Menu

Contents

Indexes

Search

Chapter 25E

Worksheet Buffer

Contents

Worksheet Buffer

Introduction

The Worksheet Buffer is a method for Macro-PLUS users to enable access to CODE V output in a rapid and easy manner, complete with tools for searching, extracting, sorting, etc. Commands have also been implemented for direct access to the data in the buffers through the UGR option for plotting (see "Plotting from the Worksheet Buffer" on page 24-70 and "Description of Output" on page 24-73).

The Worksheet Buffer collects CODE V output into a RAM-memory based worksheet buffer. The output is split (via a default parser) into cells that can be of type string or number. Once the data are collected, the output in the worksheet buffer can be searched, listed, sorted, copied to other worksheets, etc. An interface to Macro-PLUS allows you to access and manipulate worksheet data.

The terms "worksheet" and "buffer" can be used interchangeably; they refer to the same collection of data. In this section, both terms are used, and hopefully no confusion is caused.

Usage of the Worksheet Buffer

The collection of CODE V output into the Worksheet Buffer is controlled by the **BUF** command. **BUF Yes** will start data collection into the buffer, **BUF No** will stop the data collection. If **BUF Yes** is given to continue data collection, the collection starts at the end of the buffer; existing data are never overwritten by new data (although old data can "scroll off" the top of the collection buffer).

When output collection is turned on (**BUF Yes**), all CODE V output that would normally be displayed will also be redirected to worksheet buffer B0. Data collection in buffer B0 is independent of the OUT setting. Thus, if OUT NO and **BUF Yes** were both specified, output would still go to the worksheet B0 even though the output is not seen on your display. Output to buffer B0 is affected, however, by the VER and ECH commands.

As each line of CODE V output is collected in B0, a default parsing algorithm is used to split the output line into distinct cells (for more information on how this is done see"Default Parsing" on page 25E-6). Once the lines are in worksheet B0, they can be accessed via Macro-PLUS database items as either strings or numbers or copied to other buffers for manipulation.

Note that buffer B0 is mainly used to collect output from CODE V. B0 is designated read-only and thus you cannot directly copy or place data into it. To manipulate or reformat data from B0, it must first be copied to another buffer (B1..n).

BUF is an immediate command; it can be given anywhere in CODE V, in the LDM or in an option, and can be given in macros. All related Worksheet Buffer commands (all start with **BUF**) are also immediate commands and can be given anywhere in CODE V.

The buffer command **BUF ?** can be used to query the status of all buffers. To list the contents of any buffer, use **BUF LIS**. Note that if data collection is enabled (**BUF Y**), then the output of the **BUF ?** or **BUF LIS** command is also collected in B0.

Description

This section gives a complete description of the Worksheet Buffer, how it is used, how it collects and parses the data, and examples of buffer usage.

Basic Concepts

To help clarify the concept and usage of the Worksheet Buffer, a description of the terminology used is given here.

Worksheets, Buffers – A worksheet or buffer in CODE V is a place where data are stored. It is composed of a collection of rows, each of which contain 0 or more cells (or columns) inside the row. The terms "worksheet" and "buffer" are used interchangeably. The B qualifier is used to specify which worksheet to use for a given command or database item. Any number of buffers can be defined; you are only limited by the amount of dynamic storage available. CODE V uses a special read-only buffer (B0) for data collection. Buffers B1, B2, B3,... are all available to you for any read/write activities.

Row – A row or line contains a horizontal collections of cells. A row may have no cells or columns associated with it (an empty row). It is important to note that not all rows have the same number of cells (columns). The I qualifier is used to specify which row to use for a given command or database item.

Column – A vertical collection of cells. The J qualifier is used to specify which column to use for a given command or database item.

Line – A string of text created by concatenating the string versions of all cells in a row together, separated by a single blank.

Cell – The smallest unit inside a worksheet. Each cell contains:

1. **Address**. This is specified using buffer, row, and column qualifiers (B, I, and J).
2. **Type**. There are 3 cell types as follows:
   a. "NUM" – Numeric data
   b. "STR" – String data
   c. "UNKNOWN"– Unknown type or cell does not exist.
3. **Value**. A real number or a string.
4. **Format**. How the cell contents are displayed on output.

Current Buffer – A current buffer number is kept internal to CODE V. You can change the current buffer by moving to another buffer. This can only be changed through use of the BUF MOV command. Most commands have an optional buffer argument and when unspecified, the current buffer is assumed.

Current Row/Column – Each buffer also has a current row and column position. As with the current buffer, many commands and database items will pick up the current row and/or column if none was specified. The current cell location can only be changed through the use of the BUF MOV or BUF FND commands. The index "c" indicates "current", i.e., Ic is current row and Jc is current column.

Addressing

Most Worksheet Buffer commands and database items take a cell or cell range as an argument. A range of cells in a given worksheet may be specified using row and column ranges. Rows are numbered starting with 1 at the top and increasing down. Columns are numbered starting with 1 at the left and increasing to the right.

The following syntax is used to specify rows, columns, and their associated ranges:

Bn	–	Buffer n
In	–	Row n
In..m	–	Rows n through m
Jn	–	Column n
Jn..m	–	Columns n through m

When using the I and J qualifier in particular, the following special characters can be used in combination:

A	–	All rows or columns, equivalent to the range 1..L
C	–	The current row or column
L	–	The last row or column.

Both C and L allow relative addressing when using a constant offset. For example, Ic+1 is one row below the current row, Jl+1 is one column to the right of the last one.

JL has a meaning that fits the command in question. For any BUF command in which a single I qualifier is given, JL will refer to the last cell in row I. If an I range is specified, JL refers to the largest of all columns within the I range. If no I qualifier is given, then JL depends on the default for I for the command given. If I defaults to Ic then JL refers to the last cell in row Ic. If I defaults to I1..L, JL refers to the largest column in the buffer. This may seem a little confusing, but the purpose is to maximize convenience.

Read-only Worksheet B0

Output from CODE V is collected into worksheet B0. When buffer collection is enabled, each line of CODE V output is appended to the end of B0. No other worksheet can collect output other than B0. Additionally, worksheet B0 is designated read-only. This means you cannot directly store, change, or copy cell data in B0. B0 can be deleted entirely, however.

As each input text line enters B0, two things happen:

1. The text line is stored.

2. The text line is parsed into cells.

Refer to “Default Parsing” on page 25E-6 to see how CODE V breaks a line of text into cells. Since the entire text line is saved in B0, you have the option of reparsing or changing the distribution of text into cells. If the default parsing algorithm did not split the text into cells in a desired way, you can use the BUF PRS command to force text into the desired cells.

B0 is the only worksheet with memory limits. This is done to prevent unexpected dynamic memory request failures. These limits are expressed in lines of collected output, controlled with BUF LEN (default is 10,000). Once B0 has collected its limit of output lines, any additional lines are scrolled off the top and new lines are added to the bottom. Thus, the most recent lines are kept while the oldest lines are lost. It is important to note that as this scrolling occurs, the line (row) numbers continue to increment. The following example illustrates this point:

Worksheet B0 has a limit of 10000 rows. There are currently 9995 lines in B0.
The following table shows the status of B0 before and after 10 lines are appended to B0:

	Before	After
First Row	1	5
Last Row	9995	10005
Number of Rows	9995	10000

Read/Write Worksheets B1..n

You may create as many additional worksheets as needed. The only limit is on the amount of available dynamic memory. These worksheets can be modified at any time. In addition to reading data out of cells, these worksheets also support copying and putting data directly into cells. Worksheets are created automatically by issuing a BUF MOV command to a previously undefined buffer. In addition to the read/write capability, buffers B1..n differ from buffer B0 in two other distinct ways:

1. Buffers B1..n do not have any size limitation. Thus B1..n buffers do not scroll and do not "lose data."

2. Only cell data is stored for B1..n. A text line is not kept in addition to cells. Consequently buffers B1..n cannot be reparsed.

Default Parsing

As CODE V output is captured in worksheet B0, each incoming line is separated into cells. Initial leading white space is ignored. Text is automatically split into cells whenever two or more spaces or a single tab are found. In this case the spaces or tabs aren't actually included in the cell contents, they just serve to indicate cell boundaries.

After automatic cell boundaries have been applied, individual words may be broken up into cells further. A word is defined as a set of text with a single space on either side. If a word looks like a number, it is made a distinct cell. Otherwise words are strung together to form a single cell of type STR.

Default Parsing Example

```
CODE V> VER YES
CODE V> ECH NO
CODE V> BUF
CODE V> WRI "The diameter is 25.4 centimeters"
CODE V> WRI "The diameter is 25.4, centimeters"
CODE V> BUF NO
CODE V> BUF LIS B0

1: CODE V> wri "The diameter is 25.4 centimeters"
2:  The diameter is 25.4 centimeters
3: CODE V> wri "The diameter is 25.4, centimeters"
4:  The diameter is 25.4, centimeters
5: CODE V> buf no

CODE V> EVA (BUF.COL I2)    ! Number of columns in row I2
(BUF.COL I2) = 3            ! Cell 1 = "The diameter is"    (STR)
                            ! Cell 2 = 25.4                 (NUM)
                            ! Cell 3 = "centimeters"        (STR)

CODE V> EVA (BUF.COL I4)    ! Number of columns in row I4
(BUF.COL I4) = 1            ! Cell 1 = "The diameter is
                            !  25.4, centimeters"           (STR)
```

Notice that the word '25.4,' in row I4 cannot be converted into a number due the trailing comma. Thus, all of row I4 is treated as a single cell of type STR.

Matching Expressions

Matching expressions provide a powerful way to find data items in a worksheet. Matching expressions apply to both the BUF FND and BUF LIS commands. Matching is NEVER case sensitive.

The most simple and most commonly used matching expressions are ordinary text strings, composed of text, numbers, and most other punctuation symbols (exceptions follow). These expressions will match cells in the worksheet if a single cell matches the user-entered string exactly. For example, if you type:

```
BUF FND "THI"          or          BUF FND THI
```

Any string cell with the substring THI, somewhere, will match. For example:

	j1	j2
i1	"CUY = "	3.4
i2	"THI = "	1.9

After BUF FND "THI": `ic = 2, jc = 1.`

It is important to note the relation of the entered expression with that of the cell to be matched. If a string is entered, (i.e. text with quotes around it) ALL cells in the worksheet, regardless of cell type, can be matched (num cells are converted to a string representation). If a number is entered (number without quotes), just the cells of type NUM can be matched. This can best be illustrated in the following example:

	j1	j2
i1	"DLY 2.00v"	3.65
i2	"DLX 3.65v"	2

The cells at i1 j1 and i2 j1 are both of type string. The cells at i1 j2 and i2 j2 are both of type NUM. Note which lines are printed in response to the following BUF LIS commands:

Command	Rows Listed	Match Condition
BUF LIS i1..2 "2"	i1 and i2	All cells with substring "2"
BUF LIS i1..2 2	i2	All NUM cells with value = 2

Special Characters

More complex matching expressions can be generated for matching cells. This is where the real power behind matching expressions lies. The functions allowed for complex matching are very similar to those for the UNIX grep command or in the UNIX vi editor. The following special characters are defined for matching expressions:

Brief Explanation of Special Characters

Character		Explanation
?	–	Matches 1 of any character.
[char-class]	–	Matches a single character in char-class, where char-class is a list of characters or character ranges, for example [ak-px] matches any of the characters a, k, l, m, n, o, p, or x.
[~char-class]	–	Matches a single character NOT in char-class.
*	–	Matches 0 or more of the preceding character, character list, or word list.
@	–	Matches 1 or more of the preceding character, character list, or word list.

Character		Explanation
{expr-list}	–	Matches any expression in expr-list, for example {IN,CM,MM} matches IN, CM or MM.
<	–	Matches the following text only if a space, tab, or cell boundary precedes the text.
>	–	Matches the preceding text only if a space, tab, or cell boundary follows the text. Use left and right angle brackets to do word searching, where the word appears in between the angle brackets (e.g., <THI> matches "THI" and "the thi = 3.2" but not "thickness").
^	–	If found as the first character in the expression, it anchors the match to the beginning of a cell.
$	–	If found as the last character in the expression, it anchors the match to the end of a cell.
\c	–	Matches character c literally, that is \? matches a "?." Also called "escaping" character c.

Detailed Explanation of Special Characters

Character		Explanation
?	–	This is used to match any single character. Use it when you do not care what character appears in the particular string position. For instance "a?c" will match all the following: "abc," "aac," "a4c," "a+c."
[ ]	–	Defines a character class. This is a set of all allowed characters for the next character in the string to be matched. For example, [abc] means that if the match is to be successful, the next character must be either a, b, or c. Ranges can also be used in character classes. This helps shorten the amount of characters to be listed in a character class. For example, [0-9] means the next character must be a number. [a-z0-9] means the next character must be either a number or a letter. Note that if the dash character, '-', is used as the first character after the left bracket it is treated as a character belonging to the class and not as a range operator. Additionally, the backslash character '\', is treated as a character belonging to the class and not as the escape operator.

Examples:

Expression	Matches	But Not
"th[ia][st]"	"this"	"th"
	"that"	"thi"
	"thas"	"thia"
	"thit"	

[~]	–	Defines a character class as above, except that in order for the match to be successful, the next character must NOT be in the character class. For example, "[~+e0-9]" means the next character must NOT be a number, plus sign, or letter e. Again, if the dash character immediately follows the '~' character, the dash is treated as belonging to the character class and not as a range operator.

Character	Explanation

* – Matches 0 or more of the preceding character, character class, or expression list. So for example, “aa*” matches “a,” “aa,” “aaa,” ... Note that “a*” would match any non- 'UNKNOWN' cell, since each cell with data has 0 or more a's in it.

Example:

Find all cells with a twelve somewhere before a twenty-one. There must be a period in between these two numbers somewhere. In addition, any number of digits may also be in between the two numbers:

Examples:

Expression	Matches	But Not	
“12[0-9]*.[0-9]*21”	35123.3421	132.421	(3 unmatched)
	12.121	12a.21	(a unmatched)
	121212.212121	1221	(. unmatched)

Use “?*” to match everything.

@ – Matches 1 or more of the preceding character, character-class, or expression list. This works very similar to the star operator and in fact can be composed from the * operator, since:

{expr}@ = {expr}{expr}*

However, in many cases the @ is much more convenient to use, especially when the expressions get long.

Example:

Find all cells with a surface range 3..5 or cells with some number followed by a 'v' character.

Expression	Matches	But Not
“{S3..5,[0-9]@.[0-9]@v}”	“DSX S3..5”	“DSX S5..3”
	“0.0200000v”	“.003v”

The last example failed due to missing digit before '.'

{ } – Matches a list of expressions separated by commas. The comma in essence acts as an OR operator, so that “{expr1,expr2,expr3}” would match either expr1 or expr2 or expr3. The expressions in between commas can be literal characters, character classes, or more word lists. The <,>,*, and @ operators are also valid within { }.

Use { } instead of parenthesis to group expressions together.

Example:

Expression	Matches	But Not
{AUT, {3.2}@}	AUTO	3.AU
	3.23.2	

Character **Explanation**

< – Matches the expression that follows the '<' only if that text is preceded by a space, tab, or is the beginning of a cell. The purpose of this operator is to match the beginning of words. For example, a search on the string "THI" reveals cells such as "thi," "John Doe the third," "something," "nearby foothills." Using the left angle bracket however, will match the text only if it starts a word.

Example:

Expression	**Matches**	**But Not**
"<thi"	"thi"	"something"
	"John Doe the third"	"nearby foothills"

> – Matches the expression that precedes the '>' only if that text is followed by a space, tab, or is the end of a cell. The purpose of this operator is to match the end of words. It works analogous to '<'.

Example:

Expression	**Matches**	**But Not**
"on>"	"distribution"	"dimensions"
"<in>"	"distribution in X"	"original optical axis"
"<[a-z]>"	"I will"	"I'll see you later"

^ – Matches the following expression if the text starts at the beginning of the cell. The '^' operator must be the first character in the expression if it is used. Otherwise it is treated as a normal character.

Example:

Expression	**Matches**	**But Not**
"^start"	"starting position"	"a starting position"

$ – Matches the previous expression if the text finishes at the end of a cell. The '$' operator must be the last character in the expression if it is used. Otherwise it is treated as a normal character.

Example:

Expression	**Matches**	**But Not**
\"end$"	"it's a trend"	"it's a trend today"
"^[~-e]@$"	"a string"	"1.85e2" (string contains e)

Character	Explanation
\ –	Matches the next character literally. The backslash is used to take away the special effect of special characters. This allows you to search for these characters as ordinary text. For example you may want to search for the string, “>9<” (i.e., right angle bracket, nine, and left angle bracket). Normally the angle brackets have special meaning. However you can search for the angle brackets by “escaping” them using the backslash. In the previous example this could be done by searching on the expression: “\>9\<” Note that both the angle brackets have been preceded by the backslash indicating that those characters should be taken literally. The backslash operator is valid everywhere except within square brackets, where it is taken literally. To search for the backslash character itself, use two backslashes (e.g. “\\”) A table is provided below showing what characters have to be escaped and when they need to be escaped.

What Characters Need to Be Escaped and When

'$'	–	Only at the end of a line (meaning BUF FND “AB$4” is ok to match AB$4 but BUF FND “AB4\$” is needed to match AB4$)
','	–	Inside { } (meaning BUF FND “A,B” is ok to match 'A,B', but BUF FND “{ab\,c,de}” is needed to match strings that have either 'ab,' or 'de')
'^'	–	When used as the first character in search string (meaning BUF FND “a^b” will match “a^b” but BUF FND “\^aa” is needed to match “^aa”)
'*'	–	Always except inside [ ]
'@'	–	Always except inside [ ]
'{'	–	Always except inside [ ]
'}'	–	Always except inside [ ]
'['	–	Always
']'	–	Always
'-'	–	See [ ] above (meaning [-az] matches a single character 'a' or 'z' or '-'. [a-z] matches a single character 'a' or 'b' or 'c' or ... or 'z')
'~'	–	See [~] above
'\'	–	Always except inside [ ]
'?'	–	Always except inside [ ]
'<'	–	Always except inside [ ]
'>'	–	Always except inside [ ]

Menu
Contents
Indexes
Search

Worksheet Buffer Commands

BUF - Output Collection On/Off

BUF Yes | No Def: No

Turns the collecting of CODE V output into worksheet B0 on or off. When collecting is on, any CODE V output is appended to worksheet B0. Buffer collection is independent of the OUT command setting but is affected by VER and ECH.

BUF ? - Worksheet Status

BUF ?

Reports status of CODE V worksheets. All non-empty worksheets are listed. This status information includes:

- The collection status of B0 (on/off)
- The current worksheet (indicated by '>')
- The number of bytes occupied by each worksheet
- The current row and column in each worksheet
- The last row and column in each worksheet
- The maximum number of allowed rows (for B0 only)

BUF ? Example

```
CODE V> buf ?
            Size        Current             Last          Maximum
Buffer     (bytes)      Row   Column      Row   Column      Rows
------     -------    ------- -------    ------- -------    -------
>B0   (on)  41493          1       1          5       2      10000
 B1           940          1       1          1       1
```

BUF COP - Copy Cells

BUF COP [Bn] [Ik | Ii..j] [Jm | Jr..s] [TRN] [VAL | FMT]

Copies cells from the specified area to the current position. The B, I, and J qualifiers define a source region of cells. If the I qualifier is omitted, I1..L is used; if the J qualifier is omitted, Jl..L is used. If the I and J qualifiers are both omitted, all of Bn is copied. The region is copied to the current position, which marks the upper left hand cell of the destination region. If the destination cells do not exist, they are created. The source and destination regions may not overlap. The destination buffer cannot be B0.

If the TRN qualifier is specified, the given cells will be transposed during copy, i.e., a column in the source block is copied into a row of the destination block and vice versa.

The VAL and FMT qualifiers allow you to copy just cell values or cell formats without copying the other. The default is to copy both the cell value and cell format during the copy; VAL limits the copying to just cell values while preserving the format of destination cells, and FMT limits the copying to cell formats while preserving the value of the destination cell. If when using the VAL qualifier the cell types do not match

(for example, copying a number into a cell formatted for a string), the value is copied and the format is set to the default format. If the cell types do not match for the FMT qualifier, no copying is done and a message indicating this is displayed. Use the FMT or VAL qualifier if you have created a preformatted table and are just filling in the values from a CODE V run.

BUF COP Example

```
CODE V> BUF LIS B1

1:       1       1.2
2:       2       3.5
3:       3       2.7
4:       4       9.8
5:       5      10.6

CODE V> BUF LIS B2 I3..7

3:       LAB1    100
4:       LAB2    200
5:       LAB3    300
6:       LAB4    400
7:       LAB5    500

CODE V> BUF MOV B2 I3 J3
CODE V> BUF COP B1 J2                ! I1..L implied
CODE V> BUF LIS B2 I3..7

3:       LAB1    100    1.2
4:       LAB2    200    3.5
5:       LAB3    300    2.7
6:       LAB4    400    9.8
7:       LAB5    500    10.6

CODE V> BUF MOV B3
CODE V> BUF COP B1 TRN
CODE V> BUF LIS B3

1:       1         2         3         4         5
2:     1.2       3.5       2.7       9.8      10.6
```

BUF DEL - Delete Worksheet Buffer

	BUF DEL	Bk | Bi..j
or	**BUF DEL**	[Bn] Ik | Ii..j
or	**BUF DEL**	[Bn] Ik | Ii..j Jm | Jr..s

Deletes a specified worksheet buffer or range of buffers or deletes portions of a specified buffer.

The first form of the command completely deletes a specified buffer or range of buffers and tells you how many buffers were deleted. A buffer qualifier is required. If the current buffer is deleted, BUF DEL will make B0 the current buffer.

The second form of the command deletes a single row or range of rows from the specified buffer (default is the current buffer). Leftover rows below the deleted rows are moved up and renumbered.

The third form of the command deletes columns of cells from a specified buffer (default is the current buffer). The column or column range is deleted for the specified row or row range. Leftover columns to the right of the deleted columns are moved left and renumbered. Note that this form of the command only deletes columns, it does not delete rows; removing all the columns in a row leaves a blank row.

BUF DEL Example

```
CODE V> buf lis
     1: A| B| C| D| E
     2: A| B| C| D
     3: A| B| C
     4: A| B
     5: A
CODE V> buf del i2..3 j2..3          ! Column delete
CODE V> buf lis
     1: A| B| C| D| E
     2: A| D
     3: A
     4: A| B
     5: A
CODE V> buf del i2..3                ! Row delete
CODE V> buf lis
     1: A| B| C| D| E
     2: A| B
     3. A
```

BUF DIF - Buffer Differences

BUF DIF [Bn] [Ik | Ii..j] [Jm | Jr..s]

Compares a range of cells in row major order with a range of cells of the same size at the current buffer location and generates a report of the differences.

The specified area of the specified buffer is compared to an area of the same size at the current buffer location. The specified area is considered to be the original version of the data, and the new version is located at the current buffer position. This distinction is necessary to interpret the difference listing. The difference report is of the form

n1	a	n3, n4
n1, n2	d	n3
n1, n2	c	n3, n4

These describe sections of differences. The difference codes “a,” “d,” and “c” indicate if the section of text has been added, deleted, or changed between the original and new versions of the data. The numbers n1, n2, n3, and n4 are cell addresses of the locations of the differences. These cell addresses have the form ImJn, corresponding to row m and column n. Two cell addresses separated by a comma indicate a range of cells. Note that the buffer range in the BUF DIF command refers to a rectangular area in the buffer, but a range specified by n1, n2 represents a row major order sequence of cells between n1 and n2 which may not be a rectangular area.

The cell addresses on the left side of the difference code delimit the range of cells in the original version of the data which are different in the new version, and the cell addresses on the right side delimit the corresponding range of cells in the new version with the differences.

Following the difference codes lines are the lines of changes. Each line is preceded by “<,” indicating it came from the original version, or by “>,” indicating it came from the new version.

Menu
Contents
Indexes
Search

BUF EXP - Export to File

BUF EXP [Bn] [Ii..j] [Jk..l] [SEP] [filename]

Exports all or a portion of specified buffer to a file (default filename CODEV.DAT, default extender .DAT). Default is to export all the contents of the current buffer; the I and J qualifiers may be used to limit the portion of the buffer to be exported. Output file is tab delimited (columns separated by single tab character, rows separated by single new line character) unless SEP qualifier is used (note that tab delimited files can be imported into many spreadsheet or database programs). SEP causes the separator specified by **BUF SEP** to be used to delimit the file.

BUF EXP Example

```
CODE V> buf sep '|'
CODE V> buf lis
     1: A| B| C
     2: A| B| C
     3: A| B| C
CODE V> buf exp file1            ! Export using tab as separator
CODE V> buf exp sep file2        ! Export using user-defined
                                 !  separator

type file1.dat                   ! DOS command
A    B    C
A    B    C
A    B    C

type file2.dat
A| B| C
A| B| C
A| B| C
```

BUF FMT - Output Format

BUF FMT [Bn] [Ik | Ii..j] [Jm | Jr..s] 'format_spec'...

Specifies an output format or formats for the given cells in the region. The default is to apply the format to the current cell of the current buffer. These formats are used to convert the cell values to text strings when the cells are listed, exported, or retrieved with database items. Note that this command does not change the value or precision of the data in the cell. The format specification must match the cell type (string formats for STR cells, number formats for NUM cells) or a warning message is printed and the cell format is unchanged. (Thus, this command can be used to set a "global" numeric format over the specified cell range without affecting string contents, for example.) The list of format specifications are the individual field pictures normally comprising a format string (e.g., '3d', 'ccc', etc.). Each individual field picture applies to the cell specified. Additionally a blank format string (" ") is allowed to clear a previous format. In this case the format reverts to a default format which matches the data type of the particular cell.

A previously formatted cell will retain its format if another value is placed using BUF PUT as long as the new value has a similar data type to the previous value. Otherwise the old format is deleted and replaced with the default format. Thus, worksheet cells can be pre-formatted so long as the data type of the eventual BUF PUT matches the format specifier.

As with BUF PUT, the BUF FMT command takes multiple field pictures as well as supporting row and column ranges. It follows the exact same rules for multiple entries as BUF PUT. That is:

1. If there are no row or column ranges specified, CODE V moves to the desired cell and for each 'format_spec' entered, cell formats are filled in horizontally to the right.

2. If a row or column range or both are specified, CODE V takes the format specifications entered and fills in the format of the cell using the priority

   a. Left to Right

   b. Top to Bottom

   within the "rectangle" specified by the I and J ranges. If the number of cells specified by the I and J ranges is less than the number of format specifications entered, all extra format specifications on the FMT command are ignored. If, however, the number of format specifications entered is less than the range spanned by I and J, the cells are filled out through the entire range by repeating the formats over the set of format specifications entered.

BUF FMT Example

```
CODE V> BUF LIS B1

1: THI =             2
2: THI =             2
3: THI =             2

CODE V> BUF FMT B1 I1..3 J1..2 '6c' 'd'    ! Repeats format for
                                           ! rows 1-3

CODE V> BUF LIS B1

  1: THI =  2
  2: THI =  2
  3: THI =  2
```

BUF FND - Find Match

```
BUF FND [Bn] [Ik | Ii..j] [Jm | Jr..s] [FWD | REV] [OR | AND | ADJ]
        "expression" | word | num...
or  BUF FND [FWD | REV]
```

Finds the first and successive matches of the given expressions, words, or numbers in worksheet Bn following (preceding for REV) the current cell position (Ic, Jc). If the search is successful, the current worksheet location is set to the cell that contains the beginning of the match. No error message is issued if the find was unsuccessful. To determine success or failure of a match use the (BUF.FND) database item immediately after issuing the BUF FND command.

The I and J qualifiers may be used to limit the search area. This is very useful to help exclude rows and columns to ignore. Matching is performed in the same manner as in the BUF LIS command:

OR	–	finds rows containing one or more cells that match ANY of the expressions, words or nums (default).
AND	–	finds rows containing one or more cells that match ALL of the expressions, words or nums.
ADJ	–	finds rows containing adjacent cells that match the set of expressions, words or nums IN THE ORDER GIVEN.
FWD	–	search in the forward direction.
REV	–	search in the reverse direction.

Matching expressions can be simple text or complex expressions. See “Matching Expressions” on page 25E-7 for a complete description of what is possible.

The second form of BUF FND is used if an expression, word or num is not given. In this case the last search is repeated, thus providing a “find next” or “find preceding” function. Note that B, I, J, OR, AND, and ADJ are not allowed in the second form of BUF FND (FWD or REV are allowed however). Whenever using BUF FND with expression, word, or num, the first cell location is allowed to match the condition(s). This is also true if the current position of the buffer being searched has changed via a BUF MOV since the last BUF FND. When using BUF FND in the “find next” or “find preceding” role, the current cell is ignored for possible match.

The default action is to search the entire buffer in the FOR direction using OR for multiple expressions.

BUF IMP - Import Data

```
BUF IMP [Bn] [LIS | SEP] filename
```

Imports data from a text file (default extender .DAT) to the end of the specified buffer (default is the current buffer). The data is parsed into rows and columns as if it were output from CODE V. BUF IMP can import CODE V output text files or any tab delimited files, such as from a spreadsheet program. If importing non-tab delimited files, use the LIS or SEP qualifier. LIS causes the default parsing algorithm to be applied to the incoming text to break it into cells. SEP causes the separator defined by the BUF SEP command to be used as the delimiter in reading the file (note that the use of letters and numbers in the separator string may confuse BUF IMP). After completion of the file import, the current position of Bn is the row following the last row imported. Files cannot be imported into B0.

Menu
Contents
Indexes
Search

BUF INS - Insert Rows and Columns

	BUF INS	[Bn] [Ik] [num_rows]
or	**BUF INS**	[Bn] [Ik | Ii..j] Jm [num_columns]

Inserts rows and columns into a specified buffer (default is the current buffer). If the specified buffer does not exist, it is created. Rows or columns cannot be inserted into buffer B0.

The first form of the command inserts blank rows (i.e., no cells in them) above row k (row k and all following rows are moved down and renumbered). If row k does not exist, rows are created as needed, with row k counting as the first of the inserted number of rows. The default is to insert one row in the current buffer at the current row location (Ic).

The second form of the command inserts columns into the specified row or row range at column m (column m and all columns to the right of m are moved right and renumbered). If the specified row or rows do not exist, rows are added as needed. If column m does not exist in one or more of the specified rows, columns are added as needed; column m counts as the first of the inserted number of columns. The default is to insert one column at column m in all rows (Ia) in the current buffer.

Note that in either form of the command, the insertion process does not move the current position pointer in the buffer.

BUF INS Example

```
CODE V> buf sep '|'
CODE V> buf lis
     1: A| B| C
     2: D| E| F
     3: G| H| I
CODE V> buf ins i2 2        ! Insert 2 rows at row 2
     1: A| B| C
     2:
     3:
     4: D| E| F
     5: G| H| I
CODE V> buf ins il+1        ! Insert one row after the last row
CODE V> buf lis
     1: A|B|C
     2:
     3:
     4: D| E F
     5: G| H| I
     6:
CODE V> buf ins i3..5 j3 2  ! Insert 2 columns starting at
                            !  column 3 in rows 3-5
CODE V> buf lis
     1: A|B|C
     2:
     3: |||
     4: D| E||| F
     5: G| H||| I
     6:
```

Menu
Contents
Indexes
Search

BUF LEN - Maximum Number of Lines

BUF LEN maximum_B0_lines	Def: 10000

This sets the maximum number of lines to be stored in worksheet B0. This enables you to "efficiently" manage memory. Once B0 contains this maximum number of lines, any more lines that are added, cause B0 to scroll, i.e., for each line added at the bottom of B0, one line is lost at the top. Note, however, that even though lines are lost, row numbers continue to increase.

If maximum_B0_lines is decreased from its original limit, old lines will be freed to accommodate the new size. For example, with 1000 lines in B0, the command issued is BUF LEN 10. The result is that the oldest 990 lines would be freed and unavailable.

BUF LIS - List Specified Worksheet

BUF LIS [Bn] [Ik | Ii..j] [Jm | Jr..s] [NOL] [OR | AND | ADJ] ["expression" | word | num...]

List specified worksheet (default is current worksheet). The I and J qualifiers may be used to limit the portion of the worksheet to be listed (the default is the entire buffer). When a row is listed, a single blank is inserted between cells to visually separate them (this can be changed with BUF SEP). If B0 collection is enabled, the listed rows are appended to B0 as with any other CODE V output.

The NOL qualifier will inhibit line numbers from being displayed during the list operation. The default is to display the line numbers before the actual row data.

The remaining optional arguments to BUF LIS allow you to list only rows that contain cells that match a given set of matching expressions. This is useful if you wish to scan the entire buffer for a given set of key strings or numbers. Matching can get complex if you desire (see the section on Matching Expressions). The matching can be further limited by the use of optional qualifiers (the default is OR):

OR	–	lists rows containing one or more cells that match ANY of the expressions, words or nums (default).
AND	–	lists rows containing one or more cells that match ALL of the expressions, words or nums.
ADJ	–	lists rows containing adjacent cells that match the set of expressions, words or nums IN THE ORDER GIVEN.

You use the BUF FMT command to change the format of a cell. This allows you to control the output precision of NUM cells and the cell width of STR cells in BUF LIS.

BUF LIS Example

```
CODE V> buf lis
      1:  cell1 cell2
CODE V> buf lis and "cell2" "cell1"   ! AND is order independent
      1:  cell1 cell2
CODE V> buf lis adj "cell2" "cell1"   ! ADJ requires the match
CODE V> buf lis adj "cell1" "cell2"   !  to occur in order
      1:  cell1 cell2
```

BUF MOV - Move Inside Buffer

BUF MOV [Bn] [Ij] [Jk]

Sets the current buffer and the current row and column inside that buffer. The default is to move to the current location in the specified (or current) buffer. Moving to an empty buffer will cause that buffer to be created.

BUF PRS - Reparse Rows

BUF PRS [Ik | Ii..j] ["format_str"]

Reparses the rows of worksheet B0. This command takes the original text output line and allows you to choose the boundaries from which to split the text into cells. The I qualifier can be used to limit the range of rows to be reparsed (default is to apply to all rows). If the format string is omitted, the default parsing algorithm is applied to the specified rows.

The format string specifies how a single line of text is to be divided into columns. It resembles a Q format specifier exactly in appearance (see Q format specifiers in the Macro-PLUS section). The field specification symbols, C, D, E, and G, are used to define the position, length, and types of the columns. Any other "constant" characters in the format specify data that is to be ignored (i.e., not placed into a cell). Note that for the purpose of parsing a row, the D, E, and G symbols all produce a column of type "NUM".

BUF PRS Examples

```
CODE V> buf lis B0 i179
  179:     DLF S1     2.0000000v   -0.002  0.002    -0.009726
CODE V> eva (buf.col i179)
   (BUF.COL I179) = 5
CODE V> eva (buf.typ i179 j2)
   (BUF.TYP I179 J2) = "STR"
CODE V> eva (buf.txt i179)
   (BUF.TXT I179) = " DLF S1 2.0000000v -0.002 0.002  -0.009726"
```

Because of the trailing 'v' the default parsing algorithm assumes cell 2 is the string "2.0000000v". We can reparse this row specifying how to interpret text on this line using the BUF PRS command.

```
CODE V> buf prs i179 " '6c' 'd.7d''c' 'dd.ddd' 'd.ddd' '2d.6d'"
```

Produces the following cells on row I179:

Cell	Value	Type
1	"DLF S1"	STR
2	2.0000000	NUM
3	"v"	STR
4	-0.002	NUM
5	0.002	NUM
6	-0.009726	NUM

```
CODE V> eva (buf.typ i179 j2)
    (BUF.TYP I179 J2) = "NUM"
CODE V> eva (buf.col i179)
    (BUF.COL I179) = 6
CODE V> eva (buf.txt i179)
    (BUF.TXT I179) = "DLF S1 2.0000000 v -0.002 0.002 -0.009726"
```

Menu
Contents
Indexes
Search

BUF PUT - Put Values in Cells

BUF PUT [Bn] [Ik | Ii..j] [Jm | Jr..s] "string" | num...

Allows one or more values to be stored in a given range of cells (default is the current cell in the current buffer). The simple version of this command takes single I and J qualifiers and a single value. The value is stored in the appropriate cell with the type of the cell derived from the type of the datum. BUF PUT also supports multiple cell entry along with initialization of rows and columns in the following manner:

1. If there are no row or column ranges specified, CODE V moves to the desired cell and for each "string"|num input, cell values are filled in horizontally to the right.

2. If a row or column range or both are specified, CODE V takes the values entered and fills in the cell using the priority:

   a. Left to Right

   b. Top to Bottom

   within the "rectangle" specified by the I and J ranges. If the number of cells specified by the I and J ranges is less than the number of values entered, all extra data on the PUT command is ignored. If, however, the number of values entered is less than the range spanned by I and J, the cells are filled out through the entire range by repeating over the set of values entered.

BUF PUT overwrites the values of any previous cells. However, any previous cell formats created using BUF FMT are **not** erased unless the data type of the value entered does not match the previously existing cell format. Issuing a BUF PUT to a previously undefined worksheet causes that worksheet to be created. BUF PUT is not allowed for buffer B0.

BUF PUT Examples

```
CODE V> BUF PUT B1 I1..10 J1..2 "THI = " 2.0
CODE V> BUF LIS B1

  2: THI =             2
  3: THI =             2
  4: THI =             2
  5: THI =             2
  6: THI =             2
  7: THI =             2
  8: THI =             2
  9: THI =             2
 10: THI =             2

CODE V> BUF PUT B2 I1 J1 "Cell 1" -0.8e-2 "Cell 3" "Cell 4"
CODE V> BUF LIS B2

 1: Cell 1        -0.008  Cell 3 Cell 4
```

BUF RES - Restore Worksheet

BUF RES [Bn] filename

Restores a worksheet from the named file (default extender .BUF) into the named buffer (default is current buffer). Worksheets cannot be restored into non-empty buffers (to do this, delete the buffer before restoring into it). Worksheets cannot be restored into worksheet B0.

BUF SAV - Save Worksheet

BUF SAV [Bn] [filename]

Saves the entire worksheet to the named file with default extension .BUF (default file is CODEV.BUF). The file is in a binary format (i.e., it cannot be edited). The default is to save the current buffer.

BUF SEP - Cell Separator String

BUF SEP [separator_string] Def: " "

Defines the string used to separate cells for BUF LIS and (BUF.TXT). For example, BUF SEP "|" will place a vertical bar in between cells for BUF LIS and (BUF.TXT).

BUF SRT - Sort Rows

BUF SRT [Bn] [Ii..j] Jk [ASC | DES]

Sorts rows of worksheet Bn (default is current buffer) based on the contents of column Jk. The default sorting order is ascending (ASC) from lowest value to highest. When sorting, empty cells have the lowest values, then numeric cells are sorted by numeric value, followed by text cells sorted alphabetically (ignoring case). Sorting in descending order (DES) does just the reverse. If two cells have exactly the equivalent keys, the order of the rows is random. The I qualifier can be used to limit the area sorted (default is all rows). Worksheet B0 cannot be sorted.

BUF SRT Example

```
CODE V> BUF LIS

 1:      This is a table
 2:      10      String 1
 3:      21      string 2
 4:      15      string 3
 5:      0       String 4
 6:              String 5
 7:      End of table

CODE V> BUF SRT J1
CODE V> BUF LIS

 1:              String 5
 2:      0       String 4
 3:      10      String 1
 4:      15      string 3
 5:      21      string 2
 6:      End of table
 7:      This is a table

CODE V> BUF SRT DES J2
CODE V> BUF LIS

 1:              String 5
 2:      0       String 4
 3:      15      string 3
 4:      21      string 2
 5:      10      String 1
 6:      This is a table   ! Note these two rows have no J2
 7:      End of table      !  so their order is undefined
```

Menu
Contents
Indexes
Search

Worksheet Buffer Database Items

The following Macro-PLUS database items allow you access to all defined worksheet data. In all cases, if the optional buffer, row, or column qualifier is absent, Macro-PLUS picks up the value from the current worksheet and current cell location. Be sure to enclose each database item in parentheses.

Database Item	Return Type	Description
BUF.B	(NUM)	Returns the number of the current worksheet.
BUF.COL [Bn] [Ik]	(NUM)	Returns the number of cells (number of columns) in row Ik of worksheet Bn.
BUF.EMP [Bn]	(NUM)	Returns 1 if worksheet Bn is empty, otherwise 0.
BUF.FMT [Bn] [Ik] [Jm]	(STR)	Returns an applicable Q format field picture for the given cell. " " if no format previously given.
BUF.FND	(NUM)	Returns 1 if the previous BUF FND was successful, otherwise 0.
BUF.FST [Bn]	(NUM)	Returns the number of the first row position in Bn (=1 for B1 and greater).
BUF.I [Bn]	(NUM)	Returns the row position of current cell in worksheet Bn (Ic).
BUF.J [Bn]	(NUM)	Returns the column position of current cell in worksheet Bn (Jc).
BUF.LEN	(NUM)	Returns the maximum number of rows allowed in worksheet B0.
BUF.LST [Bn]	(NUM)	Returns the row position of last row in worksheet Bn.
BUF.MXJ [Bn]	(NUM)	Returns the largest column number for all rows in worksheet Bn (Jl).
BUF.NUM [Bn] [Ik] [Jm]	(NUM)	Returns the numeric representation of specified cell. An error occurs if the type of the cell is not NUM.
BUF.ON	(NUM)	Returns 1 if output is currently being collected in B0, otherwise 0.
BUF.STR [Bn] [Ik] [Jm]	(NUM)	Returns the string representation of specified cell. An error occurs if the type of the cell is UNKNOWN.
BUF.TXT [Bn] [Ik]	(NUM)	Returns a properly formatted, concatenated text string of all cells in row Ik.
BUF.TYP [Bn] [Ik] [Jm]	(STR)	Returns the type of the specified cell as a text string: "NUM" – Numeric Cell "STR" – String Cell "UNKNOWN" – Cell that does not exist or has no type.

Menu
Contents
Indexes
Search

Database Item Examples

```
CODE V> VER YES
CODE V> ECH NO
CODE V> buf
CODE V> buf len 5
CODE V> eva (buf.on)
     (BUF.ON) = 1
CODE V> eva (buf.len)
     (BUF.LEN) = 5
CODE V> buf no
CODE V> buf lis
      6: CODE V> eva (buf.on)
      7:       (BUF.ON) = 1
      8: CODE V> eva (buf.len)
      9:       (BUF.LEN) = 5
     10: CODE V> buf no
CODE V> eva (buf.fst b0)
     (BUF.FST B0) = 6
CODE V>
CODE V> buf mov b1
      Created new worksheet B1.
CODE V> buf sep '|'
CODE V> buf put i1 "This" "is" 4.0 "cells"
CODE V> buf put i3 "A 3x3 table follows:"
CODE V> buf put i4..6 j2..4 1.2 8.7 4.3e-12
CODE V> buf fmt i1 ' ' 'ccccc' 'd.d' ' '
CODE V> buf fmt i4..6 j3 'ddd.d'
CODE V> buf lis
      1: This|is   |4.0|cells
      2:
      3: A 3x3 table follows:
      4: |1.2|  8.7|4.300000e-12
      5: |1.2|  8.7|4.300000e-12
      6: |1.2|  8.7|4.300000e-12
CODE V> eva (buf.emp b1)
     (BUF.EMP B1) = 0
CODE V> eva (buf.emp b99)
     (BUF.EMP B99) = 1
CODE V> eva (buf.col i1)
     (BUF.COL I1) = 4
CODE V> eva (buf.col i3)
     (BUF.COL I3) = 1
CODE V> eva (buf.fmt i4 j3)
     (BUF.FMT I4 J3) = "3D.1D"
CODE V> eva (buf.fmt i1 j1)
     (BUF.FMT I1 J1) = ""
CODE V> eva (buf.fnd)
     (BUF.FND) = 0
CODE V> buf fnd table
CODE V> eva (buf.fnd)
     (BUF.FND) = 1
CODE V> eva (buf.b)
     (BUF.B) = 1
CODE V> eva (buf.i)
     (BUF.I) = 3
CODE V> eva (buf.j)
     (BUF.J) = 1
```

Menu
Contents
Indexes
Search

```
CODE V> eva (buf.lst)
     (BUF.LST) = 6
CODE V> eva (buf.mxj)
     (BUF.MXJ) = 4
CODE V> eva (buf.num i4 j1)
     (BUF.NUM I4 JL) = 0.43E-11
CODE V> eva (buf.str i3 j1)
     (BUF.STR I3 J1) = "A 3x3 table follows:"
CODE V> eva (buf.txt i1)
     (BUF.TXT I1) = "This|is   |4.0|cells"
CODE V> eva (buf.typ i1 j1)
     (BUF.TYP I1 J1) = "STR"
CODE V> eva (buf.typ i4 j2)
     (BUF.TYP I4 J2) = "NUM"
CODE V> eva (buf.typ i2)
     (BUF.TYP I2) = "UNKNOWN"
```

Technical Notes

Memory Use

Worksheet buffers can consume large amounts of memory. It is advisable to frequently check the amount of dynamic memory used for worksheets with the BUF ? query. If there are buffers not being used anymore, delete them. Many other options within CODE V require dynamic memory so it is wise to free up unused buffers when they are not needed anymore. Note that buffers are never deleted automatically—you must specifically delete them.