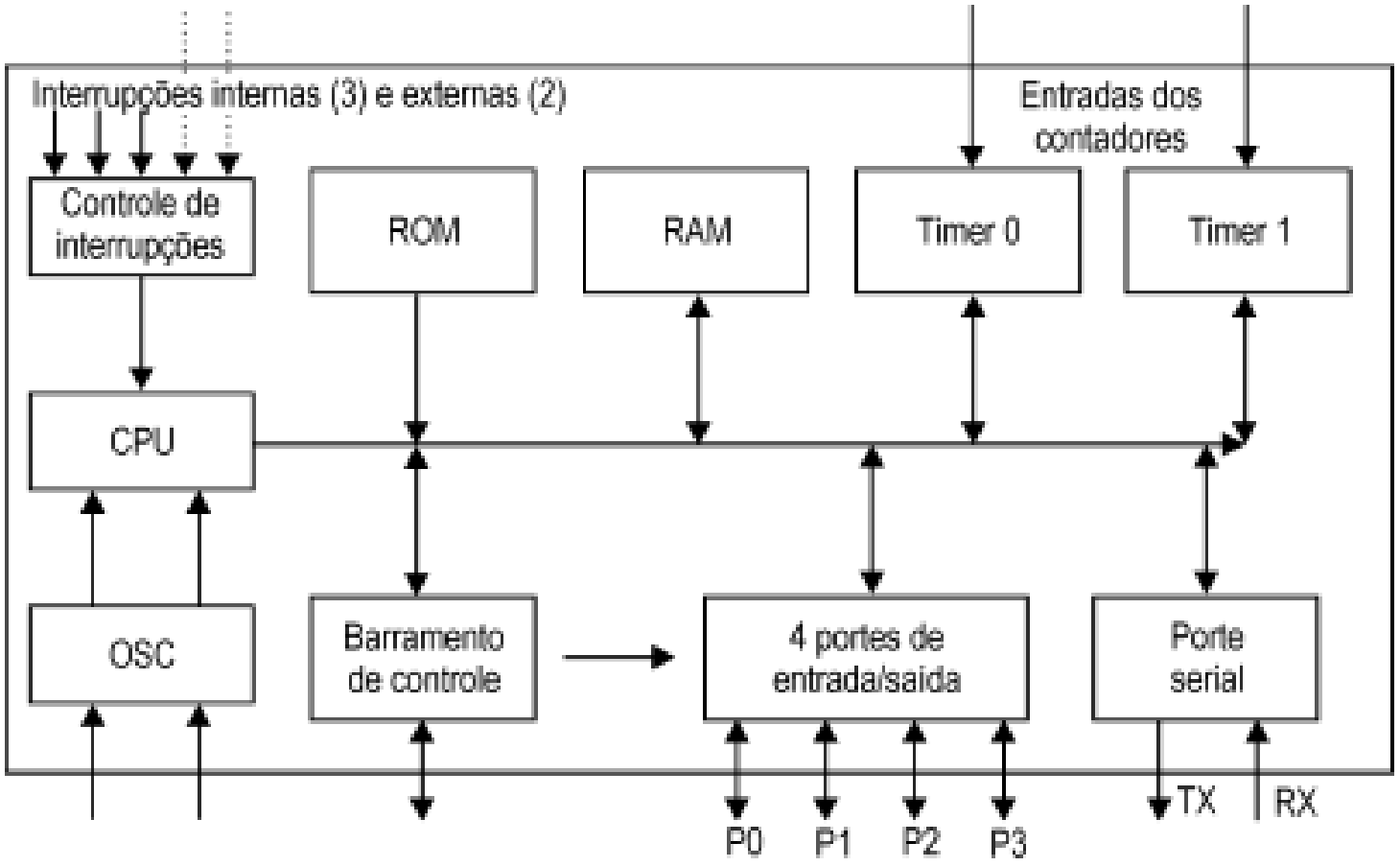
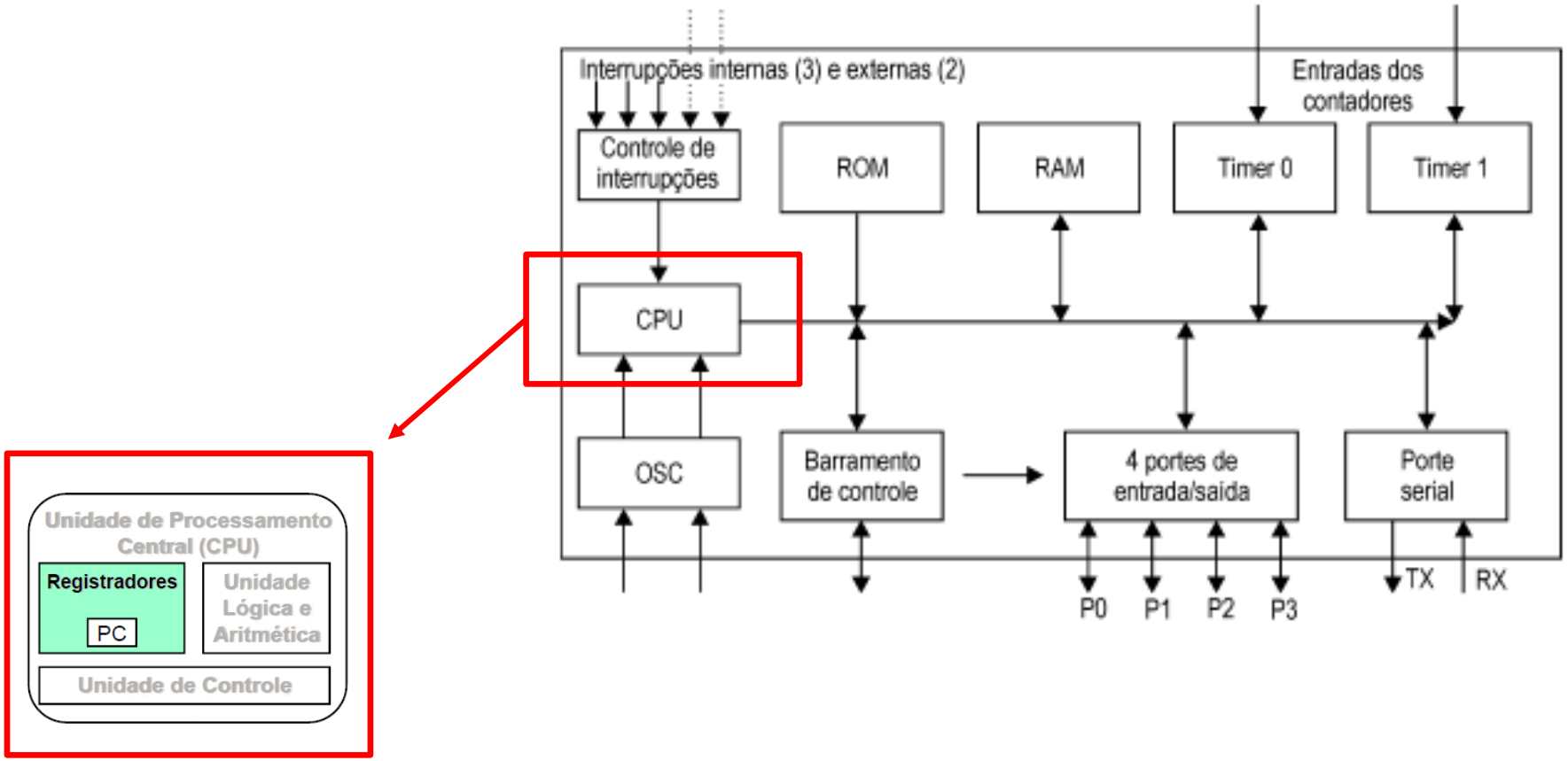
Arquitetura de Computadores

PROF. DR. ISAAC

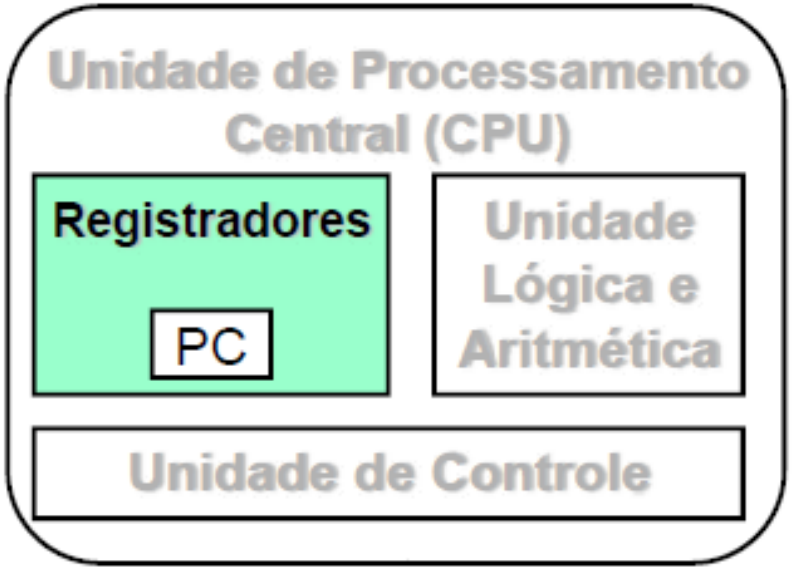
Microcontrolador 8051



Microcontrolador 8051



Microcontrolador 8051



CPU do microcontrolador

8051 -Registradores

- Usados para armazenar temporariamente informações enquanto os dados estão sendo processados.
- São as estruturas de memória mais rápidas e caras.
- Registradores mais comuns:
  - A, B, R0 - R7: registradores de 8 bits.
  - DPTR : [DPH:DPL] Registrador de 16 bits.
  - PC : Contador do Programa–16 bits.
  - 4 conjuntos de bancos de registradores R0-R7.
  - Ponteiro da pilha – SP.
  - PSW: Program Status Word (flags).
  - SFR : Special Function Registers. Controla os periféricos onboard.

8051 -Registradores

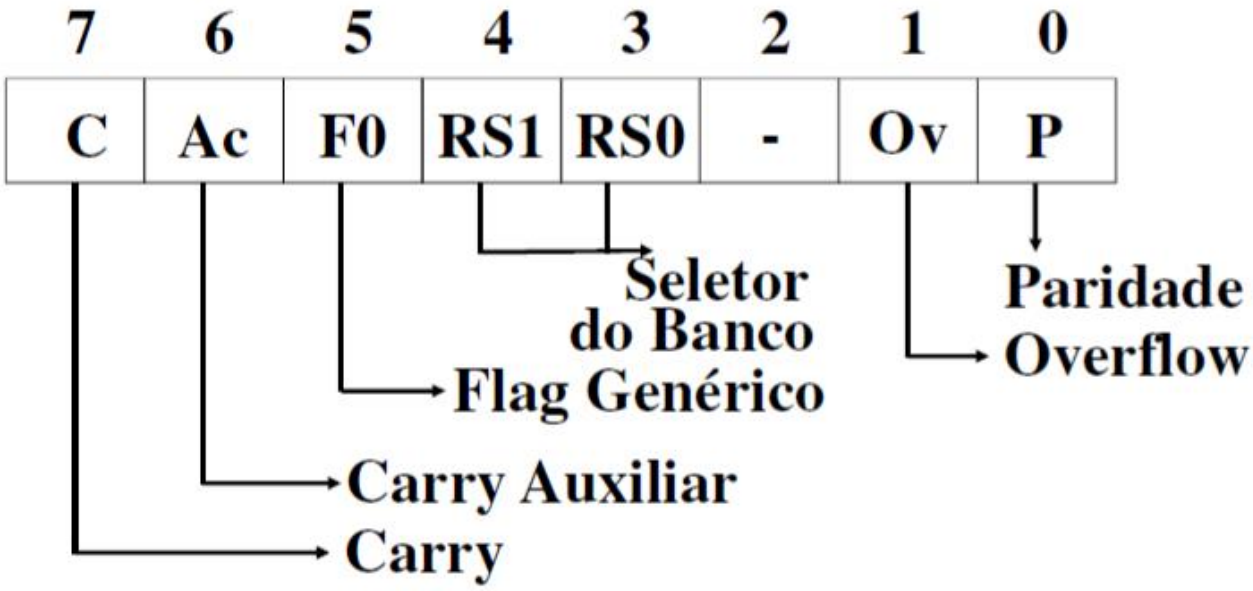
- Usados para armazenar temporariamente informações enquanto os dados estão sendo processados.
- São as estruturas de memória mais rápidas e caras.
- Registradores mais comuns:
  - A, B, R0 - R7: registradores de 8 bits.
  - DPTR : [DPH:DPL] Registrador de 16 bits.
  - PC : Contador do Programa–16 bits.
  - 4 conjuntos de bancos de registradores R0-R7.
  - Ponteiro da pilha – SP.
  - PSW: Program Status Word (flags).
  - SFR : Special Function Registers. Controla os periféricos onboard.

8051 –Registrador PSW

Todo processador possui um registrador especial onde ficam armazenadas informações sobre o estado do processamento e também sobre a última operação realizada pela unidade de lógica e aritmética. Usam-se vários nomes para designá-lo, sendo o mais comum Registrador de Flags.

Na arquitetura 8051, ele recebe o nome **Palavra de Estado do Programa,** representado pela sigla **PSW (Program Status Word).**

8051 –Registrador PSW

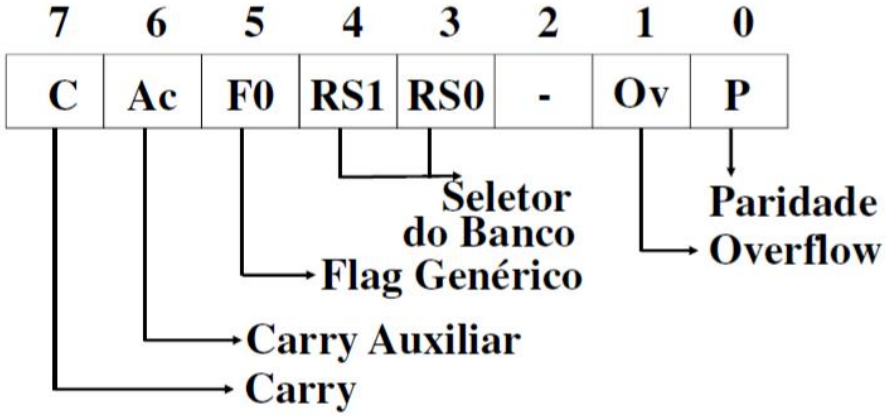


8051 –Registrador PSW

P é o bit de paridade gerado a partir do conteúdo do acumulador.

- P = 1 indica uma quantidade ímpar de bits “1”.
- P = 0 indica uma quantidade par de bits “1”.

Ov é o bit de overflow, ou estouro.

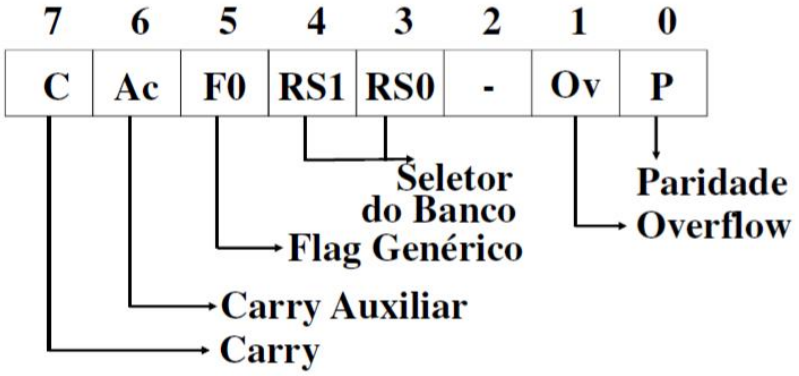


8051 –Registrador PSW

F0 é um bit de uso genérico.

Ac é o carry auxiliar, correspondendo ao “vai-um” do bit 3 para o bit 4. Ele é empregado pela instrução de ajuste decimal (DA A), que deve ser executada logo após uma soma de números representados em BCD.

C é o carry das operações lógicas e aritméticas. Tem uso nas mais diversas operações e corresponde também ao “vai-um” do bit 7 para o bit 8 nas operações de soma.



Instrução - ADD

Operação: ADD

Função: Adiciona o valor do operando ao valor do acumulador.

Sintaxe: ADD A, *operando*

Descrição : ADD adiciona o valor do operando ao valor do acumulador.

- **Carry bit (C)** – setado se existe um carry saindo do bit 7. Em outras palavras, se a soma do valor no acumulador e o operando excede 255.
- **Overflow (OV)** – setado se existir um carry saindo do bit 6 ou do bit 7, mas não dos dois. Em outras palavras, se a soma do acumulador e operando excede a faixa do valor armazenado em um byte com sinal (-128 até +127) o OV é setado, caso contrário, seu valor estará em 0.

Exemplo:

- ADD A, #03h
- ADD A, R0
- ADD A, @R1

Instrução - ADD

Operação: ADD

Função: Adiciona o valor do operando ao valor do acumulador.

Sintaxe: ADD A, *operando*

Descrição : ADD adiciona o valor do operando ao valor do acumulador.

- **Carry bit (C)** – setado se existe um carry saindo do bit 7. Em outras palavras, se a soma do valor no acumulador e o operando excede 255.
- **Overflow (OV)** – setado se existir um carry saindo do bit 6 ou do bit 7, mas não dos dois. Em outras palavras, se a soma do acumulador e operando excede a faixa do valor armazenado em um byte com sinal (-128 até +127) o OV é setado, caso contrário, seu valor estará em 0.

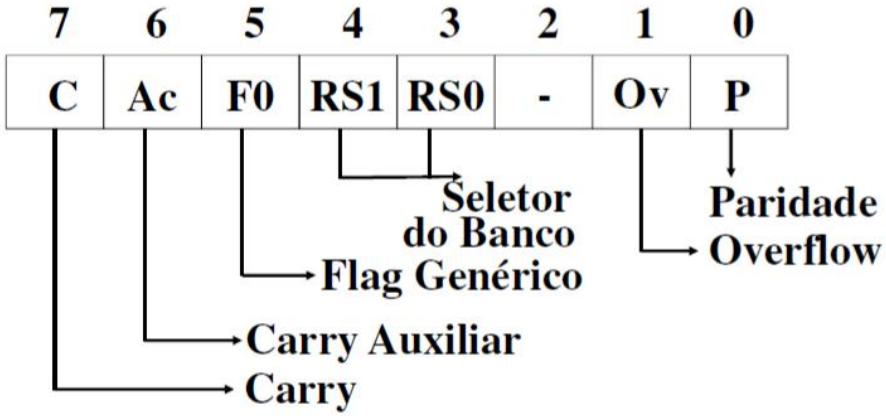
Exemplo:

- ADD A, #03h
- ADD A, R0
- ADD A, @R1

8051 –Registrador PSW

RS1:RS0 permitem que o programador selecione o banco de registrador:

RS1	RS0	Banco Selecionado
0	0	Banco 0
0	1	Banco 1
1	0	Banco 2
1	1	Banco 3



8051 –Registrador PSW

A arquitetura disponibiliza ao programador 4 conjuntos de registradores.

RS1	RS0
0	0
7: R7	
6: R6	
5: R5	
4: R4	
3: R3	
2: R2	
1: R1	
0: R0	

Banco 0

RS1	RS0
0	1
F: R7	
E: R6	
D: R5	
C: R4	
B: R3	
A: R2	
9: R1	
8: R0	

Banco 1

RS1	RS0
1	0
17: R7	
16: R6	
15: R5	
14: R4	
13: R3	
12: R2	
11: R1	
10: R0	

Banco 2

RS1	RS0
1	1
1F: R7	
1E: R6	
1D: R5	
1C: R4	
1B: R3	
1A: R2	
19: R1	
18: R0	

Banco 3

Mapeamento dos bancos de registradores sobre a RAM interna (endereços em hexadecimal).

Instruções do MSC-51

Convenções empregadas no estudo do conjunto de instruções.

Símbolo	Significado
Rn	Qualquer um dos registradores: R0, R1, R2, R3, R4, R5, R6, R7.
@Ri	Qualquer um dos registradores: R0, R1.
#dt8	Um número de 8 bits.
#dt16	Um número de 16 bits.
end8	Um endereço de 8 bits, faz referência à RAM interna.
end11	Um endereço de 11 bits, faz referência à memória de dados externa.
end16	Um endereço de 16 bits, faz referência à memória de dados externa.
rel	Um deslocamento relativo de 8 bits, em complemento 2: de –128 a +127.
bit	Endereço de um bit da RAM interna (da área acessível bit a bit).
A	Acumulador.
Acc	Endereço do acumulador (E0H).

Instruções do MSC-51

Tipo	Quantidade
Aritméticas	24
Lógicas	25
Transferência (cópia) de Dados	28
Booleanas	17
Saltos	17

Instruções aritméticas: envolvem operações do tipo soma, subtração, multiplicação, divisão, incremento e decremento.

Instruções lógicas: fazem operações bit a bit com registradores e também rotações.

Instruções do MSC-51

Instruções de transferência (cópia) de dados: copiam bytes entre os diversos registradores e a RAM interna.

Instruções booleanas: essas instruções são denominadas booleanas porque trabalham com variáveis lógicas (variável de 1 bit). Como o próprio nome sugere, elas são talhadas para resolver expressões booleanas.

Instruções de salto: desviam o fluxo de execução do programa, chamam subrotinas, fazem desvios condicionais e executam laços de repetição.

Instruções do 8051

Lógica	Aritmética	Memória	Outros
ANL	ADD ✓	MOV ✓	NOP
ORL	ADDC	MOVC	RET e RETI
XRL	SUBB	MOVX	ACALL e LCALL
CLR	MUL	PUSH	JMP
CPL	DIV	POP	AJMP
RL	INC ✓	XCH	LJMP
RLC	DEC ✓	XCHD	SJMP
RR	DA		JB e JNB
RRC			JZ e JNZ
SWAP			JC e JNC
SETB			JBC
			DJNZ
			CJNE

Instruções do 8051

Lógica	Aritmética	Memória	Outros
ANL	ADD ✓	MOV ✓	NOP
ORL	ADDC	MOVC	RET e RETI
XRL	SUBB	MOVX	ACALL e LCALL
CLR	MUL	PUSH	JMP
CPL	DIV	POP	AJMP
RL	INC ✓	XCH	LJMP
RLC	DEC ✓	XCHD	SJMP
RR	DA		JB e JNB
RRC			JZ e JNZ
SWAP			JC e JNC
SETB			JBC
			DJNZ
			CJNE

Instruções Lógicas

Instrução - ANL

Operação: ANL

Função: AND bit a bit

Sintaxe: ANL *operando1*, *operando2*

Descrição : ANL realiza um AND entre o *operando1* e *operando2*, deixando o resultado no *operando2*.

Exemplo:

- ANL A, #10H
- ANL 20h, #00H

Instrução - ANL

		Bytes	MC	Op1	Op2	Op3
ANL A,	Rn	1	1	58+n	-	-
	end8	2	1	55	end8	-
	@Ri	1	1	56+i	-	-
	#dt8	2	1	54	dt8	-
ANL end8,	A	2	1	52	end8	-
	#dt8	3	2	53	end8	dt8

Instruções para realizar o AND Lógico.

***MC (Machine Cycle) ciclos de máquina.**

Instrução - ORL

Operação: ORL

Função: OR bit a bit

Sintaxe: ORL *operand1*, *operand2*

Descrição : ORL realiza um OU lógico bit a bit entre o *operando1* e *operando2*, deixando o resultado no *operando2*.

Exemplo:

- ORL A, #10H
- ORL 20h, #00H

Instrução - ORL

		Bytes	MC	Op1	Op2	Op3
ORL A,	Rn	1	1	48+n	-	-
	end8	2	1	45	end8	-
	@Ri	1	1	46+i	-	-
	#dt8	2	1	44	dt8	-
ORL end8	A	2	1	42	end8	-
	#dt8	3	2	43	end8	dt8

Instruções para realizar o OR Lógico.

***MC (Machine Cycle) ciclos de máquina.**

Instrução - XRL

Operação: XRL

Função: OU exclusivo bit a bit

Sintaxe: XRL *operando1*,*operando2*

Descrição : XRL realiza um ou exclusivo entre o *operando1* e o *operando2*, deixando o resultado no *operando1*. Um ou exclusivo compara os valor dos bits de cada operando e seta o bit correspondente se um dos bits (mas não os dois) dos operandos estiver setado.

Exemplo:

- **XRL A, #10H**

Instrução - XRL

		Bytes	MC	Op1	Op2	Op3
XRL A,	Rn	1	1	68+n	-	-
	end8	2	1	65	end8	-
	@Ri	1	1	66+i	-	-
	#dt8	2	1	64	dt8	-
XRL end8	A	2	1	62	end8	-
	#dt8	3	2	63	end8	dt8

Instruções para realizar o XOR Lógico.

***MC (Machine Cycle) ciclos de máquina.**

Exemplos com as instruções ANL, ORL e XRL

Zerar os bits 3, 5 e 6 do acumulador	Ativar os bits 3, 5 e 6 do acumulador	Inverter os bits 3, 5 e 6 do acumulador
`ANL A,#1001 0111B`	`ORL A,#0110 1000B`	`XRL A,# 0110 1000B`
X X X X X X X X **1 0 0 1 0 1 1 1** **X 0 0 X 0 X X X**	**X X X X X X X X** **0 1 1 0 1 0 0 0** **X 1 1 X 1 X X X**	**X X X X X X X X** **0 1 1 0 1 0 0 0** **X $\overline{X}$ $\overline{X}$ X $\overline{X}$ X X X**

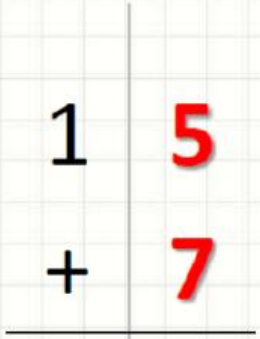
Neste exemplo a operação AND serve para zerar bits, a operação OR serve para ativar (colocar em 1) bits e a operação XOR serve para inverter bits.

Instruções Aritméticas

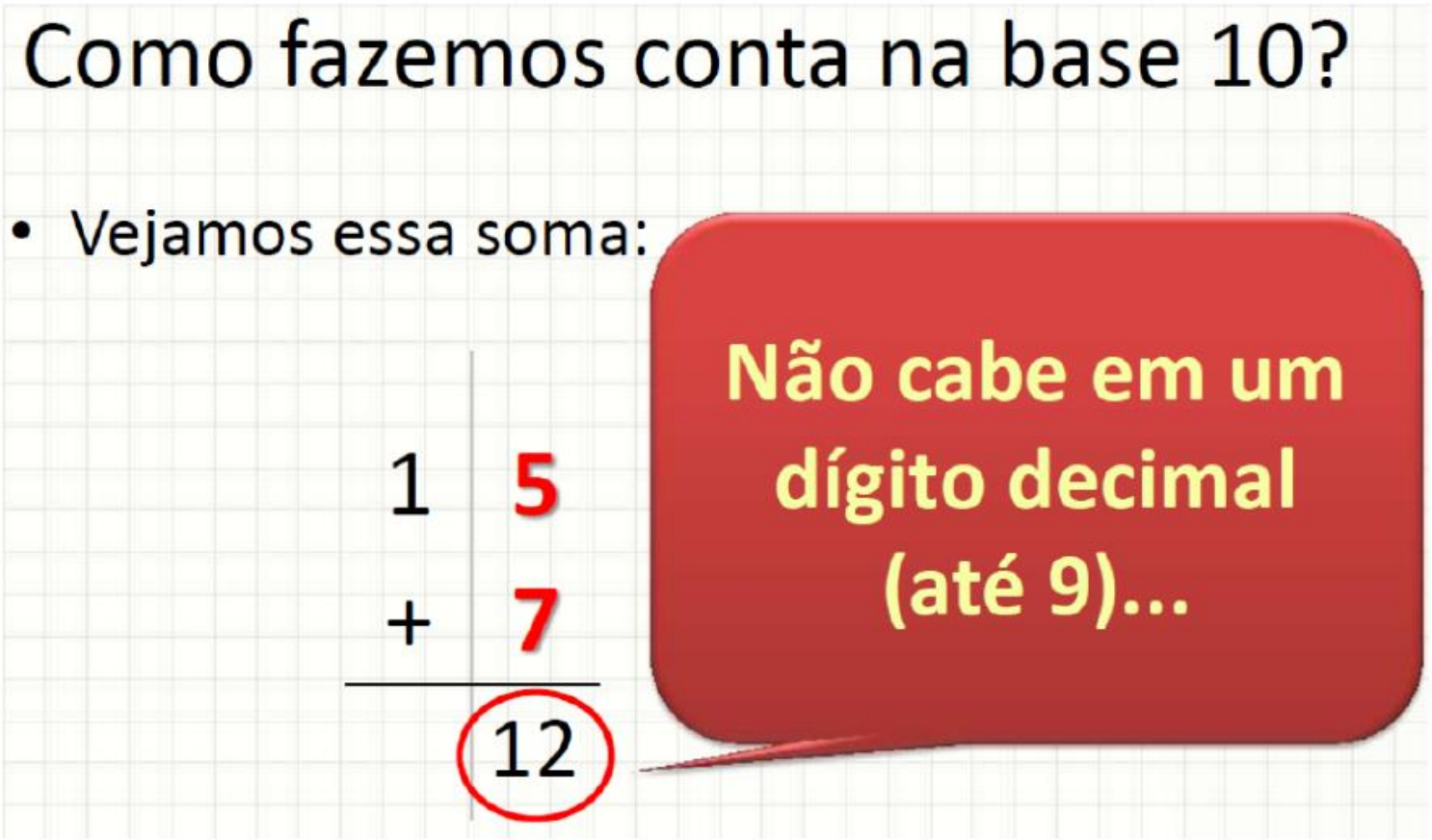
Aritmética binária

Como fazemos conta na base 10?

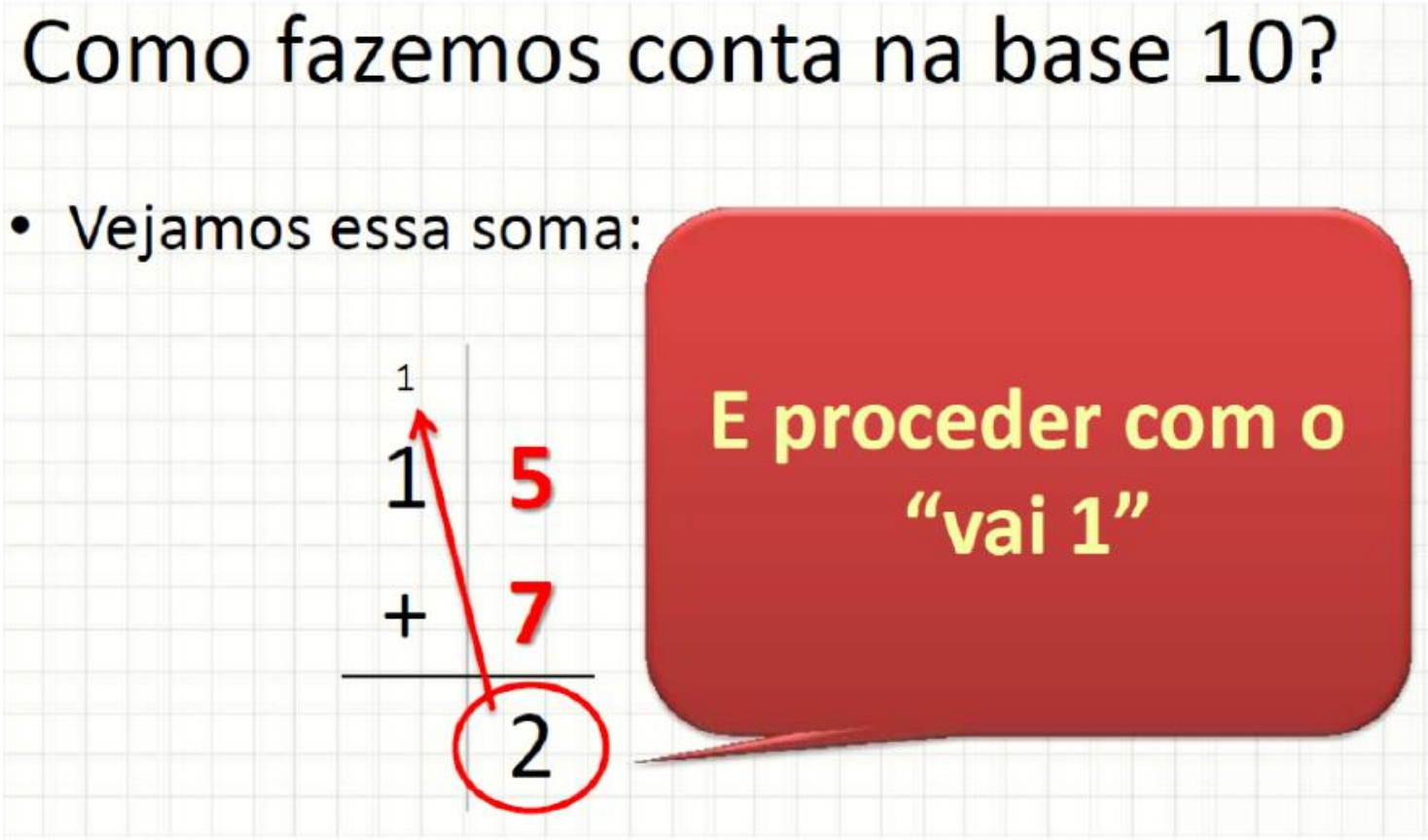
- Vejamos essa soma:



Aritmética binária



Aritmética binária



Aritmética binária

As operações aritméticas com binários podem ser feitas da mesma forma que fazemos com os decimais.

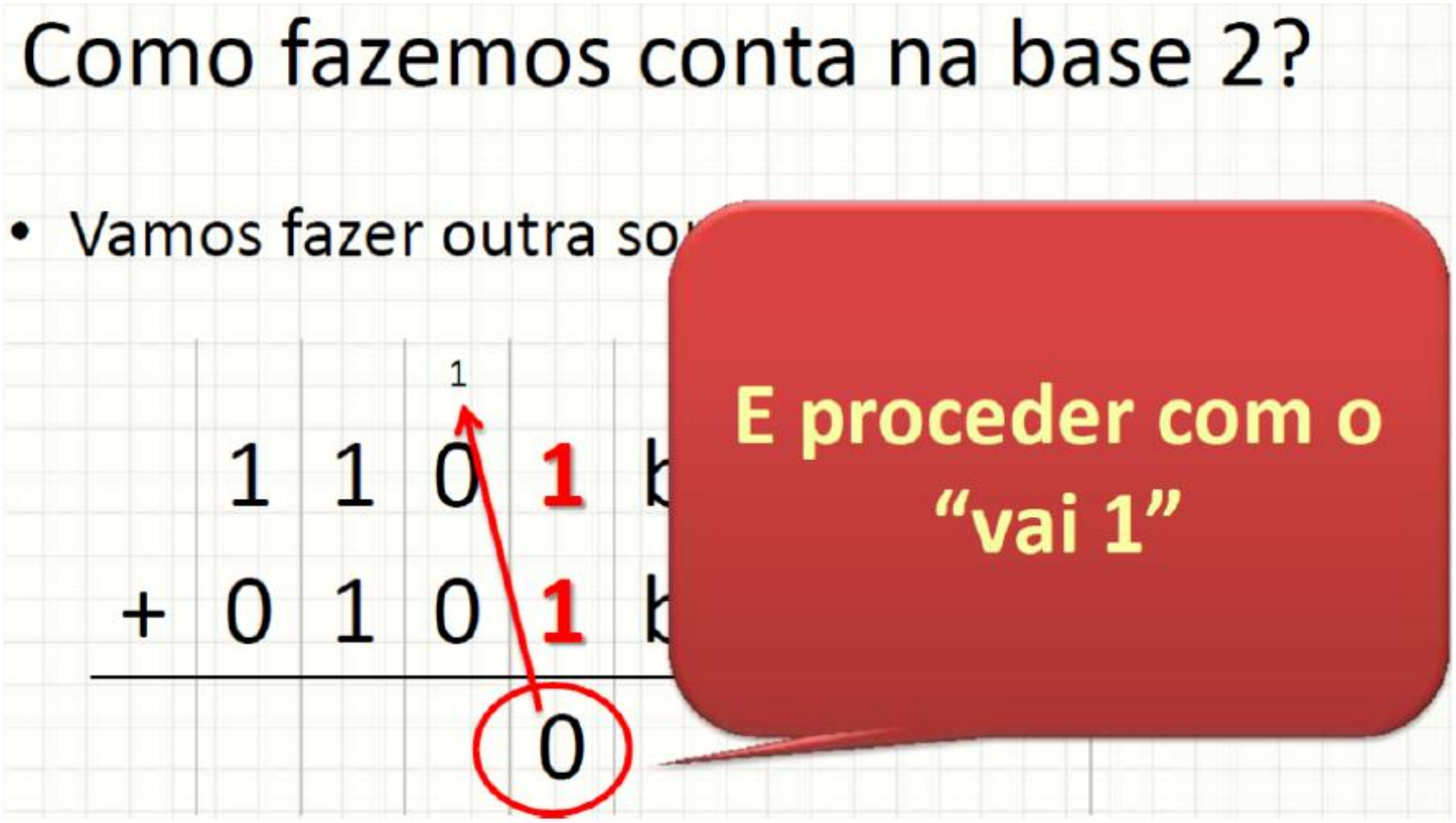
Aritmética binária

Como fazemos conta na base 2?

- Vamos fazer outra soma, agora em binário:

	1	1	0	1	b	1	3
+	0	1	0	1	b	+ 5	

Aritmética binária



Aritmética binária

Como fazemos conta na base 2?

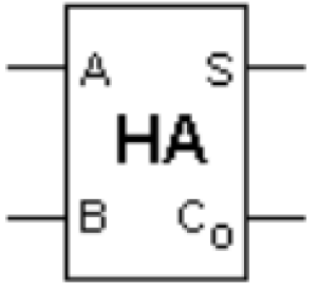
- Vamos fazer outra soma, agora em binário:

1	1		1					
	1	1	0	1	b		1	3
+	0	1	0	1	b		+	5
1	0	0	1	0				

Meio Somador

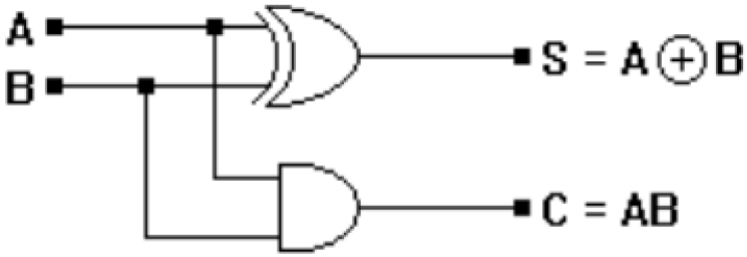
O meio somador (do inglês Half-Adder - **HA**) é utilizado para somar os primeiros bits menos significativos, onde **não há carrier anterior para somar**.

A	B	Soma	*Carry*
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1



Meio somador.

O circuito lógico correspondente é:

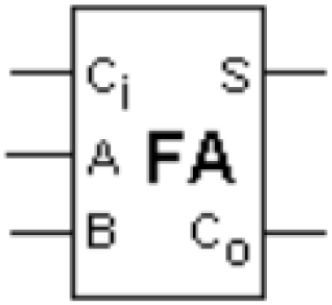


Meio somador.

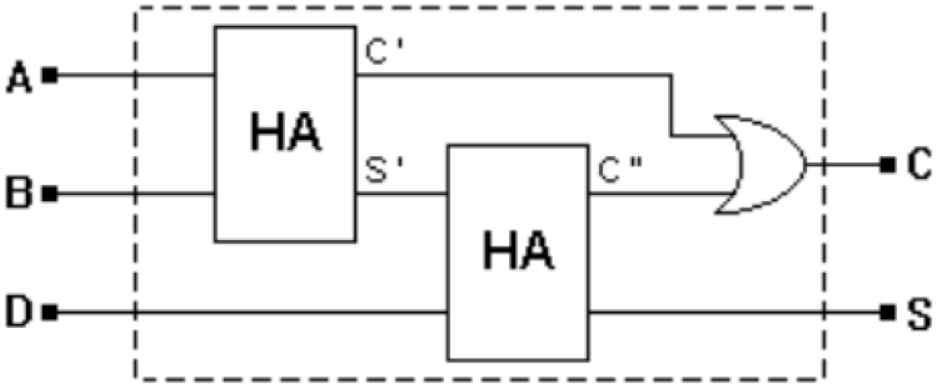
Somador Completo

O somador completo (do inglês Full-Adder - **FA**) soma os bits mais o carry anterior e pode ser obtido a partir de dois HA.

C_{in}	A	B	Soma	C_{out}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1



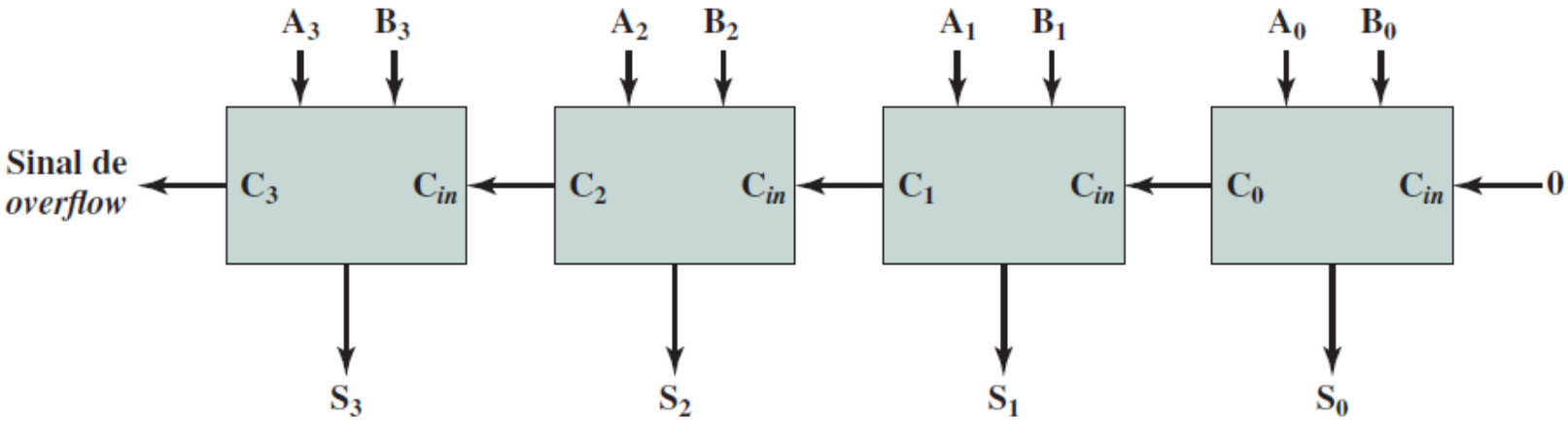
Somador inteiro.



Somador inteiro.

Somador Completo

Somador cascateado, recebe dois binários de 4 bits e retorna a soma em 4 bits e o bit de carry.



Instrução - ADDC

Operação: ADDC

Função: Adiciona ao acumulador mais o carry.

Sintaxe: ADDC A, *operando*

Descrição : Esta instrução orienta ao μC para fazer a soma do acumulador com o operando especificado, mais o carry, sendo que o resultado é colocado no acumulador. Esta instrução é usada para se fazerem somas com números inteiros representados com mais de 8 bits.

Exemplo:

- ADDC A, #03h
- ADDC A, R0
- ADDC A, @R1

Instrução - ADDC

		Bytes	MC	Op1	Op2
ADDC A,	Rn	1	1	38+n	-
	end8	2	1	35	end8
	@Ri	1	1	36+i	-
	#dt8	2	1	34	dt8

Instruções de soma aritmética com o carry.

***MC (Machine Cycle) ciclos de máquina.**

Instrução - SUBB

Operação: SUBB

Função: Subtrai do acumulador com empréstimo

Sintaxe: SUBB A, *operando*

Descrição : SUBB subtrai o valor do *operand* o do valor do acumulador, deixando o reultado no acumulador.

Carry (C) é setado se um empréstimo aconteceu com o bit 7, caso contrário seu valor será 0. Em outras palavras, se o valor sem sinal sendo subtraído é maior que o acumulador o flag de carry será setado.

Carry Auxiliar (AC) é setado se houver um empréstimo do bit 3, caso contrário ele será colocado em 0.

Overflow (OV) será setado se houve um empréstimo no bit 6 ou no bit 7, mas não nos dois. Em outras palavras, a subtração de dois valores com sinal resultou em uma valor for a da faixa de um byte com sinal (-128 a 127). Caso contrário o bit será limpo

Exemplo:

- SUBB A, #10H

Instrução - SUBB

		Bytes	MC	Op1	Op2
SUBB A,	Rn	1	1	98+n	-
	end8	2	1	95	end8
	@Ri	1	1	96+i	-
	#dt8	2	1	94	dt8

Instruções de subtração aritmética com o borrow.

***MC (Machine Cycle) ciclos de máquina.**

Instrução - MUL

Operação: MUL

Função: Multiplica o Acumulador por B

Sintaxe: MUL AB

Descrição : Multiplica o conteúdo (sem sinal) do acumulador pelo valor sem sinal de B. O byte menos significativo do resultado é armazenado no acumulador e o mais significativo em B.

Carry (C) é sempre limpo.

Overflow Flag (OV) será setado se o resultado for maior que 255, caso contrário seu valor será 0.

Exemplo:

- MUL AB

Instrução - DIV

Operação: DIV

Função: Divide o acumulador por B

Sintaxe: DIV AB

Descrição : Divide o valor sem sinal do acumulador pelo valor sem sinal do registrador B. O resultado é armazenado em A e o resto em B.

Carry (C) é sempre levado a 0.

Overflow flag (OV) é setado se houve uma tentativa de divisão por 0, caso contrário seu valor será 0.

Exemplo:

- DIV AB

Instruções – MUL e DIV

Multiplicação ⇒ A x B → Resultado: A ⇐ LSB, B ⇐ MSB;
Divisão ⇒ A / B → Resultado: A ⇐ quociente, B ⇐ resto.

Observação: na divisão, se B = 0, o resultado de A e B é indeterminado e a flag de overflow é ativada, ou seja, Ov = 1. Caso contrário, Ov = 0.

		Bytes	MC	Op
MUL	AB	1	4	A4
DIV	AB	1	4	84

Instruções para multiplicação e divisão de 8 bits.

***MC (Machine Cycle) ciclos de máquina.**

Instruções Lógicas:
Operações Lógicas com o Acumulador

Instruções de operações lógicas com o acumulador

CLR A: inicializa o acumulador com zeros;

CPL A: calcula o complemento 1 do acumulador (inverter todos os bits);

RL A: roda o acumulador à esquerda;

RLC A: roda o acumulador à esquerda, usando o carry;

RR A: roda o acumulador à direita;

RRC A: roda o acumulador à direita, usando o carry;

SWAP A: troca de posição as nibbles do acumulador.

Instruções de operações lógicas com o acumulador

		Bytes	MC	Op
CLR	A	1	1	E4
CPL				F4
RL				23
RLC				33
RR				03
RRC				13
SWAP				C4

Instruções para operações lógicas com o acumulador.

***MC (Machine Cycle) ciclos de máquina.**

		bytes	MC	Op1	Op2	Op3
MOV A,	Rn	1	1	E8+n	-	-
	end8	2	1	E5	end8	-
	@Ri	1	1	E6+i	-	-
	#dt8	2	1	74	dt8	-
MOV Rn,	A	1	1	F8+n	-	-
	end8	2	2	A8+n	end8	-
	#dt8	2	1	78+n	dt8	-
MOV end8,	A	2	1	F5	end8	-
	Rn	2	2	88+n	end8	-
	end8	3	2	85	end8 (fonte)	end8 (destino)
	@Ri	2	2	86+i	end8	-
	#dt8	3	2	75	end8	dt8
MOV @Ri	A	1	1	F6+i	-	-
	end8	2	2	A6+i	end8	-
	#dt8	2	1	76+i	dt8	-
MOV DPTR	#dt16	3	2	90	MSB(dt16)	LSB(dt16)

		Bytes	MC	Op1	Op2
ADD A,	Rn	1	1	28+n	-
	end8	2	1	25	end8
	@Ri	1	1	26+i	-
	#dt8	2	1	24	dt8

		Bytes	MC	Op1	Op2
ADDC A,	Rn	1	1	38+n	-
	end8	2	1	35	end8
	@Ri	1	1	36+i	-
	#dt8	2	1	34	dt8

		Bytes	MC	Op1	Op2
SUBB A,	Rn	1	1	98+n	-
	end8	2	1	95	end8
	@Ri	1	1	96+i	-
	#dt8	2	1	94	dt8

		Bytes	MC	Op
MUL	AB	1	4	A4
DIV	AB	1	4	84

		Bytes	MC	Op1	Op2	Op3
ANL A,	Rn	1	1	58+n	-	-
	end8	2	1	55	end8	-
	@Ri	1	1	56+i	-	-
	#dt8	2	1	54	dt8	-
ANL end8,	A	2	1	52	end8	-
	#dt8	3	2	53	end8	dt8

		Bytes	MC	Op1	Op2
DEC	A	1	1	14	-
	Rn	1	1	18+n	-
	end8	2	1	15	end8
	@Ri	1	1	16+i	-

		Bytes	MC	Op1	Op2
INC	A	1	1	04	-
	Rn	1	1	08+n	-
	end8	2	1	05	end8
	@Ri	1	1	06+i	-

	Bytes	MC	Op
INC DPTR	1	2	A3

		Bytes	MC	Op
CLR	A	1	1	E4

		Bytes	MC	Op1	Op2	Op3
ORL A,	Rn	1	1	48+n	-	-
	end8	2	1	45	end8	-
	@Ri	1	1	46+i	-	-
	#dt8	2	1	44	dt8	-
ORL end8	A	2	1	42	end8	-
	#dt8	3	2	43	end8	dt8

		Bytes	MC	Op1	Op2	Op3
XRL A,	Rn	1	1	68+n	-	-
	end8	2	1	65	end8	-
	@Ri	1	1	66+i	-	-
	#dt8	2	1	64	dt8	-
XRL end8	A	2	1	62	end8	-
	#dt8	3	2	63	end8	dt8

Exercícios

1) Some dois números de 8 bits. Armazene em R7 o resultado da soma de R1 com R0.

decimal	hexadecimal
R1 = 20	R1 = 14H
R0 = 30 +	R0 = 1EH +
R7 = 50	R7 = 32H

(a) Soma sem resultar em carry (C=0).

decimal	hexadecimal
R1 = 200	R1 = C8H
R0 = 100 +	R0 = 64H +
	C →
R7 = 300	R7 = 1 2CH

(b) Soma resultando em carry (C=1).

Realize os exercícios observando a flag carry do registrador PSW

Exercícios

2) Desenvolva um código para as operações a seguir:

a) Mova para os registradores R0 e R7 do banco 01 o valor de FFh;

b) Mova para os registradores R0 e R7 do banco 02 o valor de AAh;

c) Mova para os registradores R0 e R7 do banco 03 o valor de BBh;

Respostas exercício

a)

```
MOV PSW, #00001000
MOV R0, #0FFh
MOV R7, #0FFh
```

b)

```
MOV PSW, #00010000
MOV R0, #0AAh
MOV R7, #0AAh
```

c)

```
MOV PSW, #00011000
MOV R0, #0BBh
MOV R7, #0BBh
```

Exercícios

Exercício 3:

Multiplique dois números de 8 bits. Armazene em [R5 R4] o resultado da multiplicação de R1 por R2. Ou seja, faça [R5 R4] ← R1 x R2.

Exercícios

Exercício 4:

Divida dois números de 8 bits. Armazene em R6 o quociente e em R7 o resto da divisão de R3 por R4, supondo R3 ≠ 0. Segundo a notação, faça: R6(q) e R7(r) ← R3 / R4.

Bibliografia

ZELENOVSKY, R.; MENDONÇA, A. Microcontroladores Programação e Projeto com a Família 8051. MZ Editora, RJ, 2005.

Gimenez, Salvador P. Microcontroladores 8051 - Teoria e Prática, Editora Érica, 2010.