Programming Dept. Memo 130-390-002-00

digital INTEROFFICE MEMORANDUM

DATE: March 31, 1969

SUBJECT: PAL-11 Assembler Specification

TO: Attached List

FROM: D. Barlow

0.1 OVERALL DESCRIPTION

This document defines the assembler which will be used on the PDP-11 computer. Briefly, the PDP-11 processes variable length instructions (16, 32, 48 bits) and possesses a 16 bit word length. It is assumed that the reader is familiar with the PDP-11 as described in Engineering Spec. #0001 (28 Mar. 1969).
The PDP-11 assembler PAL-11 is a multi-pass, absolute symbolic assembler which has been implemented using the PDP-11 assembly language.

0.1.1 CONVENTIONS

TERM	DEFINITION
Byte	An 8 bit quantity
Word	A 16 bit quantity
	denotes an ASCII blank or space
	denotes an ASCII horizontal tab
	denotes an ASCII carriage-return
	denotes an ASCII line-feed
F/F	denotes an ASCII form-feed

1.0 GENERAL SPECIFICATIONS

1.1 PAL-11 is designed for execution on the PDP-11 Processor with 4K words of memory with either a Model 33 or 35 Teletype.

1.2 The assembler will process significantly more user defined symbols if additional memory is available. Furthermore, the assembler will utilize high speed paper tape I/O devices and a KSR Teletype if available.

2.0 DESIGN SPECIFICATIONS

2.1 DESIGN GOALS

The primary design goal is minimal size yet maintaining upward compatibility.

2.2 INPUT

The following subsections provide a comprehensive description of the assembler's inputs.

2.2.1 INPUT FORMAT

Source programs consist of a series of source statements where every statement is terminated by either a line-feed or form-feed character. Furthermore, statements must be solely composed of printable characters (i.e. characters whose ASCII encoded value lies between (inclusive) 040_8 and 137_8. The obvious exceptions are: blanks, tabs, carriage-returns, line-feeds and form-feeds. Null and rubout characters are unconditionally ignored.

2.2.1.1 ELEMENTS OF A SOURCE IMAGE

There are potentially, four syntactical parts or fields within a source image. The non-void fields must appear in the following order:

LABEL OPERATOR OPERAND COMMENTS

The fields are identified primarily, by their order of appearance and secondarily, by the specific delimiting or terminating character.

LABEL FIELD

A label defines a user's symbol and the assembler equates that symbol with the user oriented execution location counter (i.e. the location counter). Note that labels and symbols are never syntactically equivalent (i.e. labels define symbols). Furthermore, note that labels are redefinable solely by direct assignments.

Syntactical constraints:

a) The label field must be the first encountered field within the source image.

b) The label field may be void or it may contain a single label or multiple labels.

Examples:

```
A : B : C:    MOV    A,B          /The labels A, B, and C
                                  /are assigned the memory
                                  /reference value where the
                                  /MOV command word will
                                  /be stored.

$$DOGA:$$DOGB:    MOV    A,B      /Both labels name the
                                  /symbol $$DOG. This
                                  /statement is errored due to
                                  /multiply defined symbols.
```

Assume that the location counter is at 100_8

```
A:    . = 300                     /The label A is assigned
                                  /the value 100 (8)
                                  /the location counter is
                                  /assigned the new
                                  /location value of 300 (8)
```

To redefine a label

```
A : B :    MOV    A,B            /The label A is now defined
A = 3                            /The symbol A has now been
                                 /redefined and associated
                                 /with the number 3
                                    / phase errors will occur
                                    / unless the following occurs
A = B                            /The symbol A has now
                                 /been redefined to its
                                 /original value.

A : B :                          /Label field followed by A
                                 /comments field

C : D :    MOV    A,B            /The labels A and B are
                                 /valid and are associated
                                 /with the same memory
                                 /reference value as are
                                 /the labels C and D

A:    MOV    A,B:                /The symbol B will be
                                 /errored.
```

OPERATOR FIELD

The operator field normally contains a user selected mnemonic obtained from any of the following classes:

a) A machine instruction mnemonic

   1) Binary Group
   2) Conditional or unconditional branches
   3) Unary Group
   4) Rotate/Shift Group
   5) Subroutine calls
   6) Operate Group

b) Assembly directives

Syntactical constraints:

a) The operator field may only be preceded by the label field.

b) A void operator field is itself a command (effectively a "word assembly directive" - refer to section 2.2.1.8.2).

c) Leading and trailing blanks and/or tabs are ignored.

OPERAND FIELD

The form of the operand field is totally dependent upon the addressing mode selected.

ADDRESSING MODES:

Let 'E' represent an expression (any combination of symbols, numbers or ASCII literals joined together using + - & ! as operators). The following is a list of valid addressing modes:

#E	Literal expression
%E	Register designator. $Ø \leq E \leq 7$
E	Memory address
(E)+	Auto-incrementing register. $Ø \leq E \leq 7$
-(E)	Auto-decrementing register. $Ø \leq E \leq 7$
$E(E_1)$	Indexing mode. $Ø \leq E_1 \leq 7$

Note that an @ may be prefixed to any of the forementioned forms to indicate indirect or deferred addressing.

INSTRUCTION FORMS:

Throughout this section let E represent an expression and A represent one of the forementioned addressing modes.

Binary Group

Op A,A

Unary Group

Op A

Rotate/Shift Group

Op E,A E = -1 or E = +1

Operate Group

Op

Branches

Op E 177600≤1 (E=.)/2 ≤177

Subroutine Calls

Op E,A Ø≤E≤7

Return from Subroutine

Op E Ø≤E≤7

Trap/Execute

Op E Ø≤E/2≤ØØØ377

COMMENTS FIELD

The comments field contains any valid, printable ASCII characters.

Syntactical constraints:

a) The comments field must originate with a slash character (/) and terminate upon encountering either a line feed or form-feed character.

2.2.1.2 SYMBOLS

A symbol symbolically represents a numerical quantity (a memory address in the case of labels or a numerical constant in the case of direct assignments).

The value associated with a permanent symbol is somewhat dependent upon its usage. Refer to the following.

1) A permanent symbol encountered in the operator field is associated with an op-code.

2) A permanent symbol encountered in the operand field is associated with a value using the following order of priority.

a) If the symbol is found to be user defined, then the user's value is associated with the symbol.

b) If the symbol is not found to be user defined, then the permanent op-code is associated with the symbol. The op-code is a word quantity. For example, "MOV MOV,A" would store Ø1ØØØØ into A.

Syntactical constraints:

a) The first character within any symbol must be selected from the following set:

A thru Z $.

b) The next four characters of the symbol extend this set to include:

Ø thru 9

c) The remaining characters extend this set even further to include all printable ASCII characters except:

+ - ! & ↓ F/F : = / , ' " → % # ( ) ↵ Δ

d) All characters after the first five are ignored.

Examples (valid)

A ABC	ABCDEF	ABC$D	SYS... .A
AT $..	AØ1	$12	TABLE[1]

Examples (invalid)

8AB	A↓B	A(I)	ABΔC	.ΔX
ABCD E				

2.2.1.3 DIRECT ASSIGNMENTS

A direct assignment is a statement which introduces a symbol into the assembler and associates with that symbol a user oriented numerical value (i.e. an expression).

Syntactical constraints:

a) A direct assignment statement may be preceded by a label field and only a label field.

b) A direct assignment statement may be followed by a comment field and only a comment field.

c) Blanks and/or tabs may precede a label or they may follow a label, even to the extent of separating the symbol from the label terminator (a colon). Imbedded blanks and/or tabs within a symbol are not permitted.

d) Every label must terminate with a colon (:) character.

e) The composition of a label is equivalent to the composition of a symbol as set forth in section 2.2.1.2.

f) Independent labels must be distinct within the first five characters as the assembler ignores the remaining characters.

A label is defined equivalent to the current value of the location counter.

Note that the location counter is defined to be the numerical value associated with the memory location where the present object data byte is to be stored. Labels are semi-redefinable in that direct assignments may redefine a label whereas a label may not redefine a label (caution must be exercised while redefining a label in order to avoid assembly phase errors). Binary output is never generated for a label.

c) Only one symbol is permitted to be defined within a direct assignment statement.

d) The symbol itself must conform to the constraints as set forth in section 2.2.1.2.

e) An equals sign (=) must separate the symbol from the expression defining that symbol.

f) Independent symbols must be distinct within the first five characters.

g) Blanks and/or tabs are ignored unless imbedded.

A direct assignment is totally redefinable in that both direct assignments and labels may redefine the symbol (cuation must be exercised when redefining a direct assignment symbol which, syntactically, is equivalent to a label in order to avoid phase errors). Furthermore, only one level of forward referencing is permitted. Binary output is never generated for a direct assignment.

Examples:

```
A = 1                          /The symbol A is equated with 1
B = 'A+1 & MASKLOW             /The symbol B is equated with the
                               /expression's resultant.
C:D=3                          /The labels C and E are equated
E:   MOV     #1, ABLE          /with the numerical memory
                               /reference value where the MOV
                               /command will be stored.
                               /The symbol D is equated with 3
F=2                            /The symbol F is equated with 2
F:G:   MOV   #1, ABLE          /The symbol F is redefined
                               /To equate the numerical memory
                               /reference value where the MOV
                               /command will be stored
                                  /phase errors will occur unless
                                  /the following statement occurs.
F = G                          /The symbol F is redefined to
                               /equal the value of the label G.
```

Only one level of forward referencing is permitted with direct assignments. For example:

```
X = Y
Y = Z
Z = 1
```

X is the second level of forward referencing involving X and Z. X will still be undefined at the end of pass two, therefore, all references to X during pass two will be errored.

2.2.1.4 NUMBERS

Numbers used within a source image are signed or unsigned octal integers.

Syntactical constraints:

a) Negative numbers must be preceded by one minus (-) character.

b) All numerical symbols are totally composed of characters from the following set:

Ø thru 7

c) Byte allocated values will be truncated to an 8 bit quantity.

d) Word allocated values will be truncated to a 16 bit quantity.

2.2.1.5 EXPRESSIONS

An expression is a combination of terms which are joined together using a specified set of operators, thus reducing to a single 16-bit value. A term may be a symbol (permanent or user defined), a number, an ASCII literal, or the special symbol period (.). The operators, which are full word operators, must be selected from the following set:

+	arithmetic addition
-	arithmetic subtraction or a unary minute
!	logical ORing
&	logical ANDing

The evaluation proceeds, as encountered, from left to right.

Void expressions and erroneous terms are equated with zero. The void operator is arithmetic addition.

2.2.1.6 THE LOCATION OR PROGRAM COUNTER

The assembler zeroes the location counter during the initialization for each pass through the source program. Thereafter, throughout the assembler's pass, consecutive memory locations are assigned to each byte generated for the user's object program. Thus, the location counter always contains the numerical value associated with the memory location where the current users object byte will be stored. The location counter may be altered by the user via a direct assignment statement of the form . = expression) l or referenced directly via the usage of the period (.) symbol.

Example:

```
MOV     #. -2,AC        / Stores the PC minus 2 into AC
                        / The PC would contain the
                        / address of the MOV command
                        / word
```

Assume the PC contains 1ØØ

```
.=. +5Ø                 / replaces the contents of the PC
                        / with the value 15Ø8
```

2.2.1.7 GOTO EXTENDED MNEMONIC

The following table contains the various forms of the GOTO mnemonic and its syntactical definition.

GOTO	E	MOV	#E,%7
GOTO	@E	MOV	E,%7
GOTO	@@E	MOV	@E,%7
GOTO	@%E	MOV	%E,%7
GOTO	@@%E	MOV	@%E,%7
GOTO	@E(E_1)	MOV	E(E_1),%7
GOTO	@@E(E_1)	MOV	@E(E_1),%7
GOTO	@(E)+	MOV	(E)+,%7
GOTO	@@(E)+	MOV	@(E)+,%7
GOTO	@-(E)	MOV	-(E),%7
GOTO	@@-(E)	MOV	@-(E),%7

2.2.1.8 ASSEMBLY DIRECTIVES

Assembly directives are solely directives or commands to the assembler. The various directives are described in detail in the following subsections.

2.2.1.8.1 DATA GENERATION

The user may generate data by either of the two following methods.

THE BYTE ASSEMBLY DIRECTIVE

The byte directive (BYTE) is used to generate bytes of data. The forms of the directive are:

BYTE expression ↵↓

or

BYTE expression, expression, etc. ↵↓

THE WORD ASSEMBLY DIRECTIVE

The word directive (WORD is used to generate words of data. The forms of the directive are:

WORD expression ↵↓

or

WORD expression, expression, etc. ↵↓

Syntactical constraints:

a) The byte or word assembly directive statement is a complete syntactical source image.

b) The mnemonic BYTE or WORD singularly occupies the operator field.

c) The operator field may only be preceded by the label field.

d) The expression or expressions must conform to the constraints as set forth in section 2.2.1.5.

e) Multiple expressions must be delimited by commas.

THE VOID OPERATOR ASSEMBLY DIRECTIVE

The void operator field, effectively, if used to generate words of data. The forms of the statement are:

expression ↵↓

or

expression, expression, etc. ↵↓

Syntactical constraints:

Note that this statement is a defaulting situation from the processing of mnemonics; thus, the constraints are fairly rigid.

a) The statement itself is a complete syntactical source image.

b) The label field is permitted.

c) Since the operator field is void, the first encountered term in the first expression must not be a recognizable machine mnemonic unless it is preceded by a quote or an expression operator (+ - ! &). Other than this constraint, the expression or expressions must conform to the constraints as set forth in section 2.2.1.5.

d) Multiple expressions must be delimited by commas.

Examples

```
A: 1, 2, 3, 4          / Generates four words of
                       / data containing 1, 2, 3, 4
                       / respectively
B: +MOV, MOV           / Generates two words of
                       / data containing the
                       / op-codes for MOV and MOV
                       / in the high order byte
                       / of each word.
```

2.2.1.8.2 TEXT GENERATION

The user may generate text characters, represented in ASCII by any of the following methods.

APOSTROPHE MARK

A single apostrophe mark causes the 7-bit, unparitied ASCII representation of the next physically encountered character to be available to the user's program. (All characters must be printable.)

QUOTATION MARK

A single quotation mark causes the 7-bit, unparitied ASCII representations of the next two physically encountered characters to be available to the user's program. (All characters must be printable.)

THE ASCII TEXT ASSEMBLY DIRECTIVE

The ascii directive (ASCII) is used to generate 7-bit, unparitied ASCII text. The form of the directive is:

```
ASCII    /character  character etc./
```

Syntactical constraints:

a) The ascii assembly directive statement is a complete syntactical source image.

b) The mnemonic ASCII singularly occupies the operator field and must terminate with either a blank or a tab character.

c) The operator field may only be preceded by the label field.

d) The delimiting character (/) may be any printable ASCII character. The delimiter is required both immediately preceding and immediately following the user's text.

Examples:

```
MOV     #'A, B          /Stores the ASCII representation
                        /of A into B
A:  ASCII  /HELLO/      /Loads the ASCII representation
                        /of H, E, L, L, O characters
    .                   /into consecutive memory bytes.
```

2.2.1.8.3 THE END ASSEMBLY DIRECTIVE

The end directive (END) serves a dual purpose:

1) indicates the physical and syntactical end of the source stream.
2) identifies the program's entry point, if there is an entry point.

The entry point information is transmitted to the absolute binary loader via the binary object program in the following manner. The assembler appends a jump instruction to the object program. If there is no entry point designated, then the appended instruction is a "HALT", thus causing the loader to halt. If the entry point was designated, then the appended instruction is a "MOV #ENTRY, PC", thus transferring execution to the users program. The form of the end directive is:

END expression

Syntactical constraints:

a) The end statement must be an individual source image.

b) The mnemonic END singularly occupies the operator field.

c) The operator field may be preceded by the label field and is terminated either by a blank or a tab character.

d) The expression must, obviously, contain no forward references and must conform to the constraints as set forth in section 2.2.1.5.

2.2.1.8.4 THE END-OF-TAPE ASSEMBLY DIRECTIVE

The EOT directive is used to indicate the physical end of the source character stream. The assembler reacts to an EOT by halting. The user continues the assembly by pressing the continue button.

2.2.1.8.5 THE REGISTER ASSEMBLY DIRECTIVE

The REG directive is of the form:

REG symbol, expression $0 \leq exp \leq 7$

The directive is used to identify a user defined register name.

2.2.1.8.6 THE MODULO ORIGIN ASSEMBLY DIRECTIVE

The MORG directive is used to set the assembly location counter equal to the next memory address value modulo N. The form of the directive is:

MORG N where N is a power of 2

2.2.2 CHARACTER SET

The subset of the ASCII character set which is acceptable to PAL-11 are only those characters whose ASCII encoded values lie between (inclusive) $\emptyset 4\emptyset_8$ and 137_8. The obvious exceptions are: blanks, tabs, carriage-returns, line-feeds, and form-feeds.

2.2.3 EXAMPLES

Examples have appeared throughout the preceding subsections.

2.3 OUTPUT

The assembler generates, upon request, an assembly listing, a symbol table listing, binary object data, and possibly, error messages on the command device (teletype).

2.3.1 OUTPUT FORMAT

2.3.1.1 ASSEMBLY LISTING

The user may request an assembly listing to be output onto the specified device via the command string interpreter (refer to section 3.4). The assembly listing is formatted as follows:

Each page is headed with a page number.
The line format of the assembly listing is:

```
ΔΔΔΔSS....S↵↓
EEEΔLLLLLLΔOOOOOOΔOOOOOOΔOOOOOO↵↓
```

E.....ERROR field
L.....LOCATION field
O.....OBJECT field
S.....SOURCE field

a) The error field is left justified and contains a maximum of three error codes (refer to section 3.6.1.1). Blanks will be printed in each unused error field print position.

b) The location field always contains the six digit octal representation of the user's location counter.

c) The object field contains the octal representation of the object data generated for the current source image. The printed digits are left justified within the field. Furthermore, the assembler interprets the object data during the formatting of the listing. For example:

Interpretation	Print Positions	Example (-1)
1 byte	OOOΔΔΔΔΔΔ	377
1 word	OOOOOOΔΔΔ	177777

Blanks will be printed in each unused object field position.

2.3.1.2 SYMBOL TABLE LISTING

The symbol table is unconditionally listed on the teletype at the end of the last assembly pass. The symbol table listing outputs all user defined symbols and their octal values at the end of the assembler's last pass.

2.3.1.3 BINARY OBJECT DATA (THE OBJECT PROGRAM)

The object program is output in binary or image mode. The program is formatted such that it is acceptable to the PDP-11 absolute binary loader (refer to document).

2.3.2 CHARACTER SET

The listing uses the printable ASCII character set.

2.3.3 EXAMPLE

A sample listing is "Not yet available".

2.4 ORGANIZATION

2.4.1 OPERATIONAL ORGANIZATION

The Command String Interpreter deduces the number of passes required to generate the desired amount of output. Using this pass count, the assembler performs the following functions in the indicated order:

Pass 1	Associates a value with each symbol. Generates an undefined symbol listing onto the teletype.
Pass 2	Generates the object program. Outputs the pass error count onto the teletype.
Pass 3	Generates the assembly listing. Generates the error listing onto the teletype. Generates the user's symbol table listing onto the teletype.

Note: If the object program is not desired then all functions of the third pass are performed on the second pass.

2.4.2 INTERNAL ORGANIZATION

Not yet available.

* S△L↓	Source input from the teletype paper tape reader
* B△L↓	Binary output goes to the teletype paper tape punch
* L△L↓	Assembly listing goes to the teletype printer

Erroneous device assignments cause the assembler to retype the current three character question.

3.5 OPERATION

Once the assembler is loaded (refer to section 3.1) the command string interpreter initiates communication with the user. After processing the command string the assembly proceeds without assistance (except for EOT processing). At the completion of pass two, the assembler types out the number$_8$ of errors:

"XXX△ERRORS".

3.6 ERROR RECOVERY

None. To restart the assembler, simply initiate execution at location 100_8.

3.6.1 INPUT ERRORS

Source input errors cause the loss of all characters already input in the current line.

3.6.1.1 ERROR CODES

Assembly errors are encoded for the assembly listing using single alphabetic characters. The error codes, their definitions, and their causes are as follows:

CODE	DEFINITION
A	Addressing error 1) An address within an instruction is incorrect.
B	Bounding error. Instructions or word data are being assembled at an odd address in memory.
D	Doubly defined symbol referenced 1) Reference was made to a symbol which already is doubly defined.

3.0 OPERATING PROCEDURE

3.1 LOADING PROCEDURE

The user manually loads the boot strap loader. Then the binary loader is booted into core. The binary loader then loads the absolute form of the assembler. The loader will automatically send the assembler into execution.

3.1.1 CONDITIONAL LOAD

None

3.2 SWITCH SETTINGS

None

3.3 START-UP PROCEDURES

When the assembler is ready to accept the command string, it will output onto the teletype:

↓↓* B Δ

and then wait for the user's response.

3.4 COMMAND LANGUAGE

The assembler's command string interpreter processes the user's I/O device assignments. The generalized form of the command string is:

* BΔL↓
* LΔL↓
* SΔL↓

The assembler outputs the three underlined characters and then waits for the user's response. The valid device assignments are:

L	Teletype
H	High Speed Paper Tape

The user's responses are interpreted in the following manner:

* SΔH↓	Source input from the high speed paper tape reader
* BΔ↓	No binary output
* LΔH↓	Assembly listing goes to the high speed paper tape punch

CODE	DEFINITION
I	Illegal character 1) A nonprinting ASCII character was detected. It was replaced by a question mark.
L	Line buffer overflow. All extra characters are ignored.
M	Multiple definition of a label 1) A label was encountered which was equivalent (in the first five characters) to a previously encountered label.
P	Phase error. 1) A label's definition or value varies from o one pass to another.
Q	Questionable syntax 1) This indicates missing arguments or that the instruction scan was not completed.
S	Symbol table overflow 1) When the quantity of user defined symbols exceeds the allocated space available in the user's symbol table the assembler outputs the current source line with the S error code, then the assembler returns to the command string interpreter to await the next command string to be input.
T	Truncation error 1) A number or quoted data generated more than 16 bits. Truncation is from left.
U	Undefined symbol 1) An undefined symbol was encountered during the evaluation of an expression. Relative to the expression, the undefined symbol is assigned a value of zero.

3.6.2 OPERATOR ERRORS

If the command string is incorrect, the assembler retypes *B, *L, or *S, whichever is applicable, and waits for the user to correctly retype the last command string.

3.6.3 SOFTWARE ERRORS

The PDP-11 assembler contains no error halts.

3.6.4 HARDWARE ERRORS

None.

3.6.5 MISCELLANEOUS ERRORS

All errored source lines are output as encountered, during the assembly listing pass. When any error is detected during the processing of a direct assignment of the form .=exp, the assembler unconditionally outputs the source line onto the teletype (normally during pass one). An error of this sort normally renders the object program useless.

4.0 INTERNAL ENVIRONMENT

4.1 TRADE OFFS

The major considerations are small core residency and ease of use.

4.2 SOFTWARE INTERFACES

Not yet available.

4.3 CONVENTIONS

Not yet available.

4.4 LANGUAGE

The assembler is written in the PDP-11 assembly language.

5.0 EXTERNAL ENVIRONMENT

5.1 EXECUTION SPEED

Not yet available.

5.2 USE

5.3 INTERFACE

6.0 DOCUMENTATION

6.1 MAJOR ASPECTS

Documentation shall consist of:

Program Listing
Program Specification

The program specification will include a complete description of the internal operations of the assembler.

6.2 CHECKOUT

The assembler will be thoroughly debugged by the implementor. In addition, a test program will be generated.

APPENDIX A
USA STANDARD ASCII CHARACTER CODES

Even Parity Bit	7-Bit Octal Code	Character	Remarks
0	000	NUL	Null, tape feed. Repeats on Model 37. Control shift P on Model 35.
1	001	SOH	Start of heading; also SOM, start of message. Control A.
1	002	STX	Start of text; also EOA, end of address. Control B.
0	003	ETX	End of text; also EOM, end of message. Control C.
1	004	EOT	End of transmission (END); shuts off TWX machines. Control D.
0	005	ENQ	Enquiry (ENQRY); also WRU, Control E.
0	006	ACK	Acknowledge; also RU, Control F.
1	007	BEL	Rings the bell. Control G.
1	010	BS	Backspace; also FEO, format effector. Backspaces some machines. Repeats on Model 37. Control H on Model 35.
0	011	HT	Horizontal tab. Control I on Model 35.
0	012	LF	Line feed or line space (NEW LINE); advances paper to next line. Repeats on Model 37. Duplicated by control J on Model 35.
1	013	VT	Vertical tab (VTAB). Control K on Model 35.
0	014	FF	Form feed to top of next page (PAGE). Control L.
1	015	CR	Carriage return to beginning of line. Control M on Model 35.
1	016	SO	Shift out; changes ribbon color to red. Control N.
0	017	SI	Shift in; changes ribbon color to black. Control O.
1	020	DLE	Data link escape. Control P (DC0).
0	021	DC1	Device control 1, turns transmitter (reader) on. Control Q (X ON).
0	022	DC2	Device control 2, turns punch or auxiliary on. Control R (TAPE, AUX ON).
1	023	DC3	Device control 3, turns transmitter (reader) off. Control S (X OFF).
0	024	DC4	Device control 4, turns punch or auxiliary off. Control T (~~TAPE~~, AUX OFF).
1	025	NAK	Negative acknowledge; also ERR, error. Control U.
1	026	SYN	Synchronous idle (SYNC). Control V.
0	027	ETB	End of transmission block; also LEM, logical end of medium. Control W.
0	030	CAN	Cancel (CANCL). Control X.
1	031	EM	End of medium. Control Y.
1	032	SUB	Substitute. Control Z.
0	033	ESC	Escape, prefix.
1	034	FS	File separator. Control shift L on Model 35.
0	035	GS	Group separator. Control shift M on Model 35.

Even Parity Bit	7-Bit Octal Code	Character	Remarks
0	036	RS	Record separator. Control shift N on Model 35.
1	037	US	Unit separator. Control shift O on Model 35.
1	040	SP	Space.
0	041	!	
0	042	"	
1	043	#	
0	044	$	
1	045	%	
1	046	&	
0	047	'	Accent acute or apostrophe.
0	050	(	
1	051	)	
1	052	*	Repeats on Model 37.
0	053	+	
1	054	,	
0	055	-	Repeats on Model 37.
0	056	.	Repeats on Model 37.
1	057	/	
0	060	0	
1	061	1	
1	062	2	
0	063	3	
1	064	4	
0	065	5	
0	066	6	
1	067	7	
1	070	8	
0	071	9	
0	072	:	
1	073	;	
0	074	<	
1	075	=	Repeats on Model 37.
1	076	>	
0	077	?	
1	100	@	
0	101	A	
0	102	B	

E Parity Bit	7-Bit Octal Code	Character	Remarks
1	103	C	
0	104	D	
1	105	E	
1	106	F	
0	107	G	
0	110	H	
1	111	I	
1	112	J	
0	113	K	
1	114	L	
0	115	M	
0	116	N	
1	117	O	
0	120	P	
1	121	Q	
[redacted]	122	R	
0	123	S	
1	124	T	
0	125	U	
0	126	V	
1	127	W	
1	130	X	Repeats on Model 37.
0	131	Y	
0	132	Z	
1	133	[	Shift K on Model 35.
0	134	\	Shift L on Model 35.
1	135	]	Shift M on Model 35.
1	136	↑	
0	137	←	Repeats on Model 37.
0	140	`	Accent grave.
1	141	a	
1	142	b	
0	143	c	
[redacted]	144	d	
0	145	e	
0	146	f	
1	147	g	

Even Parity Bit	7-Bit Octal Code	Character	Remarks
1	150	h	
0	151	i	
0	152	j	
1	153	k	
0	154	l	
1	155	m	
1	156	n	
0	157	o	
1	160	p	
0	161	q	
0	162	r	
1	163	s	
0	164	t	
1	165	u	
1	166	v	
0	167	w	
0	170	x	Repeats on Model 37.
1	171	y	
1	172	z	
0	173	{	
1	174	|	
0	175	}	This code generated by ALT MODE on Model 35.
0	176	~	This code generated by ESC key (if present) on Model 35
1	177	DEL	Delete, rub out. Repeats on Model 37.

Keys That Generate No Codes

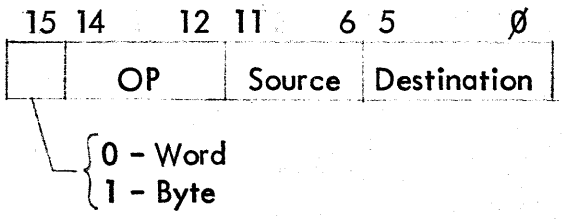
REPT	Model 35 only: causes any other key that is struck to repeat continuously until REPT is released.
PAPER ADVANCE	Model 37 local line feed.
LOCAL RETURN	Model 37 local carriage return.
LOC LF	Model 35 local line feed.
LOC CR	Model 35 local carriage return.
INTERRUPT, BREAK	Opens the line (machine sends a continuous string of null characters).
PROCEED, BRK RLS	Break release (not applicable).
HERE IS	Transmits predetermined 21-character message.

APPENDIX B

PDP-11 Machine Codes

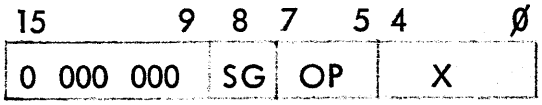
I. Binary Group

OP	Mnemonic
1	MOV
2	ADD
3	SUB
4	CMP
5	AND
6	----
7	----



II. Operate Group

SG	OP	X	Mnemonic
Ø	Ø	Ø	HALT
Ø	Ø	1	WAIT



Ø	Ø	2	RTI
Ø	Ø	3-37	Reserved
Ø	1-3	Ø-37	Reserved

Ø 4 | Ø | D | REG | RTS

Register
Deferred

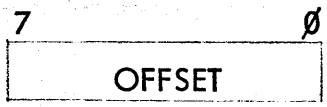
Ø	4	2Ø-37	Reserved

Ø 5 | | N | Z | V | C |

Ø - clear
1 - set

Micro-programmed operator on condition codes

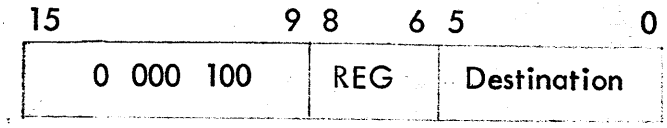
Ø	6-7	Ø-37	Reserved



BR ±128 words unconditional.

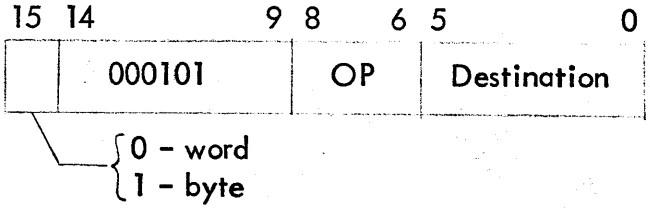
III. Subroutine Call

JSR REG,ADR

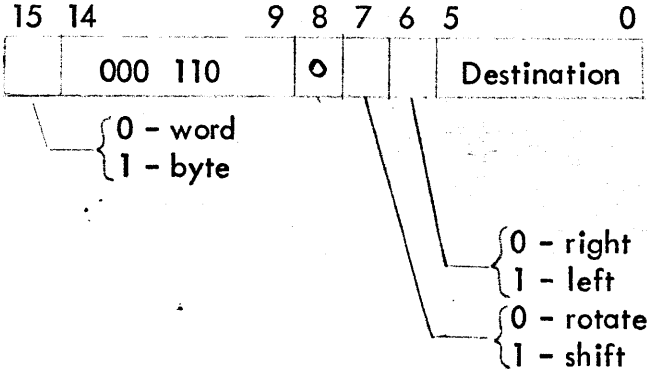


IV. Unary Group

OP	Mnemonic
ø	CLR
1	COM
2	INC
3	DEC
4	NEG
5	ADC
6	DIC
7	TST

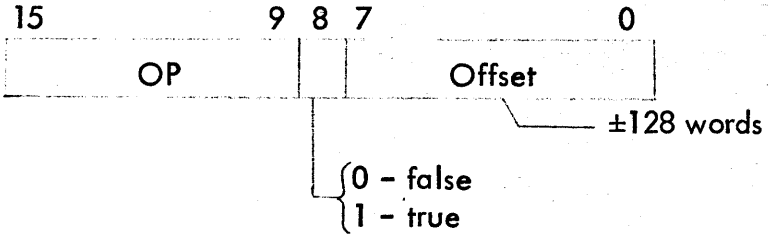


V. Rotate/Shift



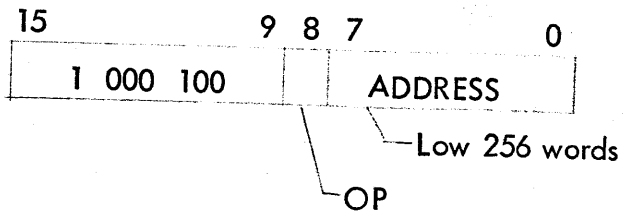
VI. Conditional Branches

OP	TF	Mnemonic
001	1	BEQ
001	0	BNE
002	1	BLT
002	0	BGE
003	1	BLE
003	0	BGT
103	1	BCS
102	1	BVS
102	0	BVC
101	1	BZS
101	0	BZC
100	1	BNS
100	0	BNC



VII. Execute

OP	Mnemonic
Ø	EXC
1	Reserved



Distribution List for PAL-11

R. Merrill
G. Thissell
L. McGowan
D. Barlow
D.Sosville
J. Hittell
R. Weaver
B. Becket
J. Cohen
G. Arnold
K. Nelson
R. Pyle
B. Young
C. Luartes
G. Butler
J. O'Loughlin
N. Mazzarese
C. Learoyd
H. McFarland
J. Davis
J. Jones
D. Dubay
G. Fligs
H. Godfrey
A. Kotok
I. Morris
G. Saviers
D. Zereski
J. Bell
B. Delagi
K. Hedberg
D. Murphy