DEEC
DEPARTAMENTO DE ENGENHARIA ELETROTÉCNICA E DE COMPUTADORES
TÉCNICO LISBOA

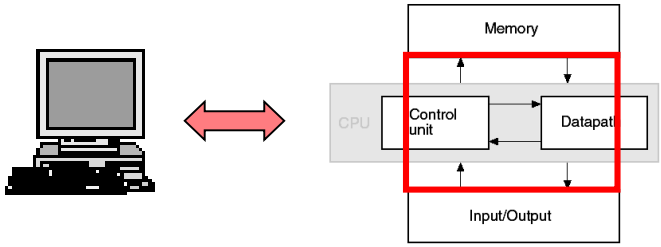
Arquitectura de Computadores

MEEC (2016/17 – 2º Sem.)

Arquitectura do Conjunto de Instruções

Prof. Nuno Horta

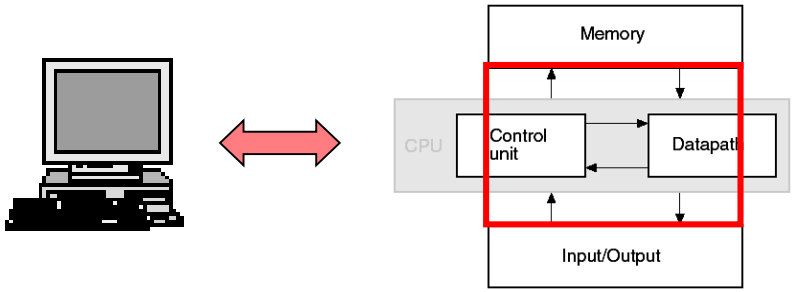
PLANEAMENTO



- *Introdução*
- *Unidade de Processamento*
- *Unidade de Controlo*
- *Arquitectura do Conjunto de Instruções*
- *Unidade Central de Processamento (CPU)*
- *Unidade de Entrada/Saída (I/O)*
- *Unidade de Memória*
- *Perspectiva Evolutiva das Arquitecturas de Computadores*

SUMÁRIO

- *Arquitectura do Conjunto de Instruções*
  - *Introdução*
  - *Endereçamento de Operandos*
  - *Modos de Endereçamento*
  - *Arquitecturas do Conjunto de Instruções*
  - *Instruções do Processador P3*
    - *Instruções de Transferência de Dados*
    - *Instruções de Manipulação de Dados*
    - *Instruções de Controlo de Programa*
    - *Interrupções*



ARQ. ConJ. Instruções

Conceitos/Definições das Arquitecturas de Computadores

Linguagem Assembly: *Linguagem simbólica, utilizada para efeito de programação, que utiliza nomes em vez dos códigos de operação, endereços e operandos binários.*

Linguagem Máquina: *Linguagem binária utilizada na definição e armazenamento de instruções em memória.*

TABELA 10.1: Instruções em linguagem máquina do processador P3.

Endereço		Valor	
Base 2	Base 16	Base 2	Base 16
0001000000000000	1000	1010111001110000	AE70
0001000000000001	1001	0000000010100000	0040
0001000000000010	1002	1000011001110000	8670
0001000000000011	1003	0000000010110000	00B0
0001000000000100	1004	1000011001110000	8670
0001000000000101	1005	0000000010110001	00B1
0001000000000110	1006	0100000000000001	4001
0001000000000111	1007	1010110001110000	AC70
0001000000001000	1008	1111000000000000	F000

TABELA 10.2: Correspondência entre as instruções *assembly* e máquina.

Endereço	Código *assembly*	Código máquina
1000h	MOV R1, M[0040h]	AE70
1001h		0040
1002h	ADD R1, M[00B0h]	8670
1003h		00B0
1004h	ADD R1, M[00B1h]	8670
1005h		00B1
1006h	NEG R1	4070
1007h	MOV M[F000h], R1	AC70
1008h		F000

ARQ. ConJ. Instruções

Conceitos/Definições das Arquitecturas de Computadores

Formato das Instruções: *O conjunto de bits que compõem a instrução organizam-se em grupos ou campos designados por (podem surgir outras tipo de campos não especificados nesta fase):*

opcode *- código da operação a ser executada;*

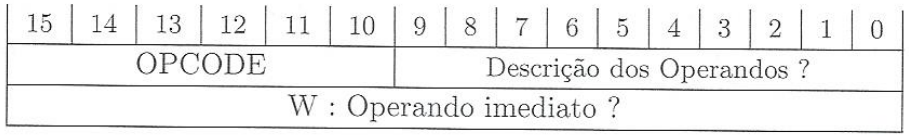
address *- endereço para seleção de uma posição de memória ou registo do processador;*

mode *- modo como o campo de endereço deve ser interpretado.*

Exemplo: Formato da Instrução para o Processador P3

Codificação com uma ou duas palavras de memória.

Segunda palavra utilizada para especificar valor no modo de endereçamento imediato ou no caso de se pretender especificar um endereço de posição de memória.



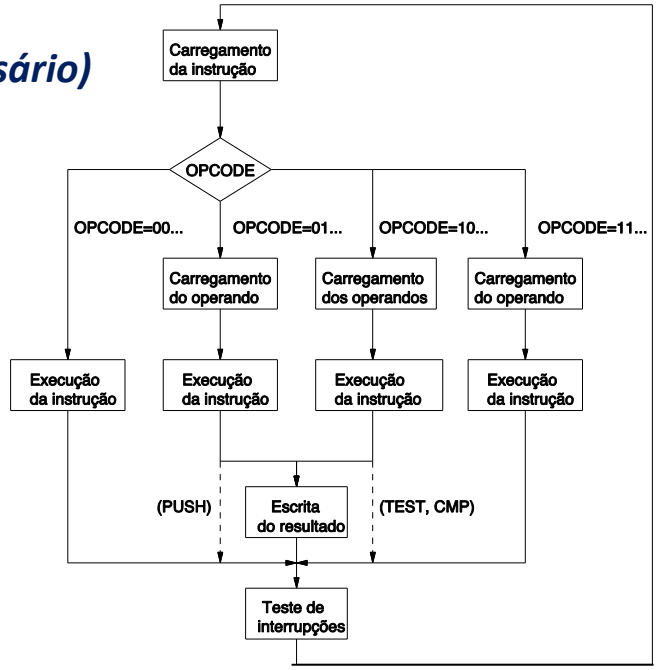
ARQ. ConJ. Instruções

Conceitos/Definições das Arquitecturas de Computadores

Ciclo Básico de Operação de um Computador: *A U. de Controlo é projetada para executar cada instrução de um programa seguindo os seguintes passos:*

1. ***Aquisição de instrução da memória para um registo de controlo.***
2. ***Descodificação de instrução***
3. ***Localização dos operandos utilizados pela instrução.***
4. ***Aquisição de operando da memória (caso seja necessário)***
5. ***Execução da operação***
6. ***Armazenamento do resultado e regresso ao passo 1.***

Exemplo: Execução de Instrução no Processador P3



ARQ. ConJ. Instruções

Conceitos/Definições das Arquitecturas de Computadores

Conjunto de Registos: *Registos da CPU acessíveis ao programador, normalmente descritos no manual de programação Assembly, neste caso:*

- ***Registos de uso geral*** *(R0 a R7)*
- ***Contador do Programa*** *(PC – Program Counter)*
- ***Registo de Estado*** *(PSR – Processor Status Register)*
- ***Apontador para a Pilha*** *(SP – Stack Pointer)*

Exemplo: Registos da UA no Processador P3

TABELA 12.1: Banco de registos.

Registo	Descrição
R0	Constante 0
R1	Registo de uso geral
R2	Registo de uso geral
R3	Registo de uso geral
R4	Registo de uso geral
R5	Registo de uso geral
R6	Registo de uso geral
R7	Registo de uso geral
R8	Registo de uso restrito
R9	Registo de uso restrito
R10	Registo de uso restrito
R11	Operando (SD)
R12	Endereço de destino (EA)
R13	Resultado (RD)
R14	Apontador da pilha (SP)
R15	Contador de programa (PC)

ARQ. ConJ. Instruções

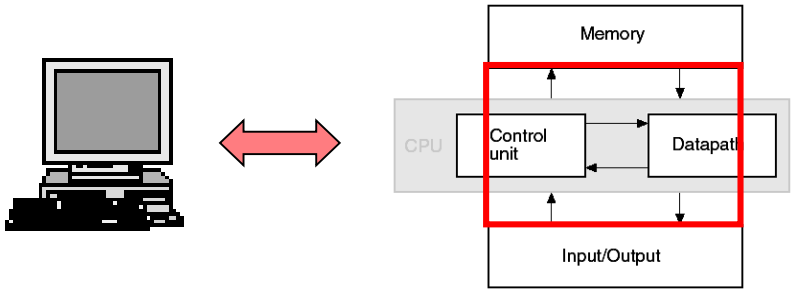
Conceitos/Definições das Arquitecturas de Computadores

Endereçamento de Operandos: *O* ***endereçamento explícito*** *de operandos pode ser através da especificação do endereço de memória ou do endereço do registo do processador, contudo, o* ***endereçamento*** *de um operando pode também ser feito* ***de forma implícita*** *através do código da operação.*

Definição da Arquitectura do Conjunto de Instruções: *O* ***número de operandos de endereçamento explícito****, numa instrução de manipulação de dados, e destes o* ***número de operandos*** *que podem ser* ***diretamente endereçados em memória*** *são fatores fundamentais na definição de uma Arquitectura para o Conjunto de Instruções* ***e****, naturalmente, na* ***dimensão das instruções****.*

SUMÁRIO

- *Arquitectura do Conjunto de Instruções*
  - *Introdução*
  - *Endereçamento de Operandos*
  - *Modos de Endereçamento*
  - *Arquitecturas do Conjunto de Instruções*
  - *Instruções do Processador P3*
    - *Instruções de Transferência de Dados*
    - *Instruções de Manipulação de Dados*
    - *Instruções de Controlo de Programa*
    - *Interrupções*



ARQ. ConJ. Instruções

Endereçamento de Operandos

*A **influência do número de operandos no desenvolvimento de programas** será ilustrada através do cálculo da seguinte expressão aritmética:*

$$X=(A+B)(C+D)$$

***Instruções com 3 Operandos,** de Endereçamento Explícito:*

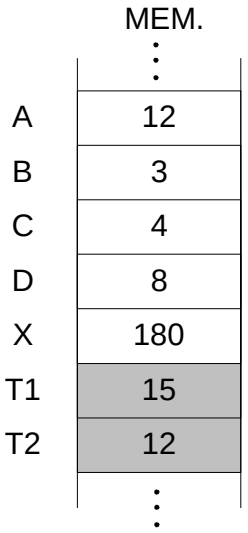
*(a) Armazenamento temporário em **memória***

(A,B,C,D,X,T1,T2 são Endereços de memória)

```
ADD T1,A,B      M[T1]←M[A]+M[B]
ADD T2,C,D      M[T2]←M[C]+M[D]
MUL X,T1,T2     M[X]←M[T1]xM[T2]
```

*(b) Armazenamento temporário em **registo***

```
ADD R1,A,B      R1←M[A]+M[B]
ADD R2,C,D      R2←M[C]+M[D]
MUL X,R1,R2     M[X]←R1xR2
```



ARQ. ConJ. Instruções

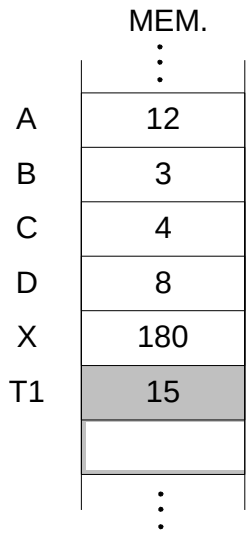
***Endereçamento de Operandos** (cont.)*

$$X=(A+B)(C+D)$$

Instruções com 2 Operandos:

Cada campo de endereço permite especificar um registo ou um endereço de memória. O registo R1 pode substituir T1 (endereço de memória) no armazenamento temporário.

```
MOVE T1,A        M[T1]←M[A]
ADD T1,B         M[T1]←M[T1]+M[B]
MOVE X,C         M[X]←M[C]
ADD X,D          M[X]←M[X]+M[D]
MUL X,T1         M[X]←M[X]x M[T1]
```



ARQ. ConJ. Instruções

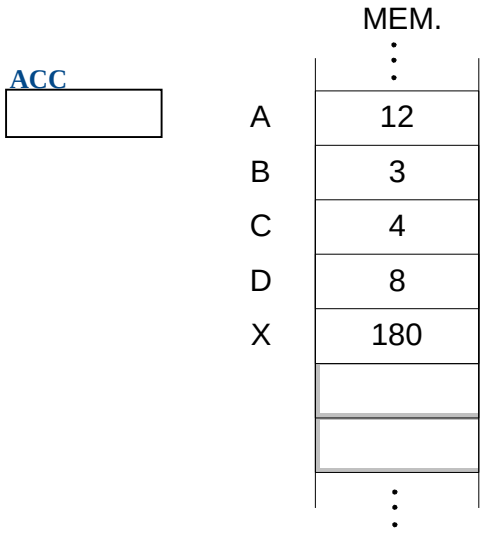
Endereçamento de Operandos *(cont.)*

$$X=(A+B)(C+D)$$

Instruções com 1 Operando:

*No caso das arquitecturas com instruções de apenas um operando (explícito) é utilizado um registo - **acumulador** - de forma implícita para obter um dos operandos e para localizar o resultado da operação.*

```
LD  A     ACC←M[A]
ADD B     ACC←ACC+M[B]
ST  X     M[X]←ACC
LD  C     ACC←M[C]
ADD D     ACC←ACC+M[D]
MUL X     ACC←ACC x M[X]
ST  X     M[X]←ACC
```



ARQ. ConJ. Instruções

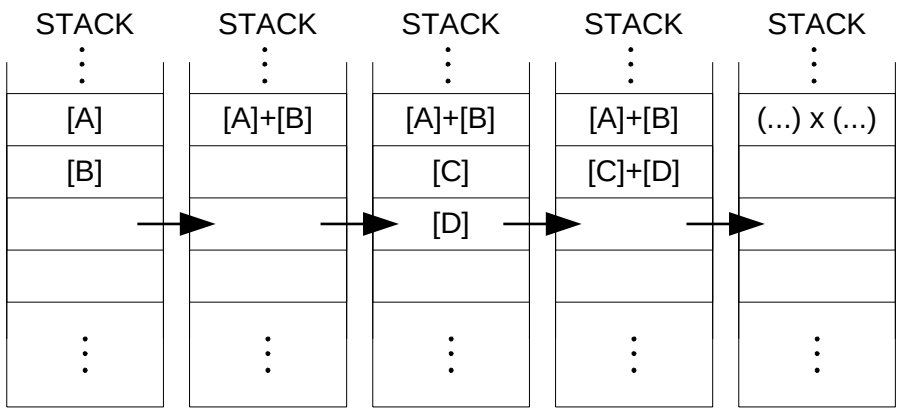
***Endereçamento de Operandos** (cont.)*

$$X=(A+B)(C+D)$$

Instruções sem Operandos:

*No caso das arquitecturas com instruções sem operandos (explícitos), todos os **operandos** terão de surgir **de forma implícita**. A forma convencional de solucionar este problema é recorrendo a um **STACK** (pilha), estrutura de memória do tipo **LIFO** para armazenamento de dados dinâmicos. (TOS – Top of Stack)*

```
PUSH A   TOS←M[A]
PUSH B   TOS←M[B]
ADD      TOS←TOS+TOS-1
PUSH C   TOS←M[C]
PUSH D   TOS←M[D]
ADD      TOS←TOS+TOS-1
MUL      TOS←TOSxTOS-1
POP X    M[X]←TOS
```



ARQ. ConJ. Instruções

Arquitecturas de Endereçamento

Classificação:

1. ***Memory-Memory:*** *O endereçamento dos operandos é na totalidade realizado sobre a memória.*
2. ***Register-Register ou Load/Store:*** *O endereçamento dos operandos faz-se com recusro a registos. O acesso à memória é restrito às instruções do tipo LD (load) e ST (store).*

Nota: *No caso de ser necessária uma palavra de memória adicional para especificar o endereço de cada operando, o número de acessos à memória no primeiro caso é de **21**, enquanto no segundo é de **18**.Justifique!*

```
MEMORY-MEMORY
ADD T1,A,B        M[T1]←M[A]+M[B]
ADD T2,C,D        M[T2]←M[C]+M[D]
MUL X,T1,T2       M[X]←M[T1]xM[T2]
```

*A solução **Memory-Memory** conduz a uma **maior complexidade das estruturas de controlo**, enquanto na **Register-Register** é necessário um **maior número de registos**.*

```
REGISTER-REGISTER
LD R1,A           R1←M[A]
LD R2,B           R2←M[B]
ADD R3,R1,R2      R3←R1+R2
LD R1,C           R1←M[C]
LD R2,D           R2←M[D]
ADD R1,R1,R2      R1←R1+R2
MUL R1,R1,R3      R1←R1xR3
ST X,R1           M[X]←R1
```

ARQ. ConJ. Instruções

Arquitecturas de Endereçamento *(cont.)*

Classificação:

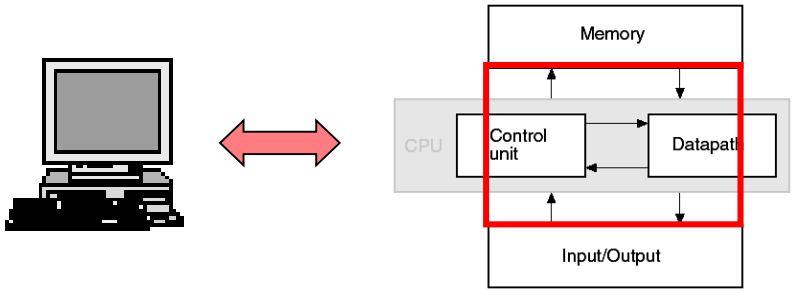
3. ***Register-Memory:*** *O endereçamento dos operandos faz-se com recurso a registos e memória.*

4. ***Single Accumulator:*** *O endereçamento não recorre a registos de uso geral, sendo as transferências realizadas na totalidade entre a memória e o registo Acumulador.*

5. ***Stack:*** *O endereçamento é realizado de forma implícita com recurso ao STACK.*

SUMÁRIO

- *Arquitectura do Conjunto de Instruções*
  - *Introdução*
  - *Endereçamento de Operandos*
  - *Modos de Endereçamento*
  - *Arquitecturas do Conjunto de Instruções*
  - *Instruções do Processador P3*
    - *Instruções de Transferência de Dados*
    - *Instruções de Manipulação de Dados*
    - *Instruções de Controlo de Programa*
    - *Interrupções*



ARQ. ConJ. Instruções

Modos de Endereçamento

*O **modo de endereçamento** descreve o **modo como os operandos são seleccionados** durante a execução do programa.*

*O **endereço do operando** obtido pelas regras subjacentes aos modos de endereçamento designa-se **endereço efectivo**.*

*Os vários **modos de endereçamento** tem por **objectivos fundamentais**:*

1. *Conferir **Flexibilidade de Programação** através da utilização de **apontadores para memória**, **contadores para controlo de ciclos**, **indexação dos dados** e **relocalização de programas**.*
2. ***Reduzir** o número de bits nos campos de endereço da **instrução**.*

*A **especificação do modo de endereçamento** pode ser através de um campo próprio ou fazer parte do código da operação.*

ARQ. ConJ. Instruções

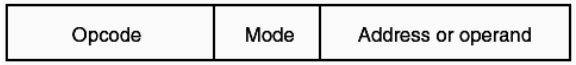
Modos de Endereçamento

TABELA 10.3: Principais modos de endereçamento utilizados.

Modo de endereçamento	Operação
Por registo	op ← RX
Indirecto por registo	op ← M[RX]
Imediato	op ← W
Directo	op ← M[W]
Indexado	op ← M[RX+W]
Relativo	op ← M[PC+W]
Baseado	op ← M[SP+W]
Indirecto	op ← M[M[W]]
Duplamente indirecto por registo	op ← M[M[RX]]
Implícito	

ARQ. ConJ. Instruções

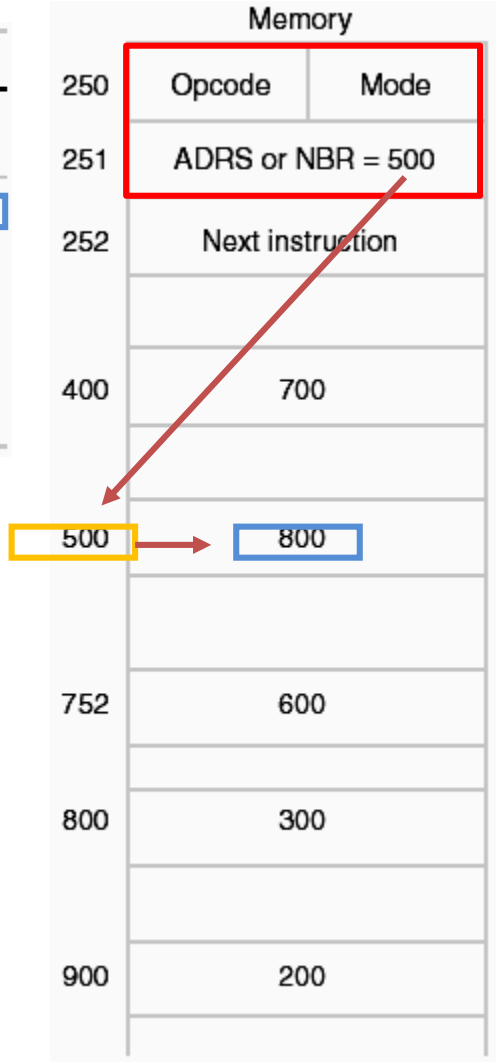
Modos de Endereçamento - Sumário



Addressing mode	Symbolic convention	Register transfer	Refers to Figure 9-6: Effective address	Refers to Figure 9-6: Contents of *ACC*
Direct	LDA ADRS	$ACC \leftarrow M[ADRS]$	500	800
Immediate	LDA #NBR	$ACC \leftarrow NBR$	251	500
Indirect	LDA [ADRS]	$ACC \leftarrow M[M[ADRS]]$	800	300
Relative	LDA $ADRS	$ACC \leftarrow M[ADRS+PC]$	752	600
Index	LDA ADRS (R1)	$ACC \leftarrow M[ADRS+R1]$	900	200
Register	LDA R1	$ACC \leftarrow R1$	—	400
Register indirect	LDA (R1)	$ACC \leftarrow M[R1]$	400	700

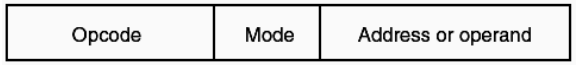
Simbologia (Mano):

ADRS	*Endereço Efectivo*
#NBR	*Número*
@ ou [ADRS]	*Endereço Indirecto*
$ADRS	*Endereço Efectivo relativo ao PC*
ADRS (R1)	*Endereço Efectivo relativo ao R1*
R1	*Registo*
(R1)	*Endereço Efectivo em R1*



ARQ. ConJ. Instruções

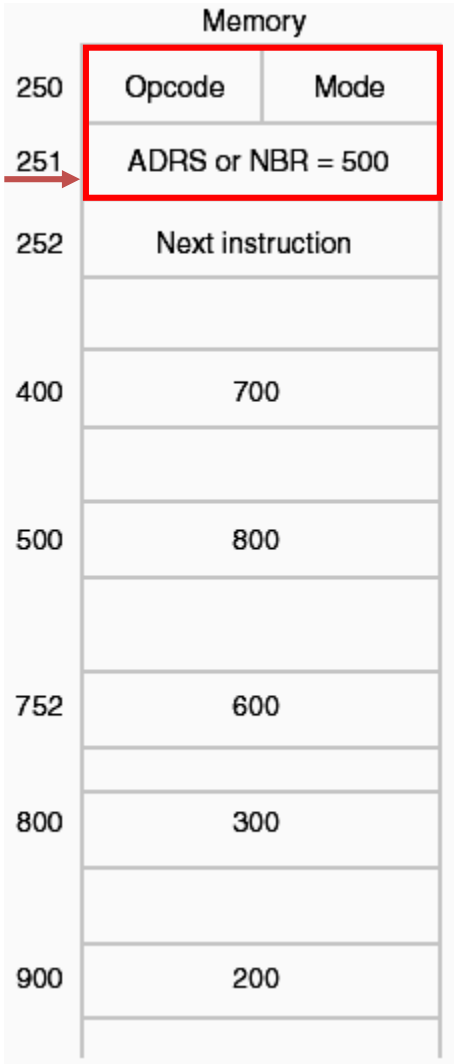
Modos de Endereçamento - Sumário



Addressing mode	Symbolic convention	Register transfer	Refers to Figure 9-6: Effective address	Refers to Figure 9-6: Contents of *ACC*
Direct	LDA ADRS	$ACC \leftarrow M[ADRS]$	500	800
Immediate	LDA #NBR	$ACC \leftarrow NBR$	251	500
Indirect	LDA [ADRS]	$ACC \leftarrow M[M[ADRS]]$	800	300
Relative	LDA $ADRS	$ACC \leftarrow M[ADRS+PC]$	752	600
Index	LDA ADRS (R1)	$ACC \leftarrow M[ADRS+R1]$	900	200
Register	LDA R1	$ACC \leftarrow R1$	—	400
Register indirect	LDA (R1)	$ACC \leftarrow M[R1]$	400	700

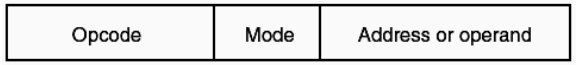
Simbologia (Mano):

ADRS	*Endereço Efectivo*
#NBR	*Número*
@ ou [ADRS]	*Endereço Indirecto*
$ADRS	*Endereço Efectivo relativo ao PC*
ADRS (R1)	*Endereço Efectivo relativo ao R1*
R1	*Registo*
(R1)	*Endereço Efectivo em R1*



ARQ. ConJ. Instruções

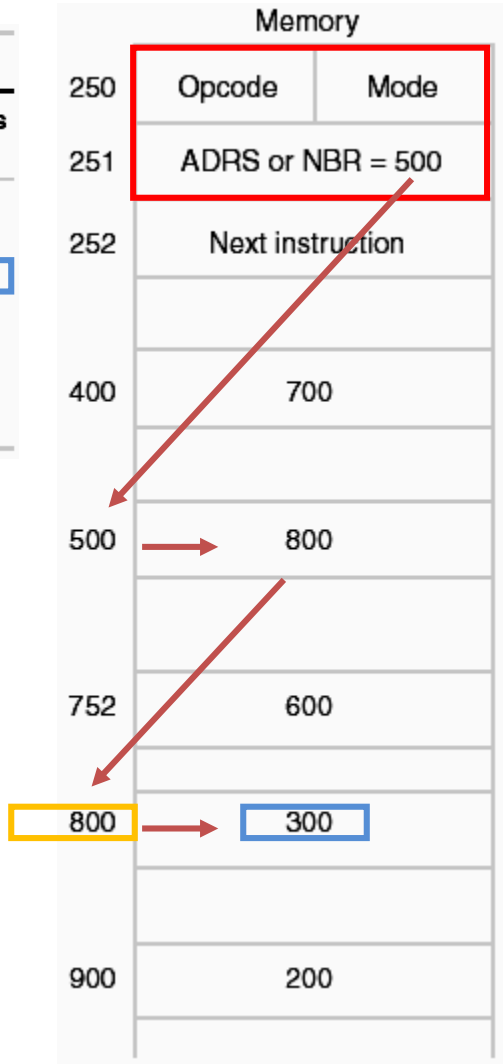
Modos de Endereçamento - Sumário



Addressing mode	Symbolic convention	Register transfer	Refers to Figure 9-6: Effective address	Refers to Figure 9-6: Contents of *ACC*
Direct	LDA ADRS	$ACC \leftarrow M[ADRS]$	500	800
Immediate	LDA #NBR	$ACC \leftarrow NBR$	251	500
Indirect	LDA [ADRS]	$ACC \leftarrow M[M[ADRS]]$	800	300
Relative	LDA $ADRS	$ACC \leftarrow M[ADRS+PC]$	752	600
Index	LDA ADRS (R1)	$ACC \leftarrow M[ADRS+R1]$	900	200
Register	LDA R1	$ACC \leftarrow R1$	—	400
Register indirect	LDA (R1)	$ACC \leftarrow M[R1]$	400	700

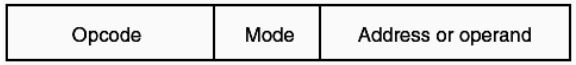
Simbologia (Mano):

ADRS	*Endereço Efectivo*
#NBR	*Número*
@ ou [ADRS]	*Endereço Indirecto*
$ADRS	*Endereço Efectivo relativo ao PC*
ADRS (R1)	*Endereço Efectivo relativo ao R1*
R1	*Registo*
(R1)	*Endereço Efectivo em R1*



ARQ. ConJ. Instruções

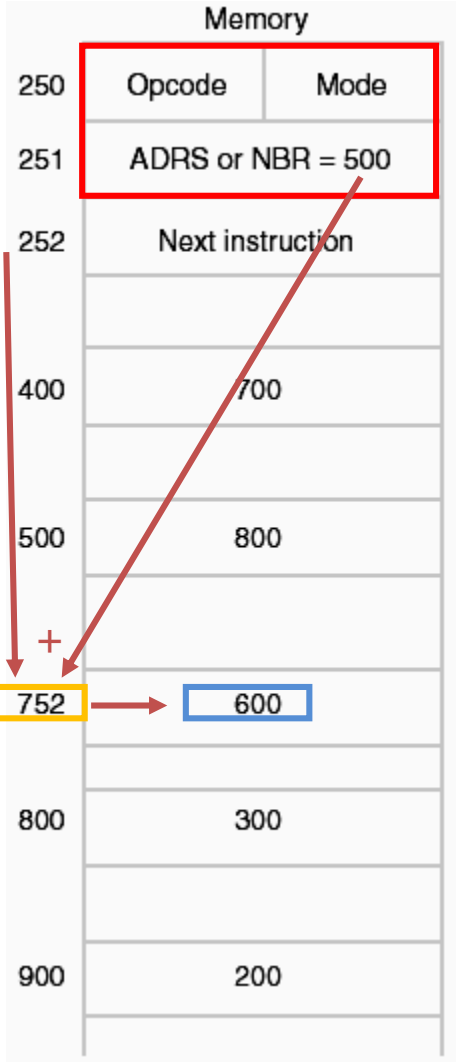
Modos de Endereçamento - Sumário



Addressing mode	Symbolic convention	Register transfer	Refers to Figure 9-6: Effective address	Refers to Figure 9-6: Contents of *ACC*
Direct	LDA ADRS	$ACC \leftarrow M[ADRS]$	500	800
Immediate	LDA #NBR	$ACC \leftarrow NBR$	251	500
Indirect	LDA [ADRS]	$ACC \leftarrow M[M[ADRS]]$	800	300
Relative	LDA $ADRS	$ACC \leftarrow M[ADRS+PC]$	752	600
Index	LDA ADRS (R1)	$ACC \leftarrow M[ADRS+R1]$	900	200
Register	LDA R1	$ACC \leftarrow R1$	—	400
Register indirect	LDA (R1)	$ACC \leftarrow M[R1]$	400	700

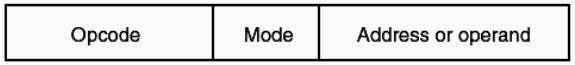
Simbologia (Mano):

ADRS	*Endereço Efectivo*
#NBR	*Número*
@ ou [ADRS]	*Endereço Indirecto*
$ADRS	*Endereço Efectivo relativo ao PC*
ADRS (R1)	*Endereço Efectivo relativo ao R1*
R1	*Registo*
(R1)	*Endereço Efectivo em R1*



ARQ. ConJ. Instruções

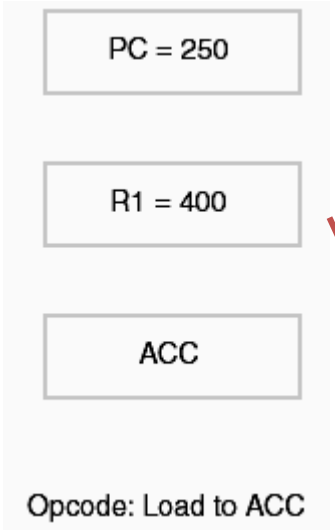
Modos de Endereçamento - Sumário

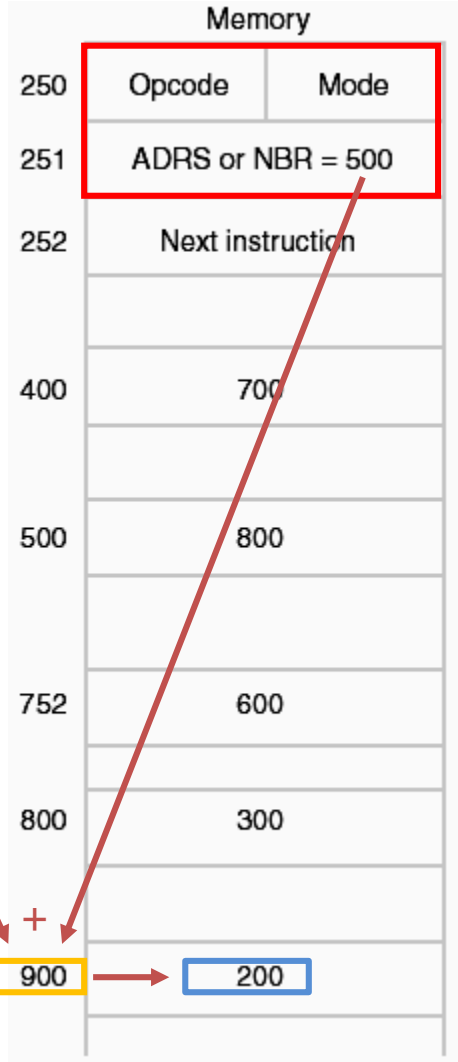


Addressing mode	Symbolic convention	Register transfer	Refers to Figure 9-6: Effective address	Refers to Figure 9-6: Contents of *ACC*
Direct	LDA ADRS	$ACC \leftarrow M[ADRS]$	500	800
Immediate	LDA #NBR	$ACC \leftarrow NBR$	251	500
Indirect	LDA [ADRS]	$ACC \leftarrow M[M[ADRS]]$	800	300
Relative	LDA $ADRS	$ACC \leftarrow M[ADRS+PC]$	752	600
Index	LDA ADRS (R1)	$ACC \leftarrow M[ADRS+R1]$	900	200
Register	LDA R1	$ACC \leftarrow R1$	—	400
Register indirect	LDA (R1)	$ACC \leftarrow M[R1]$	400	700

Simbologia (Mano):

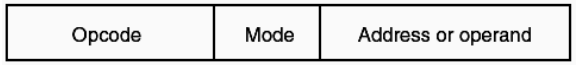
ADRS	*Endereço Efectivo*
#NBR	*Número*
@ ou [ADRS]	*Endereço Indirecto*
$ADRS	*Endereço Efectivo relativo ao PC*
ADRS (R1)	*Endereço Efectivo relativo ao R1*
R1	*Registo*
(R1)	*Endereço Efectivo em R1*





ARQ. ConJ. Instruções

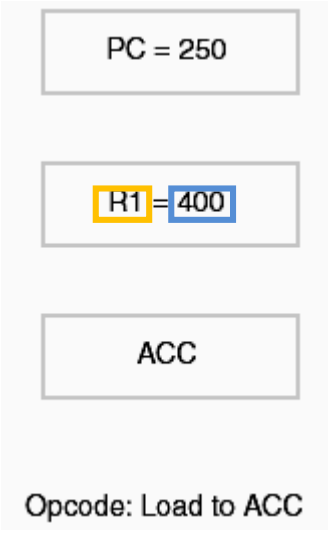
Modos de Endereçamento - Sumário

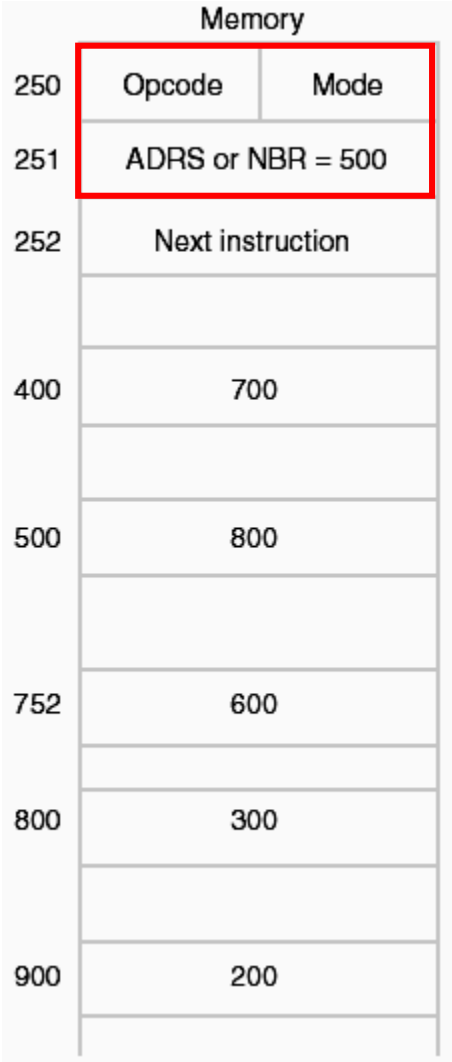


Addressing mode	Symbolic convention	Register transfer	Refers to Figure 9-6: Effective address	Refers to Figure 9-6: Contents of *ACC*
Direct	LDA ADRS	$ACC \leftarrow M[ADRS]$	500	800
Immediate	LDA #NBR	$ACC \leftarrow NBR$	251	500
Indirect	LDA [ADRS]	$ACC \leftarrow M[M[ADRS]]$	800	300
Relative	LDA $ADRS	$ACC \leftarrow M[ADRS+PC]$	752	600
Index	LDA ADRS (R1)	$ACC \leftarrow M[ADRS+R1]$	900	200
Register	LDA R1	$ACC \leftarrow R1$	—	400
Register indirect	LDA (R1)	$ACC \leftarrow M[R1]$	400	700

Simbologia (Mano):

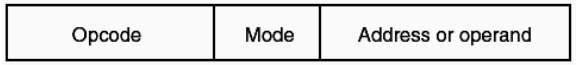
ADRS	*Endereço Efectivo*
#NBR	*Número*
@ ou [ADRS]	*Endereço Indirecto*
$ADRS	*Endereço Efectivo relativo ao PC*
ADRS (R1)	*Endereço Efectivo relativo ao R1*
R1	*Registo*
(R1)	*Endereço Efectivo em R1*





ARQ. ConJ. Instruções

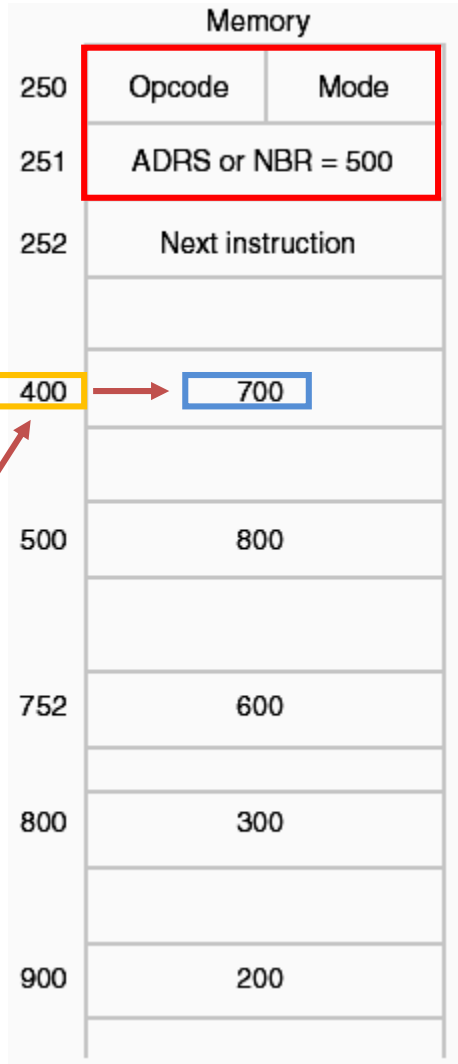
Modos de Endereçamento - Sumário



Addressing mode	Symbolic convention	Register transfer	Refers to Figure 9-6: Effective address	Refers to Figure 9-6: Contents of *ACC*
Direct	LDA ADRS	$ACC \leftarrow M[ADRS]$	500	800
Immediate	LDA #NBR	$ACC \leftarrow NBR$	251	500
Indirect	LDA [ADRS]	$ACC \leftarrow M[M[ADRS]]$	800	300
Relative	LDA $ADRS	$ACC \leftarrow M[ADRS+PC]$	752	600
Index	LDA ADRS (R1)	$ACC \leftarrow M[ADRS+R1]$	900	200
Register	LDA R1	$ACC \leftarrow R1$	—	400
Register indirect	LDA (R1)	$ACC \leftarrow M[R1]$	400	700

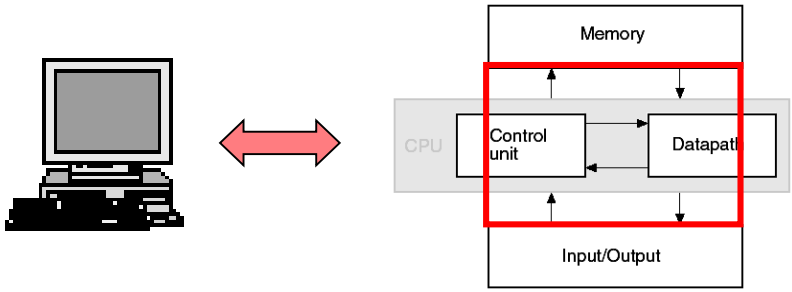
Simbologia (Mano):

ADRS	*Endereço Efectivo*
#NBR	*Número*
@ ou [ADRS]	*Endereço Indirecto*
$ADRS	*Endereço Efectivo relativo ao PC*
ADRS (R1)	*Endereço Efectivo relativo ao R1*
R1	*Registo*
(R1)	*Endereço Efectivo em R1*



SUMÁRIO

- *Arquitectura do Conjunto de Instruções*
  - *Introdução*
  - *Endereçamento de Operandos*
  - *Modos de Endereçamento*
  - *Arquitecturas do Conjunto de Instruções*
  - *Instruções do Processador P3*
    - *Instruções de Transferência de Dados*
    - *Instruções de Manipulação de Dados*
    - *Instruções de Controlo de Programa*
    - *Interrupções*



ARQ. ConJ. Instruções

Arquitecturas do Conjunto de Instruções

O conjunto de instruções de diferentes computadores varia em vários aspectos, e.g., código de operação no campo opcode da instrução, o nome simbólico dado às instruções, etc.

Classes de Arquitecturas de Instruções:

RISC – Reduced Instruction Set Computers

1. ***Acesso à memória** restrito a instruções de carregamento – **load -** e armazenamento – **store.** As instruções de **manipulação de dados** são do tipo **Registo-Registo**.*
2. ***Modos de Endereçamento** em **número limitado**.*
3. ***Formatos das Instruções** todos da **mesma dimensão**.*
4. *As **instruções** correspondem à execução de **operações elementares***

*O objectivo das arquitecturas **RISC** é conferir um **elevado ritmo de execução**, para isso **minimiza os acessos à memória** e em contrapartida **aumenta o número de registos da CPU**. A dimensão fixa das instruções e a simplicidade das suas operações conduz a **unidades de controlo** relativamente **simples**, tipicamente "**Hardwired**" e arquitecturas **pipelined**.*

ARQ. ConJ. Instruções

Arquitecturas do Conjunto de Instruções

Classes de Arquitecturas de Instruções: *(cont.)*

CISC – Complex Instruction Set Computers

1. ***Acesso à memória*** *disponível para a generalidade dos tipos de instrução.*
2. ***Modos de Endereçamento*** *em* ***número elevado****.*
3. ***Formatos das Instruções*** *de* ***diferentes dimensões****.*
4. *As* ***instruções*** *executam tanto* ***operações elementares como operações complexas.***

O objectivo das arquitecturas ***CISC*** *é conferir uma* ***maior proximidade*** *entre as operações utilizadas em* ***linguagens de programação*** *e as operações desencadeadas por cada instrução. Facilita o desenvolvimento de* ***programas compactos*** *e eficiência em termos de desempenho advém de um* ***menor número de acessos à memória em relação ao número de operações elementares realizadas****. Neste caso, a CPU apresenta um* ***menor número de registos*** *do que nas arquitecturas RISC e a utilização do* ***controlo microprogramado*** *é o mais usual dada a variedade de formatos de instruções utilizados.*

Nota*: Cada instrução CISC, em geral, corresponde a uma sequência de instruções RISC.*

ARQ. ConJ. Instruções

Arquitecturas do Conjunto de Instruções

Classes de Arquitecturas de Instruções: *(cont.)*

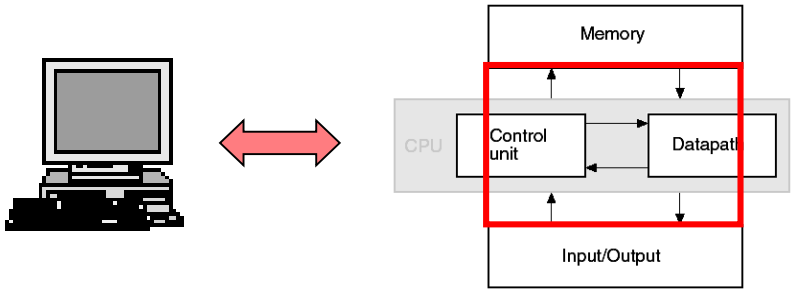
RISC, CISC

Independentemente das arquitecturas serem CISC, RISC ou uma solução híbrida CISC-RISC, existe, tipicamente, um ***conjunto de operações elementares*** *disponíveis na generalidade dos casos e associadas às seguintes classes:*

- ***Instruções de Transferência de Dados:*** *Transferência de dados de uma localização para outra sem alterar a informação.*
- ***Instruções de Manipulação de Dados:*** *Execução de operações aritméticas, lógicas e de deslocamento.*
- ***Instruções de Controlo de Programa:*** *Permitem decidir sobre o fluxo do programa durante a sua execução.*

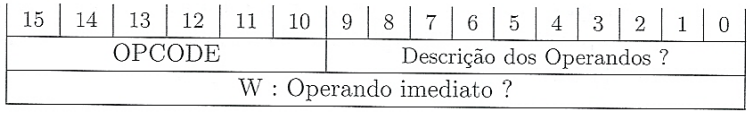
SUMÁRIO

- *Arquitectura do Conjunto de Instruções*
  - *Introdução*
  - *Endereçamento de Operandos*
  - *Modos de Endereçamento*
  - *Arquitecturas do Conjunto de Instruções*
  - *Instruções do Processador P3*
    - *Instruções de Transferência de Dados*
    - *Instruções de Manipulação de Dados*
    - *Instruções de Controlo de Programa*
    - *Interrupções*



ARQ. CONJUNTO DE INSTRUÇÕES

Formato da Instrução



Modos de Endereçamento

TABELA 10.13: Modos de endereçamento do processador P3.

M	Endereçamento	Operação
00	Por registo	op = RX
01	Por registo indirecto	op = M[RX]
10	Imediato	op = W
11	Indexado, directo, relativo ou baseado	op = M[RX+W]

TABELA 10.12: Códigos de operação do processador P3.

Mnemónica	Código	Mnemónica	Código
NOP	000000	CMP	100000
ENI	000001	ADD	100001
DSI	000010	ADDC	100010
STC	000011	SUB	100011
CLC	000100	SUBB	100100
CMC	000101	MUL	100101
RET	000110	DIV	100110
RTI	000111	TEST	100111
INT	001000	AND	101000
RETN	001001	OR	101001
NEG	010000	XOR	101010
INC	010001	MOV	101011
DEC	010010	MVBH	101100
COM	010011	MVBL	101101
PUSH	010100	XCH	101110
POP	010101	JMP	110000
SHR	011000	JMP.cond	110001
SHL	011001	CALL	110010
SHRA	011010	CALL.cond	110011
SHLA	011011	BR	111000
ROR	011100	BR.cond	111001
ROL	011101		
RORC	011110		
ROLC	011111		

ARQ. CONJUNTO DE INSTRUÇÕES

Instruções sem Operandos

NOP, ENI, DSI, STC, CLC; CMC, RET, RTI

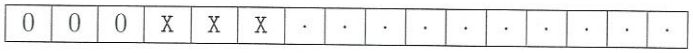


FIGURA 10.8: Codificação de instruções sem operandos.

Instruções sem Operandos e um Parâmetro

INT e RETN

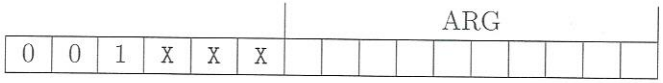


FIGURA 10.9: Codificação de instruções com um parâmetro.

ARQ. CONJUNTO DE INSTRUÇÕES

Instruções com um Operando

NEG, INC, DEC, COM, PUSH, POP

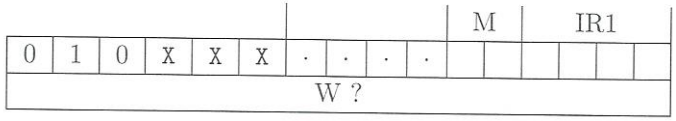


FIGURA 10.10: Codificação de instruções com um operando.

Instruções com um Operando e um Parâmetro

SHR, SHL, SHRA, SHLA, ROR, ROL, RORC, ROLC

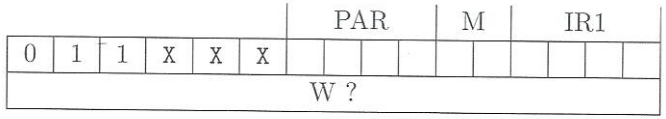


FIGURA 10.11: Codificação de instruções com um operando e um parâmetro.

ARQ. CONJUNTO DE INSTRUÇÕES

Instruções com dois Operandos

CMP, ADD, ADDC, SUB, MUL, DIV, …

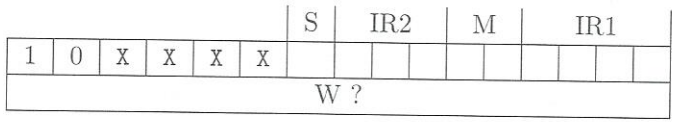


FIGURA 10.12: Codificação de instruções com dois operandos.

Instruções de Controlo

Instruções de salto absoluto – JMP, JMP.cond, CALL e CALL.cond

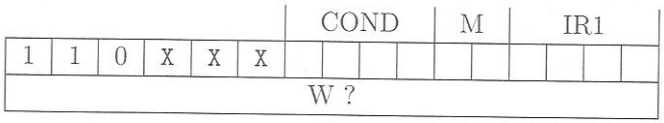


FIGURA 10.13: Codificação das instruções de salto absoluto

Instruções de salto relativo – BR e BR.cond

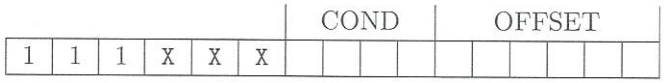


FIGURA 10.14: Codificação das instruções de salto relativo.

TABELA 10.14: Codificação das condições de teste.

Condição	Mnemónica	Código
Zero	Z	0000
Não-zero	NZ	0001
Transporte	C	0010
Não-transporte	NC	0011
Negativo	N	0100
Não-negativo	NN	0101
Excesso	O	0110
Não-excesso	NO	0111
Positivo	P	1000
Não-positivo	NP	1001
Interrupção	I	1010
Não-interrupção	NI	1011

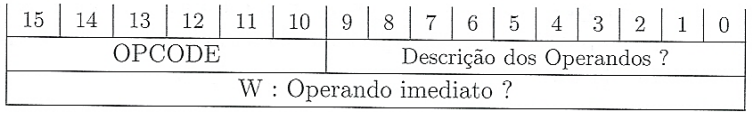
ARQ. CONJUNTO DE INSTRUÇÕES

Conjunto de Instruções do Processador P3

TABELA 10.4: Conjunto de instruções do processador P3.

Aritméticas	Lógicas	Deslocamento	Controlo	Transferência	Genéricas
NEG	COM	SHR	BR	MOV	NOP
INC	AND	SHL	BR.cond	MVBH	ENI
DEC	OR	SHRA	JMP	MVBL	DSI
ADD	XOR	SHLA	JMP.cond	XCH	STC
ADDC	TEST	ROR	CALL	PUSH	CLC
SUB		ROL	CALL.cond	POP	CMC
SUBB		RORC	RET		
CMP		ROLC	RETN		
MUL			RTI		
DIV			INT		

Formato das Instruções



ARQ. CONJUNTO DE INSTRUÇÕES

Instruções Aritméticas

TABELA 10.5: Instruções aritméticas do processador P3.

Instrução	Mnemónica	Exemplo
Complemento aritmético	NEG	NEG R1
Incrementar	INC	INC M[R2]
Decrementar	DEC	DEC M[R3+A5A5h]
Adicionar	ADD	ADD R3, M[R5+4]
Adicionar com transporte	ADDC	ADD R3, M[R6]
Subtrair	SUB	SUB R3, M[R5+4]
Subtrair com empréstimo	SUBB	SUBB R1, R2
Comparar	CMP	CMP R1, R2
Multiplicar	MUL	MUL R3, R4
Dividir	DIV	DIV R3, R4

Instruções Lógicas e de Deslocamento

TABELA 10.6: Instruções lógicas do processador P3.

Instrução	Mnemónica	Exemplo
Conjunção	AND	AND R1, M[R3]
Disjunção	OR	OR R1, 00FFh
Disjunção exclusiva	XOR	XOR M[R1], R2
Complemento lógico	COM	COM M[R2+4]
Teste	TEST	TEST R5, M[R4]

TABELA 10.7: Instruções de deslocamento do processador P3.

Instrução	Mnemónica	Exemplo
Deslocamento lógico à direita	SHR	SHR R1, 4
Deslocamento lógico à esquerda	SHL	SHL M[R1], 2
Deslocamento aritmético à direita	SHRA	SHRA M[R1], 2
Deslocamento aritmético à esquerda	SHLA	SHLA M[R2], 4
Rotação para a direita	ROR	ROR R4, 15
Rotação para a esquerda	ROL	ROL R4, 1
Rotação para a direita, com transporte	RORC	RORC R4, 15
Rotação para a esquerda, com transporte	ROLC	ROLC R2, 15

ARQ. CONJUNTO DE INSTRUÇÕES

Instruções de Controlo

TABELA 10.8: Instruções de controlo do processador P3.

Instrução	Mnemónica	Exemplo
Salto incondicional relativo	`BR`	`BR Pos1`
Salto condicional relativo	`BR.cond`	`BR.cond R3`
Salto incondicional absoluto	`JMP`	`JMP M[R3+1]`
Salto condicional absoluto	`JMP.cond`	`JMP.cond Rot1`
Chamada incondicional a subrotina	`CALL`	`CALL Rotina1`
Chamada condicional a subrotina	`CALL.cond`	`CALL.cond Rot2`
Retorno de subrotina	`RET`	`RET`
Retorno de subrotina com N parâmetros	`RETN`	`RETN 4`
Interrupção	`INT`	`INT 55`
Retorno de interrupção	`RTI`	`RTI`

e Condições de Salto

TABELA 10.9: Condições de salto para o processador P3.

Condição	Mnemónica	Descrição
Zero	`Z`	Última operação deu resultado zero
Não-zero	`NZ`	Última operação deu resultado não zero
Transporte	`C`	Última operação gerou transporte
Não-transporte	`NC`	Última operação não gerou transporte
Negativo	`N`	Última operação deu resultado negativo
Não-negativo	`NN`	Última operação deu resultado não negativo
Excesso	`O`	Última operação gerou excesso (*overflow*)
Não-excesso	`NO`	Última operação não gerou excesso (*overflow*)
Positivo	`P`	Última operação deu resultado positivo
Não-positivo	`NP`	Última operação não deu resultado positivo
Interrupção	`I`	Existe uma interrupção pendente
Não-interrupção	`NI`	Não existe interrupção pendente

ARQ. CONJUNTO DE INSTRUÇÕES

Instruções de Transferência de Dados

TABELA 10.10: Instruções de transferência de dados do processador P3.

Instrução	Mnemónica	Exemplo
Copiar o conteúdo	MOV	MOV R1, M[R2]
Copiar octeto menos significativo	MVBL	MVBL M[Pos1], R3
Copiar octeto mais significativo	MVBH	MVBL R3, R4
Trocar o conteúdo	XCH	XCH R1, M[R2]
Colocar na pilha	PUSH	PUSH R1
Remover da pilha	POP	POP M[R5+4]

outras instruções

TABELA 10.11: Outras instruções do processador P3.

Instrução	Mnemónica
Activar interrupções	ENI
Desactivar interrupções	DSI
Activar bit de transporte	STC
Desactivar bit de transporte	CLC
Complementar bit de transporte	CMC
Operação nula	NOP

UNIDADE DE CONTROLO BIBLIOGRAFIA

- **Bibliografia**

[1] M. Morris Mano, Charles R. Kime, "Logic and Computer Design Fundamentals", 5th Edition, Prentice-Hall International, 2016.

- Cap. 9: Computer Design Basics

[2] G. Arroz, J. Monteiro, A. Oliveira, "Arquitectura de Computadores: dos Sistemas Digitais aos Microprocessadores", IST Press, 2009.

- Cap. 10: Estrutura Interna de um processador

- **Outras Referências**

[3] J. Hennessy, D. Patterson, "Computer Architecture – A Quantitative Approach", Morgan Kaufmann, 2007.

[4] D. Patterson, J. Hennessy, "Computer Organization and Design", Morgan Kaufmann, 2009.