

1. ESPECIFICAÇÃO CLEÓPATRA 2.2 - RESUMO

Registadores:

1 Registrador Acumulador		AC
1 Contador de Programa		PC
1 Registrador de Subrotina		RS
4 Biestáveis de Estado -	<i>Vai-um</i>	C
	<i>Transbordo</i>	V
	<i>Negativo</i>	N
	<i>Zero</i>	Z

Modos de Endereçamento:

- **00 - Imediato** (Operando é o dado de 8 bits junto à instrução);
- **01 - Direto** (Operando é endereço de memória de 8 bits. Diz onde se encontra o dado);
- **10 - Indireto** (Operando é endereço de memória de 8 bits. Diz onde se encontra o endereço do dado);
- **11 - Relativo** (Operando é deslocamento de 8 bits em complemento de 2, para ser adicionado ao valor atual do PC, gerando o endereço do dado).

Instruções:

Código	Mnemônico	Operação
000x	NOT	Complementa (inverte) todos os bits de AC
001x	STA oper	Armazena AC na memória dada por oper
0100	LDA oper	Carrega AC com conteúdos de memória
0101	ADD oper	Adiciona AC a conteúdo da memória dada por oper
0110	OR oper	Faz OU lógico do AC com conteúdo da memória dada por oper
0111	AND oper	Faz E lógico do AC com conteúdo da memória dada por oper
1000	JMP oper	PC recebe dado especificado por oper
1001	JC oper	Se C=1 então PC recebe valor dado por oper
1110	JV oper	Se V=1 então PC recebe valor dado por oper
1010	JN oper	Se N=1 então PC recebe valor dado por oper
1011	JZ oper	Se ZR=1 então PC recebe valor dado por oper
1100	JSR oper	RS recebe conteúdo de PC, PC recebe dado de oper
1101	RTS	PC recebe conteúdos de RS
1111	HLT	Termina ciclo de busca e execução

Observações

- **oper** - operando que depende do modo de endereçamento para ser determinado;
- Instruções lógicas (NOT, AND e OR) e LDA - afetam os códigos de condição N e Z;
- ADD - Afeta C, V, N e Z;
- Instruções restantes (STA, JMP, JC, JV, JN, JZ, JSR, RTS, HLT) - não alteram códigos de condição.
- Nas instruções sem operando (NOT, RTS e HLT), o campo de modo de endereçamento não é utilizado;
- Para as instruções de desvio (JMP, JC, JV, JN, JZ, JSR) e STA, os modos de endereçamento direto e imediato são idênticos (possuem o mesmo efeito).

Formatos de Instrução

- Existem dois tipos de instrução: com ou sem operando.
- Instruções com operando ocupam dois bytes da memória de programa, enquanto que instruções sem operando ocupam apenas 1 byte. A seguir, descreve-se o formato da primeira palavra de qualquer instrução:

Formato da primeira palavra de uma instrução

Os quatro primeiros bits contêm o código da operação (**operation code**, ou **OPCODE**), no campo **INS**, e os dois bits seguintes especificam o Modo de Endereçamento (**addressing mode**), no campo **ME**, a ser usado. Os 2 últimos bits da primeira palavra de 1 instrução são sempre **00**, não sendo utilizados. O formato é o seguinte:

INS	ME	Não usados
XXXX	XX	00

Linguagem de Montagem (Assembly) - Sintaxe:

- Os códigos de operação são reservados, correspondendo aos 14 mnemônicos mencionados acima, no item Instruções.
- Rótulos podem denotar posições da memória de programa, quando precederem algum dos 14 mnemônicos, ou da memória de dados, quando precederem a diretiva **"DB"**;
- Rótulos são especificados por qualquer texto diferente de um mnemônico, sucedido do caracter ":";
- Comentários vão do primeiro caracter ";" até o final da linha;
- Formato para os operandos é o seguinte:
#<rótulo> ou #<número>; <rótulo> ou <número>; <rótulo>,I ou <número>,I; <rótulo>,R ou <número>,R
- Os símbolos "#", "I" e "R" indicam o modo de endereçamento a usar na interpretação do operando, correspondendo a Imediato, Indireto e Relativo, respectivamente. O modo direto é assumido se nenhum dos outros for fornecido. <rótulo> representa qualquer rótulo como especificado acima, sem o caracter delimitador ":". <número> representa um número em um de três formatos:
<numhexa>H / <numbin>B / <numdec>
- A diretiva **DB** ("define byte") não é instrução do montagem, mas uma informação ao programa montador para reservar um byte de memória de dados para uso pelo programa, quando em execução. Outras diretivas são **.CODE** e **.ENDCODE**, que delimitam o texto do programa e **.DATA** e **.ENDDATA**, que delimitam a área de dados. Finalmente, existe a diretiva **ORG**, definindo o próximo endereço de montagem

Exemplos de Programas

(Endereços da memória de programa e de dados e código objeto de cada linha colocados como comentário.)

- Exemplo: Este programa executa um algoritmo irrelevante, apenas serve para ilustrar o formato da linguagem de montagem, a tradução de linguagem de montagem para linguagem de máquina (em hexadecimal) e o funcionamento dos modos de endereçamento. Note que o número das linhas não faz parte do programa, está indicado apenas para fins didáticos.

Explicação: O programa inicia carregando o registrador acumulador AC com o conteúdo da posição 90 de memória, na linha 3 (posição inicializada com 30 pela diretiva DB na memória de dados); a este dado, a segunda instrução soma o conteúdo da posição de memória 93 (dado 5B), colocando o resultado da soma de volta no acumulador, obtendo o dado via endereçamento indireto; a terceira instrução realiza o armazenamento do conteúdo de AC na posição de memória END3 (endereço 92, inicialmente contendo 00). Após, o mesmo dado é transformado pelo zeramento de seus quatro bits mais significativos, pela instrução da linha 6, e o resultado volta à AC. Se o resultado desta operação for 0, o programa pára, senão AC tem todos os seus bits invertidos e só aí o programa pára. Identifique se com os dados fornecidos abaixo a linha 8 da memória de programa é executada ou não.

Memória de Programa:

	Rótulo	Mnem	Operando	End	Opcode	Oper
1						
	.CODE					;Início do programa
2		ORG	#00H			;Monta a partir de 00H
3	INIT:	LDA	END1	:00	44	90
4		ADD	END2,I	:02	58	91
5		STA	END3	:04	24	92
6		AND	#0FH	:06	70	0F
7		JZ	FIM,R	:08	BC	01
8		NOT		:0A	00	
9	FIM:	HLT		:0B	F0	
10	.ENDCODE					;Fim do programa

Memória de Dados

	Rótulo	Mnem	Operando	End	Dado
1					
	.DATA				;Início dos dados
2		ORG	#90H		;Monta a partir de 90H
3	END1:	DB	#30H	:90	30
4	END2:	DB	#END4	:91	93
5	END3:	DB	#00H	:92	00
6	END4:	DB	#5BH	:93	5B
7	.ENDDATA				;Fim dos dados

- Exercício: Faça um programa para somar dois vetores de n posições, onde n está armazenado numa posição de memória ao iniciar o programa.