Especificação da Codificação das Instruções

OPCODES

0000 - NOP
0010 - MOVEQ
0011 - MOVE
0100 - ADD
0110 - SUBX
1111 - JUMP
1001 - CMP
1010 - BEQ
1011 - BGT

SELEÇÃO DE OPERAÇÃO DA ULA

00 - SOMA
01 - SUBTRAÇÃO
11 - ENT0

Carga de constante: MOVEQ

Página 114

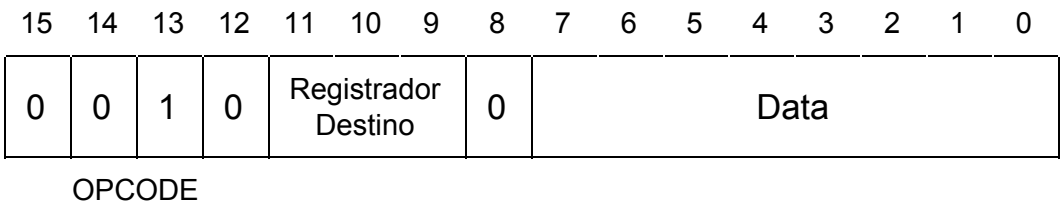
MOVEQ **Move Quick** **MOVEQ**

Operation: Immediate Data → Destination

Assembler Syntax: MOVEQ # < data > ,Dn

Attributes: Size = Long

Description: Moves a byte of immediate data to a 32-bit data register. The data in an 8-bit field within the operation word is sign- extended to a long operand in the data register as it is transferred.



Cópia de Valor Entre Registradores:MOVE/MOVEA - Move dados da origem para o destino

Página 105

MOVE
MOVEA

Move Data from Source to Destination

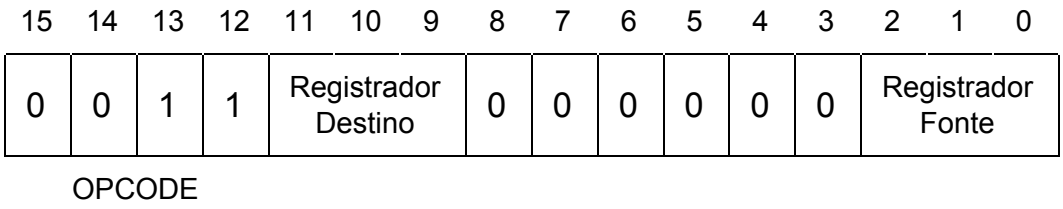
MOVE
MOVEA

Operation: Source → Destination

Assembler Syntax:
MOVE < ea > , < ea >
MOVEA <ea>, An

Attributes: Size = Byte, Word, Long

Description: Moves the data at the source to the destination location and sets the condition codes according to the data. The size of the operation may be specified as byte, word, or long word.



Soma de dois valores: ADD - Adição

Página 55

ADD **Add** **ADD**

Instruction Fields:

Register field—specifies any of the 8 data registers.

Opmode field

Long	Operation
010	< ea > + Dn
110	Dn + < ea > → < ea >

Effective Address field—determines addressing mode

a. If the location <ea> specified is a source operand, use addressing modes listed in the following table:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	Registrador Destino			0	0	0	0	0	0	Registrador Fonte		

OPCODE

Subtração de dois valores: SUB - Subtração

Página

SUB **Subtract** **SUB**

Operation: Destination – Source → Destination

Assembler Syntax: SUB < ea > ,Dn
SUB Dn, < ea >

Attributes: Size = Long

Description: Subtracts the source operand from the destination operand and stores the result in the destination. The size of the operation is specified as a long word. The mode of the instruction indicates which operand is the source and which is the destination.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	0	Registrador Destino			0	0	0	0	0	0	Registrador Fonte		

OPCODE

Desvio incondicional: JUMP - Salto incondicional

Página 99

JMP **Jump** **JMP**

Operation: Destination Address → PC

Assembler Syntax: JMP < ea >

Attributes: Unsized

Description: Program execution continues at the effective address specified by the instruction. The addressing mode for the effective address must be a control addressing mode.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	1	1	0	0	0	0	0	0	0	0	0	Registrador com Endereço		

OPCODE

Desvio condicional: BGT - Salto com condição “MAIOR QUE”

Página 71

Bcc **Branch Conditionally** **Bcc**

Operation: If Condition True
Then PC + $d_n \rightarrow$ PC

Assembler Syntax: Bcc < label >

Attributes: Size = Word or Long

Description: If the specified condition is true, program execution continues at location (PC) + displacement. The program counter contains the address of the instruction word for the Bcc instruction, plus two. The displacement is a two’s-complement integer that represents the relative distance in bytes from the current program counter to the destination program counter. If the 8-bit displacement field in the instruction word is 0, a 16-bit displacement (the word immediately following the instruction) is used. Condition code cc specifies one of the following conditional tests (refer to Table 3-19 for more information on these conditional tests):

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	1	1	0	0	0	0	Data							

OPCODE

Desvio condicional: BEQ - Salto com condição “IGUAL”

Página 71

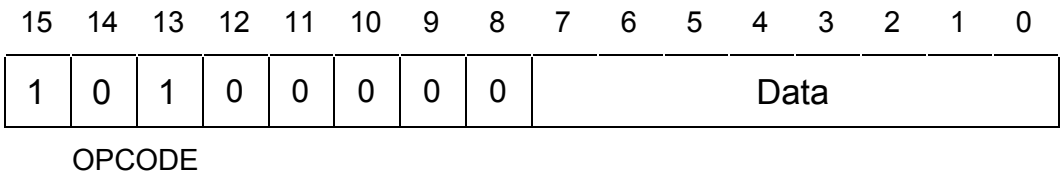
Bcc **Branch Conditionally** **Bcc**

Operation: If Condition True
Then PC + $d_n \rightarrow$ PC

Assembler Syntax: Bcc < label >

Attributes: Size = Word or Long

Description: If the specified condition is true, program execution continues at location (PC) + displacement. The program counter contains the address of the instruction word for the Bcc instruction, plus two. The displacement is a two’s-complement integer that represents the relative distance in bytes from the current program counter to the destination program counter. If the 8-bit displacement field in the instruction word is 0, a 16-bit displacement (the word immediately following the instruction) is used. Condition code cc specifies one of the following conditional tests (refer to Table 3-19 for more information on these conditional tests):



Comparação entre registradores: CMP

Página 90

CMP **Compare** **CMP**

Operation: Destination – Source → cc

Assembler Syntax: CMP < ea > , Dn

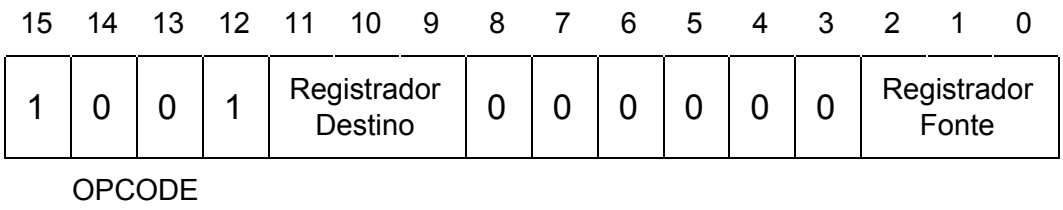
Attributes: Size = Long

Description: Subtracts the source operand from the destination data register and sets the condition codes according to the result; the data register is not changed. The size of the operation is specified as a long word.

Condition Codes:

X	N	Z	V	C
—	*	*	*	*

X — not affected
N — set if the result is negative; cleared otherwise
Z — set if the result is zero; cleared otherwise
V — set if an overflow occurs; cleared otherwise
C — set if a borrow occurs; cleared otherwise



CÓDIGO ASSEMBLY

-- INSTRUCOES	ASSEMBLY	PASSO
0 => 0010011000000000	-- MOVEQ #0, D3	- 1
1 => 0010100000000000	-- MOVEQ #0, D4	- 2
2 => 0100100000000011	-- MOVE D3, D4	- 3
3 => 0010001000000001	-- MOVEQ #1, D1	-|
4 => 0100011000000001	-- ADD D1, D3	|- 4 -|
5 => 0010010000011110	-- MOVEQ #30, D2	-|
6 => 1001011000000010	-- CMP D2, D3	| -|- 5
7 => 1011000011111100	-- BGT -4	| -|
8 => 0011101000000100	-- MOVE D4, D5	- 6