Applications Program Series

Waterloo SCRIPT REFERENCE MANUAL

University of Waterloo

Waterloo, Ontario, Canada

January 13, 11378

PREFACE

The quality and accuracy of a document depend greatly on the ease with which revisions can be made to the document. This statement is particularly true of technical documents (into which class the present manual falls), which should always accurately reflect the status of the things they describe.

It is natural that computer software solutions to the problems of document production should be devised. CTSS's "runoff", MULTIC's "runoff" and CMS's "SCRIPT" represent similar such solutions.

This manual describes "SCRIPT", an outgrowth of NSCRIPT intended for use under TSO (currently still available from the SHARE programme library) and of SCRIPT intended for use under CMS on a System/360 model 67 running under CP/67. SCRIPT running under 360/OS/TSO or 370/VM/CMS has the same outward appearance as it did when running under CMS. Its set of command words encompasses most of those belonging to SCRIPT, MULTIC's "runoff" and TSO's FORMAT. In most cases they perform identical functions and have the same symbolic notation.

TABLE OF CONTENTS

SCRIPT Control Words by Function

* INDICATES CONTROL WORD CAUSES A BREAK

HYPHENATION CONTROL

INDEX

LINE SPACING

MACROS

MARGIN MODIFICATION

MULTIPLE COLUMNS

OVERLAYING TEXT

PAGE DIMENSIONS

PAGE NUMBERING

PAGE SPACING

TABLES OF CONTENTS

TERMINAL INPUT AND OUTPUT

TRANSLATION OF CHARACTERS

UNDERSCORING AND UPPERCASE

WIDOW CONTROL

Purpose:

SCRIPT is used to format an input file containing text and control lines. The file may contain fixed or variable length records or WYLBUR Edit-format records. Formatting is specified by control lines (lines that begin with a control indicator, normally a period, followed by a two letter control word). The appendix titled "Sample Input to SCRIPT" contains an unformatted listing of a sample input file.

The "SCRIPT" command causes the specified file to be edited and formatted, directed by control lines contained within the input file. Each line read from the input file is inspected for a first character equal to the control identifier, normally a period, which identifies a control line. Control lines are not output, but are interpreted to specify the format of the output. The control word may be specified in either upper or lower case (or a mixture of the two) and must be separated from its arguments (if any) by one or more spaces. Control lines may appear anywhere in the file, and have effect on all output produced after their appearance.

Output can be directed to the user's terminal, the offline printer by submitting a batch job, or to a permanent file and then printed on the offline printer using a utility program. Certain differences in the processing occur depending on where the output is directed. These differences are described below under individual headings.

Options:

The options control formatting of output and the mechanical processes of the output device. Options may be delimited by blanks and keyword operands may be delimited on the left with an equal sign or the operand surrounded by parentheses. Capital letters indicate minimum shortforms. Underscores indicate default values.

ADjust / NOADjust / ADjust=<0|n>
: Causes all the output to be printed up to fifty columns from the left print margin. ADJUST by itself defaults to thirty columns, but ADJUST= may specify from zero to fifty columns.

CEntre / NOCEntre / CEntre=<0|n>
CENTEr / NOCENTEr / CENTEr=<0|n>
: Alternate forms of ADJUST, for compatability with former versions of SCRIPT.

COntinue / NOCOntinue / COntinue=<0|n>
: Specifies the number of errors which are to be allowed before processing is terminated. The value of "n" may range from 0 to 32767; NOCO is the same as CO=0, and CO is the same as CO=32767.

DEBUG / NODEBUG
SPIE / NOSPIE
: This is a debugging tool for the person maintaining the SCRIPT program. It prevents a program interrupt exit from being set, thus preventing a SCRW999T diagnostic from occurring.

FIle / NOFIle
: When used in combination with the TERMINAL option, this causes output to be written to the usual offline file but in a format suitable for an online terminal.

FOrmatted / UNformatted
: The specified file is printed along with all control lines. No other processing takes place.

FNSize=<200|n>
: Specifies the maximum number of output lines (from 1 to 32767) that may be outstanding in a Keep or Footnote at any one time.

HSFSover=<9|n>
: Specifies the maximum number of top and bottom titles that may be defined. See the .TT control word for a description of how these titles are shared. The numeric operand may range from 2 to 19.

LINenumber=<0|n> / LEGalnumber=<0|n>
LIN= causes the current output line number within the page to be displayed if the output line is non-blank and that portion of the output line starting at column "n" (ranging from 1 to 125) is otherwise blank. LEG= numbers only non-blank lines in the text area. LIN= and LEG= are mutually exclusive.

LOCAL / GLOBAL
Causes the parameter variables "&0", "&1", etc. to be entered in a dictionary local to the file being called or to be entered in the global dictionary.

MArk / NOMArk
Marks the beginning of each line of the original input by underscoring the first character of each record.

NUmber / NONUmber / NUmber=<0|n>
Displays the number of the record in the input file being processed when the current output line was printed. The value of "n" (from 1 to 100) defines the starting column in the output line where this information is to appear, if those columns would otherwise be blank. NONU is the same as NU=0. NU is the same as NU=1 with an ADjust of 30.

OFfline or PRinter
Causes the output to be edited and formatted for the off-line printer.

ONline or TErminal
Causes output lines to be formatted for an online terminal. This option is mutually exclusive of the PRinter option.

PAge=<1|m<:n>>
Input is processed but output is not produced until the start of page "m". If a second number "n" is specified, processing and output will terminate after page "n" has been produced. The PAGE= option may be useful for restart purposes (i.e. PAGE=5) or for checking single pages of output (i.e. PA=5:5).

PASses=<1|n>
Causes SCRIPT to make "n-1" complete passes through the input file, performing all indicated formatting but not producing any output. A last pass is then made, during which output is produced. The value of "n" may range from 1 to 10. This option is useful for processing a file which uses reference variables defined later in the input.

PRinter or OFfline
Causes the output to be edited and formatted for the off-line printer. It is recommended that PRinter be used when proof-reading the output.

PROFile / NOPROFile
Causes an imbed (.IM) of a file called PROFILE automatically at the start of each pass.

TWOPass or 2Pass / ONEPass or 1Pass
The TWOPASS option is a shortform for PASSES=2. The ONEPASS option is a shortform for PASSES=1.

QUiet / NOQUiet
If TErminal is in effect, QUiet causes the SCRIPT version identification line to be suppressed.

RMSize=<200|n>
Specifies the maximum number of input records (from 1 to 32767) that may be included in any single remote (see .RM) definition.

SCREen / NOSCREen
Specifies that the output device is a SCREEN or CRT device. It is effective only for TErminal output. SCREEN is set as the default if the output device is known to be a CRT. The effect is to print overstruck output on one or more consecutive lines instead of using "backspace and underscore".

SEQColumn=<240|n>
Specifies the first of eight contiguous columns (from 1 to 240) where the input record sequence number may be found for variable-length input files. SEQ=0 means that no sequence numbers are present. If "n" is greater than the length of a record, then the last eight columns are presumed to contain the sequence number for that record.

SIX / EIGHT
The SIX option defines the initial Page Length for six lines per inch on an eleven inch page and line three set to the first normal print line on the page. The EIGHT option defines the initial Page Length for eight lines per inch on an eleven inch page and line four set to the first normal print line on the page.

SRLength=<150|n>
Specifies the maximum length of character-string operands for reference variables; "n" may range from 4 to 240.

STATistics / NOSTATistics
Causes various and sundry statistics information to be printed on the SYSTERM data set.

STop / NOSTop
Causes a pause at the bottom of each page when producing online output. This pause allows the user to insert the next page. The pause is terminated by a carriage return.

TABLeft=<0|1>
Controls the operation of a Left Tab. A "0" value makes Left Tabs work like a typewriter, with the character following a tab being placed in the column specified. A "1" value makes the character following a Left Tab go immediately after the column specified. See the Tab control word (.TB).

TErminal or ONline
Causes output lines to be formatted for an online terminal. This option is mutually exclusive of the PRinter option.

UPcase / NOUPcase
TRanslate / NOTRanslate
Causes character translation using a translate table that contains as default lower to upper case mapping. Other types of translation may be specified with the .TR control word.

UPPer / NOUPPer
The UPPER option causes all SET REFERENCE symbols to be converted to upper case before symbol table lookups.

WAit / NOwait
Causes output to begin immediately without waiting for the first page to be positioned by the user. This option has no effect when producing off-line output.

&name=string OR +name=string
At the beginning of the first pass, a ".SR name=string;" control line will be executed for each such option in the options list. See the ".SR" control word for operand format rules. This is useful for setting switch values externally to the SCRIPT input.

The following options maybe used to set initial and default values for the corresponding control words:

Parameter	Minimum	Maximum
ADjust=<0|n>	0	50
BMargin=<6|n>	0	33
DArk=<1|n>	1	10
FMargin=<1|n>	0	33
HMargin=<1|n>	0	33
LLength=<60|n>	10	144
PLength=<66|n>	10	176
TMargin=<6|n>	0	33

Error Messages:

All messages, whether error or informative, are written to the DDNAME SYSTERM. An error number suffix of 'W' indicates a warning only, with a four return code. A suffix of 'E' indicates an error with return code eight. A suffix of 'S' indicates a severe error and a 'T' indicates a terminal error with return codes twelve and sixteen respectively.

For error messages produced by incorrect control lines, the following message is printed:

ERROR OCCURRED AT LINE nnnnnnnn OF FILE xxxxxxxx:

plus zero or more file traceback messages:

CALLED FROM LINE nnnnnnnn OF FILE xxxxxxxx:

along with the line in error and, if possible, an asterisk "*" under the incorrect operand.

SCRW001E UNRECOGNIZABLE CODES IN PARM FIELD.
An invalid option has been specified in the options list.

SCRW002T OUTPUT FILE (SYSPRINT) CANNOT BE OPENED.
The file for the formatted output cannot be opened. Processing is terminated.

SCRW003E OMITTED CONTROL WORD PARAMETER.
No argument was provided for a control word that requires an argument.

SCRW004E UNDEFINED CONTROL WORD.
An unrecognizable control card has been encountered in the input file(1).

SCRW005E INVALID CONTROL WORD PARAMETER VALUE.
The argument portion of a control line is invalid. This could possibly be caused by a character being used where only a numeric argument is acceptable.

SCRW006E OMITTED CONTROL WORD TERMINATION.
A control word section was started, but never ended. For example a footnote was started with ".FN BEGIN" but an ".FN END" termination was omitted.

---

(1) This error is commonly caused by presenting an input line, not intended to be a control line, which begins with a control indicator, normally a period.

SCRW007E TOO MANY REFERENCE NAMES.
This message indicates that no more working storage is available to create another symbolic reference name. Increase the program region or decrease the number of variable symbols in the input.

SCRW008E REFERENCE NAME OR ARGUMENT TOO LONG.
A reference name being assigned a value or a reference name being substituted or the character argument being assigned to a reference name exceeds its maximum allowable length. (See also the SRLENGTH= option.)

SCRW009E LOOP IN REFERENCE NAME SUBSTITUTION.
While processing an input text line under ".SU□ON" or a ".SE" control line, each time a reference name was substituted its value contained another reference name. This process was terminated after 100 iterations.

SCRW010T OUTPUT LINE HAS ZERO WIDTH.
The available line length to format text has disappeared. Examine Line or Column Lengths (.LL or .CL) and Indents and Offsets (.IN and .OF).

SCRW011E SAVE/RESTORE STATUS STACK ERROR.
A Save Status ".SA" request would have exceeded the maximum push down stack level, normally nine, or a Restore Status ".RE" request was issued when the push down stack was empty.

SCRW012E ILLEGAL PLACEMENT OF CONTROL WORD.
A valid control word was encountered where that control word is not allowed. Example: ".FN□begin" within a footnote or a Page Eject ".PA" within a conditional keep.

SCRW013E CONTROL WORD PARAMETER TOO BIG.
The argument portion of a control line is too large to be valid as an argument. A smaller numeric argument must be specified.

SCRW014E CONTROL WORD PARAMETER TOO SMALL.
The argument portion of a control line is too small to be valid as an argument. A larger numeric argument must be specified.

SCRW015E MISMATCHED DELIMITER ON STRING.
A character string starting with a delimiter, normally a quote, was not terminated with the same delimiter.

SCRW016E HANGING INDENT INVALID.
The value of Hanging Indent is currently invalid. This may be because the Line Length has decreased or the Indent has increased since the Hanging Indent was defined.

SCRW017E MACRO/REMOTE NAME UNDEFINED.
A reference has been made to a macro or remote that does not exist. Either the name or number is incorrect or the call count of the remote is exhausted and the remote has been automatically deleted.

SCRW018E INSUFFICIENT STORAGE FOR MACRO/REMOTE.
No storage is available to store away the definition of the current macro or remote. This may be solved by deleting unused macros and remotes or increasing the program region size.

SCRW019E TOO MANY RECORDS IN ONE MACRO/REMOTE.
More than the maximum records, as defined by the RMSIZE= parameter, are now in one macro or remote definition. Probable cause of this diagnostic is a failure to close a macro with a ".dm□end" or a remote with a ".rm".

SCRW020E EXCESSIVE NUMBER OF LINK ELEMENTS.
An insufficient number of internal control blocks were available for processing the current input text line. The problem can usually be circumvented by breaking the problem line into two lines. Installation personnel should also be notified.

SCRW021E BACKSPACE BEFORE COLUMN ONE.
The current input text line has uncanonicalized backspaces that attempt to backspace beyond the start of the record. These backspaces will be ignored.

SCRW022E .IF NESTING TOO DEEP.
A .IF control word was the object of a Then or Else to a nesting level greater than ten. Restructure the logic of the sequence.

SCRW023E .TH OR .EL NOT AFTER .IF.
A Then (.TH) or Else (.EL) control word did not follow a .IF. A Then is only valid immediately following an If and an Else is only valid after a Then or an If.

SCRW024E WORKING STORAGE EXHAUSTED.
No storage is available for a new item. Either increase program region or make less use of such features as Floating Keep, Footnotes or Labels.

SCRW025E DUPLICATE LABEL.
An identical label exists at a different input record within the current input file. User labels must be unique within each input file.

SCRW026E FORWARD .GO TARGET NOT FOUND.
The target of a Goto statement was not known at the time the .GO control word was interpreted. The rest of the current input file was processed without finding the target.

SCRW027E BACKWARD .GO TARGET NOT FOUND.
The target of a Goto statement was already known at the time the .GO control word was interpreted. However, possibly because of variable substitution, the Label could not be found within the record which formerly contained the Label.

SCRW028E INCORRECT PLACEMENT OF .DO OR .EN.
A Do (.DO) control word did not follow a Then (.TH) or an Else (.EL). Alternatively, an End (.EN) control word was encountered with no preceeding Do (.DO).

SCRW029W POSITIONAL PARAMETER ASSUMED.
An argument containing an equal sign failed to evaluate without error as a Keyword Parameter. This warning message indicates that the argument was then taken to be a Positional Parameter. Delimit the string with a character such as a quote.

SCRW030E HYPHEN DELETE WORD NOT FOUND.
An attempt was made to delete a word from the Hyphenation Exception Dictionary. That word, including matching break points, was not found.

SCRW031E INVALID WORD TO HYPHENATE.
An Exception word contained characters other than alphabetics. This may also be caused by a hyphen at the start or end of an Exception word.

SCRW032E TOO MANY HYPHEN BREAK POINTS.
An Exception word contained more than seven hyphens or break points. This word is considered invalid.

SCRW033E HYPHEN WORD TOO LONG.
An Exception word contained more than thirty-five characters or had one syllable containing more than fourteen characters. This word is considered invalid.

SCRW034E TOO MANY QUEUED LINES FOR .xx.
More than the maximum number of lines, as defined by the FNSIZE= option, are now queued for a ".xx" control word. Probable cause of this diagnostic is a failure to terminate with a ".xx□end" control word. ".xx" may be ".CC", ".CP", ".FB", ".FK", ".FN" or ".HN".

SCRW035E CONTROL WORD PARAMETER SHOULD BE NUMERIC.
The operand of a control word could not be evaluated as a numeric or as a numeric expression. Correct the operand with a valid numeric.

SCRW037E HEADNOTE TOO LONG FOR PAGE.
A Headnote has been defined that is too long to appear in its entirety on the current page. The Headnote is not printed.

SCRW038E INVALID SUBSCRIPT.
The subscript expression of a reference variable name could not be evaluated or the subscript was outside of the range of -32K to +32K.

SCRW039E INVALID SUBSTRING.
The first or second substring expression of a reference variable name could not be evaluated or the substring value was outside of the range of 1 to 255.

SCRW040E INVALID CONDITION OPERATOR IN .IF.
The condition operator of an .IF control word has not been recognized. See the command description for a list of valid operators.

SCRW041E .HM + .HS GREATER THAN .TM.
The Top Margin value cannot be less than the sum of the Heading Margin plus the Heading Space.

SCRW042E .FM + .FS GREATER THAN .BM.
The Bottom Margin value cannot be less than the sum of the Footing Margin plus the Footing Space.

SCRW043E ILLEGAL CONTROL WORD WITHIN KEEP OR FOOTNOTE.
A valid control word was placed where it is not allowed. Example: A multiple column definition within a footnote.

SCRW044W GENERATED INPUT LINE TOO LONG.
An input line that has been modified by SCRIPT exceeds 240 characters in length. Example: A "line" operand of UNDERSCORE (.US) is too long.

SCRW045E .TM + .BM TOO BIG FOR .PL.
The Page Length setting is too small for the current definition of Top and Bottom Margins.

SCRW050? .ER <line operand>.
A user error control word (.ER) has been used. The severity "?" and the message "<line operand>" is taken from the user data.

SCRW051E FILE NOT FOUND.
Cannot find a reference to the file specified in an ".ap" or an ".im" control statement.

SCRW052E FILE NOT PARTITIONED, MEMBER NAME IGNORED.
An ".ap" or ".im" control statement included a member reference. The filename specified is not partitioned so the member specification is ignored.

SCRW053E BUFFER STORAGE UNAVAILABLE, FILE SKIPPED.
No memory was available to buffer the file specified in an ".ap" or ".im" control statement. The file will be skipped. Either reduce the blocksizes of input files or increase the region size.

SCRW054E RECORD FORMAT NOT SUPPORTED, FILE SKIPPED.
The input file is not RECFM=F or RECFM=V or a WYLBUR EDIT-format file. The IMBED or APPEND control statement is skipped.

SCRW055E INVALID INPUT RECORD, FILE SKIPPED.
The input file contains a bad record. The rest of the file will be skipped.

SCRW056E I/O ERROR.
An I/O error occurred while reading an input file. Information about the error follows the diagnostic.

SCRW057I SCRIPT -- SYSTERM FAILED TO OPEN.
The SCRIPT error file could not be opened successfully. Processing continues with error and informational messages disabled. The final return code from SCRIPT is the only indication of success or failure.

SCRW058E UNABLE TO OPEN WORK FILE.
An output file for PUT WORKFILE (.PU) could not be opened successfully. The command request is skipped.

SCRW998T A LOGIC ERROR HAS BEEN DETECTED.
This message indicates an error in the SCRIPT command module. The appearance of this message should be reported to Installation personnel and a copy of the SYSIN file should be saved for their use in diagnosing the difficulty. Usually, the user can bypass the problem by changing the input file. This error cannot occur if DEBUG or NOSPIE has been specified as an option. A Program Interrupt will result instead.

Code Number		Meaning
1001	---	Negative spacing error.
1002	---	Queueing incore print line error.
1003	---	Footnote print error.
1005	---	Freecell error.
1006	---	Remote imbed call error.
1007	---	Bad Floating/Conditional Keep.
1010	---	Negative Online spacing error.
1012	---	Missing Output Overlay data.
1013	---	Record too small to print.
1015	---	Multi-column print mismatch.
1016	---	Multi-column line save error.
1017	---	Multi-column line count mismatch.

SCRW999T A PROGRAM CHECK HAS OCCURRED.

This message indicates an execution error in the SCRIPT command module. The appearance of this message should be reported to Installation personnel and a copy of the SYSIN file should be saved for their use in diagnosing the difficulty. Usually, the user can bypass the problem by changing the input file. This error cannot occur if DEBUG or NOSPIE has been specified as an option; instead, a system abend will result.

CMS Online Procedures:

The CMS command format is as follows:

SCript	filename <(options ... >

where the CMS file "filename" must have a filetype of SCRIPT. A "filename" of question mark "?" will type a file of help documentation from the system disk.

When output is to be produced on the user's terminal:

(1) Unless the QUIET option was specified, the immediate response to issuing the SCRIPT command is the following line:

UOW SCRIPT - VERSION(3.2) 78JAN13

(2) Unless the NOWAIT option was specified, the next response to issuing the SCRIPT command is the following line[2]:

Load paper; hit return:

The user should remove the ordinary terminal paper and insert the paper on which the output is to appear. The paper should be positioned in such a way that the top edge of the paper is just below the typeball runner. Output will be started by typing a carriage return[3].

(3) If the STOP option was supplied, output will cease at the end of each page. The user should insert the next piece of paper, aligning it as described in (2), and type a carriage return.

(4) At the conclusion of the last page of output, the carriage will be spaced several lines into the following page before the READY message is typed[¼]. The user may re-insert the terminal paper at this time.

---

(2) If the PASS= option is supplied, this message is not produced until the start of the last (output) pass.
(3) Note that the paper will be ready for typing on the top line following the carriage return.
(4) If STOP is specified, the carriage will pause at the end of the last page.

Under CMS, a user should be aware of the eight character limit per "token" in a CMS command line. The recommended method of specifying a keyword parameter is with a separating blank. So to specify an initial Line Length the user should specify "LL 66".

SCRIPT Reference Variables may be defined in the options by entering a plus sign '+' or an ampersand '&' immediately before the name of the variable. The value of the Reference Variable follows immediately as the next option. If the next option is a single "?" then the value of the Reference Variable will be prompted at the terminal in Upper and Lower case.

A special set of Reference Variables may be entered by specifying "SYSVAR (" in the options. Following this, in pairs, are the SYSVAR name qualifier and the value. The name qualifier may be from one to four characters long and multiple pairs may be entered, terminated by end of parm or a right parenthesis. As for other Reference Variables, the value for a SYSVAR variable may be entered as a single "?" which causes the value to be prompted at the terminal.

SCRIPT FN (QUIET +X 99 +Y ABC SYSVAR (1 ONE EXEC 3270

Produces as the parm:

QUIET +X=99 +Y=ABC +SYSVAR1=ONE +SYSVAREXEC=3270 TERM NOFILE

If a file "PROFILE SCRIPT" exists, then the default setting of the NOPROFILE option will be changed to PROFILE.

Under CMS, several parameters may be specified to direct the formatted output. These are mutually exclusive, and if more than one is specified then the last encountered will be in effect.

ONline | OFfline
: These options specify of the output target is an online terminal or an offline device.

FIle | NOFIle
: These options specify if the target is a disk file or not. Such a file may be typed or printed depending on the ONLINE or OFFLINE option.

DIsk (or OFFLINE FILE)
: The formatted output will be directed to the input disk or the "A1" disk if the input disk is not a R/W extension of the "A" disk, with a filename of "filename LISTING". By default this file has a variable record format and ASA carriage control.

MEmo (or ONLINE FILE)
The formatted output is directed to a file with a filename of "filename MEMO" in terminal format so that it may be TYPEd later at the user's terminal.

PRt / PRinter (or OFFLINE NOFILE)
The formatted output is spooled to the user's printer.

NOPRt / NOPRint
The formatted output is thrown away to a DUMMY file. Errors, however, are still displayed at the terminal.

TErminal (or ONLINE NOFILE)
The formatted output is displayed at the user's terminal. This is the default under CMS.

TSO Online Procedures:

Any help from a TSO installation for this section will be appreciated.

Online Terminal Support

SCRIPT has enhanced support for Online ASCII terminals. In particular this support would be most useful for hardcopy terminals using the 'Diablo' print mechanism. For such terminals, if the user includes terminal escape codes as part of the document, then these escape codes will be retained within the text without confusing the justification algorithm.

For terminals supporting a Negative Line Feed it will be used to support the SPace zero or Don't Count control words. If the user makes use of the Null Character (X'00'), TN superscripts and box drawing characters or ALA subscripts, these will be altered to the appropriate escapes and printable characters for the terminal. For multiple column output SCRIPT uses the absolute Horizontal Tabbing facility of the terminal. And lastly, SCRIPT will take advantage of the variable Horizontal Motion Index feature to display output from 4 to 18 characters per inch and will generate proportional interword spacing codes.

NOTE: This version of SCRIPT has not solved the problem of Output Overlay and the vertical sides of Boxes. These characters may be incorrectly placed in formatted text because of the variable width of the interword blanks.

But all of this comes at a price. New positional and keyword parameters must be specified in order to make use of the facilities of the terminal.

CPInch=<10|n>
: This parameter specifies the horizontal number of characters per inch for ONline output. For this to have any effect, the Horizontal Motion Index (HMI) must be defined for the terminal. The default CPI is 10, but may range from 4 to 18.

X1620
: This positional parameter defines the output terminal to be a XEROX 1620 diablo terminal. SCRIPT makes use of the HMI, Negative Line Feed, Superscript, Subscript and absolute Tabbing features of the terminal.

MULTI3
: This positional parameter defines the output terminal to be an Ahearn and Soper Multiwriter III terminal. It is currently defined identical to the X1620. The terminal however will do backward typing and horizontal tabbing to optimize the speed of output.

QUME

This positional parameter defines the output terminal to be a QUME terminal. Currently it is defined with none of the attributes that SCRIPT requires for enhanced output. The function codes are known to be different from the X1620 but what they are exactly is not known for this version of SCRIPT.

ASISterm

This positional parameter defines an output terminal that knows nothing about escape characters and special characters such as Superscripts and Subscripts.

MYTERM

This positional parameter defines the output terminal to an installation dependent terminal. By default this parameter defines a 2741 type of terminal that cannot do any of the features described above. Special characters are simulated on such a terminal however.

Offline Printer Support:

SCRIPT can also be executed as a background job or an over-the-counter batch job so that it will print directly on the offline printer. The appendix titled "Offline Control Statements" contains the JCL for submitting such a job.

Keyword parameters may be specified to define the physical characteristics of the output printer.

FFCHannel=<1|n>
The Form Feed Channel option specifies the printer channel from 1 to 9 that SCRIPT will use to align output below a page perforation.

FFTOp=<3|n>
The Form Feed Top option specifies the line number on the page that corresponds to the FFCHANNEL= skip on an output page. The value of this parameter may vary from line 1 to 175.

When output is to be formatted for the offline printer:

(1) The first page of output will be positioned at a page crease (perforation) before normal first page processing is initiated if a title or text is to be printed above the Form Feed Top line (FFTOP=) of the page.

(2) If a title is to be printed on the first page, a Top Title (.TT) control word must preceed the first text line in the input stream.

(3) All pages of an OFFLINE file may print from the first line just below the page perforation for the full Page Length (.PL). Therefore the .PL value, the FFCHANNEL= and FFTOP= parameters should agree with the physical setup of the printer paper.

(4) Blank output lines preceeding the FCHANNEL= line on each output page are not printed. Thus SCRIPT output to an IBM 3800 printer will be correct if the user does not put titles or text in the top and bottom one half inch of an output page.

Creation of Input for SCRIPT

All input must be in the form of fixed or variable length records or WYLBUR Edit-format records. A fixed file is assumed to be sequenced in the last eight columns of each record, an assumption that is dropped with the first record per file or member that doesn't have only numerics in those columns. A file with a variable format is also assumed to be sequenced in a specifiable eight columns of each record, normally the last eight. Input can have a maximum length of 240 characters‘½’. SCRIPT will format these input lines into lines of length specified by ".ll", normally 60‘¾’.

Formatting of a line with characters from a following line can be prevented by creating what is called a "break". A break is caused explicitly by use of leading blanks (one is sufficient), a tab, or the break command word (.br). Implicit breaks are caused by any of the commands that cause normal line spacing to be interrupted. Breaking capabilities for each command word can be found in its complete description in the remainder of this manual.

Input lines containing backspaces may be canonicalized or uncanonicalized. This means that if "<" represents the backspace character then "ABC<<<___" and "A<_B<_C<_" will produce identical results. The correct number of character positions will be typed per line if online and will overprint properly if offline. The UNDERSCORE control words, ".UC", ".UD", ".US", may alleviate some problems encountered when editing backspaces with a text editor.

An input line may contain an Escape Character (X'27') followed by a single Function Character. These two characters do not count toward filling the length of the line. If "!" represents the Escape Character, "!D" on your terminal moves the paper down one half line and "!U" moves the paper up one half line, then "X!D2!U" could be used to create the symbols for "X squared". For offline output to a printer, the Escape Character and the following Function Character are lost.

Escape Characters must be entered in canonicalized order. If "<" represents a backspace, then to enter "A" sub "i+1" to the "n-2" power requires:

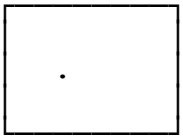
```
A!Ui<!D!Dn!U!U+<!D!D-!U!U1<!D!D2!U
```

The following, which may be more natural, is not canonicalized and will not work:

```
A!Ui+1<<<!D!Dn-2!U
```

---

(5) It is possible to input more than 130 characters per line if backspaces and underscores are used.
(6) If the sequence character, backspace, character is found, the backspace/character will not be counted.



The line operand of the NULL control word will be treated as input, either text or control word.

The ".□□" control word is specified by the control word indicator, immediately followed by one or more blanks. Its function is to treat the "line" operand as SCRIPT input. The "line" operand starts with the first non-blank character and may be text or another control word.

This control word has two uses. The first allows text followed by control words to be entered on the same input line. The second allows input lines to be indented for readability.

Defaults:

This control word will not cause a break. However, a control word specified in the "line" operand may cause a break to occur. If the "line" operand is omitted, a blank input line is assumed.

Notes:

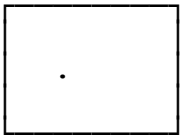
(1) While usually specified with a "blank" control word, the implementation of this control word allows ".NL" (NulL) to be specified. The result is the same.

Examples:

(1) . this is some text;.sp;and this is more.

is the same as:

```
this is some text
.sp
and this is more.
```



(2)
```
.dm TEST begin
.if '&1' = '' .th .do begin
.                     .fo on
.                     .sp
.                     .cp 5
.                 .do end
.             .el &1
.dm TEST end
```

Done this way for readability.

The ADJUST control word causes all output to be moved to the right of the physical left print margin.

.AD	<EVEN|ODD> <0|n|+n|-n>

The .AD control word causes the logical left margin of the formatted printout to be moved "n" spaces to the right of the actual left margin of the output device (printer or terminal). This adjustment value remains in effect for all subsequent lines until altered by another .AD control word. An operand of the form "+n" adds the value "n" to the current adjust value. An operand of the form "-n" subtracts the value "n" from the current adjust value. The current adjust value may be accessed via the &SYSAD system variable symbol.

If "EVEN" is specified as an optional parameter then even numbered pages will be adjusted by the sum of the adjust value plus the even adjust. The "ODD" option works in the same way for odd numbered pages.

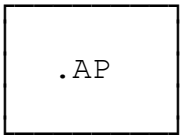
Defaults:

This command creates a break. When the operand for a simple adjust is omitted then the value of "n" comes from the initial value set by ADJUST or CENTER in the parm field, normally zero. The default for an EVEN or ODD Adjust is always zero.

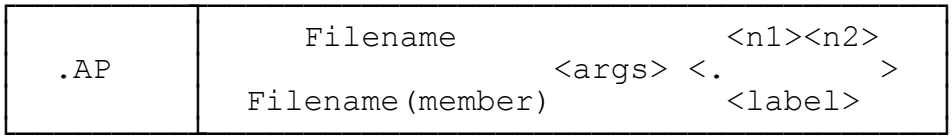
Examples:

(1) To check if adjust value is at least five:

```
.ur .if &SYSAD LT 5;.th .ad 5
This text is at least five spaces from left margin.
```



The APPEND control word allows an additional SCRIPT file to be appended to the file just printed.



Where:

<args> is a list of "n" elements separated by blanks, or delimited strings separated by blanks, such that special set reference symbol &1 is set to the value of the first, &2 to the second, &n to the last and &0 to "n" even if "n" is zero.

Alternatively, a keyword parameter may be set in this list of arguments if it contains an equal sign, no undelimited blanks and is a valid .SR assignment statement. If keyword parameters are used then &0 will be set, even if zero.

The special reference symbol &* is set to the string of all positional and keyword arguments.

When the default LOCAL option is specified, all of these reference symbols are defined as local variables and may only be used within the file being appended.

. is the current control word indicator character, normally period ".".

n1 specifies the start of the range of records to be included.

n2 specifies the end of the range of records to be included.

label specifies the label which marks the first record to be included.

When the .AP control word is encountered, the current SCRIPT file will be closed. Records "n1" through "n2" of the file or records following ".LB label" will be read and printed as a continuation of output already produced.

.AP

Each "Filename" that has not been used in a previous IMBED or APPEND Control word must have an associated "DD Statement" or "TSO FILE ALLOC" that references a dataset. If the "DD Statement" is missing SCRIPT will write an error message before closing all opened files.

Defaults:

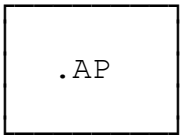
The entire file is read in starting at the first line and no break is created.

Notes:

(1) "*" may be coded in place of "n1 n2" to indicate that the entire file is to be read. If no arguments appear after the filename, "*" is assumed.
(2) If "n1" is coded, "*" may be coded in place of "n2" to indicate that the entire file starting with item "n1" is to be read. If the third argument is omitted, "*" is assumed.
(3) The .AP control word only allows files to be appended to the end of the current file. If it is desired to insert file contents at some point besides the end, the .IM control word should be used.
(4) If .AP FILENAME is used to supply the filename, a four step procedure is followed in searching for the referenced file. In order, a search is conducted for:
    -1- A DD statement "FILENAME" which points to a sequential dataset.
    -2- A DD statement "FILENAME" which points to a partitioned dataset, which itself contains a member "FILENAME'.
    -3- A member "FILENAME" in an active partitioned dataset. (The active partitioned datasets are searched for member "FILENAME" in reverse order of opening.)
    -4- A member "FILENAME" in the optional "SYSLIB" partitioned dataset.

If nothing is found, a message is printed out and an empty file is APpended.

If .AP FILENAME(MEMBERNAME) is used to supply the file name, step -3- of the search is skipped, and in steps -2- and -4-, the partitioned datasets in question are searched for a member named "MEMBERNAME" not "FILENAME".



Examples:

(a) .AP CHA4CONT
The file named CHA4CONT SCRIPT will be read and formatted for output as a continuation of the current SCRIPT file. The value of reference symbol "&0" is unchanged. If the reference symbol "&0" is set to zero. The value of the reference symbol "&*" is set to the null string. If the output is being created in batch, a "DD Statement" similar to one of the statements below would have to be included in the job setup for input of this file. If CHA4CONT were a sequential file:

```
//CHA4CONT DD DSNAME=xxxxxx,DISP=SHR
```

If CHA4CONT were a member of a library it could be referenced by including a "SYSLIB DD" statement.

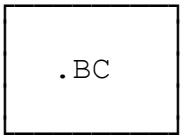
```
//SYSLIB   DD DSNAME=xxxxxx,DISP=SHR
```

If CHA4CONT were a member of a particular partitioned dataset the following "DD statement" would be necessary:

```
//CHA4CONT DD DSNAME=xxxxxx(CHA4CONT),DISP=SHR
```

(b) .AP FIG2 one 'one plus one' . 12 *
The file named FIG2 will be read starting with record 12 and output as a continuation of the current input file. The value of reference symbol "&0" will be set to "2", with "&1" and "&2" taking values "one" and "one plus one" respectively. The value of "&*" is set to "one 'one plus one'".

(c) .ap TEST type=CHAPTER TITLE='New World' err(5)=4*4
The file named TEST will be read with keywords TYPE, TITLE and ERR set. &0 will be set to zero.



The BALANCE COLUMNS control word enables or disables column balancing for multiple column output.

.BC	<ON|OFF>

When multiple column output is in effect, the columns of text on a page that is not full may be balanced when a page eject or column definition is encountered. This means that the number of lines in each column of text is made as equal as possible before being output. When balanced columns is turned off, the number of lines in each column is determined by filling each column with text or by column begin (.CB) or conditional column begin (.CC) control words. In this case the columns will not be equalized or balanced before being output.

Defaults:

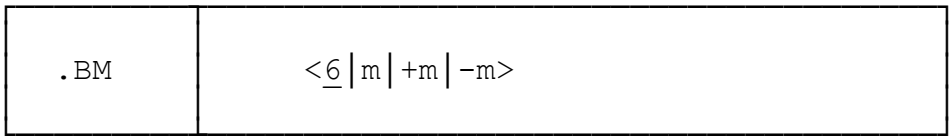
This control word does not create a break. The initial and default operand is "ON".

Notes:

(1) Even if BALANCE COLUMNS is ON, a column may be ineligible for balancing if it is terminated by a conditional or explicit column begin (.CB) control word or if the column fills a page and contains a footnote.

(2) A page eject or column begin that forces the termination of the last column on a page, does not mark a column ineligible for balancing.

The BOTTOM MARGIN control word specifies the number of lines which are to appear between the bottom of the output page and the last line of ordinary or footnote text.



At the bottom of all subsequent output pages (including the current page), m lines will appear between the bottom of printed text‘⅓’ and the physical bottom of the page. An operand of the form "+m" or "-m" adds this value algebraically to the current bottom margin setting, so long as the resulting value is not negative.

Any footing lines will appear within these m lines‘⅔’.

Defaults:

This command does create a break. Unless otherwise specified m = 6 will be in effect. When this command word is encountered and the operand is omitted then the value of the BMargin= parm will be taken, normally 6.

Notes:

(1) At no time may the value set in .BM be smaller than the sum of the .FM and .FS values.

---

(7) Footnote text included.
(8) Do not confuse these with footnote lines. Footing lines are those "bottom title" lines defined by the ".BT", ".EB" and ".OB" control words and are printed in the area defined by ".FS".

BREAK causes the immediately previous line to be typed without filling in with words from the next line.

BREAK is used to prevent concatenation of lines such as paragraph headings or the last line of a paragraph. It causes the preceding line to be typed as a short line if it is shorter than the current line length.

The optional "line" operand starts one blank after the control word and may be text or another control word.

Defaults:

This command does create a break. That is its only function.

Notes:

(1) Many of the other control words act as a BREAK. No BREAK is necessary when one of these is present. The description of each control word under Defaults, states if the control word acts as a break or not.
(2) A blank or tab in column one of an input line has the effect of a BREAK immediately before the line.
(3) If NO CONCATENATE is in effect, all lines appear to be followed by a BREAK.

Examples:

(1) Heading:
.br
First line of paragraph ...

This example would be printed as:

Heading:
First line of paragraph ...

Without the BREAK, it would be printed:

Heading: First line of paragraph ...

.BS

The BACKSPACE control word specifies the input character to be treated as a logical backspace and a hex join character.

.BS	<char> <HJOIN|NOHJOIN>

In all subsequent input text and control lines, each occurrence of the character "char" will be replaced with the backspace character. This is most useful when the user's input device does not offer a backspace facility.

In addition, the user defineable backspace character has the function of "hex join" unless "NOHJOIN" is specified. Thus if the character before the backspace is from "0" to "9", "A" to "F" or "a" to "f" and the character after the backspace is in the same range, then the three characters will be replaced by a single character. This hex join facility is supported only for the user backspace character and not the real backspace character X'16', although the real backspace could be specified as the user's backspace.

Defaults:

This command will not create a break when encountered. The initial default user backspace character is not defined. An omitted character operand will disable the facility.

Examples:

(1) .BS +;ABCDEFG+++++++_______

will produce:

ABCDEFG

(2) .bs ¬ HJOIN;In f¬_i¬_g¬_ B¬9B¬1¬¬__.

will produce:

In fig ⅛¹.

.BT

The BOTTOM TITLE control word is used to define three items of title information to be printed at the bottom of even and odd numbered pages.

.BT	<1|n> /S1/S2/S3/

where optional "n", from one to the value of the HSFSOVER option, gives the footing line number and S1, S2 and S3 are character strings not containing the delimiter character "/". The delimiter character can be any character, defined as the first character of the operand. Any of the fields may be omitted, but the delimiter character must be included to indicate missing fields, e.g., $S1$$S3$.

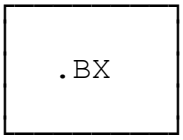
The .BT control word is used in a way similar to the .TT control word. The title items defined with .BT will be printed in footing lines near the bottom of even and odd numbered pages. The number of footing lines printed is set by .FS (Footing Space).

Defaults:

A break is not created by this command. Unless otherwise specified ".BT□////" will be in effect.

Notes:

(1) The BOTTOM TITLE control has the same effect as the FOOTING (.FE) control. The difference is that the field delimiter is self-defined by the first character of the operand.



The BOX control word creates the horizontal lines and initiates the vertical rules that will surround user text.

.BX	<v1 <v2 <v3 ... >>> <OFF> <DELETE>

With numeric operands, this command creates a horizontal line with vertical connectors above joining a previous .BX command (if any) and vertical connectors below in the specified columns to the next .BX command. The numeric operands specified must be in increasing column order and a signed operand is relative to the preceeding operand.

If no operand is present, the last Box verticals are repeated. A single operand draws a line, not a box.

The "OFF" operand terminates the box with a final horizontal line. The "DELETE" operand terminates the box with no final line.

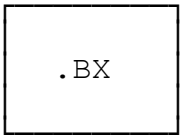
Two character sets are used by SCRIPT to draw the boxes. The first consists of '+', '-' and '|' which is used with the ONLINE option. The second set is the special box drawing characters of the TN print train, which is used with the OFFLINE option.

Defaults:

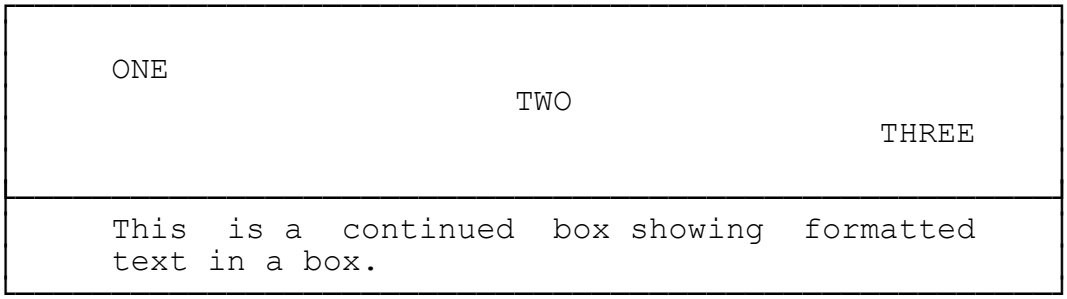
This command creates a break.

Examples:

```
(1) .in +5 -5
    .bx 1 &sysll
    .sk;.la ONE;.ce TWO;.ra THREE
    .sk;.bx
    This is a continued box showing
    formatted text in a box.
    .bx off
```

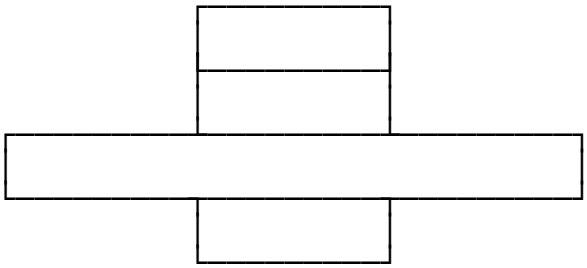


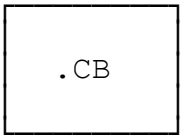
produces:



(2) .bx 20 30;.sp;.bx;.sp;.bx 10 40;.sp
.bx 20 30;.sp;.bx off

produces:





The COLUMN BEGIN control word causes subsequent text to begin at the top of a new column.

When in multiple column mode, this control word causes the following text to appear at the top of the next output column. If this control word is encountered while processing the last column on a page then it is equivalent to a page eject.

The same action may be caused by a CONDITIONAL COLUMN BEGIN (.CC).

Defaults:

This control word causes a break. No operand may be specified.

Notes:

(1) This control word is illegal in a Keep or Footnote.

The CONDITIONAL COLUMN BEGIN control word causes a column begin to occur if insufficient lines remain in the current column for text.

.CC	<0|n> <BEGIN|END <0|m>>

With a numeric operand, a check is made for "n" lines remaining in the current column. If "n" lines do not remain, a COLUMN BEGIN is performed which makes the current column ineligible for balancing. If "n" lines do remain, no action is performed, but if BALANCE COLUMNS is "ON" then output in the current column may be split by later balancing of column text.

Instead of counting output lines, input text may start with a "BEGIN" operand and terminate with "END". This works within columns the way ".CP" works with pages. The "END" operand may be further qualified with a numeric argument "m" where this value is added to the length of the formatted block before deciding if the block will fit on the current column.

Defaults:

This control word does not create a break. The default operand is zero which will not perform any action.

The COLUMN DEFINITION control word defines how many columns of output are to be formatted on each page and where each column is to start.

.CD	<n <d1 <d2 ... d9>>>

The first operand of the COLUMN DEFINITION control word defines the number of columns of output text that are to formatted from now on. The operand may range from 1 to 9 columns and may be signed, indicating a change relative to the current definition.

The following numeric operands specify up to nine displacements relative to the current adjust (.AD) setting, after which the formatted output columns are to appear. A zero displacement means no displacement at all. If a displacement is signed, it means a change relative to the previous displacement in the same control word.

If all operands are omitted, it is a redefinition of the current COLUMN DEFINITION. Note the .CD control word causes all input text to that point to be printed with the former definition. If displacement operands are omitted, their former value is retained.

The COLUMN DEFINITION does not set the width of each column. See the COLUMN LENGTH (.CL) control word. If text should overflow past the displacement of a following column because of FORMAT NO text or overlays, the text of the following column will replace the former.

Defaults:

This control word causes a break. The initial number of columns is one, with displacements 0, 46, 92, 0, 0, 0, 0, 0, 0. Omitted operands in this control word retain their former value.

Notes:

(1) This control word is not allowed in Keeps or Footnotes.

Examples:

(1) .cd 1 0
This is the same as single column mode. It specifies one column at displacement zero.

(2) .ll 60;.cl 27;.cd 2 0 33
This specifies a Line Length of 60 and a Column Length of 27. Text will be formatted from column 1 (displacement 0) to column 27 and from column 34 (displacement 33) to column 60, leaving a 6 character "gutter" or white space between columns.

.CE

The line following the CENTER control word will be centered between the margins.

.CE	<1|n|YES|NO|line>

The "line" operand or the next "n" text lines in the input file, including leading blanks, will be centered between the left and right margins.

Defaults:

This command causes a break. A numeric operand will center the following "n" input lines. A "YES" operand will center all following input lines until a "NO" operand is encountered.

Notes:

(1) If the line to be centered is longer than the current line length, it will be truncated and not centered.
(2) The left and right margins are the value of any indent value (.IN) and the current Line Length altered by Right Indent, respectively.

Examples:

(1) .CE Other Methods

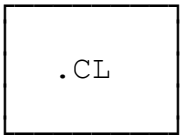
When this line is typed, the characters "Other Methods" will be centered:

Other Methods

(2)
```
.ce 3;first line
.sp;number two line
third line
```

produces:

```
                first line

              number two line
                third line
```



The COLUMN LENGTH control word defines the width in characters of each column in multiple column mode.

.CL	<0|n|+n|-n>

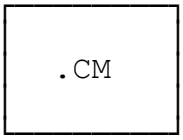
The .CL control word defines the number of characters in each column of formatted output, when in multiple column mode. The COLUMN LENGTH may be set with a numeric argument or be changed relative to its current value with a signed argument.

A zero argument is a special case. This sets the Column Length equal to the current Line Length (.LL).

This control word is used with COLUMN DEFINITION (.CD) to define the column and gutter placement on a page.

Defaults:

This control word causes a break. The initial and default value is equal to the Line Length.



The COMMENT control word is ignored and may be used to enter comments into a SCRIPT file.

.CM	<anything>

The .CM control word allows comments to be stored in the SCRIPT file. These comments may be seen whenever the input file is edited or printed.

Comment lines may be used to store unique identifications for use during editing of the file.

Defaults:

This command does not cause a break.

Notes:

(1) Since only the first two characters of a control line (exclusive of the control word indicator, normally ".") are examined to determine what control word is present, and since undefined reference names have null-string ("") values, a reference name can be given the value 'CM' and used to control conditional recognition of other control words. (See Example 2).

(2) If the control word cannot be identified then a first control word character of asterisk "*" will be recognized as a comment that goes to the very end of the current input record, regardless of control word separators and other control words.

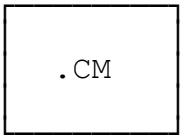
Examples:

(1) .CM .NF used below.

The comment line will be seen when examining the printout of the SCRIPT file and will serve to explain what is going on.

(2)
```
.sr stop0='CM '
.ur .ur .&&stop&n..im setup
```

If the reference name "n" has any value other than



0, the line will be:

```
.im setup
```

If the reference name "n" has the value 0, the line will be:

```
.CM im setup
```

which is interpreted as a comment and thus ignored.

(3)
```
.cm This is a comment
.cmThis is a comment
.*  This is a comment
.* This is a comment
.*This is a comment
```

The control lines above are equivalent.

(4)
```
.cm This is a comment followed by a space;.sp
.* This is a comment followed by a space;.sp
```

The control lines above are not equivalent.

CONCATENATE cancels a previous NO CONCATENATE control word and causes output lines to be formed by concatenating input lines and truncating at the nearest word boundary to fit within the specified line length.

.CO	<YES|NO>

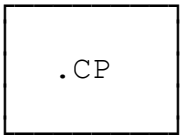
The CONCATENATE control word specifies that output lines are to be formed by shifting words to or from the next input line. The resulting line will be as close to the line length as is possible without exceeding it or splitting a word.

Defaults:

This command creates a break and is in effect until a ".CO□NO" is encountered. An omitted operand is treated as "YES".

Notes:

(1) CONCATENATE is the normal mode.
(2) Output produced with CONCATENATE in effect is similar to what a typist produces manually.



The CONDITIONAL PAGE control word causes a page eject to occur if insufficient lines remain on the current page for text.

.CP	<0|n> <BEGIN|END <0|m>>

The .CP control word will cause a page eject to occur if "n" lines do not remain on the current page.

By putting ".CP BEGIN" before a block of text and ".CP END" after, the length of the formatted block need not be known and still a page eject will be generated before printing the block if insufficient space is available on the current page. The "END" operand may be further qualified with a numeric argument "m" where this value is added to the length of the formatted block before deciding if the block will fit on the current page.

This control word is especially useful:

(1) Before a .SP control word which is used to leave room for a drawn figure. Use of .CP will guarantee that all of the spaces are contiguous‘⅛’.

(2) Preceding a section heading to insure that the heading will not be left alone at the bottom of the page.

Defaults:

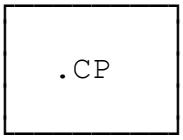
This command will not cause a break. If "m" is omitted then zero is assumed. If "n" is omitted or not positive then the command will be ignored.

Notes:

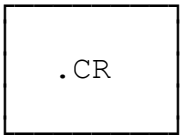
(1) By "remainder of current page" it is meant "the area between the last typed line and the beginning of saved footnote text (if any) or the current bottom margin (if no footnotes are saved)."

---

(9) If a numeric argument is used with .CP, then .LE YES must be specified if the spaces entered in this way are to be left at the top of the next page.



(2) A ".CP BEGIN/END" sequence within a Floating Block, a Floating Keep or a Footnote is considered illegal.



The CONTROL WORD REPLACEMENT control word is used to change the control word that identifies a SCRIPT function, to delete a SCRIPT function or to reset the set of SCRIPT control words.

.CR	<OO <WW>>

The original control word "OO" will be replaced by a working control word "WW". The "OO" control word will no longer be recognized as a valid control word since the "WW" control word has replaced it. If "WW" is also a valid control word then it will lose its former function. The replacements made are not cumulative but always work from the original control words. If the second argument "WW" is missing, then the original control word is reset for that word and any working control words equal to "WW" are reset to their original values. If both arguments are omitted then all working control words are reset to their original values.

Defaults:

This command will not create a break when encountered.

Examples:

(1) .cr sp jp
The SPACE control word is replaced by JUMP. The "jp" control word has all the functions and attributes of "sp" which is no longer recognized.

(2) .cr te rc
The TERMINAL READ control word is replaced by READ CONTROL. The "rc" control word has all the functions and attributes of "te" which is no longer recognized. Moreover, the REVISION CODE "rc" control word is no longer available.

(3) .cr rc
Following example (2) this will reset TERMINAL READ to "te" and restore the availability of the REVISION CODE control word, "rc".

The CONDITIONAL SECTION control word causes only selected portions of the total document to be printed by associating a conditional section code with selected portions of the document.

.CS	<ON|OFF> n <INCLUDE|IGNORE>

n represents a conditional section code and may be any number from 1 to 99. If "n" is omitted then reference is being made to a conditional section always to be ignored.

ON specifies following text and control words are part of conditional section "n". A ".cs□on" creates the start of a conditional section to be ignored.

OFF specifies that conditional section "n" has ended. A ".cs□off" is the only way to terminate a previous ".cs□on".

INCLUDE specifies that all following conditional sections with code "n" are to be included. This is initially the default for all conditional sections.

IGNORE specifies that all following conditional sections with code "n" are to be ignored.

The CONDITIONAL SECTION facility provides the means for printing only unclassified portions of a confidential report or including information on different versions of a system within the same input file, and having only the information pertaining to a given version printed. This may be done for entire chapters, single paragraphs or words within a sentence.

Defaults:

This command does not create a break when encountered. The second operand must always be present; "n" is optional.

Examples:

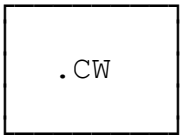
(1)
```
.sr incl=1;.ur .cs &incl ignore
This is
.cs 1 on;a User's
.cs 1 off
.cs 2 on;an Implementation
.cs 2 off
Guide.
```

Produces:

This is an Implementation Guide.

(2)
```
 ...
.su on
.if '&debug' = 'NO'
.cs on
.br
DEBUG CALL TO CHECKER (&debug.)
.im CHECKER &debug
.br
.cs off
 ...
```

The message about and the call to CHECKER will not be made if '&debug' has the value "'NO'".



The CONTROL WORD SEPARATOR control word defines the delimiter between control commands so that multiple controls and text may be entered on one line.

.CW	<;|character>

All subsequent input lines with a control word at the start are examined for the CONTROL WORD SEPARATOR. If found then the single input line is logically divided into multiple input lines. The subsequent line will be taken following the separator up to the next separator or to end of line.

Defaults:

This command will not create a break when encountered. The initial default CONTROL WORD SEPARATOR character is the semi-colon ";". Any character may be used as an argument. If the argument is omitted then more than one control word may not be entered per line and control words and text may not be entered on one line.

Examples:

(1) .sp 2;.un 4;(a) this is example number one ...

(2) .cw #;.he 'left;'center;'right;'#.cw ;#.sp;.cm
The above example makes the CW "#", defines a heading using a semi-colon, redefines the CW to be ";" and then uses it.

The DARK OUTPUT control word specifies the number of times that Offline Output is to be overstruck.

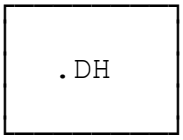
.DA	<1|n|+n|-n|YES|NO>

The .DA control word defines the number of times that all Offline Output lines are to be overstruck. In the case of Footnotes and Keeps, they are overstruck according to the DARK value when these are defined, not when they are printed. Only entire lines can be overstruck, not parts of lines. See .OC for partial line overstriking. The control word has no effect for Online Output, with or without the SCREEN option.

A "NO" operand is the same as 1; the "YES" operand is the same as 2.

Defaults:

This control word does not create a break. Unless otherwise specified n = 1 is in effect. If the operand is omitted then the value of the DArk= parm will be taken, normally one.



The DEFINE HEADING control word defines the formatting conditions for the various heading levels in the Table of Contents.

.DH	<SET m> n <<SKBF n> <SPAF n> <TCIN n> <TCOF n> <BR|NBR> <OJ|NOJ> <PA|NPA> <TC|NTC> <TO|NTO> <TS|NTS> <UP|NUP> <US|NUS>>

The .DH control word is used to respecify the formatting options for Head Level (.HL) operands. The "SET" option may specify a Table of Contents number "m" from 0 to 9. Thus up to ten concurrent Tables of Contents may be accumulated for printing later with the Table of Contents (.TC) control word.

The head level to be modified is specified with "n" and may range from 0 to nine. See the Head Level control word (.HL) for the initial setting of all options.

SKBF n "n" is the number of skips (.SK) to be printed before the Head Level in the formatted text.

SPAF n "n" is the number of spaces (.SP) to be printed after the Head Level in the formatted text.

TCIN n "n" is the number of columns to indent (.IN) a title in the Table of Contents.

TCOF n "n" is the number of columns to offset (.OF) a multiple line title in the Table of Contents.

BR do a break after the Head Level in formatted text.

NBR don't do a break.

OJ Out Justify (.OJ) the Head Level in formatted text.

NOJ don't Out Justify.

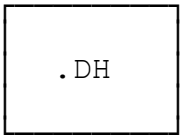
PA do a page eject before the Head Level if not already at the top of a page.

NPA don't do a page eject.

TC include this Head Level entry in the Table of Contents.

NTC don't include in the Table of Contents.

TO include this Head Level entry only in the Table of Contents.



NTO include this Head Level entry in the text.

TS space one line (.SP) before a Head Level entry in the Table of Contents.
NTS don't space before in Table of Contents.

UP convert the Head Level in formatted text to upper case.
NUP don't convert the Head Level to upper case.

US Underscore (.US) the Head Level in formatted text.
NUS don't underscore the Head Level.

Defaults:

This control word does not create a break.

The DEFINE MACRO control word defines a sequence of input lines to be invoked by "name" as a user-defined control word or as a SIGNAL (.SI) operand.

.DM	name /line1/.../linen</> name <BEGIN|END> name DELETE

The .DM control word is used to define and delete a user macro. such user macros may be used for common sequences of control words and text. Keyword and positional parameters "&*", "&0", "&1", etc. may be checked and substituted when the macro is called.

The user macro is known by "name", a one to eight character identifier. The macro defines a sequence of control words and text lines that are invoked by a ".name" macro call or ".SI name" Signal.

The user macro may be defined in two ways. The first is on one input line with the lines of the macro being separated by a self-defining character shown in the command prototype as "/". Longer user macros are defined with a "name␣BEGIN" at the start and "name␣END" to terminate. The ".DM␣name␣END" must start in column one of an input record.

A user macro may be deleted by specifying "name␣DELETE" as an operand. "name␣OFF" is an alternate way to delete a macro.

Defaults:

This control word does not create a break when defined. It does not create a break when called either, although the macro may contain control words which will cause a break.

Notes:

(1) The calling of defined user macros by ".name" can be suppressed with the ".MS" (Macro Substitution) control word. Calling by ".SI␣name" cannot be suppressed.

(2) Currently the ".DM␣name␣END" operands are not verified. Only ".DM" starting in column one is checked.

The DO control word may be used following a Then or Else control word to allow multiple input lines to be conditionally included.

.DO	<BEGIN|END>

This control word may only be used as the object of a Then (.TH) or Else (.EL) control word. The input control word and text lines from the .DO to the next corresponding ".DO□END" or ".EN" control word are all treated as the object of the Then or Else and are included or ignored depending on the truth value of the preceeding If.

Further Ifs within a Do group are valid, as are Imbeds (.IM) and Signals (.SI).

Defaults:

The ".do" control word does not act as a break in itself. However, control words within the Do Group may create a break. If no operand is specified then "BEGIN" a Do Group is assumed.

Examples:

(1)
```
.ur .if '&copy' = 'F';.th .do begin
.tt //FINAL COPY//
.do end
.el .do begin;.tt //DRAFT COPY//;.do end
```

This will define one of two Top Titles depending on the value of variable symbol "&copy".

.EB

The EVEN BOTTOM control word is used to define three items of title information to be printed at the bottom of even numbered pages.

.EB	<1|n> /S1/S2/S3/

where optional "n", from one to the value of the HSFSOVER option, gives the even footing line number and S1, S2 and S3 are character strings not containing the delimiter character "/". The delimiter character can be any character defined as the first character of the operand. Any of the fields may be omitted, but the delimiter character must be included to indicate missing fields, e.g., $S1$$S3$.

The .EB control word is used in the same way as the .BT control word. The footings defined with .EB will appear only on even numbered '10' pages, however. The number of footing lines printed on even pages is set by .FS (Footing Space).

Defaults:

A break is not created by this command. Unless otherwise specified ".EB□////" will be in effect.

Notes:

(1) The EVEN BOTTOM control has the same effect as the EVEN FOOTING (.FV) control. The difference is that the field delimiter is self-defined by the first character of the operand.

---

(10) Even numbered pages are bound on the right margin.

.EF

The END OF FILE control word causes immediate termination of the current input file and resumption of the next higher level file if any, or termination of processing if none.

.EF	<YES|NO>

When SCRIPT encounters an END OF FILE control word in a file that is being imbedded, the imbedded file is terminated and the higher level file is resumed. Similarly, a Remote containing this control word will be terminated, returning to the next higher level file. When SCRIPT encounters an END OF FILE control word in a file that is not imbedded, the output advances to the top of the next page and prints any stacked footnotes or text created by ".CP or .FK begin/end" sequences before termination of all processing.

Defaults:

This command does not create a break.

The ELSE control word causes an input line to be conditionally included depending on the truth value of a previous IF control word.

The line which begins one blank after the .EL control word is included for processing only if the preceeding .IF statement had a "false" value.

The line may include any control word except another Then (.TH) or an Else (.EL). The object line may be another .IF and these may be nested up to ten levels. The object may be an Imbed (.IM) or Signal (.SI), in which case the current IF status and its nesting level will be saved and later restored when the current file nest level is resumed. When using a nested IF, the Else is always matched to the innermost unpaired Then. It may thus be necessary to enter an Else with no object line to define the required branching structure.

This control word may be used only immediately following a .IF, a .TH or a .TH .DO/.EN group.

Defaults:

The ".el" control word does not act as a break itself. However, control words within the line may create a break if the preceeding .IF is "false" and the interpreted line creates a break. If "line" is omitted then the object of the .EL has no effect.

Examples:

(1) Have a
```
.if &retcode = 0;.th great
.el .if &retcode = 4;.th good
.el .if &retcode = 8
.th bad
.el .if &retcode = 12;.th terrible;.el *!?
day.
```

This sequence will format a sentence describing the kind of day that is to be hoped for.

(2)
```
.ur .if &type = FINAL;.th .do begin
.tt //SCRIPT//
.bt //-%-//
.do end
.el .do begin
.tt //SCRIPT DRAFT//
.bt //-%-/Please Return/
.do end
```

This sequence will set page titles depending on the value of variable &type.

.EM

The EMPTY PAGE control word is used to control suppression of empty (except for headings and footings) pages.

.EM	<YES|NO|OFFNO>

Empty pages[11] can be generated in a number of ways and are not normally printed by SCRIPT.

By specifying ".EM□YES", empty pages are to be printed. ".EM□NO" specifies that empty pages are not to be printed. ".EM□OFFNO" specifies that empty pages are not to be printed and the page number is not to be incremented.

Defaults:

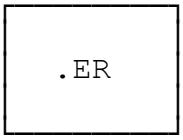
A break will not be created and empty pages will not be printed unless ".em□yes" or ".em" is encountered.

Notes:

(1) If the operand is omitted, "YES" is assumed, since the user presumably intends to accomplish something with the control word.
(2) ".EM□NO" is the initial value for EMPTY PAGE.
(3) The first page of an online file is considered empty, whereas the first page of an OFFLINE file is not.
(4) If pages are suppressed as a result of ".EM□NO", page numbers will still increment unless ".PN□OFFNO" has been specified.

---

(11) Pages which would, if printed, contain only headings, footings, and possibly footnotes.



The ERROR control word sets a program return code for SCRIPT and/or displays a message on the Error file.

.ER	<n|*> <line>

The optional first operand of ERROR may be used to set the return code passed back to the caller of SCRIPT. If the first operand is a numeric expression then the maximum of that value and the current return code is set. If the first operand is omitted or '*' then no return code is set.

One blank following the first operand is a line of information that is displayed as a SCRIPT error message on the Error file (SYSTERM), regardless of ONLINE or OFFLINE, or the current Pass number. If no string follows the return code operand then no output is done.

The system variable symbol &SYSRETCODE is provided to examine the current return code.

Defaults:

This command word will not create a break. Zero is the default return code value if none is specified.

An error return code of sixteen or higher is considered to be a Terminal Error and will terminate all input and output processing.

Examples:

(1) .er 4 This is a comment about an error.
This example sets the program return code to "four" and prints a warning message on SYSTERM.

(2) .er 16
This example sets the return code to "sixteen" and prints nothing.

(3) .er * HI!
This example leaves the return code unchanged and prints a "HI!" message on SYSTERM.

.ET

The EVEN TITLE control word is used to define three headings to be printed at the top of even numbered pages.

.ET	<1|n> /S1/S2/S3/

where optional "n", from 1 to the value of the HSFSOVER option, gives the heading line number and S1, S2 and S3 are character strings not containing the delimiter character "/". The delimiter character can be any character defined as the first character of the operand. Any of the fields may be omitted, but the delimiter character must be included to indicate missing fields, e.g., $S1$$S3$.

The .ET control word is used in the same way as the .TT control word. The headings defined with .ET will appear only on even numbered(12) pages, however. The number of even page heading lines printed is set by .HS (Heading Space).

Defaults:

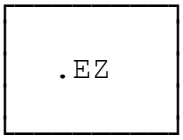
A break is not created by this command. Unless otherwise specified ".ET□///PAGE□%/" will be in effect.

Notes:

(1) The EVEN TITLE control has the same effect as the EVEN HEADING (.HV) control. The difference is that the field delimiter is self-defined by the first character of the operand.

---

(12) Even numbered pages are bound on the right margin.



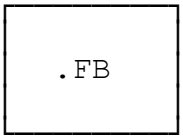
The EasySCRIPT control word is used to initiate and provide input to the EasySCRIPT macro processor.

.EZ	<ON <lastDewey>> <OFF > <tagval line >

For now, refer to pages 60-65 and 92-93 of the IBM SCRIPT/370 Version 3 User's Guide (SH20-1857) for a description of the language and facilities.

Notes:

(1) EasySCRIPT is currently in development and test. It is not now distributed.



The FLOATING BLOCK control word allows the user to enter a block of text that will print later in the document.

.FB	<BEGIN|END <0|m>> <DUMP <n>>

The .FB control word defines the beginning and ending of a block of text that is to be printed later in the document. When the ".FB□BEGIN" control word is encountered, the values of all relevent print control variables are saved and SCRIPT prepares to accept Floating Block text. When the ".FB□END" control word is encountered, a break is caused for text within the block and the former values are restored.

More than one Floating Block may be defined; each new Block is added internally to the end of the current formatted blocks. In this way, a list of Endnotes could be accumulated within a chapter to be printed at the end, or a list of Reference or Bibliography entries could be entered at the point of reference and be printed later.

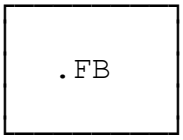
The ".FB□DUMP" operand is used to cause printing of the formatted block after the "END" operand is encountered. A second numeric operand may be specified to indicate how many Floating Blocks are to be printed. If not specified then all blocks accumulated to this point are printed.

Defaults:

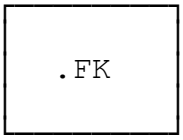
This command does not create a break. A first operand must be provided as none will be assumed. The default length adjustment for "END" is zero. The default count of blocks to print for "DUMP" is all blocks.

Notes:

(1) A Conditional Page (.CP BEGIN/END) or a Floating Keep (.FK BEGIN/END) or a Footnote (.FN BEGIN/END) sequence within a Floating Block is considered illegal.



(2) The SYSFBC system variable symbol can be examined after the ".FB□END" control word to determine the count of queued lines within all outstanding blocks. The SYSFBF system variable symbol can be used to determine the count of queued lines within the first outstanding block.



The FLOATING KEEP control word enables the user to enter a block of text that will print together, either immediately or at the top of the next page.

.FK	<BEGIN|END <0|m>> <DUMP <n>>

The .FK control word defines the beginning and ending of a block of text that is to print together on one page. When the ".FK BEGIN" control word is encountered, the values of all relevent print control variables are saved and SCRIPT prepares to accept Floating Keep text. When the ".FK END" control word is encountered the former values are restored.

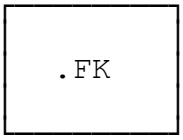
The ".FK END" routine also decides if there is space on the current page to print the Floating Keep in its entirety. An optional numeric operand may be used to adjust the actual size of the Floating Keep for comparison purposes. If yes, then it will be printed. If not, the formatted text will be saved for the top of the next page immediately following the top margin spacing. If the text saved exceeds one page in length then it will print starting at the top of the next page and continue onto subsequent pages till finished.

If the Floating Keep text is forced to the next page, the current page will be filled with the text which follows the ".FK□BEGIN/END" sequence.

The ".FK DUMP" operand is used to print an outstanding Floating Keep. Each outstanding block of text will print at the top of a page if there is insufficient room on the current page. A second numeric operand may be specified to indicate how many Floating Keeps are to be printed. If not specified then all blocks accumulated to this point are printed.

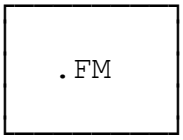
Defaults:

This command does not create a break. An operand must be provided as none will be assumed. The default length adjustment for "END" is zero.



Notes:

(1) A Conditional Page (.CP BEGIN/END) or a Floating Block (.FB BEGIN/END) or a Footnote (.FN BEGIN/END) sequence may not be defined within a Floating Keep.
(2) The SYSFKC system variable symbol can be examined after the ".FK END" control word to determine if the Floating Keep has printed on the current page or will print on the next page.



The FOOTING MARGIN control word specifies the number of blank lines which are to left between the bottom of formatted text and any footing line.

.FM	<1|n|+n|-n>

The .FM control word defines the footing margin, which is the number of blank lines which will be left between the bottom of formatted text and the first footing line on all pages. The first footing line is the top of the FOOTING SPACE area. An operand of the form "+n" or "-n" adds this value algebraically to the current footing space setting, so long as the resulting value is not negative.

If the footing margin is fm, the footing space is fs and the bottom margin bm, the bottom of each page will appear as follows:

(1) fm blank lines,
(2) the fs footing lines,
(3) bm - fm - fs blank lines.

Defaults:

This control word creates a break when encountered and until then n=1 will be in effect. If the operand is omitted then the value of the FMargin= parm will be taken, normally 1.

Notes:

(1) The FOOTING MARGIN plus FOOTING SPACE must always be less than or equal to the BOTTOM MARGIN.

The FOOTNOTE control word allows the user to enter a footnote which will be printed at the end of the current page.

.FN	<BEGIN|END> SET n </S1/S2/S3>

When the ".fn□begin" control word is encountered, the current values of all relevent control variables are saved and SCRIPT prepares to accept footnote text.

When the ".fn□end" control word is encountered, the values saved by the ".fn□begin" are restored and formatting of output continues.

SCRIPT saves enough room at the bottom of the page to print the footnotes and separator lines. If enough room does not exist, footnotes will be continued at the bottom of the next page. Any formatting control word may appear within a footnote except another Footnote (.FN) or Keep (.CP, .FK, .FB) control word, a page eject (.PA) or an Immediate Line (.LN). Space (.sp) requests in footnotes are normally forced to single space mode and all formatting requests apply only to the footnote[13].

The SET function is used to define the footnote separator lines. The ".FN□SET□n" form is used to define the number of lines that separate the bottom of the body of text from the top of the Bottom Margin. The initial value of "n" is three and "n" must be a numeric value from one to the HSFSOVER parameter, normally nine.

The ".FN□SET□n□/S1/S2/S3/" form is used to define line number "n" of the footnote separator. The "/S1/S2/S3/" operand is like Top Title (see .TT) where "S1" would be left adjusted in footnote separator line "n", "S2" would be centered and "S3" would be right adjusted. The initial

---

(13) Format controls in effect when the ".fn□begin" is encountered remain in effect during formatting of the footnote unless explicitly overridden. For example, if indentation of the text has been requested with a .IN control word, the footnote will also be indented unless the .IN is altered within the footnote.

.FN

values for all separator lines is blank except for the second which is a centered row of dashes "-" of length one third the line length.

Defaults:

This command does not create a break. An operand must be provided as none will be assumed.

Notes:

(1) Footnotes are always initialized to single spaced output. The Line Spacing may be altered and will remain in effect until the end of the Footnote.
(2) No more than the value of the FNSIZE= parameter (normally 200) footnote lines may be waiting for output at any time.
(3) The ".fn□begin" control does not act as a break. The next regular input line‘¼’ will be concatenated with the previous line.
(4) No footnotes will appear on a page if at least 4 lines are not available at the bottom of the text area. If footnotes are generated within 5 lines of the bottom, they will be saved for the next page. This assumes a footnote separator depth of three lines.
(5) The line which precedes a footnote definition will always be printed on the current page. It can never be displaced to the next page by accumulation of footnotes.
(6) Footnotes appear at the bottom of the text area, NOT in the Bottom Margin area set by the ".bm" control word.
(7) A ".FN BEGIN/END" sequence within a Keep or another footnote is considered illegal.
(8) A Floating Keep (.FK) or Conditional Page (.CP or .CC) text block that prints starting at the top of a page will displace footnote lines on that page if the text block extends down into the footnote area above the Bottom Margin.
(9) Footnote numbering, as used in this manual, is described in the section "NOTES ON THIS DOCUMENT", in the Appendices.

---

(14) That is, the first line after the ".fn□end".

.FN

Examples:

```
.un 4
(a) As Anderson -1-
.fn begin;.in 5;.un 5;-1- Anderson, D.A.,
.us Grundlagen_der_Madchenjadgt,
MIT Press, August 1932.
.fn end
has noted, this phenomenon is indigineous to ...
```

Can be used to obtain output similar to that shown here:

(a) As Anderson -1- has noted, this phenomenon is indigineous to ...

-1- Anderson, D.A., Grundlagen der Madchenjadgt, MIT Press, August 1932.

The FORMAT control word combines the effect of CONCATENATE and JUSTIFY.

.FO	<YES|NO LEFT|RIGHT|CENTRE INSIDE|OUTSIDE|HALF>

The FORMAT control word is a short-hand way to specify CONCATENATE and JUSTIFY. This control word specifies that output lines are to be formed by shifting words to or from the next line (concatenate) and padded with extra blanks to produce an even right margin (justify).

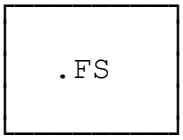
The "NO" operand is equivalent to the control words CONCATENATE NO and JUSTIFY NO. The other possible operands specify CONCATENATE and the appropriate mode of JUSTIFY.

Defaults:

This command creates a break. It is in effect unless a ".FO NO" is encountered. An omitted operand is treated as "YES".

Notes:

(1) Since FORMAT is the normal mode, the control word is used only to cancel a previous FORMAT NO control word.
(2) The .FI and .NF control words are provided as aliases for compatibility with FILL and NO FILL in other text formatters. FILL performs and functions the same as .FO.



The FOOTING SPACE control word specifies the number of footing lines to be printed at the bottom of both even and odd numbered pages.

.FS	<1|n|+n|-n>

The .FS control word controls the number of footing lines to be printed at the bottom of a page. Up to HSFSOVER footings may be defined and printed at the bottom of each page. An operand of the form "+n" or "-n" adds this value algebraically to the current footing space setting, so long as the resulting value is not negative.

If the footing margin is fm, the footing space is fs and the bottom margin bm, the bottom of each page will appear as follows:

(1) fm blank lines,
(2) the fs footing lines,
(3) bm - fm - fs blank lines.

Defaults:

This command word creates a break when encountered and until then n=1 will be in effect. The value of the operand may range from zero to the HSFSOVER parameter, normally nine. If the operand is omitted n=1 will be assumed.

Notes:

(1) The FOOTING MARGIN plus FOOTING SPACE must always be less than or equal to the BOTTOM MARGIN.
(2) See the description of Top Title (.TT) for notes on the interrelation of Top and Bottom Titles.

The GOTO control word specifies that the next input record is to be selected out of normal sequence.

.GO	<ident|n|+n|-n>

The .GO control word defines an identifier that must match the identifier of a LABEL (.LB) control word within the current input file. The identifier is converted to upper case prior to use and may range from one to eight characters in length. Alternatively an absolute record number or signed relative record number may be specified. In either case input processing will continue at the specified record.

The transfer of control may be forward or backward within the input file if on a DASD device (Disk) but may only be forward if on a Unit Record device (Card Reader).

Defaults:

This command does not create a break.

Examples:

```
(1) .ur .if '&1' = 'DONE';.th .go DONE
    Not finished yet.
    .lb DONE The End.

(2) .se i=0;.lb LOOP .se i=&i+1
    .ur &i
    .ur .if &i lt 50;.go LOOP
    End of Block of fifty numbers.
```

The HANGING INDENT control word specifies an indent of additional spaces in each line of a paragraph after the first.

.HI	<0|n|+n|-n|YES|NO>

The ".HI" control word causes "n" spaces to be inserted at the beginning of each line of a "paragraph", except the first. See the PARAGRAPH INDENT control word (.PI) for control of the first line. The operand may be signed, in which case the current value is incremented or decremented appropriately. A "YES" argument will set the hanging indent value to five. A "NO" argument is equivalent to a zero operand and terminates the hanging indent function. The resulting value of hanging indent may never be negative or exceed the current line length ".LL" minus one.

A hanging indent takes effect following a paragraph indent. See the PARAGRAPH INDENT control word (.PI) for rules that determine the start of a new "paragraph".

Note that the Hanging Indent function is in addition to the current INDENT (.IN) value.

The Hanging Indent function may be overridden by use of the OFFSET control word (.OF). Setting a Hanging Indent value will clear any OFFSET currently in use.

Defaults:

This control word creates a break. An initial value of zero is assumed, which is also assumed if the operand is omitted.

Examples:

```
(1) .in +5
    .hi 5
    (a)  This is point number one that will demonstrate
    the function of the Hanging Indent control word.
    .sp
    (b)  After every break in the text, the letter eye
    catchers are reset to the current Indent value.
    .sp
    (c)  But the second and subsequent lines of each point
```

.HI

are indented further from the left margin as if an OFFSET control word had preceeded each point.
.sp
.hi

Produces:

(a) This is point number one that will demonstrate the function of the Hanging Indent control word.

(b) After every break in the text, the letter eye catchers are reset to the current Indent value.

(c) But the second and subsequent lines of each point are indented further from the left margin as if an OFFSET control word had preceeded each point.

The HEADING LEVEL control word adds a Heading to the text and/or to the current Table of Contents.

.HL	<0|n> line

The .HL control word adds a "line" heading to the formatted output and/or to the current Table of Contents. The first parameter "n" defines what level of heading it is and may range from 0 to 9.

The meaning for the options may be found in the Define Heading (.DH) control word.

.HL	0	1	2	3	4	5	6	7	8	9
SKBF	0	0	3	3	3	1	1	0	0	0
SPAF	0	5	2	2	2	0	0	0	0	0
TCIN	0	0	0	2	4	6	8	0	0	0
TCOF	1	1	1	1	1	1	1	0	0	0
BR		Y	Y	Y	Y					
OJ		Y								
PA		Y								
TC	Y	Y	Y	Y						
TO	Y									
TS		Y								
UP		Y	Y	Y		Y				
US		Y	Y		Y	Y	Y			

Defaults:

This control word causes a break.

.HM

The HEADING MARGIN control word specifies the number of blank lines which are to be left between the HEADING SPACE area and the first line of the text area.

.HM	<1|n|+n|-n>

The last heading line produced on subsequent output pages will be separated from the first line of text by n blank lines. An operand of the form "+n" or "-n" adds this value algebraically to the current heading margin setting, so long as the resulting value is not negative.

If the current top margin is tm, the heading margin is hm and the heading space is hs, the top of each page will appear as:

(1) tm - hm - hs blank lines,
(2) the hs heading lines,
(3) hm blank lines.

Defaults:

This command creates a break and until it is encountered n=1 will be in effect. If the operand is omitted then the value of the HMargin= parm will be taken, normally 1.

Notes:

(1) The HEADING MARGIN plus HEADING SPACE must be less than or equal to the TOP MARGIN.

HEADNOTE

The HEADNOTE control word defines a block of formatted text to appear at the top of all pages.

.HN	<EVEN|ODD> <BEGIN|END> <EVEN|ODD> <PURGE|DUMP>

The .HN control word with "BEGIN" and "END" operands, defines the limits of text to be formatted and printed following the Top Margin area of each subsequent output page. The formatting environment is saved and restore while defining the Headnote.

The "PURGE" operand deletes the current Headnote. Note that "CANCEL" and "DELETE" are equivalent operands to "PURGE". The "DUMP" operand may be used to print the Headnote at a place other than the top of a page.

An optional parameter "EVEN" may be specified to define a Headnote that will appear only at the top of even numbered pages. If both a Headnote and an Even Headnote are defined, the EVEN Headnote takes precedence on even pages. An optional parameter "ODD" has the corresponding effect on odd numbered pages.

Any outstanding Conditional or Floating Keep text will print following the Headnote.

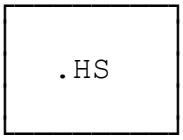
Defaults:

This control word does not create a break. No operand is assumed by default.

Examples:

(1)
```
.HN BEGIN
.SU .BX 1 &sysll
.SP
.CE;.UC headnote
.SP
.BX off
.SP
.HN END
```

This example appears at the top of this page.



The HEADING SPACE control word specifies the number of title lines to be printed at the top of both even and odd numbered pages.

.HS	<1|n|+n|-n>

This control word controls the number of heading lines to be printed at the top of a page. Up to HSFSOVER headings may be defined and printed at the top of each page. An operand of the form "+n" or "-n" adds this value algebraically to the current heading space setting, so long as the resulting value is not negative.

If the current top margin is tm, the heading margin is hm and the heading space is hs, the top of each page will appear as:

(1) tm - hm - hs blank lines,
(2) the hs heading lines,
(3) hm blank lines.

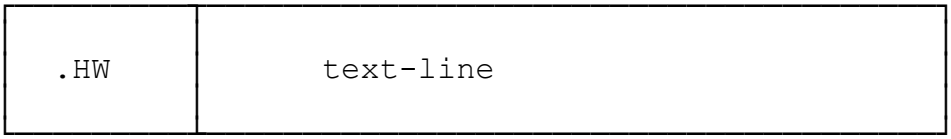
Defaults:

This command creates a break and until it is encountered n=1 will be in effect. The value of the operand may be from zero to the HSFSOVER option, normally nine. If the operand is omitted n=1 will be assumed.

Notes:

(1) The HEADING MARGIN plus HEADING SPACE must be less than or equal to the TOP MARGIN.
(2) See the description of Top Title (.TT) for notes on the interrelation of Top and Bottom Titles.

The HYPHENATE WORD control word allows a user to specify text with conditional hyphenation break points.



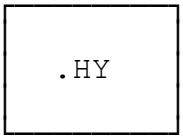
This facility allows a word or line of text to be formatted with conditional hyphens. If the "text-line" will fit on the current line then it will be printed without hyphens. Otherwise, SCRIPT will attempt to use one of the indicated hyphens as a hyphenation point and the others will not be printed. Automatic Hyphenation may be ON or OFF at the time this control word is encountered. The hyphenation exception dictionary is not affected by this. The "text-line" operand starts one blank after the control word. A compound word such as "brother-in-law" must be entered with double hyphens, indicating that a single hyphen must remain. A hyphen is only considered to be a hyphen if preceeded and followed by an alphabetic character.

Defaults:

This control word does not create a break. If the operand is omitted, no action is performed.

Examples:

```
(1) .HW I need a pter-o-phyl-lum sca-la-re
    .HW and a le-bis-tes
    .HW re-tic-u-la-tus.

(2) .HW It's for my mo-ther--in--law.
```



The HYPHENATION control word is used to set the level of hyphenation required and to manipulate the automatic hyphenation exception word list.

.HY	<ON|USER|OFF> <SET THRESH <5|n|+n|-n>> <SET MINPT <3|n|+n|-n>> <SET ENDPT <3|n|+n|-n>> <SET SUP <3|n|+n|-n>> <SUP> <ADD|DELETE|CHANGE> word-with-breaks <TEST word-without-breaks|DUMP|PURGE>

The .HY control word is used to set the desired level of hyphenation. The "OFF" operand says that no hyphenation at all is desired and that only blanks between words are to be considered for the purpose of concatenating text.

The "USER" operand is one level higher in that words containing hyphens, such as "brother-in-law", may be broken after a "-" when concatenating text. Secondly, the "USER" operand allows an input line to end with a hyphen treating that as a conditional hyphen, so that the hyphen will remain if needed as a break point within a word at the end of an output line, but will be discarded if that point of the word appears within a formatted output line.

The "ON" operand enables automatic hyphenation of words at the end of formatted lines by employing some algorithms for English words with an exception dictionary facility.

The "SET THRESH" defines the hyphenation threshold. At least this number of spaces must remain at the end of a line to be formatted before hyphenation will be attempted. The value must be positive and has an initial and default value of 5.

The "SET MINPT" defines the first hyphenation point beyond the beginning of a word to be considered for hyphenation. The initial and default value of 3 means at least three characters at the start of a word will be kept together.

The "SET ENDPT" defines the minimum number of characters at the end of a word to be considered for hyphenation. The initial and default value of 3 means at least three characters at the end of a word will be kept together.

.HY

The "SET SUP" defines the maximum number of consecutive output lines that will be eligible for automatic hyphenation. After "n" successfully hyphenated lines on the same page, the next output line will not be hyphenated.

For all of the Hyphenation "SET" options, a value of zero means its largest possible value.

The "SUP" operand suppresses automatic hyphenation to the "OFF" level until the next break occurs. This is useful to effectively turn hyphenation off until the end of a paragraph, at which point the current setting is restored.

The "ADD" operand is used to insert a user word into the hyphenation exception word dictionary. This is necessary to distinguish between words such as the noun "pres-ent" and the verb "pre-sent". A word may be specified without break points such as "WATFIV" to prevent it from being broken across output lines. A word already in the dictionary may be added again as these duplicates are entered in a last in, first out order.

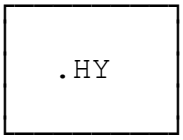
The "CHANGE" operand operates as "ADD" if the alphabetic characters of the word operand do not match any current entry. If the characters do match an entry, that entry is replaced with its new break points.

To "DELETE" a word, the letters of the word and its break points must match.

To find all the hyphen break points of a word use the "TEST" operand. SCRIPT will reflect the input word and all of its break points on SYSTERM.

The "PURGE" operand is used to delete all words in the current exception dictionary.

The exception word dictionary is also used because not all English words hyphenate correctly when the algorithmic rules are applied. A master list of "common" exception words is distributed with this version of SCRIPT and is normally accessible via ".IM□SYSHYPH". This master exception dictionary is never brought in automatically.
Beware -- each exception word requires sixteen bytes of storage and the 2733 exception words in that list require some 44K of main storage. When running with multiple passes, care should be taken to add a large exception dictionary only on the first pass as the exception dictionary is not cleared between passes.



The "DUMP" operand may be used to list all words in the current exception dictionary on the SYSTERM error file. Note that the words do not list alphabetically but in order by frequency of letter occurrence in English.

Defaults:

The HYPHENATION control word does not cause a break. There is no default operand(s). The initial setting of the hyphenation level is "USER" with no entries in the exception word dictionary.

Examples:

(1) .HY ADD Assembler
Never hyphenate that dying word "assembler".

(2) .HY ADD pres-ent
A happy birthday present.

(3) .HY CHANGE pre-sent
The University of Waterloo has presented ...

(4) .HY off;.* I'd rather do it myself.

Notes:

(1) The automatic hyphenation rules employed have been taken from a program package called HYPHENATION/360. See the IBM Application Description Manual, form E20-0257, for a clear description in ten pages of the algorithms employed.

.H0

The HEAD LEVEL "n" control word adds a Heading to the text and/or to the current Table of Contents.

.H0 to .H9	line

The .H0 to .H9 control words are short form commands for Heading Level (.HL) with an operand of 0 to 9.

See Heading Level (.HL) for all the information.

Defaults:

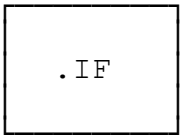
This control word causes a break.

Examples:

(1) .h1 This is a level one heading

is equivalent to:

.hl 1 This is a level one heading



The IF control word causes the next input line to be processed or not depending on the result of a comparison.

.IF	S1 "op" S2 <line> n1 "op" n2 <line>

Where:

S1 and S2 are undelimited character strings, starting with an alphabetic and not containing blanks or delimited character strings that contain blanks,
n1 and n2 are decimal integers or expressions containing no blanks, and
"op" is one of the following delimited before and after by at least one blank:

=	or	EQ	-	equal
¬=	or	NE	-	not equal
<	or	LT	-	less than
>	or	GT	-	greater than
<=	or	LE	-	less than or equal
>=	or	GE	-	greater than or equal

<line> is a command or text line starting one blank after the second operand.

If the comparison operation is true, the <line> operand is processed normally. If the <line> operand is omitted then the next logical input line is processed normally. Otherwise, the next input line is skipped (ignored). The next logical input line is up to the first following control word separator (see .CW) or the next physical input record, whichever occurs first.

As an alternative to the logical record immediately following being used in the "true" case and ignored in the "false" case the then (.TH) and else (.EL) control words may be used instead. In this second mode the .TH case is executed if the .IF is "true" and the .EL case if the .IF is "false". A series of .IFs may be nested up to ten levels by making an .IF the object of a .TH or an .EL. See descriptions of THEN and ELSE for more information.

If the first operand begins with a delimiter, normally "'", or with an alphabetic, the comparison will be between two character strings using the standard EBCDIC collating

.IF

sequence.

If the first operand begins with "+", "-", "(" or a numeric, the comparison will be between two signed decimal integers.

Multiple comparisons may be done in one statement by using one of the following operations between comparisons. Evaluation of multiple comparisons is done left to right.

AND or &, OR or |

Defaults:

This command does not create a break when encountered.

Notes:

(1) If the two character strings to be compared have unequal lengths, then the comparison will be done from the start of both strings with the minimum length of the two. If that is equal then the shorter string is considered to be the smaller.

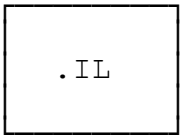
Examples:

(1) .sr a % / 2 * 2 - %
.ur .if &a = 0
Will cause the next input line to be processed if the current page number is even.

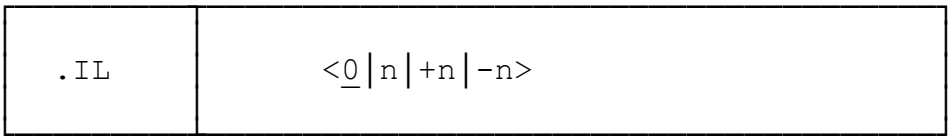
(2) .if 'A' eq A
This will always be a true character comparison.

(3) .if 9 EQ +9
This will always be a true numeric comparison.

(4) .if &number = 'roman' or &number = 'ROMAN'
.th .pn roman;.el .pn arabic
This will set subsequent page numbering to roman numerals or arabic numerals depending on the value of variable "&number".



The INDENT LINE control word forces the next output line to start a specified number of columns to the right of the current input column.



The .IL control word causes the next output line to begin "n" spaces to the right. A relative operand alters an immediately preceeding INDENT LINE or UNDENT.

Defaults:

This command will create a break. If the operand is omitted then 0 will be assumed.

Examples:

(1) .in +5;.il +3;This is a first text line of text on two output lines.

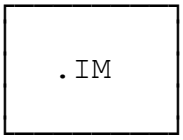
produces:

This is a first text line of text on two output lines.

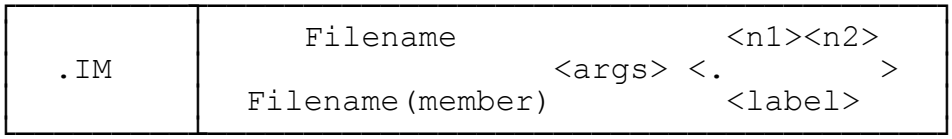
(2) .in +5;.il -3;This is a second test of a line of test on two output lines.

produces:

This is a second text of a line of text on two output lines.



The IMBED control word is used to insert the contents of a specified file into the printout of another file.



Where:

<args> is a list of "n" elements separated by blanks or delimited strings separated by blanks such that special set reference symbol &1 is set to the value of the first, &2 to the second, &n to the last and &0 to "n" even if "n" is zero.

Alternatively, a keyword parameter may be set in this list of arguments if it contains an equal sign, no undelimited blanks and is a valid .SR assignment statement. If keyword parameters are used then &0 will be set, even if zero.

The special reference symbol &* is set to the string of all positional and keyword arguments.

When the default LOCAL option is specified, all of these reference symbols are defined as local variables and may only be used within the file being imbedded.

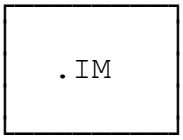
. is the current control word indicator character, normally period ".".

n1 specifies the start of the range of records to be included.

n2 specifies the end of the range of records to be included.

label specifies the label which marks the first record to be included.

The first argument names a text file whose contents are to be included in the input. Each "Filename" that has not been used in a previous IMBED or APPEND Control word must have an associated "DD Statement" or "TSO FILE ALLOC" that references a dataset containing valid input records. If the



"DD Statement" is missing SCRIPT will write an error message before closing all opened files.

"n1" gives the item number of the first line of the imbedded file to be read, "n2" gives the item number of the last line to be read. Instead of "n1 n2", the single character "*" may be used to indicate that the entire file is to be included. The "*" may also be used instead of "n2" to indicate that the rest of the file (that is, records "n1" through the end) are to be included. "label" may be specified instead of "n1" and "n2". This specifies that the first record to be included in the file is a Label (.LB) control word with a matching "label".

The .IM and .AP control words perform similar functions, but .IM allows the contents of a second file to be inserted into the printout of an existing file rather than appended to it. Imbedding may be used to insert standard sets of control words at desired spots, or to control formatting of a long document out of individual files.

The number of levels to which files may be imbedded is limited by the amount of storage available to contain control blocks and buffers. Said in a different way: a file may imbed another file or itself, and then that file may imbed another file or itself, and then that file ... this can continue until the stack of imbedded files becomes unreasonable and all available storage contains imbedded file control blocks.

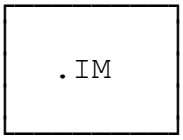
Defaults:

This command word does not create a break when encountered. The entire file is read in starting at the first line, i.e., * is assumed.

Examples:

(a) .IM CHAPTER4
The contents of the file named CHAPTER4 will be included in the input. When the end of CHAPTER4 is reached, processing of the current file will continue.

(b) .IM SYSCONS . 2 *
All but the first line of SYSCONS will be imbedded. Note that SYSCONS is the Filename used by ".te" and ".rd". This can be used to give added flexibility to the SCRIPT file. For example, suppose there are



several ".te's" within the file and it is executed both on-line and off-line. Because ".te" always starts reading at the first line the following technique can be used.

```
.sr test 1
.ty If output is off-line, enter ".sr test 0",
.ty otherwise nothing.
.te
.ty Enter Date of this manual.
.ur .if &test = 0
.th .im control . 2 2
.el .te
```

SYSCONS would reference a dataset if the output was directed off-line, and the first line of that dataset would be ".sr test 0". If the output is directed to the offline printer a "DD Statement" similar to the one below would have to be included in the job setup for input of this file.

```
//SYSCONS  DD  DSNAME=pref.PS,DISP=SHR
```

If SYSCONS was a member of a partitioned dataset then the following "DD statement" would be necessary:

```
//SYSCONS DD DSNAME=pref.PO(SYSCONS),DISP=SHR
```

(c) .IM FOOTNOTE one 'one plus one' . 12 15
Lines 12, 13, 14 and 15 of FOOTNOTE will be imbedded with &1 set to "one", &2 set to "one plus one" and &0 set to "2". The value of "&*" is set to "one 'one plus one'".

(d) .IM ERROR 'ERROR MESSAGE TEXT' severity=4
File ERROR will be imbedded with the message text in the &1 variable. Variable &0 is set to one and keyword variable &SEVERITY is set to the value four.

The INDENT control word causes the logical left and/or right margins of the printout to be indented a specified number of spaces.

.IN	<0|m|+m|-m|*> <0|n|+n|-n|*>

The .IN control word causes printout to be indented "m" spaces from the left margin and "n" spaces from the right margin. This indentation remains in effect for all subsequent lines‘½’ until another .IN is encountered. An operand of the form "+m" adds the value m to the current indent setting. An operand of the form "-m" subtracts the value m from the current indent setting or makes it zero if this is negative. An operand of the form "*" leaves the current left indent value unchanged.

The second operand specifies a relative change to be applied to the current line length. ".IN * 0-5" will place the right margin for formatted text five spaces to the left of the current line length. A second operand of the form "*" leaves the current right indent value unchanged.

".IN 0 0" or ".IN" will cancel the indentation and cause printout to be formatted between the original left margin and the current line length.

Defaults:

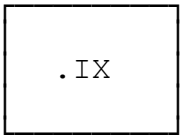
When this control word is encountered it creates a break and until then "m" = 0 is in effect. When the operand is omitted "m" = 0 is assumed.

Notes:

(1) The .IN request causes any offset (.OF) setting or undent (.UN) setting to be cleared unless a first operand of "*" is specified.

---

(15) Including new paragraphs, new footnotes and new pages. It remains in effect even if ".FO NO" is specified.



The INDEX control word builds up to 9 index structures with references, or causes a specified index structure to be printed (or purged).

.IX	<1|n> 'S1' <'S2' <'S3'>> <<.> <ref>>
	<1|n> . <DUMP|PURGE>

BUILDING THE INDEX. In its first form, the .IX control word adds up to three levels of index entries and a reference entry to an index structure. If the reference entry is omitted, the current page number will be used as the reference entry (see Example 1 below).

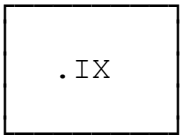
If a reference entry other than the current page number is desired and S3 (or S2 and S3) are to be omitted, then the control word indicator (normally period) must be placed between the index level(s) and the reference operand. If all four operands are present, the use of the control word indicator as a separator is optional. Use of the control word indicator means a reference must follow, even if null (see Example 4 below).

PRINTING THE INDEX STRUCTURE. In its second form, the .IX control word with the "DUMP" operand causes the specified index structure to be printed. Or, "PURGE" may be specified to delete the specified index struture. The index structure will be printed in alpabetic order within levels. For purposes of alphabetizing the level entries, lower- and upper-case are considered to be identical.

Four REMOTEs must be defined to control how the index levels are to be printed. These REMOTEs must be defined with the "SAVE NOSAVE" attributes (see Example 4 below).

(1) Remote SYSIX0 is signalled when the first character of a level one index changes. It is signalled with one operand, so that variable "&1" may be used to access what character is now starting.
(2) Remote SYSIX1 is signalled before a level one index.
(3) Remote SYSIX2 is signalled before a level two index.
(4) Remote SYSIX3 is signalled before a level three index.

The reference variable "&SYSIXREF." is placed between an index level and its reference data if reference data is present. The initial value for this system reference variable is ',□' (comma-blank) but may be redefined by the



user. Therefore substitution (.SU) must be turned on when printing the index.

Defaults:

This control word does not create a break. The index number "n" is optional and defaults to one. No index level operands will be assumed. A reference of the current page number will be assumed.

Examples:

(1) .ix 'level1' 'level2'
adds a first and a second level entry and the current page number as the reference entry, to index structre 1.

(2) .ix 2 'level1' . 'see...'
adds a first level entry and the reference string "see..." to index structure 2.

(3) .ix 'level1' . ''
adds a first level entry and a null reference entry to index structure 1.

(4) The following shows how index structure 1 can be printed.

```
.rm SYSIX0 save nosave;.* <----------
.in;.sp 2;.cp 7
.bx 2 6
   &1
.bx off
.rm
.rm SYSIX1 save nosave;.* <----------
.in;.cp 3;.of 1
.rm
.rm SYSIX2 save nosave;.* <----------
.in 3;.cp 2;.of 1
.rm
.rm SYSIX3 save nosave;.* <----------
.in 6;.of 1
.rm
.* <------------------------------>
.se SYSIXREF=',□□';.co;.ju no;.su on
.ix . DUMP;.fo;.su off
.rm SYSIX0 delete;.rm SYSIX1 delete
.rm SYSIX2 delete;.rm SYSIX3 delete
```

The JUSTIFY control word causes output lines to be padded with extra blanks so that the right margin is justified.

.JU	<YES|NO| LEFT|RIGHT|CENTRE| INSIDE|OUTSIDE|HALF>

The .JU control word specifies that all subsequent output lines are to be formed by padding with extra blanks to cause the right margin to be justified.

An operand of "LEFT", "RIGHT", "CENTRE" or "CENTER" causes the output line to be left justified, right justified or centred. The "INSIDE" operand caused odd pages to be left justified and even pages to be right justified, and the opposite for "OUTSIDE". The "HALF" operand causes one half the number of blanks for full justification to be inserted. This produces an effect between right ragged and fully justified.

Defaults:

This command creates a break and is in effect until a ".JU□NO" is encountered. An omitted operand is treated as "YES". A "NO" operand is equivalent to ".NJ" (See .NJ control word).

Notes:

(1) Since JUSTIFY is the normal mode, this control word is used to cancel a previous JUSTIFY NO control word or the JUSTIFY NO part of a FORMAT NO control word.

(2) If a line exceeds the current line length, and CONCATENATE NO is in effect, the line is printed as is.

(3) This control word is seldom used without CONCATENATE. Specifying FORMAT combines the two functions. which combines the two functions.

The LEFT ADJUST control word causes the next input line to be left adjusted in the output line, as if under FORMAT NO.

.LA	<1|n|YES|NO|line>

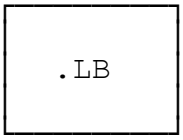
The "line" operand or the next "n" lines in the input file, including any leading blanks, will be left adjusted to the left indent margin in the output lines. The "line" operand starts one blank after the control word.

Defaults:

This command will create a break when encountered. A numeric operand will left adjust the following "n" input lines. A "YES" operand will left adjust all following input lines until a "NO" operand is encountered or until a CENTER or RIGHT ADJUST control.

Notes:

(1) The LEFT ADJUST control word is a way of specifying FORMAT NO control without changing the current FORMAT setting.



The LABEL control word specifies an identifier that is to be associated with the current input record.

.LB	<ident|n> <line>

The .LB control word defines a character identifier or number to be associated with the current input record. If the operand is an identifier, then it is converted to upper case before use. An identifier may consist of a maximum of eight characters. If the operand is a number it must be equal to the input record number. Thus ".LB five" would be valid anywhere but ".LB 5" would only be valid on the fifth record of any input file, macro or remote. The label operand must be unique within each input file, macro or remote.

The optional "line" starting one blank after the operand is interpreted normally after the label field is scanned.

The LABEL control word is normally used as a target by the ".GO" control word. It may also be used to verify that a statement is correctly placed in an input file.

Defaults:

This command does not create a break. However, control words within the "line" operand may create a break when the "line" is subsequently interpreted. If "line" is omitted, no other action is performed.

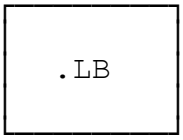
Notes:

(1) The LABEL ".LB" control word has an alias. Three occurrences in a row of the control word indicator character, normally "..." is equivalent. A blank is not required after the control word in this form. Thus, ".LB□label", "...□label" and "...label" are all equivalent.

Examples:

(1) .LB uow .sp 2

This defines a 'UOW' label on a space two statement.



(2) .LB 99 This is line ninety-nine.

This verifies that the line of text occurs in input line 99 of the current input file.

.LE

The LEADING BLANK LINE control word is used to control suppression of blank lines at the beginning of a page.

.LE	<YES|NO>

Blank lines may occasionally be the first lines of formatted output to be printed on a page. For example, a ".SP 10" may have appeared five lines from the bottom of the previous page. SCRIPT does not normally print such blank lines(16).

Since blank lines may sometimes be desirable at the beginning of a page(17) the .LE control is provided.

".LE YES" allows spaces to begin a page of output. ".LE NO" deletes spaces at the top of an otherwise empty page. This is true even if the spaces occur within a text block such as a Conditional Page (.CP) or Floating Keep (.FK).

Defaults:

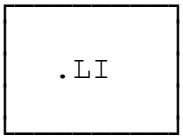
This command does not create a break and ".LE NO" is in effect until ".LE YES" or ".LE" is specified. If the operand is omitted, "YES" is assumed.

Notes:

(1) If the operand is omitted, "YES" is assumed.
(2) ".LE NO" is the normal mode.
(3) Leading spaces are allowed on the first page of printed output.
(4) This discussion applies only to formatted text, not to headings or top margins.

---

(16) Except at the top of the first page of output.
(17) For example, when centering a figure on a page.



The LITERAL control word allows the control word indicator in the first column to be ignored or changed on following lines.

.LI	<1|n|ON|OFF|char|line>

The .LI control word allows subsequent input lines to start with the control indicator and yet be treated as input text when the argument is omitted or numeric. If the single character of the argument is non-numeric, then that character replaces the initial period "." character as the control word indicator in column one of input lines.

The character asterisk "*" may not be used as the control word character. If asterisk is used, it means restore the default of period.

If the ".LI ON" is specified then it can only be terminated by ".LI OFF" starting in column one of an input record.

Defaults:

This command word will not create a break. A check for a control word indicator character in the following "n" lines is to be omitted. A non-numeric operand changes the control word indicator character from a period "." to the single character of the operand.

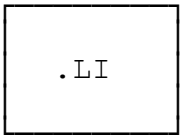
Notes:

(1) The LITERAL control would be used when a text line must begin with a period, such as when entering a number preceded by a decimal point or beginning a line with an ellipsis. Even then it would only be necessary to use LITERAL control if the line could not be reworded, rearranged or altered with a leading blank.

(2) A second control word indicator character may be set that has a slightly different interpretation. A command of the following form:

```
.LI , NOBREAK
```

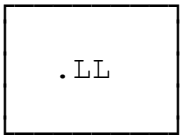
will define the comma "," to be the Nobreak Control



Word Indicator. Then the use of comma will suppress the normal "break" action of all control words. For example ",SP" will leave an immediate space without flushing the current bufferred text first and ",BR" would be a no-op. This facility can be very useful for changing indents or spacing within a paragraph without destroying the justification. But beware that the current partially buffered output line is not alterable and that not all "NOBREAK" combinations may produce valid combinations with other control words.

```
.LI * NOBREAK
```

can be used to cancel this facility.



The LINE LENGTH control word specifies the number of horizontal character positions which are to be printed in subsequent output lines.

.LL	<60|n|+n|-n>

The .LL control sets the length of subsequent output lines to "n" characters, where "n" is less than or equal to 144. An operand of the form "+n" or "-n" first adds this value algebraically to the current line length, so long as the resulting value is not negative.

Defaults:

When this control word is encountered it creates a break. Unless otherwise specified n=60 characters per line (including blanks) will be in effect. If the operand is omitted then the value of the LLength= parm will be taken, normally 60.

Notes:

(1) Practically all terminals and printers print 10 characters per horizontal inch. The default value of 60 is sufficient (when used in combination with ".AD 10") to give 1.5 inch left and 1 inch right margins.

The IMMEDIATE LINE control word causes output to resume at an absolute line number further down the current page or up on the next.

.LN	<n|+n|-n>

This control word will position the next line of user output to a specified line between the top and bottom margins. An operand of "1" is the first line following the Top Margin.

Normally the process involves spacing down on the current page. If the operand specified is higher on the page than the current line position, then a page eject will be done first followed by the required number of spaces. An operand of the form "+n" spaces down on the current page and an operand of the form "-n" spaces down on the next page.

When spacing, automatic remotes are not triggered. Leading spaces are allowed on a page whether or not the .LE control word has been specified.

This control word is considered illegal within a Footnote or a Keep.

Defaults:

This command does create a break when encountered. An operand must be specified as none will be assumed.

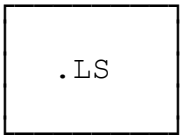
Examples:

(1) .LN 22;This is on Line 22 of the text area.

After causing a break, the output spaces and/or ejects to the next occurrence of Line twenty-two following the top margin.

(2) .LN -5

This ejects the current page and spaces to five lines higher than the former page position. This would cause an error if the former page position were not at least five lines down from the top margin.



The LINE SPACING control word causes zero or more blank lines to be skipped between each line of formatted output.

.LS	<0|n|+n|-n|YES|NO>

Subsequent output lines will have "n" blank lines inserted after each line of body text. The operand may be signed, in which case the current line space value is incremented or decremented. A "YES" operand implies one line or double spacing (See .DS). A "NO" operand implies zero lines or single spacing (See .SS).

Defaults:

This command does create a break. An omitted operand is treated as "Single Space".

Notes:

(1) Single spacing ".LS□0" is the normal mode.
(2) ".SP" control words encountered with ".LS□n" in effect will produce "n" times the normal number of blank lines.
(3) ".SP n A" control words will always produce "n" blank lines in the output.
(4) Footnote lines are not multi-spaced by default. This may be altered within the footnote.
(5) The operand to the ".CP" control word is not multiplied by "n" when ".LS□n" is in effect.

The MULTIPLE COLUMN control word restores multi-column output after it has been suppressed by .SC (SINGLE COLUMN).

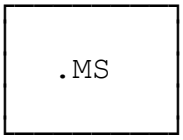
If SINGLE COLUMN had been previously specified, this control word will restore the former multiple column environment. If SINGLE COLUMN had not been previously specified, then a break will be caused and no other action.

Defaults:

This control word causes a break. It takes no operands.

Notes:

(1) This control word may not be specified within a Keep or Footnote.



The MACRO SUBSTITUTION control word prevents user-defined macros from being invoked.

.MS	<ON|OFF>

This control word interacts with user macros defined by .DM (Define Macro). An "ON" operand causes SCRIPT to compare the name of each defined macro with each control word encountered. An "OFF" operand will suppress this comparison.

Defaults:

This control word does not create a break. No default operand is assumed and "OFF" is the initial setting.

Notes:

(1) Even with Macro Substitution "OFF", the macro may still be invoked by using .SI (Signal).
(2) Macro Substitution only affects macros specified by .DM (Define Macro). Named Remotes are not affected.

.OB

The ODD BOTTOM control word is used to define three headings to be printed at the bottom of odd numbered pages.

.OB	<1|n> /S1/S2/S3/

Where the optional "n", from 1 to the value of the HSFSOVER option, gives the footing line number and S1, S2 and S3 are character strings not containing the delimiter character "/" which can be any character defined as the first character of the operand. Any of the fields may be omitted, but the delimiter character must be included to indicate missing fields, e.g., $S1$$S3$.

The .OB control word is used in the same way as the .BT control word. The footings defined with .OB will appear only on odd numbered[12] pages, however. The number of footing lines printed on an odd page are defined by .FS (Footing Space).

Defaults:

A break is not created by this command. Unless otherwise specified ".OB□////" will be in effect.

Notes:

(1) The ODD BOTTOM control has the same effect as the ODD FOOTING (.FD) control. The difference is that the field delimiter is self-defined by the first character of the operand.

---

(18) Odd numbered pages are those which have the binding to the left of the left-hand margin in a book.

The OFFSET control word causes all but the first line of a section to be indented.

.OF	<0|n|+n|-n>

The .OF control word is used to indent the left side of the printout "n" spaces after printing the next output line. The extra indentation remains in effect until another .HI, .IN or .OF control word is encountered. An operand of the form "+n" adds the value n to the current offset setting. An operand of the form "-n" subtracts the value n from the current offset setting.

The .OF control word may be used within a section which is also indented with the .IN control. Any subsequent .IN control word causes the indentation set by .OF to be cleared.

If it is desired to start a new section with the same offset as the previous section, it is necessary to repeat the .OF request.

Defaults:

This command does create a break when encountered. If no argument is supplied then n = 0 will be assumed.

Notes:

(1) If the following sequence is input:

```
.of m
<text>
.of n
<text>
```

The second offset will be added to the first if n is a signed integer. If n is unsigned, this addition will not occur. Phrased differently: if n is signed, all but the first line of the first section together with the first line of the second section will be indented m spaces and all but the first line of the second section will be indented m + n spaces.

(2) If HANGING INDENT is in effect, an OFFSET control word will override the next time the HANGING INDENT is to be used.

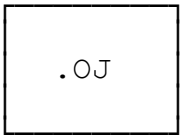
Examples:

(1) Many of the sections of this manual[11⅛] are produced by a sequence like this:

```
<section header>
.in 5
.of 4
(1) <text>
.of 4
(2) <text>
   .
   .
   .
.in
```

---

(19) Including the example text.



The OUT JUSTIFY control word causes the specified lines to be aligned to the outside of the paper.

.OJ	<1|n|YES|NO|line>

The "line" operand or the next "n" lines in the input file, including any leading blanks, will be adjusted to the outside indent margins. This means that .OJ performs like Left Adjust (.LA) on even numbered pages and like Right Adjust (.RA) on odd numbered pages. The "line" operand starts one blank after the control word.

Defaults:

This command will create a break when encountered. A numeric operand will out justify the following "n" input lines. A "YES" operand will outjustify all following input lines until a "NO" operand is encountered or until a Centre (.CE), Left Adjust (.LA) or Right Adjust (.RA) control.

The OUTPUT OVERLAY control word causes blanks in formatted output records to be overlaid with a user defined string.

.OO	<1|n|ON <string>> <OFF|DELETE>

The .OO control word is used to alter occurrences of blanks in the next "n" formatted output lines and spaces. Every occurrence of blank after the ADJUST (see .AD) column will be replaced by the corresponding character in the output overlay "string". The "string" is defined starting one blank after the first operand on the same line. If omitted, the "string" will be the next input record.

The first operand specifies the number of times the overlay operation is to be performed. An "ON" operand continues the process indefinitely. More than one Output Overlay "string" may be specified. These are printed in the order in which they are defined. An "OFF" operand terminates the first entry on the queue; a "0" operand defines an entry that is never used; a "DELETE" operand purges the entire queue.

Defaults:

This control word does not cause a break.

Examples:

(1) to facilitate margin headers:

```
.oo 1 Head One
.oo 1
Head Two
.in 20
This is some text that will have a margin header
defined in the left margin.  Those header lines
read "Head One" and "Head Two".
```

Produces:

```
Head One            This is some text that will have a
Head Two            margin header defined in the left
                    margin.  Those header lines read "Head
                    One" and "Head Two".
```

.OT

The ODD TITLE control word is used to define three headings to be printed at the top of odd numbered pages.

.OT	<1|n> /S1/S2/S3/

where the optional "n", from 1 to the value of the HSFSOVER option, gives the heading line number and S1, S2 and S3 are character strings not containing the delimiter character "/" which can be any character defined as the first character of the operand. Any of the fields may be omitted, but the delimiter character must be included to indicate missing fields, e.g., $S1$$S3$.

The .OT control word is used in the same way as the .TT control word. The headings defined with .OT will appear only on odd numbered(20) pages, however. The number of heading lines printed on an odd page are defined by .HS (Heading Space).

Defaults:

A break is not created by this command. Unless otherwise specified ".OT□///PAGE□%/" will be in effect.

Notes:

(1) The ODD TITLE control has the same effect as the ODD HEADING (.HD) control. The difference is that the field delimiter is self-defined by the first character of the operand.

---

(20) Odd numbered pages are those which have the binding to the left of the left-hand margin in a book.

The OVERLAY control word causes following input text lines to be overlaid by the contents of the text line following this command.

.OV	<1|n|YES <string>> <OFF|DELETE>

The .OV control word is used to alter occurrences of blanks in the next "n" input text lines. Every occurrence of blank in the text lines will be replaced by the corresponding character in the overlay operand line, up to the maximum length of the overlay operand line and the input text line. A "YES" operand continues the process indefinitely. A "NO" operand terminates the process. A "DELETE" operand purges the entire queue.

Multiple Input Overlay strings are queued. See Output Overlay (.OO) for more information.

Defaults:

This control word does not create a break.

Examples:

(1) To facilitate block building:

```
.fo no;.ov yes
        |       *
****************
*
*    Title
*
****************
.ov no;.fo yes
```

Produces:

```
****************
*       |      *
*    Title     *
*       |      *
****************
```

.PA

The PAGE control word causes an output page eject.

.PA	<%+1|n|+n|-n <m|+m|-m>> <YES|NO|ODD|EVEN>

When the .PA control word is encountered, the rest of the current page is skipped, any saved footnote lines are printed, the Footing Space lines are printed, and a new page is begin whose number is specified by the operand.

The "YES" operand is equivalent to the default. The "NO" operand suppresses the page eject. The "ODD" operand causes output to continue on the next odd numbered page. Similarly, "EVEN" continues on the next even page.

An operand of the form "n" will change the current page number to the value specified; "+n" will increment the current page number by "n"; "-n" will decrement the current page number by "n".

A second operand of the form "m" may optionally be specified. If used then SCRIPT will number and count pages in the form "n.m", incrementing "m" on each page. Thus a chapter or an update could be numbered 3, 3.1, 3.2, ... by specifying ".PA□3□0".

Defaults:

This command word does create a break when encountered. If the operand is omitted then the current page number plus one (%+1) is assumed.

Notes:

(1) If the STOP option was specified in the control line which invoked SCRIPT, output will cease at the end of the current page to allow the typist to insert the next page of paper.

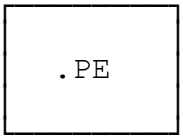
```
.PA
```

(2) If the current page is empty and ".EM NO" is in effect the .PA control has no effect other than causing the page number to be incremented or reset. Consequently, heading lines are not printed until there is a text line about to be printed. This allows dynamic changes to page headings and possibly remote lines imbedded in the text of that page.

(3) Heading lines, margins, etc., must be specified before the first line of text which will appear on the new page.

(4) When using page numbers with a decimal portion, the even or odd attribute is determined by summing the two portions of the page number. Thus 3.0 is odd and 3.1 is even, 4.5 is odd and 4.6 is even.

(5) If the current page number is "3.4" then a ".pa" control word would place the output at the top of page "4".



The PERFORM control word causes the remainder of the same or the next input line to be executed multiple times.

.PE	<1|n|YES|NO|DELETE>

The PERFORM control word provides a primitive facility for DO-LOOPs. The string following on the same input line is executed "n" times before passing on to the next physical record. A "YES" operand executes the line an infinite number (actually 32,767) of times. A "NO" or zero operand terminates processing of the current line immediately. Performs within performs are supported. A "DELETE" operand terminates all nested levels of performs.

Defaults:

This control word does not cause a break. An omitted operand will be treated as one.

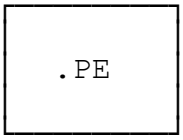
Examples:

(1) To build a block of twenty asterisk lines, centered on the page:

```
.pe 20;.ce;******************************
```

(2) Alternatively the operand string to be performed may be entered on the following input record:

```
.pe 20
.ce;******************************
```

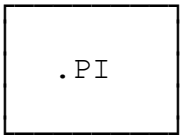


(3) To format the first 10000 numbers in paragraphs of 100:

```
...
.sr j=100;.cm paragraph count
.sr max=10000;.cm total number count
.rm 255 save nosave
.se i=&i+1
.se t=-&i/&j*&j+&i;.cm remainder mod j
.ur &i
.ur .if &t = 0
.sp
.rm
.pe yes;.si 255;.ur .if &i ge &max;.pe delete
...
```

(4) Same example as above using nested perform:

```
...
.sr j=100;.* numbers per paragraph count
.sr max=10000;.* total number count
.sr i=0;.* clear starting value
.ur .pe &max/&j;.sp;.ur .pe &j;.se i=&i+1;.ur &i
...
```



The PARAGRAPH INDENT control word specifies the number of blank spaces to be inserted at the beginning of each paragraph.

.PI	<0|n|+n|-n|YES|NO>

The ".PI" control word causes "n" spaces to be inserted at the beginning of the first line of each paragraph. The operand may be signed, in which case the current value is incremented or decremented appropriately. A "YES" argument will set the paragraph indent value to three. A "NO" argument is equivalent to a zero operand and terminates the paragraph indent function. The value of paragraph indent may never be negative or larger than the current line length ".LL" minus one.

The beginning of a paragraph is determined when:

(1) The line of text is not under .CE, .LA or .RA control.
(2) Text is being formatted with Concatenation on ".CO" or under Format control ".FO".
(3) The line of text does not start with a tab character.

Defaults:

This control word creates a break. An initial value of zero is assumed, which is also assumed if the operand is omitted.

Examples:

(1)
```
This is paragraph number one.  Note
that there is no indentation at the start
of this paragraph.
.pi 3
.sp
This is paragraph number two and it is
indented.
.sp
This is paragraph number three and it is
also indented because a paragraph indent
stays around until explicitly cleared.
.pi +3
.sp
```

.PI

```
This paragraph is indented even further
although the input started in column
number one.
.sp
.pi
Now paragraph indent is cleared.
```

Produces:

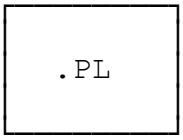
This is paragraph number one. Note that there is no indentation at the start of this paragraph.

   This is paragraph number two and it is indented.

   This is paragraph number three and it is also indented because a paragraph indent stays around until explicitly cleared.

      This paragraph is indented even further although the input started in column number one.

Now paragraph indent is cleared.



The PAGE LENGTH control word specifies the physical size of the output page in units of typewriter lines.

.PL	<66|n|+n|-n>

The .PL control word allows the use of various paper sizes for output, by setting the length of subsequent output pages to "n". An operand of the form "+n" or "-n" first adds this value algebraically to the current page length, so long as the resulting value is greater than the TOP MARGIN (.TM) plus BOTTOM MARGIN (.BM).

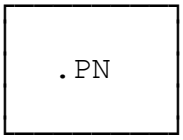
Normal 8.5" x 11" paper is 66 lines long on a IBM 1403 printer, an IBM 2741 type of terminal, or any other device which types 6 lines per vertical inch.

Defaults:

This command word will create a break and unless otherwise specified n=66 will be in effect. If the operand is omitted then the value of the PLength= parm will be taken, normally 66.

Notes:

(1) Use of the .PL control word for any other purpose than specifying the actual physical size of the output page is discouraged. The TOP MARGIN and BOTTOM MARGIN control words should be used to control the dimensions of printed text.



The PAGE NUMBER control word allows the user to control the incrementing and printing of page numbers.

.PN	<ON|OFF|OFFNO> <ARABIC|ROMAN <LOWER|UPPER>> <PREFIX|SUFFIX <string>> <FRAC|NORM>

ON Causes incrementing of page numbers to resume.

OFF Causes external page numbering to be discontinued.

OFFNO Suppresses incrementing of page numbers internally and externally.

ARABIC Causes page numbers produced in headings and footings to be printed in arabic numerals.

ROMAN Causes page numbers produced in headings and footings to be printed in LOWER or UPPER case roman numerals.

PREFIX Causes the Page Symbol character in titles to be prefixed with an up to eight character "string".

SUFFIX Causes the Page Symbol character in titles to be suffixed with an up to eight character "string".

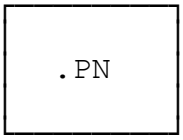
FRAC Causes fractional incrementing of page numbers to occur. The next page eject that goes from an even numbered page to an odd page will initiate page numbering in steps of ".1". So after page 4 will be pages 4.1, 4.2, ...

NORM If FRAC was last specified, a page eject occurs and normal integer numbering resumes. If FRAC was not in effect, nothing is done.

The .PN control word is used to control the automatic incrementing of page numbers. If OFFNO is specified, page numbers will not increase as subsequent pages are output. With OFF, external page numbering is not incremented, but internal page numbering continues. If ON is specified, internal and external incrementing will resume.

Defaults:

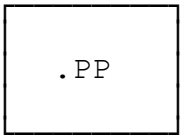
This command word will not create a break. Unless otherwise specified "ON" will be in effect. If an operand is missing then "ON" will be assumed.



Notes:

(1) If page incrementing is suppressed, the even and odd page force control words (".PA□ODD" and ".PA□EVEN") function exactly like .PA.

(2) The "OFF" and "OFFNO" operands will suppress the default top title of "PAGE□%". If the user has defined titles then these operands will replace the page symbol character with null.



The PARAGRAPH START control word defines the beginning of a new paragraph.

.PP	<line>

This command performs actions typical at the start of a new paragraph. This includes leaving a blank line, testing for bottom of current page and causing an indent on the next output line.

The "line" operand starts one blank after the control word. If the operand is missing, the text for the start of the paragraph comes from the next input text line.

Defaults:

This control word creates a break.

Examples:

(1) .pp Line of text ...

is equivalent to:

```
.sk
.cc 2
.il +3
Line of text ...
```

(2) .pp First line of text;Second line of text.

is equivalent to:

```
.pp
First line of text
Second line of text.
```

because the semicolon is the control word separator.

.PS

The PAGE NUMBER SYMBOL control word defines the character in headings and footings to be replaced by the current page number.

.PS	<%|character>

The percent sign "%" is usually reserved for use as the page number symbol. SCRIPT substitutes the current page number for each occurrence of the percent sign within a heading or footing line. To allow the percent sign to appear in titles the .PS control word is supported to change the character to be replaced by the current page number.

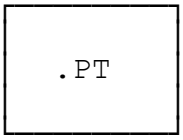
Defaults:

This command will not create a break when encountered. The initial default PAGE NUMBER SYMBOL character is the percent sign "%". Any character may be used as an argument. If the argument is omitted then the percent sign is restored as the default character.

Notes:

(1) The character specified in the PAGE NUMBER SYMBOL control word takes effect for all subsequent headings and footings. This means heading and footing specifications may have to be respecified at the same time.

(2) In the Set Reference (.SR) control word, the percent sign "%" or the ampersand "&" or the .PS argument will all be recognized as the current page number symbol.



The PUT TABLE OF CONTENTS control word adds an input line to the Table of Contents.

The .PT control word supplements the ".HL" (Head Level) control word for adding text and control words to the current Table of Contents.

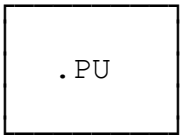
The "line" operand starts one blank after the end of the control word. In this way leading blanks may be part of the text added. If the "line" operand starts with the control word indicator, normally ".", the operand line is entered as is. If the "line" operand does not start with the control word indicator it is taken to be a text line and the current page number is appended to the "line".

Defaults:

This control word does not create a break. A command operand could create a break when the Table of Contents is formatted with the ".TC" control word.

Notes:

(1) A .PT control word used within a Keep Block or Footnote does not add the "line" operand to the Table of Contents until the Keep Block or Footnote is actually printed. This ensures the ordering and page numbers in the Table of Contents will always be correct.



The PUT WORKFILE control word allows the user to output records of data to an output file. This file may be imbedded later or be kept.

.PU	<1|n> <line>

With this control word, records of data may be written to output utility files. The first operand may range from one to nine. The first occurrence of .PU control word for the file causes the file to be opened. If the "line" operand is omitted then the output file will be closed.

If the file is defined with a disposition of MOD then output records will be added to the end of the file. If the file is not defined with a disposition of MOD then output records will replace the file.

An IMBED (.IM) or APPEND (.AP) with a numeric filename will close and input a workfile created with .PU.

Defaults:

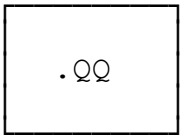
This control word does not create a break. If the file number is not specified then file one is assumed.

Notes:

In OS batch mode, workfiles must be allocated with a DDNAME of "SYSUSR0n" where "n" ranges from 1 to 9.

In CMS, workfiles are allocated for you with a "fileid" of "SYSUSR0n SCRIPT" but this may be overridden with your own FILEDEF with the PERM option before SCRIPT is invoked or with a FILEDEF in a SYSTEM (.SY) control word within the SCRIPT file.

The default file attributes are RECFM=VB, LRECL=136, BLKSIZE=800. A fixed file may also be created, in which case the defaults are RECFM=FB, LRECL=80, BLKSIZE=800.



The QUIT QUICKLY control word causes immediate termination of all input and output file processing.

.QQ	<YES|NO>

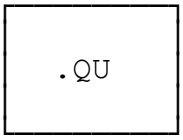
YES Allows the QUIT QUICKLY function to operate.

NO Suppresses the QUIT QUICKLY function.

When SCRIPT encounters a QUIT QUICKLY control word with an affirmative operand, even in a Remote Sequence or in a file that is being Imbedded, the processing terminates for the current pass. If multiple passes have been specified, these will be done normally.

Defaults:

This command creates a break. An omitted operand is treated as "YES".



The QUIT control causes immediate termination of all input file processing.

.QU	<YES|NO>

YES Allows the QUIT function to operate.

NO Suppresses the QUIT function.

When SCRIPT encounters a QUIT control word with an affirmative operand, even in a Remote Sequence or in a file that is being Imbedded, the output advances to the top of the next page and clears any stacked footnotes or text created by ".CP or .FK begin/end" sequences before termination of all processing for the current pass.

Defaults:

This command creates a break. An omitted operand is treated as "YES".

Notes:

(1) The .QT control word is supported for compatibiblity with QUIT in former versions of SCRIPT. The two functions are identical.

The RIGHT ADJUST control word causes the next input line to be right adjusted in the output line.

.RA	<1|n|YES|NO|line>

The "line" operand or the next "n" input text lines, including any leading or trailing blanks, will be right adjusted in the output lines.

Defaults:

This control word will create a break when encountered. A numeric operand will right adjust the following "n" input lines. A "YES" operand will right adjust all following input lines until a "NO" operand is encountered or until a CENTER control.

The "line" operand starts one blank after the control word.

Notes:

(1) If the next line is longer than the current line length, it will be truncated.

(2) The .RI control is provided for compatibility with RIGHT ADJUST in other versions of SCRIPT. The two functions are identical.

Examples:

(1) .ra (1.5)

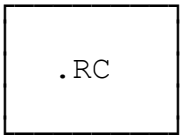
Produces:

(1.5)

(2) .ur .ra &sysdate

Produces:

July 3, 2012



The REVISION CODE control word allows selected portions of the total document to be marked in the left margin with a settable revision code character.

.RC	n <ON|OFF|ON/OFF> n <char|SET string>

n represents a revision code number and may be any numeric from 1 to 99.

ON specifies following text records are part of revision section "n".

OFF specifies that text for revision section "n" has ended.

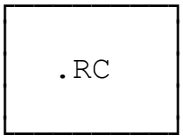
ON/OFF specifies that only the first following text record is part of revision section "n".

char specifies the character to be printed in the left margin for revision section "n".

SET string specifies the character string to be printed in the left margin for revision section "n". The "string" may be up to eight characters long and may be delimited if it contains blanks.

The REVISION CODE control word sets the revision character and the input text to be associated with each revision level. When a document contains multiple revision levels and the modifications overlap, it is important that revision codes be turned off in the reverse order that they were turned on. Put another way, only the current active revision code may be turned off.

Space is made for revision characters in the left margin by shifting all output text up to two spaces to the right. The revision character will be printed one blank before the text area. Even text under indent control will be shifted. However, if sufficient left margin space is available because of ADJUST or CENTER parameters, no right shifting is done. When specifying revision "string"s it is the user's responsibility to set the ADJUST value large enough to allow the "string" to be inserted. The "string" is right



adjusted, one blank before the text area and will be truncated on the left if insufficient space is available. If no such parameters are specified you should define the revision code characters before any text is formatted or immediately after a page eject so that entire pages will be indented uniformly.

Defaults:

This command does not create a break when encountered. Operand one must be present; if operand two is omitted then the revision code character is set to blank if the current revision code character is defined. Similarly, if the operand after "SET" is omitted then the revision code is set to blank if the current revision code is defined.

Notes:

(1) A Revision Code "string" cannot contain tab or backspace characters.

Examples:

(1)
```
.rc 3 on
text3...
.rc 1 on
text1...
.rc 2 on/off;text2...
text1...
.rc 1 off
text3...
.rc 3 off
```

(2)
```
.rc 1 'Rev. 1'
.rc 2 'Rev. 2'
This text is unrevised.
.rc 1 on
This text is from the first revision.
.rc 2 on/off
This text is from the second revision.
And this is the last line from the first revision.
.rc 1 off
This text is again unrevised.
```

The READ TERMINAL control word allows the user to type a line on the output page during SCRIPT output.

.RD	<1|n>

When the .RD control word is encountered during output to the user's terminal, SCRIPT will spin the typeball and unlock the keyboard until "n" carriage returns have been typed. The line typed is ignored completely except for purposes of counting lines on the current page.

When the .RD control word is encountered during output to the offline printer, SCRIPT will insert n blank lines in the output.

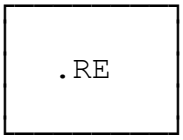
This control word may be useful to allow addresses to be inserted in form letters or to allow the user to change typeballs.

Defaults:

This control word creates a break. If the operand is omitted "n=1" is assumed.

Notes:

(1) If output is offline and the .RD is received within n lines of the bottom margin and ".LE NO" is in effect, spaces will not appear at the top of the next page. If however, output is online and the other two conditions are met, spaces will appear at the top of the next page.



The RESTORE STATUS control word restores the page environment to the last status saved with a SAVE STATUS.

The RESTORE STATUS control word resets control word values saved in a push down stack by the preceding SAVE STATUS control word. An error message will result if there is no corresponding SAVE STATUS control word preceding.

Defaults:

This command causes a break.

Examples:

(1) An example of SAVE and RESTORE status is shown in the following Remote example:

```
 ...
.rm 7 NOSAVE
.sa;.cm This saves the current page environment
.in 10
.tb 10 15
.ls 0
.ce
-FIGURE ONE-
Text ... text   text ...
.pa
.re;.cm This restores the current page environment
.rm
 ...
```

The REMOTE control word allows the user to save one or more input lines, which will be automatically or under user control imbedded at a specific place on the current page, the next page, or subsequent pages.

.RM	n <m|SAVE|NOSAVE> <SAVE|NOSAVE> n DELETE DELETE

Where "n" is:

-- an asterisk "*"
-- a positive integer from one to PAGE LENGTH
-- a positive integer greater than PAGE LENGTH and less than 32767.
-- an identifier of eight characters or less.

The input lines between the first .RM and the next .RM are saved. When the next line "n" is to be printed or a SIGNAL REMOTE ".SI□n" is encountered, the saved lines are automatically interpreted.

If NOSAVE is specified as a second operand or assumed by default, the saved lines are erased after the first use. If SAVE is specified, the remote sequence is saved and invoked on line "n" of every page until a ".RM□n□DELETE" is received specifying the same line number. If a number "m" is specified then the remote will be deleted following its "m"th invocation. A first operand of DELETE will remove all current saved remote sequences. This is also done automatically between passes.

If a second operand is specified then SAVE or NOSAVE may be specified as a third operand. The SAVE third operand, which is default, saves the current page formatting options and sets default page formatting options when the remote is entered and then restores these options at the end of remote processing following an implied break. The NOSAVE third operand prevents current page formatting options from being maintained through the remote processing and then restored at the end of the Remote. This is most useful when used with SIGNAL REMOTE ".SI", allowing it to be used like a Local Imbed.

The .RM control word is useful for defining multi-line headers or for causing automatic insertion of figures.

.RM

A numbered Remote with a number greater than PAGE LENGTH may only be triggered with a SIGNAL (.SI) control word.

A named Remote may be triggered with a SIGNAL (.SI) or with a user control word of the same name. The user may intercept native control words by defining a named remote with the name of a native control word.

Defaults:

This command word will not create a break when encountered. The first operand, the name or number of the remote, must be specified. If the first operand is an asterisk "*" then the value of "n" will be set to the TOP MARGIN plus one, i.e. (n = TM + 1).

Notes:

(1) Any lines whatever may appear within the .RM (except another REMOTE ".RM") and are interpreted when the remote sequence is inserted.
(2) Since Remote lines are saved away without interpretation or examination, it is necessary that the terminating ".RM" control word start in column one of an input line.
(3) Remote sequences are "triggered" when line "n" of the output page is about to be printed. While a remote sequence is being used for input, no other remote may be automatically triggered. If two or more remotes specify the same remote name they are ordered in a last in, first out order so that only the newest will be triggered. For numbered remotes, they are queued first in, first out order so that the oldest will be triggered first.
(4) A ".RM n DELETE" deletes the first sequence to be selected for the specified remote identifier.
(5) A user may write his own versions of SCRIPT control words in terms of other control words or do more checking before issuing the intended command by naming a remote with a control word. Note that a control word starting with two control word indicator characters will bypass the initial search of named remotes.

Examples:

(1) Assume a TOP MARGIN of 6. The following sequence places a figure at the top of the next page:


```
.rm 7
.in
   (Figure)
.ce yes
FIGURE ONE: Water Concentration in Lake Erie
as a Function of GNP.
.ce no
.rm
```

(2) The following remote would define a user control word for new page:

```
.rm $PA
.sp 2
.cp 5
.se $PACOUNT=$PACOUNT+1
.rm
 ...
.$PA;.* this triggers the above remote
.SI $PA;.* this does the same
```

(3) The following remote would trap page ejects and merely leave two spaces instead.

```
.rm PA save nosave
.sp 2
.rm
```

(4) The following remote would trap spaces and reduce them to single spaces.

```
.rm SP save nosave
..sp 1
.rm
```

The READ VARIABLE control word allows the user to set the value of a variable symbol from an online terminal.

.RV	name

The .RV control word verifies that the "name" operand is a valid reference name. If it is then one line of data is read from the terminal and this line is used as the right hand operand of a SET REFERENCE (.SR) control line.

No message is displayed at the terminal indicating a READ VARIABLE is being done. This may be done with a TYPE ON TERMINAL (.TY) control word preceeding the .RV.

Defaults:

This command word does not create a break.

Examples:

(1) .rv TEST

If 'NONE' were entered at the terminal then this would be equivalent to:

.sr TEST='NONE'

(2) .rv array(1)

If "3*99/5" were entered at the terminal then this would be equivalent to:

.sr array(1)=3*99/5

```
.SA
```

The SAVE STATUS control word saves the current page environment in a push down stack to be restored later with a RESTORE STATUS.

```
.SA
```

In order to design a modular SCRIPT input file, it may be necessary to alter one or more formatting characteristics for a section or chapter. For example, a new absolute INDENT value may be desired without knowing the value of the existing INDENT and that value must be restored before returning control. The SAVE STATUS control word will save the current page environment in a push down stack of up to nine levels, to be popped up later with a RESTORE STATUS. Note that SAVE STATUS does not change any current values.

The status of the following control words are saved:

.AD Adjust	.LA Left Adjust
.BC Balance Columns	.LE Leading Space
.BM Bottom Margin	.LI Literal
.CD Column Definition	.LL Line Length
.CE Centre	.LS Line Spacing
.CL Column Length	.OC Overlay Character
.CO Concatenate	.OF Offset
.DA Dark Output	.PI Paragraph Indent
.DS Double Space	.PL Page Length
.FM Footing Margin	.PN Page Numbering
.FO Format	.RA Right Adjust
.FS Footing Space	.SS Single Space
.HI Hanging Indent	.SU Substitute Mode
.HM Heading Margin	.TB Tab Setting
.HS Heading Space	.TI Translate on Input
.HY Hyphenation Set	.TM Top Margin
.IL Indent Line	.TR Translate
.IN Indent	.UN Undent
.JU Justify	

Defaults:

This command does not cause a break.

The SINGLE COLUMN control word temporarily suppresses multiple column output.

When in Multiple Column mode, this control word may be used to temporarily revert to single column output. This may be useful for defining a single column figure in the middle of text. The text to that point will be formatted according to the current COLUMN DEFINITION and BALANCE COLUMN control words. The Line Length (.LL) is restored before proceeding. Multiple Column output is restored by the next MULTIPLE COLUMN (.MC) or COLUMN DEFINITION (.CD) control word.

If MULTIPLE COLUMN is not in effect when this control word is encountered, a break will be caused, and no other action.

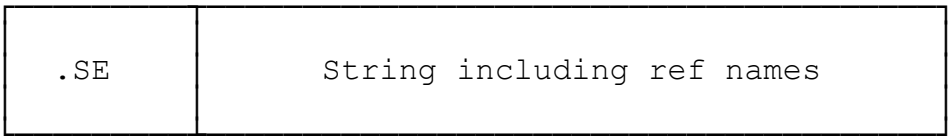
Defaults:

This control word creates a break. It takes no operands.

Notes:

(1) This control word is illegal within a Keep or Footnote.

This alternate SET REFERENCE control word allows the user to assign a character or numeric value to a symbolic reference name using other symbolic reference names.



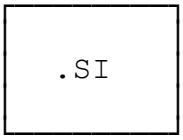
The operand of the SET REFERENCE control word will be reformatted by substitution with current values of reference names before being interpreted as an ".SR" control word. The substitution will repeat according to the ".UR" control word as often as is required to remove all reference names from the operand string.

Defaults:

This control word will not create a break when encountered. See the ".SR" control word for rules about the operands after substitution.

Examples:

```
(1) .se i=0;.cm This makes &i zero.
    .se i=&i+1;.cm This makes &i one.

(2) .se c='&&c';.cm This makes &c "&c".
    .se c='&c';.cm This is an infinite substitution.

(3) .se alpha='ABCDEFGHIJKLMNOPQRSTUVWXYZ'
    .se Lalpha=L'&alpha;.cm Assign length of 26
    .se Letter(1)=&alpha(1:1);.cm Get first letter
    .se Letter(&Lalpha)=&&alpha(&Lalpha:&Lalpha)
```



The SIGNAL REMOTE control word causes the REMOTE defined for this line number or identifier to be executed immediately.

.SI	<*|n> <args> <. <N1><N2> <label> >

The REMOTE defined for line "n" will be triggered as if line "n" of the output page was about to be printed.

The value of "n" must lie in the range 1 to 32767 or be a string identifier of eight characters or less. Note that a value of "n" greater than the page length can never be triggered automatically during formatting.

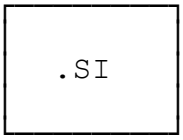
The optional inclusion of "args" sets reference symbol "&0" to the count of arguments and "&1", "&2"... to the arguments. The reference symbol "&*" is set to the string of all arguments. When the LOCAL option is specified, all of these reference symbols are defined as Local variables and may only be used within the remote being signalled. Keyword parameters may be specified with a valid .SR assignment as an argument. Keyword parameters are assigned as Global variables.

N1 and N2 may be specified to include only a portion of remote "n". "label" may be specified for an implied Goto at the start of the target remote. See the description of Append ".AP" or Imbed ".IM" for more information.

Every control word is first tried as a signal operand. That is to say that a control word of the form ".XX", which is not recognized by SCRIPT, will first be tried as ".SI XX" before producing an unrecognized control word diagnostic. Such string identifiers are converted to upper case for matching. It is recommended that any such user defined control words start with a non-alphabetic such as "$" or "!" to prevent name conflicts with current or future SCRIPT control words.

Defaults:

This command will not create a break when encountered. If the operand is missing or coded as asterisk "*" then the value of "n" will be set to the TOP MARGIN plus one, i.e. (n=TM+1). If no REMOTE has been defined for line "n", then this command is treated as a no-op.



Examples:

(1)
```
...
.rm 201 save nosave
the party of the first part
.rm
.rm 202 save nosave
the party of the second part
.rm
This is to inform
.si 201
that
.si 202
agrees with the proposal.
...
```

Produces:

This is to inform the party of the first part that the party of the second part agrees with the proposal.

(2)
```
.SI $$ arg1 arg2 kw1='string' kw2=4*4
.$$ arg1 kw1='string' arg2 kw2=4*4
```

The above two are equivalent and

```
.SI SPACE
.SPACE
```

so are these two.

(3)
```
.SI SP
.SP
```

These are equivalent if a named remote exists with the name of "SP".

The SKIP control word generates a specified number of blank lines before the next output line, except at the top of a page.

.SK	<1|n> <A> <C>

The SKIP control word causes "n" blank lines to be output. If the end of the page is reached before satsifying the request or if output is at the top of a page then these blank lines will be omitted. Skips at the start of a text block, such as a .CP or .FK, are deleted if the text block prints at the top of a page. If DOUBLE SPACE is in effect, twice the specified number of spaces are output.

The "A" or "ABS" operand may be specified to space only the specified number of output lines regardless of the current line spacing (.LS) value.

The "C" or "COND" operand may be specified to define a conditional skip. These conditional skips are ignored if a SKIP, SPACE or PAGE Eject command follows immediately with no intervening text. If a Conditional Skip occurs at the start of a Conditional Page or Floating Keep block, it will be ignored if that block prints at the top of a page. Conditional Skips at the end of the same blocks will be in effect after printing those blocks, even if on a later page.

If the operand is zero, then the next input text line will be overprinted on the last output line, if OFFLINE or if the ONLINE terminal has a negative Line Feed capability. In the case of ONLINE output, the next output text line will be output normally, but will not be counted towards the page length. This facility is useful to allow the user to store equations and such when two or more typeballs are necessary during output.

Defaults:

This command does create a break when encountered and if the operand is omitted "n = 1" will be assumed.

The SPACE control word generates a specified number of blank output lines.

.SP	<1|n> <A> <C>

The SPACE control word causes n blank lines to be output. If the end of the page is reached before satisfying the request, the remaining spaces may or may not be output on the next page depending on whether .LE YES has been specified. Spaces at the start of a text block, such as .CP or .FK, are treated the same way if the text block prints at the top of a page. If DOUBLE SPACE is in effect, twice the specified number of spaces are output.

The "A" or "ABS" operand may be specified to space only the specified number of output lines regardless of the current line spacing (.LS) value.

The "C" or "COND" operand may be specified to define a conditional space. These conditional spaces are ignored if a SKIP, SPACE or PAGE Eject command follows immediately with no intervening text.

If the operand is zero, then the next input text line will be overprinted on the last output line, if OFFLINE or if the ONLINE terminal has a negative Line Feed capability. In the case of ONLINE output, the next output text line will be output normally, but will not be counted towards the page length. This facility is useful to allow the user to store equations and such when two or more typeballs are necessary during output.

Defaults:

This command does create a break when encountered and if the operand is omitted n = 1 will be assumed.

Notes:

(1) .SP control words within footnotes cause blank lines to be generated within the printout of the footnote at the bottom of the page.

The SET REFERENCE control word allows the user to assign a character or numeric value to a symbolic reference name.

.SR	name<=> delimited string undelimited string numeric-expression

The reference name named by the first operand is given the value of the second operand. If the reference name does not exist, it will be created. The first operand may optionally be followed by an equal sign "=".

The .SR control word may be used for a variety of purposes. For example:

(1) Symbols may be defined as having as their value the number of the page on which they appear and can be used to construct a table of contents.
(2) Symbols can be assigned to count the number of equations being used and to number the equations as they appear on the output.

When the UPPER parm is in effect a reference name may be any character string of ten or fewer characters where each character is greater than or equal to lower 'a' (all upper and lower alphabetics and numerics) or the national characters ('$', '@', '#') or an underscore ('_'). The name is converted to uppercase prior to use.

When the NOUPPER parm is in effect a reference name may be any character string of length ten or fewer which does not contain a blank or any special characters. The list of special characters terminating reference names includes the characters:

.	period	=	equal sign
+	plus sign	-	minus sign
*	asterisk	/	slash
(	left parenthesis	)	right parenthesis
'	quote mark	&	ampersand

The reference name may optionally be subscripted with an integer, signed or unsigned. A subscript may range from -32767 to +32767. A zero subscript is logically the same as having no subscript. The subscript value may be implied by specifying a null subscript in the form "()". Thus:

```
.sr symbol()=...
```

.SR

is a short form of

```
.se symbol=&symbol+1
.se symbol(&symbol)=...
```

or equivalently

```
.ur .sr symbol=&symbol+1
.ur .sr symbol(&symbol)=...
```

If a character string is the assigned value, it must consist of no more characters than the maximum allowed by the SRLENGTH= parameter, normally one hundred and fifty, or the string will be truncated on the right. A character argument may be a delimited string or an undelimited string. In the first case, if the first character is a ' (quote), " (double quote), / (slash), | (or bar), ! (exclamation mark), ¬ (not sign), or ¢ (cent sign) then the character value assigned will be from the character following the delimiter up to the character preceeding a matching delimiter that is followed by a blank. Blanks are then valid within the operand and the delimiter character can be entered into the operand if not followed by a blank. In the second case, an undelimited string is terminated by the first blank or end of record. The single quote and double quote are recommended as delimiters.

The syntax supported in numeric expressions is like that of FORTRAN. Decimal terms and binary, character and hexadecimal self-defining terms in the ASSEMBLER sense are supported. Parentheses and unary plus and minus operators are fully supported. Blanks between terms and operators are optional. See Example (1) for sample expressions.

The following <escape>'s are recognized:

(1) "%" Current page number.
(2) "&" Current page number.
(3) "PS" Current page number, where PS is the symbol defined by .PS.

Defaults:

This command word will not create a break when encountered. If one or both operands are omitted it will be treated as an error.

Notes:

(1) See the description of the .UR control word and the TWOPASS option for further hints on the use of .SR.

(2) See Appendices for a list of System Reference Names with their formats and meanings.

Examples:

(1) The following demonstrate valid expressions:

```
.sr i=5
.sr i= +5
.sr i=((+5*3-1)*7)+1
.sr i = % + 1
.sr i = ((X'F05'+C'A')* B'1111')/16
.sr i(1+1) = (1+1)
```

(2) The following demonstrate character assignments:

```
.sr c=abcdefg
.sr c = 'ABCDEFG'
.sr c = "don't be silly"
.sr alpha(1) = 'A'
.sr alpha(2) = B
```

(3) The sequence:

```
.sr equno=0
    .
    .
    .
.ur .sr equno=&equno + 1
.ur .sr xeqn &equno
.ur x = erfc(a - bc)                              (&equno)
.ur .sr equno = &equno +1
.ur y = erfc(a +bc)                               (&equno.)
    .
    .
    .
.ur Using the previous result for x (Equation &xeqn.)...
```

Produces:

```
x = erfc(a - bc)                                  (1)
y = erfc(a + bc)                                  (2)
Using the previous result for x (Equation 1)...
```

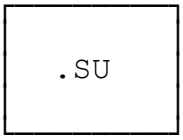
.SR

(4) The sequence:

```
.sr a 0
.sr b -5
.sr c a
.sr d 'b'
.ur .ur .sr &c &&&d + 1
.ur a = &a..
.ur b = &b..
```

Produces:

```
a = -4.
b = -5.
```



The SUBSTITUTE SYMBOL control word causes subsequent input text and control lines to be reformatted by substituting the current values of specified reference names in the line and to be processed as if it had been in the input.

.SU	<1|n|ON|OFF|line> <UPPER|NOUPPER> <TRACEON|TRACEOFF>

The "line" operand or the next "n" text and control lines in the input file will be reformatted by substitution with current values of reference names. In other words, the next "n" lines of input will be interpreted as if sufficient ".UR" USE REFERENCE control words preceded them to remove all reference names from the string.

The "UPPER" argument causes all Reference Variables to be converted to upper case in an assignment or when being used. Thus "&X" and "&x" would refer to the same variable. The "NOUPPER" argument causes all Reference Variables to be used as entered. This option may also be set using the UPPER or NOUPPER option when SCRIPT is invoked.

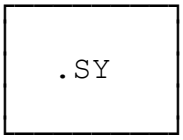
The "TRACEON" argument uses the SYSTERM error file to print a record of substitutions under .SU or .UR control words. The initial input record is listed with double spacing and each level of substitution is listed with single spacing. The "TRACEOFF" argument is used to turn this debug feature off.

Defaults:

The ".SU" control word does not act as a break itself. A numeric operand will substitute in the following "n" input text lines. A missing operand will substitute in the one following input text line. A "ON" operand will substitute in all following input text lines until an "OFF" or numbered "n" operand is encountered.

Notes:

(1) See the description of the ".SR" and ".UR" control words for additional information.



The SYSTEM COMMAND control word passes a command line to the host operating system.

The .SY control word should be used whenever a terminal system command is to be issued each time a SCRIPT document is formatted.

Under CMS only CP and CMS Subset commands may be issued.

The System Variable Symbol "SYSRET" provides the return code from the last .SY command issued.

Defaults:

This control word does not cause a break. If the "line" operand is omitted, no action is performed.

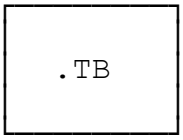
Notes:

(1) The only host operating system this command is currently supported under is CMS. In TSO or OS Batch, the command is bypassed.

Examples:

(1) .sy cp spool prt cont
.ur .if &SYSRET ne 0 .qq

This example issues a "CP" command to create a continuous print file. If the command was not successful, SCRIPT terminates using ".qq".



The TAB SETTING control word specifies the tab stops to be assumed for the following lines when converting the TAB character (X'05') generated by the typewriter TAB key into the appropriate number of spaces.

.TB	<n1 n2 n3 ...> <<'string'|char/>n<L|R|C|'char'> ...> <SET <char>>

Tabulation characters present in the input file are expanded by SCRIPT into one or more fill characters, blanks by default, or the fill parameter if specified, to simulate the effect of several logical tab positions.

The logical tab position n can be used with the optional fill parameter and alignment parameter. The .TB control word operand n specifies the location of the logical tab stops. The tab columns may be specified with signed values in which case the tab is set in the last tab column plus n. As text is entered the blank space between the end of this text and the next tab position (tabulation gap) is used to accomodate the fill parameter. The fill parameter char or string is associated with a particular column n and may be a single character or a string delimited by ', ", or /. If the fill string is a single character, this character is propagated in the tabulation gap. (see examples 1 and 2). For a fill string greater than one character the following situation results. The fill string is propagated in a work area and the particular column bounds of the tabulation gap are used to extract the expanded fill string, that is, a fill string of 'abc' in a tabulation gap from column 5 to 9 inclusive, will be filled in as bcabc. (see example 3). If the tabulation gap is smaller than the fill string then only a subset of the fill string is extracted. (see example 4).

The alignment parameter associated with column n can be used for left or right justification, centering, or character alignment. When tab positions are specified, fields are left justified in column n. If fields are to be right justifed or centred then an R or C should immediately follow the tab position. (See examples 6 and 7). Fields can also be aligned on a particular character by specifying the quoted character following the n operand. The alignment processing is done during input before any other formatting occurs. This functions by logically tabbing to column n and searching for the alignment character. The length of the

.TB

tabulation gap is subsequently adjusted so this character will be aligned in column n. (see example 8).

A .TB control word with no operands causes reversion to the default tab settings.

A .TB control word with a "SET" operand may be used to define a single character that is to be treated as a user tab character. The user Tab character is in addition to the normal Tab character and is treated as a Tab in text and control commands. A "SET" with no character operand following only allows the normal Tab character (X'05').

Defaults:

This control word will create a break. Until ".tb" is encountered with operands, the tab columns are 5, 10, 15, ..., 80. They will be in effect from the beginning or when ".tb" is encountered without operands.

Notes:

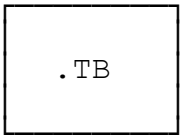
(1) The tab stops must be specified in ascending order.
(2) Upon encountering a phrase which can not be centered or right justified on a particular column, then the next tab setting is obtained which allows the desired result. Thus each phrase under an alignment constraint is bounded by the previous and next tab stops specified. (see example 5).
(3) Under CONCATENATE (.CO) or FORMAT (.FO), attempting to tab outside the limit of the LINE LENGTH (.LL) produces unpredictable results.

Examples:

(1)
```
.tb set :
.tb 5 +/15 -/25 */35
:a:b:c:d
```

will produce

```
5         15         25         35
↓         ↓          ↓          ↓
a+++++++++b---------c*********d
```



(2) An equivalent result is produced by:

```
.tb 5 +/+10 -/+10 */+10
```

(3) .tb set :

```
.tb 20 'xyza'30
12345:123:123
```

produces

```
          20        30
           ↓         ↓
12345      123zaxyzax123
```

(4) .tb set :

```
.tb 1/10 '234'25 "5678"40 /1234567890/55
this:is a:tab:fill:test.
```

produces

```
        10        20        30        40        50
         ↓         ↓         ↓         ↓         ↓
this11111is a34234234234tab856785678567fill45678901234test.
```

(5) .tb set :

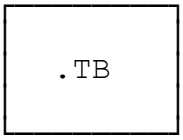
```
.tb 5c 10c
:2345678
.sp
:234 67890
```

will produce

```
5   10
↓    ↓
2345678

 234 67890
```

(6) .tb set :

```
.tb 19c 30c 50c
.nf
.ur .uc command:break:default:meaning
.sp
.tb 20r 30c 40l
 cw c:no:c=;:control word separator is c
.sp
 in n:yes:n=0:indent left margin n spaces
.sp
 br:yes::break
```



produces

```
                    19          30                  50
                     ↓           ↓                   ↓
COMMAND          BREAK      DEFAULT             MEANING

 cw c              no         c=;         control word separator is c

 in n             yes         n=0         indent left margin n spaces

 br               yes                     break
```

```
     (7) .tb set :
         .sp
         .tb   10r  12l 50r
         :sin x =:x - x**3/3! + x**5/5! - ...:(9)
         .sp
         :sinh x =:x + x**3/3! - x**5/5! + ...:(10)
```

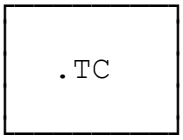
produces

```
   sin x = x - x**3/3! + x**5/5! - ...          (9)

  sinh x = x + x**3/3! - x**5/5! + ...         (10)
```

```
     (8) .tb set :
         .tb 25'.'
         CPU charges:$123.45
         .sp
         I/O costs:$1.26
```

produces

```
CPU charges          $123.45

I/O costs              $1.26
```



The TABLE OF CONTENTS control word causes the current Table of Contents to be printed. The control words .H0 through .H9 and .PT are used to generate entries in the Table of Contents.

.TC	<1|n|* <CONTENTS|line>> <ADD <m ... >> <PURGE>

Up to ten concurrent Table of Contents may be accumulated for printing and the table printed by this command is determined by the current .DH Table of Contents number.

The operand "n" of the .TC control word specifies the number of pages reserved for the Table of Contents. If the Table of Contents requires other than n pages, a gap or overlap in pagination will occur. A default of 1 page is used if this operand is not specified. An operand of * causes page numbering to proceed sequentially from the Table of Contents to the text following.

The "line" operand is the phrase used for the Table of Contents heading. A default phrase of CONTENTS is generated if "line" is omitted. The heading is created by generating a pseudo head-level 1 with an operand of CONTENTS or "line". The "line" operand would be formatted with head-level 1 options in effect from the Define Heading (.DH) control word. This pseudo head-level becomes the first entry in the Table of Contents and normally causes a page eject if not already at the top of the page.

Entries in the Table of Contents are formatted according to the line and page dimensions that are in effect upon encountering the .TC control word.

The ADD operand causes the Tables of Contents specified by <m ... > to be added to the current Table of Contents. Attempting to chain the current table to itself will result in a null operation.

The PURGE operand will delete the current Table of Contents.

The .TC control word is not allowed in a Floating Keep.

.TC

Defaults:

This control word will cause a break.

Examples:

```
.dh 4 tc
.h1 1.0 This is a level one heading.
    .
    text
    .
.h2 1.1 This is a level two heading.
    .
    text
    .
.h3 1.1.1 This is a level three heading.
    .
    text
    .
.h4 1.1.1.1 This is a level four heading.
    .
    text
    .
.pt .sp 2
.pt End of Table of Contents
.tc * Example Contents
```

produces:

.TC

1.0 THIS IS A LEVEL ONE HEADING.

A level one heading has the following default characteristics:

- Starts at the top of a new page.
- Is right justified if it falls on an odd numbered page.
- Is typed in uppercase and underscored.
- Is followed by five spaces.

1.1 THIS IS A LEVEL TWO HEADING.

A level two heading has the following default characteristics:

- Has three line skips before it.
- Is typed in uppercase and underscored.
- Is followed by two spaces.

1.1.1 THIS IS A LEVEL THREE HEADING.

A level three heading has the following default characteristics:

- Has three line skips before it.
- Is typed in uppercase letters.
- Is followed by two spaces.

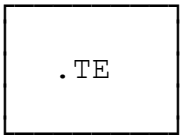
1.1.1.1 This is a level four heading.

A level four heading has the following default characteristics:

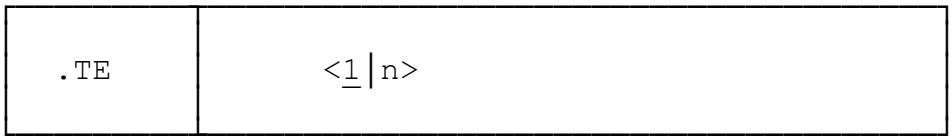
- Has three line skips before it.
- Is underscored.
- Is followed by two spaces.

.TC

EXAMPLE CONTENTS



The TERMINAL INPUT control word allows the user to enter control or input lines[21] during processing of the input file.



When the .TE control word is encountered, the typeball is jiggled, the user's terminal keyboard is unlocked, and "n" lines are accepted and processed as if they had been in the input file. The lines thus input may be text or control information.

Defaults:

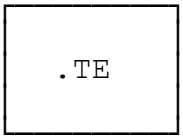
The ".te" control word does not act as a break in itself. However, control words input under its control may act as breaks. If the operand is omitted then n=1 is assumed.

Notes:

(1) If output is being placed on the online terminal, the typing element will space but not print[22]. The user should manually space the carriage back one line to leave it properly aligned after the carriage return which ends the input line.
(2) A completely null line is taken to mean that the user is done typing input lines. The .TE is terminated regardless of whether "n" lines have been input. A null line signals an End Of File.
(3) The user might wish to use .TE in order to dynamically specify control words or text, or change the typeball.
(4) The DDNAME of SYSCONS is used as input for the ".te" control word.

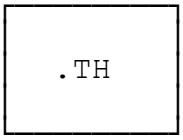
---

(21) Note that the .RD control word merely unlocks the keyboard to allow the user to type. It does not look at what the user types.
(22) The typing element will print during the first pass of a TWOPASS run even if output is online.



Examples:

(1) .cm User will enter name and address here
.te 4



The THEN control word causes an input line to be conditionally included depending on the truth value of a previous IF control word.

The line which begins one blank after the .TH control word is included for processing only if the preceeding IF statement had a "true" value.

The line may include any control word except another Then (.TH) or an Else (.EL). The object line may be another .IF and these may be nested up to ten levels. The object line may also be an Imbed (.IM) or Signal (.SI) in which case the current IF status and its nesting level will be saved and later restored when the current file nest level is resumed.

Defaults:

The ".th" control word does not act as a break in itself. However, control words within the line may create a break if the preceeding .IF is "true" and the interpreted line creates a break. If "line" is omitted then the object of the Then has no effect.

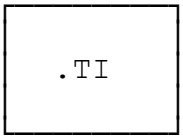
Examples:

(1) .if &sptype = 'page';.th .pa;.el .sp 2

This example either starts a new page or spaces two lines depending on whether the value of reference variable is 'page' or not.

(2) .if &i le 1;.im afile

This example will imbed an input file named "AFILE" if the value of "i" is less than or equal to one. Note that a .TH control word is assumed following a .IF if the immediately following record is not .TH or .EL.



The TRANSLATE ON INPUT control word allows the user to specify an escape character and the contents of a translate table to be used on input lines.

.TI	<s <s|t>> <<s1 t1> <s2 t2> ...> <SET <char>>

In all subsequent input text and control lines, each character following the escape character "<char>" will be translated with all occurrences of "S1" replaced by "T1", etc. and the escape character is removed. If no operand is present the translate table specified by the TRANSLATE option will be reinstated and the escape character will be nullified. If the SET operand is missing, the escape character facility is removed, but the current input translate table remains. With the escape character removed, no action is performed on input lines.

The .TI control word is primarily of use when output must have a character set larger than the input character set. For example, a 029 keypunch lacks the lower case alphabetics but using .TI and .TR translate tables "$A" could print as upper case and "A" as lower.

Defaults:

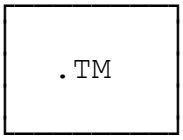
This command will not create a break when encountered. See the TRANSLATE ".TR" control word for more information on specifying the input translate table.

Examples:

(1)
```
.ti < AD > BD ( 8B ) 9B . AF
.ti set ¢
¢. INDex¢<-Queues¢> ¢(¢<Time=(¢<mm¢>¢<,ss¢>)¢)¢>
```

Produces:

- INDex[-Queues] {[Time=([mm][,ss])}]



The TOP MARGIN control word specifies the number of lines which are to be placed between the physical top of the output page and the first line of the text area.

.TM	<6|m|+m|-m>

Subsequent output pages will begin with m lines (which may include heading lines) before the first line of text. An operand of the form "+m" or "-m" adds this value algebraically to the current value of the top margin.

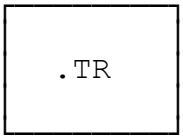
Defaults:

This command word will create a break and until encountered m=6 will be in effect. When encountered without an operand m=6 will be assumed.

Notes:

(1) The TOP MARGIN must never be smaller than the sum of the HEADING MARGIN plus the HEADING SPACE.

(2) The margin specified by .TM will apply to the current page only if no output has yet been produced on it.



The TRANSLATE control word allows the user to specify the contents of a translate table to be used for output.

.TR	<s <s|t>> <<s1 t1> <s2 t2> ...>

Where S1, T1, etc., are single characters or two-digit hexadecimal numbers using upper or lower case letters.

All subsequent output lines will be printed with all occurrences of "S1" replaced by "T1", etc. If no operand is present the translate table specified by the TRANSLATE option will be reinstated.

The .TR control word is primarily of use when output must use a different character set than was used to create the SCRIPT files. For example, the user may print online a file which uses special characters not available on the terminal[23] or use a "correspondence coded" secretary's typewriter typeball with a different type style to be used in place of the normal "terminal coded" typeball by using a special translate table.

Defaults:

This command will not create a break when encountered. No translation, except upper case conversion if the TRANSLATE option was specified, will be in effect until ".tr" is encountered with operands.

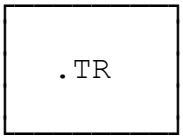
Notes:

(1) Heading, footing, and footnote lines are translated under control of the translate table current when the line is output.

(2) SCRIPT control lines are never translated.

---

(23) For example, the superscript characters are not available on the 2741 terminal.



(3) Translate pairs remain active until explicitly re-specified.

(4) Hexadecimal numbers are recognized by the presence of two characters (instead of one) and may use upper or lower case letters A-F.

(5) The last pair in a .TR line may consist of only one argument, which indicates that the corresponding character is to be translated into itself (left unchanged).

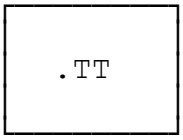
Examples:

(1) .TR 8D (9D) B0 0 ... B9 9
Causes the UN Trains's superscript parentheses and numbers to print as ordinary parentheses and numbers.

(2) .TR % 7C
Causes occurrences of the character "%" to be replaced by the character X'7C', the "@" character, which may not be easy to enter into a file by virtue of being the character erase character in some terminal systems.

(3) .TR 40 ?
Causes all blanks in the file to be typed as "?" on output.

(4) .TR 05 40
This is probably an attempt to remove all tabulation characters from the file, but it has no effect since all tab characters are removed prior to printing the output.



The TOP TITLE control word is used to define three headings to be printed at the top of both even and odd numbered pages.

.TT	<1|n> /S1/S2/S3/

where optional "n", from 1 to the value of the HSFSOVER option, gives the heading line number, and S1, S2 and S3 are character strings not containing the delimiter character "/" which can be any character defined as the first character of the operand. Any of the fields may be omitted, but the delimiter character must be included to indicate missing fields, e.g., $S1$$S3$.

The .TT control word specifies 3 items of character data to be printed on one line near the top of all subsequent output pages‘²¼’.

The heading line is generated by:

(1) Substituting the current page number for each appearance of the Page Number Symbol, normally "%".
(2) Left-adjusting S1.
(3) Centering S2.
(4) Right-adjusting S3.

If the top margin is tm, the heading margin hm and the heading space hs the top of the output page will consist of:

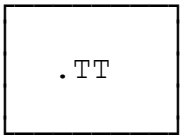
(1) tm - hm - hs blank lines,
(2) the hs heading lines and
(3) hm blank lines.

Defaults:

A break is not created by this command. The default top title on each page after page one is ".tt□///PAGE□%/". This default may be suppressed by .PN with an OFF or OFFNO operand.

---

(24) Including the current page if nothing has yet been written.



Notes:

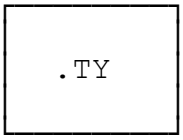
(1) S3 may overlay S2 if necessary, and S2 may overlay S1 if necessary.
(2) Appearance of the .TT control word overrides any previous values of the heading items. If no items are specified, an empty heading line will be generated'²½'. Furthermore, items which are omitted become null and do not retain their previous values.
(3) The Heading and Footing lines share the same buffers. The first to last are referenced by Heading 1 to 9 and the last to first referenced by footing 1 to 9. i.e. .TT□1... and .BT□9... define the same Heading/Footing and .TT□2... and .BT□8... the same, and so on. Use .HS (Heading Space and .FS (Footing Space) with the rule that their sum should not exceed 9 for discrete results. The overlap value of nine is settable from 2 to 19 by using the HSFSOVER parameter.
(4) The TOP TITLE control has the same effect as the HEADING (.HE) control word. The difference is that the field delimiter is self-defined by the first character of the operand.

Examples:

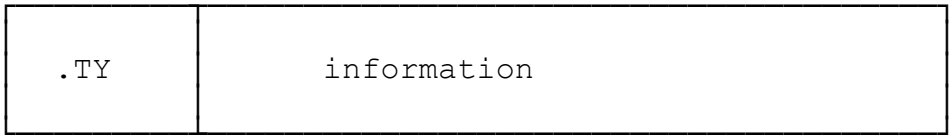
```
.tt   '**********************''***************'
.tt 2 '*Department of Alchemy*''*ATN-05-3-70-%*'
.tt 3 '**********************''***************'
.tm 7;.hm 2;.hs 3
```

---

(25) Setting a null header or setting the Heading Space to zero are the only ways to achieve the effect of an empty Top Margin.



The TYPE ON TERMINAL control word causes one line of information to be typed on the user's terminal.



If output has started to the user's terminal, the .TY control word is ignored. Otherwise, the entire operand field is printed on the terminal.

The .TY control may be of use, for example, immediately preceding a .TE (Terminal Input) control word as a reminder of what to do.

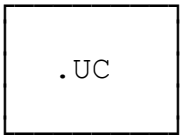
Defaults:

This command word will not create a break when encountered.

Examples:

(1) The following could be used for interactive prompting:

```
.ty Type Name and Address in three lines:
.te
.br
.te
.br
.te
.br
```



The UNDERSCORE CAPITALIZE control word underscores and capitalizes an input line.

.UC	<line>

The .UC control word applies the rules of UPPERCASE (.UP) and UNDERSCORE (.US) to the "line" operand. The "line" operand starts one blank after the control word and may be text or another control word.

Defaults:

This command does not create a break. If the "line" operand is omitted, no action is performed.

Examples:

(1) This is UC example number
.uc one.

produces:

This is UC example number ONE.

The UNDERSCORE DEFINITION control word is used to specify the characters to be underscored with the automatic underscoring commands.

.UD	<<ON|OFF> c1 <c2 ... >> <SET|INCLUDE|IGNORE> char

The .UD control word with the "ON" operand specifies which characters in character or two digit hexadecimal representation are to be automatically underscored with the .US (UNDERSCORE) and the .UC (UNDERSCORE CAPITALIZE) control words. Conversely, the "OFF" operand specifies which characters are not to be automatically underscored with the same facility.

The "INCLUDE" operand defines an escape character that enables the automatic underscoring facility within a line. The "IGNORE" operand defines an escape character that disables the facility within a line. In this way only parts of lines may be underscored. The "SET" operand defines both these escape characters to the same character resulting in the first occurrence disabling the facility, the next enabling it, and so on.

All characters except the following are underscored by default:

Hex	Char	Hex	Char	Hex	Char	Hex	Char
05	TAB	5A	!	6F	?	AD	[
16	BACKSPACE	5D	)	7A	:	BD	]
40	BLANK	5E	;	7F	"		
4B	.	6B	,	8B	{		
4D	(	6D	_	9B	}		

Defaults:

This command does not create a break. This command with no operands clears the escape characters and resets the characters to be underscored. An omitted character operand for SET, INCLUDE or IGNORE disables this facility which is the initial setting.

Examples:

(1) .US ,SPACE=(TRK,(1,1,1))
would give:
,SPACE=(TRK,(1,1,1))

(2) .ud on ( ) ,;.ud off =
.us ,SPACE=(TRK,(1,1,1))
would then give:
,SPACE=(TRK,(1,1,1))

(3) .ud set !
.us !,SPACE=(!TRK,(1,1,1)!)
would then give:
,SPACE=(TRK,(1,1,1))

The UNDENT control word forces the next output line to start a specified number of columns to the left of the current indent.

.UN	<0|n|-n|+n>

The .UN control word causes the next output line to begin n spaces further left than the current indentation. A relative change as in "+n" or "-n" changes the previously specified .UN or .IL (Indent Line).

The .UN control word serves a similar purpose as .OF, but in a different way. The choice between the two is largely a matter of personal preference.

Defaults:

This command will create a break when encountered. If the operand is omitted then n = 0 will be assumed.

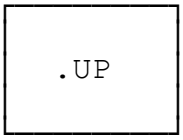
Notes:

(1) The undentation may not exceed the current indentation.

Examples:

(1) The following two sequences are equivalent:

```
.in 5
.of 4
<text>

.in 9
.un 4
<text>
```



The purpose of the UPPERCASE control word is to capitalize an input line.

The .UP control word converts each lower case alphabetic in the "line" operand to upper case. The "line" operand starts one blank after the control word and may be text or another control word.

Use the TRANSLATE parameter option if the whole document is to be printed in upper case.

Defaults:

This command does not create a break. If the "line" operand is omitted, no action is performed.

Examples:

(1) This is UP example number
.UP one.

produces:

This is UP example number ONE.

(2) .up .tt //up test//

centres "UP TEST" at the top of each subsequent page.

The USE REFERENCE control word causes an input line to be reformatted by substituting the current values of specified reference names in the line and to be processed as if it had been in the input.

The line which begins one blank after the .UR control word is reformatted by replacing occurrences of strings of the form "&name" or "&name(i)" with the current value of the reference name called "name" or "name(i)".

A reference name instance is denoted by the appearance of a "&" followed by the name of the desired reference name followed by a period, a blank or a special character. See the ".SR" control word for a list of special characters. The variable name "&*" is also valid as a special case. If the reference symbol scanned is null, the ampersand is left unchanged in the string. The current value of the reference name is converted (if necessary) to a character string and substituted for the string "&name." (if a period is used for delimiting) or the string "&name" (if a blank or special character is used).

Certain implied subscripts for reference names are also supported. An implied subscript of "A(*)" concatenates all the entries of that reference name together, from "A(-32767)" to "A(+32767)" except "A(0)", each separated by a comma and blank ", ". Similarly "A(*-)" refers to the concatenation of all negative subscript elements of "A" and "A(*+)" refers to all positive subscript elements of "A".

Substring facilities on variables are supported by specifying a start column and an end column separated by a colon ":" following or instead of a subscript. Thus "C(j:k)" refers to columns "j" to "k" of variable "C", and "V(i,j:k)" refers to columns "j" to "k" of variable "V(i)". The value of "j" may not be omitted but "k" may be omitted as it defaults to the end of the variable. Substring operations outside the length of the variable return null.

Substring facilities on variables are also supported by specifying a start column and a length separated by an "or□bar". Thus "C(j|k)" refers to column "j" for length "k" of variable "C", and "V(i,j|k)" refers to column "j" for

.UR

length "k" of variable "V(i)". The value "j" may not be omitted, but "k" may be omitted as it defaults to the end of the variable.

The line thus constructed is processed as if it had been in the original input stream.

Defaults:

The ".ur" control word does not act as a break itself. However, control words within the "line" may create a break when the line is reinterpreted after substituting symbolic references with their values. If "line" is omitted or becomes null after substitution then no action is performed.

Notes:

(1) See the description of the .SR control word for additional information.
(2) If reference names appearing in the line have not yet been assigned a value via .SR, a null character string value will be assumed. Note, however, the effect of the "TWOPASS" option previously described.
(3) Text &'s in a .UR line are indicated by "&&" or "&" followed by a special character. Only one "&" appears in the reformatted line.
(4) Negative values are allowed for numeric reference names and result in a signed integer suitable for use with the .SR control word.
(5) The TAB character is not a blank.
(6) If the variable string is preceeded by an "L'" then the "L'" and the variable will be the Length of that variable. For a numeric variable the length is the number of characters to format the number including a sign if negative.
(7) If the variable string is preceeded by a "T'" then the "T'" and the variable will be replaced by the Type of that variable. For a character variable a 'C' is returned and for a numeric variable a 'N' is returned.
(8) See Appendices for a list of System Reference Names with their formats and meanings.

.US

The UNDERSCORE control word is used to underscore an input line.

.US	<line>

The .US control word underscores each character of the "line" operand according to the UNDERSCORE DEFINITION (.UD) specifications. Special characters and punctuation are normally not affected. The "line" operand starts one blank after the control word.

Remember that each character to be underscored is replaced by three characters: the character, a backspace and an underscore. The maximum input line length after underscoring is 240 characters.

Defaults:

This command does not create a break. If the "line" operand is omitted, no action is performed.

Examples:

(1) This is US example number
.us one.

produces:

This is US example number one.

(2) .UD SET !;.us !1.2 !CHAPTER_TITLE

produces:

1.2 CHAPTER TITLE

The WIDOW control word prevents the first and last lines of a paragraph from being split across a page boundary from the rest of a paragraph.

.WD	<YES|NO>

The .WD control word will cause a page eject to occur if the first line of a multi-line paragraph is to print on the last line of the text area. It will also cause a page eject before the second last line of a paragraph if the last line of a paragraph would have appeared alone (widowed) at the top of a page.

Defaults:

This command will not cause a break. The initial value of this control word is "NO". An omitted operand is equivalent to "YES".

Notes:

(1) This is a control word that is still in "under development". Results should be examined carefully if this facility is enabled, because it sometimes creates a page eject one line prematurely.

(2) If a three line paragraph would be printed with its last line on the next page, note that all three lines will be ejected to the next page.

OTHER CONTROL WORDS AND ARGUMENTS

There exists a growing set of basic SCRIPT control words that are undocumented in this reference guide. These commands are undocumented because they are special purpose, redundant or no longer useful. Most such control words exist in this version of SCRIPT for compatibility with other versions of SCRIPT but some are intended for use only by programmers debugging a programme function.

These control words have been omitted from the body of this Reference Guide in order to make the beginner less overwhelmed by the quantity of material to be learned. The objective is a smaller set of control words with common arguments that are easier to remember and thus easier to use effectively.

An example of control words is the ".PA ODD" control word that ejects to the top of the next odd numbered page. The other two implementations ".PD" for PAGE ODD and ".OP" for ODD PAGE are difficult to remember because of infrequent use.

An example of common arguments is ".FN BEGIN", ".CE YES" and ".SU ON". For less frequently used control words it can be difficult to remember which pair to use -- ON/OFF, YES/NO, BEGIN/END or SAVE/NOSAVE. Actually, ON, Yes, Begin and SAve are all equivalent positive operands and OFf, No, End, STop and QUit are all negative operands. So if you know ".CE" controls centering of text it doesn't matter if you enter ".CE YES", ".CE ON" or ".CE B", the function will be the same.

Numeric arguments are much the same. Where possible, the relative argument is supported. For example, ".IN +5" can be very powerful if you don't want to worry about the current indent setting. And always, a numeric expression may be used in place of a single term. ".IN 30" may be what you code but ".IN &SYSLL/2" is what you mean; ".SI 201" may be what you code but ".(100*2)+%" is what you mean.

The undocumented control words are listed here with a brief explanation of their function for complete documentation of the current implementation. It is suggested that the preferred commands should be used wherever possible.

A complete documentation of all control words may be formatted by specifying '+TYPE=ALL' in the invocation parameter list. This manual was formatted with TYPE=''.

.AB <1|n|YES|NO|TRACEON|TRACEOFF>
The ABEND control word is special purpose. It will take effect only if DEBUG has been specified at invocation time. A number or YES will terminate SCRIPT with the appropriate User Abend. A NO is a no-op and TRACEON or TRACEOFF interacts with SUPRTRAC for debugging.

.AR <LOWER|UPPER>
The ARABIC control word causes page numbers to use Arabic numerals.
".PN ARABIC <LOWER|UPPER>" is the preferred form.

.DC
The DON'T COUNT control word causes all output produced directly by the immediately following text line to not be counted for purposes of page length control.
".SP 0" is the preferred form.

.DS
The DOUBLE SPACE control word causes a line to be skipped between lines of output except in footnotes.
".LS <0|n|YES|NO>" is the preferred form.

.EN
The END DO GROUP control word terminates the range of a Do Group started with a ".DO".
".DO END" is the preferred form.

.EP
The EVEN PAGE control word causes output to continue on the next even numbered page.
".PA EVEN" is the preferred form.

.FD <1|n> <S1|'S1'S2'S3'>
The ODD FOOTING control word specifies three items of title information to be printed at the bottom of odd numbered pages.
".OB <1|n> /S1/S2/S3/" is the preferred form.

.FE <1|n> <S1|'S1'S2'S3'>
The footing control word specifies three items of title information to be printed at the bottom of even and odd numbered pages.
".BT <1|n> /S1/S2/S3/" is the preferred form.

.FI <YES|NO|LEFT|RIGHT|CENTRE|INSIDE|OUTSIDE|HALF>
The FORMAT control word combines the effect of CONCATENATE and JUSTIFY.
".FO" is the preferred form.

.FV <1|n> <S1|'S1'S2'S3'>
The EVEN FOOTING control word specifies the three items of title information to be printed at the bottom of even numbered pages.
".EB <1|n> /S1/S2/S3/" is the preferred form.

.HD <1|n> <S1|'S1'S2'S3'>
The ODD HEADING control word is used to define three headings to be printed at the top of odd numbered pages.
".OT <1|n> /S1/S2/S3/" is the preferred form.

.HE <1|n> <S1|'S1'S2'S3'
The HEADING control word is used to define three headings to be printed at the top of both even and odd numbered pages.
".TT <1|n> /S1/S2/S3" is the preferred form.

.HV <1|n> <S1|'S1'S2'S3'>
The EVEN HEADING control word is used to define three headings to be printed at the top of even numbered pages.
".ET <1|n> /S1/S2/S3/" is the preferred form.

.NC <YES|NO>
The NO CONCATENATE control word stops words from shifting to or from the next line.
".CO NO" is the preferred form.

.NF <YES|NO>
The NO FORMAT control word causes lines to be output as they appear in the input by negating the effect of JUSTIFY and CONCATENATE.
".FO NO" is the preferred form.

.NJ <YES|NO>
The NO JUSTIFY control word stops justification of text to the right margin.
".JU NO" is the preferred form.

.OC <char|ON|OFF|n> <c1 ...>
char/<ON|OFF|n> <c1 ...>
The OVERLAY CHARACTER control word causes characters in input text lines to be overlaid with a backspace and user defined character or itself.
".US <line>" is the preferred form.

.OP
The ODD PAGE EJECT control word causes output to continue on the next odd numbered page.
".PA ODD" is the preferred form.

.PD
The ODD PAGE control word causes output to continue on the next odd numbered page.
".PA ODD" is the preferred form.

.PR information
The PRINT control word causes one line of information to be typed on the user's terminal.
".TY information" is the preferred form.

.PV
The EVEN PAGE control word causes output to continue on the next even numbered page.
".PA EVEN" is the preferred form.

.RI <1|n|YES|NO|line>
The RIGHT ADJUST control word causes the next input line to be right adjusted in the output line.
".RA" is the preferred form.

.RO <LOWER|UPPER>
The ROMAN control word causes page numbers to use lower-case Roman numerals.
".PN ROMAN <LOWER|UPPER>" is the preferred form.

.SS
The SINGLE SPACE control word causes output to be single spaced.
".LS <0|n|YES|NO>" is the preferred form.

.UL <1|n|ON|OFF> <line>
The UNDERLINE control word causes alphanumeric characters in subsequent text lines to be overstruck with a BACKSPACE and an underscore.
".US <line>" is the preferred form.

SAMPLE INPUT TO SCRIPT

```
.tt 'SCRIPT'APPENDIX C'Page %'
.ur .bx 1 &sysll
.sp;.ce SAMPLE OUTPUT FROM SCRIPT
.sp;.bx off
.rm 45
.ra EXAMPLE OF REMOTE CONTROL
.rm
.sp 2
"Oh nuts!  I have to type this whole page over".  That
complaint is probably heard many times in the course of
preparing manuscripts--reports, letters, minutes and so on.
Then there is the problem of making duplicate
.us original
copies--of a typed
manuscript.  When we use an ordinary typewriter,
text-processing, as this is called, can be a time-consuming
and harrassing job.  The SCRIPT facility, which operates
under CMS(*),
.fn begin
(*) Conversational Monitor System for VM/370
.fn end
was written to handle these procedures for you;
you merely type in the first version and make
any necessary corrections later.  By
using the "commands" that you type in with your lines of
text, the computer prints out as many "first" copies as you
wish, with margins, spacing, indentation, etc., performed
in accordance with your commands.
.sp 1
One of the most useful features of SCRIPT is right-margin
justification, as in book and newspaper printing.  This
means that your text is spaced as evenly as possible
between the margins of your printed page, filling in with
blanks where  necessary.  The printed line is
under control of a "line-length" command; any
short line that you type in will "grab" words
from a longer line above or below it, and fill out the
line with blanks where necessary to avoid splitting words
at the margins.  This is the ordinary (or default) mode
under SCRIPT.  You can, however, have lines printed out
exactly as you type them in, with no justification
performed by SCRIPT, by including a command that instructs
the computer to turn off the format mode.  This is useful
for typing figures and charts.  Line justification can
later be respecified if desired.
.cm Last line of Input Text.
```

SAMPLE OUTPUT FROM SCRIPT

"Oh nuts! I have to type this whole page over". That complaint is probably heard many times in the course of preparing manuscripts--reports, letters, minutes and so on. Then there is the problem of making duplicate original copies--of a typed manuscript. When we use an ordinary typewriter, text-processing, as this is called, can be a time-consuming and harrassing job. The SCRIPT facility, which operates under CMS(*), was written to handle these procedures for you; you merely type in the first version and make any necessary corrections later. By using the "commands" that you type in with your lines of text, the computer prints out as many "first" copies as you wish, with margins, spacing, indentation, etc., performed in accordance with your commands.

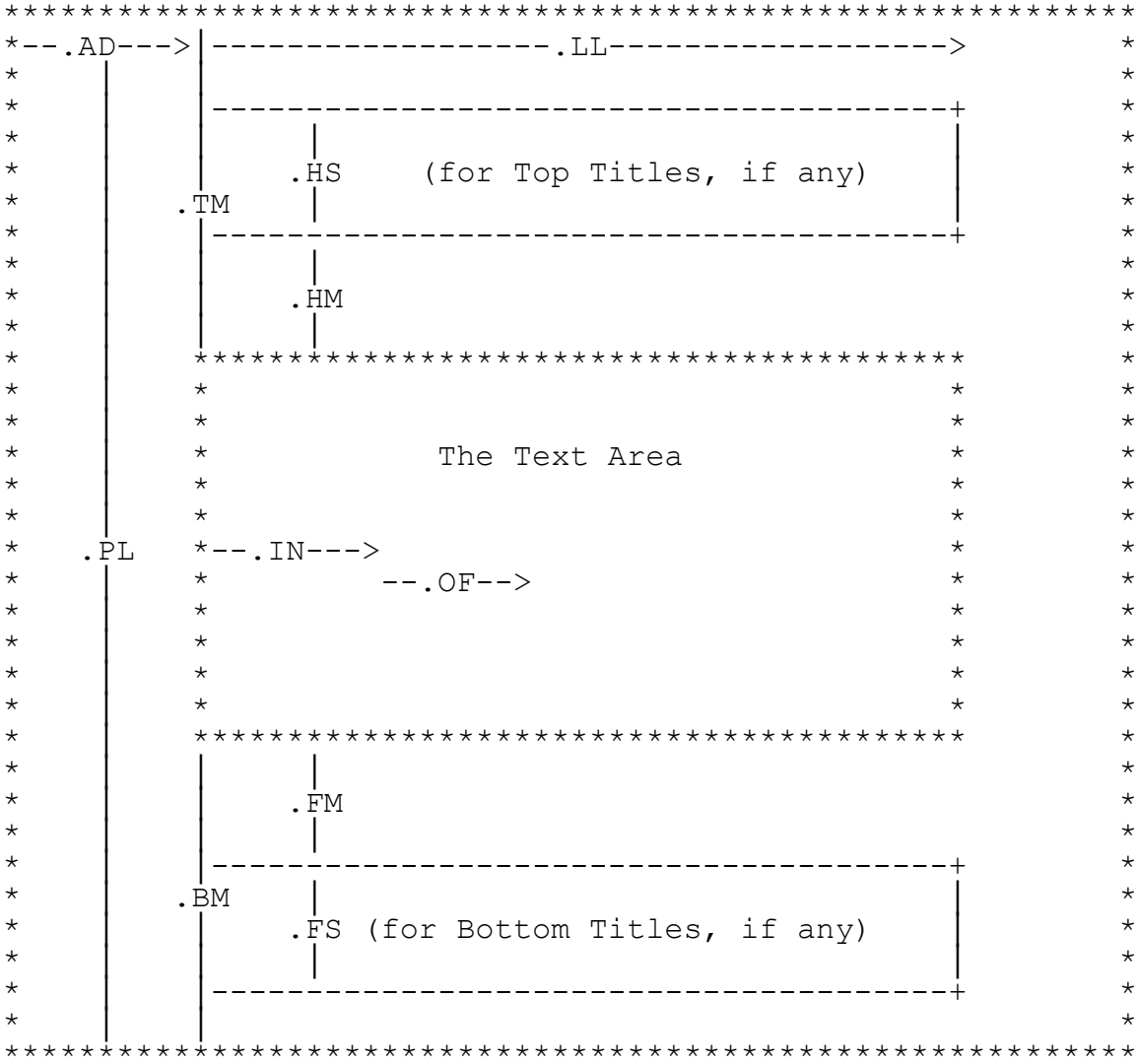
One of the most useful features of SCRIPT is right-margin justification, as in book and newspaper printing. This means that your text is spaced as evenly as possible between the margins of your printed page, filling in with blanks where necessary. The printed line is under control of a "line-length" command; any short line that you type in will "grab" words from a longer line above or below it, and fill out the line with blanks where necessary to avoid splitting words at the margins. This is the ordinary (or default) mode under SCRIPT. You can, however, have lines printed out exactly as you type them in, with no justification performed by SCRIPT, by including a command that instructs the computer to turn off the format mode. This is useful for typing figures and charts. Line justification can later be respecified if desired.

EXAMPLE OF REMOTE CONTROL

(*) Conversational Monitor System for VM/370

PAGE SETUP

The outer rectangle of *'s in the diagram below indicate the physical dimensions of a sheet of paper. The inner rectangle of *'s represents the "text area" (that part of the physical page into which SCRIPT formats the input text).



The default values for the control words shown above are as follows:

.AD 0	(Adjust)	.IN 0	(Indent)
.BM 6	(Bottom Margin)	.LL 60	(Line Length)
.FM 1	(Footing Margin)	.OF 0	(Offset)
.FS 1	(Footing Space)	.PL 66	(Page Length)
.HM 1	(Heading Margin)	.TM 6	(Top Margin)
.HS 1	(Heading Space)		

OFFLINE CONTROL STATEMENTS

```
//jobname  JOB  ...  Accounting Information
//stepnam1 EXEC PGM=SCRIPT,REGION=128K,PARM='options'
//SYSPRINT DD  SYSOUT=A
//SYSTERM  DD  SYSOUT=A
//SYSIN    DD  DSNAME=input.file,DISP=SHR
//SYSCONS  DD  DSNAME=console.file,DISP=SHR
//SYSLIB   DD  DSNAME=library.file,DISP=SHR
/*
```

Where:

options The PARM parameter on the EXEC statement contains the "options" which are as specified and used as described on page 2.

DDNAMES required by SCRIPT and their functions are listed below:

SYSIN is for the primary input dataset. The dataset name referenced should be equivalent to the Online Command Format's "filename" parameter being the dataset qualification level preceeding SCRIPT.

SYSTERM is for output of messages such as those from ".pr" or ".ty" and those initiated by errors and ".er". It also is used when output is online for the message "Load paper; hit return:".

SYSCONS is used for input and responses to messages printed in SYSTERM. The SCRIPT control words ".rd", ".rv" and ".te" use this DDNAME to receive input lines. If these control words are not used and OFFLINE is specified this DD Statement can be omitted.

SYSPRINT is for the formatted and edited output from SCRIPT.

SYSLIB is a partitioned dataset for .AP and .IM by member name only.

Notes:

(1) Output datasets used by SCRIPT must have variable length records with ASA carriage control. Any BLKSIZE that does not span tracks can be used for the input datasets. The record length of output datasets may range up to 244 with 137 as the default.

(2) Besides the required DD Statements mentioned above each IMBED or APPEND Control word whose Filename has not been used in any previous text files must have an additional DD Statement included in the job setup with the Filename used as the DDNAME.

(3) A catalogued procedure exists to invoke SCRIPT and supply output DD statements. The following is a possibility for such a procedure:

```
//SCRIPT   PROC PROG=SCRIPT,SCRIPT=,SIZE=128K
//SCRIPT   EXEC PGM=&PROG,PARM='&SCRIPT',REGION=&SIZE
//*
//*      S C R I P T
//*
//SYSHYPH   DD  DSN=SCRIPT.SYSHYPH,DISP=SHR
//SYSPRINT  DD  SYSOUT=A
//SYSTERM   DD  SYSOUT=A
```

NOTES ON THIS DOCUMENT

Footnote Numbering:

In order to allow the greatest flexibility in the placement and numbering of footnotes and in order to take advantage of the superscript characters available on a TN print train, several things were done:

(1) The Variable Symbol "NFN" was used to count footnotes. Just prior to each footnote reference, the control line ".FNADD" was used to cause the following Remote file to be "executed":

```
.se NFN=&NFN+1;.se LP='';.se OFN=L'&NFN+3;.se i=0
.ur .pe L'&NFN
.se i=&i+1;.se FNn=&&NFN(&i:&i);.se LP='&LP&&I&FNn'
.se LP='&L&LP&R'
```

(2) A file named "SUPER" was created which contained the following:

```
.bs +
.sr L='8+D'
.sr R='9+D'
.sr I0='B+0'
    .
    .
.sr I9='B+9'
```

In file "SUPERT" the same variable names were defined with their non-superscript equivalents. Then if the "&SYSOUT" system variable symbol had the value "PRINT" indicating off-line output, the command ".im□super" was executed.

(3) Each footnote reference of the form "<word><ref>" was entered as follows:

```
.ur <word>&LP
```

which produces a reference of the form "<word>(1)" for on-line output or <word>'¹' for off-line output.

(4) Each footnote began:

```
.ur .of &OFN
.ur (&NFN.) ...
```

Table of Contents:

(1) Each line in the table of contents was a ".UR" containing an instance of a mnemonic reference name, which was set to the page number of the start of the section via ".sr ... %".

(2) The Preface and Table of Contents print with Roman numerals. The body of the manual starting with the Introduction starts on page one with Arabic numerals.

(3) The Preface and Table of Contents prints correctly following the initial title page if the PASS=2 option is specified. It is suggested however that PASS=1 be used to save on CPU and I/O costs and that the Preface and Table of Contents which will then print last be manually moved to the start. This manual was formatted with PASS=2.

Manual Dating:

The date which appears on the initial title page is a line of text in input file member "HEADER". The date reflects the version of SCRIPT to which this manual corresponds.

File Organization:

The following SCRIPT files were used during production:

SCRIPT Manual driver containing general description and imbedding "HEADER" page, Preface and Table of "CONTENTS", "OPTIONS", "ERRORS" and the appendices.

NSCRFC1 Contains necessary ".SR", ".PA" and ".IM" control words to format the individual control word descriptor files and generate values for reference names used in the Table of Contents. A table of control words to include is defined by "FCLIST". This file is imbedded by "SCRIPT".

AD, AP, etc. Contain detailed descriptions of the control words. Imbedded by "NSCRFC1".

Manual Format Options:

By setting reference symbol "&BINDING" to 'CENTRE' the manual will be formatted for printing on both sides of each page. This may be changed by specifying '+BINDING=CENTRE' in the parm field. The current value for "&BINDING" is ''.

Manual Updates:

By setting reference symbol "&UPDATE" to a date in the form "YYMMDD" only update pages on or after the date specified will be printed. Pages are not numbered, leaving it to the

user to insert them at the appropriate locating in an existing Reference Manual. The current value for "&UPDATE" is ''.

SYSTEM REFERENCE SYMBOLS AND REMOTES

There are System Reference Variables that are initialized to values describing the SCRIPT environment and information about the date and time. The following first table shows the variables, and they all must be in uppercase if the NOUPPER option was specified, and describes their initial values. These values may be altered with SET REFERENCE control words as regular reference variables, but this is not recommended. All System Reference Variables begin with 'SYS' or '$', so that "SYSDATE" and "$DATE" return the same value.

DATE AND TIME

SYSYEAR The value of the current year. In this year 2012, SYSYEAR has the value '12'.

SYSPYEAR The value of the current year, including the century. so the current SYSPYEAR would be '2012'.

SYSMONTH The two-digit number of the current month of the year with high order zeros. For today, July 3, SYSMONTH has the value '07'.

SYSPMONTH The alphabetic name of the current month, with only the first character in uppercase. This month SYSPMONTH has the value 'July'.

SYSDAYOFY The three-digit Julian day number in the current year, with high order zeros. SYSDAYOFY now has the value '185'.

SYSDAYOFM The two-digit day number in the current month with high order zeros. For today, July 3, SYSDAYOFM has the value '03'.

SYSDAYOFW Today Tuesday, July 3, SYSDAYOFW has the value '3'. Sunday is considered as the first day of the week.

SYSPDAYOFW The alphabetic name of today, with only the first character in uppercase. Today SYSPDAYOFW contains 'Tuesday'.

SYSDATE The current month, day, and year. SYSDATE now has the value 'July 3, 2012'.

SYSHOUR The two-digit hour of the day on the twenty-four hour clock, with leading zeros. It is now 11:26 and SYSHOUR has the value '11'.

SYSMINUTE The two-digit minute of the hour with leading zeros. It is now 11:26 and SYSMINUTE has the value '26'.

SYSSECOND The two-digit seconds of the minute with leading zeros. It is now 11:26:36 and SYSSECOND has the value '36'.

SYSTIME The current time, in hours, minutes, and seconds. SYSTIME now has the value '11:26:36'.

CURRENT VERSION OF SCRIPT

SYSVERSION A character string describing the SCRIPT program currently being used. SYSVERSION now has the value 'UOW SCRIPT - VERSION(3.2) 78JAN13'.

Certain SCRIPT system reference variables exist to provide "programming language" capabilities. These variable names may not be used as the target of a Set Reference operation.

INPUT

SYSFILE The current input DDNAME or FILENAME. SYSFILE now has the value 'SYSLIB'.

SYSFLNEST The current file nesting level of input files as a result of IMBEDs and SIGNALs. The current value of SYSFLNEST is 3.

SYSMEMBER The current input member within SYSFILE. This will be null if there is no member. SYSMEMBER now has the value 'APENDIXG'.

SYSRECNO The current record number within the current file or file member. SYSRECNO now has the value 108.

SYSRMNEST The current file nesting level of Remotes as a result of automatic and signalled remotes. The current value of SYSRMNEST is 0.

SYSSEQNO The eight-digit sequence number, actual or generated, on the current input record. SYSSEQNO now has the value '00000115'.

OUTPUT

SYSDPAGE The current decimal portion for the page number of the page now being formatted. This value may be null if the page numbers are all integer. The current value of SYSDPAGE is ''.

SYSLINE The current output line number on the page now being formatted. SYSLINE now has the value 8.

SYSPAGE The current integer page number of the page now being formatted. This could be used instead of the Page Number Escape symbol in SET REFERENCE control statements. The current value of SYSPAGE is 192.

SYSPPAGE The current page number as it appears in a Title line. This might be in Roman, Arabic or Arabic with a Decimal portion format. The current value of SYSPPAGE is 192.

SYSRETCODE The current return code for SCRIPT set by errors in processing or by the user with the ".ER" control word.

SYSRET The return code from the last SYSTEM (.SY) control word. SYSRET now has the value '0'.

CURRENT VALUE OF CONTROL WORDS

SYSAD	Current Adjust Value.	SYSAD=10
SYSADEVEN	Current Even Page Adjust Value.	SYSADEVEN=0
SYSADODD	Current Odd Page Adjust Value.	SYSADODD=0
SYSBC	Balance Column Switch.	SYSBC=Y
SYSBM	Current Bottom Margin.	SYSBM=5
SYSBS	Current User Backspace Character	SYSBS=
SYSBX	Box in progress Switch.	SYSBX=N
SYSCC	Conditional Column Switch.	SYSCC=N
SYSCCC	Buffered Conditional Column Lines.	SYSCCC=0
SYSCD	Current Column Definition Count.	SYSCD=0
SYSCL	Current Column Length.	SYSCL=0
SYSCP	Conditional Keep Switch.	SYSCP=N
SYSCPC	Buffered Conditional Page Lines.	SYSCPC=0
SYSDA	Current Dark Output Value.	SYSDA=1
SYSDHSET	Current Table of Contents number	SYSDHSET=0
SYSFB	Floating Block Switch.	SYSFB=N
SYSFBC	Total Floating Block Lines.	SYSFBC=0
SYSFBF	First Floating Block Lines.	SYSFBF=0
SYSFK	Floating Keep Switch.	SYSFK=N
SYSFKC	Buffered Floating Keep Lines.	SYSFKC=0
SYSFM	Current Footing Margin.	SYSFM=2
SYSFN	Current Footnote Switch.	SYSFN=N
SYSFNC	Current Buffered Footnote Lines.	SYSFNC=0
SYSFNCOUNT	Count of Footnotes.	SYSFNCOUNT=27
SYSFS	Current Footing Space.	SYSFS=1
SYSHI	Current Hanging Indent Value.	SYSHI=0
SYSHM	Current Heading Margin.	SYSHM=2
SYSHN	Current Headnote Switch.	SYSHN=N
SYSHNC	Buffered Headnote Lines.	SYSHNC=0
SYSHS	Current Heading Space.	SYSHS=1
SYSHY	Current Hyphenation Mode.	SYSHY=USER
SYSHYC	Exception Dictionary word count.	SYSHYC=0

SYSIN	Current Indent Value.	SYSIN=0
SYSINR	Current Right Indent Value.	SYSINR=0
SYSLL	Current Line Length.	SYSLL=60
SYSLS	Current Line Spacing.	SYSLS=0
SYSMC	Multiple Column Mode Switch.	SYSMC=N
SYSPI	Current Paragraph Indent Value.	SYSPI=0
SYSPL	Current Page Length.	SYSPL=66
SYSSC	Single Column Switch.	SYSSC=N
SYSTM	Current Top Margin.	SYSTM=6
SYSWD	Current Widow Switch.	SYSWD=N

OPTIONS

SYSCONT The error CONTINUE count. SYSCONT now has the value 0.

SYSCPI The value of the Characters Per Inch (CPINCH=) parameter. SYSCPI now has the value 10.

SYSPARM The list of options passed to SCRIPT when it was invoked. SYSPARM now has the value 'TWOPASS'.

SYSPASSNO The current pass number on the input. The current value of SYSPASSNO is 2.

SYSPASSOF The total number of passes to be made on the input files to produce one output file. This value is set by PASSES= at invocation. The current value of SYSPASSOF is 2.

SYSOUT Returns 'TERM' if output is online and 'PRINT' if output is offline. The current value of SYSOUT is 'PRINT'.

SYSONLINE Returns 'Y' if output is online and 'N' if not. The current value of SYSONLINE is 'N'.

SYSOFFLINE Returns 'Y' if output is offline and 'N' if not. The current value of SYSOFFLINE is 'Y'.

SPECIAL REMOTES

If the user chooses to define a named remote with a name that is in the following list, then SCRIPT will automatically signal the remote whenever the appropriate circumstance occurs.

SYSBLANK is signalled with no operands if a blank input line is encountered. If this REMOTE does not exist, then a ".SK" is substituted. The following is a useful possibility for such a remote:

```
.rm SYSBLANK save nosave
.sp;.cp 2
.rm
```

SYSIX0 is signalled during the printing of an Index whenever the first letter of the first-level index entry changes, with that first letter as the value of reference variable "1".

SYSIX1 is signalled before each first-level index entry is printed.

SYSIX2 is signalled before each second-level index entry is printed.

SYSIX3 is signalled before each third-level index entry is printed.

The following named remotes are automatically signalled when a circumstance occurs. They may manipulate reference variables or titles. They should not generate text or do spacing as SCRIPT will continue processing the current page with the event that caused the signal.

SYSBM is signalled before printing the Bottom Margin area.

SYSBOT is signalled after printing the Bottom Margin area.

SYSTM is signalled after printing the Top Margin area.

SYSTOP is signalled before printing the Top Margin area.

PRINT TRAINS

The following table shows the special characters and their hex equivalents on the UN (modified ALA) print train. Only characters which cannot directly be input on a 2741 terminal are listed.

Ø	X80	°	XB0	[	XAD	]	XBD	≠	XBE	□	X3A	È	X74
é	X51	¹	XB1	{	X8B	}	X9B	≥	XAE	□	X3B	°	X71
ê	X52	²	XB2	⌜	XAC	⌝	XBC	≤	X8C	□	X3C	`	X79
ë	X53	³	XB3	⌞	XAB	⌟	XBB	±	X9E		X41	Ì	X78
è	X54	¼	XB4	┼	X8F	─	XBF	Â	X62	â	X42	÷	XE1
í	X55	½	XB5	⅞	XFA	€	XFB	Ä	X63	ä	X43	Ï	X77
î	X56	¾	XB6	¤	X9C	┤	XEC	À	X64	à	X44	Ê	X72
ï	X57	⅓	XB7	▉	X9F	├	XEB	Á	X65	á	X45	Î	X76
ì	X58	⅔	XB8	●	XAF	□	XDF	Ã	X66	ã	X46	η	X70
↓	X59	⅛	XB9	£	XFC	▫	X90	Å	X67	å	X47	⅝	XEA
$	XDC	‘	X8D			⅜	XDA	Ç	X68	ç	X48	-	XCF
┬	XCC	’	X9D	Ë	X73	ü	XDB	¥	XFD	▫	XFE	∧	XEF
Δ	XCA	˛	X8E	ª	X9A	\	XE0	}	XD0	{	XC0	Ñ	X69
┴	XCB	¯	XA0					⁄	XCE	®	XEE	↑	X8A

The following table depicts the hexadecimal representations of the characters and symbols on the current print train.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□
1	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□
2	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□
3	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□
4			â	ä	à	á	ã	å	ç	ñ	¢	.	<	(	+	⎪
5	&	é	ê	ë	è	í	î	ï	ì	↓	!	$	*	)	;	¬
6	-	/	Â	Ä	À	Á	Ã	Å	Ç	Ñ	¦	,	%	_	>	?
7	η	°	Ê	Ë	È	Í	Î	Ï	Ì	`	:	#	@	‾	=	"
8	Ø	a	b	c	d	e	f	g	h	i	↑	{	≤	‘	˛	┼
9	▫	j	k	l	m	n	o	p	q	r	ª	}	¤	’	±	▉
A	¯	~	s	t	u	v	w	x	y	z	Ξ	⌞	⌜	[	≥	●
B	°	¹	²	³	¼	½	¾	⅓	⅔	⅛	∨	⌟	¬	]	≠	─
C	{	A	B	C	D	E	F	G	H	I	Δ	┴	┬	∨	⁄	-
D	}	J	K	L	M	N	O	P	Q	R	⅜	ü	$	¶	←	□
E	\	÷	S	T	U	V	W	X	Y	Z	⅝	├	┤	©	®	∧
F	0	1	2	3	4	5	6	7	8	9	⅞	€	£	¥	▫	‰

CONTROL WORD SUMMARY

Break Control Words indicated by "*"

.CW <;|character> 49
The CONTROL WORD SEPARATOR control word defines the delimiter between control commands so that multiple controls and text may be entered on one line.

.DA <1|n|+n|-n|YES|NO> 50
The DARK OUTPUT control word specifies the number of times that Offline Output is to be overstruck.

.DC .....* 179
The DON'T COUNT control word causes all output produced directly by the immediately following text line to not be counted for purposes of page length control.

.DH <SET m> n <<SKBF n> <SPAF n> <TCOF n> <TCIN n>
<BR|NBR> <OJ|NOJ> <PA|NPA> <TC|NTC>
<TO|NTO> <TS|NTS> <UP|NUP> <US|NUS>> 51
The DEFINE HEADING control word defines the formatting conditions for the various heading levels in the Tables of Contents.

.DM name /line1/.../linen</> 53
name <BEGIN|END>
name DELETE
The DEFINE MACRO control word defines a sequence of input lines to be invoked by "name" as a user-defined control word or as a .SI operand.

.DO <BEGIN|END> 54
The DO control word may be used following a THEN or ELSE control word to allow multiple input lines to be conditionally included.

.DS .....* 179
The DOUBLE SPACE control word causes a line to be skipped between lines of output except in footnotes.

.EB <1|n> </S1/S2/S3/> 55
The EVEN BOTTOM control word is used to define three items of title information to be printed at the bottom of even-numbered pages.

.EF <YES|NO> 56
The END OF FILE control word causes immediate termination of the current input file and resumption of the next higher level file if any, or termination of processing if none.

.EL line 57
The ELSE control word causes an input line to be conditionally included depending on the truth value of a previous IF control word.

.LS <0|n|+n|-n|YES|NO> .....* 104
The LINE SPACING control word causes zero or more blank lines to be skipped between each line of formatted output except when within a footnote.

.MC .....* 105
The MULTIPLE COLUMN control word restores multi-column output after it has been suppressed by .SC (SINGLE COLUMN).

.MS <ON|OFF> 106
The MACRO SUBSTITUTION control word prevents user-defined macros (see .DM) from being invoked.

.OB <1|n> </S1/S2/S3/> 107
The ODD BOTTOM control word is used to define three headings to be printed at the bottom of odd-numbered pages.

.OC <char|ON|OFF|n> <c1 ...> 180
char/<ON|OFF|n> <c1 ...>
The OVERLAY CHARACTER control word causes characters in input text lines to be overlaid with a backspace and user defined character or itself.

.OF <0|n|+n|-n> .....* 108
The OFFSET control word causes all but the first line of a section to be indented.

.OJ <1|n|YES|NO|line> .....* 110
The OUT JUSTIFY control word causes the specified lines to be aligned to the outside of the page.

.OO <1|n|ON <string>>
<OFF|DELETE> 111
The OUTPUT OVERLAY control word causes blanks in formatted output records to be overlaid with a user defined string.

.OT <1|n> </S1/S2/S3/> 112
The ODD TITLE control word is used to define three headings to be printed at the top of odd numbered pages.

.OV <1|n|ON <string>> 113
<OFF|DELETE>
The OVERLAY control word causes following input text lines to be overlaid by the contents of the text line following this command.

.PA <%+1|n|+n|-n <m|+m|-m>> .....* 114
<YES|NO|ODD|EVEN>
The PAGE control word causes an output page eject and optionally resets the page number.

.PE <1|n|YES|NO|DELETE> 116
The PERFORM control word causes remainder of the same or the next input line to be executed multiple times.

.PL <66|n|+n|-n> .....* 120
The PAGE LENGTH control word specifies the physical size of the output page in units of physical output lines.

.PN <ON|OFF|OFFNO> 121
<ARABIC|ROMAN <LOWER|UPPER>>
<PREFIX|SUFFIX <string>>
<FRAC|NORM>
The PAGE NUMBER control allows the user to control the incrementing and formatting of page numbers.

.PP <line> .....* 123
The PARAGRAPH START control word defines the beginning of a new paragraph.

.PS <%|character> 124
The PAGE NUMBER SYMBOL control word defines the character in headings and footings to be replaced by the current page number.

.PT <line> 125
The PUT TABLE OF CONTENTS control word adds an input line to the Table of Contents.

.PU <1|n> <line> 126
The PUT WORKFILE control word allows the user to output records of data to an output file. This file may be imbedded later or be kept.

.QQ <YES|NO> .....* 127
The QUIT QUICKLY control word causes immediate termination of all input and output file processing.

.QU <YES|NO> .....* 128
The QUIT control word causes immediate termination of all input file processing.

.RA <1|n|YES|NO|line> .....* 129
The RIGHT ADJUST control word causes the next input line to be right adjusted in the output line.

.RC n <ON|OFF|ON/OFF> 130
n <char|SET string>
The REVISION CODE control word allows selected portions of the total document to be marked in the left margin with a revision code character.

.SS .....* 181
The SINGLE SPACE control word causes output to be single spaced.

.SU <1|n|ON|OFF|line> 149
<UPPER|NOUPPER>
<TRACEON|TRACEOFF>
The SUBSTITUTE SYMBOL control word causes subsequent input text and control lines to be reformatted by substituting the current values of specified reference names in the line and to be processed as if it had been in the input.

.SY <line> 150
The SYSTEM control word passes a command line to the host operating system.

.TB <n1 n2 n3 ...> .....* 151
<<'string'|char/>n<L|R|C|'char> ...>
<SET <char>>
The TAB SETTING control word specifies the tab stops to be assumed for the following lines when converting the TAB character (X'05') generated by the terminal TAB key into the appropriate number of spaces.

.TC <1|n|* <CONTENTS|line>> .....* 155
<ADD <m ... >>
<PURGE>
The TABLE OF CONTENTS control word causes the specified Table of Contents to be printed.

.TE <1|n> 159
The TERMINAL INPUT control word allows the user to enter control or input lines during processing of the input file.

.TH line 161
The THEN control word causes an input line to be conditionally included depending on the truth value of a previous IF control word.

.TI <s <s|t>> 162
<<s1 t1> <s2 t2> ...>
<SET <char>>
The TRANSLATE ON INPUT control word allows the user to specify an escape character and the contents of a translate table to be used on input lines.

.TM <6|n|+n|-n> .....* 163
The TOP MARGIN control word specifies the number of lines to be placed between the physical top of the output page and the first line of text.