

CONTROL DATA



CONTROL DATA
CORPORATION

Documentation Department

3145 Porter Drive
Palo Alto, California 94304

60191900A ©CONTROL DATA CORPORATION Printed in U.S.A.
APRIL 1968



INSTANT
6400/6500/6600
COMPASS

6400/6500/6600 COMPASS

COMPASS is the comprehensive assembly system for the CONTROL DATA® 6400, 6500, and 6600 computers. Running under control of the SCOPE Operating System, COMPASS makes the instruction repertoire of the central and peripheral processors of these machines accessible to programmers who wish to exercise direct control of computer operations. The extensive pseudo instruction repertoire and macro facilities give programmers the advantage of maximum program construction flexibility, including control of the assembly process.

FEATURES

Language Format	Free field
Assembly Mode	Relocatable or absolute; decimal or octal
Conditional Assembly	Character string comparison; expression comparison; symbol attribute tests
Code Duplication	Number of iterations specified by programmer
Data Generation	Data may be declared in sub-program or generated at load time
Macros	System and programmer defined
Micros	Character string definitions
Listing Control	Many output listing options

SOURCE DECK



END

IDENT SCAMB

COMPASS LANGUAGE

Character Set

All characters with display code values from 01-76

Source Statement Fields

LOCATION

The location field must begin in column 1 or 2. An asterisk in column 1 indicates a comment statement; a comma in column 1 indicates a continuation line. Nine continuation cards are permitted.

OPERATION

The operation field begins after the first blank following the location field, not before column 3, and before column 36.

VARIABLE

The variable field begins after the first blank following the operation field and before column 36.

COMMENTS

The comments field begins after the first blank following the variable field, and not before column 36, if the variable field is empty.



Symbols

Up to 8 characters; the first may not be \$ = or numeric and none may be + - * / blank or comma.

Linkage symbols are restricted to 7 characters and must begin with an alphabetic character. Linkage symbols must be used for:

- Subprogram names
- External symbols
- Entry points
- Common block names

PP subprogram names may begin with an alphanumeric character and must not exceed 3 characters.

Names

A name may be formed from 1-8 characters except , - ; or blank, and may not be used in an address expression. Names are used in the following contexts:

- Block names
- Macro names
- Micro names
- Instruction bracket names

Absolute Data

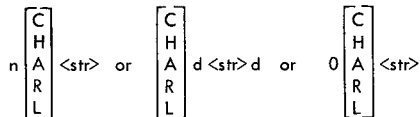
Absolute data may be used in data items (LIT and DATA subfields, and literals) or in address expressions as constants.

Register Names

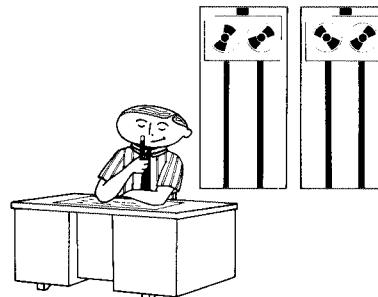
Address registers, 18 bits A0,A1,...,A7 or A.n
 Index registers, 18 bits B0,B1,...,B7 or B.n
 Arithmetic and operand registers, 60 bits X0,X1,...,X7 or X.n

n is a symbol or a single digit between 0 and 7

CHARACTER DATA



- n Character count; if n is preceded by -, character string is complemented; n must not be blank for address constants
- d Delimiter character; if n is 0 string is terminated by + - * / , Δ for address constants, or by blank or comma for DATA, LIT, or a literal
- C Left justify character string with zero fill; two terminating zeros guaranteed
- H Left justify character string with trailing blanks
- A Right justify character string with leading blanks
- R Right justify character string with leading zeros
- L Left justify character string with trailing zeros
- str Character string excluding d and ; and -



NUMERIC DATA

$$\pm \begin{bmatrix} D \\ O \\ B \end{bmatrix} n.n \begin{bmatrix} E \\ EE \end{bmatrix} \pm n \quad S \pm n \quad P \pm n$$

Omitted signs are assumed positive; modifiers following n.n may appear in any order. Only fixed point values are permitted in peripheral assemblies.

$\begin{bmatrix} D \\ O \\ B \end{bmatrix}$ Radix; identification of n.n as a decimal or octal number may appear either at beginning or end; if omitted, the radix is determined according to the BASE pseudo instruction. (Assumed decimal if no BASE.)

n.n Integer and fractional parts; if .n is omitted, the value is integer. The maximum value is 32 significant octal digits or $7.9 \cdot 10^{28}$.

$\begin{bmatrix} E \\ EE \end{bmatrix} \pm n$ Single or double precision decimal scale, maximum value 32767

$S \pm n$ Binary scale, maximum value 32767

$P \pm n$ Binary point position for floating point numbers; the binary point will occur to right of nth bit; the exponent will be adjusted to a value of - (P scale factor).

Default Symbols

=Sname name is written according to the rules for symbols. name will be defined by COMPASS if not defined by programmer.

=Xname name will be defined as external symbol if not defined by programmer.

Literals

CHARACTER DATA

Character data literals use the same basic format as data items:

=0 $\begin{bmatrix} C \\ H \\ A \\ R \\ L \end{bmatrix} \langle str \rangle \Delta$ Delimited by subfield end

=n $\begin{bmatrix} C \\ H \\ A \\ R \\ L \end{bmatrix} \langle n \text{ characters} \rangle$ Delimited by character count

= $\begin{bmatrix} C \\ H \\ A \\ R \\ L \end{bmatrix} d \langle str \rangle d$ Delimited by character d

NUMERIC DATA

The numeric value may be integer, fixed point, or floating point data item; if the sign is omitted, the value is assumed positive.

=Bnumeric or =numericB Octal
 =Onumeric or =numericO Octal
 =Dnumeric or =numericD Decimal

Counters

* or *L Current value of Location counter
 *O Origin counter
 \$ Position counter

CENTRAL PROCESSOR CODES

Address Expression

Elements are joined by multiplication or division operators to form terms; terms are joined by addition or subtraction operators to form address expressions.

+	Addition
-	Subtraction
*	Multiplication
/	Division

Terms are evaluated from left to right, then expressions are evaluated from left to right. Two or more contiguous operators are assumed to have intervening elements with a value of zero. Literals may be used only as the last term in expressions. Evaluated expressions must result in an absolute value, an external value \pm constant, or \pm relocatable value \pm constant.

ELEMENTS

Symbols

Constants

*, *L, *O, or \$

=Ssymbol or =Xsymbol

TERMS

Within a term the following rules apply:

Remainders of division are dropped

Division by zero produces a zero result, no error

Only one relocatable or external element

Only absolute values as divisors or to the left of a division

Omitted last element is assumed zero

X_i, X_j, X_k X register symbols

A_i, A_j, A_k A register symbols

B_i, B_j, B_k B register symbols

$i, j,$ and k may have values 0-7

K Address expression, 18 bits

n Absolute address, 6 bits

MNEMONIC	OCTAL	BITS	INSTRUCTION
PS	0000 000000	30	Program stop
RJ K	0100 K	30	Return jump to K
RE B_j+K	011j K	30	Read extended core storage
WE B_j+K	012j K	30	Write extended core storage
XJ B_j, K	0130 000000 46000 46000	60	Exchange jump
JP K	0200 K	30	Jump to K
JP B_j+K	020j K	30	Jump to B_j+K
ZR X_j, K	030j K	30	Jump to K if $X_j=0$
NZ X_j, K	031j K	30	Jump to K if $X_j \neq 0$
PL X_j, K	032j K	30	Jump to K if $X_j \geq 0$
NG X_j, K	033j K	30	Jump to K if $X_j < 0$
IR X_j, K	034j K	30	Jump to K if X_j is in range
OR X_j, K	035j K	30	Jump to K if X_j is out of range
DF X_j, K	036j K	30	Jump to K if X_j is definite
ID X_j, K	037j K	30	Jump to K if X_j is indefinite
ZR K	0400 K	30	Jump to K
EQ K	0400 K	30	Jump to K

EQ	B_i, K	04i0 K	30	Jump to K if $B_i = 0$	$LX_i \begin{bmatrix} B_j, X_k \\ X_k, B_j \end{bmatrix}$	22ijk	15	Shift X_k B_{j0-5} places $\rightarrow X_i$ Left shift if B_j positive Right shift if B_j negative
ZR	B_i, K	04i0 K	30	Jump to K if $B_i = 0$	$AX_i \begin{bmatrix} B_j, X_k \\ X_k, B_j \end{bmatrix}$	23ijk	15	Shift X_k B_{j0-5} places $\rightarrow X_i$ Right shift if B_j positive Left shift if B_j negative
EQ	B_i, B_j, K	04ij K	30	Jump to K if $B_i = B_j$	$NX_i \begin{bmatrix} X_k \\ B_j, X_k \\ X_k, B_j \end{bmatrix}$	24i0k	15	Normalize $X_k \rightarrow X_i$
NZ	B_i, K	05i0 K	30	Jump to K if $B_i \neq 0$	$NX_i \begin{bmatrix} X_k \\ X_k, B_j \end{bmatrix}$	24ijk	15	Normalize $X_k \rightarrow X_i$ Shift count $\rightarrow B_j$
NE	B_i, K	05i0 K	30	Jump to K if $B_i \neq 0$	$ZX_i \begin{bmatrix} X_k \\ X_k, B_j \end{bmatrix}$	25i0k	15	Round and normalize X_k in X_i
NE	B_i, B_j, K	05ij K	30	Jump to K if $B_i \neq B_j$	$ZX_i \begin{bmatrix} B_j, X_k \\ X_k, B_j \end{bmatrix}$	25ijk	15	Round and normalize $X_k \rightarrow X_i$ Shift count $\rightarrow B_j$
PL	B_i, K	06i0 K	30	Jump to K if $B_i \geq 0$	$UX_i \begin{bmatrix} X_k \\ B_j, X_k \\ X_k, B_j \end{bmatrix}$	26i0k	15	Unpack X_k to X_i
GE	B_i, K	06i0 K	30	Jump to K if $B_i \geq 0$	$UX_i \begin{bmatrix} B_j, X_k \\ X_k, B_j \end{bmatrix}$	26ijk	15	Unpack X_k : coefficient $\rightarrow X_i$, Exponent $\rightarrow B_j$
GE	B_i, B_j, K	06ij K	30	Jump to K if $B_i \geq B_j$	$PX_i \begin{bmatrix} B_j, X_k \\ X_k, B_j \end{bmatrix}$	27ijk	15	Pack coefficient X_k , Exponent $B_j \rightarrow X_i$
LE	B_j, K	060j K	30	Jump to K if $B_j \leq 0$	$FX_i \begin{bmatrix} X_j + X_k \\ X_j - X_k \end{bmatrix}$	30ijk	15	Floating $X_j + X_k \rightarrow X_i$
LE	B_j, B_i, K	06ij K	30	Jump to K if $B_j \leq B_i$	$FX_i \begin{bmatrix} X_j - X_k \end{bmatrix}$	31ijk	15	Floating $X_j - X_k \rightarrow X_i$
NG	B_i, K	07i0 K	30	Jump to K if $B_i < 0$	$DX_i \begin{bmatrix} X_j + X_k \end{bmatrix}$	32ijk	15	Floating double precision $X_j + X_k \rightarrow X_i$
LT	B_i, K	07i0 K	30	Jump to K if $B_i < 0$	$DX_i \begin{bmatrix} X_j - X_k \end{bmatrix}$	33ijk	15	Floating double precision $X_j - X_k \rightarrow X_i$
LT	B_i, B_j, K	07ij K	30	Jump to K if $B_i < B_j$	$RX_i \begin{bmatrix} X_j + X_k \end{bmatrix}$	34ijk	15	Rounded floating $X_j + X_k \rightarrow X_i$
GT	B_j, K	070j K	30	Jump to K if $B_j > 0$	$RX_i \begin{bmatrix} X_j - X_k \end{bmatrix}$	35ijk	15	Rounded floating $X_j - X_k \rightarrow X_i$
GT	B_j, B_i, K	07ij K	30	Jump to K if $B_j > B_i$	$IX_i \begin{bmatrix} X_j + X_k \end{bmatrix}$	36ijk	15	Integer $X_j + X_k \rightarrow X_i$
BXi	X_j	10ijj	15	$X_j \rightarrow X_i$	$IX_i \begin{bmatrix} X_j - X_k \end{bmatrix}$	37ijk	15	Integer $X_j - X_k \rightarrow X_i$
BXi	$X_j * X_k$	11ijk	15	Log. prod. of X_j and $X_k \rightarrow X_i$				
BXi	$X_j + X_k$	12ijk	15	Log. sum of X_j and $X_k \rightarrow X_i$				
BXi	$X_j - X_k$	13ijk	15	Log. diff. of X_j and $X_k \rightarrow X_i$				
BXi	$-X_k$	14ikk	15	Comp. of $X_k \rightarrow X_i$				
BXi	$-X_k * X_j$	15ijk	15	Log. prod. of X_j and X_k comp. $\rightarrow X_i$				
BXi	$-X_k + X_j$	16ijk	15	Log. sum of X_k comp. and $X_j \rightarrow X_i$				
BXi	$-X_k - X_j$	17ijk	15	Log. diff. of X_k comp. and $X_j \rightarrow X_i$				
LXi	jk	20ijk	15	Shift X_i left-circular jk places				
AXi	jk	21ijk	15	Shift X_i right jk places				

FXi	$X_j * X_k$	40ijk	15	Floating $X_j * X_k \rightarrow X_i$
RXi	$X_j * X_k$	41ijk	15	Rounded floating $X_j * X_k \rightarrow X_i$
DXi	$X_j * X_k$	42ijk	15	Floating double precision $X_j * X_k \rightarrow X_i$
MXi	n	43in	15	Form mask of n bits in X_i
FXi	X_j / X_k	44ijk	15	Floating $X_j / X_k \rightarrow X_i$
RXi	X_j / X_k	45ijk	15	Rounded floating $X_j / X_k \rightarrow X_i$
NO		4600	15	No operation
CXi	X_k	47ikk	15	Ones in $X_k \rightarrow X_i$
SAi	$A_j + K$	50ij K	30	$A_j + K \rightarrow A_i$
SAi	K	51i0 K	30	$K \rightarrow A_i$
SAi	$B_j + K$	51ij K	30	$B_j + K \rightarrow A_i$
SAi	$X_j + K$	52ij K	30	$X_j + K \rightarrow A_i$
SAi	X_j	53ij0	30	$X_j \rightarrow A_i$
SAi	$\begin{bmatrix} X_j + B_k \\ B_k + X_j \end{bmatrix}$	53ijk	15	$X_j + B_k \rightarrow A_i$
SAi	A_j	54ij0	15	$A_j \rightarrow A_i$
SAi	$\begin{bmatrix} A_j + B_k \\ B_k + A_j \end{bmatrix}$	54ijk	15	$A_j + B_k \rightarrow A_i$
SAi	$\begin{bmatrix} A_j - B_k \\ -B_k + A_j \end{bmatrix}$	55ijk	15	$A_j - B_k \rightarrow A_i$
SAi	B_j	56ij0	15	$B_j \rightarrow A_i$
SAi	$B_j + B_k$	56ijk	15	$B_j + B_k \rightarrow A_i$
SAi	$-B_k$	57i0k	15	$-B_k \rightarrow A_i$
SAi	$\begin{bmatrix} B_j - B_k \\ -B_k + B_j \end{bmatrix}$	57ijk	15	$B_j - B_k \rightarrow A_i$
SBi	$A_j + K$	60ij K	30	$A_j + K \rightarrow B_i$
SBi	K	61i0 K	30	$K \rightarrow B_i$
SBi	$B_j + K$	61ijk	30	$B_j + K \rightarrow B_i$
SBi	$X_j + K$	62ij K	30	$X_j + K \rightarrow B_i$

SBi	X_j	63ij0	15	$X_j \rightarrow B_i$
SBi	$\begin{bmatrix} X_j + B_k \\ B_k + X_j \end{bmatrix}$	63ijk	15	$X_j + B_k \rightarrow B_i$
SBi	A_j	64ij0	15	$A_j \rightarrow B_i$
SBi	$\begin{bmatrix} A_j + B_k \\ B_k + A_j \end{bmatrix}$	64ijk	15	$A_j + B_k \rightarrow B_i$
SBi	$\begin{bmatrix} A_j - B_k \\ -B_k + A_j \end{bmatrix}$	65ijk	15	$A_j - B_k \rightarrow B_i$
SBi	B_j	66ij0	15	$B_j \rightarrow B_i$
SBi	$B_j + B_k$	66ijk	15	$B_j + B_k \rightarrow B_i$
SBi	$-B_k$	67i0k	15	$-B_k \rightarrow B_i$
SBi	$\begin{bmatrix} B_j - B_k \\ -B_k + B_j \end{bmatrix}$	67ijk	15	$B_j - B_k \rightarrow B_i$
SXi	$A_j + K$	70ij K	30	$A_j + K \rightarrow X_i$
SXi	K	71i0K	30	$K \rightarrow X_i$
SXi	$B_j + K$	71ijK	30	$B_j + K \rightarrow X_i$
SXi	$X_j + K$	72ij K	30	$X_j + K \rightarrow X_i$
SXi	X_j	73ij0	15	$X_j \rightarrow X_i$
SXi	$\begin{bmatrix} X_j + B_k \\ B_k + X_j \end{bmatrix}$	73ijk	15	$X_j + B_k \rightarrow X_i$
SXi	A_j	74ij0	15	$A_j \rightarrow X_i$
SXi	$\begin{bmatrix} A_j + B_k \\ B_k + A_j \end{bmatrix}$	74ijk	15	$A_j + B_k \rightarrow X_i$
SXi	$\begin{bmatrix} A_j - B_k \\ -B_k + A_j \end{bmatrix}$	75ijk	15	$A_j - B_k \rightarrow X_i$
SXi	B_j	76ij0	15	$B_j \rightarrow X_i$
SXi	$B_j + B_k$	76ijk	15	$B_j + B_k \rightarrow X_i$
SXi	$-B_k$	76i0k	15	$-B_k \rightarrow X_i$
SXi	$\begin{bmatrix} B_j - B_k \\ -B_k + B_j \end{bmatrix}$	77ijk	15	$B_j - B_k \rightarrow X_i$

PERIPHERAL PROCESSOR CODE

The variable field may contain index or address values. Subfields are separated by commas.

m	Address value, 12 bits
c	Address value, 18 bits
d	Index value, 6 bits
r	$-31 \leq r \leq 31$
M	Indexed address (mtd), 18 bits

MNEMONIC	OCTAL	LENGTH	INSTRUCTION
PSN	0000	12	Pass
LJM	m,d 01dd mmmm	24	Long jump to M
RJM	m,d 02dd mmmm	24	Return jump to M
UJN	r 03rr	12	Unconditional jump r locations
ZJN	r 04rr	12	Jump r locations if (A) = 0
NJN	r 05rr	12	Jump r locations if (A) \neq 0
PJN	r 06rr	12	Jump r locations if (A) \geq 0
MJN	r 07rr	12	Jump r locations if (A) < 0
SHN	d 10dd	12	Shift (A) d places, d positive: left circular; d negative: right end off, no sign extension
LMN	d 11dd	12	Log. diff. of d and (A ₅₋₀) \rightarrow (A ₆₋₁₁) unchanged
LPN	d 12dd	12	Log. prod. of d and (A ₅₋₀) \rightarrow A (A ₆₋₁₁) are zero.
SCN	d 13dd	12	Clear bits in A ₀₋₅ for corresponding ones in d
LDN	d 14dd	12	d \rightarrow A
LCN	d 15dd	12	-d \rightarrow A

ADN	d 16dd	12	(A) + d \rightarrow A
SBN	d 17dd	12	(A) - d \rightarrow A
LDC	c 20cc cccc	24	c \rightarrow A
ADC	c 21cc cccc	24	c + (A) \rightarrow A
LPC	c 22cc cccc	24	Log. prod. of A and C \rightarrow A
LMC	c 23cc cccc	24	Log. diff. of A and c \rightarrow A
EXN	d 260d	12	Exchange jump
MXN	d 261d	12	Monitor exchange jump
RPN	2700	12	Central processor address \rightarrow A
LDD	d 30dd	12	(location d) \rightarrow A
ADD	d 31dd	12	(A) + (location d) \rightarrow A
SBD	d 32dd	12	(A) - (location d) \rightarrow A
LMD	d 33dd	12	Log. diff. of A and location d \rightarrow A
STD	d 34dd	12	(A ₁₁₋₀) \rightarrow location d
RAD	d 35dd	12	(A) + (location d) \rightarrow d and A
AOD	d 36dd	12	(location d) + 1 \rightarrow location d and A
SOD	d 37dd	12	(location d) - 1 \rightarrow location d and A
LDI	d 40dd	12	((location d)) \rightarrow A
ADI	d 41dd	12	(A) + ((location d)) \rightarrow A
SBI	d 42dd	12	(A) - ((location d)) \rightarrow A
LMI	d 43dd	12	Log. diff. of A and (location d) \rightarrow A
STI	d 44dd	12	(A ₁₁₋₀) \rightarrow ((location d))
RAI	d 45dd	12	(A) + ((location d)) \rightarrow A
AOI	d 46dd	12	((location d)) + 1 \rightarrow (d) and A

SOI	d	47dd	12	((location d)) - 1 → (d) and A
LDM	m,d	50 dd mmmm	24	(M) → A ₀₋₁₁ ; 0 → A ₁₂₋₁₇
ADM	m,d	51dd mmmm	24	(A) + (M) → A
SBM	m,d	52dd mmmm	24	(A) - (M) → A
LMM	m,d	53dd mmmm	24	Log. diff. of A and (M) → A
STM	m,d	54dd mmmm	24	(A ₀₋₁₁) → M
RAM	m,d	55dd mmmm	24	(A) + (M) → M and A
AOM	m,d	56dd mmmm	24	(M + 1) → M and A
SOM	m,d	57dd mmmm	24	(M - 1) → M and A
CRD	d	60 dd	12	Read central memory (A) → d, ..., d + 4
CRM	m,d	61dd mmmm	24	Read central memory (d) words, beginning with (A) to m, m + 1, ..., m + 5d - 1
CWD	d	62dd	12	Write from locations d to d + 4 into central memory address (A)
CWM	m,d	63 dd mmmm	24	Write d words beginning with m to (A) in central memory
AJM	m,d	64dd mmmm	24	Jump to m if channel d active
IJM	m,d	65dd mmmm	24	Jump to m if channel d inactive
FJM	m,d	66dd mmmm	24	Jump to m if channel d full
EJM	m,d	67dd mmmm	24	Jump to m if channel d empty
IAN	d	70dd	12	Input word from channel d to A ₀₋₁₁
IAM	m,d	71dd mmmm	24	Input A words to m from channel d
OAN	d	72dd	12	Output from A on channel d
OAM	m,d	73dd mmmm	24	Output A words from m on channel d

ACN	d	74 dd	12	Activate channel d
DCN	d	75dd	12	Deactivate channel d
FAN	d	76dd	12	External function code (A ₀₋₁₁) → channel d
FNC	m,d	77d mmmm	24	External function code (m) → channel d



Storage Allocation

BSS expression	Storage Reservation
	Location field symbol is assigned the value of the location counter, and location and origin counters are incremented by the value of the address field expression.
BSSZ expression	Storage Reservation
	Location field symbol is assigned the value of the location counter, and the counters are incremented by the value of the address field expression. At load time, the number of words specified by the expression will be set to zero.

Symbol Definition

symbol EQU expression	Symbol Definition
	Symbol in the location field is assigned the value of the address field expression.
symbol SET expression	Symbol Redefinition
	Symbol in the location field is redefined to the value of the address field expression.

Data Generation

DATA data items	Data Declaration
	Symbol in location field is assigned the value of the location counter. Subfields, separated by commas, may be numeric or character data items.

DIS wc,characters	Display Code Lines
	Symbol in the location field is assigned the value of the location counter. wc is the word count: wc - 10 (CP) or wc - 2 (PP) characters beyond the comma are extracted. If wc is blank or zero, the first character after the comma is considered a delimiter, and characters are extracted until the delimiter is again encountered.
LIT data items	Literal Values
	Symbol in the location field is assigned the value of the location counter. Up to 100 words of data items, separated by commas, may be included in one LIT instruction
VFD subfields	Field Definition
	Symbol in the location field assigns the subfields beginning in a new word. A - in the location field positions the counter at the next quarter word boundary in a CP assembly. Subfields appear as n/v; n is a single element bit count, previously defined and absolute, maximum value of 60; v is an expression. If v is not absolute, the field must be at least 18 bits long, ending at bit number 0, 15, or 30.
REP and REPI subfields	Data Generation
	Generate data at load time. Subfields, up to 5, may appear in any order, separated by commas. Subfield format is: specification/non-external address expression
	Specifications
	S Source address, mandatory
	D Destination address, S+B if omitted
	C Repetition count, 1 if omitted
	B Code block size, 1 if omitted
	I Increment, B if omitted

Conditional Operations

IFxx field₁, field₂, count

Compare Values

Location field contains instruction bracket name or blank; variable field contains 2 address expressions, separated by commas, for comparison.

Optional count is number of lines to be assembled if comparison is satisfied.

xx	Comparison of Fields
EQ	Equal
NE	Not equal
GT	Greater than
GE	Greater than or equal
LT	Less than
LE	Less than or equal

IFyy count

Test Assembly Environment

Location field contains instruction bracket name or blank. Optional count is number of lines to be assembled if condition is true.

yy	Condition
PP	Peripheral assembly in progress
CP	Central assembly in progress

IF attribute, symbol expression count Test Symbol or Expression Attribute

Location field contains instruction bracket name or blank. Optional count is number of lines to be assembled if attribute is true. A minus before an attribute tests for the negative condition.

Attribute	Test
SET	Symbol defined by SET
ABS	Absolute expression
REL	Common or program relocatable expression
REG	Register name in the expression
COM	Common relocatable expression
EXT	External symbol in expression
LOC	Program relocatable expression
DEF	All symbols in expression defined

IFC xx,dc...cdc...cd,count

Test Character Strings

Location field contains instruction bracket name or blank. Optional count is number of lines to be assembled if comparison is true.

Delimiter is d; c...c is a character string; xx is EQ,NE,GT,GE,LT, or LE

ENDIF

Conditional Assembly Terminator

Location field contains instruction bracket name or blank. ENDF is ignored if it appears within a line count-controlled range.

List Control

Extent of Listing

List options

Options are separated by commas.

Option	Listing	Nominal Condition
L	List control	on
M	Macro expansion control	off
E	DUP control	off
D	VFD, DATA, DIS; RMT; literals, deferred symbols	off
F	Conditional assembly	off
C	Control cards EJECT, SPACE, TITLE	off
R	Reference table	on
X	XTEXT text	off
S	Systems macro expansion	off
G	Code generation	off
A	Actual assembly list	off
N	Programmer nulls	off
T	SST nulls	off

EJECT Start New Page

SPACE exp Skip Line

Skip number of lines indicated by value of address field expression.

TITLE string Titling

First title string in subprogram is listed on every page; subsequent TITLES are subtitles which cause page ejects before listing. The title string begins immediately after the pseudo operation code and continues for 79 columns or to end-of-statement.

Code Duplication

DUP exp1,exp2 Duplication

Location field may contain an instruction bracket name or blank. Replication count is specified by the value of the first address expression. Optional second address specifies number of succeeding lines to be assembled.

ENDD End Duplication

Terminates range if second address field expression was omitted in preceding DUP. Location field may contain an instruction bracket name or blank.

STOPDUP Stop Duplication

Stops duplication at end of current iteration.

Micros

name MICRO n1,n2,dccc...ccd Micro Definition

The micro string is formed by extracting n2 characters from ccc...cc, beginning with the character specified by n1. If n1 is zero or blank, the character string is empty. If n2 is zero or blank, the length of the string is delimited by the character d.

~~name~~ Micro Substitution

Named micro string is substituted by COMPASS wherever ~~name~~ appears in the line.

Remote Assembly

RMT Save Code

Instructions up to the next RMT pseudo instruction are saved for later assembly.

HERE Assemble RMT Code

Saved remote instructions are assembled at this point.

Loader Directives

LCC string Loader Directives

Character string is passed to binary output file for subsequent recognition by SCOPE loader.

ERR Forced Error

A fatal error is produced.

file XTEXT record External Input
Assembles data from named record on named indexed file. If record name is not given, first record of file is used.

SST System Symbols
Defines system symbols from the system file as if they had been defined by the routine.

Macros

name MACRO arguments Standard Macro Heading
Arguments must begin with a letter; up to 63 may be listed, separated by special characters: ,.+*/)(\$. Subsequent instructions until ENDM are saved as a macro definition.

MACRO name, arguments Alternate Macro Heading
The first subfield is the macro name; subsequent subfields are macro arguments. Subfields are separated by commas.

name OPDEF arguments Special Macro Form Heading
Location field entry is abbreviated description of entire instruction to be recognized as an OPDEF call. Address subfields are formal arguments listed as for MACRO. Provides convenient description of macros in CP machine instruction format.

LOCAL symbols Local Symbols
Symbols local to macro are separated by commas. Total number of LOCAL symbols and macro arguments must be less than 64.

ENDM Macro Terminator
Location field contains macro name or blank.

name arguments Standard Macro call
Non-blank location field forces upper. Arguments are substituted for formal arguments of definition.

name arguments Alternate Macro Call
Macro call location field is substituted for first parameter in definition argument list.

Assembler Input/Output

COMPASS (Input/Output option list) COMPASS Call
Calls COMPASS Assembler, Input/Output options separated by commas if non-blank.

L	Listing
L or blank	According to internal LIST control on file OUTPUT
L=fname	According to internal LIST control on file fname
L=0	Headings only on file OUTPUT

I	Input
I or blank	From file INPUT
I=fname	From file fname

B	Binary Output
B or blank	On file LGO
B=fname	On file fname

S	Systems Text
S or blank	From SYSTEXT
S=rname	From library overlay named rname
S=SCPTXT	From library overlay SCPTXT

CHARACTER CODES

Character	Display	External BCD	Hollerith Punch	Character	Display	External BCD	Hollerith Punch
A	01	61	12-1	7	42	07	7
B	02	62	12-2	8	43	10	8
C	03	63	12-3	9	44	11	9
D	04	64	12-4	+	45	60	12
E	05	65	12-5	-	46	40	11
F	06	66	12-6	*	47	54	11-8-4
G	07	67	12-7	/	50	21	0-1
H	10	70	12-8	(51	34	0-8-4
I	11	71	12-9)	52	74	12-8-4
J	12	41	11-1	\$	53	53	11-8-3
K	13	42	11-2	=	54	13	8-3
L	14	43	11-3	blank	55	20	space
M	15	44	11-4	,	56	33	0-8-3
N	16	45	11-5	.	57	73	12-8-3
O	17	46	11-6	≡	60	36	0-8-6
P	20	47	11-7	[61	17	8-7
Q	21	50	11-8]	62	32	0-8-2
R	22	51	11-9	:	63	00	8-2
S	23	22	0-2	≠	64	14	8-4
T	24	23	0-3	-	65	35	0-8-5
U	25	24	0-4	√	66	52	11-0
V	26	25	0-5	^	67	37	0-8-7
W	27	26	0-6	†	70	55	11-8-5
X	30	27	0-7	‡	71	56	11-8-6
Y	31	30	0-8	<	72	72	12-0
Z	32	31	0-9	>	73	57	11-8-7
0	33	12	0	≦	74	15	8-5
1	34	01	1	≧	75	75	12-8-5
2	35	02	2	┌	76	76	12-8-6
3	36	03	3				
4	37	04	4				
5	40	05	5				
6	41	06	6				