NEWKEY

TABLE OF CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

CHAPTER 1
INTRODUCTION

1.1 Purpose and Terminology

The Newkey keyboard enhancer simplifies the entry of common keystroke sequences by allowing these sequences to be assigned to any key desired. Once assigned to a particular key, whenever that key is struck the predefined sequence of keystrokes will be substituted for the struck key.

For example, the <altc> key might be defined as "copy" and whenever <altc> is struck the string "copy" will be substituted for it. The process of assigning a sequence of keystrokes to another key is called "macro definition". The process of substituting the Newkey-assigned sequence for a key that is actually struck is called "playback".

When we refer to specific keystrokes, such as <altc> in the above example, the keystroke name will be enclosed in a "<" and a ">". The keystroke name will be made up of an optional prefix (alt, caps, ctrl, or num) plus the a key's name. To enter <altc> for example, press the "alt" key and then while continuing to hold the alt key down, press the "c" key.

Newkey allows customization of software packages, ready creation of boiler plate passages, keyboard redefinition, and other useful purposes. Newkey also provides many more features including:

- Ability to define any key support by your keyboard
- Shorthand mode
- Display windows
- Menu macros
- Full-featured macro editor
- Multi-character macro names
- Cut & Paste
- Extended keyboard buffer
- Screen saver feature
- Saving macros from memory to disk
- Many more features too numerous to list

Newkeysp, the Newkey Support Program, provides many functions to complement Newkey, including:

- Execute macros from batch files
- Execute Newkey functions from batch files
- Set menu colors and other options

1.2 Installation Tips

- First make a copy of the Newkey disk and put away the supplied disk as a backup.

- Next read the "readme" file. This file will contain the latest Newkey information which has come up since this manual was printed plus a section on using Newkey with other software.

- Users of earlier versions should consult the "Summary of Changes" in appendix F for important conversion information.

- Hard disk users will want to put the "newkey.exe" and "newkeysp.exe" programs in a directory which is specified in their "path" command so that they are readily accessible from all of their sub-directories.

1.3 Guided Tour

A guided tour comes with Newkey and may be invoked by issuing the command "demo" while the Newkey disk is in the default disk drive. This tour is done entirely using Newkey macros and illustrates many of Newkey's major features.

1.4 Files Supplied With Newkey

For a list of the files supplied with Newkey print the "readme" file. We used to supply a list in the manual, but found that the list would eventually became obsolete. The list on the readme will be up-to-date.

1.5 Sample Macro Files

Several sample macro files are supplied with Newkey to illustrate its use and provide you with a starting point. These macro files are self-documenting and you may find it useful to print them before they are used.

CHAPTER 2
GETTING STARTED

2.1 How Newkey Works

A basic understanding of how Newkey works is important to using it well. When you load Newkey, it makes itself part of the software controlling your computer. From that time forward it is continually "listening in" to the conversation you are having with your computer as you type at the keyboard. It listens when you are at the DOS command line, it listens when you are working in your word-processor, your spreadsheet, or whatever program you are running at the time.

As it listens, it checks each keystroke against its list of macros. When a keystroke with a macro assigned to it is recognized, Newkey jumps in and plays the macro back. It does this by fooling the computer's software into thinking that it is reading from the keyboard when in fact, it is Newkey which is feeding the keystrokes to the computer's software and not you at the keyboard. When the macro is done, Newkey returns everything to normal and lets your computer's software start reading from the keyboard once again.
Of course Newkey actually does much more than simple macro playback, but an understanding of its basic functioning will help you to use it to its fullest.

2.2 Loading Newkey

To load Newkey, type

newkey [/*buffer size*] [/f] [/noems]

and press the <enter> key. Within a second or two a short message will be displayed with the line

Newkey loaded

at the bottom.

When you receive this message, Newkey is loaded and ready to use. You may wish to load a predefined set of macros. To do this, refer to section 11.2, "Load File", for more information.

2.2.1 Buffer Size

Newkey comes with a default macro buffer size which will accept up to 1000 characters of macro definitions. This default may be changed at run time by specifying the parameter */xxxxx* when first invoking Newkey where *xxxxx* is up to a 5 digit number specifying the number of characters to be reserved for the macro buffer. Newkey will reserve twice this number of bytes in memory (each character requires two bytes).

For example:

newkey /1000

will reserve enough space for 1000 characters of playback (2000 bytes of storage.)

The legal range for this number is 5 - 32000.

2.2.2 Expanded Memory (EMS) Usage

Newkey will now automatically use EMS memory to store its macro and data buffers if your computer has enough EMS available. This can save substantial amounts of your precious 640K memory but can

also introduce compatibility problems with other software (Windows in some cases). To prevent EMS usage use the "/noems" parameter when loading Newkey. For example:

newkey /3000 /noems

will load Newkey and prevent usage of EMS memory.

2.2.3 Forced Load

Occasionally Newkey will think that it is already loaded when in fact it is not. In order to force Newkey to load, add the parameter '/f' when initially loading Newkey.

For example:

newkey /f
newkey /1000 /f

will force newkey to load even if it thinks it is already loaded.

The most likely time this will be necessary is when you are using a warm boot program which preserves certain areas of the computer's memory (JBOOT, for example). If Newkey is accidently loaded a second time, results will be unpredictable.

2.2.4 Newkey Load Order

The number of programs interacting with the operating system is growing every day. Many of these programs make themselves resident and latch on to the same interrupts Newkey uses. Version 5.4 has been written to provide the maximum possible compatibility with these programs.

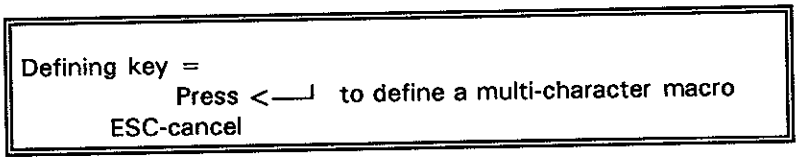
Generally speaking, it is impossible to make any hard and fast rules about the order in which your memory-resident programs should be loaded. Some programs require Newkey to be loaded before they are loaded, some after. For further information on using Newkey with other software, refer to the readme file on the Newkey disk.

2.3 Defining a Macro

To define a macro follow these steps:

1. Press the <alt=> key.

A window will open up looking like this:



2. Press the key you wish to define.

Let's assume that you are defining <altc> to be "copy". Press <altc> and it will be displayed in the window.

```
Defining key = <altc>
Description  =
             ESC-cancel, <┘ -end description
```

3. Now you will be asked to enter a description.

This optional field allows you to create a short description of what the macro does. In the example below, "Start copying" is the description that has been entered.

```
Defining key = <altc>
Description  = Start copying
             ESC-cancel, <┘ -end description
```

When you are finished entering the description press <enter>. At this point the following 4 things will happen:

- The window will close
- A status line will be displayed at the top of your screen
- The cursor will take on a block shape
- A beep will sound each time you press a key

As long as you are in macro definition mode, the status line will be displayed, the cursor will retain some sort of block shape, and each keystroke will cause a beep.

4. Now enter the keystrokes you want the macro to represent ("copy " in our example.)

Your software will continue to act on these keystrokes as if macro definition was not occurring. Newkey is just listening in as you type and remembering your keystrokes. The software you are using, whether you are at the DOS command line, in your word-processor, spreadsheet, or whatever does not realize that Newkey is there. This enables you to monitor the macro definition to ensure that you are actually creating a macro that will do what you wish, because as you type, your software is reacting to your keystrokes just as it will when you later play the macro back.

5. When you have finished defining the macro press the <alt-> key.

The status line will disappear, the cursor will take on its normal shape, and your macro definition is finished.

Now press <altc> and watch "copy" be played back. Unless saved in a disk file, the new macro will be lost when the system is rebooted. For more information on saving macros, refer to section 11.1, "Save file".

Macros may also be created or modified by using Newkey's macro editor on macros in memory or with your word processor or text editor on a saved macro file. For further information on editing macros, refer to chapter 5, "Editing Macros".

Most of the macro definition messages and other aids described in this chapter as well as others can be customized to suit your preferences. If you do not need these aids or protection you can eliminate them and the additional keystrokes they require. These options are described in section 10.4, "Customizing Macro Definition Messages".

2.4 Macro Definition Status Line

Newkey will display a status line during macro definition. This status line will display in reverse video the key being defined, the current defining mode, and which keys to press to end or cancel macro definition. Possible defining modes are:

TEXT = text mode
FIX = fixed length pause
VAR = variable length pause

The status line will not be displayed if your program is currently in graphics mode.

2.5 Space Considerations

Unless told otherwise, Newkey automatically reserves enough macro space for 1000 characters. If new macro definitions total more than 1000 characters in a session, Newkey will display this message in reverse video:

OUT OF MEMORY - PRESS ESC TO CONTINUE

If you receive this message, all further new macro definition will be halted, including the one you are currently creating. At this point there are two ways to get more macro definition space:

Method 1

1. Save the current macros (section 11.1)
2. Unload Newkey (section 4.5).
3. Reload Newkey increasing the size of the macro buffer (section 2.2.1).
4. Reload the saved macros (section 11.2).

Method 2

Delete some unused or unnecessary macros (section 5.4, "Deleting a Macro").

2.6 Error Correction

During macro definition, Newkey treats the <ctrlh> key as a back space and delete. This allows you to correct a mistake during macro definition without the correction becoming part of the macro.

You may also edit the macro being defined by pressing the <ctrl\> key. This will take you into the Newkey macro editor for the key being defined. You may also edit the macro by entering the macro editor in the normal manner through the pop-up key.

2.7 Pop-up Access to Newkey's Features

Most of Newkey's features may be invoked at any time without leaving your current program by pressing the pop-up key, <alt/>. Newkey will save your current screen, present you with a menu of options, and when you are done, restore your screen just where you left off.

This feature is not available in the smaller versions of Newkey, Newkeysm and Newkeyvs.

2.8 Menu Access to Macro Definition Commands

This option allows you another method of executing the macro commands described later besides pressing the appropriate control key. If you don't remember what key does what, instead of looking it up call up the pop-up menu and select the "Execute macro commands" option.

A menu will be displayed listing all of your possible options. Only those which are highlighted may be selected. Those which are highlighted will depend on several factors, such as whether you are in macro definition mode or not.

CHAPTER 2
GETTING STARTED

CHAPTER 3
MACRO FUNCTIONS

This chapter describes many of the special purpose functions which can be invoked by Newkey's macros. A function may not be described in this chapter if it is normally inserted by the display or menu macro editors. For a complete list of macro functions refer to Appendix B.

3.1 Entering a Macro Function

Most of Newkey's macro functions must be entered by editing the macro and inserting the desired functions. Newkey allows you to edit a macro as it is being defined or after it has been defined.

To enter a function as a macro is being defined, press the "edit macro being defined" key, <ctrl\>. This will cause you to enter the macro editor with the cursor positioned at the end of the macro you are currently defining. At this point you may insert special functions (F2) or do any of the other normal macro editor's functions. When you are done, Newkey will return you to where you left off.

To enter a function after the macro has been defined use the macro editor as described in Chapter 5, 'Editing Macros'.

3.2 Beep

Macro function: {beep}

Newkey allows you to beep during macro playback.

3.3 Cancel

Macro function: {cancel}

This will cancel all currently executing macros and is used most often in conjunction with the if screen macro functions.

3.4 Clear Macros

Macro function: {clearmac}

This function will clear macros from memory. It will not clear any macros currently being played back or guarded macros.

3.5 Clear Screen

Macro function: {cls}

This function will clear the screen during macro playback.

3.6 Cursor On/Off

Macro function: {cursoff}
{curson}

These functions will turn the cursor off or on.

3.7 Cut From Screen

Macro function: {cut *macroname row,column,lines,length*}

This function will cut the specified number of *lines* for the specified *length* starting at *row* and *column* on your screen and assign it to the specified macro. It works just like the cut and paste option, except that the macro does the cutting for you.

3.8 Date/Time

Macro function: {date}*date template*{enddate}
{time}*time template*{endtime}

Newkey now allows you to display the date and time during macro playback in an almost unlimited number of formats. Using the basic building blocks of the date and time functions you can build your own date/time template. These building blocks are:

Time functions

hh = hours, twelve hour clock
mm = minutes
ss = seconds
tt = hours, 24 hour clock
xx = am,pm
XX = AM,PM

Date functions

dd = day
mm = month
yy,yyy,yyyy = year
ddd,mmm = abbreviated name
dddd,mmmm = full name

You are not limited to just these building blocks. Any characters found in the date/time function which can not be interpreted as one of the building blocks will be returned just as they are. This means that you have complete flexibility in creating your own date/time function.

The following examples illustrate some of the more popular date/time formats. For example, assume that it is 1:13:03 pm on Sunday, September 11, 1986.

Sample Date Functions

Template	What Newkey returns
mm/dd/yy	5/11/86
yy/mm/dd	86/5/11
mm-dd-yy	5-11-86
mm/dd/yyyy	5/11/1986
mmmm dd, yyyy	September 11, 1986
mmm dd, yyyy	Sep 11, 1986
mmm. dd, yyyy	Sep. 11, 1986
dddd, mmm dd, yyyy	Sunday, Sep 11, 1986
ddd, mmm. dd, yyyy	Sun, Sep. 11, 1986

Sample Time Functions

Template	What Newkey returns
hh:mm:ss	1:13:3
hh:mm xx	1:13 pm
hh:mm XX	1:13 PM
tt:mm:ss	13:13:3

For further examples load the DATE.KEY macro file.

3.8.1 Zero Fill Option

The zero fill option will cause the date and time functions to zero fill the date/time template. For example, assume the time 8:03:01 am is being displayed by the time function, {begtime}hh:mm:ss{endtime}. Normally, "8:3:1", would be displayed, but with the zero fill option on, "08:03:01" would be displayed.

3.9 Define a Macro to a Specific Definition

Macro function: {define *macroname*}*macro definition*{defend}

This function will define the macro specified by *macroname* to *macro definition*. For example, when the macro <defineit>, which is defined as:

{begdef defineit}Let's define alta {define alta}I'm alta{defend}{enddef}

is executed, it will define <alta> as:

{begdef alta}I'm alta{enddef}

Now if <alta> is pressed, "I'm alta" will be played back.

3.10 Delete a Macro

Macro function: {delete *macroname*}

This function will delete the macro specified by *macroname*. It may be used to reclaim memory by deleting macros no longer used.

3.11 Fixed Length Fill-in-the-Blanks

Macro function: {ffld}
{ffldtran}

Newkey provides you with the ability to define fixed length fill-in-the-blanks fields within macros. When one of these is encountered in a macro, Newkey will stop playback and wait for the user to enter one keystroke from the keyboard. There are two types of fixed length fields, one which will accept the keystroke and return it without checking to see if a macro is assigned to it. This is the {ffld}

function. The {ffldtran} function behaves the same, except that if there is a macro assigned to the keystroke, it will playback the macro.

In addition to using the macro editor, a {ffld} may be inserted during macro definition by following these steps:

1. Press <ctrl]>. The cursor will change from a full block to half block with its top missing and the status line mode will change from "TEXT" to "FIX".

2. Enter whatever keystrokes you wish.

3. Press <ctrl]>. The cursor will change back to a full block and the status line mode will change from "FIX" to "TEXT".

For example, suppose you frequently started a letter with "Enclosed are the xx (some number) items per your request", you might wish to define one macro to mean "Enclosed are the xx items per your request". To do this you would follow these steps:

1. Press <alt=> to start macro definition.
2. Press <altl>, the key you wish to define.
3. Press <enter> to skip the description.
4. Type "Enclosed are the ".
5. Press <ctrl]> to start fixed length field.
6. Type in any two characters.
7. Press <ctrl]> to end fixed length field.
8. Type in " items per your request".
9. Press <alt-> to end macro definition.

Now if you were to press <altl>, "Enclosed are the " would be displayed and the system would wait for you to enter any two characters. Once you had entered the second character, the system would continue with " items per your request".

3.12 Guard a Macro

Macro function: {guard}

This function will guard a macro from being deleted when loading or clearing macros. For further information, refer to section 11.6, "Guarded Macros".

3.13 If Screen Equal or Not Equal

Macro function: {ifescr row,col}*TEXT*{endife}
{ifnescr row,col}*TEXT*{endifne}
{ifefscr}*TEXT*{endifef}
{ifnefscr}*TEXT*{endifnef}

These functions will test the contents of the screen and conditionally execute the immediately following macro function.

{ifescr row,col} - If the contents of the screen at the specified row and column equal "TEXT". (IFEqualSCReen)

{ifnescr row,col} - If the contents of the screen at the specified row and column do not equal "TEXT". (IFNotEqualSCReen)

{ifefscr} - If "TEXT" is anywhere on the screen. (IFEqualFullSCReen)

{ifnefscr} - If "TEXT" is not anywhere on the screen. (IFNotEqualFullSCReen)

"TEXT" can be any characters that can be displayed on the screen.

An example:

{ifescr 0,75}READY{endife}<go123><nextmac>

Assume the above is in a macro designed to be used by Lotus 1-2-3. When executed, Newkey would check row 0, column 75 of the screen for "READY" and if found, it would execute the <go123> macro, otherwise the <go123> function would be skipped and execution would continue with <nextmac>.

The if functions are commonly used to delay macro execution until a particular message is displayed telling the macro that it is ok to continue.

These are among the most difficult of Newkey's macro functions to use. See EXAMPLE.KEY for some sample if macros that will work at the DOS prompt and demonstrate some of the powerful things that can be done.

HINT: To help determine what row and column a piece of text starts on, the "Cut" screen has been changed to display the row and column of the current cursor position. Just pop up the Newkey menu (alt/), select "U", move the cursor to the start of the text you want to check for, note the row & column, and press ESC to cancel the cut. Then use the macro editor to insert the if function.

3.14 Input to Macro

Macro function: {input *macroname row,col*,1,*length*}

This function will build a window of 1 line and *length* columns starting at *row* and *col* and then accept input from the user. When <enter> is pressed, the user's input is assigned to the macro called *macroname*. Usually this function will be entered for you by the display macro editor.

3.15 Load Macros

Macro function: {begload}*filename.ext*{endload}
{begmerge}*filename.ext*{endmerge}
{begmergo}*filename.ext*{endmergo}

The {begload} function will load the macros from *filename.ext* into memory.

The {begmerge} function will merge the macros from *filename.ext* into memory.

CHAPTER 3
MACRO FUNCTIONS

The {begmergo} function will merge the macros from *filename.ext* into memory, overwriting the duplicates already in memory.

These functions will load macros more slowly than manually loading the macros because of the need to protect the macro(s) currently being played back. They will be ignored in the smaller versions, Newkeysm and Newkeyvs.

3.16 Playback Bypass

Macro function: {notran *macroname*}

Newkey provides the ability to prevent further playback within a macro. This is similar to turning Newkey off for just one keystroke. To invoke playback bypass follow these steps:

1. Press the no-translate key, <ctrl2>.
2. Press the key you wish to always be returned.

When Newkey encounters this key at this spot within the macro, it will return the key without further playback, even if the key is defined in the macro file. For example, suppose you wish to define the key <altk> to be "<ctrlk><ctrld>", but <ctrlk> is already defined as "<ctrlj><ctrlq>". Follow these steps:

1. Press <alt=> to start macro definition.
2. Press <altk>, the key you wish to define.
3. Press <enter> to skip the description.
4. Press <ctrl2> to signal that the next key is not to be translated.
5. Press <ctrlk>, the key you wish to be returned.
6. Press <ctrld>.
7. Press <alt-> to end macro definition.

Now when you press <altk>, <ctrlk>, not "<ctrlj> <ctrlq>", plus <ctrld> will be returned. The playback bypass key only affects the next key entered. To prevent playback of more than one key, it is necessary to press the playback bypass key before entering each of them.

3.17 Pop-up Newkey Menu

macro function: {cmd}*input to Newkey*{endcmd}

This will pop-up the Newkey menu and feed the *input to Newkey* string to Newkey. See section 6.7, "Invoking Newkey From its Own Macros", for further information.

3.18 Print Screen

macro function: {*prtscrn*}

This will invoke your computer's "print screen" function.

3.19 Return to Calling Macro

macro function: {return}

This will cause the current macro to stop and return control to the macro which called it. If the current macro was not called by another macro then it will stop as if it had terminated normally. This is used most often in conjunction with the if screen functions.

3.20 Send to Printer

Macro function: {begprint}*output to printer*{endprint}

This will send all characters enclosed between the {begprint} and {endprint} functions to the designated parallel printer. The printer defaults to LPT1, but may be changed by using the {set printer *number*} option.

HINT: To send control codes (x'01'-x'26') use <ctrla> - <ctrlz>.

3.21 Restore Screen

Macro function: {restore}

This function will restore the portion of your screen that was just used by a display macro. Usually this function will be entered for you by the display macro editor.

3.22 Save Macros

Macro function: {begsave}*filename.ext*{endsave}

The {begsave} function will save the macros from memory into *filename.ext*. It will be ignored in the small version, Newkeysm, and the very small version, Newkeyvs.

3.23 Screen off/on

Macro function: {scroff}
{scron}

This function will cause your screen to be turned off/on in the same way as the screen blanker works. To turn the screen back on just press any key a couple of times.

3.24 Set Option

Macro function: {set *option value*}

CHAPTER 3
MACRO FUNCTIONS

This macro function can be used to set any of the values that can be set on the "Parameters" screen. For further information on these parameters refer to section 10.2, "Parameters". Valid options are:

Name	Values	Description
alphamsg	on off	turn alphabetic warning message on/off
beepdef	on off	turn beep while defining on/off
blackwht	on off	turn black and white mode on/off
blnkwait	1-99	set video blank wait time to 1-99 minutes
capsctrl	on off	turn enhanced keyboard caps/ctrl switch on/off
click	on off	turn key click on/off
curschg	on off	turn cursor change on/off
defmsg	on off	turn macro defined message on/off
descmsg	on off	turn description request on/off
diskwait	on off	turn disk wait on/off
displock	on off	turn display num/caps lock display on/off
extbufer	on off	turn extended buffer on/off
fastdlay	1-18	set fast key delay to 1-18
fastkey	on off	turn fast key mode on/off
fastrate	1-99	set fast key rate to 1-99
newkey	on off	turn Newkey on/off
overwrit	on off	turn load/save file overwrite warning on/off
printer	1-3	set printer to lpt1, lpt2, lpt3
scrblank	on off	turn screen blanker on/off
shiftsta	on off	turn shift status reset on/off
shortcap	on off	turn shorthand capitalization on/off
shrthand	on off	turn shorthand mode on/off
singstep	on off	turn single step mode on/off
slowrate	1-255	set slow typing rate to 1-255
slowtype	on off	turn slow typing mode on/off
statline	on off	turn status line display on/off
statpos	0-24	set status line position
videoio	on off	turn video i/o restore screen on/off
zerofill	on off	turn zero fill date on/off

The following set options are used in display and menu macros and are usually entered by the display and menu macro editors:

bgcolor	0-7	set background color to 0-7
border	on off	turn border on/off
bordtype	0-3	set border type to 0-3
fgcolor	0-15	set foreground color to 0-15

for example:

{set slowtype on}	turns slow typing mode on
{set slowrate 3}	sets slow typing rate to 3
{set scrblank off}	turns screen blanking off
{set curschg off}	turns off cursor change

3.25 Control Caps, Num, and Scroll Shift States

Macro function:	{capsoff}	{numoff}	{scrloff}
	{capson}	{numon}	{scrlon}
	{capsrest}	{numrest}	{scrlrest}
	{capssave}	{numsave}	{scrlsave}

The functions ending with 'on' and 'off' will turn the caps, num, and scroll shift states on and off. The functions ending with 'save' and 'rest' will save and restore the shift states.

The restore and save functions allow you to preserve the current shift state, switch into the desired state, and when done restore the previous state.

3.26 Slow Typing Mode

Macro function: {slowoff}
{slowon}

These functions will turn slow typing mode on and off. For further information, refer to section 4.6, "Slow Typing Mode".

3.27 Time Delay

Macro function: {wait *MM:SS:HH*}

Macro playback may be delayed for up to 99 minutes, 99 seconds, and 99 hundredths of a second. During playback the cursor will become a fat bar in the middle of the line.

Time delays are used when your application program can not handle keys returned at Newkey's normal rate. In these cases, it is necessary to give the program some breathing room, otherwise it will throw keys away. It can also be used for those situations in which a delay is necessary in order to wait for some external event to occur (a remote site to respond to your communications requests, for example).

3.28 Variable Length Fill-in-the-Blanks

Macro function: {vfld}

Newkey provides you with the ability to define variable length fill-in-the-blanks fields within macros. When one of these is encountered during playback, Newkey will stop playback and wait for the user to enter any keystrokes he wishes. Newkey will continue to accept keystrokes until the <enter> key is pressed, when normal playback will continue. To define a variable length fill-in-the-blanks field follow these steps:

1. Press <ctrl[>. The cursor will change from a full block to half block with its bottom missing and the mode in the status line will change from "TEXT" to "VAR".

2. Enter whatever keystrokes you wish.

3. Press <ctrl[>. The cursor will change back to a full block and the mode in the status line will change from "VAR" to "TEXT".

For example, suppose you frequently started a letter with "My dear John (Frank, Susan, etc.) it is now time", you might wish to define one macro to mean "My dear xxxxxx it is now time". To do this you would follow these steps:

1. Press <alt=> to start macro definition.
2. Press <altl>, the key you wish to define.
3. Press <enter> to bypass the description.
4. Type "My dear ".
5. Press <ctrl[> to start variable length field.
6. Type in any name, although it is not necessary to the definition.
7. Press <ctrl[> to end variable length field.
8. Type in " it is now time".
9. Press <alt-> to end macro definition.

Now if you were to press <altl>, "My dear " would be displayed and Newkey would wait for you to enter a name. Once you had entered a name and pressed the <enter> key, the Newkey would continue with " now is the time".

3.28.1 Ending a Variable Pause Without Actually Typing the Pause End Key

Variable length pauses will end automatically if the last key in a macro is the pause end key (<enter>) and that macro is invoked during the variable length pause. For example:

<altf> is defined as "xxx<vfld>end"

<altg> is defined as "yyyy<enter>"

If you were to press <altf>, "xxx" would be returned, Newkey would enter the variable pause and wait for you to enter text. If you now pressed <altg>, "yyyy" would be returned and since <enter> is the last key in the macro, Newkey will end the variable length pause in <altf> and return "end".

3.29 Wait for key

Macro function: {waitany}
{waitanyk}
{wait4key *keyname*}

The {waitany} function will stop playback and wait for any key to be pressed. The keystroke will be discarded and execution will continue.

The {waitanyk} function is identical to the {waitany} function except that the keystroke will not be discarded. Instead, it will be left in the keyboard's buffer.

The {wait4key *keyname*} function will stop playback and wait for the specified key to be pressed before execution continues. The keystroke will be discarded.

These functions are especially useful in combination with the display macros.

CHAPTER 4
CONTROLLING MACRO EXECUTION

Newkey offers many methods of controlling how and when macros will be executed. You can permanently unload Newkey or just turn if off for the next keystroke. You can slow macro playback to a crawl or let it zip along. These methods and more are discussed below.

4.1 Switching Newkey Off/On

Newkey may be deactivated/activated by pressing <ctrl6>. This acts as a toggle switching Newkey from one status to another. When deactivated, Newkey is still resident in memory, but will not playback any macros.

4.2 Playback Bypass

Macro playback may be bypassed by pressing <ctrl2> followed by the key desired. This is the same procedure used to enter a playback bypass key during macro definition described in section 4.2, "Playback Bypass".

4.3 Inactivated Macros

Sometimes you may wish to turn off a macro for an extended period of time. Rather than continually pressing the playback bypass key described in the previous section it would be easier to temporarily inactivate the key. Newkey now allows you to do this by displaying the key's macro and pressing 'I'.

Inactivated macros will be marked by a 'I' in the status field when displaying the macro directory. To reactivate the macro follow the same steps, but press 'A' instead of 'I' when displaying the macro. Unlike guarded macros, inactivated macros will not be saved to macro files as inactivated macros.

4.4 Cancel Newkey Processing

Newkey processing may be canceled at anytime by pressing the cancel key, <ctrldel>. This feature will immediately cancel macro playback or definition and clear the keyboard buffer. This is especially useful for ending macros which call themselves or long macros using slow typing mode or time delays.

4.5 Unload and Reclaim Memory

Type in "Newkeysp /u" and press <enter>. This function will unload Newkey completely from the system and free up its memory for reuse. Caution must be exercised in using this feature. If another program has been permanently loaded after Newkey, then its storage will also be freed.

This function will not always free up Newkey's storage if it is executed from a batch file (unless it is the last command). Some versions of DOS will not always free the storage, even though they report that they did. If this occurs, try loading and unloading Newkey again. This will sometimes free the old storage as well as the new.

4.6 Slow Typing Mode

Newkey may be operated in a special slow typing mode. This mode is provided because some programs cannot handle the fast rate at which Newkey normally returns keys, with the result that the keys are discarded. By operating in slow typing mode, you can give these programs enough time to complete their tasks before being fed the next key. Macro execution may also be delayed by using the time delay function described in section 3.27, "Time Delay".

Newkey comes with slow typing mode turned off, but this may be changed by using Newkeysp, through the pop-up menus, or from within a macro itself using the {set slowtype on} and {set slowtype off} functions.

4.6.1 Setting Slow Typing Speed

You may change the amount of time Newkey will wait between keystrokes. This value divided by 18 gives you the number of seconds between each keystroke. Legal values are 1 (.05 second) to 255 (14 seconds). Newkey comes with this delay value set to 1. To change select "Parameters" on the Newkeysp menu or Newkey's pop-up menu. The slow typing speed may also be set within a macro by using the {set slowrate xx} macro function.

4.7 Single Step Macro Playback

Macro playback may be put in a special single step mode in which Newkey will wait for you to press any key before returning the next keystroke in your macro back to the calling program. This feature can be very useful in debugging a complicated macro.

Two caveats are in order: first, use of the single step features eliminates certain timing problems that might be experienced if playback were allowed to occur at its normal rate. So if playback in single step mode works correctly, but it doesn't in normal mode you are probably encountering a timing problem which can be solved by inserting time delays or turning on slow typing mode at appropriate spots in your macro. For more information on these refer to section 4.6, "Slow Typing Mode", and section 3.27, "Time Delay".

The second caveat is that you may notice that you must press a key more than once before your program appears to take any action on the first keystroke. This is because some programs (DOS included) will issue a second keyboard read before acting on the first keystroke. If this occurs, just keep pressing keys until you notice some action taking place. Usually it will not take more than one or two more presses.

4.8 Disk Wait During Macro Playback

Newkey will now allow you to delay macro playback while a program is accessing your floppy disk drives. This is important because some programs will throw away keystrokes while your floppy disk drives are spinning.

Newkey comes with disk wait mode turned off, but this may be changed by using Newkeysp, through the "Parameters" screen, or from within a macro itself using the {set diskwait on} and {set diskwait off} functions.

CHAPTER 5
EDITING MACROS

You may display and edit macros at any time, even while you are defining them.

5.1 Display Directory and Descriptions

Select the "Display directory and descriptions" option on the pop-up menu screen. A directory of all macros that have been defined, their status, and one line's worth of their associated descriptions will be displayed. If the macro has not been given a description Newkey will display part of the macro's translation. Possible values in the status field are:

G	Guarded macro
I	Inactive macro

5.2 Displaying a Macro

To display a macro, display the macro directory as described in the previous section and press that macro's key. To see a multi-character macro, display the macro directory, press enter, type in the multi-character macro name you wish to display, and press <enter>.

5.3 Editing a Macro

To edit a macro, display it using one of the methods described in the previous sections and then press 'e' to enter edit mode. Editing is always done in insert mode. With the following exceptions your keystrokes will be translated into their ascii equivalent:

Back tab - move 5 keystrokes to the left

Bksp - delete keystroke at left of cursor position

Cursor keys - move cursor

Del - delete keystroke at current cursor position

F1 - escape editor's command key processing

F2 - insert one of Newkey's special macro functions described in Chapter 3.

F3 - perform one of the following commands

- copy another macro in
- move another macro in
- capture the keyboard buffer

F4 - edit current macro as a display macro

F10 - stop editing. You will be asked whether you want to save your changes.

Pgdn - scroll screen down

Pgup - scroll screen up

Tab - move 5 keystrokes to the right

5.3.1 Escaping Editor Command Key Processing

Sometimes you may want to insert a key that is normally treated as a command to the editor. To do this press <F1> followed by the desired key to "escape" the editor's command key processing. For example:

Pressing	Action taken	Pressing	Action taken
<F1><lft>	inserts <lft>	<lft>	moves cursor
<F1><F2>	inserts <F2>	<F2>	displays menu
<F1><F1>	inserts <F1>	<F1>	starts escape

By pressing <F1> first you are telling the editor not to do anything special with the next key, just insert it into the macro being edited.

5.3.2 Copying/Moving a Macro

Start editing the target macro and move the cursor to the point at which the selected macro is to be copied/moved in. Press F3 to bring up the special functions screen. Select the "copy outside macro" or "move outside macro" option. You will be prompted for the macro to be copied/moved.

5.3.3 Copying the Keyboard Buffer

Follow the same steps as described in section 5.3.2 but select the "copy keyboard buffer" option instead. This option will copy the last 128 keystrokes typed into the macro being edited.

5.4 Deleting a Macro

Macros may be deleted in two ways:

1. Undefine the macro.

    a. Start the normal macro definition process (<alt=>).
    b. Press the key you wish to undefine.
    c. Press <enter> to skip the description request.
    d. Press <alt-> to end macro definition.

2. Display the macro using the "Display Macro" option and press "d" to delete the macro.

5.5 Editing Using a Text Editor

You may also edit a saved macro file using a text editor or word-processor capable of editing ascii files. This is a slightly more error prone process than using Newkey's built-in macro editor, but if you follow the following rules you should not have any problems.

1. All macro definitions must begin with "{begdef *macroname*}" and end with "{enddef}".

2. All single characters may be placed in the macro without the surrounding carats.

3. To define a macro within a macro, enter a "{begdef *macroname*}" followed by the key you wish to define. Do not end it with an "{enddef}".

4. Descriptions must be in the lines preceding the macro's {begdef} and must begin with a "*". There is no limit on a description's length.

All non-single character keys may be in upper or lower case. All characters in between macro definitions will be ignored. This allows you to document the macro definitions if desired.

CHAPTER 5
EDITING MACROS

CHAPTER 6
ADVANCED MACRO FEATURES

Many of these features are illustrated in the sample macro file, example.key and the guided tour (see section 1.3, "Guided Tour").

6.1 Defining a Multi-Character Macro

Newkey allows you to define macros with names up to 8 characters in length. This allows you to assign meaningful names to your macros, names that you will remember. For example, you might have a macro named "sales" that will generate your sales report or a macro called "address" that will take the address out of a letter in your word processor and print it on an envelope.

To define a multi-character macro, follow the same steps as described in "Defining a Macro", except at the point where you would normally enter the key to be defined press <enter> instead. Then type the name you have chosen and press <enter> when you are done. Macro definition will then continue as normal.

Macro names must be between 2-8 characters in length and can be any combination of alphanumeric and some special characters.

6.2 Playing back a Multi-Character Macro

To playback a multi-character macro, press <alt,>. A window will open up asking you to enter the name of the multi-character macro you wish to playback. Once you have keyed in the name, press <enter> and playback will occur just as it would if you had pressed a normal macro key.

If you request a multi-character playback while defining another macro and you do not have a macro defined with that name, the following message will be displayed:

"Macro not found - save in current definition? (y/n)"

If you wish to save the name in the macro definition even though it does not yet exist, press "y". This option allows you to define multi-character macros which call themselves or which you plan to define later.

Except for the methods of defining and playing back, multi-character macros will function just as regular macros do.

6.3 Shorthand Mode

An alternate method of playing back multi-character macros is provided through Newkey's shorthand mode. In shorthand mode Newkey watches your keystrokes as you type and when they match the name of a multi-character macro, replaces them with the multi-character macro. For example, in preparing this manual I defined a multi-character macro called "ne" to be "Newkey". Then, whenever I typed "ne", Newkey was substituted for the "ne".

Newkey will not recognize a macro name until you have completely typed it in and then pressed a key which cannot be part of a macro name to delimit it. This can be any key except for the standard alphanumeric character set. This ensures that Newkey will not interpret an imbedded character string as a macro. The key that you press to end the macro should be the key that you would normally press after the macro had executed. Newkey will save the key and return it after the macro has finished.

If you do not wish the delimiter key to be returned after the macro has finished then you may press the shorthand delimiter key. This key defaults to '\', but may be changed by selecting "control Keys" on the main menu.

For example, if "ne" is defined as "Newkey":

Typing	Yields
plane	- no playback
nest	- no playback
honest	- no playback
ne<space>	- "Newkey "
ne is the best	- "Newkey is the best"
I love ne.	- "I love Newkey."
ne\sp is useful.	- "Newkeysp is useful"

Be careful in choosing macro names. It can be very mysterious to be typing along and all of a sudden have your computer take off on its own when you least expect it.

6.3.1 Capitals in Shorthand Mode

Multi-character macro names are case insensitive. They are always translated to lower case no matter how you enter them. This poses a potential problem in shorthand mode since in many cases you might not want the upper case macro name to cause macro playback. For example, I might want to type "NE" for Nebraska rather than for "Newkey", yet if Newkey automatically translated "NE" to lower case and then executed the "ne" macro I would be frustrated in my intention. True, I could always press <ctrl2>, the playback bypass key, but this would soon become very tiresome.

To help alleviate this situation, Newkey offers the ability to control whether macro names with any capitalized letters should be treated as real macros or ignored. To turn this on/off select the "Upper case short hand" parameter on the "Parameters" screen described in section 10.2.

There is one exception to Newkey's capitalization rules. If you capitalize the first character in the multi-character name Newkey will capitalize the first character in your macro. This means that you can start a sentence with the macro and have the first word capitalized. This feature will only work if the "Upper case short hand" parameter is off and only the first character in the macro name is capitalized.

6.3.2 Controlling Shorthand Mode

You may control shorthand mode as follows:

1. Turning shorthand mode off/on.
2. Turning Newkey off/on.
3. Pressing <ctrl2>, the playback bypass key either just before, during, or just after typing the macro will cause the macro playback to be ignored. For example,

Typing	Yields
<ctrl2>ne	ne
n<ctrl2>e	ne
ne<ctrl2>	ne

4. Inactivating the macro will suppress it until you reactivate it again.

6.3.3 Shorthand Mode Backspacing

When Newkey recognizes a multi-character macro it causes your program to backspace back over the macro name by feeding it the appropriate number of backspace keys required to erase the macro name and then feeding it the contents of the macro.

For most programs the regular backspace key works well enough to erase the macro name, but if it does not for your program Newkey offers you the ability to change the key that will be used for backspacing. Just follow the instructions in section 10.2, "Parameters", to update the "shorthand backspace" key. You can even assign a macro to the key you select.

6.3.4 Shorthand Mode Restrictions

Shorthand mode will not work when you are:

1. Defining a macro.
2. Playing back a macro.
3. Newkey is inactive.
4. Shorthand mode is turned off.
5. Any part of the macro name is entered in upper-case and the shorthand capitalization feature is turned off.

If you want to invoke a multi-character macro during definition or playback you should use the standard method of invoking them described in Section 6.2, "Playing back a multi-character macro".

6.4 Nested Macro Playback

Newkey will translate macros within macros down to 8 levels. For example, assume the following macros have been defined as follows:

<alt1> = level 1 <alt2>
<alt2> = level 2 <alt3>
<alt3> = level 3 <alt4>
<alt4> = level 4 <alt5>
<alt5> = level 5 <alt6>
<alt6> = level 6 <alt7>
<alt7> = level 7 <alt8>
<alt8> = level 8 <alt9>
<alt9> = level 9 <alt0>

When <alt1> is pressed, Newkey will check each character returned ("L", "e", "v", "e ", "l"," ","1","<alt2>") to see if they too have been defined and if so, will return their translation. Newkey will find that <alt2> has been defined and will check its definition for further translation to a depth of 8 levels. At the eighth level, Newkey stops checking and returns that translation without further checking. In this case, Newkey will return "level 1 level 2 level 3 level 4 level 5 level 6 level 7 level 8 <alt9>".

Try it out. Define <alta> as "calling", <altb>. Now define <altb> as " altb". Press <alta>.

6.4.1 Recursion

If at any time Newkey detects a recursive or circular macro definition which would result in an infinite playback loop (a=b, b=c, c=a for example), it will display the following message in reverse video:

RECURSION DETECTED - PRESS ESC TO CONTINUE

Press the ESC key to continue. The key for which recursion was detected will be returned untranslated.

6.5 Defining Macros From Within Macros

To define a macro from within a macro, just press the <alt=> key and Newkey will start defining a new macro just as if you had not been already been defining a macro. When you are done defining the new macro, end it as you would normally (<alt->), and you will take up where you left off defining the original macro.

Now when you playback the second macro, Newkey will return what you just defined it to. If you playback the original macro, it will stop at the point where you requested it to begin defining the second macro and start a new definition for the second macro. This definition will continue until you end it (<alt->), just as you would if you had started it yourself.

The only difference between defining a macro from within another macro's playback versus defining it in the normal manner is that the Newkey window and warning messages will not be issued if the definition is coming from within a playback.

Try out the following example:

1. Press <alt=>. (start <alta> definition)
2. Press <alta>.
3. Press <enter> to skip description.
4. Type "alta defining altb ".
5. Press <alt=>. (start <altb> definition)
6. Press <altb>.
7. Press <enter> to skip description.
8. Type "altb here!"
9. Press <alt->. (end <altb> definition)
10. Type "- altb now defined"
11. Press <alt->. (end <alta> definition)

Now press <altb>. "altb here!" should be returned. Press <alta>. "alta defining altb " will be returned and Newkey's macro definition mode (cursor change, status line) activated. Type "new altb definition" followed by <alt->. Now press <altb>.

6.6 Macro May Call Itself

In certain special circumstances, a macro can call itself. In order for this to occur, the macro must call itself as the last thing it does before ending. To invoke this feature, follow these steps:

1. Start defining your macro and when you reach the end, press the key you are defining.

2. Newkey will display a recursion detected message. Press <esc> to exit this message.

3. End the defining process as you would normally (<alt->).

Now the recursive macro is ready to go. Just press the key you defined and watch it go. To stop the playback press the Newkey cancel key, <ctrldel>.

Warning

Be careful how you use this feature. If you create a macro without a pause, time delay, or using slow typing mode, you may find that you have created a monster. Newkey will feed keys to your program at a rate beyond your ability to control. You can always stop the playback using the cancel key, but your reaction time will probably not be fast enough to stop at the point you wish.

For this reason, we recommend that all recursive macros include a pause or turn on slow typing mode for a portion of the macro (Most of the time you will probably want to do this anyway.)

6.7 Invoking Newkey From its Own Macros

Subject to the limitations discussed below in section 6.7.1, "Scope of Macro Definition and Playback", Newkey and Newkeysp can be called from Newkey's own macros.

To invoke Newkey's pop-up menu features follow these steps:

1. Start your macro definition.
2. Call up the pop-up menu.
3. Toggle the "Record mode" option on.

Now your keystrokes will be captured in the macro.

6.7.1 Scope of Macro Definition and Playback

Except for rare compatibility problems with other software (see section 16.9, "Software Compatibility" and the readme file on the Newkey disk) Newkey's macros can be used with any other application or program, even to do Newkey functions themselves. Although most of the pop-up functions can be done through special purpose macro functions, there are still occasions where it would be useful to invoke Newkey using its own macros. For example, a macro could be created that would pop-up the cut function with one keystroke.

6.8 Cut and Paste

Newkey allows you to "cut" information off of your current screen and then "paste" it into another application. For example, you could cut a cost/benefits analysis from your spreadsheet and paste it into a report you are preparing on your word-processor.

6.8.1 Cutting

To cut a piece of your screen, invoke the pop-up features and select the "cut" option. Your screen will be redisplayed. Move the cursor to the point where you wish to start cutting and press the <home> key to start marking.

Now move the cursor to the opposite end of the screen block you wish to cut and press the <end> key. Newkey will save the marked screen block in the macro <ctrlins> and return you to the main pop-up menu.

You may change the key the cutting is assigned to by pressing <F1>. You will be prompted for a new macro name.

6.8.2 Pasting

Once you have successfully cut a part of the screen, you may paste it into another application by playing back the macro to which the screen was cut. Unless otherwise specified, <ctrlins> will receive the cutting.

A cutting can use up to 2k of your macro buffer, so you may find it useful to delete the <ctrlins> macro once you are done with it (see section 5.4, "Deleting a Macro").

6.8.3 Cut and Paste Line End Key

When you cut from the screen, a line end key is appended to each line. This line end key defaults to <enter> and identifies the end of the line to the program into which the cutting is pasted. Usually this is best for word processors and text editors, but for some other programs a different line end key may work better. In these cases, the line end key may be changed by selecting the "control Keys" on the pop-up menu and updating the "cut and paste line end key".

CHAPTER 7
DISPLAY MACROS

Display macros will open up a window on your screen and display the text you have defined for the window and wait for the user to take some specified action, such as hitting the enter key or filling in an input field.

For example, <altw> may be defined as an display macro which prompts for a file name and then starts your word-processor with the file name you just entered. When you press <altw>, a display window will pop up looking something like this

and Newkey will wait for you to type a file name. When you are finished press <enter> and the file name will be assigned to the input macro specified and inserted into the macro you are running. The macro could then continue by starting up your word-processor and retrieving the file you just specified.

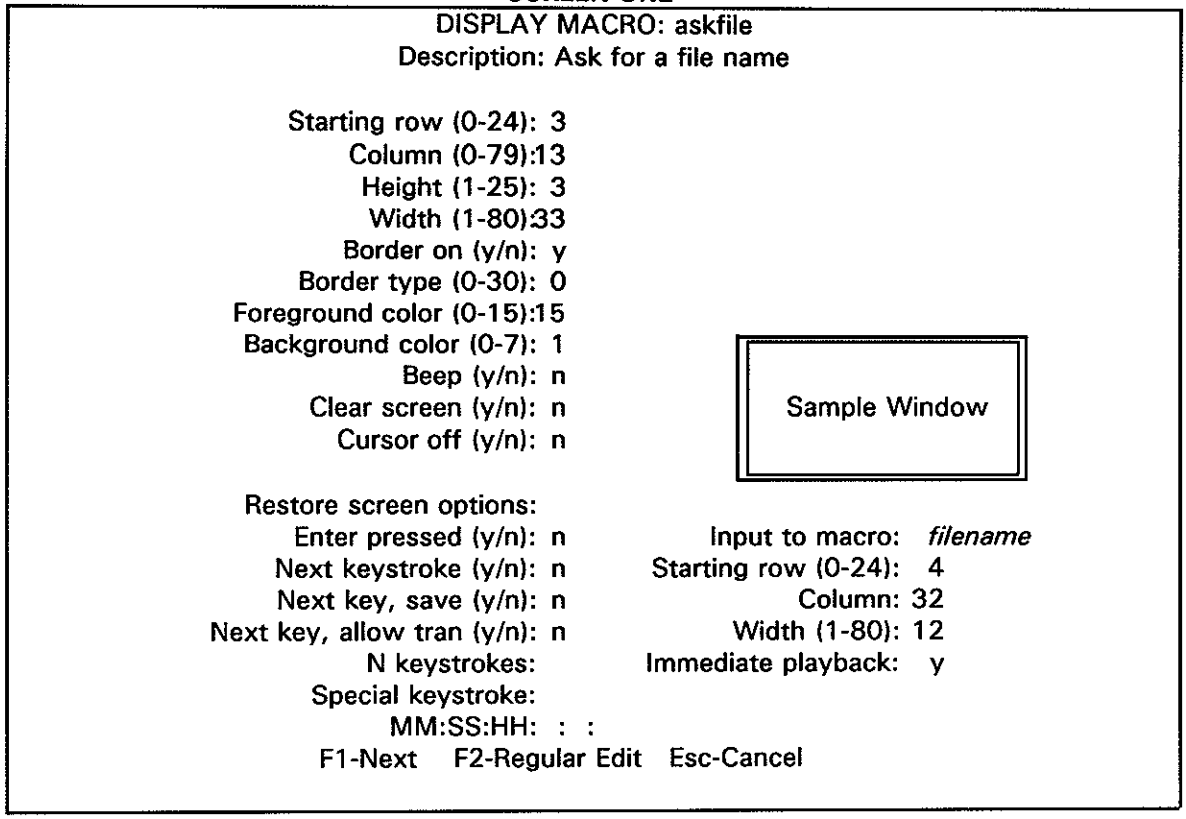
7.1 Creating and Editing a Display Macro

Editing a display macro is a two step process. The first step displays a menu and requests information, the second step displays the window, allows you to enter the text to be displayed, optionally move or resize the window, and define the input field. As you edit, you may toggle between steps one and two by pressing <F1>.

To create or edit a display macro, select the "Display/edit macros" option on the pop-up menu. This will display a list of currently defined macros. Press the key of the macro you wish to define or <enter> for a multi-character macro. The macro's definition will be displayed. Now press 'f' to invoke the display macro editor. A typical screen one follows.

CHAPTER 7
DISPLAY MACROS

SCREEN ONE



Many of the fields will be filled in with default values or the values from the macro you are editing. These are the fields to the right of the ':'s and filled in with italicized characters. In the screen above, a display macro is being defined to pop-up a window and ask for a file name.

Window definition options:

These describe how the window will look when it pops up.

Field name	Value	Description
Description	text	macro's description
Starting row	0-24	top row in which to start window
Starting column	0-79	left most column to start window
Foreground color	0-15	see appendix E for list of values
Background color	0-7	see appendix E for list of values
Height	1-25	number of lines in window
Width	1-80	number of columns in window
Border on	y/n	display border
Border type	0-3	border type of window 0 = double line 1 = single line 2 = reverse video double line 3 = reverse video single line

Beep	y/n	beep when window displayed
Clear screen	y/n	clear screen when window displayed
Cursor off	y/n	turn cursor off when window displayed

Restore screen options:

The following options signal when the screen should be restored.
Only one option should be chosen.

Enter pressed	y/n	wait until <enter> is pressed
Next keystroke	y/n	wait until the next key is pressed
Next key, save	y/n	wait until the next key is pressed and leave the keystroke in the buffer
Next key, allow tran	y/n	wait until the next key is pressed and if the key has a translation associated with it, playback the translation
N keystrokes	0-99	wait for specified number of keystrokes
MM:SS:HH		wait for the specified number of minutes, seconds, and hundredths of a second to elapse

These next options apply only if "input to macro" is selected. The "input to macro" option will cause Newkey to wait for the user to enter some input into the defined field. This input will then be assigned to the specified macro. The input field must fit within the window and can only be one line in height.

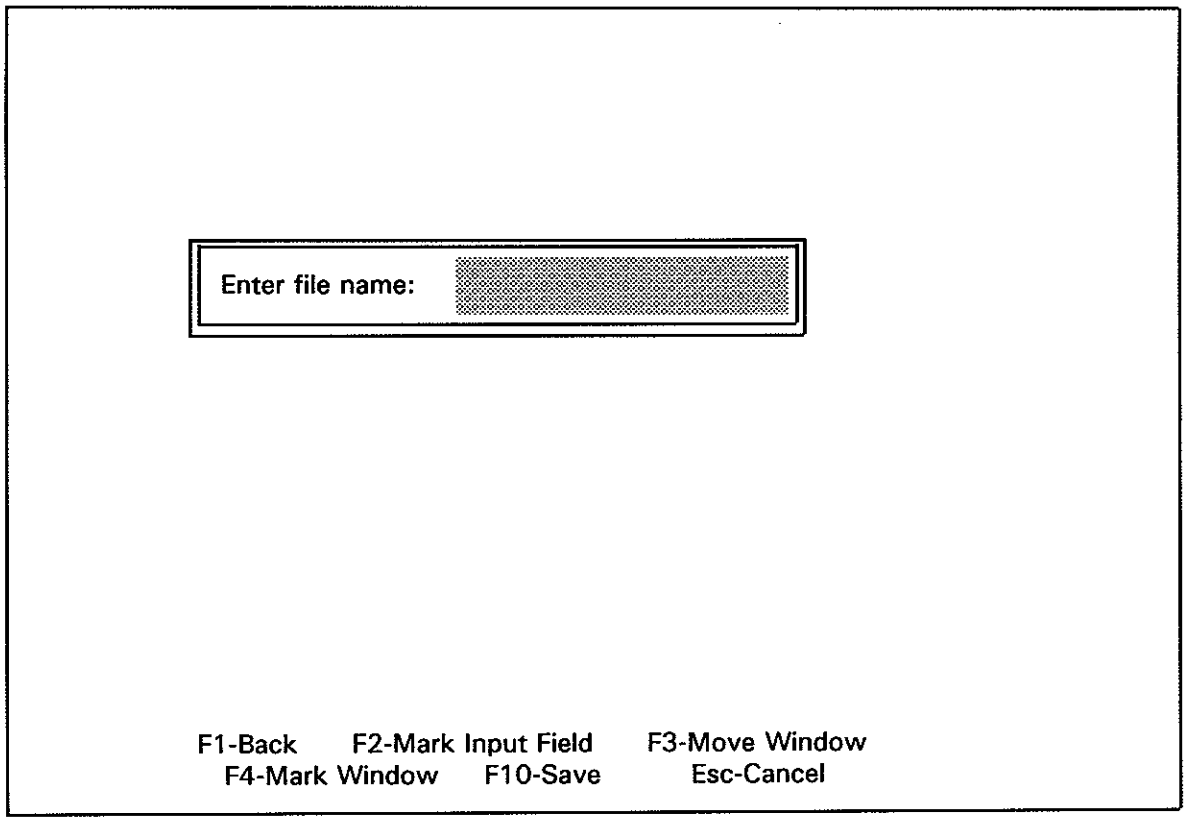
Input to macro	macro	macro to receive user input, press key to receive input or type multi-character macro name
Starting row	0-24	top row in which to start input row
Starting column	0-79	left most column to start input field
Width	1-80	number of columns in input field
Immediate playback	y/n	playback input macro immediately after the screen is restored

The following keys will cause these actions:

F1-Next: move to screen two
F2-Regular Edit: edit the macro as if it were a regular macro
<ESC>-cancel: cancel editing

The sample window displayed on the right will display the options of the window you are defining.

SCREEN TWO



Screen 2 will display the window you defined in screen 1 and allow you to enter the text to be displayed. Just type the text where you would like to see it displayed. In the screen displayed above a message requesting a file name has been typed in. The shaded area identifies the input field which will be used for the filename. While on screen 2 you can take the following actions:

F1-Back: Goto screen one

F2-Mark input field: Use the cursor to set the starting row, starting column, and width of the "input to macro" field. If an "input to macro" field was not defined on screen 1, you will be prompted for the macro to be used.

F3-Move Window: Use the cursor to move the window to a different position on the screen.

F4-Mark Window: Use the cursor to change the window's starting row, column, height, and width.

F10-Save: Save the edited macro.

<ESC>-Cancel: Cancel editing the macro.

7.2 Suggestions on Creating and Editing

When initially creating a display macro, you may find it easiest to immediately move to screen 2 and then use the "mark window" and "mark input field" options to build the window. This will allow you to see exactly where the window and input field will be displayed on the screen. Now enter the window's text. When you are finished go back to screen one by selecting "back" and select the other window options desired. When you are done, go to screen 2 and check the window. When you are satisfied with its appearance select "save" on screen 2.

CHAPTER 7
DISPLAY MACROS

CHAPTER 8
MENU MACROS

Menu macros will open a window and display a list of selections. Each selection corresponds to a macro. When a selection is made, the associated macro is played back. These macros can be used to develop anything from a simpler interface for your software to a full customized menu interface.

8.1 Creating and Editing a Menu Macro

To create or edit a menu macro, select the "Display/edit macros" option on the pop-up menu. This will display a list of currently defined macros. Press the key of the macro you wish to define or <enter> for a multi-character macro. The macro's definition will be displayed. Now press 'm' to invoke the menu macro editor. A screen looking like this will be displayed (italicized fields illustrate typical information as might be entered in a completed screen):

EDITING MENU MACRO: *<alt0>*

Description: *Sample Menu Macro*

Title: *Sample Menu Macro for Dos command line*

	Macros	Exit	Macros	Exit
1	*alt1*	n	11	n
2	*alt2*	n	12	n
3	*alt3*	n	13	n
4	*alt4*	n	14	n
5	*alt5*	n	15	n
6	*alt6*	n	16	n
7	*alt7*	n	17	n
8		n	18	n
9		n	19	n
10		n	20	n

<alt0>	Sample Menu Macro
<alt1>	list ^ Directory
<alt2>	start ^ Wordperfect
<alt3>	start ^ Lotus 123
<alt4>	^ Erase a file
<alt5>	^ Copy a file
<alt6>	start ^ PC-File+
<askfile>	Ask for a file name

Footer: *Select option, ESC to cancel*

Starting row (0-24): *6*
Column (0-79): *17*
Foreground color (0-15): *15*
Border on (y/n): *y*
Border type (0-3): *0*

Beep (y/n): *n*
Clear screen (y/n): *n*
Cursor off (y/n): *y*
Exit key: *esc*

PGDN PGUP F10-Save Esc-cancel

On the right hand side of the screen is a window displaying the macros eligible for inclusion in the menu (sample macros included for illustration). To be eligible a macro must have a description, because when the menu is displayed, the description is used as the menu choice. Although macros without descriptions can be put in the selection list, they will not be displayed when the menu macro is played back. The macro list may be scrolled up and down by the PGUP and PGDN keys.

The following fields may be entered:

Field name	Value	Description
Description	text	macro's description
Title	text	title displayed at top of menu
Macros	macro	macro to be on selection list, press key or type multi-character macro name, up to 20 macros are allowed
Exit	y/n	exit menu macro after playback
Footer	text	footer displayed at bottom of menu
Starting row	0-24	top row in which to start menu
Starting column	0-79	left most column to start menu
Height	1-25	number of lines in window
Width	1-80	number of columns in window
Foreground color	0-15	see appendix E for list of values
Background color	0-7	see appendix E for list of values
Border on	y/n	display border around menu
Border type	0-3	Border type of window 0 = double line 1 = single line 2 = reverse video double line 3 = reverse video single line
Beep	y/n	beep when menu displayed
Clear screen	y/n	clear screen when menu displayed
Cursor off	y/n	turn cursor off when menu displayed
Exit key	key	keystroke that will cause menu to exit

8.2 Preparing to Define a Menu Macro

Creating a menu macro will be made easier if some initial preparation has already been done. First, identify the menu entries to be included and define a macro for each entry. At this point it is only really necessary to create a description for each macro, the body can be defined later. Keep in mind that the description will be used as the menu selection when the menu macro is played back.

When the description is entered you may also specify a select letter by putting a '^' in front of the letter. Later, when the menu macro is played back, the select letter can be used to choose the menu selection as well as the moving bar. Although not necessary, it will be easier to identify the select letter if it is capitalized or is the first letter in the description.

8.3 How Menu Macros Work

When a menu macro is played back, a window is built with the title at the top, the footer at the bottom, and the macro descriptions in between. At this point Newkey waits for a selection to be made or the exit key to be pressed. A selection may be made by moving the bar to the desired choice and pressing enter or by pressing the macro's select letter. When a selection is made, the associated macro will be invoked. If the exit option was specified for this macro control will not be returned to the menu macro.

For a demonstration of menu macros see the "menu.key" file on the Newkey disk.

8.4 Hints

If you wish to do more in a menu macro than allowed by the menu macro editor, then define another macro to contain the menu macro and put the extra macro functions before and after the invocation of the menu macro as desired. For example:

{begdef alta}Extra stuff before<menumac>extra stuff after{enddef}

This will allow you to edit the menu macro, <menumac>, without losing any of the extra stuff.

CHAPTER 8
MENU MACROS

CHAPTER 9
KEYBOARD AND SCREEN FEATURES

9.1 Screen Saver Mode

The screen saver feature will blank your screen after five minutes of inactivity. The screen will be restored whenever you press a key or your program writes to the screen. This feature protects your video terminal from "burn in" caused by prolonged display of the same text over a long period of time. You do not have to worry about turning down your video when you leave your PC.

The screen saver may be turned off by changing the "Blank screen" option. This option and all of the other Screen saver options described below may be changed by selecting the "Parameters" option from the pop-up menu.

If you are have trouble restoring the screen, try pressing the alt, ctrl, left shift, or right shift keys.

9.1.1 Blank Delay Time

The screen saver comes set up to wait 5 minutes before blanking the screen, but you can change this to be any time between 1-99 minutes by changing the "Blank screen delay" option.

9.1.2 Video I/O Restore Screen

Normally any type of video i/o will cause Newkey to restore your blanked screen. But some programs are constantly writing to your screen, even when they are not writing anything new. This keeps the screen saver from working when you would usually want it to. If you use programs like these, you should try turning off the "Video i/o restore screen" option. Some versions of PC-Write and Word Perfect do this.

If you are unsure about whether your program falls into this category, start it up and then let it sit for more than 5 minutes (or whatever you have set the blank delay time to). If the screen does not blank, turn the "Video i/o restore screen" option off and repeat the process. If the screen still does not blank you may have an incompatible video board.

9.2 Extended Keyboard Buffer

This option increases the size of DOS's type-ahead buffer from 16 to 128 keystrokes. Although it has been designed for maximum compatibility, it's possible that it may cause problems when run with other software, especially other memory-resident programs. If you are having trouble using Newkey, try turning the "Extended Buffer" option off. This will cause Newkey to use DOS's normal buffer. This option may be found by selecting the "Parameters" on the pop-up menu.

9.3 Fast Key Parameters

Newkey will allow you to dramatically increase the speed at which your keyboard will repeat a key which is pressed and then held down. You may set both the delay before repetition will begin and the rate at which repetition will occur.

To activate this feature select the "Fast key" option
on the "Parameters" menu and set the rate and delay parameters described below. The best way to choose settings for these options is to try various combinations out.

Newkey helps you control this option by monitoring the state of your keyboard buffer and refusing to insert any additional keys if your program has not already dealt with the previous ones. This helps to prevent the overrun which might otherwise occur as your keyboard buffer fills up with keystrokes that

your program cannot process. This means that when you release the key, your program will not receive any more unwanted keystrokes.

The method Newkey uses to prevent overrun does not work with all programs. If you find that key repetition is actually slower, try the "Alternate fast key mode" described in section 9.3.3.

9.3.1 Fast Key Delay

The time Newkey will wait before beginning repetition may be set to be between 1-18 in 1/18 second intervals. To do this select the "Fast key delay" option on the "Parameters" menu.

9.3.2 Fast Key Rate

The rate at which Newkey will repeat may be set to between 1 - 99 characters per second. To do this select the "Fast key rate" option on the "Parameters" menu.

9.3.3 Alternate Fast Key Mode

Newkey uses a special method to prevent overrun. While this method works for most programs, such as 1-2-3, in a few cases, such as Wordstar 4.0, it will actually slow up repetition. To take care of this situation, a new parameter has been added to the "Parameters" screen. This new parameter, "Alternate fast key mode", changes the way the fast key option works so that it will work with programs such as Wordstar 4.0.

If you are unsure which mode works best with a specific program, try each mode out and pick the one which works the best.

9.4 Keyboard Click

Newkey now offers an optional keyboard feature that will cause a click each time you press a key. To turn it off/on, select the "keyboard click" option on the "Parameters" screen.

9.5 Cap/Num Lock Indicators

Newkey will display the status of your Caps and Num lock keys in the top right hand corner of your screen. To turn this feature off select the "Cap/Num Lock Indicators" option on the "Parameters" menu.

9.6 Caps/ctrl Key Switch on Enhanced Keyboards

In order to accommodate the many requests we have received to switch the caps and ctrl keys on the enhanced keyboard, a "switch caps/ctrl" option has been added. To invoke this option, select the "switch caps/ctrl" option the "Parameters" screen.

CHAPTER 10
CUSTOMIZING NEWKEY

Virtually every aspect of Newkey's operation can be customized to accommodate your own unique hardware and software environment. This chapter describes how to change Newkey's "hot" keys and parameters.

10.1 Control Keys

The Newkey control keys are those keys which have a special meaning to Newkey and are used to control Newkey's operation. These keys are:

Key	Description
<alt=>	Begin macro definition
<alt->	End macro definition
<alt,>	Playback multi-character macro
<ctrldel>	Cancel Newkey processing
<ctrlh>	Backspace and delete during macro definition
<ctrl]>	Begin/end fixed length pause
<ctrl[>	Begin/end variable length pause
<alt/>	Pop-up features request
<ctrl2>	Do not translate next key
<bks>	Shorthand backspace & correct
<\>	Shorthand delimiter key
<ctrl\>	Edit macro being defined
<ctrl6>	Newkey on/off toggle
<enter>	End variable length pause
<ctrlins>	Default key to cut to
<enter>	Cut & paste line end key

You have complete control over which keys will be used for these functions. You can even deactivate them completely if you wish! To change them, select the "control Keys" option on the menu screen, or type in "Newkeysp /x" and press <enter>. This will display the control keys and allow you to change or completely deactivate them if desired. Any changes will take effect immediately. You will then have the option of making them permanent by updating the Newkey.exe program.

10.2 Parameters

Newkey operates in several modes which are controlled by various parameters. These settings may be customized to meet your needs. These modes and their parameters are:

Options
- Newkey status
- Shorthand
- Upper case short hand
- Slow typing
- Slow typing delay
- Disk wait
- Single Step
- Shift Status Reset
- Black & White
- Fast key
- Fast key delay
- Fast key rate
- Alternate fast key mode

Messages
- Macro already defined
- Defining alpha character
- Enter description request
- Defining status line
- Status line position
- Beep while defining
- Change cursor shape
- Load/save file overwrite
- Cap/Num lock indicators

Other options
- Blank screen
- Video i/o restore screen
- Blank screen delay

- Keyboard Click
- Zero fill date
- Switch caps/ctrl

To change them, select the "Parameters" option on the menu screen. This will display the parameters. To make a change select the number/letter of the parameter you want to change and press it. If it is a mode parameter, the status will be toggled on/off. If it is a number, the cursor will move to the selected field and the new value can be entered.

These options may also be set within a macro by using the {set} option.

10.3 Black and White Mode

The colors Newkey normally uses were chosen for their compatibility with monitors (amber, for example) which use the graphics card, but display in one color. But, just in case the colors do not display on your monitor, a new option has been added to Newkey. To invoke this option, load newkey and then type in the following:

newkeysp/b

and press <enter>. This will cause Newkey to toggle its display mode into black & white. Entering the command again will cause the black & white mode to be toggled off. To make these changes permanent, select the "Parameters" option on the pop-up screen.

An alternative method is to use Newkeysp's color customization described in section 10.6, "Customizing Colors".

10.4 Customizing Macro Definition Messages

Much of the macro definition processing (the presence of the window, warning messages, the description request, and status line) can be customized to suit your preferences. If you do not need these aids or protection you can eliminate them and the additional keystrokes they require. These options are described below.

10.4.1 Macro Already Defined Warning Switch

Newkey will warn you when you are about to define a macro
which has already been defined. You will be requested to confirm that you wish to reuse this key for a new macro.
Newkey comes with this option turned on, but you may change it by selecting the "Macro already defined" option on the "Parameters" menu.

10.4.2 Defining Alpha Character Warning Switch

Newkey will warn you when you are about to define a normal
letter (a-z) or number (0-9). You will be requested to confirm that you wish to actually use this key for a macro.

Newkey comes with this option turned on, but you may change it by selecting the "Defining alpha character" option on the "Parameters" menu.

10.4.3 Enter Description Request Switch

You have the option to turn off the "Description =" request in the macro definition window. When turned off, Newkey will automatically close the window after you have entered the key to be defined.

Newkey is designed to save you keystrokes and even that one additional keystroke at macro definition time can be annoying to some people. To turn this off, select the "Enter description request" option on the "Parameters" menu.

10.4.4 Status Line Position

The macro definition status line's position may be moved to any line on the screen by selecting the "Status line position" option on the "Parameters" menu and choosing a number between 0 - 24. Usually you will want to move it to the bottom of the screen, especially when using a program such as Lotus 123 which uses the top line as its status line.

10.5 Preventing Key Creation

Newkey creates several new keys which are not normally supported by IBM. For example, pressing shift and enter together will normally return the same keystroke as just pressing the enter key. Newkey, however, returns a new key code, <capsent>, when you press shift and enter together. This allows you to assign a macro to shift enter and extends the number of keys to which you can assign macros. Sometimes this can cause problems because some programs will recognize that an enter key has been struck, check to see if the shift key is being pressed, and if so, treat shift enter as a separate key. But since Newkey has already given the shift enter its own code, the program will never recognize that shift enter has been pressed, leaving you without a way to create a shift enter for the program.

Newkey now offers you the chance to specify a list of ignored keys. Any key specified on this list will be ignored by Newkey when it is creating new key codes causing the key to revert to its original meaning. To put a key on the ignore list follow these steps:

1. Run Newkeysp.
2. Select the "update Ignored keys" option.
3. Select a line number and press the key you wish to ignore. Alternatively, you may select the last option to ignore all of the keys that are created.

When you exit you will be given the option of making the change permanent.

For example, assume that your IBM AT compatible used alt ctrl / to switch back and forth between turbo and normal mode. Since Newkey will normally create a new key code for both <alt/> and <ctrl/> you will need to put both of these keys on the ignored list. This will let your computer recognize the alt ctrl / combination.

Another possible solution in some cases is to just use Newkey to redefine shift enter as enter. This will work in many cases. For a list of key codes created by Newkey refer to appendix C of the User's Guide.

CHAPTER 10
CUSTOMIZING NEWKEY

10.6 Customizing Colors

Newkey now offers you the ability to customize its menu and message colors. To change Newkey's colors follow these steps:

1. Run Newkeysp.
2. Select the "update message cOlors" option.
3. Follow the instructions on the screen.

When you exit you will be given the option of making the change permanent.

CHAPTER 11
LOADING & SAVING MACROS

This chapter describes Newkey's many options that allow you to control the contents of your macro buffer.

11.1 Save File

This function will save your current macros from memory in a disk file, which may be loaded later. If you are about to overwrite an existing file Newkey will warn you and request a confirmation.

Select the "Save file" option on the menu screen and another screen will be displayed requesting a file name.

To execute in batch mode use the following command:

    newkeysp filename.ext/s

where "filename.ext" is the name of the macro file in which you wish to save your current macros. The {begsave} macro function may be used to save from within a macro being played back.

11.2 Load File

This function will load macros from a disk file into memory where they can be used by Newkey. Any previous macros in memory will be erased (unless they have been "guarded" by Newkey's guard function). If you have created any new macros, Newkey will warn you and request confirmation before continuing.

Select the "Load file" option on the menu screen. Another screen will be displayed requesting a file name.

To execute in batch mode use the following command:

    newkeysp filename.ext/l

where "filename.ext" is the name of the macro file in from which you wish to load a new set of macros. The {begload} function may be used to load from within a macro being played back.

11.3 Merge File

This function will merge macros from a disk file into memory, where they can be used by Newkey. The previous macros in memory will be kept. If the same macro is defined both in memory and in the merge file, the in-memory definition will be retained, unless the overwrite option is specified. If you have created any new macros, Newkey will warn you and request confirmation before continuing.

To merge, select the "Merge file" option on the menu screen. Another screen will be displayed requesting a file name and whether you want to use the overwrite option.

To execute in batch mode use the following command:

newkeysp filename.ext/m - or -
newkeysp filename.ext/m/o

"Filename.ext" is the name of the macro file you wish to merge into memory. The second form, ending in "/o", will invoke the overwrite option. The {begmerge} and {begmergo} functions may be used to merge from within a macro being played back.

CHAPTER 11
LOADING & SAVING MACROS

11.4 Default Path and Extension for Macro Files

Newkey allows you to set a default path and extension for macro files. When a macro file is loaded or saved, the default path and extension will be used to build a full file name if a partial file name is specified. To set the default path and extension select the "options" choice on the Newkeysp menu. Some examples:

Default path:	c:\newkey\macros
Default extension:	key

Loading from or saving to file	Generates
ed	c:\newkey\macros\ed.key
ed.mac	c:\newkey\macros\ed.mac
d:\macros\ed	d:\macros\ed.key
d:\macros\ed.mac	d:\macros\ed.mac

11.5 Clear Macros

This function will clear all current macros from memory.

To invoke it select the "Clear macros" option on the menu screen. To execute in batch mode type in "Newkeysp /c" and press enter.

11.6 Guarded Macros

To protect a macro in your macro buffer from being overwritten when loading a new macro file or from being deleted when clearing macros from your macro buffer you can guard it. This will protect those macros you want to always be present from being accidently deleted. To guard a macro follow these steps:

1. Edit the macro
2. Press F2.
3. Select the guard option.
4. Stop editing.

Guarded macros will be marked by a 'G' in the status field when displaying the macro directory. To unguard a guarded macro just edit it and delete the {guard} function. If at some time you want to remove the guarded macro, display the macro and press 'd' to delete it.

CHAPTER 12
NEWKEYSP OVERVIEW

Newkeysp, the Newkey Support Program, provides many features to complement Newkey functions. You will need to use Newkeysp in the following situations:

- To perform functions from batch files
- To perform functions not available in the small version
- To access customization features only available in Newkeysp

12.1 Invoking Newkeysp

Newkeysp may be invoked in one of two ways:

1. Type Newkeysp and press <enter>. A menu of options will be displayed.

2. Type Newkeysp plus one or more parameters, and Newkeysp will perform the requested functions. This option is particularly useful in a batch file invoked at system initialization. See the file "example.bat" for examples of Newkeysp's batch functions.

12.2 Execute Multiple Newkeysp Commands

Newkeysp will process more than one command when invoked in command mode. For example,

"newkeysp ed.key/l /on /ek=[ctrle]"

might be used to automatically load a newkey macro file, turn newkey on, and then type a key to invoke your editor. The only restrictions on this feature are that only one file oriented command may be executed and it must be first on the command line.

If you should want to execute multiple commands and then enter Newkeysp's menu mode, place "/run" at the end of the command string.

12.3 Newkeysp Batch Commands

To execute Newkeysp in batch mode, type "newkeysp" + one of the parameters below.

Command	Function
/b	Toggle black & white mode
/c	Clear current macros from memory
/d	Display macro directory
/eb	Clear Keyboard buffer
/ek=[*macro*]	Execute *macro* (see section 12.4)
filename/l	Load macros from *filename*
filename/m	Merge macros from *filename*
/off	Turn Newkey off
/on	Turn Newkey on
filename/s	Save macros to *filename*
/t	Toggle Newkey on/off
/u	Unload Newkey and reclaim memory
/x	Display control keys
/zoff	Turn off slow typing mode
/zon	Turn on slow typing mode

For example:

```
newkeysp ws.key/l          - Load Wordstar macro file
newkeysp /off              - Turn off Newkey
```

The file "example.bat" contains a complete set of examples.

12.4 Executing Macros From a Batch File

This feature allows you to enter macros from a batch file just as if you were typing them yourself. To invoke this feature issue the following command:

"newkeysp /ek="

followed by the key you wish to type as described in appendix A.
For example:

"newkeysp /ek=h /ek=e /ek=l /ek=l /ek=o /ek=[altn]"

would type "hello<altn>". If any of these keys has a macro associated with it, that macro will be invoked. "ek" stands for "enter key". Be sure to use "[" and "]" instead of "<" and ">". DOS uses the carats for its own special purposes. Only one multi-character macro may be executed on a command line.

The way this operates can be confusing. Assume that you have a macro defined for <alta>. To execute this macro in a batch file you would place the following statement:

"newkeysp /ek=[alta]"

in the batch file. This will cause the key code associated with the <alta> key to be inserted in the keyboard buffer. Now, the next time the keyboard buffer is read, <alta> will be found and the macro associated with it executed, just as if you had pressed the <alta> key yourself.

The confusing part of this is that *the macro will only execute when the keyboard is read, but the keyboard is not read while DOS is processing a batch file.* What this means is that the macro will not execute until either:

A. Control is returned to the DOS command line, or
B. A program is run in the batch file and the program reads the keyboard.

If you wish to execute a macro to drive a program during a batch file then you should insert the macro into the buffer and then start the program. For example, to start WordPerfect and execute a macro when Wordperfect starts up that will format your letter heading, record a macro (lets call it WPSTART) within WordPerfect to create the letter heading and place the following two statements in your batch file:

```
newkeysp /ek=[wpstart]
wp
```

CHAPTER 12
NEWKEYSP OVERVIEW

Now when the batch file is run, the <wpstart> macro will be inserted into the keyboard buffer and when WordPerfect starts it will read the keyboard and the <wpstart> macro will execute.

CHAPTER 12
NEWKEYSP OVERVIEW

CHAPTER 13
NOTES ON KEYS SUPPORTED

Newkey supports all of the keys normally supported on your keyboard. Many other keys are also supported (For a detailed list of the keys supported refer to appendix A.) These include most keys in ctrl, alt or shift modes, but not combinations of these modes. For example, <ctrlins> is supported, but <alt ctrl ins> is not.

Newkey gives these new keys their own unique scan and ascii codes and treats them just as if they were normal keys. This means that, if they have not been defined, they will be passed back to the requesting program.

This should not cause any problems if the requesting program ignores unknown keys, as most do. But, some programs do not handle strange keys well, and you should be aware of this as a potential problem, especially since you may accidently press one of these keys. See appendix C for a detailed list of these keys and their codes.

13.1 Synonyms - Different Keys, Same Ascii Codes

There are several keys which generate the same ascii code, but with a different scan code. These keys have been separated. Only the original key will be translated and not its synonym. This provides more flexibility in defining macros and eliminates a potential source of confusion. A full listing of synonyms can be found in appendix D.

13.2 Generating Extended Ascii Codes

Normally pressing <alt> plus some number(s) on the numeric keypad will generate the key code equal to the numbers pressed. For example, to generate the <enter> key (ascii code = 13), you would press <alt>, 1, and 3 on the numeric keypad and then release the alt key with the same effect as pressing the <enter> key. But Newkey treats the same sequence as two different keystrokes, <altend> and <altpgdn>.

In order to invoke this function, press the left shift key at the same time as the alt key and hold until you release the alt key.

CHAPTER 13
NOTES ON KEYS SUPPORTED

CHAPTER 14
TIPS ON USING NEWKEY

14.1 Use Your Autoexec.bat File

Put the commands to load Newkey and your macro files in your autoexec.bat file. This will cause Newkey and your macros to be automatically loaded whenever you boot. If you have disks for different purposes, each disk could contain a copy of Newkey, Newkeysp, and your macro files.

An example of the statements you would want to use can be found in LOADNEW.BAT, a batch file supplied with Newkey. Use your editor to copy these statements into your autoexec.bat file and change the parameters as appropriate.

14.2 Do Not Forget to Save Your Macros

Do not forget to save your macros at the end of a session. Unless you save the macros you have defined during the current session, they will be lost when you re-boot or turn off the computer.

It is suggested that as soon as you have finished defining a set of macros you wish to keep, that you save them immediately. It is very easy to start a session defining new macros, proceed to use them for the rest of the session, and then turn off the computer. If this happens, you will have to redefine each macro again. For information on saving your macros refer to section 11.1, "Save File".

14.3 Combining Macro Files

Occasionally, you may wish to combine two macro files. This may be done by loading one of the files into memory, merging the second into memory, and then saving the macros to disk.

You may also use your text editor to combine the two macro files.

14.4 Multiple Macro Files

You may wish to define multiple macro files based on function. This would allow you to easily build different macro files without redefining each function's macros for each macro file.

For example, you could define a file for your word processor, spreadsheet, general DOS commands, etc. and then merge various combinations of them together to create one file. They could also be used to create in-memory macros at boot time (using the merge function), without combining them all into one macro file.

14.5 Save All Macros Files in One Directory

Newkey now offers an option that makes it easy to save all of your macro files in one directory. This option allows you to set the default path and extension to use when loading or saving macro files. This makes it easy to load and save macro files because it is no longer necessary to specify the path and extension when the file is accessed. Instead the default path and extension will be added to the file name automatically. See section 11.4, "Default Path and Extension for Macro Files", for further information.

CHAPTER 14
TIPS ON USING NEWKEY

CHAPTER 15
COMMON PROBLEMS

15.1 Keys That Used to Work No Longer Work

Several reasons why keys that used to work no longer work follow:

- You are trying to use one of the synonym keys that Newkey has assigned a different key code to.

- The program you are using is trying to assign its own key code to one of the unused key combinations not supported by IBM, yet Newkey has already intercepted that key combination and assigned it a different code.

- The key is also used as one of Newkey's control keys. See section 10.1, "Control keys", for further information.

See section 10.5,"Preventing Key Creation" and "Appendix D" for more information on how to correct this problem.

15.2 Other Memory-resident Programs Hang

When invoked it appears that other memory-resident programs have hung up or will not read certain keystrokes. Try turning off the extended keyboard buffer feature and changing the load order.

15.3 Monitor Does Not Clearly Display Newkey Menus

You are probably using an amber or other monochrome monitor with a graphics card. In this case you should turn on Newkey's black and white option. See section 10.3, "Black and White Mode" for more information on how to correct this problem.

15.4 Screen Does Not Blank

There are many reasons why your screen may not blank depending on the hardware and software you are using. Newkey provides several screen blanking options designed to work with these configurations. Before trying these options, please read the full description of them in section 9.1, "Screen Saver Mode". It is not always obvious exactly how they work.

One reason why screen blanking might not work is if your software is continually writing to the screen, redisplaying the same exact screen, even though you have not touched a key for several minutes. In this case try turning the "Video I/O Restore Screen Option" off. Some versions of PC-Write and Word Perfect do this.

15.5 Cannot Restore Blanked Screen

Normally any keystroke or video i/o will restore the screen. However, some programs completely takeover the interrupts Newkey uses to determine when the screen should be restored. If you are having trouble restoring the screen, try pressing the alt, ctrl, left shift, or right shift keys.

If you are still unable to restore your screen then you may find it easiest to turn off the screen saver mode before using the problem program (see section 9.1, "Screen Saver Mode").

15.6 Newkey Does Not Work Right With Certain Programs

Some programs throw away parts of Newkey and you will notice that certain Newkey features have ceased to function. When these discourteous programs are running, they completely takeover the

interrupts Newkey uses. When this happens, these parts of Newkey are lost until the guilty program stops running. Fortunately only a few applications programs such as XYWrite and Smartcom are guilty of this.

Newkey has been designed to work around these programs, but it cannot protect itself completely. Rather than fight with these programs, Newkey lets them take over, but tries to make sure that the disappearance of one part will not affect another part's functioning. Otherwise, you will have to live without the missing Newkey functions while running these programs.

15.7 Cannot Access Extended Ascii (upper 128) codes

See section 13.2, "Generating Extended Ascii Codes".

15.8 Losing Part of Macro Definition

If your macro definition does not include all of the keystrokes you originally typed you have probably used Newkey's error correction key, the <ctrlh> key. If you wish to use the <ctrlh> key with its normal meaning, you must either redefine or deactivate Newkey's error correction key. For more information, see section 2.6, "Error Correction".

Another reason part of your definition may seem to be lost during playback is that some programs ignore keys which are returned faster than they expect. For more information on resolving this problem refer to section 4.6, "Slow Typing Mode", and section 3.27, "Time Delay".

15.9 Limited Memory Considerations

If you are running into memory problems consider using Newkeysm or Newkeyvs, which can save you 25K-38K at the cost of the pop-up features plus some of the macro functions. See also section 16.1, "Newkey Requirements."

15.10 Computer Hangs up and Must be Rebooted

Refer to the "readme" file and section 16.9, "Software Compatibility" for information on interactions with other software. Check to make sure that you are not using an old version of Newkeysp with a new version of Newkey or vice-versa. If these actions fail to help solve the problem, then refer to section 17.2, "Reporting Newkey Bugs" for more information on how to notify us of the problem.

CHAPTER 16
TECHNICAL OVERVIEW

16.1 Newkey Requirements

Newkey requires PC-DOS or MS-DOS 2.0 or greater and will run on either a monochrome or color monitor. Newkey comes in three versions, the normal full power version; Newkeysm, which lacks the pop-up features of the full-power version, but also requires 25K of memory less; and Newkeyvs, which is an even more abbreviated version of Newkeysm. Newkey requires approximately 60k (35K for the small version) of memory, plus an additional 2 bytes per character of translation and reserved macro buffer.

16.2 Newkeysm, The SMall Version

Newkeysm lacks the pop-up features of the full-power version and does not support the {begload}, {begmerge}, {begmergo}, and {begsave} macro functions. Otherwise Newkeysm will work just as the full-power version will at a savings of 25k.

16.3 Newkeyvs, The Very Small Version

This version does not have the pop-up features and lacks many other macro functions. It requires only about 22k, 38k less than the largest version.

16.4 Newkey's Limitations

Newkey allows redefinition of almost any key, including an extended set of keys beyond those supported by IBM (except ctrl break, and on the AT, SysReq). Newkey will handle up to approximately 64k of macros, and any macro may be as long as you wish up to this maximum. The 64k maximum includes 16 bytes for overhead and two bytes per character of translation.

16.5 Hardware Compatibility

Newkey is compatible with IBM PC, XT, Jr, AT, and all true compatibles. Users of amber or similar monitors that display in two colors only, but use the graphics card should consult section 10.3, "Black and White Mode".

16.6 Enhanced Keyboard

Newkey recognizes and supports the enhanced keyboard. For a list of the additional keys supported refer to Appendix A, "Keys Supported.

16.7 Hercules Graphics boards

Newkey has been designed to recognize Hercules cards and use a different method of saving your screen. This method was provided by the Hercules Technology Corporation for use with their cards so it should be safe. Please note that the Hercules Technology Corporation has not tested this implementation and does not approve or endorse it. We have tested it and found no problems.

16.8 Graphics Modes

Newkey is able to pop-up in the middle of programs using the standard graphics modes for the CGA and EGA boards. You may notice some strange behavior such as:

- When you start a macro definition your screen will appear to

have become garbage except for the Newkey macro definition window. This is simply your graphics output being interpreted as text. When you continue, your screen will be redisplayed unchanged.

- The status line will not display. This is because Newkey cannot display the status line while in graphics modes. Newkey will beep to remind you that you are defining a key.

- Your screen may be slightly changed when it is restored, but it should still be recognizable.

16.9 Software Compatibility

Many programs operate at Newkey's level. For the most part these programs work very well with each other, but inevitably problems arise. As a result, it has become impossible to give guidelines for which programs should be loaded in what order.

If you are having trouble, try loading your memory-resident programs one at a time in several different orders until you isolate which programs are causing your problems. Once you have determined which programs are causing the problem and found that changing their order of loading does not solve the problem, try changing the Newkey parameters, especially:

- Extended keyboard buffer
- Defining status line
- Ignore keys option on Newkeysp menu
- Using the /noems option when loading Newkey

For the most part you will have to use trial and error to determine these for yourself. For information on specific programs refer to the "readme" file on the Newkey disk.

16.10 Shift Status Reset

Some programs check the shift status when they read certain characters, enabling them to effectively extend the character set normally supported by the IBM PC. This causes problems if you have defined some of these characters in a macro which is invoked by a shift key (such as ctrl, alt, or caps), because the shift status will remain set as the playback occurs. As a result, the application program may interpret certain characters incorrectly.

Newkey now has an option that which will turn the shift flags off during playback and restore them when the playback is finished. This option comes turned off, but may be turned on by selecting the "Parameters" option in Newkeysp and following instructions. We recommend that you do not activate this option unless you have a specific need.

Timing problems, however, may nullify some of this option's usefulness. Application programs frequently read keys into their buffers before they are actually ready to process them. When this happens, the shift status may have been restored before the character is processed. The best way around this is to allow the playback to finish before continuing.

PC-Write is one program which does this. Whenever it reads a space, it checks the shift status to see if the alt or ctrl flags are set. If so, it interprets the space differently than it would a plain space. For

example, if you assigned your name and address to <altf10>, on playback you would find that they had run together and all the spaces had disappeared! Shift status reset can help solve this problem.

CHAPTER 16
TECHNICAL OVERVIEW

CHAPTER 17
MISCELLANEOUS

17.1 Support

Telephone and mail support is available to registered users of Newkey from FAB Software or our local representatives. FAB Software provides support on the following basis: Telephone support is generally available from 7 PM - 9 PM weekdays and 10 AM - 9 PM weekends. If we are not available, an answering machine will take your message. For mail support, enclose a self-addressed stamped envelope for a faster response.

FAB Software
P.O. Box 336
Wayland, MA 01778

Phone number: (508) 358-6357
Compuserve: 75206,1366

17.2 Reporting Newkey Bugs

It is our intention to provide a high quality product at very reasonable prices. To this end we do our best to make our products as error free and compatible with other software as we can. Please notify us of any problems that you find, so that we can provide you with as high a quality product possible.

To encourage this, if you are the first person to notify us of a particular problem, we will send you a free update when the problem is corrected (and if not, you will receive a letter explaining why.) This offer applies only to bugs in the Newkey programs and not to incompatibilities with specific other programs or non-IBM hardware.

We have had few true user reported bugs since introducing Newkey in March 1984, so we do not expect you to have any problems, but if you believe you have discovered a Newkey bug, follow these steps:

1. Create a disk which may be used to recreate the problem. This disk should include the following:

   - Your copy of Newkey
   - Your macro files
   - Any other programs, memory-resident and otherwise, that must be present when the problem occurs (if legally possible).
   - A batch file that recreates your environment
   - A readme file which describes the problem and how to recreate it. Mention which version of DOS you are using and your phone number.

2. Send the above items to the address listed in section 17.1, "Support".

17.3 Restricted Permission to Copy

Currently Newkey is distributed with a letter describing how to turn off the "evaluation notice" screen that is normally displayed at startup. Because we have adopted this method of distributing Newkey, a registered user may freely share the full Newkey disk subject to the restrictions listed below. The only difference between the evaluation version and the version available to registered users is the "evaluation notice" screen that is displayed when Newkey or Newkeysp is first started up.

17.3.1 Rules For Copying Evaluation Version

Individuals are granted permission to freely copy the Newkey disk for their own evaluation or for others to evaluate, so long as no price or other consideration is charged.

Computer clubs are granted permission to freely copy the Newkey disk and share it with their members for the purposes of evaluation, so long as:

1. No price or other consideration is charged. However, a distribution cost may be charged for the cost of the diskette, so long as it is not more than US $10 total. This includes shipping, handling, and all other charges.

2. Club members who receive the programs are informed in writing of the user-supported concept and encouraged to support it with their payments.

3. The programs and documentation are not modified in any way and are distributed together.

4. The Newkey disk may not be packaged together with any other programs or materials. It may not be bundled and sold as part of some other more inclusive package.

5. The programs may not be "rented" to others.

6. The printed manual is not copied or reproduced in any way.

Companies, schools, universities, government, and other organizations are granted permission to copy the Newkey programs for use on other computers and at other locations in the company, so long as:

1. The full registration fee has been paid for each and every system on which the program will be used.

2. The printed manual is not copied or reproduced in any way.

17.4 Price List and Order Form

For a detailed price list and order form print the file "order.frm". Dealers and resellers please write or call for latest prices.

17.5 The Shareware Concept

Shareware, also know as "User-supported software", is an experiment in distributing computer programs, based on these beliefs:

1. That the value and utility of software is best assessed by the user on his/her own system.

2. That the creation of personal computer software can and should be supported by the computing community.

Subject to the rules described in section 17.3, "Restricted Permission to Copy" above, anyone may legally obtain an evaluation copy of the program from a friend or computer club. After you have had

a chance to use and evaluate the program in your own environment, you are trusted to either forward a payment to the FAB Software, or to discontinue use of the program.

Free distribution of software and voluntary payment for its use eliminates costs for advertising and copy protection schemes. Users obtain quality software at greatly reduced cost. They can try it out before buying, and do so at their own pace and in the comfort of their own home or office. The best programs will survive, based purely on their quality and usefulness.

17.6 Association of Shareware Professionals

The author of Newkey, Frank Bell, is a member of the Association of Shareware Professionals (ASP), an organization formed in April 1987 to strengthen the future of shareware as an alternative to software distributed through the normal commercial channels. Its members, all of whom are programmers, subscribe to the ASP's code of ethics. Software distributed by ASP members must meet the ASP's software standards. These standards include:

PROGRAMMING STANDARDS:

The program meets the ASP's definition of "shareware" (i.e., it is not a commercial demo with major feature disabled, nor a time-limited program).

The program has been thoroughly tested by the author and should not be harmful to other files or hardware if used properly.

DOCUMENTATION STANDARDS:

Sufficient documentation is provided to allow the average user to try all the major functions of the program.

Any discussion of the shareware concept and of registration requirements is done in a professional and positive manner.

SUPPORT STANDARDS:

The member will respond to people who send registration payments, as promised in the program's documentation. At a minimum, the member will acknowledge receipt of all payments.

The member will establish a procedure for users to report, and have acknowledged, matters such as bug reports, and will describe such means in the documentation accompanying all versions of the programs. The member will respond to written bug reports from registered users when the user provides a
self-addressed, stamped envelope.

Known incompatibilities with other software or hardware and major or unusual program limitations are noted in the documentation that comes with the shareware (evaluation) program.

CHAPTER 17
MISCELLANEOUS

GENERAL:

Members will keep the ASP apprised of changes in mailing address, of which shareware programs they have published and are currently supporting, and of the current version numbers and of any changes in the status of their programs.

If a user has a dispute with an ASP member-author, the user may appeal to the ASP to mediate for arbitration of the dispute.

APPENDIX 18
KEYS SUPPORTED

Newkey supports all keys supported by IBM, plus many more. For a list of additional keys supported by Newkey and not by IBM, refer to appendix C.

The following list includes all keys supported by Newkey. The list is the same format as used by Newkey to load from.

<a>	<A>	<alta>	<ctrla>	<s>	<S>	<alts>	<ctrls>	< >
<b>	<B>	<altb>	<ctrlb>	<t>	<T>	<altt>	<ctrlt>	<=>
<c>	<C>	<altc>	<ctrlc>	<u>	<U>	<altu>	<ctrlu>	<[>
<d>	<D>	<altd>	<ctrld>	<v>	<V>	<altv>	<ctrlv>	<]>
<e>	<E>	<alte>	<ctrle>	<w>	<W>	<altw>	<ctrlw>	<;>
<f>	<F>	<altf>	<ctrlf>	<x>	<X>	<altx>	<ctrlx>	<'>
<g>	<G>	<altg>	<ctrlg>	<y>	<Y>	<alty>	<ctrly>	<`>
<h>	<H>	<alth>	<ctrlh>	<z>	<Z>	<altz>	<ctrlz>	<\>
<i>	<I>	<alti>	<ctrli>	<1>	<!>	<alt1>	<ctrl1>	<,>
<j>	<J>	<altj>	<ctrlj>	<2>	<@>	<alt2>	<ctrl2>	<.>
<k>	<K>	<altk>	<ctrlk>	<3>	<#>	<alt3>	<ctrl3>	</>
<l>	<L>	<altl>	<ctrll>	<4>	<$>	<alt4>	<ctrl4>	<esc>
<m>	<M>	<altm>	<ctrlm>	<5>	<%>	<alt5>	<ctrl5>	<tab>
<n>	<N>	<altn>	<ctrln>	<6>	<^>	<alt6>	<ctrl6>	<bks>
<o>	<O>	<alto>	<ctrlo>	<7>	<&>	<alt7>	<ctrl7>	<enter>
<p>	<P>	<altp>	<ctrlp>	<8>	<*>	<alt8>	<ctrl8>	<prt>
<q>	<Q>	<altq>	<ctrlq>	<9>	<(>	<alt9>	<ctrl9>	<->
<r>	<R>	<altr>	<ctrlr>	<0>	<)>	<alt0>	<ctrl0>	<_>

<+>	<alt=>	<ctrl=>	<f1>	<capsf1>	<altf1>	<ctrlf1>
<{>	<alt[>	<ctrl[>	<f2>	<capsf2>	<altf2>	<ctrlf2>
<}>	<alt]>	<ctrl]>	<f3>	<capsf3>	<altf3>	<ctrlf3>
<:>	<alt;>	<ctrl;>	<f4>	<capsf4>	<altf4>	<ctrlf4>
<">	<alt'>	<ctrl'>	<f5>	<capsf5>	<altf5>	<ctrlf5>
<~>	<alt`>	<ctrl`>	<f6>	<capsf6>	<altf6>	<ctrlf6>
<|>	<alt\>	<ctrl\>	<f7>	<capsf7>	<altf7>	<ctrlf7>
<caps,>	<alt,>	<ctrl,>	<f8>	<capsf8>	<altf8>	<ctrlf8>
<caps.>	<alt.>	<ctrl.>	<f9>	<capsf9>	<altf9>	<ctrlf9>
<?>	<alt/>	<ctrl/>	<f10>	<capsf10>	<altf10>	<ctrlf10>
<alt->	<ctrl->					

<home>	<num7>	<althom>	<ctrlhom>	<up>	<num8>	<altup>	<ctrlup>
<pgup>	<num9>	<altpgup>	<ctrlpgu>	<rgt>	<num6>	<altrgt>	<ctrlrgt>
<ctr>	<num5>	<altctr>	<ctrlctr>	<lft>	<num4>	<altlft>	<ctrllft>
<end>	<num1>	<altend>	<ctrlend>	<dn>	<num2>	<altdn>	<ctrldn>
<pgdn>	<num3>	<altpgdn>	<ctrlpgd>	<ins>	<num0>	<altins>	<ctrlins>
<del>	<num.>	<altdel>	<ctrldel>	<num->	<capsnu->	<altnu->	<ctrlnu->
<num+>	<capsnu+>	<altnu+>	<ctrlnu+>				

<capspace>	<altspace>	<ctrlspac>	<capsesc>	<altesc>	<ctrlesc>
<capstab>	<alttab>	<ctrltab>	<capsbks>	<altbks>	<ctrlbks>
<capsent>	<altent>	<ctrlent>	<altprt>	<ctrlprt>	

APPENDIX 18
KEYS SUPPORTED

Enhanced Keyboard Entries for separated keys

<f11>	<capsf11>	<altf11>	<ctrlf11>	<f12>	<capsf12>	<altf12>	<ctrlf12>
<shom>	<shom>	<altshom>	<ctrlshom>	<sup>	<sup>	<altsup>	<ctrlsup>
<spgu>	<spgu>	<altspgup>	<ctrlspgu>	<srgt>	<srgt>	<altsrgt>	<ctrlsrgt>
<slft>	<slft>	<altslft>	<ctrlslft>	<send>	<send>	<altsend>	<ctrlsend>
<sdn>	<sdn>	<altsdn>	<ctrlsdn>	<spgd>	<spgd>	<altspgdn>	<ctrlspgd>
<sins>	<sins>	<altsins>	<ctrlsins	<sdel>	<sdel>	<altsdel>	<ctrlsdel>
</>	</>	<altnum/>	<ctrlnum/>	<nument>	<nument>	<nument>	<ctlnuent>

All keys in IBM's extended ascii set, those with values in the 128 - 255 range, may be entered as a single character in the same manner as any character in the alphabet. See "Generating Extended Ascii Codes" for information on accessing the extended set.

Undefinable Keys

These keys (including any combination such as ctrl break) cannot be redefined:

Alt
Break
Caps Lock
Ctrl
Num Lock
Scroll Lock
Shift (left and right)
Shift-Prtsc
Sys Req

APPENDIX 19
FUNCTIONS

Function	Description
{beep}	beep
{begdef *macroname*}	begin *macroname* definition
{begdesc}	begin a description
{begfoot}	begin a footer, used for menu macros
{begload}	load a macro file
{begmenu *row,col*}	begin displaying menu at *row* and *col*
{begmerge}	merge a macro file into memory
{begmergo}	merge a macro file with overwrite
{begprint}	begin sending to printer
{begsave}	save macros to a disk file
{begtitle}	begin a title, used for menu macros
{begwind *row,col,#lines,length*}	begin displaying window starting at *row, col,* for *#lines* and *length*
{cancel}	cancel all macros executing
{capsoff}	turn caps lock off
{capson}	turn caps lock on
{capsrest}	restore caps lock status
{capssave}	save caps lock status
{clearmac}	clear macros from memory
{cls}	clear the screen
{cmd}	pop up the newkey menu
{cursoff}	turn the cursor off
{curson}	turn the cursor on
{cut *macroname row,col,#lines,length*}	cut from screen starting at *row, col* for *#lines* and *length* and assign to *macroname*
{date}	begin a date function
{defend}	end the define function
{define *macroname*}	begin defining *macroname* within a macro
{delete *macroname*}	delete *macroname*
{endtime}	end time function
{endcmd}	end of pop up
{enddate}	end date function
{enddef}	end of definition
{enddesc}	end description
{endfoot}	end footer function
{endife}	end {ifescr} function
{endifne}	end {ifnescr} function
{endifef}	end {ifefscr} function
{endifnef}	end {ifnefscr} function
{endload}	end load file function
{endmenu}	end menu function
{endmerge}	end merge function
{endmergo}	end merge with overwrite function
{endprint}	end the print function
{endsave}	end save file function
{endtitle}	end title function
{endwind}	end display window function
{exitkey *key*}	exit from menu macro when *key* pressed
{ffld}	fixed length pause
{ffldtran}	fixed length pause, but allow translation
{ifescr *row,col*}	if screen equal function

APPENDIX 19
FUNCTIONS

{ifnescr *row,col*}	if screen not equal function
{ifefscr}	if any place on screen equal function
{ifnefscr}	if any place on screen not equal function
{goto *macroname*}	goto *macroname*, do not return to current macro
{guard}	protect the macro from being deleted when a new macro file is loaded
{input *macroname row,col*,1,*length*}	stop playback and wait for input entered by user starting at *row* and *col* for *length*
{notran *macroname*}	do not translate *macroname*
{numoff}	turn num lock off
{numon}	turn num lock on
{numrest}	restore num lock status
{numsave}	save num lock status
{prtscrn}	invoke the print screen function
{restore}	restore screen saved by display macro
{return}	return to macro which called this macro
{scrloff}	turn scroll lock off
{scrlon}	turn scroll lock on
{scrlrest}	restore scroll lock status
{scrlsave}	save scroll lock status
{scroff}	turn screen off
{scron}	turn screen on
{set *option value*}	set option to value
{time}	begin time function
{vfld}	variable length function
{wait *MM:SS:HH*}	wait for *MM* minutes, *SS* seconds, *HH* hundredths of a second
{wait4key *key*}	wait for *key* to be pressed, then discard
{waitany}	wait for any key to be pressed, then discard
{waitanyk}	wait for any key, leave in buffer

APPENDIX 20
EXTENDED CODES

Second code	Function
1(@)	ALT Esc
2(*)	CTRL 1
3	CTRL 2
4-6(*)	CTRL 3,4,5
7(*)	CTRL =
8-11(*)	CTRL 7,8,9,0
12(*)	CTRL Space
13(*)	CAPS Bksp
14(@)	ALT Bksp
15	Back Tab
16-25	ALT Q,W,E,R,T,Y,U,I,O,P
26-28(@)	ALT [ ] Enter
29(*)	CTRL ;
30-38	ALT A,S,D,F,G,H,J,K,L
39-41(@)	ALT ; ' ‘
42(*)	CTRL ' (apostrophe)
43(@)	ALT \
44-50	ALT Z,X,C,V,B,N,M
51-53(@)	ALT , . / (comma, period, slash)
54(*)	CTRL ‘ (back apostrophe)
55(@)	ALT Prtsc
56-58(*)	CTRL , . /
59-68	F1-F10
69(*)	CTRL Esc
70(*)	CAPS Esc
71	Home
72	Up cursor
73	PG UP
74(@)	ALT NU-
75	Left cursor
76(@)	CTR
77	Right cursor
78(@)	ALT NU+
79	End
80	Down cursor
81	PG DN
82	INS
83	DEL
84-93	F1-F10 (Upper case)
94-103	F1-F10 (CTRL)
104-113	F1-F10 (ALT)
114	CTRL Prtsc
115	CTRL left cursor
116	CTRL right cursor
117	CTRL End
118	CTRL PG DN
119	CTRL Home
120-131	ALT 1,2,3,4,5,6,7,8,9,0,-,=

APPENDIX 20
EXTENDED CODES

(Top of keyboard)

132	CTRL PG UP
133(#)	F11
134(#)	F12
135(#)	CAPS F11
136(#)	CAPS F12
137(#)	CTRL F11
138(#)	CTRL F12
139(#)	ALT F11
140(#)	ALT F12
141(@)	CTRL UP
142(@)	CTRL NU-
143(@)	CTRL CTR
144(@)	CTRL NU+
145(@)	CTRL down cursor
146(@)	CTRL INS
147(@)	CTRL DEL
148(@)	CTRL TAB
149(#)	CTRL NU/
150(#)	CTRL NU+
151(#)	ALT home (separate from numeric keypad)
152(#)	ALT up cursor (separate from numeric keypad)
153(#)	ALT PGUP (separate from numeric keypad)
154(*)	CAPS Space
155(#)	ALT left cursor (separate from numeric keypad)
156(*)	ALT CTR
157(#)	ALT right cursor (separate from numeric keypad)
158(*)	ALT SPACE
159(#)	ALT end (separate from numeric keypad)
160(#)	ALT down cursor (separate from numeric keypad)
161(#)	ALT PGDN (separate from numeric keypad)
162(#)	ALT INS (separate from numeric keypad)
163(#)	ALT DEL (separate from numeric keypad)
164(@)	ALT NU/
165(@)	ALT TAB
166(@)	ALT ENTER (on numeric keypad)
167(*)	CAPS NU-
168(*)	CAPS NU+
169(*)	CAPS ENTER

The following keys refer to the numeric keypad.

170(*)	ALT home
171(*)	ALT end
172(*)	ALT down cursor
173(*)	ALT up cursor
174(*)	ALT left cursor
175(*)	ALT right cursor
176(*)	ALT page down

177(*)	ALT page up
178(*)	ALT page ins
179(*)	ALT page del

These keys are part of IBM's keyboard extended functions in which an ascii code = 0 and a scan code equal to the number in the list above is returned to the program.

(*) Keys with this next to their number have been added by Newkey and will be returned to your program if a translation is not found. This might cause problems, if your program does not ignore unknown keys. Most programs do ignore unrecognizable keys. If you are having this problem, try redefining the key to a space or use the 'ignore' keys feature.

(@) Keys with this next to their number are created by IBM's new keyboard or by Newkey if you are not using the new keyboard.

(#) Keys with this next to their number are created by IBM's new keyboard, but not by Newkey.

APPENDIX 20
EXTENDED CODES

APPENDIX 21
SYNONYMS

These synonyms are keystrokes which return the same ascii value and sometimes with different scan codes. In most programs, they are treated the same. Newkey now allows you to redefine these synonyms individually without redefining its mate. This provides you with greater flexibility and a wider range of keys which can be redefined.

To do this Newkey must give some of these synonyms (those with both identical ascii and scan codes) their own unique key codes. These are the keys in the right hand column in the table below. These keys will not act as they did before Newkey was loaded. For example, the caps enter key will not act like an <enter> key as it used to. If it has not been given a macro it will probably be ignored by your programs. If you want them to act like they used to just define them using their native meaning or use the "ignore keys" option on the Newkeysp main menu. Following our previous example, you would define the caps enter key as enter.

SYNONYMS		Synonyms With New Codes
0-9	numeric 0-9	
+	numeric +	caps numeric +
-	numeric -	caps numeric -
period	numeric period	
*	prtsc *	
enter	ctrl m	caps enter
backspace	ctrl h	caps backspace
ctrl enter	ctrl j	
fwd tab	ctrl i	
esc	ctrl [	caps & ctrl esc
space		alt, caps, ctrl space

APPENDIX 21
SYNONYMS

APPENDIX 22
COLOR CODES

When defining display or menu macros a foreground or background color will be requested. The numbers associated with these colors are:

COLOR	CHARACTER	BACKGROUND
Black	0	0
Blue	1	16
Green	2	32
Cyan	3	48
Red	4	64
Magenta	5	80
Brown	6	96
Light Grey	7	112
Dark Grey	8	
Light Blue	9	
Light Green	10	
Light Cyan	11	
Light Red	12	
Light Magenta	13	
Yellow	14	
White	15	

Character + background number = attribute

For example, red on blue is 4 + 16 = 20
Add 128 to get blinking

APPENDIX 22
COLOR CODES

APPENDIX 23
SUMMARY OF CHANGES

Important conversion information for users of earlier versions is highlighted.

Versions 5.1 - 5.4

- Added ability to use EMS memory
- Updated user interface
- Added if screen macro functions
- Added Print screen macro function
- Added Return and Cancel macro functions

Version 5.0 (November 1988)

The version 5.0 macro file format is different than version 4.0. For instructions on how to convert your files, run the program CONV4TO5.

Several control keys have been changed so that the same set of keys may be used on both the enhanced and regular keyboards. These keys are:

Old	New	Description
<ctrl/>	<alt/>	Pop-up Newkey menu
<ctrlesc>	<ctrldel>	Cancel Newkey processing
<cutpaste> <ctrlins>	Default key to cut to	

If you are unhappy with the new key assignments the "Update Control Keys" menu selection may be used to permanently change them.

The <cutpaste> key generated by alt-shift-p is no longer created.

The following functions can no longer be entered by pressing a hot key (use the macro editor instead):

- time delay (<ctrl->)
- slow typing off (<ctrl3>)
- slow typing on (<ctrl4>)

The ability to make absolute parameter changes has been replaced by the {set} option.

All of the extra keys created by Newkey on the regular keyboard are now also created on the enhanced keyboard.

- Improved, mnemonic user interface
- Significantly faster file loading and macro clearing
- Display macros allow text to be displayed and input requested
- Menu macros allow custom build menu systems
- Edit macros as they are being defined
- Default path and extension for macro files
- Easy control of shift status (alt, ctrl, and scroll)
- Clear macros, a new macro function
- Load and merge macros from disk, a new macro function
- Save macros to disk, a new macro function
- Define a macro to a specific definition, a new macro function
- Set Newkey options from within a macro, new macro functions
- Delete a macro from within a macro, a new macro function

APPENDIX 23
SUMMARY OF CHANGES

- Send a macro to the printer, a new macro function,
- Wait for next key before continuing, a new macro function
- Wait for next key before continuing, but keep key, a new macro function
- Wait for specific key before continuing, a new macro function
- Fixed field which allows macro playback of key struck, a new macro function
- Turn cursor on/off, new macro functions
- Cut from screen, new macro function
- Specify the key to cut to when cutting
- Specify the cut and paste line end key
- Signal shorthand macro end without having delimiter included in playback
- Zerofill date function switch
- Caps/ctrl key switch on enhanced keyboard
- Alt, caps, and ctrl space supported
- Improved support on enhanced keyboards for additional key codes

Version 4.0 (April 1987)

The version 4.0 macro file format is slightly different than version 3.0. For instructions on how to convert your files run the program CONV3TO4.

I N D E X

I N D E X

J

K

L

M

N

O

P

W

X

Y

Z

I N D E X