ANL-AMD-TM-294

ARGONNE NATIONAL LABORATORY

WYLBUR REFERENCE MANUAL

by

Ron F. Krupp
Paul C. Messina
James M. Peavler
Steve Schustack
Tom Starai

U of C-AUA-USERDA

APPLIED MATHEMATICS DIVISION

DISCLAIMER

The facilities of Argonne National Laboratory are owned by the United States Government. Under the terms of a contract (W-31-109-Eng-38) between the U. S. Energy Research and Development Administration, Argonne Universities Association and The University of Chicago, the University employs the staff and operates the Laboratory in accordance with policies and programs formulated, approved and reviewed by the Association.

MEMBERS OF ARGONNE UNIVERSITIES ASSOCIATION

The University of Arizona
Carnegie-Mellon University
Case Western Reserve University
The University of Chicago
University of Cincinnati
Illinois Institute of Technology
University of Illinois
Indiana University
Iowa State University
The University of Iowa
Kansas State University
The University of Kansas
Loyola University
Marquette University
Michigan State University
The University of Michigan
University of Minnesota
University of Missouri
Northwestern University
University of Notre Dame
The Ohio State University
Ohio University
The Pennsylvania State University
Purdue University
Saint Louis University
Southern Illinois University
The University of Texas at Austin
Washington University
Wayne State University
The University of Wisconsin

WYLBUR REFERENCE MANUAL

APPLIED MATHEMATICS DIVISION

TECHNICAL MEMORANDUM NO. 294

ARGONNE NATIONAL LABORATORY

APRIL, 1977

Revised and Edited from
WYLBUR Reference Manual
by

Ron F. Krupp
Paul C. Messina
James M. Peavler
Steve Schustack
Tom Starai

Table of Contents

PREFACE

WYLBUR was obtained by Argonne from the Stanford Linear Accelerator Center (SLAC) and Stanford University. This reference manual is a revised version of the Stanford WYLBUR Reference Manual. Permission to reproduce and revise this copyrighted material was graciously granted by SLAC. The Argonne changes to WYLBUR consist mainly of different SIGNON procedures, dataset naming conventions, and ASP-related commands. Changes to the Stanford WYLBUR Reference Manual include a complete reorganization and revision of chapters 2 and 3 with a view toward making the manual more generally useful to Argonne users. Chapter 4 has been modified primarily by deleting references to commands not available at Argonne, changing language or syntax to make WYLBUR commands more similar to TSO commands, and adding examples of some commands, to clarify their use at Argonne. Most of the commands have been checked to assure that they perform as they are described. If you find errors or inaccuracies in this manual, please advise Users' Services (221, A-142, ext. 4282), so that the document can be corrected.

Stanford WYLBUR was originally designed by Roger Fajman and John Borgelt under the direction of Rod Fredrickson in June 1967 and implemented at the Campus Facility of the Stanford Computation Center. Based on the Campus WYLBUR, an MVT version of WYLBUR was developed under Roger Fajman at the NIH Computation Center at Bethesda, Maryland. In January 1970, the NIH MVT version was installed on the IBM 360/91 at the SLAC Center of the Stanford Computation Center. Since 1970, John Halperin and Joe Wells have made many improvements to produce the current version of the Stanford WYLBUR system. During 1974, WYLBUR was converted to run under IBM's operating system VS2 R1.6 by Carol Farlow, Vicki Griffeth, John Halperin, and Carol Lennox.

Many extensions to the WYLBUR system were originally designed and developed at Stanford; Brian Cox, Rich Levitt, and Ed Russell made significant contributions. In addition, some extensions were originally implemented at NIH and Columbia University.

This document was revised using WYLBUR, as adapted for use at Argonne National Laboratory.

.0 Introduction

WYLBUR is a system for manipulating various kinds of text, such as computer programs, manuscripts, letters, forms, proposals, articles, reports, or nearly any kind of document you might want to write. Its on-line interactive text editing capabilities allow the user to create, change and correct text, and to search and display it. WYLBUR also has facilities for job submission and retrieval from remote terminals that make it possible for a user to inquire about the status of any job in the system, cancel jobs that are executing or awaiting execution, reroute output, raise job priority, or get information on the backlog of batch jobs. WYLBUR also has excellent recovery capabilities and a fast response time.

This manual describes the WYLBUR version currently used at Argonne. Versions at other installations differ principally in the signon sequence, dataset naming conventions, and in the commands related to batch jobs.

Features of WYLBUR Language

The WYLBUR command language is easy to learn and easy to use. WYLBUR is easy to learn because it has an English-like command structure, most of its command syntax and vocabulary closely resembling an English imperative sentence. For example, the command CENTER 1/8 LENGTH 72 means center lines 1 through 8 within 72 column spaces.

Where no ambiguity is likely to arise many command parameters may be specified in any order, and most may be abbreviated with three or more characters. WYLBUR is a forgiving text editor that allows the user to make errors without ruining his text, and provides for simple correction of most errors. Typing errors can be corrected by backspacing over the error and typing the correct letters, or the attention key may be used to reprompt for an entire line. A feature called MODE RETRY allows one to use WYLBUR's powerful editing facilities to correct invalid commands, or to edit the last command issued. When the user forgets to give WYLBUR all the information necessary for a given command, WYLBUR prompts him for the missing data. A preprocessor and special files called EXEC files allow frequently used sets of commands to be saved and used at will.

Should WYLBUR or the operating system fail, or should the communication connection be broken while the user is working on a file, WYLBUR will usually retain the file in a special kind of dataset called ACTIVE (See Section 2.5.3).

Using this Manual

This manual is intended primarily as a reference manual; thus, examples of WYLBUR commands have been kept to a minimum. Section) describes the WYLBUR environment and features, while Section 3.0 ves an overview of specific WYLBUR capabilities, describing the unction of each WYLBUR command. Section 4.0 discusses the WYLBUR commands in detail. The commands are presented in alphabetic order to simplify finding information about specific commands.

Section 5.0 describes the WYLBUR preprocessor and those WYLBU commands that are related to the preprocessor. Because of th nature of this material, section 5.0 is more tutorial than reference.

2.0 WYLBUR Description

This section describes how to get an account number, the WYLBUR terminal features, WYLBUR's use of certain keyboard keys, the WYLBUR file system, and other aspects of WYLBUR operation.

2.1 Getting an Account Number

Timesharing accounts may be obtained from the Data Management and Accounting Services office, Bldg. 221, office A-143, extension 4979. You will need to supply an activity code and cost center to which charges may be billed. All timesharing users will be automatically enrolled in both WYLBUR and TSO, and the same userid and password will be valid on both systems.

2.2 The WYLBUR Terminals

Currently, WYLBUR communicates with two major kinds of terminals:

1) ASCII type terminals such as the Texas Instruments Silent 700, Tektronix 4010, 4014, 4016, 4023, teletypes, and others.
2) IBM 2741 and MCST terminals.

The characteristics distinguishing the two kinds are the character set, the transmission code, and line speed. The transmission code is not likely to be of any concern to most users. The line speed for ASCII type terminals may be 10, 30, or 120 characters/second; for the 2741-type of terminals it is fixed at 15. The character set differences are described below. The IBM 2260's cannot communicate with WYLBUR.

2.2.1 Typewriter Terminals

Among the available 2741-type of terminals are the EBCDIC and the Correspondence. The two have slightly different keyboard arrangements, the keyboard arrangement of the Correspondence terminal being identical to the IBM office Selectric typewriter. The EBCDIC terminal, with a slightly different keyboard, is the most commonly used.

IBM 2741 terminals and MCST terminals can be linked to the central computing system with a standard telephone line or they may be disconnected and used as standard typewriters. The typewriter terminals, when connected to the computer, have several keys that function differently from those of the standard typewriter. The use of these keys, and the additional ATTN key, is discussed in section 2.3.

2.2.2 ASCII Terminals

Differences among various manufacturers' terminals of the ASCII type tend to be more numerous than those among the 2741 type of terminals. There will be differences in character sets, keyboard arrangement, line speed, and hardware features. The following is a general description of ASCII-type terminals.

Current line speeds offered are 10, 30, and 120 characters/second (110, 300, and 1200 baud).

- If the terminal has physical tab stops, logical tabs may be used. If the terminal does not have physical tabs and you wish to list already tabbed material, we suggest that you give the command SET SLOWLIST. (See Section 2.3.5).

- There is no difference between upshift and downshift backspaces. If the SET BACK command is issued, any backspace will be accepted as the backspace character (X'16'). Characters entered in error can then only be "erased" by the use of the CTRL-X (CAN) character.

- Character set differences include:

  The cent sign ¢, is represented on the ASCII terminal as a caret (i.e., an isosceles triangle with the base missing) or an upward arrow.

  The NOT sign (¬) is represented on the terminal as a tilde (i.e., an approximation sign).

  The OR symbol (|) is often shown on the keyboard as two vertical dashes, one above the other.

The Tektronix 4012 Scope is an ASCII terminal that is similar to the typewriter terminals in that it has a keyboard for character input, but the input is displayed on a screen rather than printed on paper. Its unique features are as follows:

- The screen is 74 characters wide; lines longer than this will wrap around onto the next line. The screen can hold 35 lines. When the screen is full, lines will wrap around to the top of the display and each line will start on the right half of the screen. Hitting the RESET-PAGE button will clear the screen at any time, but such clearing must be done by the user.

- If a terminal is left idle for more than 90 seconds, the scope will go into a "hold" status (the character brightness dims); if the shift key is depressed or any character is entered, the scope will return to its normal display status. Keeping the same image on the screen for 15 minutes in display status, or 1 hour in hold status, could damage the screen. Therefore, users should always clear the screen by hitting the RESET-PAGE button between sessions.

- The scope has no facility for setting hardware or physical tabs (as a typewriter does). (See Section 2.3.5).

- WARNING: If you hit the ESCAPE key, the character represented by hex 27 will be entered. Unfortunately, on the scope this character is displayed as a null character. On an offline printing, it will usually show up as a blank. On the typewriter terminals, it will print as the character #. If you wish to remove the escape character, you can give the command: CHANGE '<hit ESCAPE key>' TO ' ' which will change it to a blank, or CHANGE '<hit ESCAPE key>' TO '' which will delete the escape character.

2.3 Use of Non-typing Keys in WYLBUR

The non-typing keys on the terminal typewriter keyboard have special functions in WYLBUR that are quite different from those on an ordinary typewriter.

2.3.1 Carriage Return (or CR)

This key returns the printing head to the left margin as on any typewriter, but, in WYLBUR, it has additional functions. If a command was being typed in response to a COMMAND prompt, CR indicates that WYLBUR is to begin executing the command. In collect mode a CR at the end of the line signals WYLBUR to store the line and prompt for a new one. Finally, CR in response to a READ command in an execute file (see section 4.3) terminates the string typed at the terminal, and causes WYLBUR to proceed to the next command. No command typed into the terminal is activated, and no line of text typed at the terminal is copied, until the CR is hit.

2.3.2 Backspace

The backspace (or CNTRL-X) key moves the printing head one space to the left, and erases whatever was in that space from the computer's memory. A whole line of type can be erased from the computer's memory by backspacing over it, but backspace cannot be used to erase characters in lines already terminated with a carriage return (CR). Also, the user must be careful not to backspace into the line number that WYLBUR prompted because any characters subsequently typed in those spaces will be lost. The command SET BACK (see section 4.3) changes the function of the backspace key so that an upper case backspace (shift and backspace on 2741 terminals) will backspace without erasing the characters from the computer's memory and will enter a backspace character in the text. This makes it possible, for example, to obtain an on-line or off-line copy of a file that contains underlining or overprinted characters. (Note, however, that ASCII terminals require special treatment. When SET BACK is in effect on an ASCII terminal any backspace (or CTRL-H) will be accepted as a backspace character. Characters can then be erased only by using the CTRL-X key.) For more information see the LIST command in section 4.3, particularly the parameter CC, BACK, and OFFLINE.

2.3.3 BREAK or ATTN

The BREAK key (or the ATTN key on some terminals) has a variety of functions. When the BREAK key is pressed WYLBUR generally responds by typing "***" if it is a line of input that is being interrupted, or "..." if it is a line of output.

Hitting the BREAK key or the ATTN key as the first character in response to a prompt

- Can be used to change from the line number prompting to command prompting and vice versa. That is, hitting the ATTN key as the first character in response to the command prompt causes WYLBUR to prompt with a line number. Hitting the ATTN key as the first character in response to a line number prompt causes WYLBUR to prompt for a command.

- Typed as the first character in answer to permission prompts, information prompts, and programmed prompts, the BREAK key causes WYLBUR to abort the command and give the command prompt (see section 2.5.2).

Hitting the ATTN key at any other position in the line

- Can be used to discard the entire line of text or a command that the user is currently typing.

- Can be used to interrupt any listing or typing WYLBUR does at the terminal.

- Can be used with special characters in the EDIT and MODIFY commands to do certain special tasks (see section 4.3).

- Causes an execute break if an execute file is being used (see the EXEC command in section 4.3).

- Stops execution of most commands.

- Can be used with special characters to enter retry mode (see section 2.5 and the command SET MODE RETRY in section 4.5).

2.3.4 Shift

There is a distinction between the effect of the shift key at the terminal and its effect on a WYLBUR text. The shift key operates just as on any typewriter as far as the terminal copy is concerned, but its effect on the WYLBUR text depends on the command SET UPLOW or SET UPPER.

If UPPER case (the default at sign-on time) is in effect, all lphabetic characters are converted to upper case including those yped at the terminal in lower case.

If UPLOW case is in effect (see SET UPLOW in section 4.3), th fcllowing is true:

- All text the user types after the COLLECT command and before he gives another command remains in the case typed (i.e., probably a mix of upper and lower case).

- All text the user types as a result of the editing commands MODIFY, EDIT, REPLACE, CHANGE and INSERT remains in the case typed.

- All command words and command parameters other than strings are recognized regardless of case. For example, in the command:

  change 'Able' to 'Mable' in 28/60

  'Able' and 'Mable' are strings and the characters will be recognized as upper/lower case, while the commands CHANGE TO will be recognized as commands whatever their case at the terminal.

2.3.5 Tabs (SET TABS)

There is a distinction between setting the logical (computer) tabs and setting the physical (terminal) tabs. The physical tabs are set, cleared, and used as on any other electric typewriter. The logical tabs are set with the command SET TABS and cleared with the command CLEAR TABS. If the user uses the physical tabs when the corresponding logical tabs have not been set, WYLBUR will not accept the line but will give the diagnostic message:

UNSET TABS-RETYPE

The user should then SET TABS, and also set the physical tabs on his terminal to correspond to the logical tabs, or simply space over the desired number of spaces.

NOTE: Many terminal types have no facility for setting hardware tabs; we recommend, therefore, that you do not set any logical tabs. If you wish to list already tabbed material and do not have tabs set, we recommend you first give the command: SET SLOWLIST (see section 2.8).

When the logical and physical tabs are in agreement and FASTLIST (WYLBUR'S default value), is in effect, tabs can be used to speed on-line listing by skipping long strings of blanks.

To set the physical and logical tabs follow this procedure. Clear the physical tabs first by spacing to the end of a line, then depress the tab CLEAR switch and carriage return or the ATTN key at the same time. Now the logical tabs may be set in one of two ways, the long way and the short way.

You can set a maximum of 16 tabs. If more tabs are set, WYLBUR recognizes only the first 16; the rest are ignored.

To set tabs the long way, give the command SET TABS. WYLBUR's response will be:

```
TYPE A "1" BENEATH EACH POSITION AT WHICH YOU SET A TAB.
          12345678911234567892123456789312345678941234567 89...
    START?
```

The user's response should be as follows: starting at the left, space over to each successive column in which you want a tab set, press the SET toggle switch on the keyboard, and then type the number 1.

WYLBUR will then respond by verifying the tab positions by typing a "1" in each column position that a tab has been set.

The example below shows the successful setting and verifying of the tabs in columns 10, 20, 30.

```
        START?          1         1         1 (CR)
                        1         1         1
```

To set tabs the short way, you may append to the SET TABS command a list of numbers representing the column numbers for the tab settings. Because the physical tab settings at the terminal must correspond to the information given when using this form of the SET TABS command, WYIBUR will type two verification lines. WYLBUR will first space out to each of the logical tab positions and type a 1. Then it will tab to each of the physical positions and again type a 1. If the 1's are not directly under each other, set the physical tabs to correspond to the first verify line. You can check that it is correct by entering a SHOW TABS VERIFY command.

```
COMMAND? SHOW TABS VERIFY (CR)
VERIFY              1         1         1
                    1         1         1
CCMMAND?
```

2.4 Establishing and Terminating Communication with the Computer

2.4.1 Sign-On Procedure

You must successfully sign-on at a terminal to use the WYLBUR system. The sign-on procedure varies slightly depending on whether the terminal is a 2741-type, Tektronix 40xx, or ASCII-type. Follow the table below for your terminal type.

2741-type terminals	ASCII terminals except Tektronix	Tektronix 4012 Scope
1. Place COM-ICL switch in COM position.	1. Place LOCAL-LINE switch to LINE. Half or full duplex settings both work at 110 or 300 baud.	
2. Turn on power. (Switch is on right of keyboard on a 2741, and on right side of the terminal case on many ASCII terminals).		2. Turn on power. (Switch is under keyboard.) Hit PRESET-PAGE key, upper left keyboard
3. Dial one of these numbers: 65, 4304, 4955, 5394, 5411, 4821, 3697, 2922, 3920 (for 110, 134.5, or 300 baud).		3. For 1200 baud full duplex dial 66 or 4869. For 1200 baud half duplex dial 5254.

If you receive a busy signal, hang up, wait, and try again. If you hear a whistle from the computer proceed with step 4 of the signon procedure.

4. Couple: a) If the terminal has an acoustic coupler, place the handset in the coupler. b) If phone has an exclusion button, pull it up. This button is a white one in the cradle of a desk phone, or a clear one on a wall phone. c) If the terminal is attached to a dataphone press the "originate" button and place the handset back onto the cradle.
5. For 110 and 300 baud and for 2741 terminals type PWPW, (CR). This enables the system to determine the speed and character-set of your terminal (from the capital P), and select WYLBUR (from the capital W). For 1200 baud terminals (full or half duplex), type W, (CR) to select WYLBUR.

6. On 2741 terminals type ATTN or (CR) when prompted by H?H?H?H?	6. Type 'S' or 'SIGNON' when prompted by EEEEZZzxs Type 'COR' if you are on an IBM 2741 Correspondence terminal.

The system will then type a message containing the date and time followed by prompts for userid and password:

```
ARGONNE LINE x  hh:mm:ss  mm/dd/yy

USERID?          enter Bnnnnn plus carriage return
PASSWORD?        enter password plus carriage return
```

This is followed by a message on your current track limit and usage and your present ration usage:

```
STORAGE= nnnn TRKS, LIMIT= mmmm TRKS
RESOURCE LIMIT=   .nn UNITS,   RESOURCES CHARGED=   .nn UNITS
```

Each timesharing user is allowed a certain number of 2314-equivalent tracks of long-term shared storage (300 tracks is the default). See Chapter 10 of the current ANL Computer Users' Guide for further information. The system may follow this with another line giving the message of the day which includes any special information for users.

After WYLBUR has received and verified the userid and password the sign-on procedure is complete. At this point one of two things may happen. WYLBUR may respond with the prompt COMMAND? This signifies that the sign-on procedure is complete and that the terminal is now ready to accept WYLBUR commands. Or, before turning control over to the user, WYLBUR may automatically execute from the member LOGON in the user's LIB PDS. However, in order for this to happen the user must have (1) created the member LOGON of his LIB PDS and (2) given the command SET EXEC AUTO in an earlier session. See the description of the command SET EXEC in section 4.3 for complete details.

To restart the sign-on procedure, the user may type SIGNON in response to any prompt. Striking the ATTN key in response to any sign-on prompt causes WYLBUR to disconnect the line.

Provided that the current user has not yet signed off, you may sign on to the same terminal without dialing up, by entering the command:

```
SIGNON (or LOGON)
```

WYLBUR will print all the statistics from the session just finished and then prompt for your userid. You may then continue the normal sign-on procedure. If you know the previous user does not want his active file saved, specify the parameter "CLEAR" on the LOGON command.

2.4.2 Sign-Off Procedure

To end a session, issue a LOGOFF, LOGOUT, or SIGNOFF command. WYLBUR will reply with various statistics on the session as shown and disconnect the telephone line.

```
COMMAND? LOGOFF (CR)
14.73 SECONDS EDITING TIME
2153 PAGEREADS 1887 PAGEWRITES
127 DISK READS 240 DISK WRITES
ELAPSED TIME = 01:03:18
END OF SESSION
please enter tt for tso or ww for wylbur
```

The editing time is the actual computer (CPU) time used during editing. Elapsed time, which is reported in hours, minutes and seconds, is the time during which you were logged on. Page reads and writes are counts of the I/O operations performed to execute the various commands during the session. Disk reads and writes are counts of the I/O operations performed to read and write external files. The 'please enter' prompt allows you to LOGON to TSO or to log back onto WYLBUR without hanging up the phone and redialing. If you do not begin LOGON within 18 seconds, the connection is automatically broken and LOGOFF is complete.

WYLBUR keeps track of activity at a terminal and if nothing has been typed in at a logged-on terminal for a period of five minutes, WYLBUR will ask if you are still there.

```
COMMAND? ###
ARE YOU STILL THERE?
```

A carriage return is a sufficient reply. If you do not respond, WYLBUR will give you another two minutes and ask again. If no reply is made, WYLBUR will wait another two minutes, then logoff the terminal without further queries.

If the user wishes not to be logged off automatically for inactivity, and if there are sufficient free lines (the current limit is 30), he may issue the command SET NOTIME. To find out if enough lines are free that NOTIME will be honored use the SHOW MINFREE command. If you issue the SET NOTIME command when there are not enough free lines WYLBUR will issue the information prompt:

```
nnLINE(S) FREE, 30 MINFREE. NOTIME NOT CURRENTLY HONORED.
```

A LOGOFF command will not be honored if there is an active file. In that case, WYLBUR will prompt for permission to clear the active file. OK TO CLEAR?

A YES, OK, or CLEAR response causes a CLEAR and normal logoff. Any other response continues the session. If you are logged off by WYLBUR or if your session is terminated abnormally by communication problems, the WYLBUR system will usually be able to save and recatalog the current active file as an external file named ACTIVE (see section 2.7.4)

f you do not want to save your current active file at the end of a session type

LOGOFF CLEAR

and the system will log you off with no further questions.

2.5 Dialogue between the User and WYLBUR

WYLBUR communicates with the user via prompts and messages. The prompts may be for commands, text lines, or additional information necessary to execute a command. The messages may be error messages or informative messages. The user communicates with WYLBUR via commands and responses to prompts. The following sections describe commands and prompts in greater detail.

2.5.1 Commands

The user may enter commands individually from the terminal or collect them into a file which he may then execute (see section 2.6). A command always has these general attributes:

- A blank, comma, and equal sign (,=) are recognized as equivalent separators. Thus,

  ```
  SET LENGTH 60
  SET LENGTH,60
  SET LENGTH=60
  ```

  are all valid commands to set line length to 60 characters.

- The slash (/) designates a range of lines. For example, the command LIST 12/14 means to list all lines whose line numbers lie in the range from 12 to 14 inclusively.

- A semi-colon (;) terminates the command; anything that follows is treated as a comment. If a command begins with a semi-colon, the whole line is a comment. This feature is most often used to explain the action of execute files. Typing the word COMMENT, followed by string a, at the beginning of a line in the execute file causes string a to be typed out at the terminal when the execute file is executing.

- Generally, command words and parameters may be abbreviated to the first 3 or more characters. Exceptions to this rule are given in Appendix A.

Command RETRY

Often a command may fail to execute because of a syntax error or because a relevant parameter has been omitted. Similarly, the user may often wish to give the same command a number of times with only minor changes. In both cases, the user could save time if he had access to the original command and could modify rather than retype it. This is exactly the facility provided by the retry capability.

Retry comes in two forms: automatic and non-automatic. Automatic retry must be requested explicitly with the command SET MODE RETRY. Non-automatic retry is available at all times, even when the user has requested automatic retry. You can be prompted to modify a command by typing @(ATTN).

Automatic retry causes WYLBUR to prompt the user to change a command whenever it fails to execute. The user may then change the command and reissue it.

Retry can be requested for commands that did not fail to execute. When automatic retry mode is in effect, the user must manually request retry (by typing @(ATTN)) for commands he wishes to change and re-execute. When automatic retry mode is not in effect, the user must manually request retry for each command that he wishes to reissue, whether it failed to execute or not. Once prompted with the ALTERS? prompt the user edits the command just as he edits a line with the MODIFY command.

See Section 4.3, SET MODE RETRY, and MODIFY.

2.5.2 Prompts

There are two kinds of prompts: prompts from WYLBUR and prompts programmed by the user.

2.5.2.1 Prompts from WYLBUR

This section will distinguish four classes of prompts from WYLBUR:

1) Prompts for commands (command prompts)

2) Prompts for data to be entered into the active file (line number prompts)

3) Prompts for the user's OK to do something (permission prompts)

4) Prompts for information that WYLBUR needs to execute a command (information prompts)

Command prompts

After the log-on procedure is complete, WYLBUR prompts "COMMAND?". The user's response can be a WYLBUR command or hitting the BREAK key (or ATTN key on IBM 2741-type terminals) (see "Line number prompts" below). The user may cause the prompt "COMMAND?" to be shortened to "?" by giving the command SET TERSE (see section 4.3 for a complete description of this command).

Line number prompts

WYLBUR responds with a line number to any one of these commands: COLLECT, INSERT, and REPLACE. For example, to the command INSERT 5 WYLBUR responds "5. ?". The user then can respond with the text to be inserted at line number 5 in the active file.

During most of a terminal session, a user will be typing either commands or data. The ATTN key has a special function that facilitates switching between command entry and data entry. Striking the ATTN key in response to the command prompt will cause WYLBUR to respond with a line number. Striking the ATTN key in response to a line number prompt will cause WYLBUR to respond with the command prompt. For example, assume the user has given the command COLLECT and WYLBUR has prompted with several line numbers to which the user has responded with data to be entered into the active file. Now, the user wants to give a command. He must strike the ATTN key to get the command prompt. (The ATTN key also has other functions, see section 2.3.3).

Permission prompts

These require the user to give permission to WYLBUR to do something. For example, assume the user has an active file and wishes to make another file active. If he forgets to clear (i.e., erase) the active file, WYLBUR will prompt "OK TO CLEAR?". The user may indicate a positive response in a number of ways: YES, OK, CLE, CLEA, or CLEAR. The latter three are valid abbreviations of the keyword CLEAR. If he does not want his active file cleared he can respond by hitting the BREAK or ATTN key or (CR) and the command will be aborted, and the user will be reprompted with COMMAND? Thus, this permission prompt prevents a user's accidently erasing a valuable dataset when he calls up a new active file. (See section 2.3.3 for more information on the BREAK or ATTN key).

Information prompts

These request the user to respond with specific information. For example, the user may tell WYLBUR to save his active file, but forget to say where. If WYLBUR can't determine where the file is to be saved, the prompt "VOLUME?" will appear. The user must then respond with a valid volume name in order to save the file. Thus, prompts in this class are usually the result of a command that didn't specify all the necessary information.

2.5.2.2 Programmed Prompts

Prompts may be programmed by the user, most often in an execute file. For example, the command READ will generate a default prompt of "ENTER?" unless the user specifies a different prompt with the PROMPT parameter of this command. The nature of the response to this kind of prompt is determined by the user.

2.6 WYLBUR Files

A WYLBUR file is a collection of lines, each of which may contain from 0 to 133 characters. For each line, WYLBUR assigns a unique line number from 0 to 99999.999. The lines in the file are arranged in ascending order by line number. A file can be a program in FORTRAN, PL/I, or any other language, a section of text such as a letter, or even a series of WYLBUR commands. WYLBUR distinguishes three kinds of files, each with unique attributes. They are the active, external, and execute files.

The active file is a file which is held in the computer's memory so that the user can make alterations and additions to it. When it is not being held in primary memory during the actual execution of a command, the active file is stored on an auxilliary storage device, such as a drum. Maximum workable size of the active file is about 6000-8000 card images or 4000-6000 print lines. Most editing commands (see section 3.1) can operate only on the active file. By executing the SAVE dsname command the user creates an external file.

Any file that is stored on an auxiliary storage device is called an external file. The maximum workable size for an external file is the same as for an active file. These external files are either sequential data sets or sequential members of partitioned data sets (also called libraries). An external file cannot be modified until it is copied into an active file by the USE dsname command. An external file is identified by a data set name (dsname) that must follow established rules (see section 2.7.1). Guidelines for using sequential and library space are discussed in section 2.7.5.

An external or active file becomes an execute file when the EXECUTE command is given. The execute file contains WYLBUR commands that WYLBUR executes sequentially unless an IF command specifies branching. Any WYLBUR command can be in the execute file, but the file can only contain about 100-150 commands. At Argonne, because WYLBUR is operated at a lower queue priority than TSO, only eight WYLBUR execute commands will be handled at once. The WYLBUR execute file is then returned to the end of the queue to await further execution. This slows the execution of WYLBUR files only if there is heavy TSO use at the time. An execute file is held in memory like an active file. The user can work with an execute file and a active file at the same time, (See Section 5 for details).

.7 WYLBUR DATA SETS

This section describes the conventions used for naming WYLBUR data sets, the WYLBUR recovery procedure, how to locate a data set whose volume residence is not known, how to create and use a partitioned data set, and how to manage the limited WYLBUR disk space effectively.

2.7.1 WYLBUR Data Set Naming Conventions

All datasets on shared volumes must have names whose first part (high-level index) is Bnnnnn, where Bnnnnn is the userid for a valid timesharing account. The dsname of all such datasets is of the form:

Bnnnnn.name

where:

name — consists of no more than 37 alphanumeric characters. Periods may be used to break the character strings into levels of no more than eight characters, but the first character following each period should be alphabetic.

*Note: The standard WYLBUR dataset is sequential. For instructions on how to create and use a partitioned dataset or PDS (called a LIB by WYLBUR) see sections 2.7.5 and 4.3.

This form of a WYLBUR dsname is called a fully qualified dsname. However, when specifying a dsname in a WYLBUR command, the user can let WYLBUR use defaults to supply a portion of the dsname such as "Bnnnnn." or "Bnnnnn.LIB". See "dsname" in section 4.3. Names of this form are automatically formed when WYLBUR or TSO are used to create datasets. For example, if user B12345 is signed on to WYLBUR and issues the command

SAVE X.Y ON DSKnnn

a dataset with the name B12345.X.Y will be created on DSKnnn and cataloged.

If you have already SAVEd the dataset X.Y and now want to replace it with an updated version, type

SAVE X.Y REPLACE or SAVE X.Y SCRATCH

This command erases the old version and replaces it with the new. If you try to replace a dataset that does not already exist (for example, if you have made a typing error in your command such as SAVE Y.X REPLACE), WYLBUR will respond with:

Bnnnnn.Y.X NOT ON CATLG,

and will not save it. You must retype the command with the correct dsname.

2.7.2 Disk storage for WYLBUR datasets

Disk storage available to WYLBUR users now includes the following:

PACK70/75
PACK87/89**
DSK200/209
DSK220,221.

**Note: PACK87/PACK89 are short term 7330 (contents are scratched seven days after creation) and therefore probably not desirable for most WYLBUR datasets. PACK70/PACK75 are Long 7330 and DSKnnn are Long 2314. See Chapter 10 of the current ANL Computer User's Guide for a description of storage disks.

To find how much space is available on a given disk type the command SHOW SPACE ON DSK (or PACK)nnn.

Datasets on all public volumes must be cataloged. For this reason the default 'volume' for searching datasets at ANL is the catalog. I.e., SET VOLUME CATALOG is in force at SIGNON time. For a complete discussion of data management policies and conventions at Argonne, see Chapter 10 of the ANL Computer User's Guide.

The name ACTIVE is a special dsname; see section 2.7.4 for the use WYLBUR makes of this name.

2.7.3 Locating a Data Set

To determine the names of your datasets on a given volume (say DSK221), type

SHOW DSNAMES ON DSK221

To determine the names of all cataloged datasets whose names start with Bnnnnn, where Bnnnnn is the signed on userid, type

SHOW CATALOG

This will cause the second index level of all your cataloged dataset names to be listed, including TSO files. Your badge number is the first level of your index, so if you give the command SHOW CAT, you might receive a message such as

Bnnnnn JUNK - ON DSK203
LIB - ON PACK70
STUFF - INDEX

UNK, LIB, and STUFF are all second level index names. The name STUFF - INDEX shows that there are at least two datasets with the second level name STUFF. To find out what they are issue the command SHOW CAT FOR STUFF. You might get a message like

```
Bnnnnn.STUFF.MORE - INDEX
TUFF - ON DSK203
```

This message gives you the third index level and indicates that there are at least two datasets named STUFF.MORE and one named STUFF.TUFF.

The batch cataloged procedure MYFILES2 also provides a way to get a complete report of all your datasets on both the batch and timesharing systems. It requires the submission of a batch job. See 'AMD Topical Note 10' for details.

2.7.4 Recovery

Recovery of the current active file is performed whenever WYLBUR fails, after some types of OS failure, or the communication connection is broken, or a session is ended with the command SIGNON without clearing the active file. Recovery means that the active file is automatically saved and cataloged as the data set name (dsname) Bnnnnn.ACTIVE (Bnnnnn is the userid currently signed on). The next time he signs on the user will be informed of the existence of an ACTIVE dataset with the information prompt:

ACTIVE SAVED BY RECOVERY LAST SESSION.

To recover this file the user should give the command USE ACTIVE.

Note: There are, of course, types of system failure where recovery is impossible. A user should, therefore, scratch and save current revisions of important or complex datasets often.

The recovery file is saved on short term volumes; that is, it will be automatically deleted seven (7) days after creation. Therefore it is important that the Bnnnnn.ACTIVE file be USEd and SAVEd under another name and then SCRATCHed as soon as possible. For example, if WYLBUR dies while one is on line, his ACTIVE file will be saved as Bnnnnn.ACTIVE. If, some six days later the system again dies while Bnnnnn is on line, and user Bnnnnn has not scratched the six-day-old ACTIVE dataset, the dataset in active use at the time of the crash will be saved over the previously stored one, but will be regarded as already six days old. Thus, not only will the user lose the dataset from six days ago, but tomorrow the dataset lost and stored as Bnnnnn.ACTIVE will be scratched.

Users should not save any data sets by the name ACTIVE since WYLBUR will use this name and replace the old data set named ACTIVE whenever it performs recovery.

2.7.5 Managing Data Sets

This section presents a few guidelines which will make it easier for the user to manage his data sets and his available WYLBUR disk pack space. Because space allocated to each user is limited, managing it well will mean he can save more data sets and do his terminal work more efficiently.

2.7.5.1 Using Library and Sequential Space

The user may save his active file as a sequential data set or as a library member. The file size is the most important criterion for choosing between the two for the effecient use of space. The minimum space allocated to a sequential data set is one track. Therefore, data sets requiring less than one track should be saved as library members; those requiring more as sequential data sets.

A maximum of 4000 to 5000 card images is workable for a sequential data set. Handling larger files, while possible for WYLBUR, is time consuming and should be avoided if possible.

The SHOW SPACE command (see the SHOW SPACE command in section 4.3) is useful for determining how many tracks of disk space the user has used and the SHOW DSNAMES command (see the SHOW DSNAMES command in section 4.3) with the SPACE option shows how large each data set is.

NOTE: A PDS cannot be created on WYLBUR. Use TSO or a batch job to create a PDS (saving a null line in it is sufficient). WYLBUR can then be used to SAVE, edit or delete individual members of the PDS. (See 2.7.5.2, below)

2.7.5.2 Partitioned Data Sets

A partitioned data set (PDS), as the name implies, is a data set that is divided into segments called members. The nature and size of the PDS must be described to the system, and a block of space has to be allocated for a directory. The system will then use the directory to find individual members of the PDS.

Since WYLBUR cannot be used to allocate a PDS, the user will have to use TSO or submit a batch job to allocate it. Once the PDS has been allocated, WYLBUR can be used to SAVE, REPLACE, CONDENSE, or EXECUTE individual members. Following are examples of how to allocate a PDS using TSO and by submitting a batch job.

TSO: Logon to TSO, and when TSO prompts with READY, type in:

```
ALLOCATE DA(dsname) NEW SP(10) BL(3520) DIR(2) UNIT(LONGSHRD)
```

The above command allocates a PDS called (dsname), it is new, and it will be given ten blocks of space. In this example the size of the block is 3520 and each block will hold 44 card images. Therefore, this example allocates enough space for about 1320 card images.

NOTE: One of the nicer features about WYLBUR is that, in EDIT format, the default of the SAVE command, WYLBUR removes all non-significant blank spaces from the data before saving. This makes it possible to save much more data in a small space. When you use the USE, SAVE and related commands, WYLBUR reinserts the blanks automatically. Thus, there is room for much more data in a dataset if you do not save in card images. The default blocksize in WYLBUR is 3520 on 3214 disks, and 3156 on 7330 disks.

The above example allocates two blocks for the directory. A directory block is 256 bytes long, and will control approximately twelve members. In this example the PDS will be stored on a long term storage disk.

Another way to allocate a PDS is to submit a batch job, using a catalogued procedure called MAKEPDSK, (described in 'Catalogued Procedures for Dataset Transfer, TM-293'). To use this procedure, type the following into an active WYLBUR file, starting in column one:

```
//JOBNAME JOB (Fnnnnn,02,,02)
          CUA CARD
//STEPA  EXEC  MAKEPDSK,
//      OUTDSN='Bnnnnn.dsname',OUTUNIT='LONG7330'
```

This job can then be submitted by the WYLBUR SUBMIT command. This procedure allocates ten tracks and five directory blocks to the PDS, which is saved on long term storage disks. These parameters can be overridden (see TM-293).

For example, MAKEPDSK allocates a dataset with a Fixed Block format, a Logical Record length of 80, and a Blocksize of 3120. This serves well only if you want to SAVE your data in CARD format. If you wish to take advantage of WYLBUR'S space saving EDIT format you must override these parameters. To SAVE in EDIT format on 7330 disks, as in the above example, insert the following card:

```
// OUTRECF='U',OUTLREC='3156',OUTBLKS='3156'
```

**Don't forget to put a comma at the end of card 4, to indicate a continuation.

If you wish to save your data on 'LONG 2314' your OUTRECL and OUTBLKS will be '3520'.

Members can then be saved in the PDS using WYLBUR commands. For example, to save the contents of your ACTIVE file into a member called STUFF in a PDS called LIB, issue the command:

```
          SAVE LIB(STUFF)

    or    SAVE LIB#STUFF
```

Member names are distinguished from dataset names by being either enclosed in parentheses, or by being preceded by '#'.

The SHOW DIRECTORY command (see section 4.3) lists the member names in a library (partitioned dataset). Note that space from deleted members does not become available for reuse until the library is CONDENSEd.

When a library member is REPLACEd, it is saved at the end of the allocated library space and is scratched from its original location. As a series of SAVE and SCRATCH commands is given, empty space develops wherever a library member has been scratched. Eventually the available library space will be used up even though there may be enough scattered space to hold additional members. When this happens, WYLBUR prompts the user with a CONDENSE? message. If the reply YES, OK, or CONDENSE is given, WYLBUR submits a batch job to condense the library--that is, to reorganize the members and leave a contiguous empty space at the end. The CONDENSE prompt may be avoided by choosing a more convenient time to condense the library by using the CONDENSE command (see section 4.3).

2.7.5.3 Scratching Obsolete Data Sets

The user should occasionally look at the names of his data sets with the command SHOW DSNAMES (see section 4.3), and scratch any data sets that are obsolete. Occasionally dsnames like ACTIVE and REPLACE may appear. ACTIVE is assigned by WYLBUR in the event of a recovery (see section 2.7.4). A data set may be named REPLACE because the user did not specify a dsname in the command SAVE <dsname> REPLACE. To SCRATCH an obsolete dataset use the command

```
SCRATCH <dsname>
```

2.8 Global Parameters

Global parameters contain values used sufficiently frequently in WYLBUR commands that it would be tedious to ask the user to supply them each time. These values are either set automatically by WYLBUR at sign-on time or by the user with a SET command. For example, at sign-on time the default line length accepted by WYLBUR without a warning message is 72 column characters. The user may change this to, say 133 characters, with the SET LENGTH command. The length thereafter used by WYLBUR as the default in any command that has a LENGTH parameter would be 133. For instance, if the user gave the command JUSTIFY 10/15, WYLBUR would justify lines 10 through 15 to a line length of 133.

The following table is intended to give the reader a summary of some of the global parameters. Each command is described in detail in section 4.3. The more frequently used global parameters DELTA, LENGTH, UPLOW/UPPER and TERSE/VERBOSE are described in some detail following the table.

SET/SHOW Command	Function of SET Command	Sign-on Value
SET (SHOW) BACK	Allows use of backspace key to enter backspaces into the active file. NOBACK returns its function to erasing characters only.	NOBACK
SET (SHOW) CURRENT	Sets current line pointer (see "line number" in section 4.3) to the specified value.	none
SET (SHOW) DELTA	Sets default value of the line increment, DELTA.	1.000
SET (SHOW) LENGTH	Sets line length to the specified number of characters. The line is accepted, but a warning message is issued when this number of characters is exceeded in a line.	72
SET MODE {RETRY|NCRETRY} SHOW MODE	Causes WYLBUR to prompt the user to make modifications to the last command issued whenever it failed to execute.	NORETRY
SET (SHOW) MEMBER	Sets default member name to be used in forming a dsname (see "dsname" in section 4.3).	null string
SET (SHOW) NAME CLEAR NAME	Sets default values for USERID, PREFIX, and/or MEMBER in one command (see "dsname" in section 4.3). To clear the name, use CLEAR NAME.	Sign-on value for USERID
SET (SHOW) PREFIX	Sets prefix that can be used in building a dsname (see "dsname" section 4.3).	null string
SET (SHOW) SLOWLIST SET (SHOW) FASTLIST	SLOWLIST prevents WYLBUR from using tabs when listing online. FASTLIST allows use of tabs.	FASTLIST

SET (SHOW) TABS CLEAR TABS	Allows setting of up to 16 tabs. CLEAR TABS erases current logical tab settings.	tabs are remembered for SIGNON and are forgotten for LOGOFF
SET TERSE, SET VERBOSE	TERSE shortens command prompt COMMAND? to ? VERBOSE returms the system to the default mode.	VERBOSE
SET UPLOW, SET UPPER, SHOW CASE	UPLOW allows WYLBUR to recognize upper and lower case alphabetic characters. UPPER converts upper and lower case to upper case only.	UPPER
SET (SHOW) USER	Sets default user code to be used in forming a dsname (see "dsname" in section 4.3).	Response given to SIGNON prompt USERID?
SET (SHOW) VOLUME	Sets default disk volume to be used in those commands where volume is required but not specified.	CATALOG

2.8.1 Case (SET UPLOW/UPPER)

Case is determined by the shift key just as it is on an ordinary typewriter. However, in order for WYLBUR to recognize upper and lower case the user must use the command SET UPLOW. Otherwise, such text will be stored in all upper case.

If UPLOW is in effect, the user may revert to having all alphabetic characters in upper case by giving the SET UPPER command. The command SHOW CASE can be used to find out whether UPPER or UPLOW is currently in effect. Also see section 2.3.4.

2.8.2 DELTA

DELTA is the increment which WYLBUR normally adds to the present line number to get the next one. It is also used as the default starting line number when collecting a file. The value for DELTA may be set at any number between .001 and 99999.999 inclusive. If the user does not set DELTA, 1.000 is used as the default. It may be overridden explicitly and implicitly in all commands that have an increment as a parameter.

2.8.3 LENGTH

LENGTH is a global parameter which may be set to any value between 1 and 133 characters. The default is 72 characters.

WYLBUR will accept lines which are longer than the set length but a message will be typed out after each text line (not command lines) that exceeds LENGTH characters. The message is in the form:

n CHARACTERS IN LINE x

In COLLECT mode WYLBUR will not accept lines of over 133 characters, but will reprompt with the line number. In MODIFY or CHANGE mode WYLBUR keeps the line, but truncates it to 133 characters and prints the warning message

TRUNCATED TO 133 IN LINE x

2.8.4 TERSE/VERBOSE

The command SET TERSE shortens WYLBUR's command prompt "COMMAND?" to a "?". The TERSE mode also eliminates the response to certain commands and causes a short form of data set names to be typed to the user instead of the full name. (See "dsnames" in section 4.3).

SET VERBOSE returns this parameter to its default.

3.0 Summary of Commands

This section is designed to give an overview of the specific WYLBUR capabilities by describing the function of each command. The commands are divided into three major functional areas:

- Editing Commands. These permit modification and manipulation of the contents of the active file.
- Control Commands. These commands interface with the operating system and ASP in performing system functions. They enable the user to perform such operations as submitting a job to the batch computer system.
- Commands relating to execute files, the WYLBUR preprocessor, and the use of conditional IF command.

All of these commands are fully described in section 4.3.

3.1 Editing Commands

ALIGN	left-justifies the contents in the specified range of lines of the active file so that each line contains as many words as possible but no more than the number of characters specified by the length parameter or the global length. See also JUSTIFY.
CENTER	centers each line of text in the specified range within the length specified or the global length setting.
CHANGE	replaces a character string with another character string in a specified range of lines in the active file. Often the simplest way to change one or more lines is with this command. See also EDIT and MODIFY.
CCLLECT	creates a line or adds one or more lines of text into the active file.
CCMPARE	compares two sets of consecutive lines, both of which must be in the active file, line by line.
COPY	copies a range of lines from an active, external, or execute file to another place in the active file.
DELETE	removes a range of lines from the active file.
EDIT	changes characters in the text of a range of lines in the active file. See also CHANGE and MODIFY.
INSERT	inserts specified lines of text into the active file.
JUSTIFY	is identical to ALIGN except that it right- and left-justifies text in specified range by inserting blanks between words.
LIST	causes part or all of the active file to be listed online or printed offline.
MODIFY	allows insertions, replacements, and deletions of characters in the text of the specified range of lines of the active file. See also CHANGE and EDIT.
MOVE	copies a specified range of lines from one part of the active file to another and deletes the range from its original location.
POINT	functions exactly the same as LIST except that POINT updates the pointer to the current line.

PUNCH produces punched card output of all or part of the active file.

RENUMBER renumbers part or all of the active file.

REPLACE replaces the contents of the specified range of lines in the active file.

3.2 Control Commands

3.2.1 ASP-related Commands

These commands interface with ASP:

BACKLOG	PUNCH
CANCEL	RELEASE
CONDENSE	ROUTE
HOLD	RUN
LIST OFF	STATUS
POINT	SUBMIT
PRTY	SUGGEST

BACKLOG allows the user to inquire about the backlog of jobs in the batch system. Alternatively, if a remote station is specified a report is given on the batch jobs whose output will be processed at that station.

CANCEL causes the specified job to be sent to the print queue. If NOPRINT is specified, output processing is cancelled.

CONDENSE submits a job that will condense the specified partitioned data set using the COMPRESS capability of the IBM utility IEBCOPY.

HOLD prevents the job from being processed beyond the current processing phase (e.g., execution, print).

LIST OFF submits a job to list active or execute files on impact or microfiche printers.

POINT OFF is almost identical to LIST OFF.

PRTY allows the user to raise the scheduling priority of his batch jobs.

PUNCH produces punched card output of the active or execute file.

RELEASE releases a previously held job so that it will begin or continue processing. See HOLD.

ROUTE	allows the user to change the location at which hi job's output will be processed.
STATUS	allows the user to inquire about the status of any job in the system, whether the job was created by WYLBUR or not.
SUBMIT or RUN	submits all or part of the active file into the job input stream of the 370/195.
SUGGEST	may be used to send the users' file of suggestions contained in the active or execute file to another user or to the User Services Group.

3.2.2 Non-ASP-related Commands

CATALOG	causes the specified data set, which has not been cataloged previously, to be entered in the system catalog. See also the RECATALOG command.
CLEAR	is used to erase the current active file, execute file, tab settings, or dsname fields.
LOGOFF	terminates the terminal session.
RECATALOG	catalogs or recatalogs a data set.
SAVE	copies all or part of the active file onto external file space.
SCRATCH	deletes the specified external file.
SET BACK	allows the user to enter an upper-case backspace into the text of the active file. Lower-case backspaces still erase characters. (2741-type terminals only: ASCII terminals must use CTRL-X to perform character erasure when BACK is set.)
SET NOBACK	returns upper-case backspaces to their normal function of erasing characters.
SHOW BACK	shows which option, BACK or NOBACK, is in effect.
SET CURRENT	sets the value of the symbolic line number CURRENT as specified.
SHOW CURRENT	shows the value of line number CURRENT. (See also SHOW line number.)
SET DELTA	sets the default value of the line number increment, DELTA.

SHOW DELTA	displays the current value of DELTA.
SET KEYWORD	allows the user to set or change his password (see section 2.4.1).
SET LENGTH	sets the line length to the specified number of characters. A line longer than LENGTH characters is accepted, but WYLBUR issues a warning message. (See Section 2.8.3).
SHOW LENGTH	displays the current value of LENGTH.
SET MEMBER	sets the default member name that will be used to form a dsname.
SHOW MEMBER	The current value of MEMBER can be displayed by SHOW MEMBER or SHOW NAME.
SET MODE	sets the RETRY mode values for the session. RETRY mode causes WYLBUR to prompt the user to modify the last command submitted whenever that command fails to execute due to syntax errors.
SHOW MODE	displays the current retry mode values.
SET NAME	allows user to set the values of the USERID, PREFIX, and/or MEMBER fields in one command. These values are used in forming a fully qualififed WYLBUR dsname.
SHOW NAME	displays the current USERID, PREFIX, and MEMBER fields.
SET NOTIME	prevents WYLBUR from signing off a user who has been inactive for 9 minutes. The request will be ignored as long as fewer than MINFREE lines are available; see SHOW MINFREE.
SET TIMEOUT	cancels NOTIME.
SHOW CATALOG	displays a list of dsnames and/or index pointers below a given index node in the system catalog.
SHOW COLUMNS	produces a line displaying column numbers.
SHOW COUNT	displays the number of users currently signed on the system.
SHOW DATE	displays the time of day and the date in these forms: hh:mm:ss MM/DD/YY and Julian date.

SHOW DIRECTORY	produces a list of member names in the specifie partitioned data set.
SHOW DSNAMES	produces a list of dsnames on the specified volume.
SHOW FREE	gives a count of the available phone lines and a list of their port numbers; see also SHOW MINFREE.
SHOW IDLE	is similar to SHOW FREE. It shows the count of all WYLBUR lines not in use now, and their port numbers.
SHOW JOBNO	displays the numeric portion of the default WYLBUR jobname to be assigned to the next job submitted from a WYLBUR terminal.
SHOW LINE	displays the port number and terminal number and other information for the specified line.
SHOW LINES	displays the status of all active remote terminals including the users who are logged on to them. SHOW USERS is a quicker command to use to find out all users who are logged on.
SHOW symbolic line-no	displays value of the symbolic line number in an active or an execute file.
SHOW MINFREE	gives a count of the available phone lines and lists the current value of MINFREE which is the minimum number of free lines required before WYLBUR honors the SET NOTIME requests on dial lines.
SHOW MESSAGE	causes the WYLBUR Message to be displayed at the terminal.
SHOW PAGES	prints the number of pages in the active file. The size must be less than or equal to three if the active file is going to be used as an execute file.
SHOW SIZE	may be used to determine which lines in the specified range are over a specified number of characters in length. It can be useful in finding lines to ALIGN.
SHOW SPACE	displays (1) the number of free tracks and extents available for all users on the specified volume, or (2) the total number of tracks currently allocated by the user on the shared packs and the total number he is allowed to use.
SHOW TIME	prints the date and time in hours, minutes and seconds and the current values for the session elapsed time and editing time.

SHOW TRACKS	see SHOW SPACE. TRACKS is equivalent to space.
SHOW USER	displays the user code currently used as default in forming a dsname.
SHOW USERS	types the USERID of all signed on users. See also SHOW LINES.
SHOW VOLUMES	lists all disk packs the user may read from and write on.
TO	may be used to send a message to another terminal.
UNCATALOG	uncatalogs the specified data set.
USE	reads a copy of an external file or any other direct access data set into the user's active file.

3.3 Commands Relating to Execute Files, WYLBUR Preprocessor, and Use of Conditional Statements

CLEAR EXEC	The EXEC option erases the execute file.
COMMENT	is designed to be used in an execute file to type a line to the user.
COPY EXEC	The EXEC option allows the user to copy all or part of an execute file to the active file.
EXECUTE	allows the user to load and to execute a file of WYLBUR commands, to branch within an execute file, or to restart execution within the execute file. Commands are usually executed sequentially unless a branch instruction is specified.
IF	evaluates a relational expression and if the result is true, executes the WYLBUR command following the relational expression.
LIST EXEC	The EXEC option allows the user to list all or part of the execute file on or offline.
POINT EXEC	functions like LIST EXEC.
PUNCH EXEC	The EXEC option allows the user to punch all or part of the execute file.
READ	is designed to be used in execute files (although it can be used at the terminal). It allows the user to enter a command or values to be assigned to a list of variables, which may be string variables, line number variables, or integer variables.

RUN EXEC	submits all or part of the execute file to th central computer as a job.
SAVE EXEC	saves all or part of the execute file.
SET CURRENT EXEC	sets the pointer to the symbolic line number CURRENT in the execute file to the specified value.
SHOW CURRENT EXEC	displays the value of CURRENT in the execute file.
SET ESCAPE	allows the user to specify an escape character. The escape character signals the preprocessor that text substitution should be made in a command line before WYLBUR executes it. Initially the escape character is null.
SHOW ESCAPE	displays the current escape character in use.
SET EXEC	allows the user to specify that certain messages not be printed while executing an execute file (TERSE mode) and that each command not be listed before it is executed (NOLOG mode). The parameter AUTO causes WYLBUR to begin the terminal session by executing from the execute file LOGON out of the user's library LIB after the logon procedure is complete.
SHOW EXEC	displays whether LOG or NOLOG, TERSE or VERBOSE, and AUTO or NOAUTO modes have been set.
SET RETURN	sets the symbolic line number RETURN to the specified value. RETURN is designed to facilitate execute file subroutines. It provides a means of returning after branching to another portion of the execute file.
SHOW RETURN	displays the value of the RETURN line number.
SET RESCAN	The preprocessor can be directed to rescan any text that it substitutes. This could lead to an infinite rescan if an error is made. So that infinite rescans do not occur a default limit of 5 rescans is set. This limit can be changed with this command.
SHOW RESCAN	displays the value of the rescan count.

SET SKIP	allows the user to specify a SKIP character. The character following the SKIP character will not be changed by the preprocessor. Thus, the user may enter a command line that will have an ESCAPE or SKIP character in it after the preprocessor has finished text substitution.
SHOW SKIP	displays the current SKIP character in use.
SET VALUE	allows the value of an expression to be assigned to a string, integer, or line number variable.
SHOW VALUE	displays the values of a list of expressions. It can be used as a desk calculator.
SHOW line-no EXEC	allows the user to display the specified value of the symbolic number in the execute file.
SUGGEST EXEC	The EXEC option allows the user to submit the execute file as a suggestion to another user.

THIS PAGE

WAS INTENTIONALLY

LEFT BLANK

4.0 Detailed Command Description

This chapter is intended as a detailed reference guide to the format of WYLBUR commands for persons already familiar with WYLBUR. For general information on how to use the commands consult chapters two and three of this manual or the WYLBUR Tutorial.

4.1 Notation Used to Define WYLBUR Commands

The notation used to define WYLBUR commands is described in the following paragraphs.

(1) Whenever the characters listed below occur in a definition, they must be entered into the command exactly as shown.

a.	apostrophe, quotes	' "	f.	period	.
b.	asterisk	*	g.	minus	-
c.	comma	,	h.	plus	+
d.	equal sign	=	i.	slash	/
e.	parentheses	()	j.	pound	#

(2) WYLBUR command keywords written in uppercase letters must be entered exactly as they appear in the definitions.

(3) Lowercase letters, words, and numbers appearing in a WYLBUR command description represent symbols for which specific values must be substituted in the actual command.

Example: If "string1" appears in a command definition, a specific value (e.g., "STUFF") is substituted in the actual command.

(4) The characters listed below are assigned special meanings in the following definitions [see (5) through (7)].

a.	or	|
b.	underscore	_
c.	braces	{}
d.	brackets	[]
e.	angle brackets	<>

(5) Items separated from each other by the "or" symbol (i.e., the character |) represent alternatives. Only one such alternative may be selected. Example:

```
ALIGN|JUSTIFY
```

indicates that a choice between ALIGN and JUSTIFY can be made.

(6) An underscore indicates a default option. If an option is nc stated in the entered command, the underscored value will be entered automatically. Example:

UPLOW|UPPER

indicates that either UPLOW or UPPER should be selected; but if UPPER is selected, it need not be written in the actual command. If nothing is entered in this option UPPER is assumed by WYLBUR.

(7) Braces group alternatives one of which must be chosen. Example:

SET VOLUME {CATALOG|<volume>}

indicates that either <volume> or CATALOG must be chosen. If you enter neither, WYIBUR will not execute the command, but will prompt you for the information.

(8) Brackets group optional alternatives; everything within the brackets is optional and may be omitted. Example:

[LIST|NOLIST]

indicates that a choice can be made between LIST and NOLIST or that the values may be omitted entirely.

(9) Angle brackets surround symbols for which the user must substitute a specific value. The explanation for such symbols is usually included with the description of the relevant WYLBUR commands, but especially important ones are explained under their cwn alphabetic headings. Three of the most important of these are <dsname>, <range>, and <line number>. Since it is important that the user understand these concepts as used in WYLBUR, you should read the entries for these concepts before reading further in the Reference Guide.

(10) The significance of spaces is discussed in section 4.2.

4.2 WYLBUR Syntax

The general syntax of WYLBUR commands is:

Command-Word Positional-Parameters Keyword-Parameters; comments

Parameters must be separated from each other and the command word by at least one separator. Blank, equal sign (=), and comma (,) are all equivalent separators. Also, in keyword parameters, the keyword must be separated from the keyword value by at least one separator.

If the character apostrophe (') occurs within a string that is specified in a positional or keyword parameter, the entire strin must be enclosed in quotes (") or two apostrophes ('). If th string contains quotes("), it must be enclosed in apostrophes or two quotes. Otherwise, a string may be enclosed in either apostrophes or quotes. For example:

```
                CHANGE "John's" TO "Pete's" IN x
or              CHANGE ''John's'' TO ''Pete's'' IN x

or              CHANGE 'He Said, "Hi."' TO 'She said "No!"' IN x
or              CHANGE "He said, 'Hi.'" TO "She said 'No!'" IN x
```

Unless noted otherwise, keyword parameters may be specified in any order, but positional parameters must be specified first when they are not omitted. The description of the parameters in section 4.3 will indicate whether a parameter is positional.

WYLBUR does not begin processing a typed command until the user terminates the command with a carriage return.

Almost all command words and keywords may be abbreviated to the beginning three letters (or more). Thus, REN, RENU, RENUM, RENUMB, and RENUMBER are all recognized as the keyword RENUMBER. All characters typed -- up to the first eight -- are checked. So, RENUBR would not be recognized as the keyword RENUMBER. However, some words, listed in Appendix A, have abbreviated or alternate forms that do not follow this general rule.

4.3 Detailed Description of Commands and Important WYLBUR Concepts

This section describes the WYLBUR commands and their parameters. Commands are listed in alphabetical order.

Each command description begins with the command syntax. The notation used in the definition is described in section 4.1. The syntax definition is followed by a description of the general purpose of the command, then each parameter is described in detail. Examples are provided for a few of the more complex or most generally useful commands.

Definitions of parameters enclosed by angle brackets <> are interspersed alphabetically among the commands (for the meaning of these symbols, see section 4.1).

{ALIGN|JUSTIFY} <range> [LENGTH xxx] [MARKER c] [NUMBER]
[EVEN] [SPACE] [INDENT x [-] [y]]

---

where:

ALIGN — left-justifies the contents of the lines in the specified range of the active file so that each line contains as many words as possible but no more than LENGTH characters.

JUSTIFY — left- and right-justifies the contents of the lines in the specified range. Lines will be padded with blanks if necessary to obtain a line length of exactly LENGTH characters.

<range> — specifies the upper and lower limit of a group of lines to be aligned or justified. This parameter is required and must be specified first. Alignment restarts whenever nonconsecutive lines in a range are encountered or when a new range is started, as in a disjoint range. See "range" in this section.

LENGTH xxx — xxx is a value of 1 to 133 characters. Contents of lines will be aligned or justified to the specified character length. Default is current value of global LENGTH (see SET LENGTH). The default at LOGON time is 80 characters.

MARKER c — c is any character. Lines containing the marker character in column 1 will remain unchanged and the alignment or justification process will be restarted with a new paragraph. it is a good idea to use marker symbols that do not appear in the text, if possible. Characters () " , ' or = must be in quotation marks when they are used as markers. Null lines are always treated as marker lines.

NUMBER — causes the entire file to be renumbered after the alignment or justification process is complete. The starting line number and the line increment will both be DELTA (see SET DELTA). The default is renumbering only lines in the aligned or justified range.

WYLBUR will automatically renumber the specified range with a delta value that will not cause the remainder of the active file to be renumbered.

EVEN must be used to cause any change in the existing indentation of text. If EVEN is specified, WYLBUR ignores existing indentations, aligns or justifies all non-marker lines, and starts a new paragraph only after a blank line, a line with the marker character, or the first line of a specified range.

SPACE removes extra spaces which may have been inserted by a previous JUSTIFY command. It is useful for previously justified text. Unless SPACE is used, extra blanks from the previous justification or alignment will be kept as part of the text when it is realigned or rejustified. Two blanks are always left after the characters ? : . !

INDENT indents aligned text according to parameters x and y, or aligns indented material which is ordinarily ignored if this parameter is not specified. If EVEN is not specified, only text whose indentation matches the specified INDENT values exactly will be aligned or justified. Text beginning in column 1 has the indentation value of 0. If the INDENT parameter is not specified, a default value of INDENT 0 0 is used. See Note 1 for examples of this parameter.

x is an integer and specifies that lines will be indented x spaces.

y y is a positive or negative integer and specifies that the first line of paragraphs will be printed with ADDITIONAL indentation of y spaces. The default value of y is 0. (See examples in note 1.)

When text must be removed from one line to another during aligning or justifying, WYLBUR will split the text at a blank; a word is never split onto two lines. WYLBUR puts the rearranged text together by inserting one blank between the two parts unless the first part is one of the characters ? : . ! Two blanks are inserted after these characters. WYLBUR renumbers that portion of the file that was aligned or justified. (See NUMBER, below.)

Note 1. The follcwing examples illustrate the use of INDENT x [-] [y] EVEN. Assume the range to be aligned consists of three paragraphs. Each paragraph contains contiguous lines of text and the paragraphs are separated by a null line. Let us say that the following is our text, and that we have been careless about indentation as we typed it in.

```
1.      This is a sample text that will be used to
2.         illustrate the use of the ALIGN|JUSTIFY
3.            commands.
4.
5.                  Notice the effect of the EVEN
6.               command and the use of the EVEN and the
7.            INDENT command.
8.
9.      There is a wide range of possible uses for these
10.     commands when they are used together with the
11.     INDENT parameters.
```

If one uses the JUSTIFY command,

```
JUSTIFY ALL LENGTH 30
```

there is no effect on indented lines:

```
1.       This is a sample text that
1.5      will be used to
2.          illustrate the use of the ALIGN|JUSTIFY
2.5            commands.
3.
3.5                 Notice the effect of the EVEN
4.          command and the use of the EVEN and the
4.5           INDENT command.
5.
5.5      There is a wide range of
6.       possible uses for these
6.5      commands when they are used
7.       together with the INDENT
7.5      parameters.
```

Notice that WYLBUR renumbers the lines when necessary, using a delta value that will not require renumbering of the entire file. If you are using a delta value of .001 there are no available line numbers between lines. An attempt to justify lines so numbered could cause a loss of lines. WYLBUR will drop blank lines first, but it is a good idea to use a delta value that leaves room for adding lines. See "default delta" and SET DELTA below.

The use of the EVEN command causes WYLBUR to treat every contiguous line the same regardless of indentation in the original. EVEN used with INDENT will produce an aligned or justified paragraph with indentation specified by integers x and y. For example, the command:

```
JUSTIFY ALL EVEN SPACE LENGTH 30 INDENT 0 5
```

causes all lines within paragraphs to be aligned, the first line after each break is indented 5 spaces, and all others are indented 0 spaces. The SPACE parameter must now be used to remove insignificant blank spaces created by the last JUSTIFY command.

```
1.          This is a sample text
1.5    that will be used to
2.     illustrate the use of the
2.5    ALIGN|JUSTIFY commands.
3.
3.5         Notice the effect of the
4.     EVEN command and the use of
4.5    the EVEN and the INDENT
5.     command.
5.5
6.          There is a wide range of
6.5    possible uses for these
7.     commands when they are used
7.5    together with the INDENT
8.     parameters.
```

Reverse indentation can be created by using a negative value for y.

```
JUSTIFY ALL EVEN SPACE LENGTH 30 INDENT 5 -5

1.      This is a sample text that
1.5          will be used to
2.           illustrate the use of the
2.5          ALIGN|JUSTIFY commands.
3.
3.5     Notice the effect of the EVEN
4.           command and the use of
4.5          the EVEN and the INDENT
5.           command.
5.5
6.      There is a wide range of
6.5          possible uses for these
7.           commands when they are
7.5          used together with the
8.           INDENT parameters.
```

BACKLOG [station name]

The BACKLOG command returns a count of the number of jobs in the ASP batch queues. For each priority level and for EXPRESS class the number and time of jobs waiting (or active on) MAIN or SETUP is listed, along with the number of peripheral jobs. When a station name is given, the backlog of lines, cards and time is given.

where:

station name — is the name of a RADS or RJP remote entry station. A station name of LOCAL may be used for the AMD computer area.

CANCEL <jobid> [PRINT|NOPRINT]

CANCEL terminates processing of the job immediately. If the job is currently printing, output processing will be continued unless NOPRINT is specified.

where:

<jobid> identifies the job to be canceled by the jobname or the ASP jobnumber. This parameter is required and must be specified first. Jobs not belonging to the signed-on user may not be cancelled. See "jobid" in this section.

PRINT causes output to be printed. If the job has not yet started execution, SYSMSG will still be printed.

NOPRINT causes no further output to be printed. If the job has not yet started execution, not even SYSMSG will be printed.

CATALOG <dsname> ON <volume> [<dsn options>]

CATALOG causes the specified data set to be cataloged in the system catalog. If the data set is already cataloged, the message "dsname NOT CATL'G - ALREADY CATALOGED" will be given. Use the RECATALOG command or RECATALOG parameter of the SAVE command to recatalog a data set. This command is seldom used at Argonne, since most data sets are automatically catalogued at creation.

where:

<dsname> is the name of the data set to be cataloged. This parameter is required. See "dsname" in this section. If the data set to be cataloged does not belong to the signed-on user, WYLBUR will respond "data set name NOT YOURS" and then prompt "KEYWORD?" (same as password). If the user replies with the correct keyword (password) of the user whose data set is to be cataloged, the command will be executed; otherwise, the command will be aborted.

CN <volume> specifies the direct access volume on which the data set resides. This parameter is required. See "volume" in this section.

<dsn options> The dsn options USER, SET|NOSET may be specified. See "dsn options" in this section.

CENTER <range> [INDENT x] [LENGTH xxx]

CENTER centers each line of text in the range within the length specified or the global length.

where:

<range> is the range of lines to be centered. This parameter is required and must be specified first. See "range" in this section.

INDENT x x may have the value 1-132. It allows centering based on a column indentation value.

LENGTH xxx xxx may be the value 1-133. It specifies the line length within which each line is to be centered. The default value is the global length; see SET LENGTH.

CHANGE {['stringa' [p/q/f] [(n)]] | [x/y/f]}
{TO {'stringb'|number} [{+|-}n] [LINE]
| USING s[/t] [COLUMNS v[/w]]
[REPEAT] [LINE]} [IN <range>] [<list options>]

The CHANGE command may be used to replace a character string with another string or an integer. It is often the simplest way to change a line or group of lines, especially if one character string is to be replaced by another of a different length. See also the MODIFY and EDIT commands.

where:

stringa is the character string to be changed.

p,q,f p and q are column numbers of a line. They restrict the search for stringa to columns p through q.

f specifies a fill (or fence) column to prevent any shifting of text in or to the right of that column. Any blanks are shifted in or out before the fill column. If any text is lost, a message is given. This parameter is quite useful when column dependent text is being edited as the CHANGE command normally shifts all of the line to the right of the change whenever characters are inserted or deleted.

These parameters may be specified in various combinations as follows:

p/q/f Columns p through q are searched and the fill column is f.

p//f stringa must begin in column p and the fill column is f.

//f p defaults to column 1 and q defaults to f-1.

p stringa must begin in column p. Text will be shifted if characters are added or deleted.

p/q Columns p through q will be searched for the occurrence of stringa. Text may be shifted.

(n) is an occurrence count. It restricts the replacement of stringa to the nth occurrence in each line of the range. If it is omitted, every occurrence of stringa is changed.

x,y,f x and y specify the beginning and ending columns,
respectively, that are to be changed. f specifies the fill column (see parameters p,q,f above). The parameters may be specified in various combinations as follows:

x/y/f Columns x through y will be changed; f is the fill column.

x//f The replacement string will be inserted before column x; f is the fill column.

x/y Columns x through y will be changed; text may be shifted if characters are deleted or inserted.

x The replacement string will be inserted before column x; text may be shifted. E.g., CHANGE 1 TO ' ' would insert a blank before column 1, whereas CHANGE 1/1 TO ' ' would change column 1 to a blank.

TO specifies a string which is to replace stringa. The replacement string may consist of any number of alphabetic, numeric, or alphanumeric characters. It need not be of the same length as stringa.

stringb — specifies the replacement string. Three types of strings should be differentiated:

(1) alphabetic string. For every occurrence of stringa or the specified columns, this type of string is directly substituted.

(2) alphanumeric string. If an increment, n, is specified, it is used as an edit pattern. That is, the numeric portion of this string is incremented or decremented after each replacement. If the increment is not specified, this type of string is treated as is an alphabetic string.

(3) numeric string. The string may optionally contain leading zeroes and blanks. If it is to be used as an edit pattern, the increment, n, must be specified. Otherwise, the number remains constant with each use.

number — may be any positive or negative integer that will be used as the replacement string. There will be no leading blanks and no leading zeroes unless explicitly typed. The number is incremented automatically by 1 if the increment is not specified.

±n — n is an integer used as an increment. See the description of parameters 'stringb' and number.

LINE — Normally a new incremented replacement string is calculated for each occurrence of stringa. This parameter specifies that all occurrences of stringa in a single line are to be replaced by the same incremented replacement string.

USING s/t — This parameter is a way of designating that the replacement string is to be taken from another line(s) in the active file. s and t are the beginning and ending line numbers, respectively, of the replacement range. s may be ALL to denote the entire active file.

COLUMNS v/w — v and w refer to the beginning and ending columns, respectively, of the replacement string in each line of the USING range. If the string contained in columns v/w does not contain w-v+1 characters, the string will be blank filled on the right to contain that many characters. If only v is specified, characters v through 133 (which may be null) are used.

REPEAT allows the user to repeat a USING replacement range as many times as necessary.

LINE Normally a new line is obtained for each occurrence of stringa. This parameter specifies that all occurrences of stringa in a single line are to be replaced by the same USING range line.

IN <range> specifies the range of lines to be searched for the occurrence of stringa. See "range" in this section. The entire file (the explicit range ALL) is default for this parameter.

<list options> The list option LIST is the default for this command. See "list options" in this section. NOLIST prevents the result of the CHANGE command from being printed at the terminal. It is a good idea not to use the NOLIST option unless you are absolutely certain of the effect of the CHANGE command on your data.

While the complete syntax of CHANGE commands is accurately represented above, most users will seldom have occasion to use it all. More often, the CHANGE commands are used in one of several basic forms. Some example CHANGE commands are given below.

1. CHANGE 'stringa' TO 'stringb' IN <range> is the simplest form; it replaces stringa with stringb wherever it finds it in the given range.

2. CHANGE 'stringa' <column range> TO 'stringb' IN <range> works just the same as example one, except that it will only replace stringa when it occurs within the specified columns.

3. Change 'stringa' <column range> TO <integer> [{+|-}n] IN <range> changes the first occurence of stringa within the column range specified to the value of integer. All following occurences of stringa are incremented by n. Default of this increment is 1. The command LINE, used in this form of the CHANGE command, prevents multiple occurences of stringa within a single line from being incremented.

4. CHANGE 'stringa' USING <replacement range> IN <range> allows the user to replace a string within the range with a string that resides elsewhere in the active dataset. Column ranges may also be specified for both source string and replacement string.

5. Say you have a dataset with lines 72 characters long, in which the first eight characters are an identification number. If you want the 8 digit number to occupy columns 73 through 80 rather than 1 through 8, you would use two CHANGE commands. CHANGE 73/80 USING ALL COLUMNS 1/8 IN ALL will change columns 73 through 80 to match columns 1 through 8 in every line of the dataset. Thus the eight digit numbers now occupy the first and the last eight columns of all

lines in the dataset. To remove the number from the first eight columns, while leaving the number in columns 73/80 you would enter the command CHANGE 1/8/72 TO '' IN ALL. This command removes the first eight columns from the data, but does not allow characters after column 72 to be shifted to the left.

6.The second command in example five illustrated another form of the CHANGE command, CHANGE <column> TO <replacement > IN <range>. If, for example, you have a dataset in which you now wish to put carriage control characters, you would want the first column to be a blank in all lines that are not affected by carriage control. The command CHANGE 1 to ' ' in <range> will shift the entire text one space to the right, making room available for the carriage control characters. The above command inserts a blank in column one.

If, for some reason, you wish to replace whatever is in column one with a blank the proper command would be CHANGE 1/1 to ' ' IN <range>.

Because the CHANGE command is so very versatile and powerful, ingenious users will find innumerable combinations of possibilities, making the CHANGE command probably the most powerful of all WYLBUR editing commands. Careless users will also find it the most dangerous.

CLEAR [ACTIVE|TEXT] [EXEC] [NAME] [TABS]

CLEAR is used to erase the current active file, the current execute file, the current logical tab settings, or the current dsname fields.
where:

ACTIVE — erases the current active file.

TEXT — is an exact equivalent of ACTIVE.

EXEC — clears the current execute file.

NAME — resets the USER, PREFIX, and MEMBER fields to the default sign-on values. These fields are used to form a dsname. See "dsname" in this section.

TABS — erases the current logical tab settings.

CLEAR as a parameter:

LOGOFF [CLEAR] clears the active file and then logsoff.

USE <dsn> CLEAR clears the current active file and brings <dsn> into the active file.

SAVE <dsn> CLEAR saves the dataset and clears the active file.

COLLECT CLEAR clears the active file and gives a line prompt for the first line of a new active file.

The settings of the global parameters remain in effect even after the active file is cleared. If the user does not clear the active file before giving the LOGOFF command, WYLBUR will prompt him to clear it; however, if the system fails, the user breaks the phone connection, or the LOGON command is given without the CLEAR parameter when there is an active file, WYLBUR will save the current active file as Bnnnnn.ACTIVE ON CATALOG. The current execute file is never saved by recovery and need not be cleared before logging off.

CCLLECT [<line number>] [BY <line number>] [MERGE] [<list options>] [CLEAR]

The COLLECT command allows the user to create a new active file or to add new lines to the current active file. Overwriting of existing lines is never permitted, and interleaving is only permitted when MERGE has been specified.

where:

<line number> begins collecting the lines into the active file at the specified line number. If this parameter is omitted, then collecting may begin at one of two points:

(1) If it is the first COLLECT after a USE, CLEAR, ALIGN, JUSTIFY, or NUMBER command, collecting will begin at END (i.e., the first multiple of DELTA that is greater than LAST). Collecting after a CLEAR command begins at line 1, unless another delta is set.

(2) Otherwise, it is a continuation of the last COLLECT and collecting is resumed starting after the last line collected using the same DELTA. Note that in this case WYLBUR also remembers the list options LIST|NOLIST and NUMBERED|UNNUMBERED. See "line number" and "list options" in this section.

BY <line no> is the line increment or DELTA to be used. If this parameter is specified and no line number is specified, then collecting begins at one of two points discussed in the description of the line number parameter, but the specified line increment is used to compute the symbolic line number END. If the line number parameter was specified, but BY was not, the default value is chosen according to the default DELTA rule. See "default DELTA" in this section.

MERGE will not allow overwriting of existing lines in the active file, but will allow interleaving of collected lines with existing lines if line numbers are chosen so that interleaving occurs. For example, if the active file consists only of odd-numbered lines, the command COLLECT 2 BY 2 MERGE will result in prompts for even-numbered lines and the result will be a merged file.

<list options> If the list option LIST is specified, the line in the active file which precedes the next line to be collected will be listed. See "list options" in this section.

CLEAR clears the active file before the new lines are collected.

COMMENT <any text>

The COMMENT command will type the text following the command word 'COMMENT' at the terminal. The character ; is not recognized as a delimiter (in other commands it delimits the command from the comments). If LOG mode (see SET EXEC LOG|NOLOG) is in effect, the COMMENT lines are listed just as all other commands are. In NOLOG mode, only text following the word COMMENT will be typed.

COMPARE <line-number1><line-number2> [<list options>]

COMPAREs line by line the line contents of two ranges until the end of the active file is reached or a comparison fails. A message will be printed giving the line numbers of two lines that were compared last or the two lines at which the comparison failed.

where:

<line-number1> is the beginning line number of the first range. This range will be compared to the second range, the beginning of which is specified by line-number2. The two ranges must be in the active file. See "line number" in this section.

<line-number2> is the beginning line number of the second range. This range will be compared to the first range, the beginning of which is specified by line-number1. The two ranges must be in the active file. See "line number" in this section.

<list options> The default list option is LIST which causes the lines that failed to compare to be listed. See "list options" in this section.

CONDENSE <dsname> <ds location> [<dsn options>] [<format options>] [<hold options>] [ID xxxx] [JCL|NOJCL]

CONDENSE submits a job to condense the specified partitioned data set using the COMPRESS option of the IBM utility IEBCOPY. The STATUS command may be used to determine when the job has finished execution. Whenever an attempted SAVE of a PDS member cannot be completed because of insufficient space in the data set, WYLBUR prompts "CONDENSE?". To have the library condensed, reply "yes", "ok", or "condense"; any other reply aborts both the SAVE command and the CONDENSE job. To determine whether a CONDENSE of a user's library should be done, the command SHOW DSNAMES LIKE <pds name> SPACE should be given to display the number of library tracks that have been used.

where:

<dsname> is the name of the specified partitioned data set to be condensed; see "dsname" in this section. This parameter is required and must be specified first. If the PDS to be condensed is not that of the signed-on user, WYLBUR will respond "data set name NOT YOURS" and then prompt "KEYWORD?". If the user responds with the correct keyword of the user whose PDS is to be condensed, the command will be executed; otherwise, it will be aborted.

<ds location> tells WYLBUR the location of the PDS. See "ds location" in this section for the specific form of this parameter. This parameter need not be specified for catalogued datasets.

<dsn options> The dsn options USER and GROUP may be specified meaningfully in this command. See "dsn options" in this section.

<format options> allows the user to specify the desired record length and how WYLBUR line numbers are to be treated. For the specific form of these options see "format options" in this section.

<hold options> allows the user to place the job and/or its output into HOLD status. See "hold options" in this section.

ID xxxx can be used to specify 1-4 characters that will be used whenever WYLBUR creates a jobname. See "jobname" in this section for details about the WYLBUR created jobname.

JCL|NOJCL JCL specifies that a listing of the condense job is to be printed on the high-speed printer; this is the default. NOJCL will cause the output to not be printed.

```
COPY   [<range>] [[TO <line number>] [BY <line number>] |
       COMBINE | PLUS [-] <line number>]
       [FROM <dsname> [<ds location>] [<dsn options>]
       [<format options>]] [MERGE] [REPLACE] [EXEC] [<list options>]
```

COPY copies all or part of an external, execute, or active file to a specified location in the active file. This command will replace existing lines in the active file only when the REPLACE parameter is specified. In order to copy a range from the active file and delete the old instance of the copied range, see the MOVE command. If COPY with no parameters is given the entire active file is copied to the end of the current active file.

where:

<range> is the range of lines to be copied. If specified, it must be the first parameter. If not specified, the entire active, execute, or external file will be copied. The active file is used if neither the FROM dsname nor the EXEC parameters are specified. See "range" in this section.

Warning: If the user chooses to specify a symbolic line number (see "line number" in this section) and he specifies the FROM dsname parameter, he should be aware that certain line numbers will not exist in the external file (e.g., FIRST in an external file has the value 0 and LAST the value 99999.999).

TC <line no> specifies the beginning line number in the active file to which the range of is lines to be copied. See Note 1.

BY <line no> is the line number increment, or delta, by which the successive line numbers of the copied range are incremented. If not specified, the default value is selected as described in the section "default delta". If the value of delta, specified or default, would cause the copied lines to interleave existing lines in the file, WYLBUR automatically calculates a new value of delta so that interleaving does not take place (see the MERGE and COMBINE parameters for exceptions). See also "default delta" in this section.

COMBINE allows the copied range of lines to retain the original line numbers. Thus it can be used to combine several files while retaining a separate numbering system for each. This implies MERGE, but MERGE need not be specified. E.g., given the ccmmand COPY 3/5 COMBINE FROM #FILE1, and given that the active file contains lines 1, 2, 10, then the copied lines will be copied after line number 2 and before line number 10 in the active file. The copied lines will retain their original numbers, i.e., 3, 4, 5. See Note 1.

PLUS [-]
<line no> specifies a positive or negative increment that is to be added to each line number of the range to be copied. PLUS is like COMBINE, except that COMBINE has an increment of zero. PLUS implies MERGE and COMBINE but they need not be specified. See Note 1.

FROM <dsname> is the name of an external file from which the specified or default range is to be copied. See "dsname" in this section.

<ds location> tells WYLBUR the location of the specified data set. This parameter may be omitted if the dataset is catalogued. See "ds location" in this section.

<dsn options> The dsn options USER and SET or NOSET may be specified. See "dsn options" in this section.

<format options> allows the user to specify the desired record length and how WYLBUR line numbers are to be treated. For the specific form of these options see "format options" in this section.

MERGE allows interleaving--but no replacement--of the copied lines with the existing lines if the line numbers interleave, e.g., the copied lines are odd-numbered and the existing lines are even-numbered.

REPLACE specifies that the copied lines can replace and/or interleave existing lines. This parameter implies MERGE, and it can be used with COMBINE.

EXEC copies all or part of the execute file to the active file.

<list options> The list option LIST will list the line immediately preceding the line copied to and, if you are copying from your current ACTIVE or EXEC file, will list each line as it is copied. LIST, however, is not allowed with COPY FROM <external file> command. The reason is that WYLBUR can read from the external file much faster than it can LIST. To LIST while copying from an external file would simply use up far too much I/O time. See "list options" in this section.

Note 1. If the TO, COMBINE, and PLUS options are omitted, TO END is the default. See "END" in this section.

"CURRENT or *"

CURRENT is a symbolic line number whose value changes and is the number of the last line edited in the active file or last line executed in an execute file. CURRENT for the active file and CURRENT for the execute file are distinct from each other, since you may have a CURRENT in both simultaneously.. CURRENT does not exist in an external file, and is invalid in an external file range. The command USE does not set CURRENT nor is it set if there is no active or execute file. Sometimes the value of CURRENT will not be reset, as for example, after a void range has been specified in an editing command. The symbolic line numbers PREVIOUS and NEXT are dependent on CURRENT and some obvious restrictions apply. For instance, if the value of CURRENT is the last line number in the file, there is no NEXT.

The symbol * as a line number is equivalent to CURRENT. Either may be used as the pointer to the CURRENT line number, but * is also used for other things in WYLBUR; for example, when specifying a dsname (see "dsname" in this section), * refers to the currently set prefix (see SET PREFIX).

The value of * is assumed to refer to the active file except when execute file line numbers are being specified such as in the commands EXEC, LIST EXEC, PUNCH EXEC, SUGGEST EXEC, and COPY EXEC. For these cases, the execute file is assumed.

The commands SET *, SHOW *, and POINT * are associated with the symbolic line number CURRENT and may be used to display or change the value of it.

See "line number" in this section for a general description of all symbolic line numbers.

"default delta"

The default delta, or line increment, is chosen as follows:

(1) If the line number to be incremented is an integer or a symbolic line number (see "line number" in this section), the global DELTA is used (see the SET (SHOW) DELTA command).

(2) If the line number to be incremented is a decimal number, a 1 in the low-order decimal position is used as the increment. For example, if the line number is 50.023, the increment would be .001.

(3) If a line number increment (or decrement) is given, it is used to calculate delta as follows:

(a) The global DELTA is used if the line number increment is an integer.

(b) If the line number increment is a decimal number, a 1 in the low-order position is used.

For example, default delta for the line number and increment FIRST+2.00 would be .01 (case 3b). If *+2 were specified, the global DELTA would be used as the default delta (case 3a).

DELETE <range> [<list options>]

DELETE erases the specified range of lines--the line numbers as well as their contents.

where:

<range> is the range of lines in the active file to be deleted. This parameter is required and must be specified first. See "range" in this section.

<list options> The list option LIST specifies that lines should be listed as they are deleted. NOLIST is the default for this command. LIST cannot be abbreviated as L because L is the abbreviation for LAST; for example, DELETE 12/14 L means delete lines 12 through 14 and the last line of the file. For other list options, see "list options" in this section.

ALTERNATE COMMAND FORM. An alternate form of DELETE consists of a single line number, immediately followed by a carriage return. Warning: The normal abbreviation of "L" and "F" for the line numbers LAST and FIRST cannot be used in this form of the DELETE command.

DELTA See "default delta" above, or "SET DELTA" below.

"ds location"

This parameter tells WYLBUR the location of the specified data set. It is a necessary parameter for the SAVE command for a new data set, but, since CATALOG is the default at Argonne, "ds location" is seldom required otherwise. (See Section 2.7.2 for discussion of disks available to WYLBUR.) The specific form of this parameter is:

[ON {<volume>|CATALOG}]

where:

<volume> specifies the direct access volume on which the data set is to be SAVEd. See "volume" in this section.

CATALOG specifies that the data set is already cataloged. The location of the data set will be found in the system catalog entry for that data set.

If this parameter is omitted, and the dataset is not catalogued, WYLBUR will check whether a volume was specified earlier in a SET VOLUME command. If so, WYLBUR will try to locate the data set on that volume. If it does not find the data set, WYLBUR will give the message "data set name NOT ON volume". If no SET VOLUME command was given earlier and this parameter is omitted, WYLBUR will do one of the following:

(a) If the data set is a member of the PDS LIB, WYLBUR will attemp to locate it by using the system catalog.

(b) If the data set is a sequential data set or a member of a PDS other than LIB, WYLBUR will prompt "VOLUME?" and the user must enter a volume or the keyword CATALOG. Immediately after the volume, the user can also add the option SET to specify that the default volume should be set to this volume just as if a SET VOLUME command had been entered.

"dsn options"

---

dsn options are effectively subparameters of the <dsname> parameter. That is, with only a few exceptions, these options are meaningful in commands that have the <dsname> parameter. The order in which these options are specified within the command does not matter, and they may even be specified before the <dsname> parameter when it is not a positional parameter. The dsn options are as follows:

[USER {&|[&]Bnnnnn}] [SET|NOSET]

The subparameter SET is implied by the commands SET NAME and USE.

A complete description of the individual options follows.

Option	Description
USER	Bnnnnn specifies the user code that is used for the duration of the command to override the value in the saved user field (see "dsname" in this section). It is often used to refer to another user's data set. The form USER & means the signed-on user.
SET and NOSET	SET specifies that the saved prefix and saved member fields (see "dsname" in this section) are to be set to the member name and/or prefix used in the data set name that is specified by the <dsname> parameter in the command. SET is the default for the USE ccmmand, but NOSET is default for all other commands in which this parameter may be specified.

"dsname"

The dsn options associated with the dsname parameter are discussed under the heading "dsn options" in this section. The following discussion of the dsname assumes the reader's familiarity with them.

The WYLBUR data set naming conventions were discussed in section 2.7.1. It described the fully qualified form of a WYLBUR data set name. However, it is seldom necessary to specify a fully qualified dsname in the commands, because WYLBUR will construct all or part of the name for the user from default or preset values. The user should note that the dsname parameter may never be omitted and that it should not contain embedded blanks. Blanks or special characters should be avoided in the 'dsname' form.

Values Used to Construct a WYLBUR dsname

WYLBUR maintains three values to be used in constructing a data set name:

1) A userid (not necessarily the userid under which the user logged on.)
2) a prefix (i.e., a dsname)
3) a PDS member name

Hereafter, these values will be referred to as SAVED USER, SAVED PREFIX, and SAVED MEMBER fields.

After the log-on procedure is complete, or after a CLEAR NAME command, the default values are:

```
SAVED USER   - userid given in response to log-on
               procedure prompt USERID?
PREFIX       - null
SAVED MEMBER- null
```

The user may set the values of the three fields explicitly at any time with the following commands:

```
SET USER [Bnnnnn]
SET PREFIX [pp]
SET MEMBER [member]
```

The values of PREFIX and MEMBER are set implicitly by the USE command because SET is the default parameter of the USE command. The current value of these fields may be displayed with similar SHOW commands as described in this section.

Using WYLBUR to Construct a dsname

Unless overridden by the user, the current value of the SAVED USER is used to construct a WYLBUR dsname in the commands in which dsname is a parameter. The user may override the use of the current values either by specifying the fully qualified WYLBUR name or by specifying the parameter USER Bnnnnn.

If the current value of PREFIX is the desired value, the user ma specify * and thus indicate that the current prefix is to be used in constructing the dsname. Furthermore, the user may add on to the current prefix value by specifying *string. "string" will be concatenated with the current prefix.

The saved member field is used by WYLBUR when accessing a PDS unless the user overrides it by specifying a member name directly as part of the dsname parameter. Specifying a member name of () alone overrides the saved member with a null value. Thus, to use the current prefix but to ignore the current saved member, the user would specify *() as the dsname.

The prefix and saved member may be set by specifying the dsn option SET. When SET is specified in a command, the dsname is constructed and the entire command line is scanned for syntactical and conflicting specification errors. Then the two fields are set to the values contained in the constructed dsname. The prefix field takes the value of the part of the dsname following the user id but excludes any member name. That is, for a constructed dsname of:

Bnnnnn.XYZ(A1)

the prefix field is set to XYZ and the saved member becomes A1. If the member name is explicitly null (for a sequential data set), saved member is set to null. If no member name is given, the field is not changed. The saved user field is not changed by the SET parameter. This field can only be changed by SET NAME/USER commands.

Specifying a dsname for a Non-WYLBUR Data Set

A dsname may refer to a non-WYLBUR data set. But in this case the fully qualified dsname must be specified and it must be enclosed in apostrophes or quotation marks like this: 'dsname'. The saved user field is not used for constructing such a data set name. The quoted dsname is also valid as the PREFIX.

Abbreviations Valid in dsname Parameter

The dsname parameter itself may contain special characters as abbreviations. The special characters used for this purpose are:

$dsname	used instead of 'dsname' unless the dsname contains delimiter characters (blank, comma, equal sign, and parentheses), or lower-case characters with UPLOW set.
#member	used instead of (member)
&Bnnnnn	used instead of USER Bnnnnn

When & occurs alone followed immediately by its delimiting period, the default value is used.

Thus, the general short form of the dsname parameter in the commands is:

[$string|[&[Bnnnnn].][*][string]][#[member]]

Note that when in VERBOSE mode (see the SET TERSE command) any WYLBUR messages referring to a data set name will give it in its fully qualified form. For example, assuming the user is B12345 then to the command:

SAVE #MEMBERA

WYLBUR may respond

MEMBER MEMBERA SAVED IN B12345.LIB

However in TERSE mode the dsname in a WYLBUR message may or may not give an indication of user, depending upon whether or not the user has given the command SET USER. If the data set belongs to the signed on user, and the saved user field is the same as the signed on user, no indication of the user is given in the dsname; otherwise, the dsname is prefixed by "&Bnnnnn".

EDIT [<range>] [SINGLE] [<list options>]

EDIT allows multiple changes to be made to a line or range of lines. Unlike MODIFY, EDIT consists of character for character replacement; thus, it is not possible to insert or delete a character without replacing the remaining characters in the line. To do that use MODIFY or CHANGE.

where:

<range> is the range of lines to be edited. If specified, it must be the first parameter. If not specified, the entire file is assumed. See "range" in this section.

SINGLE causes WYLBUR to prompt "EDIT?" only once per line rather than the default of prompting until the user responds with a carriage return as the first character of the line.

<list options> The list option LIST, which is the default for this command, lists the corrected line; this may be suppressed by specifying NOLIST. For other list options, see "list option" in this section. The first character position of the line on which the user types his corrections is reserved for N

(NOLIST), L (LIST), or one of the special character (listed below) used in conjunction with the ATTN key. If none of these are used, this character position must remain blank.

The line to be edited is always listed first (or just the line number if the list option NOTEXT is specified). The user is then prompted with EDIT? In response the user spaces over to the characters to be replaced and types in the replacement characters directly under the old characters. To replace a character with a blank, a vertical bar, |, must be typed. If a | is typed under a blank, | will be inserted. To end the EDIT of a line, the user responds to the EDIT? prompt with a carriage return as the first character in the line.

In the EDIT (and MODIFY) command, ATTN (or BREAK) is a special key which, when used with other keys, allows processing of the line being edited just as a carriage return does. This is unlike the use of the ATTN key in other WYLBUR commands. The following summarizes the use of the ATTN key in the EDIT command:

(1) (ATTN): When pressed as the first character in response to the prompt EDITS?, it will abort the command and leave the line currently being edited unchanged. However, remember that when ATTN is pressed after the first character in response to the EDITS? prompt, the typed line will be canceled and WYLBUR will prompt EDITS? again for the line being edited. Lines already terminated with a carriage return are not affected.

(2) $(ATTN): Does the editing specified, then terminates the EDIT command. Unedited lines in the specified range will not be listed and prompted. Lines already terminated with a carriage return or alterations in the line terminated with $(ATTN) are not affected.

(3) @(ATTN): Does the editing specified and then gives one more prompt for the current line in SINGLE mode or goes on to the next line in multiple mode.

(4) ¢(ATTN) (on many ASCII terminals the ¢ character is an upward arrow or a caret): Does the editing specified and then switches to the MODIFY command to continue processing the current line. The next prompt will be ALTERS? rather than EDITS? If the current mode is ALTERS, then the switch is to EDITS? This use of the ATTN key can be used to switch back and forth between the EDIT and MODIFY command for any line. However, each new line in the range always starts out in the command originally specified. See MODIFY, below.

"END"

END is a symbolic line number which is after the symbolic line number LAST (i.e., the last line of the file). Its value is the first line number greater than LAST that is a multiple of DELTA. Thus, if the last line number is 10 or 10.1, 10.2, etc. and DELTA is 1, the value of END is 11. That does not mean that line 11 exists, but rather that the line number of the next line to be added at the end of the file would be 11. If the active file is empty, END has the value of the default delta setting (see "default delta" in this section).

EXECUTE [[START] <line number>] [{ACTIVE [DELETE|NODELETE]|
FROM <dsname> [<ds location>] [<dsn options>]
[<format options>]}] [LINES <range>] [CLEAR] [PAUSE]
[NOLOG|LOG] [TERSE|VERBOSE] [SAVE]

EXECUTE will execute a file of WYLBUR commands from an active or external file that has been loaded, and, thus, made into an execute file. Before commands can be executed, the file must be loaded from the current active file or from an external file. If the execute file is loaded from an external file, the active file is not affected by this command. If all or part of the active file is loaded, the active file, or the specified portion of it, will be deleted upon successful loading unless NODELETE has been specified.

Commands to be executed are selected sequentially unless the line number parameter is used to specify a branch instruction in the execute file. Selection of commands continues until the end of the file is reached, an execute break occurs due to an error in the command, the ATTN (or BREAK) key is pressed, or an EXECUTE command is executed from the file with the PAUSE parameter.

where:

START
<line number> specifies the line number where execution is to begin. The default is the line number FIRST if the parameter ACTIVE or FROM <dsname> has been given, or the line number NEXT to continue execution. The pointer to the CURRENT line number for the execute file will be set to that line number. The line number can be specified without the keyword START. See "line number" in this section.

ACTIVE — specifies that the execute file should be loaded from the active file. If neither ACTIVE nor FROM dsname is specified, the current execute file is assumed.

DELETE — DELETE, the default, specifies that the range specified in the LINES parameter or its default be deleted when it has been loaded successfully.

NODELETE — prevents deletion. See DELETE.

FROM <dsname> — specifies that the execute file is to be loaded from an external file. If neither ACTIVE nor FROM dsname is specified, the current execute file is assumed. See "dsname" in this section.

<ds location> — tells WYLBUR the location of the specified data set. See "ds location" in this section. This parameter is not required for catalogued EXECUTE files.

<dsn options> — The dsn options USER and SET or NOSET may be specified. See "dsn options" in this section.

<format options> — allows the user to specify the desired record length and how WYLBUR line numbers are to be treated. For the specific form of these options see "format options" in this section.

LINES <range> — specifies the range of lines to be loaded from the active or external file. The parameter ACTIVE or FROM dsname must be specified for this parameter. If ACTIVE is used, the specified range will be deleted from the active file after it has been loaded successfully. To prevent deletion, specify the NODELETE parameter. If a disjoint explicit range is given, it must be in ascending order. See "range" in this section. Note that when you refer to an external file, the symbolic line numbers FIRST and LAST have the values 0 and 99999.999 respectively. Thus, LINES FIRST/LAST for an external file would mean LINES 0/99999.999.

CLEAR — specifies that the current execute file should be cleared before performing the rest of the EXECUTE command. Normally this parameter is used with ACTIVE or FROM dsname to avoid a prompt to clear the current execute file before continuing execution.

PAUSE — directs WYLBUR to return control to the terminal after all parameters have been set without executing any other commands. This parameter can be used to load an execute file and delay execution until later.

LOG — specifies that all commands should be typed before execution. If this parameter is omitted and a SET EXEC NOLOG was not done earlier in the session, the default sign-on value of LOG will be in effect. If the listing of a command is interrupted by pressing ATTN, an execute break (see Note 1) will occur and the command will not be executed.

NOLOG — If NOLOG is specified, it will be in effect until another EXEC command with LOG or a SET EXEC LOG command is given. NOLOG is the opposite of LOG and inhibits the typing of execute file commands before they are executed. (see also the COMMENT command).

TERSE — specifies that some of the informational messages normally typed by commands are to be suppressed. If this parameter is omitted and a SET TERSE has not been done earlier in the session, the default sign-on value of VERBOSE will be in effect. If TERSE is specified, it will be in effect until another EXEC command with VERBOSE or a SET EXEC VERBOSE command is given.

VERBOSE — is the opposite of TERSE. See the TERSE parameter.

SAVE — specifies that the symbolic line number RETURN should be set to the line number of the next line of the execute file. This provides a branch and link capability within an execute file or a set of execute files. For example: assume an exec file contains the command EXEC SAVE at line 11. This causes the symbolic line number RETURN to be set to the next line, line 12. Further, assume the command EXEC RETURN is at line 20. This would cause the command at the line number RETURN, i.e., line 12, to be executed.

A pointer is maintained to the currently executed line in the execute file. However, this pointer is distinct from the pointer to the CURRENT line number in the active file. The command SHOW * EXEC will show the value of the pointer to the CURRENT line in the execute file.

Section 5.0 explains more about how to write execute files.

Note 1. An exec break occurs when a command being executed in the execute file is aborted for any reason (e.g., the user pressed the ATTN key, the command was incorrectly specified, etc.). EXEC NEXT or simply EXEC can be used to continue executing the command immediately following the command that caused the exec break. To continue executing at the point of interruption, give the command EXEC *.

"FIRST"

FIRST is a symbolic line number which has the value of the line number of the first line in the file. When FIRST is used in a range for an external file, it is assigned the value 0.

"format options"

The <format options> are meaningful in the commands COPY FROM, EXEC FROM, SAVE, and USE.

The format options are defined as:

[LRECL nnn|EDIT|CARD|PRINT] [NUMBERED|INTEGER|UNNUMBERED] [SEQFLD ({END|pos}[,len])]

The <format options> specify to WYLBUR the desired record length (EDIT, CARD, PRINT, LRECL nnn), the position of the line number within the record (SEQFLD), and whether WYLBUR line numbers are part of the data (NUMBER, INTEGER, UNNUMBERED). SEQFLD and INTEGER imply NUMBERED. NUMBERED, INTEGER, and UNNUMBERED imply fixed block or variable blocked format; CARD, PRINT, and LRECL imply fixed blocked format.

WYLBUR can read and write records that are from 1 to 133 characters long. ("Read" refers to the USE, COPY FROM, and EXEC FROM commands. "Write" refers to the SAVE command.) The records may be fixed-length (RECFM=F or RECFM=FB) and variable-length (RECFM=V or RECFM=VB), but records may not be spanned (RECFM=VBS) nor variable length for write. Undefined data sets (RECFM=U) are assumed to be WYLBUR internal or EDIT format.

When data sets are written in EDIT format, the WYLBUR line numbers are always stored, but not as part of the data. Line numbers are normally NOT placed in the records when a fixed blocked data set is written (UNNUMBERED). When such a data set is read, it is renumbered.

A description of the <format options> follows.

EDIT — EDIT format is the default for writing (e.g., SAVE) sequential data sets, and, internally, active files are always treated as EDIT format. PDS members assume as the default the format of the PDS. (See Note 1.)

LRECL nnn — specifies that the data set is to be treated as fixed block (RECFM=FB) with a record length of nnn. nnn may be an integer between 1 and 133 inclusively. If the actual physical block size is not an integral multiple of nnn on a read, any characters remaining after all possible lines of nnn characters have been extracted will become a separate record. See Note 2.

CARD — is an abbreviation for LRECL=80. If UNNUMBERED, NUMBERED, INTEGER, or SEQFLD have been specifed without any explicit format for a sequential data set save, then CARD format is assumed. See Note 2.

PRINT — is an abbreviation for LRECL=133. All provisions for LRECL apply to PRINT. See Note 2.

NUMBERED — On a write operation, NUMBERED specifies that this is a fixed block save and that line numbers should be saved in the record in decimal form with leading and trailing zeros suppressed. On a read operation, it specifies that the data set is to be treated as NUMBERED fixed or variable-length blocked. For non-EDIT format data sets WYLBUR uses the line number stored in the record and deletes from the record the columns that contain the line number.

UNNUMBERED — On a write operation, UNNUMBERED specifies that this is a fixed block save without line numbers in the record. UNNUMBERED may not be specified for saving an EDIT format data set. On a read operation, it specifies that the data set is to be treated as UNNUMBERED fixed or variable-length blocked.

INTEGER — has essentially the same meaning as NUMBERED except that the line numbers will not contain a decimal point and zeroes are not suppressed.

SEQFLD — On a write operation, SEQFLD specifies that this is a NUMBERED fixed block save. On a read operation, it specifies that the data set is to be treated as NUMBERED fixed or variable-length blocked. The subparameters of SEQFLD specify the position and length of the line numbers in the record. The default position is the last 8 positions of each record (SEQFLD=(END,8)). Any column position, pos, and any length, len, less than 10 can be specified. On a write operation, line numbers are truncated on the left to the length specified without checking for loss of digits. The parentheses can be omitted if the length is not changed.

Note 1. Special caution must be used when specifying an explic format for saving PDS members. The format attributes apply to the entire WYLBUR user library and not the library members individually. Thus, if the format of the PDS member does not agree with that of the library, WYLBUR will issue a warning message such as this:

> WARNING--DATA SET TYPE WILL BE CHANGED TO specified format
> OK?

If the user replies OK or YES, the record format field in the dataset control block (DSCB) is changed, but, since the DSCB is in the volume table of contents (VTOC) and is not part of the data, the actual format of the other members of the library will not be affected. Therefore, the new format attributes of the library, as defined in the VTOC, may not reflect the actual format attributes of the older library members. Thus, programs using these members may encounter difficulties, and the CONDENSE program may destroy the old members. WYLBUR will be able to access the old members only if their format is explicitly given. Therefore, it is essential to keep all members in a PDS in the same format.

Note 2. The user should be aware that using these parameters may produce some rather odd looking data if the data set being read does not actually contain fixed records.

HELP

The HELP command prints the phone extension of User Services and the dispatching area in the Applied Mathematics Division.

HOLD <jobid>

The HOLD command allows the user to stop the job from further processing and put the job into HOLD status. The job may be released for further processing with the RELEASE command.

where:

<jobid> identifies the job to be held by jobname or ASP jobnumber. This parameter is required and must be specified first. Jobs not belonging to the signed-on user may not be held. See "jobid" in this section for details about the forms of the jobid.

"hold options"

The form of this parameter is:

<command> [HOLD [JOB]]

Specifying the hold option puts a job being submitted into user hold. It may be released with the RELEASE command.

where:

JOB refers to the job to be held (redundant at ANL).

IF (<expression> <relop> <expression>) command

The IF command evaluates the relational expression in paretheses. If the result is true, the WYLBUR command following the parentheses is executed. If the result is false, the command is skipped without further inspection. This command is primarily useful in execute files. It is described in detail in section 5.0.

INSERT <line number> [NODITTO|DITTO] [<list options>]

INSERT inserts specified lines of text into the active file. Only those line numbers specified will be inserted; insertion of lines with an automatic increment must be done with the COLLECT command.

where:

<line number> is the line number(s) to be inserted into the active file. This parameter is required but can be specified anywhere. See "line number" in this section. Up to 20 line numbers may be specified. WYLBUR will prompt with each line number specified unless the DITTO parameter has been specified. The specified line number(s) cannot already exist in the active file. See Note 1.

DITTO repeats the contents of the first specified line number at the location of subsequently specified line numbers; e.g., INSERT 3,5,7, DITTO prompts for line 3 and then inserts the contents of line 3 also at lines 5 and 7. WYLBUR will prompt only with the first line number (e.g., 3). NODITTO, the default, does not repeat the contents of the first line at other locations.

<list options> The list option LIST, if specified, causes the lir before each inserted line to be listed. NOLIST is the default. See "list options" in this section for other list options.

Note 1. Using the INSERT <line-number> form of this command in an execute file to insert an already existing line in the active file will cause an error message and an execute break (see the command EXECUTE). However, the alternate form described below can be used in the execute file to insert a line or to replace an already existing line.

ALTERNATE COMMAND FORM. An alternate form of the INSERT command allows the user to insert (or replace if the line already exists) a single line in the active file. This form consists of a line number, followed by a single blank character and the contents of the new line of text.

"jobid"

This is a parameter of the CANCEL, HOLD, LOCATE, PRTY, RELEASE, ROUTE, and STATUS commands. It is of the form:

[jobname|jobnumber]

where:

jobname is the jobname on the JOB card.

jobnumber is the job number assigned by the job entry subsystem ASP. It will be a number from 0 through 9999.

"jobname"

The jobname is a 1-8 character identifier beginning with the user code. It is used by ASP to identify a job. When WYLBUR commands result in the preparation of a batch job (e.g., LIST OFFLINE, CONDENSE), WYLBUR will create a unique jobname consisting of the letters 'WYL' and a sequence number. The characters specified in the ID parameter (if supplied) will be appended to the jobname and the prefix shortened to 'W'. For example, assume WYLBUR would have created the jobname WYL123. Then if ID TRY were specified by a user the resultant jobname would be W123TRY. If this user specified ID were X, the jobname would be W123X.

JUSTIFY see the ALIGN command

"LAST"

LAST is a symbolic line number which has the value of the line number of the last line in the file. If a file has only one line, FIRST and LAST are equal. When LAST is used in a range for an external file, it is assigned the value of 99999.999.

"line number"

Line numbers may be expressed in this form:

{number}[{+|-}increment]

where:

number — may be either (1) a constant between .001 and 99999.999, or (2) one of the symbolic line numbers described below.

increment — may be either (1) a constant between .001 and 99999.999, or (2) the current value of DELTA (see SET DELTA).

Concerning the use of line numbers, the following should be observed:

• Leading and trailing zeroes may be dropped (e.g., 1.2 rather than 00001.200) without affecting the significance except in determining a default DELTA in the commands COPY, MOVE, and COLLECT. See "default DELTA" in this section.

• The decimal point may be dropped whenever the number is an integer.

• Additional rules must be observed if this form is used:

number{+|-}increment

(a) No blanks may appear before the sign.

(b) The resultant number must be a number between 0 and 99999.999

Symbolic line numbers

There are 7 symbolic line numbers:

FIRST	PREVIOUS
LAST	CURRENT or *
END	NEXT
	RETURN

Symbolic line numbers are pointers to lines in an active or execute file. Using a symbolic line number moves the pointer to the specified line. They are especially useful in execute files.

FIRST and LAST are the first and last lines in a file and always exist even if the file has no lines. Both have the value 0 if the file is empty. END points to the line number which would be added after the last line. It has the value of DELTA when the file is empty (see the SET DELTA command).

CURRENT is the last line edited or executed; NEXT is the line immediately after CURRENT, and PREVIOUS is the line immediately before it. An "edited" line is one to which one of the following commands has been executed: ALIGN, CENTER, COPY, DELETE, EDIT, INSERT, JUSTIFY, MODIFY, MOVE, NUMBER, POINT, or REPLACE. If no editing or executing has been done on the file, or if the file is empty, PREVIOUS, CURRENT, and NEXT do not exist. Under these conditions, the value given to PREVIOUS, CURRENT, or NEXT will be -1.000 if they are part of an <expression> (see section 5.0 for a definition of <expression>). This value assignment is useful in execute files. See section 5.0 for more details.

For an external file, the symbolic line numbers CURRENT and LAST are assigned the values 0 and 99999.999 respectively, and the symbolic line numbers PREVIOUS and NEXT never exist.

The symbolic line number RETURN is designed to be used in execute files and provides a subroutine facility. RETURN is initialized to 0 and can only be changed with the commands SET RETURN and EXECUTE SAVE.

For a more detailed explanation of these six symbolic line numbers, refer to the individual entries in this section.

<line number>(CR)

This command is an alternate form of the DELETE command. A single line number that is to be deleted can be specified. The line number must be followed immediately by a carriage return (CR). Warning: RETURN and the normal abbreviations of "L" and "F" for the line numbers LAST and FIRST may not be used in this form of the DELETE command.

<line number> text

This command is an alternate form of the INSERT and REPLACE commands. A single line number that is to be inserted (or replaced) can be specified. The line number must be followed by one blank and the text that is to be inserted or is to replace existing text. Warning: RETURN and the normal abbreviations of "L" and "F" for line numbers LAST and FIRST may not be used in this form of the INSERT or REPLACE commands.

```
LIST [<range>] [EXEC] [<list options>] [MARKER c] [BLANK c] [NONL]
     [CLEAN] [(n)] [OFFLINE] [NOEJECT|GRAPH]
     [{CC|ASA}|MC] [COPIES nn] ['title']
     [[SYSOUT class]|UPLOW|FICHE]
     [CASESIM [UPPER|LOWER|NEITHER]] [DOUBLE|TRIPLE] [DARK [n]]
     [SQUASHED] [<hold options>] [COLUMNS m[/n]] [BACK|NOBACK]
     [ID xxxx] [DEST xxxxxxxx|ORG xxxxxxxx]
```

LIST may be used to list all or part of an active or execute file online or offline. Pressing the ATTN key when listing a file online will stop the listing process.

where:

<range> specifies the range of lines to be listed. The default range, ALL, causes the entire file to be listed. If this parameter is used, it must be specified first. See "range" in this section.

EXEC specifies that all or part of the EXEC file is to be listed. The default is that all or part of the active file is listed.

<list options> The UNNUMBERED option lists only the text of the lines in the range. NOTEXT suppresses printing of the text and lists only the numbers. TEXT and NUMBERED are the default list options. See "list options" in this section.

MARKER c c is any character. Any lines in the range that have the marker character in column 1 cause the online listing to be suspended until it is restarted by pressing carriage return. For offline listings, MARKER lines are not printed but cause a page eject.

BLANK c — c may be any character that is to be treated as nontrivial blank. Before the line is listed online or printed offline character c is changed to a blank; it does NOT change the contents of the active file. Use of the parameter SQUASHED will not remove these "blanks".

NONL — suppresses the automatic insertion of the NL (New line or carriage return) character in text being listed at the terminal. It is particularly well suited for on-line plotting at the terminal. However, use of NONL requires that an NL (X'15') be included in the text stream wherever a new line is to occur.

CLEAN — will prevent messages from other terminals from interrupting an online listing. Messages from the operator, however, will interrupt.

(n) — inserts n blanks at the left of each line listed online or offline. If n=0, printer control characters are assumed to be in column 1 of each line of the file and the UNNUMBERED parameter is assumed. ASA characters are expected unless the MC parameter is also used. CC or MC alone may be used to get numbered listings with carriage control. (0) is equivalent to CC UNNUMBERED.

OFFLINE — signals part or all of the active file to be listed on the highspeed printer. The default option is that the file is to be listed online, i.e., at the terminal.

NOEJECT — permits lines to be printed to the bottom of the page and over the page perforations. The default is an automatic page eject after printing 60 lines.

GRAPH — is equivalent to NOEJECT.

CC — causes the character in column 1 of each line in the range to be treated as a printer control character. See "printer control characters" in this section for permissible printer control characters. See also the (n) parameter.

ASA — is equivalent to CC.

MC — is valid only for listing files created by some of the IBM assemblers or compilers or fetched using the CCNTROL parameter. This parameter specifies machine printer control characters, rather than ASA characters. Also see the (n) parameter.

COPIES nn — allows up to 99 copies to be made of the output. The default is one copy.

'title' — permits a title of up to 72 characters to be printed on the output after the JCL and before the listing of the file.

SYSOUT class — class may be one of these SYSOUT output classes: A, B, or F (see "output classes" in this section). If this parameter is omitted, the default is class A--printed output.

UPLOW — specifies upper and lower case printed output.

FICHE — specifies microfiche output. It is equivalent to SYSOUT F.

CASESIM — CASESIM allows you to simulate upper and lower case on high-speed print chains other than the TN chain and is valid only with the OFFLINE parameter.

UPPER — It will translate the text and cause upper-case letters to be underlined if UPPER was specified or

LOWER — lower-case letters if LOWER was specified. Also, in both of these cases, any character on the TN (UPLOW) print train will have some similar representation (by overprinting two characters). A backspace character will be printed as "%". Any unprintable characters (i.e. characters with no printable representation on the TN train) will be

NEITHER — printed as "% " (instead of blank). If NEITHER is specified, neither case of letters will be underlined; however, a CASESIM representation of the backspace, TN, and unprintable characters will be used.

DOUBLE | TRIPLE — produces a listing with double or triple spacing. Single spacing is the default.

DARK n — produces an offline listing in which each line is printed two (or n if n is specified) times. Sometimes overprinting can facilitate better reproduction, but this parameter should be avoided whenever possible because it reduces printing speed considerably.

SQUASHED — squashes all multiple blanks to single blanks before listing the line. The active file is not changed. This parameter is very useful for looking at batch output since listing lines containing many blanks is very slow at a terminal.

<hold options> — allows the user to place the job and its output into HOLD status. This option is meaningful for OFFLINE listings only.

COLUMNS m/n will cause only columns m through n of a line to b listed. m defaults to column 1; n to column 133 if not specified.

BACK This parameter is only valid if the parameter OFFLINE is also used. BACK simulates backspacing of a terminal on an offline listing. An upper case backspace (shift and backspace, or CNTL and backspace) will be treated as a backspace only after the command SET BACK has been given; otherwise, it will erase the characters. The default value at sign-on time is NOBACK.

BACK can be used to underline characters or to create characters like Ø by backspacing rather than using the printer control character + (see "printer control characters" in this section).

Note that currently the ALIGN, JUSTIFY, and CENTER commands will treat backspaced characters as any other characters: i.e. WYLB UR will count backspace characters, plus the overstrike or underline characters, as if they were regular characters. Thus lines with backspace characters will appear to WYLBUR to be much longer than they look in print, and will be truncated to 133 characters, causing the over-strike characters to appear in unexpected places, and, perhaps, cause loss of text. Therefore, it is essential that you have your text ALIGNed, JUSTIFYed or CENTERed exaxtly as you want it before you add backspace characters.

If you attempt to backspace beyond the WYLBUR line number, characters will be lost. That is, assume ycu typed character 'A' into column 1 and then backspaced three character positions, two of those backspaces will be "lost".

NOBACK This is the default. Backspace characters will be treated as normal characters in the online or offline listing.

ID xxxx can be used to specify 1-4 characters that will be used whenever WYLBUR creates a jobname. See "jobname" for a description of the WYLBUR-created jobname.

DEST xxxxxxxx can be used to specify a destination for job output. It is used to create an appropriate ASP //*MAIN card. The destination can be LOCAL for the computer center printers, RADSnn for RADS station nn, or RMOnn for HASP workstation nn. The default is LOCAL and must be used if UPLOW is specified.

ORG xxxxxxxx has the same effect as DEST.

"list options"

[LIST|NOLIST] [NOTEXT|TEXT] [NONL] [NUMBERED|UNNUMBERED|NONUMBER|INTEGER]

The list options are actually a collection of associated keyword parameters valid in most editing commands (i.e., CHANGE, COLLECT, COMPARE, COPY, DELETE, INSERT, LIST, MODIFY, MOVE, POINT, REPLACE). The parameters are as follows:

where:

LIST	causes the lines to be listed.
NOLIST	prevents lines from being listed.
NOTEXT	causes line numbers to be listed, but not the text of the lines.
TEXT	causes the text of the lines to be listed, but not the line numbers.
NUMBERED	causes the text and line numbers to be listed.
UNNUMBERED	supresses the line numbers and causes the text of lines to be listed beginning in column 1.
NONUMBER	is similar to UNNUMBERED except that blanks are substituted for the line numbers.
INTEGER	causes the text of lines to be listed with the line number listed as an eight digit integer (with leading and trailing zeroes as necessary).

Some editing commands like INSERT, MOVE, and COPY have a default NOLIST option. Other commands like EDIT, MODIFY, and CHANGE have a default LIST option. The LIST command, used with the COLLECT or COPY commands, causes the immediately preceding line to be listed.

LOCATE [jobid|badgenumber [BADGE|B]]

LOCATE is a synonym for STATUS. See STATUS for the format of the responses from the LOCATE command.

{LOGOFF|LOGOUT|SIGNOFF} [CLEAR]

LOGOFF terminates the session.

where:

CLEAR — erases the current active file. If this parameter is not specified and there is an active file, WYLBUR will prompt to verify that it is OK to clear it. Any negative response to the prompt causes the command to be aborted and the session to continue.

{LOGON|SIGNON} [CLEAR]

LOGON allows a new user to sign on a terminal without having to dial the computer again.

where:

CLEAR — erases the active file of the old user. If the old user has an active file and CLEAR is not specified, the file is saved and cataloged under the name Bnnnnn.ACTIVE.

MODIFY [<range>] [SINGLE] [<list options>]

MODIFY allows insertion, replacement, and deletions in the contents of each line in the specified range. Successive lines are prompted for alterations until the range is exhausted.

where:

<range> — is the range of lines that are to be changed.

SINGLE — causes WYLBUR to prompt for alteration of a line only once.

<list options> — The list option NOLIST means that WYLBUR will not type the changed line but will still issue prompts. The list option LIST is the default.

For each line the following process is carried out.

(1) The unmodified line is typed (or just the line number if the list option NOTEXT is specified).

(2) WYLBUR prompts ALTER? allowing the user to make alterations.

(3) The user signals the end of his alterations with a carriage return.

(4) WYLBUR types the new image of the line for inspection, unless NOLIST is specified, and prompts for more alterations to the line unless SINGLE is specified.

(5) The user signals his desire to move on to the next line by answering the alter prompt with an immediate carriage return.

Alterations may be one of the following:

INSERT (I)--Type the letter I immediately below the character in the line before which characters are to be inserted. This indicator is followed by the string of characters which are to be inserted. For example, if you want to insert NON before the E in ENTITY, you type an I under the E, then type NON. The string of characters to be inserted is terminated by a carriage return.

REPLACE (R)--This indicator is typed immediately below the first character to be replaced in the line. The string of replacement characters is typed immediately following the R. As many characters are replaced as there are characters in the replacement string. The string of characters to be replaced is terminated by a carriage return.

DELETE (D)--A string of characters to be deleted from the line is designated by typing a D under the first character to be removed. The deletion range may be terminated by typing an I, an R, or a carriage return to stop deletion. If terminated by an R, the characters following the deleted string are replaced by the replacement string that follows the R. If the deletion range is terminated by an I, the deleted characters are replaced with the string to be inserted. If the deletion range is terminated by a carriage return, all characters from the first D up to the carriage return are deleted. A deleted character is not replaced by a blank. Instead, WYLBUR shifts the remaining text to the left by the number of spaces deleted. Although you may type as many D's as you wish, only a D underneath the first character in the string to be deleted is necessary.

BLANK (B)--Individual characters may be blanked in the line by typing a B under each position to be blanked. Only those positions are blanked and no shifting of the line takes place. Any other alteration may follow the last B on a line.

The printing of the altered image of the line may be suspended by yping an N indicator into the alteration line before any other lteration indicators. An L indicator may be typed to override a NCLIST option and to have the altered image listed.

Typing L and a carriage return, causes WYLBUR to type the image o the line being modified as it currently stands.

The use of the ATTN key in combination with special characters or alone is the same as in the EDIT command. See the EDIT command in this section for a discussion of the ATTN key.

```
MCVE  <range> {TO <line number> [BY <line number>]| COMBINE |
      PLUS [-] <line number>} [MERGE] [REPLACE] [<list options>]
```

MOVE copies a range of lines from one place in the active file to anctner and deletes the old copy of the lines. It functions exactly like the COPY command with these exceptions:

(1) The old lines which were copied are deleted.

(2) It is not possible to apply the MOVE command to an external or execute file as the design of WYLBUR allows the user to take destructive action only upon the current active file.

(3) The range must be specified and TO, PLUS, or COMBINE must be specified.

For a description of the parameters, see the COPY command in this section.

"NEXT"

NEXT is a symbolic line number that refers to the line immediately fcllowing CURRENT. If the value of CURRENT has not been set, NEXT does not exist. Also if the value of CURRENT is greater than or equal to the last line number in the file, NEXT does not exist. NEXT is not defined for an external file.

```
NUMBER  [<line number>/<line number>]
        [[BY <line number>] [{FROM|START} <line number>] |
        TIMES <line number> | PLUS [-] <line number>]
```

The NUMBER command renumbers part or all of the active file.

where:

line-no/line-no gives the beginning and ending line numbers in the active file that are to be renumbered. If the user specifies only one line number, all of the file is renumbered beginning with that number. If no number is specified, all of the file will be renumbered. This parameter must be specified first if it is used. See "line number" in this section.

BY <line-no> is the increment to be used to derive successive line numbers. If this parameter and the TIMES or PLUS parameter is not given, the specified range (or all of the file) will be renumbered using the current value of the global DELTA (see SET (SHOW) DELTA). If the increment is too large for the range given, a smaller increment that fits will be used.

FROM <line-no> specifies the first line number for the range to be renumbered. See "line number" in this section. START is identical to FROM.

TIMES <line-no> is the scale factor that multiplies each of the existing line numbers in the range to be renumbered to form the new line numbers. Although the scale factor may be any line number, including symbolic line number, it will usually be a numeric value.

PLUS <line-no> specifies a positive or negative increment to be added to each line number in the range <line number/line number> (or all of the file). Although the increment may be any line number, including symbolic line number, it will usually be a numeric value.

"cutput classes"

Cutput classes that may be specified are:

A--upper case only print train (the default in all commands)
B--card punch
F--microfiche printer (essentially same character set as the TN train on impact printers).

POINT same parameters as in LIST command

The POINT command has all the parameters of the LIST command. See LIST for a description of the individual parameters.

POINT functions exactly as the LIST command except that it updates the pointer to the CURRENT line (or *) as it lists a line. (See "line number" and "CURRENT" in this section) The list option NOLIST can be used to set the pointer to the CURRENT line without listing any lines. For example, POINT 6/10 NOLIST will set the pointer to the CURRENT at line 10 if it exists. If line 10 does not exist, the pointer will be set to the highest line number less than 10. If no lines exist in the range 6/10, the pointer will be unset. The lines will not be listed.

"port-id"

The port-id may be one of these:

n	is the line number assigned to a terminal in the identification message typed at the start of the signon procedure.
Bnnnnn	is the user code under which the user signed on.
OPERATOR	is the operator's console in the machine room.

"PREVIOUS"

PREVIOUS is the symbolic line number of the line immediately preceding the symbolic line number CURRENT. PREVIOUS is not set if the value of CURRENT has not been set, or if CURRENT is less than or equal to the first line in the file. PREVIOUS is not defined for an external file.

"printer control characters"

A complete set of printer control characters is available. In all cases, the control specified is performed before the line is printed. The control characters are:

Printer Control Characters	Function
blank	single space; skip to next line
1	page eject; skip to top of next page
0	double space; skip to next line + 1
-	triple space; skip to next line + 2
+	no skip; used for underlining or overstriking.

PRTY <jobid> [S|L|N|H|T]

The PRTY command enables users to increase the scheduling priority of their batch jobs, regardless of how they were submitted. The priority levels and their charging factors are:

where:

<jobid> identifies the job to have its priority raised by jobname or ASP jobnumber. This parameter is required. Jobs not belonging to the signed-on user may not have their priority raised.

Priority		Charging Multiplier	
T	(top)	3.0	
H	(high)	1.5	
N	(normal)	1.0	
L	(low)	0.8	
S	(standby)	0.8	(but no ration charged)

PUNCH [<range>] [NUMBERED|UNNUMBERED|INTEGER] [EXEC]
[LIST|NOLIST] [hold options] [COLUMNS n[/m]] [ID xxxx]

PUNCH produces punched card output of all or part of an active or execute file.

where:

<range> is the range of lines in the active file that is to be punched. If not specified, the entire file will be punched. If specified, it must be the first parameter. See "range" in this section.

JMBERED causes 8-digit line numbers to be punched in columns 73-80 of the card (even if this means replacing previously existing data in these columns). See note 1.

INTEGER	INTEGER has the same meaning as NUMBERED, except that the line number is an 8-digit integer. See Note 1.
UNNUMBERED	specifies that the WYLBUR line numbers are not to be punched in columns 73-80 of the cards. See Note 1.
EXEC	causes all or part of the current execute file to be punched.
LIST	LIST produces an offline listing of that part of the file that is punched.
NOLIST	NOLIST, the default, means that no listing of the punched file will be produced.
<hold options>	allows the user to place the job and its output into HOLD status.
COLUMNS n[/m]	specifies the beginning (n) and ending columns (m) on a card image that are to be punched. The value of n defaults to 1; m to 80 if not specified.
ID xxxx	can be used to specify 1-4 characters that will be used whenever WYLBUR creates a jobname. See "jobname" for a description of the WYLBUR-created jobname.

Note 1. If this parameter is omitted entirely, line numbers are punched in those records in which columns 73-80 are blank; those records in which these columns are not blank, are not affected.

"range"

A range is defined to be either an "explicit range" or an "associative range". Each of these is defined below.

Explicit range

An explicit range may be of three forms:

x x/y x(z)

where:

x	is a line number (for a definition see "line number" in this section) giving the beginning of the range.
y	is a line number giving the end of the range.

(z) is a positive integer enclosed in parentheses designated as the count; e.g., 12(5) means 5 lines starting at line 12.

Notes: If the range x/y is used, x≤y. If x=y, only x need be specified. The first and last line of the active file may be referred to simply as FIRST and LAST. The explicit range ALL is equivalent to FIRST/LAST.

An additional form of an explicit range, sometimes referred to as a "disjoint range", is a list of up to 10 line numbers. The list can contain single lines or groups, and the ranges can overlap (for an active or execute file), but each group counts as two of the numbers allowed. For example, 10, 21, 20, 12, LAST, 125.2 is a valid range for an ACTIVE file.. The use of a disjoint explicit range has two restrictions:

1) A disjoint explicit range for an external file must be in ascending order and cannot overlap.

2) A disjoint range CANNOT be used if an occurrence list is present in an associative range. The occurrence list is specified by the option (x,x,...,x/y); see the description of this option below. E.g., List 'A' (1) in 4/6, 6/8, 9 is not valid.

Associative range

An associative range is a group of lines, the contents of which must satisfy certain conditions. Furthermore, in order to be included in the associative range, the group of lines must be within the specified explicit range or the default explicit range ALL (i.e., the entire file).

Basically, a line is included within the associative range if it has met the following three conditions in the order specified:

(1) The line is within the explicit range (specified or default).
(2) The line passes the string test, which means:
    (a) If a single string is specified, that string must or must not occur in the specified or default columns.
    (b) If multiple strings are specified, those strings must or must not occur in the specified or default columns and they must satisfy the logical conditions AND or OR.
(3) The line is in the specified (if any) occurrence list.

The exact form of an associative range is:

[¬] 'string' [m[/n]] [{AND|OR} [¬] 'string' [m[/n]...]
[(x,x,...,x/y)] [IN explicit range]

where:

¬ means that the string condition has been satisfied if the line does NOT contain the string within the columns searched.

'string' may consist of alphanumeric characters, special characters, and blanks. The null string is also allowed. The string condition has been satisfied when a line contains 'string' in the columns searched.

m/n restricts the search for 'string' to columns m through n in a given line. If only m is specified, 'string' must begin in that column. If m and n are omitted, the entire line (columns 1 through 133) is searched. When the null string is specified, m/n must be omitted.

AND|OR Multiple string tests can be specified by using AND or OR. All tests separated by AND and any one or more tests separated by OR must be successful in order for a line to pass the string test. AND and OR cannot be mixed in a single range. The maximum number of string tests that may be specified is about 6 (depending on the length of the strings). A message will be given if too many tests have been specified.

(x,x,...,x/y) Specifies an occurrence list which restricts the range to the ordinal subset of lines that meet the string test. x and/or y specify the ordinal positions of that subset. Thus, (3) and (3/7) restrict the subset of lines to the third line and the third through seventh lines, respectively, if they are in the explicit range and satisfy the string tests.

If x and y are omitted, all lines which pass the string test and which are in the explicit range are included in the associative range.

IN expl. range gives an explicit range (see above) which will limit the group of lines to searched. The default, if omitted, is the entire file. The entire file can be explicitly specified by IN ALL or IN FIRST/LAST.

Note: All lines except blank lines have one instance of a blank string at the end of the line that make the line 133 characters long. Thus, the correct associative range to retrieve blank lines is the null string--'' or "". Associative ranges consisting of one or more blanks, such as ' ' or " ", will retrieve lines which have at least one non-blank character; blank lines will not be retrieved.

READ [STRING <S-variable>|VALUE <list of variables>] [EXEC]
[[USING] <line number>] [COLUMNS m[/n]] [DELETE]
[PROMPT 'string']

READ is designed to be used in execute files order to obtain commands or values of variables from the active file from or the user at the terminal.

where:

STRING|VALUE — Both of these parameters are defined in section 5.0. They are primarily useful in execute files. If both of these are omitted, the READ command executes the result as a WYLBUR command.

EXEC — indicates the command is operating on an execute file rather than the active file.

USING <line no> — specifies that the line or part of a line read is a line from current active or execute file. Only one line number may be specified. If this parameter is omitted the user is prompted at the terminal for the contents to use in the command.

COLUMNS m,n — means that the contents of the specified columns in the given line number are to be read. If beginning and ending columns are given, they must be separated by a slash, e.g., 2/5.

DELETE — may only be used on the active file. DELETE specifies that after the contents of the specified line have been obtained, that line should be deleted before the contents are processed as a command, as a string, or as a value to be assigned to variables.

PROMPT 'string' — The default prompt is 'ENTER? '. This parameter allows the user to specify any prompt he wishes. For example:

```
READ PROMPT 'ENTER USE COMMAND--'
```

would cause ENTER USE COMMAND-- to be printed at the terminal. The user would then enter a command which would then be executed.

RECATALOG <dsname> ON <volume> [<dsn options>]

RECATALOG catalogs the specified data set in the system catalog. The data set is cataloged whether it was already cataloged or not. RECATALOG is also allowed as a parameter of the SAVE command.

where:

<dsname> is the name of the cataloged data set that is to be recataloged. The user may recatalog data sets not belonging to him. In that case, WYLBUR responds "data set name NOT YOURS", and then prompts "KEYWORD?". If the user can respond with the correct keyword of the user whose data set is to be recataloged, WYLBUR will execute the command; otherwise, the command will be aborted. This parameter is required and must be specified first. See "dsname" in this section.

ON <volume> is the name of the disk volume on which the data set resides. This parameter is required.

<dsn options> The dsn options USER and SET or NOSET may be specified. See "dsn options" in this section.

RELEASE <jobid>

The RELEASE command allows the user to release a job or a job's output from HOLD status.

where:

<jobid> identifies the job to be released by jobname or ASP jobnumber. This parameter is required. Jobs not belonging to the signed-on user may not be released.

RENUMBER [<line number>/<line number>]
[[BY <line number>] [{FROM|START} <line number>] |
TIMES <line number> | PLUS [-] <line number>]

The RENUMBER command renumbers part or all of the active file. It is identical to the NUMBER command which is described fully in this section.

REPLACE [<range>] [DITTO|NODITTO] [<list options>]

REPLACE allows the user to replace the contents of any specified range in the active file. WYLBUR will prompt with successive line numbers in the specified range; the user types the new contents of each line after the prompt until the range is exhausted. If the DITTO option is specified, WYLBUR will prompt only once and uses the line to replace all other lines in the range. If the user replies to a prompt only with a carriage return, WYLBUR will not replace that line and will proceed to the next line in the range.

where:

<range> is the range of lines in the active file that is to be replaced. See "range" in this section. If used, this parameter must be specified first.

DITTO specifies that the new contents of the first line in the range is to be used to replace the contents of all other lines in the range.

NODITTO assumes that the contents for each line in the range is unique and will be entered separately for each line. It is the default.

<list options> The list option LIST will list each line before it is replaced. LIST cannot be abbreviated as L because L is the abbreviation for LAST; for example, REPLACE 12/14 L means replace line 12 through 14 and the last line of the file. The list option NOLIST is the default. See "list options" in this section.

ALTERNATE COMMAND FORM. A single line may be replaced (or inserted if it does not exist) by giving the line number, followed by a blank, and then the contents of the line. Warning: The normal abbreviations "L" and "F" for the symbolic line numbers LAST and FIRST may not be used in this form of the REPLACE command.

RESTORE [CLEAR]

RESTORE is used to move the current execute file to the active file so that editing can be done on the file. The current active file must be cleared before the restore; the execute file is cleared after the restore. WYLBUR will prompt CLEAR? if there is an active file and CLEAR was not specified on the RESTORE command.

"RETURN"

RETURN is the value of the symbolic line number set to the value of the symbolic line number NEXT in the execute file whenever the SAVE parameter is specified on an EXEC command. RETURN may also be set by the command SET RETURN.

ROUTE <jobid> <destination>

ROUTE may be used to change the destination of the printed output of a job in the ASP queues. It changes the ORG parameter of the //*MAIN card. Datasets routed in the //*FORMAT card are not rerouted, however.

<jobid> identifies the job to be rerouted by jobname or ASP jobnumber. This parameter is required. Jobs not belonging to the signed-on user may not be rerouted.

destination is the new destination for the job's output. This may be a RADS or RJP station name or LOCAL for the AMC computer area.

RUN [<range>] [NUMBERED|UNNUMBERED|INTEGER] [EXEC]
[<hold options>]

RUN has exactly the same function and operand syntax as SUBMIT. See SUBMIT below for a description.

SAVE <dsname> [<ds location>] [<dsn options>] [<format options>]
[LINES <range>] [SCRATCH|REPLACE] [CLEAR] [(n)] [EXEC]
[SSI <hex number>]

SAVE allows the user to save a copy of the current execute or active file as an external file. See the USE command for retrieving saved data sets.

where:

<dsname> is the name of the data set to be saved. If the data set is not to be saved under the account of the signed-on user, the user must supply the correct keyword of the user under whose account the data set is to be saved. See Note 1. This parameter is required and must be specified first. See "dsname" in this section.

<ds location> tells WYLBUR the disk volume on which to save the specified data set. See "ds location" in this section. See Section 2.7.2 for a list of disks available to WYLBUR.

<dsn options> The dsn options USER, SET and NOSET may be specified.

<format options> allows the user to specify the desired record length and how WYLBUR line numbers are to be treated. For the specific form of these options see "format options" in this section.

LINES <range> cause the specified range of lines in the active or execute file to be saved. See "range" in this section.

SCRATCH SCRATCH is synonomous with REPLACE. To SAVE a new version of an existing sequential data set or a PDS member by the same name requires that this parameter be specified. REPLACE will cause the old sequential data set to be erased before the new one is saved. In the case of a PDS member, this parameter will cause the new member to be added to the PDS and the directory to be updated after the new member has been written. The old member is not erased; it still uses PDS space until the PDS is condensed (see below). If this parameter is omitted and the request is to SAVE a data set with an already existing name, WYLBUR will prompt REPLACE?; responses other than YES, OK, REPLACE (or any abbreviation of REPLACE) will cause the SAVE request to be aborted. Also, see Note 1.

REPLACE is equivalent to SCRATCH.

CLEAR may be used to erase the current active or execute file after the specified data set has been SAVEd. If the SAVE is unsuccessful, the file is not erased.

(n) specifies the blocking factor such that the data set BLKSIZE will be LRECL*n. If any formats other than EDIT are specified, the record format will be fixed blocked. If the records in the active file are shorter than the specified LRECL, they are left-justified and right-padded with blanks; records longer than the specified LRECL are truncated without any notification (use SHOW SIZE to check if any truncation might occur).

EXEC causes the execute file to be saved. If not specified, the active file is assumed.

SSI is only meaningful for saving PDS members. It specifies that the SSI (System Status Information) is to be the 1-8 hex digits specified. If less than 8 digits are given, this field is always expanded by right-adjusting and filling with zeroes. If SSI is not specified, the default is 0000YNNN where Y is the right-most digit of the current year and NNN is the Julian date.

Occasionally when the user issues a command to SAVE a member of a PDS, WYLBUR will reply that the library is full and asks the user whether it is OK to condense the library. A positive response of 'YES', 'OK', or 'CONDENSE' will cause WYLBUR to submit a utility job which will condense the library.

Also see the CONDENSE command.

Note: If the data set to be saved or scratched does not belong to the signed-on user, WYLBUR will respond "data set NOT YOURS" and then prompt "KEYWORD?". If the user responds with the correct keyword of the user whose data set is referenced, the command will be executed; otherwise, it will be aborted.

SCRATCH <dsname> [<ds location>] [<dsn options>] [PDS]

SCRATCH allows the user to scratch any external file. The SHOW DIRECTORY and SHOW DSNAMES also have a SCRATCH parameter.

where:

<dsname> is the name of the external file that is to be deleted. If the user wishes to scratch an entire PDS, he must specify the PDS operand or explicitly specify a null member and give a positive response tc the verification prompt. If the data set does not belong to him, WYLBUR will respond "data set NOT YOURS" and then prompt "KEYWORD?" If the user responds with the correct keyword of the user whose file is to be scratched, the command will be executed; otherwise, the command is aborted. This parameter is required and must be specified first. See also "dsname" in this section.

<ds location> tells WYLBUR the location of the specified data set. See "ds location" in this section.

<dsn options> The dsn options USER and SET or NOSET may be specified. See "dsn options" in this section.

PDS specifies that the entire data set is to be scratched even if it is a PDS.

SET {BACK|NOBACK}
SHOW BACK

SET BACK alters the function of the backspace key. If BACK has been set an upper case backspace (a shift or CTRL H and backspace) will enter a backspace in the file. On 2741's a lower case backspace continues to erase the previous character. On ASCII terminals CTRL-X must be used to perform character erasure when BACK is set.

SET NOBACK returns the backspace key to its usual function.

SHOW BACK tells which option--BACK or NOBACK--is in effect.

```
SET {COMM|NOCOMM}
SHOW {COMM|NOCOMM} [<port-id>]
```

The SET NOCOMM command prevents the user's terminal from receiving any messages sent by any other user (including the operator) with the TO command. The message:

Bnnnnn - NOT ACCEPTING MESSAGES

will be printed at the sender's terminal whenever the destination terminal has NOCOMM set.

The SET COMM command allows the terminal to accept messages sent by a TO command from another terminal. This mode is in effect at the beginning of a new session.

The commands SHOW COMM and SHOW NOCOMM display the current COMM setting in effect. See "port-id" in this section for an explanation.

```
SET {CURRENT|*} <line number> [EXEC]
SHOW {CURRENT|*} [EXEC]
```

---

where:

CURRENT — sets or displays the value of the symbolic line number (see "line number" in this section). CURRENT and * are equivalent.

<line number> — specifies the line number value to which the symbolic line number CURRENT is to be set. See "line number" in this section.

EXEC — sets or displays the value of the CURRENT symbolic line in the execute file instead of the active file.

SHOW CURRENT is actually just one of the possibilities of the SHOW <line number> command.

SET DELTA xxxxx.xxx
SHOW DELTA

SET DELTA sets the global value of the line number increment, DELTA. xxxxx.xxx may be any value between .001 and 99999.999 inclusively. DELTA has a default value of 1.000 unless specified otherwise. See also "default delta" in this section.

It is wise to use a DELTA value large enough that lines may be added between line numbers. A DELTA value of .001 leaves no such space available.

SET ESCAPE <character>
SHOW ESCAPE

SET ESCAPE allows the user to set an escape character. The escape character signals the preprocessor that text substitution should be made in a command line before WYLBUR executes the command. Initially, the escape character is null. SHOW ESCAPE displays the escape character. These commands are primarily useful in execute files. They are described in detail in section 5.0.

{SET|SHOW} EXECUTE [LOG|NOLOG] [TERSE|VERBOSE] [AUTO|NOAUTO]

SET EXEC allows the user to set two modes governing what is to be typed at the terminal during exec file processing. It also allows the user to control automatic execution of the user's #LOGON file at log-on time.

SHOW EXEC displays which of the three sets of parameters is in effect: LOG or NCLOG, TERSE or VERBOSE, AUTO or NOAUTO.

where:

LOG — LCG specifies that all commands should be typed (or logged) before execution.

NCLOG — suppresses listing of commands before execution. However, EXEC commands in the execute file are still logged even in NOLOG mode unless TERSE is in effect.

TERSE	TERSE specifies that some of the informational messages issued by commands are not to be typed. It also stops logging EXEC commands in the execute file if NOLOG is also in effect.
VERBOSE	is the opposite of TERSE and causes such messages and commands to be typed.
AUTO	causes WYLBUR to begin each new WYLBUR session with the command: EXEC FROM #LOGON NOLOG TERSE as if the user had typed it at the terminal. The AUTO parameter is remembered across terminal sessions until reset with the NOAUTO parameter.
NOAUTO	The default for each new account, is no automatic execution of #LOGON. NOAUTO may be used to prevent automatic execution of #LOGON at sign-on time. Note that once the command SET EXEC AUTO has been given, it will remain in effect for every WYLBUR session until the command SET EXEC NOAUTO is given.

SET LENGTH n
SHOW LENGTH

SHOW LENGTH displays the current value of LENGTH.

where:

n	is the maximum number of characters that the user normally wants in any line of text as the active file is collected and edited. n may be from 1 to 133 characters inclusively. The initial value at signon is 72.

If the specified or default length is exceeded, WYLBUR will accept the line but will issue a warning message that y characters are in line x.

SET MEMBER [member]
SHOW MEMBER

SHOW MEMBER and SHOW NAME will display the current value of MEMBER.

where:

member — consists of 1 to 8 alphanumeric characters that will be the default member name to be used in forming a dsname. See "dsname" in this section. If member is omitted, the member name is set to null, which is also the default member name at sign-on time.

SET MODE [RETRY|NORETRY] [MODIFY|EDIT]
SHOW MODES

Command retry provides the capability of modifying the text of the last command submitted and re-executing it (see note 2). If the RETRY parameter has been specified, WYLBUR will automatically prompt for the command retry when errors occur. Automatic prompts never occur for commands that execute normally or are aborted by striking the ATTN key. However, at the COMMAND? prompt (? in TERSE mode) a user may always request a retry prompt for the previous command, even in NORETRY mode (see note 1). Command retry can not always be done for the previous command, because WYLBUR must destroy the text copy of some commands in order to execute them. If the command text has been destroyed, WYLBUR simply ignores the retry request. Also, the user should be careful about requesting retry after interrupting commands like MOVE, COPY, and CHANGE. It is likely that WYLBUR will have partially processed such a command, and re-executing the same command is not likely to be what the user wishes to do.

where:

RETRY — allows the user to enter automatic retry mode. This means that whenever a WYLBUR command fails to execute (e.g., due to a syntax error), WYLBUR will prompt the user to modify the command so that he can resubmit it for execution. Retry may also be invoked for commands that executed satisfactorily; see Note 1.

NORETRY — allows the user to leave automatic retry mode. However, retry may still be invoked for the previous command and the current command line even in this mode; see Note 1. This is the initial value at signon.

MODIFY — causes WYLBUR to prompt ALTERS? After this prompt the user may change the previous command just as he might if he had given the command MODIFY SINGLE NOLIST. This is the initial value at signon. The user has full EDIT/MODIFY facilities and may use the special ATTN key functions to change back and forth between EDIT and MODIFY.

EDIT causes WYLBUR to prompt EDITS? After this prompt the user may change the previous command just as he might if he had given the command EDIT SINGLE NOLIST.

SHOW MODES shows whether RETRY or NORETRY and EDIT or MODIFY are in effect.

Note 1. Retry may be invoked whether the current mode is RETRY or NORETRY. The following table summarizes what action the user must take in order to cause WYLBUR to type out the typed or submitted command line and prompt for modification. The action the user takes depends on the current mode - RETRY or NORETRY. Retry may be invoked for commands that executed satisfactorily, not satisfactorily, and typed or partly typed but not yet submitted (that is, the user has not yet typed carriage return).

State of WYLBUR Command	Action by User	
	Mode RETRY	Mode NORETRY
1) Command submitted executed satisfactorily	<CR>	@<ATTN>
2) Command submitted did not execute satisfactorily	none needed	@<ATTN>
3) Command is being typed but has not been submitted	@<ATTN>	@<ATTN>

Note 2. Execute file commands can be retried, but the actual contents of the file will not be changed. If the command being retried was originally executed as part of the execute file, execution will continue if the retry is successful.

```
SET NAME <dsname> [<dsn options>] [KEYWORD]
SHOW NAME
CLEAR NAME
```

SET NAME allows the user to set the values of the USER, PREFIX, and/or MEMBER fields in one command. The dsname is formed in the usual way (see "dsname" in this section), except that the default PREFIX is null rather than LIB.

where:

<dsname> allows the user to set the PREFIX and/or MEMBER fields that are to be used in forming a dsname. This parameter is required and must be specified first. See "dsname" in this section.

<dsn options> The dsname option USER may be specified. See "dsn options" in this section.

KEYWORD allows jobs and data sets belonging to the saved user to be treated as if they belonged to the signed-on user.

SHOW NAME will display the current values of USER, PREFIX, and MEMBER.

CLEAR NAME will cause the name fields to be set to their signed-on values.

SET NOTIME
SET TIMEOUT

SET NOTIME prevents WYLBUR from signing off a user who has been inactive for 20 minutes. Though the command may always be used, its effect will be overridden when fewer than MINFREE lines are available; see SHOW MINFREE.

SET TIMEOUT cancels the SET NOTIME command.

SET PREFIX [[*]prefix]
SHOW PREFIX

SHOW PREFIX or SHCW NAME can be used to show the current value of PREFIX.

where:

prefix is a character string comprised of a prefix that can be used in building a dsname. See "dsname" in this section. If the prefix is omitted, it is set to null, which is also the default value at signon time.

*prefix sets the prefix to a concatenation of the old value of the prefix and the prefix string.

SET RESCAN <integer>
SHOW RESCAN

The preprocessor can be directed to rescan any text that it substitutes. This could lead to an infinite rescan if an error is made. So that this does not happen, a default limit of 5 rescans is set. SET RESCAN allows the user to change this limit. SHOW RESCAN displays the rescan limit. These commands are primarily useful in execute files. They are discussed in detail in section 5.0.

SET RETURN <line number>
SHOW RETURN

SET RETURN sets the value of the symbolic line number RETURN in the execute file to the specified line number. Note that the line number chosen is in the execute file, not in the active file. Also see "symbolic line number", "RETURN", and the command EXECUTE in this section.

SHOW RETURN displays the value of RETURN. (Actually, RETURN is just one of the values that may be specified in the command SHOW <line number>.)

SET SKIP <character>
SHOW SKIP

SET SKIP allows the user to specify a SKIP character. The character following the SKIP character will not be changed by the preprocessor. Thus, the user may enter command lines that will have an ESCAPE or a SKIP character in them after the preprocessor has finished text substitution. SHOW SKIP displays the SKIP character. SET SKIP with no cperand restores to default no skip character. These commands are primarily useful in execute files. They are fully descibed in section 5.0.

```
SET   {SLOWLIST|FASTLIST}
SHOW  {SLOWLIST|FASTLIST}
```

SET SLOWLIST prevents WYLBUR from using any tabs when listing a file at the terminal. SET FASTLIST allows WYLBUR to use tabs when listing a file if the logical tabs have been set. See the SET TABS command.

When the user first calls the access telephone number and logs on, FASTLIST is enabled. After a user sets SLOWLIST it remains in effect across individual sessions until reset or until the user hangs up the telephone.

Either SHOW FASTLIST or SLOWLIST will display which option is currently in effect.

```
SET TABS   [t1,t2,...,t16] [VERIFY|NOVERIFY]
SHOW TABS  [VERIFY|NOVERIFY]
CLEAR TABS
```

SET TABS allows the user to set logical tabs in WYLBUR so that the physical tabs may be used. WYLBUR will prompt with directions for what the user is to do. Specifying the tab positions t1,t2,...t16 is the fast way of setting tabs; a slower way of setting them is by omitting all the parameters (see section 2.3.5).

where:

t1,t2,...,t16 — are the column positions at which the logical tabs are to be set. A maximum of 16 tabs may be set. This parameter assumes that the user has first set the physical tabs on his terminal. Tabs must be specified in ascending order (e.g., 5, 10, 15 rather than 5, 15, 10).

VERIFY
NOVERIFY — NOVERIFY will cause WYLBUR to not verify that the physical tabs and logical tabs are in accordance. Verify is the default of the SHOW TABS command. NOVERIFY is the default of the SET TABS command.

If the user presses the tab key when no tabs have been set WYLBUR will issue an error message and not process the line. If the user enters more than 16 tab settings, only the first 16 are recognized; additional tabs are ignored. Tab settings are retained across terminal sessions until the user hangs up the telephone.

If the command to set tabs has an error, WYLBUR will reject the command and clear all logical tabs currently in effect, if any. For example, assume logical tab positions 5, 10, and 15 are set. The user gives the command: SET TABS 10, 20, 25, 15. The command will be rejected because the tabs are not specified in ascending order, and tab positions 5, 10, and 15 will be cleared.

Tabs are used when possible to speed online listing of the active file; see the SET SLOWLIST command.

SHOW TABS allows the user to display the current tab positions or to verify them by specifying the VERIFY parameter.

SET {TERSE|VERBOSE}

SET TERSE shortens the WYLBUR command prompt from COMMAND? to ?. Also, in WYLBUR messages that give the dsname (e.g., "dsname NOT YOURS"), TERSE mode will cause the dsname to be given in an abbreviated form rather than in the fully qualified form. For example, the fully qualified dsname Bnnnnn.FILE might be abbreviated to FILE. In this example, the abbreviated form gives no indication of user (i.e., Bnnnnn). In TERSE mode, if the data set belongs to the signed-on user and the saved user field is the same as the signed-on user, nc indication of the user is given in the dsname; if they are not the same, the dsname is prefixed by "&Bnnnnn." This rule applies to standard WYLBUR names (see section 2.7.1). Whenever a non-standard WYLBUR name is involved, the name is preceded by a $ in TERSE mode. See "dsnames" in this section for an explanation of the saved user field.

SET VERBOSE will cause the fully qualified dsname and COMMAND? prompt to be given. VERBOSE is the initial value at signon.

SET TIMEOUT see SET NOTIME

SET {UPLOW|UPPER}
SHOW CASE

SET UPLOW instructs WYLBUR to recognize upper and lower case alphabetic characters.

SET UPPER, the default option, causes WYLBUR to convert all lower case alphabetic characters typed by the user to upper case.

SHOW CASE displays which option is currently in effect.

SET VALUE <predefined variable> = <expression>
SHOW VALUE <expression>(s)

The SET VALUE command assigns an expression to a predefined variable. SHOW VALUE displays the value of the expression. These commands are primarily useful in execute files. They are described in detail in section 5.0.

SET USER [&|[&]Bnnnnnn] [KEYWORD]
SHOW USER

---

where:

Bnnnnnn — sets the default value of the user code to Bnnnnnn. The USER value will be used in forming a dsname and in the SHOW SPACE command. See "dsname" in this section. At signon time, the value of USER is set to the user code (userid) of the signed-on user. By omitting all parameters or by entering just &, the value of USER is reset to the signon value.

KEYWORD — allows jobs and data sets belonging to the saved user (see "dsname" in this section) to be treated as if they belonged to the signed-on user. The user must enter the correct keyword for the saved user. The user may then SAVE, CATALOG, RECATALOG, and UNCATALOG data sets belonging to the saved user.

SHOW USER or SHOW NAME displays the current value of USER.

```
SET VOLUME {CATALOG|<volume>}
SHOW VOLUME
SHOW VOLUMES
```

---

where:

CATALOG|<volume> specifies the system catalog or the name of the disk volume which will be used as the value of the vclume parameter in the following commands if no other volume name is specified in the command: USE, SAVE, SCRATCH, COPY FROM, EXEC FROM, SHOW DSNAMES, and SHOW DIRECTORY. CATALOG is the default vulume at Argonne.

SHOW VOLUME shows the current volume name. Note that at signon time the default volume is null for the SAVE command if you are saving a new, uncataloged dataset. Otherwise, for the USE, SCRATCH, COPY FROM, SHOW DSNAMES and SHOW DIRECTORY commands the default at Argonne is CATALOG.

SHOW VOLUMES lists all volumes that may be accessed by the user and all volumes on which data sets may be SAVEd. See "volume" in this section.

```
SHOW CATALOG [FOR {*|index pattern}] [<dsn options>]
             [UNCATALOG] [NOTYPE]
             [{FROM|EXCLUDE} string1] [{THROUGH|TO} string2]
```

SHOW CATALOG is used to display a list of the first index level of cataloged data sets or index pointers below a given index node. If the ccmmand specifies no parameters, all WYLBUR data sets and indices for the saved user immediately below the user badge number node will be listed with their type. See "dsname" in this section, cr Section 2.7.3 for an explanation of index levels and index patterns.

where:

FOR * causes the value of the current PREFIX to be used in locating the dsnames of cataloged data sets. See "dsname" in this section and Note 1 below.

FOR specifies the level or levels of index used tc search the system catalog. Only those dsnames found at the next level of index below the pattern will be listed. See "dsnames" in this section and Note 1 below.

<dsn options> The dsn options USER and SET or NOSET may be specified. See "dsn options" in this section.

NOTYPE suppresses TYPE information which is normally given for every catalog entry listed. The information consists of the type of entry and the volume name if it is a CVOL or a DSNAME.

FROM|EXCLUDE
THROUGH|TO
string1
string2

The range of the displayed list may be limited by specifying FROM and/or THROUGH. FROM string1 specifies the lower limit of the displayed list and THROUGH string2 specifies the upper limit. Either or both strings may specify * ; in this case the current value of PREFIX will be used (see SHOW PREFIX or SHOW NAME). If PREFIX is null and * is specified, the value of the string is also null. Both string1 and string2 refer to that part of the catalog entry which appears after the pattern specified by the FOR parameter or the default pattern. If EXCLUDE rather than FROM is specified, all entries except those between the specified limits are displayed. If TO is specified instead of THROUGH, then entries up to but not including string2 are displayed.

Note 1. If the FOR parameter is omitted, the saved user (see "dsname" in this section) is used to form the index pattern.

SHOW COLUMNS

SHOW COLUMNS produces a line containing column numbers. It is particularly useful for deciding in which column text begins.

SHCW COUNT

SHCW COUNT displays the number of users currently signed on the system.

SHOW DATE

SHOW DATE displays the time of day and the current date in two forms: MM/DD/YY and the Julian date

---

SHOW DIRECTORY [IN <dsname>] [<ds location>] [<dsn options>] [{FROM|EXCLUDE} {string1|#}] [{THROUGH|TO} {string2|#}] [SCRATCH] [ALL]

SHOW DIRECTORY produces a list of member names in the specified partitioned data set.

where:

IN <dsname> specifies the name of the library whose member names are to be listed. If omitted, dsname is assumed to be LIB for the current saved user value. Use SHOW NAME to display that value. See "dsname" in this section.

<ds location> tells WYLBUR the location of the specified data set. See "ds location" in this section. CATALOC is the default at Argonne.

<dsn options> The dsn options USER and SET or NOSET may be specified. See "dsn options" in this section.

FROM|EXCLUDE
THROUGH|TO
string1|#
string2|# The meaning and use of these parameters are identical to those of the SHOW DSNAMES command except that they refer to member names rather than to data set names. If "#" is specified for either string, the current value of saved member (see "dsname" in this section) is used for that string.

SCRATCH offers the user the option of scratching members of libraries after the name of the member has been listed. WYLBUR prompts SCRATCH?. A positive response ("yes", "ok", or "scratch") causes the listed member to be scratched. Any other response is interpreted as a negative response and WYLBUR lists the name of the next member.

ALL causes a hexadecimal display of all directory information to be printed in addition to the member names. If omitted, only member names are shown. For each member of a PDS that is saved by WYLBUR, an SSI field is written into the directory. The default SSI gives the Julian date on which the member was saved in the form of 0000YNNN. This field can be used to tell what members have and have not been changed recently. See the SSI parameter of the SAVE command.

SHOW DSNAMES [LIKE <dsname>] [ON <volume>] [<dsn options>]
[{FROM|EXCLUDE} {string1|*}] [{THROUGH|TO} {string2|*}]
[SKIP n] [SCRATCH] [DATED] [TYPE] [SPACE|SIZE|TRACKS]
[ALL]

SHOW DSNAMES produces a list of dsnames of data sets on a specified volume. which satisfy the criteria specified in the command.

where:

IIKE <dsname> dsname is used as a pattern against which all data sets on the specified volume are matched. Only those data set names beginning with the specified pattern are selected. If specified, this parameter must be given first. If not specified, the default pattern will be Bnnnnn, (where Bnnnnn is the saved user value).

To list the names of data sets other than WYLBUR data sets, specify the dsname in quotes or precede it by $. The quotes or $ perform the same function as in the dsname parameter of other commands. LIB is not the default in this case. See "dsname" in this section.

ON <volume> specifies the direct access volume which is to be searched for data set names. If ON is not specified and no volume is set (see SET VOLUME command), the error message "CATLG INVALID; MUST HAVE SPECIFIC VOLUME OR DSNAME," will be printed at the terminal.

<dsn options> The dsn options USER and SET or NOSET may be specified. See "dsn options" in this section.

FROM|EXCLUDE
THROUGH|TO
string1|*
string2|*

FROM and/or THROUGH limit the range of the displayed list. FROM string1 specifies the lower limit of the displayed list and THROUGH string2 specifies the upper limit. Either or both may specify * to use the current PREFIX. If PREFIX is null, and * alone is specified, the value of the string is also null; i.e., the parameter is ignored. Both string1 and string2 refer to that part of the WYLBUR dsname which appears after the pattern constructed from the LIKE operand or the default pattern. Thus, if saved user is set to B12345, the command:

SHOW DSNAMES LIKE XYZ FROM F THROUGH G

would cause a display of all data set names of the form:

B12345.XYZFxxxxx through B12345.XYZGxxxxx

If the FROM operand is omitted, the range starts from the lowest data set name found. If THROUGH is omitted, the range extends through the last data set name found. If both are omitted, all data sets for the default pattern or specified LIKE pattern are listed. If TO is used, the range extends up to but not including data sets matching the TO string.

The user may reverse this range by specifying EXCLUDE instead of FROM. In this case, all data sets except those between the specified limits are displayed.

SKIP n

can be used to skip the first n dsnames that would have normally been selected for display. Dsnames displayed are ordered by physical position in the Volume Table of Contents (VTOC) and are not alphabetical. Therefore, the SKIP parameter is probably only useful to restart an interrupted SHOW DSNAMES command after the first n dsnames.

SCRATCH

offers the user the option of scratching (erasing) any sequential external file. After the name of a data set has been listed, WYLBUR prompts SCRATCH?. A response of YES, OK, or SCRATCH causes the listed data set to be scratched. Any other response is interpreted as a negative response and WYLBUR lists the name of the next data set.

LATED

causes WYLBUR to list the date on which the data set was created and the date on which it was last accessed.

TYPE causes TYPE information to be listed for each data set. This consists of DSORG RECFM/LRECL/BLKSIZE separated by slashes as indicated. For example, EDIT format (see format parameter of the SAVE command) data sets are normally U/3520/3520 for 2314 disks and U/3156/3156 for 3330 disks. If the values are not valid, an asterisk appears instead.

SPACE SPACE means the amount of space occupied by the data set and the number of extents, if it is not 1. To show how much space has been used in the WYLBUR library, LIB, enter the command SHOW DSNAMES LIKE LIB SPACE. SPACE, SIZE, and TRACKS are all equivalent keywords.

ALL specifies that the DATED, TYPE, and SPACE information be given for each data set name.

SHCW FREE
SHOW MINFREE

SHOW FREE gives a count of the free phone lines followed by a list of their port numbers.

SHOW MINFREE gives a count of the free phone lines and lists the current value of MINFREE. MINFREE is a WYLBUR parameter set by the operator. It is the minimum number of free lines required before WYLBUR will honor a SET NOTIME command. A typical response to this command might be: 6 LINE(S) FREE, 30 MINFREE. NOTIME NOT CURRENTLY HONORED.

SHOW IDLE

SHOW IDLE gives a count of WYLBUR ports currently free and lists their port numbers. These include both dial-up lines and hard-wired lines (i.e., connection is made simply by turning terminal on). This command differs from SHOW FREE in that it gives the hard-wired ports whereas SHOW FREE does not.

SHOW JOBNO

SHOW JOBNO produces the numeric portion of the default WYLBUR jobname to be assigned to the next job submitted from a WYLBUR terminal. A typical response to this command might be: 664 - NEXT JOBNO.

SHOW {LINE|LINES} [<port-id>]

SHOW LINE lists the port number, the terminal id, the signed-on user, and other information about the specified line number if given, or about the user's terminal if no line number is specified.

SHOW LINES lists the line information for all lines signed-on or signing on beginning with the specified port-id or with the first port if no port-id is given.

where:

port-id — is the line number or identification number of the terminal, the user code, or the console operator. See "port-id" in this section for the specific forms of this parameter.

SHOW <line number> [EXEC]

This command displays the value of the specified line number in the current active or execute file.

where:

<line number> — may be any valid line number, but it will usually be one of these: FIRST, LAST, END, CURRENT or *, PREVIOUS, NEXT, RETURN. See "line number" or the entry for each symbolic line number in this section. This parameter is required and must be specified first. The active file is assumed unless EXEC is specified.

EXEC — shows the value of the specified line number in the current execute file.

SHOW MINFREE see SHOW FREE

This command is associated with the SET NOTIME command.

SHOW {MESSAGE|MSG}

This command types the WYLBUR message. No message is typed if there is no message.

SHOW PAGES

This command will print the number of pages in the current active file. To be used as an execute file, the active file must be less than or equal to 3 pages. But, a file that has been edited may be more than 3 pages because the pages are only partly full. Thus, to get an accurate page size this command should be given after a USE command and before editing the file.

SHOW SIZE [[LINES] <range>] [[OVER {n|LENGTH}] [UNDER {n|LENGTH}]| [EQUAL {n|LENGTH}]] [NOCOUNT]

SHOW SIZE displays the number of characters in each line of a specified range. It can be helpful in finding lines to ALIGN.

where:

LINES <range> is the range of lines to be examined. The default range is the entire file. If this parameter is specified as LINES <range> it may appear anywhere; however, if it is specified as <range> it must be given first. See "range" in this section.

CVER n|LENGTH
UNDER n|LENGTH
EQUAL n|LENGTH — n may be any integer from 0 to 133. LENGTH specifies that the current value of the global LENGTH (see SET LENGTH) be used. OVER will list line numbers and counts for lines longer than n or LENGTH; UNDER for lines less than n or LENGTH; and EQUAL for lines equal to n or LENGTH. In addition, the OVER and UNDER parameter can be combined in one command to process only lines that are between the two lengths specified. For example, OVER 72 UNDER 80 will list lines that are 73 to 79 characters long. The default is to list all line numbers and their character count.

NOCOUNT — causes line numbers to be printed without the character count.

SHOW {SPACE|TRACKS} [<dsn options>] [ON <volume>]

These commands give information about the amount of free space, in tracks, that remains on a specified volume or the space usages for a user.

where:

SPACE & TRACKS — are identical. The space values will be shown in tracks.

<dsn options> — The dsn option USER specifies that WYLBUR show the accounted and maximum allowed space for USER Bnnnnn. See "dsn options" in this section for details.

ON <volume> — specifies the volume name for which the amount of free space is to be indicated. See "volume" in this section.

SHOW TIME

SHOW TIME lists the date and time in hours, minutes, and seconds and the current values for elapsed time and editing time.

SHOW USERS [<port-id>]

SHOW USERS lists the users, starting at the port-id given, or if no port-id is given, lists all users.

where:

port-id — is the line number or identification number of the terminal, the user code, or the console operator. See "port-id" in this section for the specific forms of this parameter.

STATUS [jobid|badgenumber [BADGE|B]]

STATUS allows the user to inquire about the current status of any job in the system whether the job was submitted via WYLBUR or not.

where:

jobid — identifies the job by the jobname or ASP jobnumber. See "jobid" in this section.

badgenumber — is the badge number in the computer use authorization card of the job. If a 5-digit number is specified instead of 'jobid' the status is given for all jobs in the system for that badge.

BADGE or B — indicates that a badge number was given instead of an ASP job number. May be omitted for 5-digit badge numbers.

If the STATUS command is given with no operands, the response gives the status of all batch jobs which have the same badge number in the computer use authorization card as the signed-on user.

The format of the response from STATUS is:

IQ906 number,name(priority,charging-priority,region,class,M(n1,n2),S(n3,n4)) dbadge (G=group) (NET-id=netid) (HOLD=(list)) list of functions

where:

priority charging — The number priority of the job between 0 and 15.

priority — The letter priority T, H, N, L or S.

region — The region size for the job. This may be absent if the region has not yet been determined.

class - The class of the job.

M(n1,n2) - The number of jobs (n1) ahead of this job waiting to run on main and their total time estimates (n2). Only jobs that are of the same job class group are counted. Thus, if this is an Express job, only Express jobs are counted, or if this is a Batch job, only Batch jobs are counted. This only appears when the job is waiting to run on main or waiting for setup.

S(n3,n4) - The number of jobs (n3) ahead of this job waiting for setup and their total time estimates (n4). This phrase only appears when the job is waiting for setup.

d-badge - The destination from the job card (d) and the badge number from the account card (badge).

group - The origin group of the job, which may be changed through WYLBUR or TSO. This is also the default destination.

netid - The network name if the job is in a network.

list - The list of hold flags:

- CU - This class of job is not being run on main at the present time.
- F - Forms hold, the job requires special printing but no printers are available at this time.
- OP - Operator hold, the operator has held the job.
- DJ - Dependent job operator hold, the operator has held this network job.
- N - Network hold, the job is waiting for other jobs in the network to release it.
- T - Terminal hold, the user has placed this job in hold and may release it using TSO, WYLBUR, or a HASP (RJP) station.
- P - Product hold, either the job's time estimate, region size or product of time estimate and region size is too large to be run at this time.
- S(n) - Setup hold. This occurs when a program calls for data that has been automatically EXPORTed and is awaiting setup for automatic IMPORT. (n) is the number of jobs awaiting setup.

list of
functions - Lists functions through which the job progresses and their status. Some of the common ones are:
RI - Reader Interpreter, where the JCL is processed and setup requirements are determined.
MA - Main
PR - Print
PU - Punch
ACDS - ASP Created Data Sets; output datasets are created so a TSO or WYLBUR user can scan them.

Following each function will be:
blank - Job has not progressed to this function
(A) job is active in this function
(C) job has completed this function
(R) the function has been rescheduled, i.e., the function has started, but could not complete, or the job required an automatic IMPORT and is awaiting setup.

For main (MA) there is a different set of letters:
L - job is waiting for unavailable volumes or for operator to locate the volumes.
U - devices are unavailable at this time for setup, or the required volume is being used by another job.
R - a request has been issued to the operator to mount the volumes on the proper devices.
Q - job is queued for main, waiting to run.
A - job is active on main.
B - job is waiting for breakdown of setup.
C - job is complete.

SUBMIT [<range>] [NUMBERED|UNNUMBERED|INTEGER] [EXEC]
[<hold options>]

SUBMIT submits all or part of the active or execute file which may consist of one or more jobs, into the job input stream of the central computer. All JCL required to run the job must be contained in the file. The first card in the range must be a JOB card.

where:

<range> is the range of lines of the active file that is to be put into the job input stream of the batch computer. If used, this parameter must be specified first. See "range" in this section.

UNNUMBERED UNNUMBERED specifies that no line numbers be written into columns 73-80 of the file. See Note 1.

UMBERED NUMBERED specifies that the line numbers be written into character positions 73-80 of the file regardless of what is there. See Note 1.

INTEGER — has the same meaning as NUMBERED except that the line number is an 8-digit integer. See Note 1.

EXEC — puts all or part of the execute file into the job input stream.

<hold options> — allows the user to place the job and/or its output into HOLD status. See "hold options" in this section.

Note 1. If this parameter is omitted entirely, a line number is entered in all records in which columns 73/80 are blank; if those columns are not blank, they will NOT be overwritten.

SUGGEST [<range>] [{TO|ACCOUNT} Bnnnnn|xnnnnn]
[<parameters of the LIST command>]

The SUGGEST command enables the user to submit a file of suggestions to a SCIP representative or any other user. The suggestion must be contained in the active or execute file at the time the SUGGEST command is given.

where:

<range> — is the range of lines in the active file to be listed offline. The default is the entire active file. If specified, this parameter must be first.

TO|ACCOUNT Bnnnnn xnnnnn — specifies the userid (Bnnnnn or xnnnnn) to whom the suggestion is to be sent. The one character destination code used in the account field of the WYLBUR-created jobcard may be supplied (where 'x' is shown) when SUGGESTing to another user. If B is supplied, this character will default to 'F' for the First floor of the AMD (221) building. SUGGEST to B12345 will create //SUGnnnnn JOB (F12345,... SUGGEST to H12345 will create //SUGnnnnn JOB (H12345,... Any character other than 'B' preceding the badge number will be treated as a destination code.

<LIST parms> — All parameters of the LIST command, except OFFLINE, are valid in this command.

TO <port-id> message

The TO command may be used to send a message to another terminal. It should be used only when necessary as it interrupts the work of the receiver. To prevent being interrupted by someone's TO command, give the command SET NOCOMM (see SET COMM in this section).

where:

port-id — is the line number or identification number of the terminal, the user code, or the console operator. See "port-id" in this section for the specific forms of this parameter.

If the user has sent a message to another terminal with the TO ccmmand, and if while that message is pending the object terminal hangs up, or his line is killed by the operator, or the sender hits ATTN, the sender receives the message: REQUEST ABORTED.

UNCATALOG <dsname> [<dsn options>]

The UNCATALOG command causes the specified data set to be uncataloged.

where:

<dsname> — is the name of the data set to be uncataloged. If the data set does not belong to the signed-on user WYLBUR will respond "DATA SET NAME NOT YOURS" and then prompt "KEYWORD?". If the user replies with the correct keyword of the user whose data set is to be uncataloged, the command will be executed; otherwise, the command will be aborted. See "dsname" in this section.

<dsn options> — The dsn options USER and SET or NOSET may be specified. See "dsn options" in this section.

USE <dsname> [<ds location>] [<dsn options>] [<format options>] [CLEAR] [SKIP n]

USE creates a copy of an external file (a data set or PDS member on a direct access device) in the user's active file.

where:

<dsname> is the name of the data set to be read. Names of WYLBUR data sets with standard WYLBUR names may be specified in their entirety or in an abbreviated form using default or preset values. This parameter is required and must be specified first. See "dsname" in this section.

<ds location> tells WYLBUR the location of the specified data set. See "ds location" in this section. This parameter is seldom used at Argonne, since the default is the CATALOG.

<dsn options> The dsn options USER and SET or NOSET may be specified. This command is the only one that has SET rather than NOSET as the default option. See "dsn options" in this section.

<format options> allows the user to specify the desired record length and how WYLBUR line numbers are to be treated. For the specific form of these options see "format options" in this section.

CLEAR erases the current active file (if any) in order to copy the specified data set into the active file, since USE cannot be executed until the active file is empty. If there is an active file, WYLBUR will ask if it is OK to clear it in order to proceed with the USE. This dialogue may be pre-empted by specifying CLEAR.

SKIP n specifies that the first n records are to be skipped, i.e., ignored.

"volume"

The term volume refers to the name of the storage device used to hold both WYLBUR and other system data sets. The command SHOW VOLUMES can be used to obtain the names of volumes that can be used. For example, the response to this command might be:

```
USE & SAVE -- volume list
TEMPORARY -- volume list
USE -- volume list
```

USE & SAVE means that the listed volumes may be used as the volume parameter of the SAVE and USE commands. A TEMPORARY volume can be used for temporary SAVEs; data sets saved on a temporary pack are automatically scratched seven days after the day of creation. USE means that the named volumes are not valid as the volume parameter of the SAVE command. Thus, any data set on one of those volumes may be USEd at the terminal, but data sets may not be saved on them.

THIS PAGE

WAS INTENTIONALLY

LEFT BLANK

5.0 WYLBUR Preprocessor

The WYLBUR preprocessor evaluates expressions in WYLBUR commands and substitutes the results before the commands are executed. The preprocessor and some extensions to the WYLBUR command language coupled with the execute file capability yield a powerful programming language. It has been used for such varied tasks as creating and editing JCL, automatic document preparation, etc.

5.1 Overview of the Preprocessor and WYLBUR Extensions

The WYLBUR preprocessor, an expression evaluator, has 30 predeclared variables that can be set and used for string manipulation and integer or line number calculation. As an expression evaluator, the preprocessor scans each command presented to it, either as entered from the terminal or from an execute file, for a user-specified character, the ESCAPE character. An ESCAPE character signals the preprocessor to evaluate expressions and/or make text substitution. After the substitution has been made, WYLBUR executes the command.

Several extensions to support the preprocessor were made to the WYLBUR command language. These extensions include three commands that control the preprocessor (SET and SHOW ESCAPE, SET and SHOW RESCAN, and SET and SHOW SKIP), two commands that facilitate string manipulation (SET and SHOW VALUE, and the READ command), and a conditional branch command (the IF command).

The preprocessor permits simple to complex expressions in all WYLBUR commands. The extensions discussed above support the preprocessor and use expressions. Thus, to fully understand these new commands the user must is first understand what constitutes an expression.

5.2 Expression Syntax

An expression consists of a single primary, or two or more primaries connected by arithmetic operators. A primary may be a constant, either of two classes of variables, a function, or an expression. An operator may be an addition, substraction, multiplication, division, or concatenation symbol.

5.2.1 Constants

A constant is a fixed, unvarying quantity. Allowable constants are: integer, decimal, and string constants.

Integer constant: a whole number written without a decimal point with a maximum length of nine digits.

Line number constant: any WYLBUR line number is allowed (see "line number" in section 4.0 for a definition).

String constant: a character string enclosed in single or double quotes having a maximum length of 255 characters. Alphanumeric and all special characters are allowed.

In addition, self-defining hex constants may be used to facilitate hex arithmetic or to create special characters that cannot be entered directly from the terminal.

N'hex string' is an integer constant (e.g., N'FFF'=4095)

W'hex string' is a line number constant (e.g., W'FFF'=4.095)

S'hex string' is a string constant (e.g., S'C8C9='HI')

5.2.2 Variables

A variable is a quantity that may assume a succession of values. The two classes of variables are predefined variables and keyword variables.

Predefined Variables

Predefined variables consist of 30 variables:

10 N-variables (or integer variables) which are represented as N0, N1,...,N9. N-variables may be set to any integer constant.

10 W-variables (or line number variables) which are represented as W0, W1,...,W9. W-variables may be set to any line number constant.

10 S-variables (or string variables) which are represented as S0, S1,...,S9. S-variables may be set to any string constant.

All N-variables and W-variables are set to zero at the beginning of a WYLBUR session. All S-variables are set to null.

Keyword Variables

Keyword variables are a subset of keywords that may be used in many other WYLBUR commands. The meaning of the keywords when used as a variable is the same as when used in any other WYLBUR command. The set of keyword variables is listed below. References given in parentheses refer to an explanation of the keyword in section 4.0.

DELTA (see "default delta")
DATE (see SHOW DATE command)
ESCAPE (see section 5.3.1)
GROUP (see "dsname")
JOBNO (see SHOW JOBNO command)
LENGTH (see SET LENGTH command)
MEMBER (see "dsname")
PREFIX (see "dsname")
SKIP (see section 5.3.1)
TERMINAL (see SET TERMINAL command)
USER (see "dsname")
VOLUME (see "volume")

and, the symbolic line numbers (see "line number")

FIRST
LAST
END
NEXT
PREVIOUS
CURRENT or *
RETURN

NOTE: If CURRENT (or *), NEXT, and/or PREVIOUS are used in any expression and they are not set, they will be given a value of -1.000 instead of causing an exec break (see description of the EXECUTE command in section 4.0). If there is no active file, FIRST and LAST have the value 0.000. For example, assume an execute file contained this command:

```
POINT 'WORD'
```

If WORD is not in the active file, the command SHOW VALUE CURRENT would return the value -1.000; otherwise, the line number of the last line in which WORD occurs would be returned.

5.2.3 Functions

A primary may be a function. There are 3 classes of functions: string functions, encode/decode functions, and conversion functions. The conversion functions are discussed in section 5.4. When specifying a function, note that one or more blanks, commas, and/or equal signs can be used between arguments, around operators, and before and after parentheses. They are not normally required if the separation is clear without them.

String functions

The string functions are INDEX, FIND, VERIFY, SIZE, and SUBSTR. The exact form of each of these functions is given below.

INDEX (expression, expression)

> INDEX finds the starting position (column) of the string specified by the second argument within the string specified by the first argument. If the second string is not contained within the first string, the value 0 is returned. E.g., INDEX ('ARGUMENT','MENT') has the value 5.

FIND (expression, expression)

> FIND examines the two arguments specified to determine the position of the first character in the first argument which is also in the second argument. The characters in the second argument need not be in the same order as those in the first argument. If there is no match, the result is zero. For example, assuming S0 has the value 'ABCDE', then FIND (S0,'DC') has the value 3. FIND ('ABC','DEF') has the value 0.

VERIFY (expression, expression)

> VERIFY examines the two arguments to verify that each character in the first string is represented in the second string. If the first string is represented, character for character, in the second string, the value 0 is returned. The characters in the second argument need not be in the same order as those in the first argument. If the first string is not represented character for character in the second string, the value returned is the column position of the first character in the first string that is not represented in the second string. For example, assume ¬ represents a blank; then VERIFY ('¬¬¬¬¬START','¬') has the value 6. This, then, is a good way to find the first non-blank character in a string. VERIFY ('CCCCC','C') has the value 0.

SIZE (expression)

SIZE simply returns the the length of the argument. E.g., SIZE ('EXPRESSION') has the value 10.

SUBSTR (expression, expression, expression)

SUBSTR produces a substring of the first argument. The second argument specifies the starting column of the substring, and the third argument specifies the length. For example, SUBSTR ('VARIABLE=STRING',10,6) has the value STRING.

If the length (third argument) is omitted, the remainder of the string after the starting position is returned. For example, SUBSTR ('VARIABLE=STRING 1',10) has the value STRING 1.

The starting column position (second argument) and the length (third argument) need not be within the string (first argument) boundaries, but they cannot be negative nor sum to more than 256. The result will be a null string if the length operand is zero, or if the length is omitted and the starting position is beyond the string operand. For example, SUBSTR ('VARIABLE=STRING',9,0) or SUBSTR ('VARIABLE=STRING',16) returns a null string.

If the specified substring extends beyond the end of the first argument, the resulting string is blank filled on the right. For example, SUBSTR ('VARIABLE=STRING',10,8) returns the substring STRING with two blanks following the word STRING.

Encode/decode Functions

The functions DISPLAY, SHEX, NHEX, and WHEX convert values to and from hexadecimal format.

DISPLAY (expression)

DISPLAY converts the argument into its internal hexadecimal representation. If the argument is a string, the result is a string twice its length, because each character of the argument string is now represented as two hexadecimal characters. For example, DISPLAY ('$R') returns the value '5BD9'.

The result of a display function with a number as its argument will always be 8 characters long, even if leading zeros must be added. For example, DISPLAY (15) has the value 0000000F. However, this function handles decimal numbers by multiplying the number by 1000 and then converts it to hex. For example, DISPLAY (1.) returns value 000003E8 since that is the hex representation of 1.*1000 or 1000.

SHEX (expression); also, S'hex string'

SHEX is the inverse of a DISPLAY function. The argument is a string. That is, the hex digits of the string argument are converted to form EBCDIC equivalents. For example, SHEX ('C1F1') returns the value A1.

This function can be used to create any of the possible 256 character codes. For example, SHEX ('5F') will return the value ¬. Most of these 256 characters' codes are not printable on the terminal; a % will be printed instead. As an extension to this feature, decimal integer arguments are allowed. Thus, the decimal representation of 5F, 95, would also yield the value ¬.

NHEX (expression); also, N'hex string'

NHEX converts a hexadecimal string to an integer. It is the inverse of the DISPLAY function and yields an integer argument (or N-variable). The string of up to 8 hex characters is packed to form an integer.

WHEX (expression); also, W'hex string'

WHEX is the inverse of the DISPLAY function and yields a line number (or W-variable). The string of up to 8 hex characters is packed to form a line number value. E.g., NHEX ('3F8') returns the value 1.000.

5.2.4 Operators

WYLBUR expressions and primaries within an expressions may be connected by any one of these five operators:

+ indicating addition
- indicating subtraction
* indicating multiplication
/ indicating division
|| indicating string concatenation

The symbols + and - may also be used to indicate a positive or negative value rather than an operation to be performed. Exponentiation is not available.

5.2.5 Expression Evaluation and Implicit Conversion

Expression Evaluation

There are three levels of priority used to evaluate an expression containing operators: multiplication and division are performed first; addition and subtraction, secondly; and string concatenation, thirdly. Parentheses may be used to change this order of evaluation. Some examples of expression evaluation follow.

Example 1.

3+10*-2 is evaluated as:

(a) 10*-2 = -20
(b) 3-20 = -17

Example 2.

(3+10)*-2 is evaluated as:

(a) 3+10 = 13
(b) 13*-2 = -26

Example 3.

'12'||'AB' results in '12AB'

Implicit Conversion

The arithmetic operators (+,-,*,/) expect numeric arguments. If any of the arguments is not numeric (i.e., it is a string), implicit conversion will cause the string argument to be converted to a numeric value. The concatenation operator (||) expects string arguments. If any of the arguments is numeric it will be converted to a string. The following tables give the various combinations of argument types, the implicit conversion performed for the type of operation indicated, and the type of the result.

For Arithmetic Operations: + - / *

Argument Type	Implicit Conversion	Result Type
all integer	none	integer
all decimal	none	decimal
mixed: integer, decimal	integer to decimal	decimal
mixed: integer string, integer	integer string to integer	integer
mixed: decimal string, integer	decimal string to decimal, integer to decimal	decimal
mixed: decimal string, decimal	decimal string to decimal	decimal
mixed: integer string, decimal	integer string to integer to decimal	decimal
mixed: non-numeric string, integer, decimal, or numeric string	none	invalid operation

For String Concatenation: ||

Argument Type	Implicit Conversion	Result Type
all strings	none	string
all numeric strings	numbers to strings	string
mixed: string and numbers	numbers to strings	string

Here are some examples of commonly performed operations:

Example 1.

```
SHOW VALUE 5/2. 5/2 5./2. 2.*2/5 2.+2/5
   2.500 2 2.500 .800 2.000
```

Example 2.

```
SET VALUE S0='WYL00'
SET VALUE N0= 1
SHOW VALUE S0||N0
   'WYL001'
```

Example 3.

```
SHOW VALUE 'ABC'||1.2
   'ABC1.200'
```

Example 4.

```
SHOW VALUE '1'||'0.'+2
      '12.000'
```

The result '12.000' is arrived at as follows:

```
'0.'+2 --> 0.000+2 --> 0.000+2.000=2.000
'1'||2.000 --> '1'||'2.000'='12.000'
```

Example 5.

```
SHOW VALUE 3*((1||71)+29)
      600
```

The result 600 is arrived at as follows:

```
1||71 --> '1'||'71' --> '171'
'171'+29 --> 171+29 --> 200
3*200=600
```

Conversion Functions

The functions NCONVERT, SCONVERT, and WCONVERT may be used to force conversion of a string to an integer or line number, an integer to a line number or string, or a line number to an integer or a string. The format of these functions is:

```
NCONVERT (expression)
WCONVERT (expression)
SCONVERT (expression)
```

NCONVERT takes the value of the specified expression and converts it into an integer. WCONVERT converts the value of an expression into a line number. SCONVERT converts the value of an expression into a tring.

5.3 Preprocessor Commands and Command Extensions

5.3.1 Control of the Preprocessor

Before any WYLBUR command is executed, it is scanned by the preprocessor for an escape character and a skip character. If both of these characters are null, the command is checked for validity and executed. If the escape character is not null, the preprocessor scans each command for the user-specified escape character. If it finds one, it evaluates the expression that follows the escape character and/or substitutes text. The command is then checked for validity and executed. The escape character is set by the SET ESCAPE command.

The SET SKIP 'character' command gives the user more refined control over the preprocessor in general, and over the use of the ESCAPE character specifically.

The SET RESCAN integer command is not likely to be needed by users in most cases; although in unusual cases it gives the user further control over the scanning mechanism of the preprocessor.

SET and SHOW ESCAPE

The purpose of the command:

```
SET ESCAPE 'character'
```

is (1) to activate the preprocessor into scanning each command presented to WYLBUR and (2) to define what character in a given command will trigger the expression evaluation and the text substitution mechanism.

'character' may be any alphanumeric or special character. The quotes may be omitted unless it is a lower case letter or one of these: ( ) = , ; ' " blank

The command SET ESCAPE '' (null) will deactivate the preprocessor. (This is the default condition at logon time). The ESCAPE character may be displayed with the command SHOW ESCAPE.

To demonstrate how the preprocessor works in general and how the SET ESCAPE command works specifically, consider the following sequence of commands in an execute file:

```
Line Number     Command

   10.000       SET ESCAPE=&
   11.000       SET VALUE S2='COMMENT'
   12.000       SET VALUE W3=1.2
   13.000       POINT '&S2' IN &W3/LAST
```

Line number	Explanation
10.000	Activates the preprocessor and tells it to search each command for the character &.
11.000	Assigns the string COMMENT to the string variable S2.
12.000	Assigns the value of the pointer to the current line in the active file to the line number variable W3.
13.000	The value of S2 is substituted for &S2. The value of W3 is substituted for &W3. The resulting command is then executed.

Let's use line 13.000 as an example to see step by step how the substitution mechanism of the preprocessor works.

(1) If the command presented to the preprocessor comes from an active file or an execute file, rather than directly from the terminal, a copy of the command is made. It is this copy that is changed. Also, the preprocessor uses a pointer to help it keep track of its scanning within any given command line. It now has a copy of the command:

```
POINT '&S2' IN &W3/LAST
```

(2) As it scans the copy, it is stopped by the ESCAPE character & and places its pointer, P, so:

```
       P
POINT '&S2' IN &W3/LAST
```

(3) In order to substitute text, the preprocessor must determine that what follows the pointer is an expression in parentheses or an N, S, or W-variable. In this case, it finds an S-variable; thus it substitutes the value of S2 for &S2 and places the pointer such that it points to the character immediately following the substituted text like so:

```
                P
POINT 'COMMENT' IN &W3/LAST
```

If the preprocessor had not found a predefined variable or a parenthesized expression after the escape character, it would have left the escape character and would not have made any substitution.

(4) The preprocessor continues to scan the command and, again, it finds an ESCAPE character. Its pointer is now positioned thus:

```
                P
POINT 'COMMENT' IN &W3/LAST
```

(5) Again, it finds a variable following the pointer and the process described in steps 3 and 4 is repeated. However, in scanning the remainder of the command it finds no more ESCAPE characters.

(6) The copy of command after being processed by the preprocessor looks like:

```
POINT 'COMMENT' IN 1.200/LAST
```

(7) WYLBUR now executes this command. Note that the command in the exec file has not been changed and is still:

```
POINT '&S2' IN &W3/LAST
```

SET and SHOW SKIP

To allow the user to specify commands that will have an ESCAPE character in them after the preprocessor has completed text substitution, a second character, called the SKIP character, is available. The character following the SKIP character in a command is always left in the command by the preprocessor. The initial value of the SKIP character is null. The command that specifies the SKIP character is:

```
SET SKIP 'character'
```

The quotes may be omitted unless lower case letters or the following characters are used: ( ) , ; ' " = blank

The SKIP character may be turned off with the command SET SKIP ''(null). If the ESCAPE character is null, the SKIP character is not processed.

For example, assume an active file and the following sequence of commands in an exec file:

```
1.000  SET ESCAPE=%
2.000  SET SKIP=$
3.000  SET VALUE S0='COMMENT'
4.000  CH '$%S0' to '%S0'
```

First, the preprocessor will change a copy of the command in line 4.000 to:

```
CH '%S0' to 'COMMENT'
```

Then, WYLBUR will execute that copy of the command and all occurrences of the string %S0 will be changed to COMMENT in the active file.

The command SET SKIP '$' sets the SKIP character to $ if it is not already $. If the SKIP character had already been $, the preprocessor would have changed the command to SET SKIP '' and the SKIP character would be turned off. Therefore, execute files should turn off the SKIP character first and then set it in order to avoid this problem.

The current value of the SKIP character may be displayed with the command:

```
SHOW SKIP
```

SET and SHOW RESCAN

Rescan occurs only when a pair of ESCAPE characters prevents the scan pointer from moving over the substituted value. What is meant by this is best shown by an example. It is assumed that the reader has understood the earlier discussion of the ESCAPE character. Consider this sequence of commands.

```
Line Number   Command

    1.000     SET ESCAPE=%
    2.000     SET VALUE W3=.125
    3.000     SET VALUE N1=243
    4.000     SET VALUE S2='ANSWERS %N1, %W3'
    5.000     COMMENT %%S0
```

Let's follow the preprocessor as it processes the command at line 5.

(1) The preprocessor scans the command, recognizes the first % as an ESCAPE character, and places the scan pointer like so:

```
            P
COMMENT %%S2
```

(2) It looks to see if the next character after the scan pointer is also an escape character. Since it is, the preprocessor recognizes a rescan situation. This causes the pointer to remain where it is.

(3) The preprocessor then determines whether an expression in parentheses or an S-, N-, or W-variable follows the pair of escape characters. In our example, it is an S-variable; therefore, the preprocessor substitutes for %%S2 the value of S2, which is 'ANSWERS %N1, %N3', like so:

```
            P
COMMENT ANSWERS %N1; %W3
```

Note that the scan pointer remains where it was before substitution occured.

(4) The pointer P tells the preprocessor to continue scanning until the command has been completely scanned like so:

```
COMMENT ANSWERS 243;  .125
```

Note that had line 5 been the command

```
COMMENT %S2
```

The processed command would have been

```
COMMENT ANSWERS %N1; %W3
```

because after the value of S2 had been substituted for %S2 the pointer would have been placed after the substituted text.

If an error is made, an infinite rescan could occur. For example, consider this sequence of commands:

```
1.000  SET ESCAPE %
2.000  SET VALUE S1='%%S1'
3.000  COMMENT %%S1
```

During the processing of the command in line 3.000 you will see that a rescan situation is recognized by the preprocessor. Therefore, the scan pointer points like so:

```
          P
COMMENT %%S1
```

After substitution for the S-variable has been made the command looks like:

```
          P
COMMENT %%S1
```

and an infinite rescan situation has developed.

To prevent an infinite rescan, a limit of 5 rescans is set initially. The user may override this by giving the command:

```
SET RESCAN integer
```

The rescan limit may be displayed with the command:

```
SHOW RESCAN
```

Summary

The following is a summary of the discussion in this section. Please take special note of points 3 and 4 as these were not discussed earlier.

1. Substitution Rules

The action taken by the preprocessor when an escape or skip character is found is as follows:

Preprocessor Scan Pointer	Action
escape character followed by N-, W-, or S-variable	Replace N-, W-, or S-variable name by the value of the variable. Continue scanning command from the first character after the substitution.
escape character followed by expression in parentheses	Evaluate the expression and replace the expression by its value. Continue scanning from the first character after the substitution.
two escape characters, followed by N-, W-, or S-variable	Replace N-, W-, or S-variable name by the value of the variable. Continue scan at first character of the substituted string.
two escape characters followed by expression in parentheses	Evaluate the expression and replace the expression by its value. Continue scan at first character of the substituted string.
skip character	Remove skip character, then position scan pointer after the next character. I.e., ignore the character following the skip character.

2. Any substitutions or changes to a command are made each time the command is executed. If the command came from a file, the file is not changed; only a copy of the command is changed and then executed.

3. All substitutions are completed before the command is executed. No substitutions are made or evaluated dynamically as the command is executed.

4. The TERSE/VERBOSE modes of typed input and EXEC processing control whether the changed command is typed before execution. If the mode is TERSE for the source of the command, the changed command is not typed. If it is VERBOSE, the changed command is typed.

5. There is no overhead before command execution if the ESCAPE character is null. If there is an ESCAPE character, a translate and test is done on the command line to locate and process any ESCAPE characters and skip characters. If the text after an ESCAPE character is not an S-, N-, or W-variable, or an expression in parentheses, there will be no substitution and the text (including the ESCAPE character) will not be changed.

5.3.2 WYLBUR Command Extensions

SET and SHOW VALUE

The SET VALUE command is used to assign the value of an expression to an S-, N-, or W-variable. The SHOW VALUE command may be used to display the current value of one or more expressions. The format of the two commands is:

```
SET VALUE <predefined variable> = <expression>
SHOW VALUE <expression> [<expression>,...,<expression>]
```

Predefined variables and expressions are discussed in section 5.2. The limit on the number of expressions that may be specified in one SHOW VALUE command is the length of the line (i.e., 133 characters).

With respect to S-variables, note that any string constant must be enclosed in either double or single quotes (see example 5 below).

The following examples illustrate the kinds of expressions that can be used in these commands.

Example 1.

```
SET VALUE W5=471.09 (line number constant)
SET VALUE W0=LAST (keyword variable)
      SHOW VALUE W0 LAST W5
      51.000 51.000 471.090
```

Example 2.

```
SET VALUE N0=10 (integer constant)
      SHOW VALUE N0
      10
SET VALUE N0=N0/3 (predefined variable+integer constant)
      SHOW VALUE N0
      3
```

Example 3.

```
SET VALUE W0=2.3 (line number or decimal constant)
      SHOW VALUE W0
      2.300
```

Example 4.

```
SET VALUE S0='$PAGE PREFIX="UN45-" EJECT=1'    (string
      SHOW VALUE S0                                 constant)
      '$PAGE PREFIX="UN45-" EJECT=1'
SET VALUE N0=INDEX (S0,'PR')     (function INDEX)
      SHOW VALUE N0
      7
SET VALUE S1=SUBSTR (S0,N0)     (function SUBSTR)
      SHOW VALUE S1
      'PREFIX="UN45-" EJECT=1'
SET VALUE S1=SUBSTR (S1,INDEX (S1,'"')+1)
      SHOW VALUE S1
      'UN45-" EJECT=1'
```

Example 5.

```
SET VALUE S0='DOC'||'TOR'    (two string constants
      SHOW VALUE S0                    concatenated)
      'DOCTOR'
```

Note that string constants, as in example 4, must be enclosed in either single or double quotes. That is, the command SET VALUE S0=DOCTOR would result in the message DOCTOR INVALID.

READ Command

The READ command is designed to be used in execute files to obtain commands or values of variables from the active file, the execute file, or from the user at the terminal. The complete format of the command is:

```
READ [STRING <S-variable>|VALUE <list of S-,N-,and/or W-variables>]
     [EXEC] [[USING] <line number> [COLUMNS m[/n] [DELETE]]
     [PROMPT 'string']
```

Refer to the description of the READ command in section 4.0 for all parameters except STRING and VALUE; these will be explained below.

READ STRING

READ STRING allows the user to assign a string from the active file, the execute file, or the terminal to the specified S-variable. The entire line, including any single or double quotes, will be assigned to the string. That is, the string need not be enclosed in quotes. Please keep in mind, however, that the length of the line is not necessarily 133 characters long, regardless of the source. Rather, the end of the string is defined as the last non-blank character.

Some examples of this command follow. The command:

```
READ STRING S0
```

will cause the default prompt ENTER? to be printed at the terminal. The user is requested to enter the string. Thus, if the user responds to the prompt with: USE MYSTUFF ON WYL003, that string will be assigned to S0.

The command:

```
READ STRING S9 USING *
```

reads the string that will be assigned to S9 from the active file at the line pointed to by the CURRENT line pointer. Thus, if the CURRENT line contains COMMENT in columns 3/9 the string COMMENT preceded by two blanks will be assigned to S9. If the user wants a string of a specific column length, he must use the COLUMN parameter. Using the same example as above, he might say:

```
READ STRING S9 USING * COLUMNS 1/50
```

S9 would contain the string COMMENT preceded by 2 and followed by 41 blanks.

Reading a line from the execute file is like reading from the active file with two exceptions: the EXEC parameter must be given and the DELETE parameter is not valid. For example:

```
READ STRING S0 USING * COLUMNS 1/50 EXEC
```

would assign to S0 the string contained in columns 1/50 of the line that is pointed to by the CURRENT line pointer in the exec file.

READ VALUE

READ VALUE allows data to be assigned to one to eight S-, N-, and/or W-variables. The data may be any valid expression(s) (see section 5.1). The expression(s) is first evaluated and then assigned to the variable(s). Strings, however, must be enclosed in quotes. The quotes are stripped and any pairs of quotes within the string will become single quotes. If the input does not contain as many expressions as variables given in the command, the remaining S-variables are set null and the remaining N- or W-variables are set to zero. Some examples follow.

Example 1.

```
SET VALUE S0='JOHN"S ADDRESS IS '
SET VALUE S9='COMMENT'
SET VALUE W9=392
SET VALUE N1=1
READ VALUE S0, N0, W1
ENTER?          (default prompt in response to READ VALUE)
S0||'392 PALM DRIVE' SIZE(S9) W9+N1
SHOW VALUE S0, N0, W1
'JOHN"S ADDRESS IS 392 PALM DRIVE' 7 393.000
```

Example 2.

Assuming line 393.001 in the active file contained the following:

```
523192 'Arthur Dunsley' 793.00 59.35
```

and this command were given:

```
READ VALUE N5, S3, W3, W2, N4, S0 USING 393.001
```

then the values of the variables would be as follows:

```
N5=523192, S3='Arthur Dunsley', W3=793.000, W2=59.350,
N4=0, S0='' (null)
```

Note that W-variables (line number variables) are always displayed with a three-digit decimal fraction, and that N4 and S0 are zero and null respectively because the READ VALUE command contains more variables than there are values in the line from the active file.

Any needed conversion of an expression to the type of the variable (i.e., integer, decimal, or string) is performed automatically. For example:

```
READ VALUE N1
ENTER?
10./3.
SHOW VALUE N1
3

SET  CURRENT=1.5
READ VALUE W3
ENTER?
SUBSTR (CURRENT,3,10)
SHOW VALUE W3
500.000
```

If you don't see why the value of W3 is 500.000, here are some hints. SUBSTR(CURRENT,3,10) could be read as: Give me a substring of 1.500 (i.e., CURRENT) beginning at position 3 (i.e., 5) having a length of 10. Since the substring has only 3 characters rather than 10, the substring is 500. But, since this value is to be a line number (i.e., W3) rather than an integer, 500 is converted to the line number 500.000.

IF COMMAND

The IF command evaluates a logical relationship. If the result is true, the command following the relationship is executed; if it is false, the command is skipped without inspection. The format of the IF command is:

IF (<expression> <relop> <expression>) WYLBUR command

Where expression is defined in section 5.1 and relop may be one of the following relational operators:

```
EQ  equal
NE  not equal
LT  less than
LE  less than or equal
GT  greater than
GE  greater than or equal
```

The symbolic line numbers CURRENT, NEXT, and/or PREVIOUS have the value -1.000 if they are not defined. For example:

```
POINT '&S0' (1) IN FIRST/LAST NOLIST
IF (CURRENT LT 0) EXEC 10
SET VALUE W0=CURRENT
```

Line 10 will be executed if the string specified by the value of S0 does not exist in lines FIRST through LAST; otherwise, N0 will contain the line number.

When an IF command is found, the preprocessor makes any necessary substitutions, evaluates the expressions, and then compares the resulting expressions. Consider this example:

```
IF ((9/3.*2) GT 5) EXEC 10
```

In this IF command no substitutions are necessary, but the right-hand expression must be evaluated before the two resulting expressions can be compared. Thus, after evaluation a comparison of 6.000 GT 5 must be made. However, though both expressions are numeric, they are not alike in type. Therefore, the integer is first converted tc a line number. A conversion would also be necessary if one expression were numeric, the other a string. The following table summarizes what conversion is done if the two expressions are not of the same type and whether the comparison performed is logical or algebraic.

Expression Type	Conversion	Compare
both integer	none	algebraic
both decimal	none	algebraic
mixed: integer, decimal	integer to decimal	algebraic
both string	shorter string padded with blanks to equal length of longer string	logical
mixed: string, integer or decimal	integer or line number converted to string, which is then padded if necessary (see both string)	logical

Strings are compared according to the list below. This gives all characters on the 2741 keyboard from the least value to the greatest.

```
Blank ¢ . < ( + | & ! $ * ) ; ¬ - / , % _ > ? : #   ' = "
a b c d e f g h i j k l m n o p q r s t u v w x y z
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
0 1 2 3 4 5 6 7 8 9
```

Here are some examples of the IF command.

Example 1.

```
IF ((2*3.5/2) GT 1) EXEC 5
```

The result of the first expression is 3.500. The types of the two expressions do not match; thus, the value 1 is converted to 1.000. The result is true.

Example 2.

```
IF ('ART' GT 'BETTY') EXEC 10
```

The string 'ART' is padded with blanks to equal the length of the string 'BETTY'. A logical compare of 'ART ' GT 'BETTY' is false.

Example 3.

```
SET VALUE S0='100'
SET CURRENT=50
IF (S0 GT CURRENT) EXEC 5
```

The result is false. Let's see why. After substitution the IF command looks like this: IF ('100' GT 50.000) EXEC 5. (Remember CURRENT is a line number.) The value 50.000 is converted to the string '50.000'. To obtain two strings of equal length, '100' is padded with 3 blanks. Thus, a logical compare of '100 ' GT '50.000' yields a result of false. Note: WCONVERT(S0) could have been used to obtain an algebraic line number compare that would have been true.

5.4 Preprocessor Execfile Examples

This section presents some examples of EXEC files that use the preprocessor and command extensions discussed in this previous section. Following each example will be some comments on the commands used in each example.

Example 1.

```
1.000   SET ESCAPE &
2.000   SET VALUE N0=1
3.000   SHOW DSN ON WYL00&N0
4.000   SET VALUE N0=N0+1
5.000   IF (N0 EQ 10) EXEC 7
6.000   EXEC 3
7.000   SHOW SPACE
8.000   SET ESCAPE ''
9.000   CLR EXEC
```

Comments on Example 1.

Line 1 sets the escape character to &.

Line 2 sets the value of N0 to the integer constant 1.

Line 3. The preprocessor will substitute the value of N0 for &N0 just before this command is executed.

Line 4 increments the value of N0 by 1.

Line 5. If N0 has the value 10 line 7 will be executed; otherwise line 6 is executed.

Line 6 causes an unconditional branch to line 3.

Line 8 sets the escape character to null.

Example 2. WYL.SF.PUB.EXECFILE(CHNGESTR)

```
 1  ; SEE WYLBUR MANUAL FOR INFO ABOUT
 2  ; USING SOME OF THE FEATURES IN THIS EXEC FILE.
 3  COMMENT CHANGE STRING S1 TO S2 IN LINE W0
 4  SET EXEC VERBOSE LOG; THIS LETS YOU SEE WHAT'S HAPPENING
 5  SET ESCAPE ''; TURN OFF ESCAPE CHARACTER
 6  READ VALUE S1,S2,W0
 7  IF (W0 EQ 0) EXEC 6 PAUSE
 8  SET ESCAPE %; TURN ESCAPE CHARACTER ON TO SUBSTITUTE
 9  CH '%S1' TO '%S2' IN %W0
10  EXEC 5
```

Comments on Example 2.

Lines 1 and 2 are comments that will not be printed at the terminal in NOLOG mode.

Line 3. The message following COMMENT will be printed at the terminal.

Line 6. WYLBUR will prompt with the default prompt ENTER? The user must then respond with values to be assigned to S1, S2, and W0 in that order. Strings must be enclosed in quotes.

Line 7. If the user did not enter a line number for W0, the value of W0 will be zero. In that case, the test will be true and the command EXEC 6 PAUSE will be executed.

Line 9. If the user had entered 'teh' 'the' '111.5', the preprocessor would change this command to CH 'teh' TO 'the' IN 111.5.

Example 3. WYL.SF.PUB.LIB(FINDCOL)

WYL.SF.PUB.LIB(FINDCOL)

```
 1  SET EXEC NOLOG TERSE
 2  ; SEE WYLBUR MANUAL FOR INFO ABOUT USING
 3  ; SOME OF THE FEATURES IN THIS EXEC FILE.
 4  COMMENT FIND COLUMNS OF A STRING IN A SPECIFIED LINE
 5  COMMENT WHERE S0 IS THE STRING, W0 THE LINE
 6  SET ESCAPE; TURN OFF ESCAPE CHAR.
 7  READ VALUE S0,W0
 8  IF (S0 EQ '') EXEC CLEAR; STOP IF NOTHING ENTERED
 9  SET ESCAPE %; SET THE ESCAPE CHAR TO DO SUBSTITUTIONS
10  POINT %W0 NOLIST; SEE IF LINE W0 EXISTS
11  IF (* LT 0) EXEC 19; NO LINE IF * WAS NOT SET
12  READ STRING S1 USING *
13  SET VALUE N1 INDEX(S1,S0)
14  IF (N1 EQ 0) EXEC *+3
15  SHO VALUE N1 N1+SIZE(S0)-1
16  EXEC 6
17  COMMENT '%S0' NOT IN LINE %W0
18  EXEC 6
19  COMMENT LINE %W0 NOT IN FILE
20  EXEC 6
```

Comments on Example 3.

Lines 2-5. Lines 2 and 3 are comments that will not be printed at the terminal. Lines 3 and 4 are comments that will be typed at the terminal.

Line 7. The user will be prompted for values of S0, and W0 with ENTER? The string must be enclosed in quotes.

Line 8. If the user does not enter a string, S0 will be set to NULL, represented by ''.

Line 9. Assuming the user had entered 11.5 for the value of W0, this command would become POINT 11.5 NOLIST after preprocessing.

Line 11. The POINT command sets the CURRENT (*) line pointer to 11.5 if that line exists; otherwise, the value of * will be set to -1.000 and line 19 will be executed.

Line 12. The line in the active file pointed to by the CURRENT line pointer will be assigned to the S-variable, S1. E.g., S1= "FORMAT('1'2E20.8)".

Line 13. W1 will be assigned the starting position of string S0 or 0 if S0 is not found. Assume S1 is the string FORMAT ('1'2E20.8) preceded by 7 blanks and S0 is the string FORMAT. The value of N1 would be 8.

Line 14. EXEC *+3 means execute the line in the EXEC file pointed to by the CURRENT line pointer (i.e.,14) + 3 (i.e.,17). Also see comments on line 11.

Line 15. Assuming N1=8 and S0=FORMAT, then N1+SIZE(S0)-1=13, then W1=8+25-1=32.

Line 16. An unconditional branch to line 6.

THIS PAGE

WAS INTENTIONALLY

LEFT BLANK

Appendix A. Short Forms of WYLBUR Command Words

Most words used in WYLBUR can be abbreviated by typing three or more characters of the word. Thus, UNC, UNCA, UNCAT, UNCATL, and UNCATLG are all recognized by WYLBUR as the keyword UNCATLG. All characters typed (up to the first eight) are checked so that UNCATLL or UNCATLGG would not be recognized as the keyword UNCATLG. In addition, some words in WYLBUR commands have short forms or alternate forms that do not obey the general abbreviation rule. The following list gives all the words used in WYLBUR commands that have alternate forms. Those words marked with an asterisk can not be abbreviated. If a command is not listed, the general abbreviation rule applies.

WORD		ALTERNATE FORM
BACK	*	none
CHANGE		C
CLEAR		CLR
COLLECT		CO
COLUMNS		COLS
COUNT	*	none
CURRENT	*	none
DELETE		D
DDNAME		DD
FIRST		F
LAST		L
LINE	*	none
LINES	*	LINS
LIST		L
LOGOFF	*	SIGNOFF,LOGOUT
LOGON	*	SIGNON,LOGIN
MESSAGE		MSG
NOBACK		5 char. or more
NOCOMM		5 char. or more
NOLIST		N
NONL	*	none
NOTIME		5 char. or more
OPERATOR	*	OPER,OPR
POINT		P
PRTY	*	none
READER		*
SAVE		S
STATUS		ST
THROUGH		THRU
VOLUMES	*	VOLS

You may use any of the valid forms of a word in a command. There are no periods after abbreviations or short forms.