumulador
e3	MUL D	Multiplica acumulador por D, resultado no acumulador
e4	ADD E	Soma acumulador com E, resultado no acumulador
e5	SUB F	Subtrai F do acumulador, resultado no acumulador
e6	STA A	Armazena acumulador no endereço A
e7	HALT	Fim do programa

A grande vantagem deste tipo de arquitetura está na **economia de acessos a memória**, realizando basicamente operações com o **acumulador**



# Arquitetura de 0 endereços

Usar do conceito de pilha: os operandos são sempre retirados do topo da pilha.

Duas instruções manipulam a pilha: PUSH E POP

**PUSH** – insere um dado no topo da pilha

**POP** – remove o dado no topo da pilha

$$A = ((B + C) \times D + E - F)$$

Endereço	Instrução	Comentário
e1	PUSH F	Coloca F no topo da pilha
e2	PUSH E	Coloca E no topo da pilha
e3	PUSH D	Coloca D no topo da pilha
e4	PUSH C	Coloca C no topo da pilha
e5	PUSH B	Coloca B no topo da pilha
e6	ADD	Topo da pilha recebe B + C (B e C são retirados da pilha)
e7	MUL	Topo recebe (B+C) * D
e8	ADD	Topo recebe (B+C) * D + E
e9	SUB	Topo recebe (B+C) * D + E - F
e10	POP A	Topo da pilha é armazenado em A
e11	HALT	Fim do programa



