

ARQUITETURA DE COMPUTADORES

Microprocessador INTEL 8085

Conteúdos

2

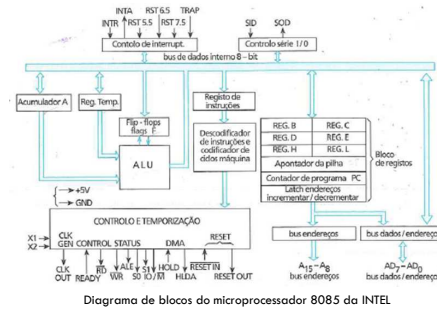
1. Organização Interna
 1. Barramentos
 2. ALU
 3. Unidade de controlo e temporização
 4. Registos
 1. Acumulador
 2. Flags
 3. Apontador da pilha
 4. Contador do programa
 5. Instruções temporárias
2. Circuito integrado
3. Modos de endereçamento
 1. Endereçamento implícito
 2. Endereçamento imediato
 3. Endereçamento por registo
 4. Endereçamento direto
 5. Endereçamento indireto por registo
4. Tipos de instruções
 1. Transferência de dados
 2. Aritméticas, lógicas e de rotação
 3. Controlo e salto
 4. Controlo do CPU, I/O e Stack

Organização interna

3

- Os principais componentes de um microprocessador são:

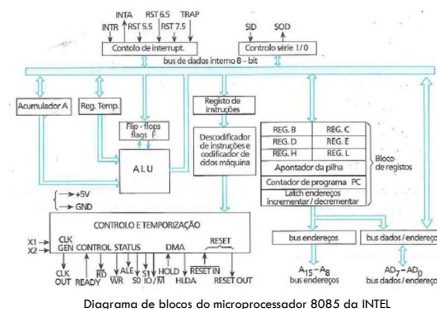
- Barramentos
- ALU (unidade lógica e aritmética)
- Unidade de controlo e temporização
- Registos



Organização interna - barramentos

4

- O barramento de endereços está adstrito às linhas de endereço A_8 a A_{15}
- As restantes linhas de endereço são servidas por um outro barramento de dados, ou seja, o microprocessador apenas dispõe de 16 linhas para servir 24 linhas, isto é, 16 linhas para os endereços e oito linhas para os dados
- Na realidade não existe um verdadeiro barramento de dados, mas um barramento multiplexado de dados e de endereços anotados por AD_0 a AD_7 .
- Como é óbvio, é necessário separar os dois conteúdos do barramento multiplexado, o que se consegue com um sinal de controlo, ALE (Address Latch Enable)



Organização interna - ALU

5

- ALU (unidade lógica e aritmética):
 - Este circuito contém toda a lógica de processamento de dados do microprocessador
 - Accede frequentemente a um registo denominado registo acumulador (A)
 - Exemplo: numa adição, uma das parcelas é colocada obrigatoriamente no registo acumulador (A), sendo o resultado enviado também para o registo acumulador, onde é, guardado temporariamente.
 - As funções possíveis são: adição, subtração, comparação, AND, OR e XOR lógicos, deslocamento para a esquerda/direita, incrementação, decrementação, etc.

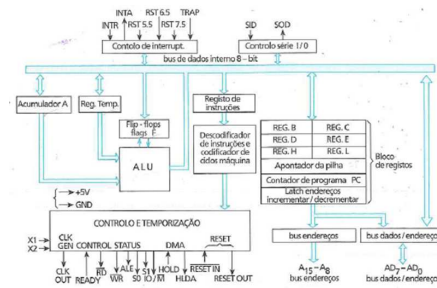


Diagrama de blocos do microprocessador 8085 da INTEL

Organização interna – unidade de controlo e temporização

6

- Unidade de controlo e temporização:
 - Trata-se da unidade que promove a correta sequência de funcionamento dos diversos blocos que compõem o microprocessador
 - O decodificador de instruções adquire as instruções do registo de instruções, determina o que deve ser feito com os dados e promove os sinais adequados para que a tarefa seja realizada
 - Desta unidade saem, para todas as unidades que constituem o microprocessador, linhas de controlo
 - É esta unidade que:
 - Dá saída ou responde aos sinais de controlo vindos do exterior
 - Controla o fluxo de dados entre o microprocessador, a memória e os dispositivos I/O

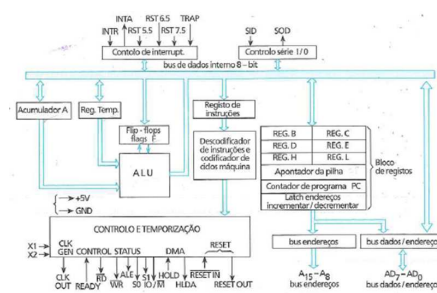
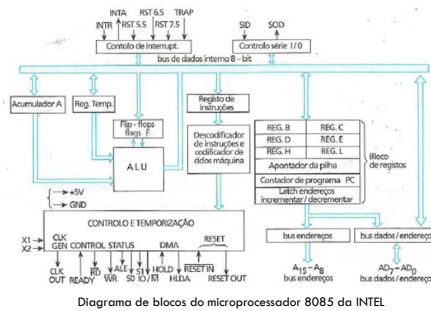


Diagrama de blocos do microprocessador 8085 da INTEL

Organização interna – registros

7

- Os registros principais são:
 - ▣ Registro acumulador (A)
 - ▣ Registro de flags (F)
 - ▣ Registos de uso geral (B, C, D, E, H e L)
- Podemos ainda observar a existência de mais registros:
 - ▣ SP (Stack pointer) – apontador da pilha
 - ▣ PC (Program Counter) – contador de programa
 - ▣ IR (Instruction Register) – registro de instruções
- Existem ainda alguns registros temporários



Organização interna – registros

8

- Registro acumulador (A):
 - ▣ É o registro mais utilizado pelo microprocessador
 - ▣ Quando se processam instruções lógicas ou aritméticas entre duas palavras, uma delas está obrigatoriamente no registro acumulador, estando a outra num registro de uso geral ou numa posição de memória
 - ▣ O resultado da operação é usualmente colocado no registro acumulador, substituindo assim o valor anterior que se encontrava guardado neste registro
 - ▣ O registro acumulador recebe e envia dados pelo barramento interno, daí o registro acumulador ter o mesmo tamanho que o barramento interno, isto é, 8 bits
 - ▣ A principal função deste registro é guardar temporariamente a informação presente na instrução

Organização interna – registros

9

□ Registro de flags (F):

- Este registro é um registro apenas de leitura com apenas 5 bits, cujos estados possíveis são **SET** (1) ou **RESET** (0)
- Há cinco flags de condição associadas à execução das instruções:
 - **Zero (Z)**: se o resultado da execução de uma instrução tem o valor zero, então faz-se o set (1) desta flag, caso contrário, é feito o reset (0)
 - **Sinal (S)**: se o bit mais significativo do resultado da operação for 1, então faz-se o set (1) desta flag, caso contrário, é feito o reset (0)
 - **Paridade (P)**: se o resultado da operação tiver paridade par, então faz-se o set (1) desta flag, caso o resultado tenha paridade ímpar, é feito o reset (0) da flag
 - **Carry (CS)**: se da execução da instrução resultar um transporte (carry na adição e borrow na subtração ou comparação) para além do bit mais significativo, então faz-se o set (1) desta flag, caso contrário é feito o reset (0)
 - **Carry auxiliar (AC)**: se a execução de uma instrução causou um transporte do bit 3 para o bit 4 do valor resultante, então faz-se o set (1) desta flag, caso contrário, é feito o reset (0). Esta flag é afetada por adições, subtrações, incrementos, decrementos, comparações e operações lógicas, mas utiliza-se principalmente com adições e incrementos que precedem a instrução DAA (ajuste decimal do acumulador)

Organização interna – registros

10

□ Registros de uso geral (B, C, D, E, H e L):

- São registros de 8 bits, que podem ser utilizados em diversas instruções
- É indiferente a escolha de qual dos registros a utilizar em certa operação
- Usualmente o par de registros HL é utilizado como apontador de memória
- Algumas instruções utilizam os registros de 8 bits agrupados aos pares, como se tratassem de registros de 16 bits, no entanto estes registros só podem ser agrupados da seguinte forma:
 - BC
 - DE
 - HL

Organização interna – registos

11

- **Registo apontador da pilha (SP):**
 - É um registo de 16 bits
 - O microprocessador dispõe de uma zona especial na memória, para aí serem guardados endereços ou dados temporariamente, a que se chama pilha (stack)
 - A informação é colocada na pilha de forma inversa à qual vai ser lida, isto é, a última informação a ser colocada na pilha será a primeira a ser lida
 - Este registo é automaticamente decrementado/incrementado de cada vez que é utilizado
 - A informação para e da pilha é obtida exclusivamente através das instruções **PUSH** e **POP**, respetivamente

Organização interna – registos

12

- **Registo contador de programa (PC):**
 - Um programa é um conjunto de instruções guardadas na memória
 - Para que o programa funcione corretamente, as diferentes instruções deverão ser executadas na devida altura e pela ordem correta
 - É a este registo que compete garantir qual e quando a instrução seguinte deve ser executada e qual a instrução que se segue
 - Para que o microprocessador possa iniciar a execução de um programa é necessário que o contador de programa seja carregado com o endereço da posição de memória que contém a primeira instrução do programa a ser executada
 - Este registo é um registo de 16 bits, uma vez que o barramento de endereços tem 16 bits, o que significa que podem ser endereçadas 64K posições de memória diferentes
 - O contador de programa incrementa automaticamente (1, 2, 3 ou 4 vezes) enquanto é executada uma instrução. O número de vezes que o contador de programa é incrementado depende do tamanho da instrução
 - Apesar do contador de programa ser incrementado automaticamente, existem instruções que permitem modificar o conteúdo deste registo e assim alterar o incremento sequencial deste registo na memória (JUMP, RESET, etc.)

Organização interna – registos

13

□ Registo de instruções (IR):

- É um local temporário de armazenamento para os códigos binários das instruções guardadas na memória
- Quando o processador procura uma instrução (*fetch*) na memória, coloca no barramento de endereços o endereço de memória onde está localizada a instrução
- Através do barramento de dados é lida a informação que está contida no conteúdo da posição de memória cujo endereço foi colocado no barramento de endereços. Essa informação binária fica armazenada no registo de instruções, até que seja procurada uma nova instrução
- O registo de instruções apenas recebe dados do barramento de dados e não pode colocar dados no barramento
- As saídas do registo de instruções estão ligadas a uma unidade denominada decodificador de instruções. Durante a execução de uma instrução o decodificador de instruções indica à unidade de controlo e temporização o que deve exatamente realizar.

Organização interna – registos

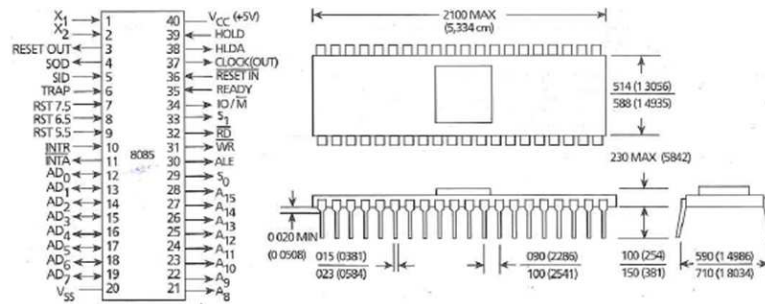
14

□ Registos temporários:

- Normalmente estes registos estão associados à unidade lógica e aritmética, uma vez que a ALU não tem capacidade de guardar informação, apenas tem a capacidade de processar informação
- Estes registos não estão acessíveis ao programador

Circuito integrado

15

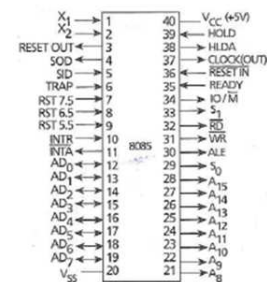


- O microprocessador INTEL 8085 é construído num circuito integrado com 40 terminais

Circuito integrado

16

Pino(s)	Designação	Descrição
12 até 19	AD ₀ até AD ₇	<ul style="list-style-type: none"> Barramento de endereço/dados multiplexado São os bits menos significativos do endereço de memória ou endereço de um dispositivo I/O (caso funcione como barramento de endereço) São os oito bits de dados (caso funcione como barramento de dados)
21 até 28	A ₈ até A ₁₅	<ul style="list-style-type: none"> Barramento de endereços São os oito bits mais significativos do endereço de memória ou endereço de um dispositivo I/O
29, 33 e 34	S ₀ , S ₁ , IO/ \overline{M}	<ul style="list-style-type: none"> Representam em conjunto o estado de funcionamento do microprocessador
30	ALE	<ul style="list-style-type: none"> Address Latch Enable Controla as entradas AD₀ até AD₇ para funcionarem como barramento de dados ou de endereços
32	\overline{RD}	<ul style="list-style-type: none"> Controlo de leitura Indica quando uma operação de leitura da memória ou de um dispositivo I/O está a ter lugar
31	\overline{WR}	<ul style="list-style-type: none"> Controlo de escrita Indica quando uma operação de escrita do CPU, para memória ou para um dispositivo I/O está a ter lugar

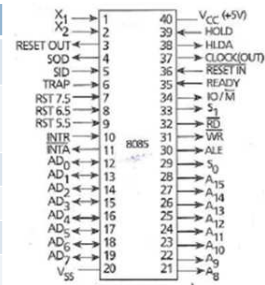


IO/ \overline{M}	S ₁	S ₀	Estado
0	0	1	Escrever na memória
0	1	0	Ler da memória
1	0	1	Escrever num dispositivo I/O
1	1	0	Ler de um dispositivo I/O
0	1	1	Procurar instrução
1	1	1	Pedido de interrupção
...

Circuito integrado

17

Pino(s)	Designação	Descrição
39	HOLD	• Indica que está a ser requisitado o uso do barramento de endereços e de dados
35	READY	• Se ficar ativo durante um ciclo de leitura ou escrita indica que a memória ou o dispositivo I/O está pronto para enviar ou receber dados
10	INTR	• Pedido de interrupção
11	\overline{INTA}	• Atendimento do pedido de interrupção
37	CLOCK(OUT)	• Saída do relógio
40	VCC	• Alimentação de 5V



Modos de endereçamento

18

- Existem cinco modos de endereçamento do microprocessador das INTEL 8085:
 - ▣ Endereçamento implícito
 - ▣ Endereçamento imediato
 - ▣ Endereçamento por registo
 - ▣ Endereçamento direto
 - ▣ Endereçamento indireto por registo

Modos de endereçamento

19

□ Endereçamento implícito:

- As instruções que utilizam este modo de endereçamento, o endereçamento está implícito na própria instrução
- Neste tipo de endereçamento, o código de operação da instrução é fixo
- Não existem campos variáveis, isto é, a origem e o destino são fixos
- Exemplo: a instrução **STC** coloca a 1ª flag de carry do registo de flags

Modos de endereçamento

20

□ Endereçamento imediato:

- As instruções que usam este tipo de endereçamento têm os dados posicionados imediatamente após o código da operação
- Exemplo: a instrução **ADI byte** soma o valor do *byte* ao registo acumulador (A) e o resultado é colocado no registo acumulador (A)

Modos de endereçamento

21

□ Endereçamento por registo:

- ▣ Nesta modalidade de endereçamento um ou vários registos do processador são endereçados
- ▣ A operação e a origem do operando são especificadas
- ▣ O formato destas instruções contém um ou vários campos que especificam quais os registos do processador a usar
- ▣ Exemplo: a instrução **ADC registo** adiciona o conteúdo do *registo* com a flag de carry e com o conteúdo do registo acumulador (A) e o resultado é colocado no registo acumulador (A)

Modos de endereçamento

22

□ Endereçamento direto:

- ▣ As instruções que usam este tipo de endereçamento têm um formato com 3 bytes:
 - 1º byte representa o código da operação
 - O 2º byte representa o byte menos significativo do endereço do operando
 - O 3º byte representa o byte mais significativo do endereço do operando
- ▣ Exemplo: a instrução **LD A,end** promove a cópia dos oito bits da posição de memória *end*, para o registo acumulador (A)

Modos de endereçamento

23

- Endereçamento indireto por registo:
 - ▣ Este tipo de endereçamento permite o endereçamento por um par de registos:
 - HL
 - DE
 - BC
 - ▣ Exemplo: a instrução **LDAX B** copia para o conteúdo do registo acumulador (A) o conteúdo da posição de memória indicada pelo par de registos BC

Tipos de instruções

24

- Existem quatro grupos de instruções no microprocessador da INTEL 8085:
 - ▣ Grupo de transferência de dados
 - ▣ Grupo aritmético, lógico e de rotação
 - ▣ Grupo de controlo e de salto
 - ▣ Grupo de controlo do CPU, I/O e da Pilha

Tipos de instruções

25

	Legenda
pr	Par de registos: HL, BC, DE, SP, PC
reg	Registo: A, B, C, D, E, H, L
M	Posição de memória
addr	Endereço de 16 bits de uma posição de memória
x	O bit do registo de flags é afetado
byte	Constante, ou expressão lógica/aritmética que representa um dado de 8 bits
double	Constante, ou expressão lógica/aritmética que representa um dado de 16 bits
[]	Conteúdo do que se encontra dentro de parênteses retos
[[]]	Conteúdo do conteúdo do que se encontra dentro de parênteses retos
CS	Flag de carry
label	Endereço de uma posição de memória
port	Endereço de um dispositivo I/O

Tipos de instruções – grupo de transferência de dados

26

- São instruções que transferem dados entre registos ou entre registos e a memória
- A origem é sempre indicada a seguir à vírgula e o destino é indicado antes da vírgula:
 - ▣ Formato das instruções: *instrução destino, origem*
- Quando se trata de uma operação com a memória, o respetivo endereço deve ser previamente carregado no par de registos HL, que funciona como ponteiro da memória
- Exemplos:
 - ▣ **MOV D,A** – esta instrução copia o conteúdo do registo acumulador (A) para o conteúdo do registo D
 - ▣ **MOV M,C** – esta instrução ao ser executada copia o conteúdo do registo C para o endereço de memória apontado pelo par de registos HL. Atenção que o uso do par de registos HL como apontador de memória obriga a que o respetivo endereço de memória seja carregado no par de registo HL (**LXI H, endereço de memória**)

Tipos de instruções – grupo de transferência de dados

27

Instrução	Operandos	Status do registro de flags					Operação realizada
		CS	AC	Z	S	P	
LDAX	pr						$[A] \leftarrow [[pr]]$ Load A using implied addressing by BC (pr=B) or DE (pr=D)
STAX	pr						$[[pr]] \leftarrow [A]$ Store A using implied addressing by BC (pr=B) or DE (pr=D)
MOV	r,M						$[r] \leftarrow [[HL]]$ Load any register using implied addressing by HL
MOV	M,r						$[[HL]] \leftarrow [r]$ Store any register using implied addressing by HL
LDA	addr						$[A] \leftarrow [addr]$ Load A using direct addressing
STA	addr						$[addr] \leftarrow [A]$ Store A using direct addressing
LHLD	addr						$[L] \leftarrow [addr]$ and $[H] \leftarrow [addr+1]$ Load H and L registers using direct addressing
SHLD	addr						$[addr] \leftarrow [L]$ and $[addr+1] \leftarrow [H]$ Store H and L registers using direct addressing
MOV	r,r						$[r] \leftarrow [r]$ Move any register to any register
XCHG							$[D] \leftrightarrow [H]$ and $[E] \leftrightarrow [L]$ Exchange DE with HL
SPHL							$[HL] \leftarrow [SP]$ Move HL to SP

Tipos de instruções – grupo de transferência de dados

28

Instrução	Operandos	Status do registro de flags					Operação realizada
		CS	AC	Z	S	P	
LXI	pr,double						$[pr] \leftarrow \text{double}$ Load 16 bits immediate data into BC (pr=B), DE (pr=D), HL (pr=H), SP (pr=SP)
MVI	M,byte						$[[HL]] \leftarrow \text{byte}$ Load 8 bit immediate data into memory location with implied addressing by HL
MVI	r,byte						$[r] \leftarrow \text{byte}$ Load 8 bit immediate data into any register

Tipos de instruções – grupo aritmético, lógico e de rotação

29

- São instruções que efetuam cálculos aritméticos, lógicos e de rotação sobre os dados
- Estas instruções afetam todas as flags do registo de flags (F)
- A maioria destas instruções apenas têm um operando uma vez que o segundo operando está implícito e é o registo acumulador (A)
- Exemplos:
 - **ADD B** – esta instrução soma o conteúdo do registo acumulador (A) com o conteúdo do registo B e coloca o resultado no registo acumulador (A)
 - **ADD M** – esta instrução ao ser executada soma o conteúdo do registo acumulador (A) com o conteúdo do endereço de memória apontado pelo par de registos HL e coloca o resultado no registo acumulador (A). Atenção que o uso do par de registos HL como apontador de memória obriga a que o respetivo endereço de memória seja carregado no par de registo HL (**LXI H, endereço de memória**)

Tipos de instruções – grupo aritmético, lógico e de rotação

30

Instrução	Operandos	Status do registo de flags					Operação realizada
		CS	AC	Z	S	P	
ADD	M	x	x	x	x	x	$[A] \leftarrow [A] + [[HL]]$ Add register A with implied addressing by HL and store the result in register A
ADC	M	x	x	x	x	x	$[A] \leftarrow [A] + [[HL]] + [CS]$ Add register A with carry with implied addressing by HL and store the result in register A
SUB	M	x	x	x	x	x	$[A] \leftarrow [A] - [[HL]]$ Subtract register A with implied addressing by HL and store the result in register A
SBB	M	x	x	x	x	x	$[A] \leftarrow [A] - [[HL]] - [CS]$ Subtract register A with carry with implied addressing by HL and store the result in register A
ANA	M	0	1	x	x	x	$[A] \leftarrow [A] \text{ AND } [[HL]]$ AND between register A with implied addressing by HL and store the result in register A
XRA	M	0	0	x	x	x	$[A] \leftarrow [A] \text{ XOR } [[HL]]$ Exclusive-OR between register A with implied addressing by HL and store the result in register A
ORA	M	0	0	x	x	x	$[A] \leftarrow [A] \text{ OR } [[HL]]$ OR between register A with implied addressing by HL and store the result in register A
CMP	M	x	x	x	x	x	$[A] - [[HL]]$ Compare register A with implied addressing by HL If register A < $[[HL]]$ then the carry flag is set (1) If register A = $[[HL]]$ then the zero flag is set (1) If register A > $[[HL]]$ then the carry and zero flags are reset (0)
INR	M	x	x	x	x	x	$[[HL]] \leftarrow [[HL]] + 1$ Increment memory
DCR	M	x	x	x	x	x	$[[HL]] \leftarrow [[HL]] - 1$ Decrement memory

Tipos de instruções – grupo aritmético, lógico e de rotação

31

Instrução	Operandos	Status do registro de flags					Operação realizada
		CS	AC	Z	S	P	
ADI	byte	x	x	x	x	x	$[A] \leftarrow [A] + \text{byte}$ Add register A with 8 bit immediate data and store the result in register A
ACI	byte	x	x	x	x	x	$[A] \leftarrow [A] + \text{byte} + [CS]$ Add register A with 8 bit immediate data with carry and store the result in register A
SUI	byte	x	x	x	x	x	$[A] \leftarrow [A] - \text{byte}$ Subtract register A with 8 bit immediate data and store the result in register A
SBI	byte	x	x	x	x	x	$[A] \leftarrow [A] - \text{byte} - [CS]$ Subtract register A with 8 bit immediate data with carry and store the result in register A
ANI	byte	0	1	x	x	x	$[A] \leftarrow [A] \text{ AND byte}$ AND between register A with 8 bit immediate data and store the result in register A
XRI	byte	0	0	x	x	x	$[A] \leftarrow [A] \text{ XOR byte}$ Exclusive-OR between register A with 8 bit immediate data and store the result in register A
ORI	byte	0	0	x	x	x	$[A] \leftarrow [A] \text{ OR byte}$ OR between register A with 8 bit immediate data and store the result in register A
CPI	byte	x	x	x	x	x	$[A] - \text{byte}$ Compare register A with 8 bit immediate data If register A < byte than the carry flag is set (1) If register A = byte than the zero flag is set (1) If register A > byte than the carry and zero flags are reset (0)
ADD	r	x	x	x	x	x	$[A] \leftarrow [A] + [r]$ Add register A with any register and store the result in register A
ADC	r	x	x	x	x	x	$[A] \leftarrow [A] + [r] + [CS]$ Add register A with any register with carry and store the result in register A

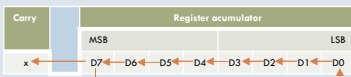
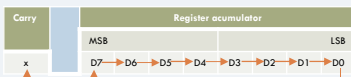
Tipos de instruções – grupo aritmético, lógico e de rotação

32

Instrução	Operandos	Status do registro de flags					Operação realizada
		CS	AC	Z	S	P	
SUB	r	x	x	x	x	x	$[A] \leftarrow [A] - [r]$ Subtract register A with any register and store the result in register A
SBB	r	x	x	x	x	x	$[A] \leftarrow [A] - [r] - [CS]$ Subtract register A with any register with carry and store the result in register A
ANA	r	0	1	x	x	x	$[A] \leftarrow [A] \text{ AND } [r]$ AND between register A with any register and store the result in register A
XRA	r	0	0	x	x	x	$[A] \leftarrow [A] \text{ XOR } [r]$ Exclusive-OR between register A with any register and store the result in register A
ORA	r	0	0	x	x	x	$[A] \leftarrow [A] \text{ OR } [r]$ OR between register A with any register and store the result in register A
CMP	r	x	x	x	x	x	$[A] - [r]$ Compare register A with any register If register A < r than the carry flag is set (1) If register A = r than the zero flag is set (1) If register A > r than the carry and zero flags are reset (0)
INR	r		x	x	x	x	$[r] \leftarrow [r] + 1$ Increment any register
DCR	r		x	x	x	x	$[r] \leftarrow [r] - 1$ Decrement any register
CMA							$[A] \leftarrow [\bar{A}]$ Complement register A

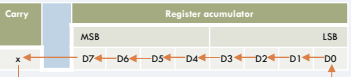
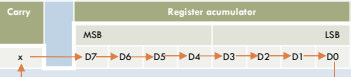
Tipos de instruções – grupo aritmético, lógico e de rotação

33

Instrução	Operandos	Status do registro de flags					Operação realizada
		CS	AC	Z	S	P	
DAA		x	x	x	x	x	<p>The contents of the accumulator are changed from a binary value to two 4-bit binary coded decimal (BCD) digits. This is the only instruction that uses the auxiliary flag (AC) to perform the binary to BCD conversion, and the conversion procedure is described below.</p> <p>If the value of the low-order 4-bits in the accumulator is greater than 9 or if AC flag is set (1), the instruction adds 6 to the low-order four bits.</p> <p>If the value of the high-order 4-bits in the accumulator is greater than 9 or if the Carry flag (CS) is set (1), the instruction adds 6 to the high-order four bits.</p>
RLC		x					<p>Each binary bit of the register accumulator is rotated left by one position. Bit D7 is placed in the position of D0 as well as in the Carry flag. The carry flag (CS) is modified according to bit D7.</p> 
RRC		x					<p>Each binary bit of the register accumulator is rotated right by one position. Bit D0 is placed in the position of D7 as well as in the Carry flag. The carry flag (CS) is modified according to bit D0.</p> 

Tipos de instruções – grupo aritmético, lógico e de rotação

34

Instrução	Operandos	Status do registro de flags					Operação realizada
		CS	AC	Z	S	P	
RAL		x					<p>Each binary bit of the register accumulator is rotated left by one position through the carry flag. Bit D7 is placed in the carry flag, and the carry flag is placed in the least significant position D0. The carry flag (CS) is modified according to bit D7.</p> 
RAR		x					<p>Each binary bit of the register accumulator is rotated right by one position through the carry flag. Bit D0 is placed in the carry flag, and the carry flag is placed in the least significant position D7. The carry flag (CS) is modified according to bit D0.</p> 
DAD	pr	x					<p>$[HL] \leftarrow [HL] + [pr]$ Add HL to a register pair BC (pr=B), DE (pr=D), HL (pr=H), SP (pr=SP) and store the result in HL.</p>

Tipos de instruções – grupo aritmético, lógico e de rotação

35

Instrução	Operandos	Status do registo de flags					Operação realizada
		CS	AC	Z	S	P	
INX	pr						$[pr] \leftarrow [pr] + 1$ Increment register pair BC (pr=B), DE (pr=D), HL (pr=H), SP (pr=SP) and store the result in a register pair BC (pr=B), DE (pr=D), HL (pr=H), SP (pr=SP)
DCX	pr						$[pr] \leftarrow [pr] - 1$ Decrement register pair BC (pr=B), DE (pr=D), HL (pr=H), SP (pr=SP) and store the result in a register pair BC (pr=B), DE (pr=D), HL (pr=H), SP (pr=SP)

Tipos de instruções – grupo de controlo e de salto

36

- São instruções que permitem efetuar saltos para a frente ou para trás num programa
- Os saltos podem ser:
 - ▣ Incondicionais
 - ▣ Condicionais: estes realizam-se testando por exemplo os bits do registo de flags (F)
- Estas instruções permitem realizar saltos para subprogramas
- Exemplos:
 - ▣ **JMP 1000H** – esta instrução redireciona incondicionalmente o contador do programa (PC) para a posição de memória 1000H
 - ▣ **JNZ 1040H** – esta instrução apenas redireciona o contador do programa (PC) para a posição de memória 1040H se o bit de zero (z) do registo de Flags (F) for igual a zero
 - ▣ **CALL 1020H** – esta instrução redireciona incondicionalmente o contador do program (PC) para o endereço de início de um subprograma que se encontra na posição de memória 1020H. Sempre que usamos a instrução CALL para voltar ao programa inicial temos de usar a instrução de return (**RET**), para que o contador de programa (PC) volte à posição original que tinha antes de ser executada a chamada do subprograma. Isto é conseguido uma vez que a instrução CALL guarda o endereço de retorno do programa principal na pilha (STACK).

Tipos de instruções – grupo de controlo e salto

37

Instrução	Operandos	Status do registo de flags						Operação realizada
		CS	AC	Z	S	P		
JMP	label						[PC] ← label Jump to instruction at address label	
PCHL							[PC] ← [HL] Jump to instruction at address contained in HL	
CALL	label						[[SP]] ← [PC] , [PC] ← label, [SP] ← [SP] – 2 Jump to subroutine starting at address label	
CC	label						[[SP]] ← [PC] , [PC] ← label, [SP] ← [SP] – 2 Jump to subroutine starting at address label if the carry flag (CS) equal to 1	
CNC	label						[[SP]] ← [PC] , [PC] ← label, [SP] ← [SP] – 2 Jump to subroutine starting at address label if the carry flag (CS) equal to 0	
CZ	label						[[SP]] ← [PC] , [PC] ← label, [SP] ← [SP] – 2 Jump to subroutine starting at address label if the zero flag (Z) equal to 1	
CNZ	label						[[SP]] ← [PC] , [PC] ← label, [SP] ← [SP] – 2 Jump to subroutine starting at address label if the zero flag (Z) equal to 0	
CP	label						[[SP]] ← [PC] , [PC] ← label, [SP] ← [SP] – 2 Jump to subroutine starting at address label if the sign flag (S) equal to 0	
CM	label						[[SP]] ← [PC] , [PC] ← label, [SP] ← [SP] – 2 Jump to subroutine starting at address label if the sign flag (S) equal to 1	
CPE	label						[[SP]] ← [PC] , [PC] ← label, [SP] ← [SP] – 2 Jump to subroutine starting at address label if the parity flag (P) equal to 1	
CPO	label						[[SP]] ← [PC] , [PC] ← label, [SP] ← [SP] – 2 Jump to subroutine starting at address label if the parity flag (P) equal to 0	

Tipos de instruções – grupo de controlo e salto

38

Instrução	Operandos	Status do registo de flags						Operação realizada
		CS	AC	Z	S	P		
RET							$[PC] \leftarrow [[SP]]$, $[SP] \leftarrow [SP] + 2$ Return from subroutine	
RC							$[PC] \leftarrow [[SP]]$, $[SP] \leftarrow [SP] + 2$ Return from subroutine if the carry flag (CS) equal to 1	
RNC							$[PC] \leftarrow [[SP]]$, $[SP] \leftarrow [SP] + 2$ Return from subroutine if the carry flag (CS) equal to 0	
RZ							$[PC] \leftarrow [[SP]]$, $[SP] \leftarrow [SP] + 2$ Return from subroutine if the zero flag (Z) equal to 1	
RNZ							$[PC] \leftarrow [[SP]]$, $[SP] \leftarrow [SP] + 2$ Return from subroutine if the zero flag (Z) equal to 0	
RM							$[PC] \leftarrow [[SP]]$, $[SP] \leftarrow [SP] + 2$ Return from subroutine if the sign flag (S) equal to 0	
RP							$[PC] \leftarrow [[SP]]$, $[SP] \leftarrow [SP] + 2$ Return from subroutine if the sign flag (S) equal to 1	
RPE							$[PC] \leftarrow [[SP]]$, $[SP] \leftarrow [SP] + 2$ Return from subroutine if the parity flag (P) equal to 1	
RPO							$[PC] \leftarrow [[SP]]$, $[SP] \leftarrow [SP] + 2$ Return from subroutine if the parity flag (P) equal to 0	
JC	label						$[PC] \leftarrow \text{label}$ Jump to instruction at address label if the carry flag (CS) equal to 1	
JNC	label						$[PC] \leftarrow \text{label}$ Jump to instruction at address label if the carry flag (CS) equal to 0	

Tipos de instruções – grupo de controlo e salto

39

Instrução	Operandos	Status do registo de flags					Operação realizada
		CS	AC	Z	S	P	
JZ	label						[PC] ← label Jump to instruction at address label if the zero flag (Z) equal to 1
JNZ	label						[PC] ← label Jump to instruction at address label if the zero flag (Z) equal to 0
JP	label						[PC] ← label Jump to instruction at address label if the sign flag (S) equal to 0
JM	label						[PC] ← label Jump to instruction at address label if the sign flag (S) equal to 1
JPE	label						[PC] ← label Jump to instruction at address label if the parity flag (P) equal to 1
JPO	label						[PC] ← label Jump to instruction at address label if the parity flag (P) equal to 0
RST	n						The RST instruction is equivalent to a 1-byte call instruction to one of eight memory locations depending upon the number. The instructions are generally used in conjunction with interrupts and inserted using external hardware. However these can be used as software instructions in a program to transfer program execution to one of the eight locations. The addresses are:
							Instruction Restart Address
							RST 0 0000H
							RST 1 0008H
							RST 2 0010H
							RST 3 0018H
							RST 4 0020H
							RST 5 0028H
							RST 6 0030H
					RST 7 0038H		

Tipos de instruções – grupo de controlo do CPU, I/O e da Pilha

40

- São instruções que permitem realizar operações diretamente com o CPU, com dispositivos I/O e com a Pilha
- Exemplos:
 - **PUSH B** – Guarda na pilha os valores do par de registos BC
 - **POP D** – copia os valores guardados na pilha para o par de registos DE
 - **OUT byte** – copia o conteúdo do registo acumulador (A) através de um porto de saída cujo endereço é *byte*
 - **IN byte** – copia um byte através de um porto de entrada para o registo acumulador (A), do endereço *byte*
 - **NOP** – é uma instrução usada para temporização, uma vez que durante a sua execução o processador não realiza operações mas o contador de programa (PC) é incrementado
 - **HLT** – é uma instrução que provoca a paragem do processador

Tipos de instruções – grupo de controlo do CPU, I/O e da Pilha

41

Instrução	Operandos	Status do registo de flags					Operação realizada
		CS	AC	Z	S	P	
IN	port						$[A] \leftarrow [\text{port}]$ Input to register accumulator (A) from I/O port
OUT	port						$[\text{port}] \leftarrow [A]$ Output from register accumulator (A) to I/O port
PUSH	pr						$[[SP]] \leftarrow [pr], [SP] \leftarrow [SP] - 2$ Push register pair BC (pr=B), DE (pr=D), H (pr=HL), PSW (pr=PSW) contents onto stack
POP	pr						$[pr] \leftarrow [[SP]], [SP] \leftarrow [SP] + 2$ Pop stack into register pair BC (pr=B), DE (pr=D), H (pr=HL), PSW (pr=PSW)
XTHL							$[HL] \leftrightarrow [[SP]]$ Exchange HL with top of stack
EI							Enable interrupts following execution of next instruction
DI							Disable interrupts
SIM							Set interrupt mask
RIM							Read interrupt mask
NOP							$[PC] \leftarrow [PC] + 1$ No operation but program counter (PC) is incremented
HLT							HALT Stop CPU operation