The COMPLETE 64

Seven powerful, sophisticated applications on disk including *SpeedScript* and *SpeedCalc*, COMPUTE!'s excellent word processing and spreadsheet programs. For the Commodore 64 and 128 personal computers.

PROGRAM DISK Enclosed

A **COMPUTE! Books** Publication $29.95

COMPUTE!'s THE COMPLETE 64

COMPUTE!™ Publications, Inc. abc
Part of ABC Consumer Magazines, Inc.
One of the ABC Publishing Companies
Greensboro, North Carolina

"High-Resolution Sketchpad" was originally published in *COMPUTE!'s First Book of Commodore 64 Sound and Graphics,* copyright 1983, COMPUTE! Publications, Inc.

"Sidplayer" was originally published in *All About the Commodore 64, Volume Two,* copyright 1985, COMPUTE! Publications, Inc.

The following articles were originally published in *COMPUTE!'s Gazette,* copyright by COMPUTE! Publications, Inc.: "Ultrafont +" (July 1984); "Mini-Filer" (February 1986).

The following articles were originally published in *COMPUTE!* magazine, copyright by COMPUTE! Publications, Inc.: "Plus/Term for the VIC & 64" (February 1985); "SpeedScript 3.0: All Machine Language Word Processor for the Commodore 64" (March 1985); "*SpeedCalc* for Commodore 64 and 128" (January 1986).

Printed in the United States of America

10 9 8 7 6 5 4 3 2 1

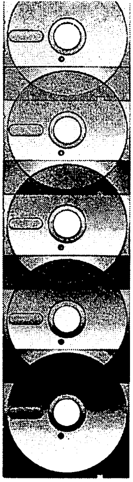
ISBN 0-87455-062-9

The authors and publisher have made every effort in the preparation of this book and disk to insure the accuracy of the programs and information. However, the information and programs are sold without warranty, either express or implied. Neither the authors nor COMPUTE! Publications, Inc., will be liable for any damages caused or alleged to be caused directly, indirectly, incidentally, or consequentially by the programs or information in this book or disk.

The opinions expressed in this book are solely those of the authors and are not necessarily those of COMPUTE! Publications, Inc.

COMPUTE! Publications, Inc., Post Office Box 5406, Greensboro, NC 27403, (919) 275-9809, is part of ABC Consumer Magazines, Inc., one of the ABC Publishing Companies, and is not associated with any manufacturer of personal computers. Commodore 64 and 128 are trademarks of Commodore Electronics Limited.

Contents



Foreword

The Commodore 64 and 128 have a reputation as great game machines. They're even more highly regarded as computers for programming. But they can do much more. *The Complete 64* includes application programs which will turn your Commodore into an extraordinarily useful tool for home and business.

Seven sophisticated applications make *The Complete 64* an easy-to-use, high-quality collection of the most popular kinds of software used on personal computers. All at a price that makes this collection the best software value anywhere for the Commodore 64 and 128.

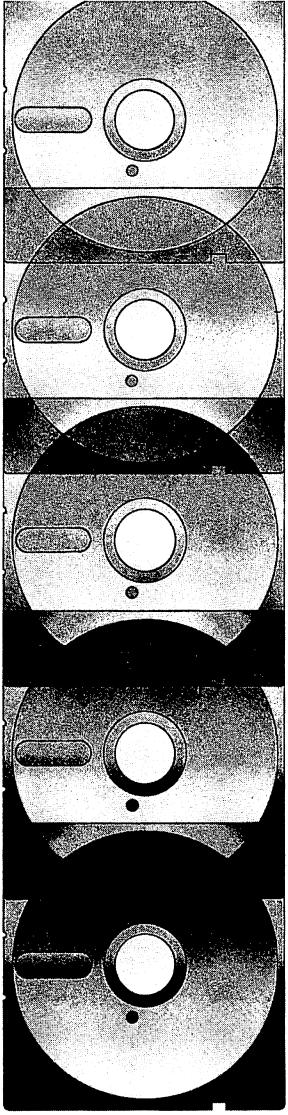
Written in your computer's native language—machine language—or in a combination of machine language and BASIC, these applications rival the best commercial software in quality, power, and speed.

Write anything from a letter to a bestselling novel with *SpeedScript,* the well-known word processor for the Commodore 64 and 128. Keep your family's budget or do cost analysis with the spreadsheet program *SpeedCalc.*

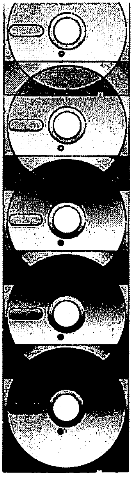
The SID chip of the 64 and 128 can produce excellent music. The "Sidplayer" editor and player make exploiting this capacity easy. You don't even have to enter the music—included on the disk are a number of interesting and complex songs to enjoy immediately. (There's even more music available—using a modem and "Plus/Term," you can download songs from online information services like Delphi, PlayNet, and GEnie.)

The Complete 64 wouldn't be complete without a drawing program—"High-Resolution Sketchpad;" a database application—"Mini-Filer;" and a terminal program—"Plus/Term."

The Complete 64 is your instant library of quality, power software for the Commodore 64 and 128.



Chapter 1
Introduction



Introduction

Your Commodore 64 or 128 is a powerful computer, great for everything from playing games to programming. With sprites, color, and plenty of memory, it's possible to write impressive software for these machines. But the Commodore 64 and 128 can be used for much more than just games and programming.

Putting your computer to work has never been easier—*The Complete 64,* a book and disk package, contains seven of the best applications ever published by COMPUTE! Publications for the Commodore 64 and 128 in 64 mode. This book includes all the information you need, and all the programs are on the disk.

The Complete 64 includes the kinds of applications computer owners use the most—like word processing, telecommunications, filing, and spreadsheet analysis. Rivaling commercial software quality, all the applications are written in the 64's native language—machine language—or a combination of machine language and BASIC.

SpeedScript. This popular word processing program lets you write, rewrite, edit, print, and save anything from a short letter to a complete novel. If you've never used a word processor, you'll be amazed at how easy and convenient *SpeedScript* is to use.

Along with *SpeedScript,* we've included five easy-to-install custom character sets. If you get tired of looking at the usual Commodore characters when writing your great American novel, you can switch to an alternate font. And if you want to design your own character set, you can use "Ultrafont +."

SpeedCalc. Imagine a large sheet of accounting paper with rows and columns. Perhaps the first column has a list of all the items your business keeps in inventory. Maybe the second column shows how many of each item you presently have in stock. The third column can have the cost of each item, and the fourth calculates the value of the inventory by multiplying the second and third columns. At the bottom of the second, third, and fourth columns are totals.

The worksheet described above is rather common. Most businesses keep this type of information. The problem with the pencil and paper method of inventory worksheet is that each time you add or subtract inventory, the totals need to be recalulated.

SpeedCalc is a spreadsheet program that does all the work for you. Change one number, and your 64 or 128 recalculates the entire sheet. But *SpeedCalc* isn't limited to just inventory management. You can create any kind of worksheet with *SpeedCalc*. Track your stock portfolio and bank accounts, or computerize your family budget—the uses for spreadsheets are limitless.

You can even add *SpeedCalc* data to a *SpeedScript* document. The process is easy and fully explained.

Mini-Filer. This general-purpose file manager can be used for almost any task. Since it's not designed to fulfill a specific purpose, it's general enough to handle any kind of file structure. For example, you could create a name and address file (name, street, city, state, zip code, and phone number), a file for your garden (plant name, gestation, watering/feeding needs, and harvest time), or a file for your library (title, author, publication date, publisher, and subject). Of course, you can save your file to disk to use later.

Sidplayer. The Commodore 64 and 128 can play great music. With Sidplayer, your computer can strut its stuff—playing music and duplicating the sounds of most instruments. Whether you're an expert or can't carry a tune, you'll be able to use and enjoy the Sidplayer.

Sidplayer includes a player program, an editor, and a number of utility programs. The *Complete 64* disk includes lots of songs you can immediately play and edit. You'll be amazed at the quality of the music your Commodore can produce. More songs, ready to download with "Plus/Term," can be found on commercial information systems like Delphi, Playnet, and GEnie.

Ultrafont +. Create your own custom character sets to use with your own programs or with *SpeedScript*. You can even create multicolor characters for game programs. Ultrafont + is easy and fun to use.

High-Resolution Sketchpad. Draw pictures or create shapes using the keyboard or a joystick. Fill a shape or change drawing colors with a touch of a key. High-Resolution Sketchpad has the features you need to create detailed and spectacular screens quickly and easily. When you're finished, save your masterpieces to disk and reload them later or use them in a BASIC program.

Plus/Term.This full-featured terminal program puts you in instant contact with other computers anywhere in the world. Use Plus/Term and your favorite modem to download songs for Sidplayer and programs from bulletin boards and information services like CompuServe, The Source, Delphi, and PlayNet.

The Complete 64 Disk

The Complete 64 consists of two parts. The book contains the documentation for each of the applications included on the disk. Although the applications are easy to use, you'll need the documentation to get full use of each of the programs.

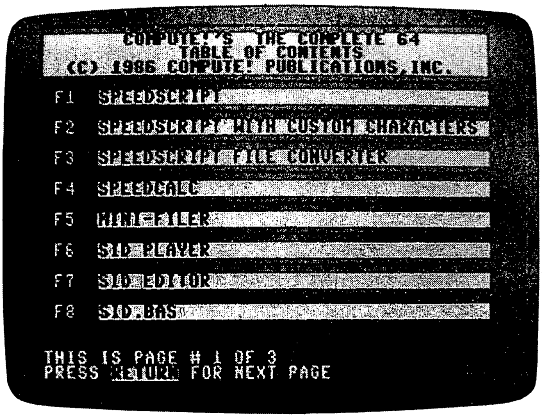
The second part of the package is the disk. On the disk are all the programs and files discussed in the documentation. *The Complete 64* includes seven practical applications, but there are numerous supporting programs and files on the disk. All such files are related to the applications; for example, in addition to the Sidplayer and the Editor, the disk contains all the songs and music discussed in text and a number of other songs you'll enjoy listening to and modifying.

Using the Disk

The Complete 64 disk will work on any Commodore 64 or 128 (in 64 mode) with a 1541 or 1571 disk drive. To begin, turn on your monitor or TV and the disk drive. Next, turn on your computer. If you're using a 128, type **GO 64** and press RETURN. Insert your *Complete 64* disk and type

```
LOAD "MENU",8
```

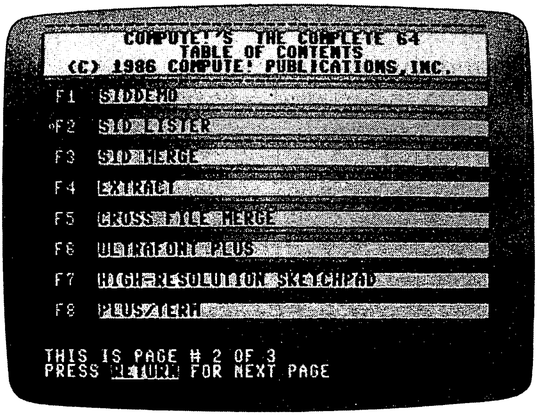
and press RETURN. The drive will spin. Once the READY prompt appears, type **RUN** and press RETURN. After a few moments you'll see the menu screen.



```
      COMPUTE!'S THE COMPLETE 64
          TABLE OF CONTENTS
 (C) 1986 COMPUTE! PUBLICATIONS,INC.
F1  SPEEDSCRIPT
F2  SPEEDSCRIPT WITH CUSTOM CHARACTERS
F3  SPEEDSCRIPT FILE CONVERTER
F4  SPEEDCALC
F5  MINI-FILER
F6  SID PLAYER
F7  SID EDITOR
F8  SID.BAS

THIS IS PAGE # 1 OF 3
PRESS RETURN FOR NEXT PAGE
```

Page 1 of the menu.

If the program you want to load is on the first page, press the appropriate key. You'll see a message indicating that the program is loading. If the program you want is not on page 1 of the menu, press RETURN once to see page 2 and twice to see page 3.



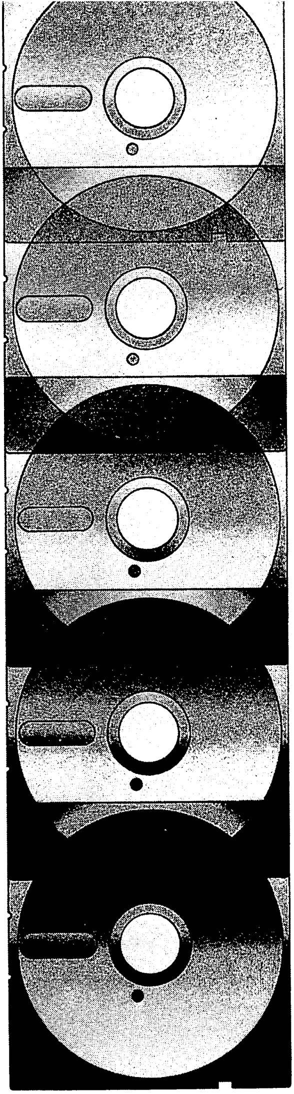
```
      COMPUTE!'S THE COMPLETE 64
          TABLE OF CONTENTS
 (C) 1986 COMPUTE! PUBLICATIONS,INC.
F1  SIDDEMO
F2  SID LISTER
F3  SID MERGE
F4  EXTRACT
F5  CROSS FILE MERGE
F6  ULTRAFONT PLUS
F7  HIGH-RESOLUTION SKETCHPAD
F8  PLUS/TERM

THIS IS PAGE # 2 OF 3
PRESS RETURN FOR NEXT PAGE
```

Page 2 of the menu.

Some of the applications in *The Complete 64* change memory pointers. To be safe, turn the computer off for a few seconds, then back on before you begin work with a second application.

Copying the Applications to Another Disk

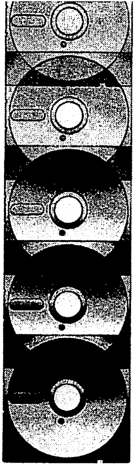
Since most of the programs and files on the *Complete 64* disk are not written in BASIC, you can't simply load and save the files to another disk (two notable exceptions are *SpeedScript* and *SpeedCalc* which can simply be loaded from the disk and resaved to another disk). To copy most of the files on the disk, you should use a copy program such as *Unicopy, Copyfile,* or the copy program that came with your disk drive. The *Complete 64* disk is *not* copy-protected. However, it is *write*-protected, so you can't write to, or erase, the disk.



Chapter 2

SpeedScript

Charles Brannon



Using *SpeedScript*

Since its introduction in the January 1984 issue of *COMPUTE!'s Gazette, SpeedScript* has been the most popular program ever published by COMPUTE! Publications. Written entirely in machine language, *SpeedScript* contains nearly every command and convenience you'd expect from a quality word processor. The version of *SpeedScript* in this book, version 3.2, incorporates a year's worth of improvements, readers' suggestions, and additional debugging. This book contains all the documentation you need to use *SpeedScript* on the Commodore 64.

SpeedScript 3.2, though compact in size (6K), has many features found on commercial word processors. *SpeedScript* is also very easy to learn and use. You can start writing with it the first time you use it. You type in everything first; preview and make corrections on the screen; insert and delete words, sentences, and paragraphs; and then print out an error-free draft, letting *SpeedScript* take care of things like margins, centering, headers, and footers.

SpeedScript is a writing tool. It won't necessarily make you a better writer, but you may become a better writer once the tedium of retyping and erasing is replaced by the flexibility of a word processor. Words are no longer frozen in place by ink; they become free-floating entities. You no longer think about typewriting; you can stand back and work directly with words and ideas. The distinction between rough and final drafts becomes blurred as you perfect your writing while you write it.

Loading *SpeedScript*

SpeedScript can be loaded from the disk in one of two ways. You can load the menu program by entering

```
LOAD "MENU",8
```

and pressing RETURN. From the menu select *SpeedScript*. You'll notice two *SpeedScript* options. One is the regular version that uses the standard character set found in the computer's ROM. The second option loads in the regular version of *SpeedScript*, but allows you to select a different display font. More on the different fonts in the next article.

If you don't want to load *SpeedScript* from the menu, you can simply enter

LOAD "SS",8

and press RETURN. When the READY prompt appears, type RUN and press RETURN.

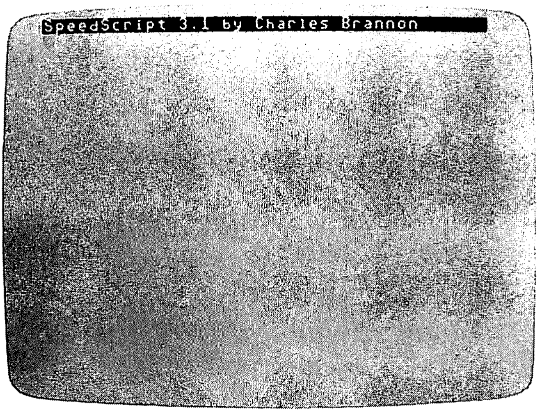
Although *SpeedScript* is an all machine language program, it can be copied to other disks just as you would copy a BASIC program. Thus, to make a copy of *SpeedScript,* load it into your 64 directly (not through the MENU program). Before you type RUN, insert the disk you wish to copy it to and enter

SAVE "SS",8

and press RETURN.

Entering Text

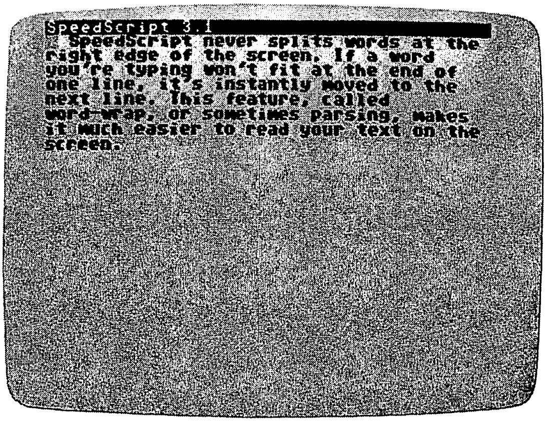
When you run *SpeedScript* on the Commodore 64, the screen colors change to dark gray on light gray. The first screen line is black with white letters. This *command line* is used to communicate with *SpeedScript. SpeedScript* presents all messages here. The remaining lines of the screen are used to enter, edit, and display your document. The *cursor* shows where the next character you type will appear on the screen. *SpeedScript* lets you move the cursor anywhere within your document, making it easy to find and correct errors.



```
SpeedScript 3.1 by Charles Brannon
```

The SpeedScript *screen before any keys have been pressed. The black line across the top is where* SpeedScript *presents messages.*

To begin using *SpeedScript,* just start typing. When the cursor reaches the right edge of the screen, it automatically jumps to the beginning of the next line, just as in BASIC. But unlike BASIC, *SpeedScript* never splits words at the right edge of the screen. If a word you're typing won't fit at the end of one line, it's instantly moved to the next line. This feature, called *word-wrap,* or sometimes *parsing,* makes it much easier to read your text on the screen. Even if you make numerous editing changes, *SpeedScript* reformats the screen and rewraps all words.



```
SpeedScript 3.1
 SpeedScript never splits words at the
right edge of the screen. If a word
you're typing won't fit at the end of
one line, it's instantly moved to the
next line. This feature, called
word-wrap, or sometimes parsing, makes
it much easier to read your text on the
screen.
```

SpeedScript *never splits a word between lines. This feature is called word-wrap.*

Scrolling and Screen Formatting

When you finish typing on the last screen line, *SpeedScript* automatically scrolls the text upward to make room for a new line at the bottom. This is similar to the way BASIC works, but with one exception: The screen can scroll both up *and* down. Imagine the screen as a 24-line window on a long, continuous document.

The Commodore 64 has more than 43K of text space available in memory, room enough for 20–40 printed pages of text. To check at any time how much space is left, press **CTRL-=** (hold down the CTRL key while pressing the = key). The number which appears in the command line indicates how much room remains for characters of text.

If you're used to a typewriter, you'll have to unlearn some habits. First, since the screen is only 40 columns wide, and most printers have 80-column carriages, it doesn't make sense to press RETURN at the end of each line as you do on a typewriter. *SpeedScript*'s word-wrap takes care of this automatically. Press RETURN only when you want to force a carriage return to end a paragraph or to limit the length of a line. To permit you to see these forced carriage returns, they appear on the screen as a left-pointing arrow. (This is called a *return mark* in this book.)

When you print your document, *SpeedScript* automatically formats your text to fit the width of the paper. Don't manually space over for a left margin or try to center a line yourself as you would on a typewriter. *SpeedScript*'s printing routine automatically takes care of all margins and lets you customize the margin settings. Also, don't worry about where a printed page will end. When printing, *SpeedScript* automatically fits your text onto separate pages and can even put short phrases and page numbers at the top or bottom of each page if you want.

Like all good word processors, *SpeedScript* has a wide selection of editing and convenience features. You can move the cursor a single space in either direction, or skip to the next or previous word, sentence, or paragraph. You can also move the cursor to the top of the screen, the top of the document, or the end of the document. The INST/DEL key is used to insert a single space or delete a single character. Other features let you erase a word, sentence, or paragraph, and move or copy sentences, words, and paragraphs to other places in your document. Using Search and Replace, you can find any phrase and even automatically change one phrase to another throughout the entire document.

You can save your text on disk, then load it later for additions and corrections. You can transpose (exchange) two characters, change the screen and text colors, send disk commands, read the disk error channel, and automatically tab over five spaces for paragraph indents. You don't need to learn all these commands right away, but you'll be glad they're available as you become more comfortable with word processing.

Using the Keyboard

Most of these features are accessed with control-key commands—you hold down CTRL while pressing another key. In this book, control-key commands are abbreviated **CTRL-*x*** (where *x* is the key you press in combination with CTRL). An example is the **CTRL-=** mentioned above to check on free memory. **CTRL-E** means hold down CTRL and press E. Sometimes you have to hold down both SHIFT and CTRL as you type the command key, as in SHIFT–CTRL-H. Other keys are referenced by name or function, such as back arrow for the left-pointing arrow in the top-left corner of the keyboard, pound sign for the British pound sign (£), CLR/HOME for the home cursor key, SHIFT–CLR/HOME for the clear screen key, f1 for special function key 1, and up arrow for the upward-pointing arrow to the left of the RESTORE key. See the next article for a complete quick-reference chart of all keyboard commands and the keyboard map.

Some keys let you move the cursor to different places in the document to make corrections or scroll text into view. *SpeedScript* uses a unique method of cursor movement that is related to writing, not programming. Programmers work with lines of text and need to move the cursor up and down a line or left and right across a line. *SpeedScript,* however, is oriented for writers. You aren't working with lines of text, but with a continuous document.

Therefore, *SpeedScript* moves the cursor by character, word, sentence, or paragraph. *SpeedScript* defines a word as any sequence of characters preceded or followed by a space. A sentence is any sequence of characters ending with a period, exclamation point, question mark, or return mark. And a paragraph is defined as any sequence of characters ending in a return mark. (Again, a return mark appears on the screen as a left-pointing arrow.)

Here's how to control the cursor:

- The **left/right cursor key** works as usual; pressing this key by itself moves the cursor right (forward) one space, and pressing it with SHIFT moves the cursor left (backward) one space.
- The **up/down cursor key** moves the cursor forward to the beginning of the next sentence. Pressing it with SHIFT moves the cursor backward to the beginning of the previous sentence.
- The **f1 special function key** moves the cursor forward to the beginning of the next word. The **f2 key** (hold down SHIFT and press f1) moves the cursor backward to the beginning of the previous word.

- The **f3 special function key** moves the cursor forward to the beginning of the next sentence (just like the up/down cursor key). The **f4 key** (hold down SHIFT and press f3) moves the cursor backward to the beginning of the previous sentence (just like pressing SHIFT and the up/down cursor key).
- The **f5 special function key** moves the cursor forward to the beginning of the next paragraph. The **f6 key** (hold down SHIFT and press f5) moves the cursor backward to the beginning of the previous paragraph.
- The **CLR/HOME key,** pressed once by itself, moves the cursor to the top of the screen without scrolling. Pressed twice, it moves the cursor to the beginning of the document.
- **CTRL-Z** moves the cursor to the bottom of the document.

Correcting Your Typing

One strength of a word processor is that you need never have mistakes in your printed document. Since you've typed everything before you print it, you have plenty of opportunities to proofread and correct your work. The easiest way to correct something is just to type over it, but there are other ways, too.

Sometimes you'll have to insert characters to make a correction. Maybe you accidentally dropped a letter, typing *hngry* instead of *hungry.* When you change the length of a word, you need to push over everything to the right of the word to make room for the insertion. Use **SHIFT-INST/DEL** to open up a single space, just as in BASIC. Merely position the cursor at the point where you want to insert a space, and press SHIFT-INST/DEL.

Insert Modes

It can be tedious to use the SHIFT-INST/DEL key to open up enough space for a whole sentence or paragraph. For convenience, *SpeedScript* has an insert mode that automatically inserts space for each character you type. In this mode, you can't type over characters; everything is inserted at the cursor position. To enter insert mode, press **CTRL-I.** To cancel insert mode, press CTRL-I again (a command key that turns something on and off is called a *toggle*). To let you know you're in insert mode, the normally black command line at the top of the screen turns blue.

Insert mode is the easiest way to insert text, but it can become too slow when working with a very long document because it must move *all* the text following the cursor position. Although *SpeedScript* uses turbocharged memory-move routines, the 6502/6510 microprocessor can go only so fast. So *SpeedScript* has even more ways to insert blocks of text.

One way is to use the **RUN/STOP** key. It is programmed in *SpeedScript* to act as a five-space margin indent. To end a paragraph and start another, press RETURN twice and press RUN/STOP. Alternatively, you can press **SHIFT-RETURN,** which does this automatically. You can use RUN/STOP to open up more space than SHIFT–INST/DEL. No matter how much space you want to insert, each insertion takes the same amount of time. So the RUN/STOP key can insert five spaces five times faster than pressing SHIFT–INST/DEL five times.

There's an even better way, though. Press **SHIFT-RUN/STOP** to insert 255 spaces. This is enough room for a sentence or two. You can press it several times to open up as much space as you need. And SHIFT–RUN/STOP is *fast.* (You don't want to be in insert mode when you use this trick; that would defeat its purpose.)

Since the INST/DEL key is also slow when working with large documents (it, too, must move all text following the cursor), you may prefer to use the back-arrow (←) key to backspace. The **back-arrow** key by itself moves the cursor left one space and blanks out that position. It's more like a backspace than a delete.

After you're finished inserting with these methods, there will probably be some inserted spaces left over that you didn't use. Just press **SHIFT-CTRL-back arrow**. This instantly deletes all extra spaces between the cursor and the start of the following text. SHIFT–CTRL–back arrow is also generally useful whenever you want to delete a bunch of spaces.

Erasing Text

Inserting and retyping are not the only kinds of corrections you'll need to make. Part of writing is separating the wheat from the chaff. On a typewriter, you pull out the paper, ball it up, and dunk it in the trash can. *SpeedScript* lets you be more selective.

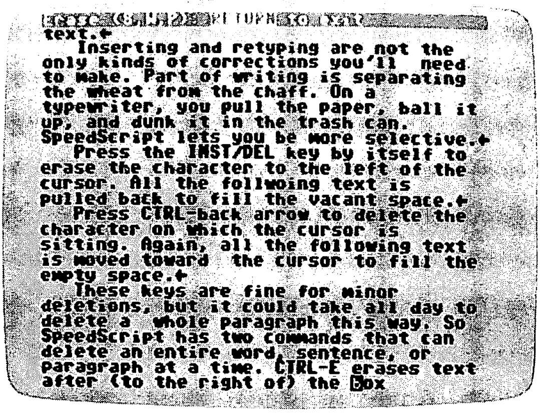
Press the **INST/DEL** key by itself to erase the character to the left of the cursor. All the following text is pulled back to fill the vacant space.

Press **CTRL–back arrow** to delete the character on which the cursor is sitting. Again, all the following text is moved toward the cursor to fill the empty space.

These keys are fine for minor deletions, but it could take all day to delete a whole paragraph this way. So *SpeedScript* has two commands that can delete an entire word, sentence, or paragraph at a time. **CTRL-E** erases text *after* (to the right of) the cursor position, and **CTRL-D** deletes text *behind* (to the left of) the cursor.

To use the **CTRL-E** erase mode, first place the cursor at the beginning of the word, sentence, or paragraph you want to erase. Then press CTRL-E. The command line shows the message "Erase (S,W,P): RETURN to exit." Press S to erase a sentence, W for a word, or P for a paragraph. Each time you press one of these letters, the text is quickly erased. You can keep pressing S, W, or P until you've erased all the text you wish. Then press RETURN to exit the erase mode.

The **CTRL-D** delete mode works similarly, but deletes only one word, sentence, or paragraph at a time. First, place the cursor after the word, sentence, or paragraph you want to delete. Then press CTRL-D. Next, press S, W, or P for sentence, word, or paragraph. The text is immediately deleted and you return to editing. You don't need to press RETURN to exit the CTRL-D delete mode unless you pressed this key by mistake. *(In general, you can escape from any command in* Speed-Script *by simply pressing RETURN.)* CTRL-D is most convenient when the cursor is already past what you've been typing.



```
[illegible]
text.←
   Inserting and retyping are not the
only kinds of corrections you'll  need
to make. Part of writing is separating
the wheat from the chaff. On a
typewriter, you pull the paper, ball it
up, and dunk it in the trash can.
SpeedScript lets you be more selective.←
   Press the INST/DEL key by itself to
erase the character to the left of the
cursor. All the follwoing text is
pulled back to fill the vacant space.←
   Press CTRL-back arrow to delete the
character on which the cursor is
sitting. Again, all the following text
is moved toward  the cursor to fill the
empty space.←
   These keys are fine for minor
deletions, but it could take all day to
delete a  whole paragraph this way. So
SpeedScript has two commands that can
delete an entire word, sentence, or
paragraph at a time. CTRL-E erases text
after (to the right of) the Box
```

You can delete a whole word, sentence, or paragraph by pressing CTRL-E or CTRL-D.

The Text Buffer

When you erase or delete with CTRL-E and CTRL-D, the text isn't lost forever. *SpeedScript* remembers what you've removed by storing deletions in a separate area of memory called a *buffer*. The buffer is a fail-safe device. If you erase too much or change your mind, just press **CTRL-R** to restore the deletion. However, be aware that *SpeedScript* remembers only the last erase or delete you performed.

Another, more powerful use of this buffer is to move or copy sections of text. To move some text from one location in your document to another, first erase or delete it with CTRL-E or CTRL-D. Then move the cursor to where you want the text to appear and press **CTRL-R.** CTRL-R instantly inserts the contents of the buffer at the cursor position. If you want to copy some text from one part of your document to another, just erase or delete it with CTRL-E or CTRL-D, restore it at the original position with CTRL-R, then move the cursor elsewhere and press CTRL-R to restore it again. You can retrieve the buffer with CTRL-R as many times as you like.

Important: The CTRL-E erase mode lets you erase up to the maximum size of the buffer (12K, or over 12,000 characters, and CTRL-E also removes the previous contents of the buffer. Keep this in mind if there's something in the buffer you'd rather keep. If you don't want the buffer to be erased, press **SHIFT-CTRL-E**. This preserves the buffer contents and adds newly erased text to the buffer.

Now you can see why CTRL-D lets you delete only a single sentence, word, or paragraph at a time. If it didn't, the deleted text would be added to the end of the buffer, and when you pressed CTRL-R to retrieve the buffer, the deleted text would be out of order (since CTRL-D deletes backward).

If you ever need to erase the contents of the buffer, press **CTRL-K** (remember *kill buffer*).

It's relatively easy to move blocks of text between documents. Using the buffer, you can load one document, erase some text into the buffer, load another document, then insert the buffer. You can also use the buffer to save an often-used word or phrase, then repeat it whenever you need it.

The Wastebasket Command

If you want to start a new document or simply obliterate all your text, press **SHIFT-CLR/HOME**. *SpeedScript* asks, "ERASE ALL TEXT: Are you sure? (Y/N)." This is your last chance. If you *don't* want to erase the entire document, press N or any other key. Press Y to perform the irreversible deed. There is no way to recover text wiped out with Erase All.

The RUN/STOP–RESTORE reset combination on the Commodore 64 has been disabled in *SpeedScript*.

As mentioned above, pressing RUN/STOP by itself inserts five spaces for indenting paragraphs.

Pressing just RESTORE brings up the message "Exit *SpeedScript*: Are you sure? (Y/N)." If you press Y for yes, you exit to BASIC (if you press N or any other key at the prompt, you return to editing text with no harm done). Once in BASIC you'll still have one chance to reenter *SpeedScript* without losing your text—simply enter RUN (but your chances decrease if you execute other commands in BASIC).

Search and Replace

Here's another feature only a computer can bring to writing. *SpeedScript* has a Hunt command that searches through your document to find a selected word or phrase. A Replace option lets you automatically change one word to another throughout the document. Since on the 64, CTRL-S is synonymous with the CLR/HOME key (try it), and since *SpeedScript* already uses CTRL-R, I have to resort to command keys which are slightly less than mnemonic for these functions.

SHIFT-CTRL-H activates the Hunt feature, **SHIFT-CTRL-J** (J is used because it's next to the H) lets you selectively hunt and replace, and **CTRL-G** (Global) is for automatically searching and replacing.

Searching for something is a two-step process. First, you need to tell *SpeedScript* what to search for, then you trigger the actual search. Press SHIFT–CTRL-H. The command line says "Hunt for:". Type in what you'd like to search for, the *search phrase,* up to 29 characters. *SpeedScript* remembers the search phrase until you change it. (Incidentally, when you are typing on the command line, the only editing key that works is the INST/DEL key for backing up. *SpeedScript* does not let you enter control codes or cursor controls when you type in the command line, and you can type no more than one screen line.) Press RETURN when you've finished typing. If you press RETURN alone without typing anything, the Hunt command is canceled.

When you are ready to search, press **CTRL-H.** *SpeedScript* looks for the next occurrence of the search phrase *starting from the current cursor position.* If you want to hunt through the entire document, press CLR/HOME twice to move the cursor to the very top before beginning the search. Each time you press CTRL-H, *SpeedScript* looks for the next occurrence of the search phrase and places the cursor at the start of the phrase. If the search fails, you'll see the message "Not Found."

CTRL-J (Replace) works together with CTRL-H. After you've specified the search phrase with SHIFT–CTRL-H, press SHIFT–CTRL-J to select the replace phrase. *SpeedScript* also remembers this replace phrase until you change it. (You can press RETURN alone at the "Replace with:" prompt to select a *null* replace phrase. When you hunt and replace, this deletes the located phrase.) To search and replace manually, start by pressing CTRL-H. After *SpeedScript* finds the search phrase, press CTRL-J if you want to replace the phrase. If you don't want to replace the phrase, don't press CTRL-J. You are not in a special search and replace mode. You're free to continue writing at any time.

CTRL-G links CTRL-H and CTRL-J together. It first asks "Hunt for:", then "Replace with:", then automatically searches and replaces throughout the document starting at the cursor position.

A few hints and cautions: First, realize that if you use *the* as the search phrase, *SpeedScript* dutifully finds the embedded *the* in words like *therefore* and *heathen.* If you changed all occurrences of *the* to *cow,* these words would become *cowrefore* and *heacown.* If you want to find or replace a single word, include a space as the first character of the word, since almost all words are preceded by a space. Naturally, if you are replacing, you need to include the space in the replace phrase, too. Also, *SpeedScript* distinguishes between uppercase and lowercase. The word *Meldids* does not match with *meldids. SpeedScript* will not find a capitalized word unless you capitalize it in the search phrase. To cover all bases, you will sometimes need to make two passes when replacing a word. Keep these things in mind when using CTRL-G since you don't have a chance to stop an out-of-control search and replace.

Storing Your Document

Another advantage of word processing is that you can store your writing on disk. A Commodore disk, with 170K of storage space, can store 80–150 pages of text as several document files.

SpeedScript can also be used as a simple database manager. Type in the information you need, then store it as a *SpeedScript* document. The search feature lets you quickly find information, especially if you use graphics characters to flag key lines. You can search for the graphics characters and quickly skip from field to field.

It's easy to store a document. First, make sure your disk drive is plugged in and functioning. Insert a formatted (NEWed) disk into the drive. Press **f8** (SHIFT-f7). You'll see the prompt "Save:". Type in a filename for your document. A filename can be up to 16 characters long and can include almost any characters, but do not use question marks or asterisks. You cannot use the same name for two different documents on a single disk. To replace a document already on disk using the same filename, precede your filename with the characters **@0:** or **@:**. You can also precede the filename with either **0:** or **1:** if you use a dual disk drive. *SpeedScript* cannot access a second disk drive with a device number of 9.

After entering the filename, answer the prompt "Tape or Disk" by pressing either the **T** or **D** key. You can cancel the Save command by pressing RETURN without typing anything else at either the "Save:" or "Tape or Disk?" prompt.

Press D for disk. If the disk is formatted and has room, your file is stored relatively quickly. After the SAVE, *SpeedScript* reports "No errors" if all is well, or reads and reports the disk error message if not.

Loading a Document

To recall a previously saved document, press **f7**. Answer the "Load:" prompt with the filename. (As explained later in this article, you can use CTRL-4 to check the disk directory for the desired filename.) Insert the disk with the file you want to load and press D. *SpeedScript* loads the file and should display "No errors." Otherwise, *SpeedScript* reads the error channel of the disk drive.

The position of the cursor before loading a file is important. *SpeedScript* starts loading at the cursor position, so be sure to press CLR/HOME twice or SHIFT–CLR/HOME (Erase All) to move the cursor to the start of text space, unless you want to merge two documents. When you press f7 to load, the command line turns green to warn you if the cursor is not at the top of the text space.

To merge two or more files, simply load the first file, press CTRL-Z to move the cursor to the end of the document, and then load the file you want to merge. Do not place the cursor somewhere in the middle of your document before loading. A LOAD does not insert the characters coming in from disk into your old text, but overwrites all existing text after the cursor position. The last character loaded becomes the new end-of-text marker, and you cannot access any of your old text that may appear after this marker.

File Compatibility

SpeedScript documents are stored as program files (a PRG type on disk). Naturally, you can't load and run a *SpeedScript* file from BASIC. The characters are stored in their screen code (POKE) equivalents. Several commercial word processors store text similarly, including *WordPro 3+* and *PaperClip*. As a matter of fact, two commercial spelling checkers designed for *WordPro* also work with *SpeedScript*: *SpellRight Plus* (from Professional Software) and *SpellPro 64* (from Pro-Line).

The *SpeedScript* File Conversion utility has a filename of SSFC (it can be loaded from the menu, which has a more descriptive name). The File Conversion utility translates *SpeedScript* screen-code program files into either Commodore ASCII or true ASCII. These translated files are stored in SEQuential format, the file type used in most file-processing applications. The file converter program can also translate a Commodore ASCII sequential file into a screen-code *SpeedScript* program file. You can use the file converter to translate a database into a *SpeedScript* file (or vice versa), and you can convert *SpeedScript* files to true ASCII and use a modem program to upload them to another computer.

SpeedCalc (Chapter 3) saves data in its own format. It is possible, though, to load *SpeedCalc* files into *SpeedScript* if you first *print* a *SpeedCalc* file to *disk* (to create an ASCII file), then use the file converter. See Chapter 3 for details on how to print *SpeedCalc* files to disk.

Disk Commands

Sometimes you forget the name of a file or need to scratch or rename a file. *SpeedScript* gives you full control over the disk drive. To view the disk directory, press **CTRL-4**. The directory will be displayed on the screen without affecting the text in memory. You can press any key to pause scrolling. Afterward, press RETURN to switch back to your text. All the other disk commands are also accessible. Just press **CTRL-↑** (up arrow), then type in a 1541 disk command. You don't need to type PRINT#15 or any quotation marks as you do in BASIC, just the actual command. If you press RETURN without typing a disk command, *SpeedScript* displays the disk status. It also displays the status after completing a disk command. Here is a quick summary of disk commands:

n:*disk name,ID* This formats (NEWs) a disk. You must format a new disk before using it for the first time. The disk name can be up to 16 characters. The ID (identifier) is any two characters. You must use a unique ID for each disk you have. Don't forget that this command erases any existing data on a disk.

s:*filename* Scratches (deletes) a file from the disk.

r:*newname=oldname* Changes the name of file *oldname* to *newname.*

c:*backup filename=original name* Creates a new file (the backup copy) of an existing file (original copy) on the same disk.

i: Initializes a disk. This resets several disk variables and should be used after you swap disks or when you have trouble reading a disk.

v: Validates a disk. This recomputes the number of available blocks and can sometimes free up disk space. Always use Validate if you notice a filename on the directory flagged with an asterisk. Validate can take awhile to finish.

uj: Resets the disk drive to power-up state.

Additional Features

SpeedScript has a few commands that don't do much, but are nice to have. **CTRL-X** exchanges the character under the cursor with the character to the right of the cursor. Thus, you can fix transposition errors with a single keystroke. **CTRL-A** changes the character under the cursor from uppercase to lowercase or vice versa. You can hold down CTRL-A to continue changing the following characters.

Press **CTRL-B** to change the background and border colors. Each time you press CTRL-B, one of 16 different background colors appears. Press **CTRL-L** to cycle between one of 16 character (lettering) colors. The colors are preserved until you change them. In fact, if you exit and resave *SpeedScript,* the program will load and run with your color choice in the future.

Printing

If you already think *SpeedScript* has plenty of commands, wait until you see what the printing package offers. *SpeedScript* supports an array of powerful formatting features. It automatically fits your text between left and right margins that you can specify. You can center a line or block it against the right margin. *SpeedScript* skips over the perforation on continuous-form paper, or it can wait for you to insert single-sheet paper. A line of text can be printed at the top of each page (a *header)* and/or at the bottom of each page (a *footer),* and it can include automatic page numbering, starting with whatever number you like.

SpeedScript can print on different lengths and widths of paper, and single-, double-, triple-, or any-spacing is easy. You can print a document as big as can fit on a disk by linking several files together during printing. You can print to the screen or to a sequential disk file instead of to a printer. Other features let you print to most printers using most printer interfaces, and send special codes to the printer to control features like underlining, boldfacing, and double-width type (depending on the printer).

But with all this power comes the need to learn additional commands. Fortunately, *SpeedScript* sets most of these variables to a default state. If you don't change these settings, *SpeedScript* assumes a left margin of 5, a right-margin position of 75, no header or footer, single-spacing, and continuous-paper page feeding. To begin printing, simply press **CTRL-P.** If your printer is attached, powered on, and selected (online), *SpeedScript* begins printing immediately. To cancel printing, hold down the **RUN/STOP** key until printing stops.

Before printing, be sure the paper in your printer is adjusted to top-of-form (move the paper perforation just above the printing element). CTRL-P assumes a Commodore printer, so it's helpful if your interface simulates the modes and codes of the Commodore 1525, MPS-801, or 1526 printer. CTRL-P prints with a device number of 4 and a secondary address of 7 (uppercase/lowercase mode).

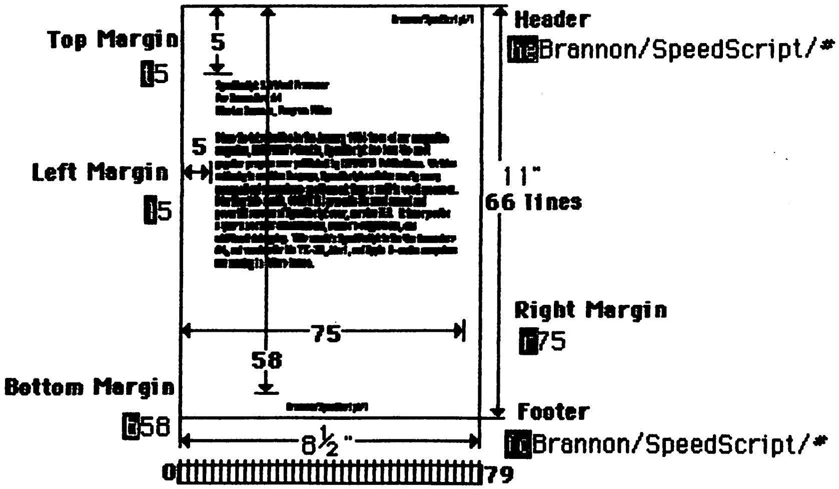
If CTRL-P doesn't work for you, try another variation, **SHIFT-CTRL-P.** Answer the prompt "Print to: Screen, Disk, Printer?" with the single letter **S, D,** or **P.** Press any other key to cancel the command.

If you press P for printer, *SpeedScript* requests two more keystrokes. First, answer "Device number" with a number from 4 through 7. This lets you print to one of several printers addressed with different device numbers. Next, answer "Secondary Address?" with a number from 0 through 9.

Figure 2-1.

Graphic Representation of Margin Settings

Values shown are default settings.



Non-Commodore Printers

The secondary address is used on most non-Commodore printer interfaces to control special features. For example, you can bypass the emulation features and use graphics mode to communicate directly with your printer (see the true ASCII command below). Consult the list of secondary addresses in your printer interface manual. *SpeedScript* does not work properly with RS-232 serial printers or interfaces.

An additional note: Some printers and interfaces incorporate an automatic skip-over-perforation feature. The printer skips to the next page when it reaches the bottom of a page. Since *SpeedScript* already controls paper feeding, you need to turn off this automatic skip-over-perf feature (usually, by sending out control codes) before running *SpeedScript*, or paging won't work properly. Remember, sometimes the printer controls the skip-over-perf feature, sometimes the interface, and sometimes even both.

I've successfully tested the Commodore 64 version of *SpeedScript* with the following printers: Commodore 1525, MPS-801, Commodore 1526 (second revision), Prowriter/C. Itoh 8510, Epson MX-80, Gemini 10-X, Star SG-10 (and SG-10C), Okimate-10, Okidata 82, Okidata 92, and Hush-80 CD.

I've also successfully tested *SpeedScript* with these printer interfaces: Cardco A/B/G+ and G Whiz, Tymac Connection, Xetec, TurboPrint, and MW-350.

SpeedScript should work even if your printer or interface is not on this list. These are just the ones I've tested.

Be sure your printer or interface supplies its own linefeeds. Again, consult your manuals and insure that either your printer or interface (but not both) supplies an automatic linefeed after carriage return. To test this, print a small sample of text with CTRL-P. Since the default is single-spacing, you should not see double-spacing, nor should all printing appear on the same line. If you still aren't getting linefeeds, use the linefeed command discussed below.

Printing to Screen and Disk

SHIFT–CTRL-P prints to the screen when you press S. The screen colors change to white letters on a black background, and what appears on the screen is exactly what would print on the printer. It takes two screen lines to hold one 80-column printed line, of course. If you use double-spacing (see below), it's much easier to see how each line is printed. With this screen preview, you can see where lines and pages break. To freeze printing, hold down either SHIFT key or engage SHIFT LOCK. The border color changes to white while SHIFT is held down. When printing is finished, press any key to return to editing.

SHIFT–CTRL-P prints to a disk file when you press D. Enter the filename when requested. *SpeedScript* sends out all printer information to a sequential file. You can use other programs to process this formatted file. Try this simple example:

```
10 OPEN 1,4
20 OPEN 2,8,8,"filename"
30 GET#2,A$:SS=ST: PRINT#1,A$;: IF SS=0 THEN 30
40 PRINT#1: CLOSE1
50 CLOSE2
```

This program dumps the disk file specified by the filename in line 20 to any printer. You can use it to print *SpeedScript* files (produced with SHIFT–CTRL-P) on another Commodore computer and printer without running *SpeedScript*. Change line 10 to **OPEN 1,2,0,CHR$(6)** to dump the file to a 300-baud modem or RS-232 printer, or **OPEN 1,3** to display it on the screen.

Formatting Commands

The print formatting commands must be distinguished from normal text, so they appear onscreen in reverse field with the text and background colors switched. You enter these reverse-video letters by pressing **CTRL-£** (pound sign) or **CTRL-3,** which is easier to type with one hand. Answer the prompt "Enter format key:" by pressing a single key. This key is inserted into text in reverse video. All lettered printer commands should be entered in lowercase (unshifted). During printing, *SpeedScript* treats these characters as printing commands. (See the next article for a quick-reference chart of format commands.)

There are two kinds of printing commands, which I'll call Stage 1 and Stage 2. Stage 1 commands usually control variables such as left margin and right margin. Most are followed by a number, with no space between the command and the number. Stage 1 commands are executed before a line is printed.

Stage 2 commands, like centering and underlining, are executed while the line is being printed. Usually, Stage 1 commands must be on a line of their own, although you can group several Stage 1 commands together on a line. Stage 2 commands are by nature embedded within a line of text. A sample Stage 1 line could look like this:

```
l10r50s2
```

Embedded Stage 2 commands look like this:

```
cThis line is centered.←
This is uunderliningu.←
```

Stage 1 Commands

l Left margin. Follow with a number from 0 to 255. Use 0 for no margin. Defaults to 5. See Figure 2-1 for an illustration of margin settings.

r Right margin position, a number from 1 to 255. Defaults to 75. Be sure the right-margin value is greater than the left-margin value, or *SpeedScript* will go bonkers. Some printer interfaces force a certain printing width, usually 80 characters wide. You'll need to disable this in order to permit *SpeedScript* to print lines longer than 80 characters.

t Top margin. The position at which the first line of text is printed, relative to the top of the page. Defaults to 5. The header (if any) is always printed on the first line of the page, before the first line of text.

b Bottom margin. The line at which printing stops before continuing to the next page. Standard 8-1/2 × 11 inch paper has 66 lines. Bottom margin defaults to the fifty-eighth line. The footer (if any) is always printed on the last line of the page, after the last line of text.

p Page length. Defaults to 66. If your printer does not print six lines per inch, multiply lines-per-inch by 11 to get the page length. European paper is usually longer than American paper—11-5/8 or 12 inches. Try a page length of 69 or 72.

s Spacing. Defaults to single-spacing. Follow with a number from 1 to 255. Use 1 for single-spacing, 2 for double-spacing, 3 for triple-spacing.

@ Start numbering *at* page number given. Page numbering normally starts with 1.

? Disables printing until selected page number is reached. For example, a value of 3 would start printing the third page of your document. Normally, *SpeedScript* starts printing with the first page.

x Sets the page width, in columns (think *a cross*). Defaults to 80. You need to change this for the sake of the centering command if you are printing in double-width or condensed type, or if you're using a 40-column or wide-carriage printer.

n Forced paging. Normally, *SpeedScript* prints the footer and moves on to the next page only when it has finished a page, but you can force it to continue to the next page by issuing this command. It requires no numbers.

m Margin release. Disables the left margin for the next printed line. Remember that this executes before the line is printed. It's used for outdenting.

a True ASCII. Every character is assigned a number in the ASCII (American Standard Code for Information Interchange) character set. Most printers use this true ASCII standard, but Commodore printers exchange the values for uppercase and lowercase to match Commodore's own variation of ASCII. Some printer interfaces do not translate Commodore ASCII into true ASCII, so you need to use this command to tell *SpeedScript* to translate. Also, you will sometimes want to disable your interface's emulation mode intentionally in order to control special printer features that would otherwise be rejected by emulation. Place this command as the first character in your document, even before the header and footer definitions. Don't follow it with a number.

Since, in effect, the true ASCII command changes the case of all letters, you can type something in lowercase and use true ASCII to make it come out in uppercase.

w Page wait. Like the true ASCII command, this one should be placed at the beginning of your document before any text. With page wait turned on, *SpeedScript* prompts you to "Insert next sheet, press RETURN" when each page is finished printing. Insert the next sheet, line it up with the printhead, then press RETURN to continue. Page wait is ignored during disk or screen output.

j Select automatic linefeeds after carriage return. Like **a** and **w**, this command must be placed before any text. Don't use this command to achieve double-spacing, but only if all text prints on the same line.

i Information. This works like REM in BASIC. You follow the command with a line of text, up to 255 characters, ending in a return mark. This line will be ignored during printing; it's handy for making notes to yourself such as the filename of the document.

h Header define and enable. The header must be a single line of text ending with a return mark (up to 254 characters). The header prints on the first line of each page. You can include Stage 2 commands such as centering and page numbering in a header. You can use a header by itself without a footer. The header and footer should be defined at the top of your document, before any text. If you want to prevent the header from printing on the first page, put a return mark by itself at the top of your document before the header definition.

f Footer define and enable. The footer must be a single line of text ending in a return mark (up to 254 characters). The footer prints on the last line of each page. As with the header, you can include Stage 2 printing commands, and you don't need to set the header to use a footer.

g GOTO (link) next file. Put this command as the last line in your document. Follow the command with the letter *D* for disk or *T* for tape, then a colon (:), then the name of the file to print next. After the text in memory is printed, the link command loads the next file into memory. You can continue linking in successive files, but don't include a link in the last file. Before you start printing a linked file, make sure the first of the linked files is in memory. When printing is finished, the last file linked to will be in memory.

Stage 2 Commands

These commands either precede a line of text or are embedded within one.

c Centering. Put this at the beginning of a line you want to center. This will center only one line, ending in a return mark. Repeat this command at the beginning of every line you want centered. Centering uses the page-width setting (see above) to center the line properly. To center a double-width line, either set the page width to 40 or pad out the rest of the line with an equal number of spaces. If you use double-width, remember that the spaces preceding the centered text will be double-wide spaces.

e Edge right. This command will cause a line to be aligned with the right margin when it is printed. That is, spaces will be inserted in front of the line so that the last character in the line will be printed in at the right margin column. Place the command at the beginning of the line you want aligned. This will align only one line, ending with a return mark. Repeat this command at the beginning of every line you want aligned right. Note that this is *not* the same as *right justification*—a feature found on some word processors that adjusts printing to align both the left and right margins. The edge right command aligns only one line, and only at the right margin. *SpeedScript* has no right justification feature.

When *SpeedScript* encounters this command, it prints the current page number. You usually embed this within a header or footer.

u A simple form of underlining. It does not work on Commodore printers, but only on printers that recognize CHR$(8) as a backspace and CHR$(95) as an underline character. Underlining works on spaces, too. Use the first **u** to start underlining and another one to turn off underlining.

Fonts and Styles

Most dot-matrix printers are capable of more than just printing text at ten characters per inch. The Commodore MPS-801 can print in double-width and reverse field. Some printers have several character sets, with italic and foreign language characters. Most can print in double-width (40 characters per line), condensed (132 characters per line), and in either pica or elite. Other features include programmable characters, programmable tab stops, and graphics modes. Many word processors customize themselves to a particular printer, but *SpeedScript* was purposely designed not to be printer-specific. Instead, *SpeedScript* lets you define your own Stage 2 printing commands.

You define a programmable *printkey* by choosing any character that is not already used for other printer commands. The entire uppercase alphabet is available for printkeys, and you can choose letters that are related to their function (like *D* for double-width). You enter these commands like printer commands, by first pressing **CTRL-3** on the Commodore 64.

To define a printkey, just press CTRL-3, then the key you want to assign as the printkey, then an equal sign (=), and finally the ASCII value to be substituted for the printkey during printing. For example, to define the + key as the letter *z,* you first look up the ASCII value of the letter *z* (in either your printer manual or user's manual). The ASCII value of the letter *z* is 90, so the definition is

```
⊞=90
```

Now, anywhere you want to print the letter *z,* substitute the printkey:

```
Gad⊞ooks! The ⊞oo is ⊞any!
```

This would appear on paper as

```
Gadzooks! The zoo is zany!
```

More practically, look up the value of reverse-on and reverse-off. Reverse-on, a value of 18, prints all text in reverse video until canceled by reverse-off (a value of 146) or a carriage return. So define SHIFT-R as 18 and SHIFT-O as 146. Anywhere you want to print a word in reverse, bracket the word with printkey R and printkey O.

You can similarly define whatever codes your printer uses for features like double-width or emphasized mode. For your convenience, four of the printkeys are predefined, though you can change them. Printkey 1 is defined as a 27, the value of the ESCape code used to precede many two-character printer commands. (With some printer interfaces, you must send two ESCape codes to bypass the interface's emulation.) For example, the Epson command for double strike is ESC-G. You can select it in *SpeedScript* with

```
▮1G
```

Printkey 2, a value of 14, goes into double-width mode on most printers, and printkey 3, a value of 15, turns off double-width on some printers and selects condensed mode on others. Printkey 4 is defined as 18, which selects reverse field with

Commodore printers (and on some graphics interfaces in emulation mode) or condensed mode on some other printers.

With so many codes available, you can even design custom logos and symbols using your printer's graphics mode. For example, on the 1525/MPS-801, you can draw a box (perhaps for a checklist) by first setting the appropriate codes:

```
1=82=253=2554=193←
```

Then display the box with text by typing

```
1344432 Toothpaste←
```

This appears on paper as

```
□ Toothpaste
```

Keep one thing in mind about printkeys. *SpeedScript* always assumes it is printing to a rather dumb, featureless printer, the least common denominator. *SpeedScript* doesn't understand the intent of a printkey; it justs sends its value out. So if you make one word within a line double-width, it may make the line overflow the specified right margin. There's no way for *SpeedScript* to include built-in font and type-style codes without being customized for a particular printer since no set of codes is universal to all printers.

Hints and Tips

It may take you awhile to fully master *SpeedScript,* but as you do you'll discover many ways to use the editing and formatting commands. For example, there is a simple way to simulate tab stops, say, for a columnar table. Just type a period at every tab-stop position. Erase the line, then restore it multiple times. When you are filling in the table, just use word left/word right to jump quickly between the periods. Or you can use the programmable printkeys to embed your printer's own commands for setting and jumping to tab stops.

You don't have to change or define printer commands every time you write. Just save these definitions as a small text file, and load this file in each time you write. You can create many custom definition files and have them ready to use on disk. You can create customized "fill in the blank" letters. Just type the letter, and everywhere you'll need to insert something, substitute a graphics symbol. When you're ready to customize the letter, just hunt for each graphics symbol and insert the specific information.

SpeedScript does not work with any 80-column video boards or software 80-column emulators. *SpeedScript* also wipes out most kinds of resident (RAM-loaded) software, including most software-simulated printer drivers. However, you can print to disk using SHIFT–CTRL-P, then dump the disk file to the printer from BASIC.

The following pages include several helpful clip-out charts for use with *SpeedScript.*

Figure 2-2. *SpeedScript* Keyboard Map

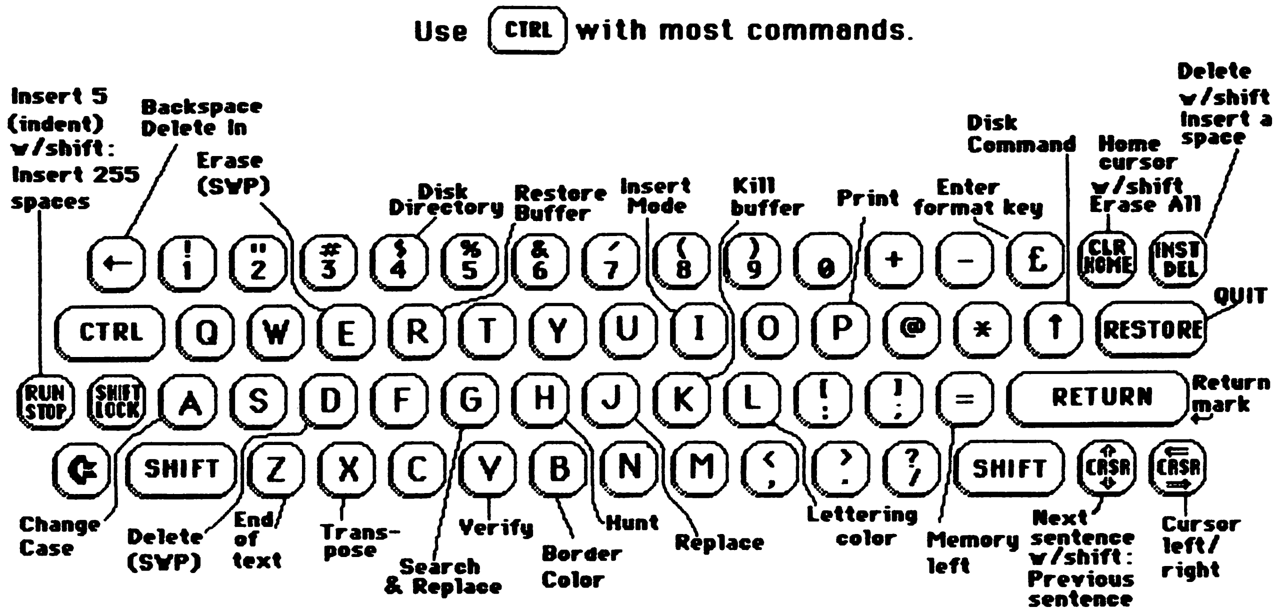


Figure 2-3. Clip-Out Quick-Reference Card—Format Commands

Enter these commands with CTRL-£ or CTRL-3:

Command	Description	Default
a	True ASCII	off
b	Bottom Margin	58
c	Centering	
e	Edge Right	
f	Footer	
g	Goto Linked File*	
h	Header	
i	Information*	
j	Select linefeeds*	
l	Left Margin	5
m	Margin Release *	

Command	Description	Default
n	Next Page	
p	Page Length *	66
r	Right Margin	75
s	Spacing	1
t	Top Margin	5
u	Underline toggle	
w	Page Wait	
x	Columns across*	80
@	Initial page # *	1
?	Skip pages *	
#	Print page number	

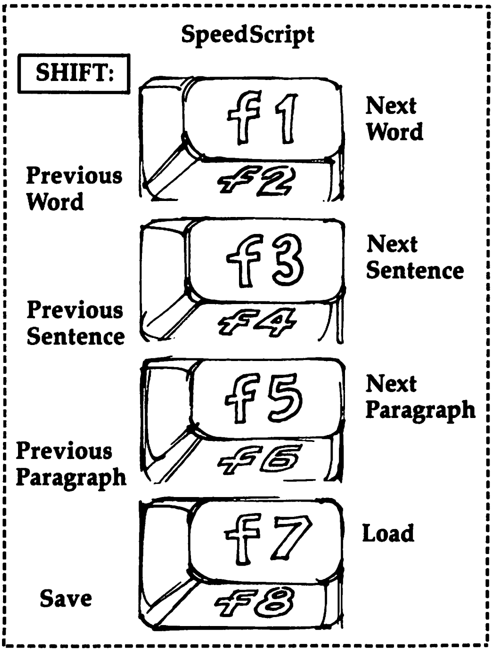
hcSpeedScript/#←	Centered Header with page number
l10r70s2←	Left margin 10, right margin 70, double spacing.
gD:SpeedScript.2←	Goto and continue printing with filename "SpeedScript.2"

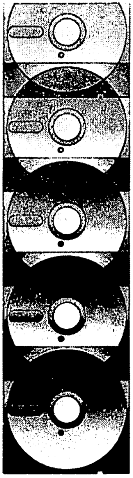
* Notes command changed or added since Version 2.0

Table 2-1. Clip-Out Quick-Reference Card—Editing Commands

CTRL A	Change case
CTRL B	Change border color
CTRL D	Delete (Sentence, Word, Paragraph)
CTRL E	Erase (Sentence, Word, Paragraph)
CTRL G	Global search and replace
CTRL H	Hunt for phrase With SHIFT: Select hunt phrase
CTRL I	Enter/exit insert mode
CTRL J	Replace With SHIFT: Select replace phrase
CTRL K	Kill buffer
CTRL L	Change text character color
CTRL P	Print
CTRL R	Restore buffer
CTRL V	Verify
CTRL X	Transpose characters
CTRL Z	Go to end of text
CTRL =	Display amount of free memory
CTRL ↑	Send disk command or read error channel
CTRL 4	Display disk directory
CTRL £	Enter format (printer) commands
CTRL 3	Same as CTRL-£
CLR/HOME	Press once to go to top of screen Hold down to go to top of text With SHIFT: Erase all text
CRSR (left/right)	Move the cursor left one character With SHIFT: Move the cursor right one character
CRSR (up/down)	Go to next sentence With SHIFT: Go to previous sentence
RUN/STOP	Indent 5 spaces With SHIFT: Insert 255 spaces
RESTORE	Exit *SpeedScript*
←	Backspace With CTRL: Delete character under cursor and close up text With SHIFT and CTRL: Delete all spaces from cursor to next character
RETURN	Return mark With SHIFT: End paragraph, add an extra return mark, and indent next paragraph
INST/DEL	Delete character With SHIFT: Insert space

Figure 2-4. Clip-Out Function-Key Overlay





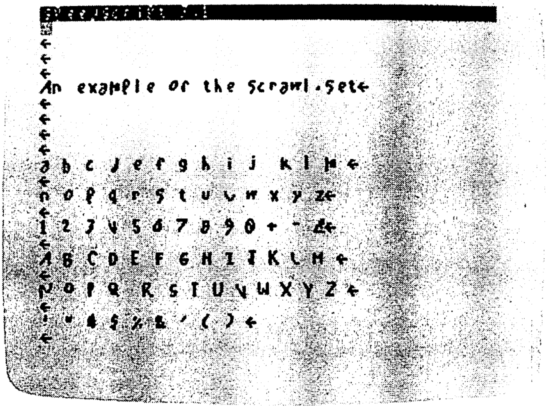
Using Special Fonts with *SpeedScript*

Writing with a word processor often means staring for hours at a video screen. For word processing, screen clarity is especially vital. It's best to have a good-quality color or monochrome monitor, but a clear, readable character set helps, too. Commodore's built-in character set works well and is especially designed for the low resolution of the average TV. However, it can be improved.

Included on your *Complete 64* disk are six custom character sets. Using them is easy. Rather than loading *SpeedScript* from the menu, select the option that allows custom characters. After a few seconds you'll see a message that *SpeedScript* is loading. Once *SpeedScript* is loaded, you can select which font you want to use. These are the fonts included on the disk:

1. **scrawl.set**
2. **computer.set**
3. **large.set**
4. **speed.set**
5. **serif.set**
6. **smallcaps.set**

Press the number of the font you want and hit RETURN. The font will be installed, and the usual *SpeedScript* screen will appear with the new font. That's all there is to it. You're now ready to use the custom font you selected.



```
[illegible]
←
←
←
An example of the scrawl.set←
←
←
←
←
a b c d e f g h i j  k l m←
←
n o p q r s t u v w x y z←
←
1 2 3 4 5 6 7 8 9 0 + - £←
←
A B C D E F G H I J K L M←
←
N O P Q R S T U V W X Y Z←
←
! " # $ % & ' ( )←
←
```

SCRAWL.SET

Creating Personalized Fonts

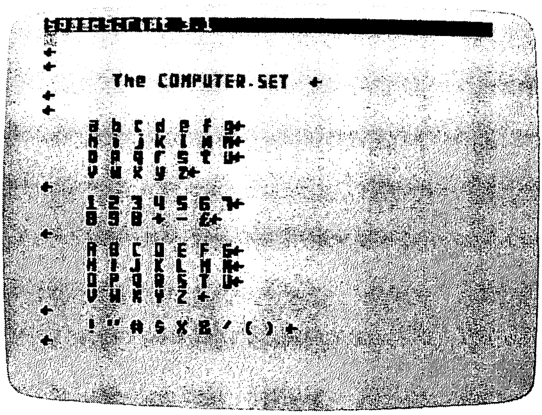
Also on the disk, but not appearing on the menu, are two important programs: Fontmaker Boot and Fontmaker. These two program are used to install fonts you have created. To use these programs you should copy them and *SpeedScript* (filename SS) to a disk that contains the font you created.

Fontmaker installs a character set that has been previously created; it has no provisions for creating the custom characters. You can easily define your own fonts or edit the supplied ones with a character editor such as "Ultrafont +" (see Chapter 6).

To run Fontmaker, load and run Font Boot (type LOAD "FONT BOOT",8, press RETURN; type RUN, and press RETURN again). Font Boot will load and run Fontmaker for you.

Fontmaker prompts you for the name of the character set you'd like to use. By default, the cursor blinks on the filename SPEED.SET. If you'd like to use a font with that name, just press RETURN. Otherwise, type in a new name, overwriting SPEED.SET. If you want to run *SpeedScript* without a custom set, just type X at the prompt (you don't need to erase SPEED.SET; just enter an X).

The character set you've previously created with a font editor program must be on the same disk as the *SpeedScript* program. Fontmaker looks for *SpeedScript* under the filename SS. Fontmaker loads in *SpeedScript*, bumps up the start of text space (reducing available memory by about 11K), loads the character set into that gap, switches the screen to the new character set, then runs *SpeedScript*.



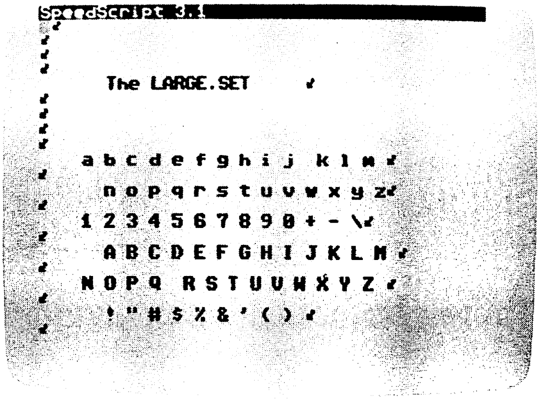
```
SpeedScript 3.1
←
←
    The COMPUTER.SET ←
←
←
      a b c d e f g←
      h i j k l m n←
      o p q r s t u←
      v w x y z←
←
      1 2 3 4 5 6 7←
      8 9 0 + - £←
←
      A B C D E F G←
      H I J K L M N←
      O P Q R S T U←
      V W X Y Z ←
←
      ! " # $ % & ' ( )←
←
```

COMPUTER.SET (designed by David Florance)

It's Only Temporary

Fontmaker does not permanently change *SpeedScript* unless you resave the word processor at this point (not recommended). In other words, Fontmaker installs the custom character set only for the current session. If you exit *SpeedScript* by pressing the RESTORE key, type POKE 53272,26 to restore the set before you type RUN to reenter *SpeedScript.*

When designing your custom character set, remember that vertical lines appear thinner and fuzzier than horizontal lines. Notice that every vertical line is doubled on the normal Commodore character set, making characters appear bold. You'll probably want to follow the same rule when designing your own sets. This is not a problem with crisp monochrome monitors. You can use the full 8 × 8 resolution of the character grid to design clean, well-formed characters.



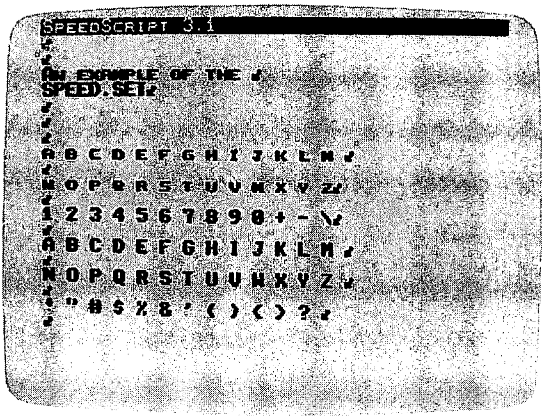
```
SpeedScript 3.1

     The LARGE.SET

   a b c d e f g h i j  k l m
     n o p q r s t u v w x y z
   1 2 3 4 5 6 7 8 9 0 + - \
     A B C D E F G H I J K L M
   N O P Q  R S T U V W X Y Z
     ! " # $ % & ' ( )
```

LARGE.SET

Another guideline for readability is that uppercase characters should be of uniform height. All lowercase characters should be the same height, except for tall characters such as *b, d, f, h, i, k, l,* and *t,* which should be the same height as uppercase letters. Normally, you'll keep the rightmost column and the lowest row blank to keep characters from running into each other and to reserve room for the lowercase descenders on the *g, j, p, q,* and *y.* Naturally, an exception is when you design cursive or script characters that should link together.

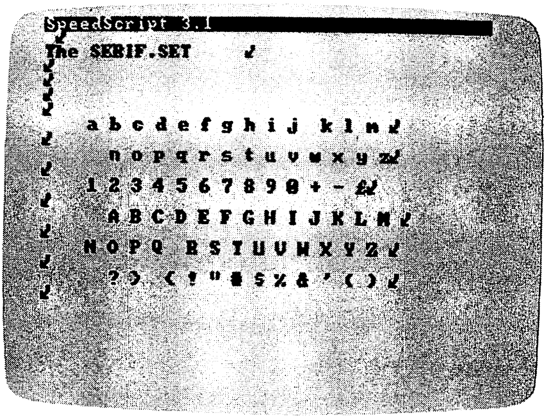
You'll also want to customize the punctuation marks and symbols. *SpeedScript* uses the back-arrow symbol as the carriage-return mark. If you don't like to see return marks, just blank out that character. You can put a tiny dot in the SHIFT–SPACE character to distinguish it from a real space. It can also be convenient to define some of the graphics characters to their printing equivalents on the printer. For example, some graphics characters print as italic or foreign-language characters. Just edit the graphics characters to look like their printing equivalents.

You can also create your own custom cursor. *SpeedScript*'s cursor just alternates between the normal and reverse-video version of whatever character it's sitting on. The last 128 characters of a character set are the reverse-video images of the first 128. If you want an underline cursor, just copy the normal set down to the reverse-video area and draw a line through the bottom row of every character. Special characters can even have a unique cursor.



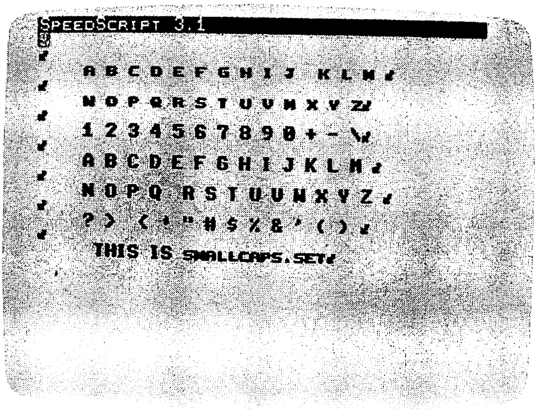
```
SPEEDSCRIPT 3.1
↵
↵
AN EXAMPLE OF THE ↵
SPEED.SET↵
↵
↵
↵
A B C D E F G H I J K L M ↵
↵
N O P Q R S T U V W X Y Z↵
↵
1 2 3 4 5 6 7 8 9 0 + - \↵
↵
A B C D E F G H I J K L M ↵
↵
N O P Q R S T U V W X Y Z ↵
↵
! " # $ % & ' ( ) < > ? ↵
↵
```

SPEED.SET



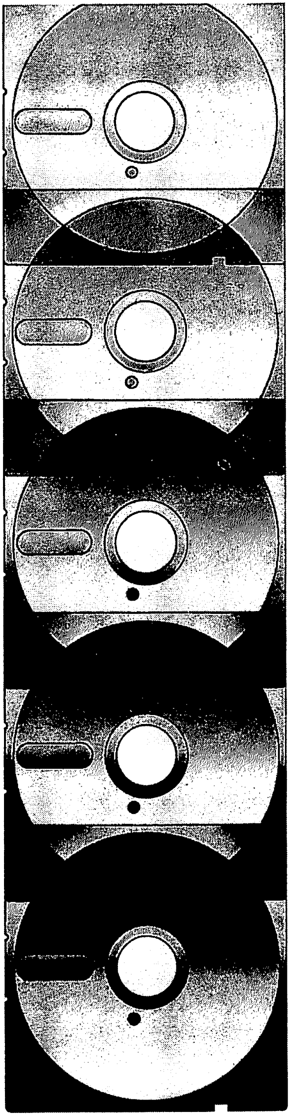
```
SpeedScript 3.1
The SERIF.SET       ↙
↙
↙
↙
↙
   a b c d e f g h i j  k l m ↙
↙
     n o p q r s t u v w x y z↙
↙
   1 2 3 4 5 6 7 8 9 0 + - £↙
↙
     A B C D E F G H I J K L M ↙
↙
   N O P Q  R S T U V W X Y Z ↙
↙
     ? >  < ! " # $ % & ' ( ) ↙
↙
```

SERIF.SET



```
SPEEDSCRIPT 3.1
↙
   A B C D E F G H I J  K L M ↙
↙
   N O P Q R S T U V W X Y Z↙
↙
   1 2 3 4 5 6 7 8 9 0 + - \↙
↙
   A B C D E F G H I J K L M ↙
↙
   N O P Q  R S T U V W X Y Z ↙
↙
   ? >  < ! " # $ % & ' ( ) ↙
↙
    THIS IS SMALLCAPS.SET↙
```

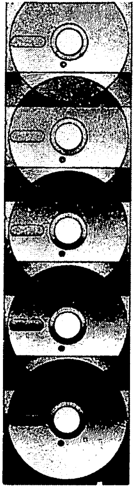
SMALLCAPS.SET



Chapter 3

SpeedCalc

Kevin Martin



Using *SpeedCalc*

Have you ever planned a budget for your home or office? If so, you probably used some sort of worksheet divided into rows and columns. Perhaps you wrote the months of the year along the top of the sheet and listed categories for earnings and expenses along one side. After entering data for each category and month of the year, you could calculate total income figures by adding or subtracting numbers in each of the sheet's "cells."

That's a classic example of a worksheet. It lets you enter and organize data, then perform calculations that produce new information. A *spreadsheet* program is an electronic version of the familiar paper worksheet. Since it does all the calculations for you at lightning speed, an electronic spreadsheet is far more convenient than its paper counterpart. And spreadsheet programs also offer built-in editing features that let you enter and manipulate large amounts of data with a minimum of effort.

SpeedCalc is an all machine language spreadsheet program for the Commodore 64. Though relatively compact in size, it's fast and easy to use, and has many of the features found in commercial spreadsheet programs. Even better, the *SpeedScript* File Converter program (see Chapter 2) included on the disk lets you merge your *SpeedCalc* files into word processing documents created with *SpeedScript*. Working together, *SpeedCalc* and *SpeedScript* make a powerful team. You can merge a chart of sales figures into a company report, create a table of scientific data for a term paper, and manipulate numeric information in many other ways. In a sense, a spreadsheet program brings to arithmetic all the flexibility and power that a word processor brings to writing.

To start *SpeedCalc,* simply load it from the menu. If you want to load it directly, enter LOAD"SPEEDCALC",8 and press RETURN. Once it's loaded, type RUN and press RETURN.

SpeedCalc, like *SpeedScript,* is a machine language program that can be loaded and saved just as you would a BASIC program; this makes it easy to make copies of the program on other disks.

The *SpeedCalc* Screen

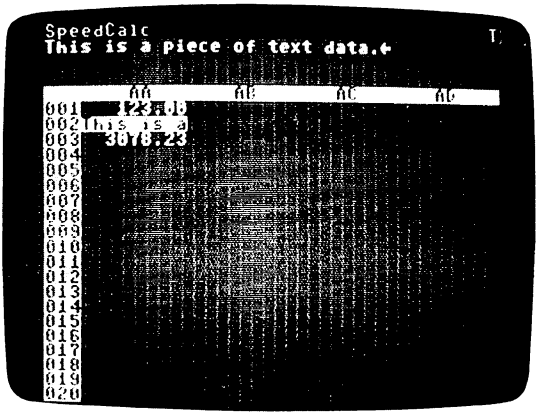
SpeedCalc uses the top line of the screen as the *command line.* This is where *SpeedCalc* displays messages and asks you questions.

Screen lines 2–4 are the *input buffer* area. This is the work area where you enter and edit data. As you'll see in a moment, the input buffer also displays the data contained in the current cell.

The lower 21 screen lines are your window into the spreadsheet. Though the spreadsheet contains many rows and columns, only a few can fit on the screen at a time. By scrolling the screen back and forth with the cursor, you can move the display window to any part of the spreadsheet.

The *SpeedCalc* worksheet consists of 50 vertical columns labeled with letters (AA, AB, AC, ..., BX) and 200 horizontal rows numbered from 1 through 200. The intersection of a row and column is called a *cell.* Cells are where you store data. With 50 columns and 200 rows, the *SpeedCalc* spreadsheet has a maximum of 10,000 (50*200) cells. Due to memory limitations, however, only about a third of these can actually contain data. But you may spread out the data over all 10,000 cells if necessary, depending on the format you need.

If you don't like the spreadsheet's screen colors, they're easily changed with the special function keys. Press the f1 key to cycle through the 16 border colors until you find one you like. The f3 key changes the background color, and f5 changes the character color.



```
SpeedCalc                                              T
This is a piece of text data.←

           AA          AB          AC          AD
001     1234.10
002This is a
003     3078.23
004
005
006
007
008
009
010
011
012
013
014
015
016
017
018
019
020
```

A typical screen from SpeedCalc.

Moving the Cursor

Each cell is identified with the letters of its column and the number of its row. For example, the cell at the extreme upper-left corner of the sheet is called AA1, since it's in column AA and row 1. The cell below that is AA2. Moving one cell to the right from AA2 puts you in cell AB2, and so on. (For the sake of clarity, this article uses uppercase letters for cell names. Note, however, that you must use lowercase letters such as *aa1* when entering cell names within *SpeedCalc.*)

Your current position in the spreadsheet is shown by the highlighted cursor. The simplest way to move around the sheet is with the cursor keys, which work just as they do in BASIC. Another way to move the cursor is with the HOME key (press CLR/HOME *without* pressing SHIFT). Press HOME once to move the cursor to the home position for the current screen—the upper-left cell. Press HOME twice in succession to move the cursor to cell AA1, the home position for the entire sheet.

SpeedCalc also has a *goto* command for moving over long distances. Press CTRL-G (hold down CTRL and press G). The command line turns blue and displays "GOTO:" followed by an underline cursor. The underline cursor generally indicates that *SpeedCalc* is waiting for data—in this case it expects the name of the cell where you wish to go. If you enter *ba188* at this point, *SpeedCalc* moves the cursor to cell BA188, adjusting the screen window as needed. Take a few moments to practice moving around the spreadsheet with all three methods—you'll be using them a lot. In a later section, we'll discuss how to change the size and format of a cell.

Keyboard Commands

SpeedCalc offers many different commands, a few of which are entered by pressing one key. However, most commands are entered by pressing CTRL along with another key. CTRL-G, as you've seen, is the goto command. CTRL-A displays the amount of free memory available, and so on. The most drastic command is CTRL-X, which exits *SpeedCalc* and returns you to BASIC. Since leaving the program effectively erases all data in memory, *SpeedCalc* asks "ARE YOU SURE Y/N?" before shutting down. To cancel the command and return to the spreadsheet, type N and press RETURN.

A few commands require you to press three keys at once. This sounds more awkward than it is in practice, since two of the three keys are SHIFT and CTRL. For instance, the command to switch between automatic and manual recalculation is performed by pressing SHIFT–CTRL-R (hold down SHIFT and CTRL, then press R). Table 3-1 in the clip-out section at the end of this chapter lists all the *SpeedCalc* commands, and Figure 3-1 is the keyboard layout with a description of what each key does. We'll be discussing each command in more detail below.

Three Data Types

Before entering any data, you must know what kind of data *SpeedCalc* accepts. There are three different types: numbers, text, and formulas. Let's look at each type in turn:

Numeric data consists of numbers—the basic stuff that spreadsheets work with. *SpeedCalc* has a few simple rules for numeric data: A number must be a decimal value (base 10, not hexadecimal) composed of one or more digits from 0 through 9, with an optional plus or minus sign. A decimal point is also optional. If you include any other characters in numeric input, *SpeedCalc* treats the entire input as text data (as explained below). Thus, the numbers 123, 0.001, and −65535 are valid numeric data. The numbers 65,535 (which includes a comma) and 312 Main Street are treated as text labels.

For example, let's enter the number 123 in cell AA1. No special commands are required to enter data. Just move the cursor to AA1 and begin typing. While you're entering the number, it appears only in the input buffer near the top of the screen. As soon as you press RETURN, the number appears in AA1 and the letter *N* appears at the upper right of the screen. The *N* signifies *numeric,* meaning that *SpeedCalc* has accepted the entry as valid numeric data. Move the cursor to a vacant cell, then move it back to AA1. The input buffer displays whatever data is found in the cell under the cursor. When the current cell is empty, the buffer is empty as well.

As you can see, pressing RETURN enters a data item into the current cell. You can also end the input by pressing a cursor key. The data is entered as if you had pressed RETURN, and the cursor moves in the indicated direction. This feature is handy for entering a lot of data: Simply type the entry, move the cursor to the next cell, enter more data, and so on.

Text data is not "data" in the strict sense, since *SpeedCalc* doesn't use it in calculations as it does numbers and formulas. Text data is there only to help humans understand what the other data means. Text may consist of comments, titles, column headings, subheadings, or whatever you need to interpret the numbers and formulas. As an example, move the cursor to cell AA2 (just under AA1) and type the following line. Note that both uppercase and lowercase letters are acceptable:

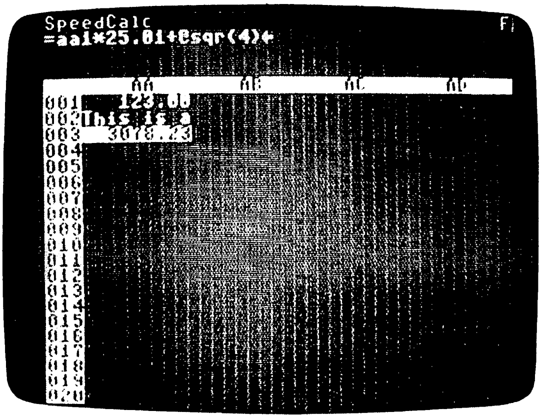
This is some text data.

You can use the DEL key to erase mistakes while you're typing. When you press RETURN, *SpeedCalc* displays *T* (for text) in the upper-right corner. In this example, the cell isn't long enough to accept all the text, so only the leftmost portion appears in AA2. But even though you can't see all the text, it's there. Move the cursor to another cell, then move it back to AA2. As soon as you return to AA2, *SpeedCalc* displays all the text in the input buffer.

Formula data is a mathematical expression or formula. It may be as simple as 2+2 or as complex as your imagination (and mathematical prowess) allows. The first character in a formula must always be an equal sign (=). If you omit this symbol, *SpeedCalc* either signals an error or treats the data as text. The true power of a spreadsheet is that a formula in one cell can refer to another cell. This is easier to demonstrate than to explain. Move the cursor to cell AA3 and type the following line:

=aa1*25.01+@sqr(4)

As soon as you press RETURN, *SpeedCalc* displays *F* (for formula) in the upper-right corner and puts the *result* of the formula (not the formula itself) into AA3. If AA1 contains 123, the value 3078.23 appears in AA3. In plain English, this formula means "multiply the contents of cell AA1 by 25.01 and add the square root of 4." Before we examine the formula more closely, here's a quick demonstration of what makes a spreadsheet such a powerful tool. Move the cursor back to AA1 and press CTRL-R. The command line displays the message RECALCULATION IS ON, meaning *SpeedCalc* now automatically recalculates the entire sheet whenever you make a change. Now change the number in AA1 to 456 (simply move to the cell and start typing). The new result (11406.53) automatically appears in cell AA3. We'll explain more about automatic recalculation later.



```
SpeedCalc                                          F
=aa1*25.01+@sqr(4)+

            AA          AB          AC          AD
001     123.00
002This is a
003    3078.23
004
005
006
007
008
009
010
011
012
013
014
015
016
017
018
019
020
```

SpeedCalc *displays the contents of the data cell under the highlighted cursor and the type of data in the cell.*

When you enter the name of another cell in a formula, the letters must be lowercase (enter *aa1,* not *AA1).* The referenced cell must contain data that *SpeedCalc* can evaluate: a number or another formula. If the formula refers to an empty cell or one that contains text, *SpeedCalc* signals an error.

Mathematical Operators

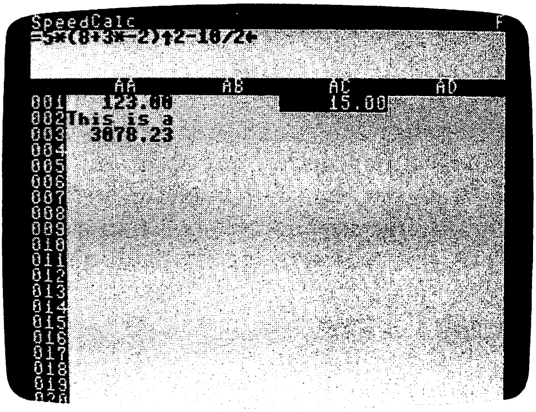
These symbols can be used as *operators* in a formula:

Operator	**Function**
+	**addition**
−	**subtraction**
*****	**multiplication**
/	**division**
↑ (up arrow)	**exponentiation**
=	**equality**

One factor that affects formulas is *precedence,* or the order in which mathematical operations are performed. In *SpeedCalc,* formula operators have the same precedence as BASIC—the same as in general math.

The first operators to be evaluated—those with the highest precedence—are those enclosed in parentheses. Where one set of parentheses encloses another, the expression in the innermost set is evaluated first. The next operators to be evaluated are exponents. Multiplication and division have equal precedence; both operations are lower than exponentiation. Addition and subtraction have the lowest precedence of all. The mnemonic "My Dear Aunt Sally" (MDAS—Multiplication, Division, Addition, Subtraction) is a reminder of mathematical precedence.

To take one example, *SpeedCalc* evaluates the formula **=5*(8+3*−2)↑2−10/+2** as the value 15, just as in ordinary math. Note how the result is affected by the plus and minus signs before the 2's.



```
SpeedCalc                                                   F
=5*(8+3*-2)↑2-10/2+

          AA            AB            AC            AD
001    123.00                       15.00
002This is a
003   3078.23
004
005
006
007
008
009
010
011
012
013
014
015
016
017
018
019
020
```

SpeedCalc *evaluates formulas just as you do in ordinary math.*

Functions

Formulas may also include any of the functions listed here:

@abs( )	**absolute value**
@atn( )	**arctangent**
@ave( )	**average of a block of cells [form: @ave(xxn:xxn)]**
@cos( )	**cosine of argument in radians**
@exp( )	**complement of log; gives *ex* (*e*=2.7182318...)**
@int( )	**integer (rounds to next lowest whole number)**
@log( )	**natural logarithm base *e* (log of zero or a negative number is illegal)**
@sgn( )	**sign (−1 for negative numbers, 0 for zero, 1 for positive)**
@sin( )	**sine of argument in radians**
@sqr( )	**square root (root of a negative number is illegal)**
@sum( )	**sum of a block of cells [form: @sum(xxn:xxn)]**
@tan( )	**tangent of argument in radians [@tan(.5*pi) is illegal]**
pi	**value of pi (3.14159265)**

All the functions except pi begin with the @ sign and are followed by parentheses. Within the parentheses of a function you may use a number or formula. For example, the formula =@sqr(4) generates the square root of 4. The formula =@sqr(aa1) returns the square root of whatever value cell AA1 contains. Note that the *argument* (value within parentheses) of the functions @tan( ), @sin( ), and @cos( ) must be expressed in radians; the result of the function @arc( ) is expressed in radians.

The function @int( ) generates an integer (whole number) by rounding to the next lowest whole number. For positive numbers, this is equivalent to dropping the fraction, but for a negative number like −4.3, the next lowest number is actually −5.

The function @ave( ) calculates the mean average of the values in a block (group) of cells. The function @sum( ) calculates the sum of a block. Both functions require that you define the block so that *SpeedCalc* knows which cells to include in the calculation. This is done by putting two cell names separated by a colon inside the parentheses. The first cell name defines the upper-left corner of the block, and the second defines the bottom-right corner. For instance, @ave(aa1: ad20) calculates the average of all the cells from AA1 to AD20. The function @sum(aa1:ad20) calculates the sum of AA1 through AD20, and so on. An error results if any cell in the block is blank or contains text data.

Math Notes

SpeedCalc uses the same ROM routines for math as BASIC. Therefore, it follows almost the same rules and has the same limitations. Numbers are accurate internally to only nine digits—although you can enter long numbers and view them exactly as you entered them, only the first nine digits are used for calculations. If you enter a very long number (more than 36 digits), your input is ignored, and the cell reverts to its former state. You can also enter long numbers in the form 1.23E+05 (scientific notation). Note, however, that *SpeedCalc* itself never uses scientific notation. It converts all numbers to their full length, so long numbers actually use more memory than shorter ones.

Beware of math errors such as division by zero, square root of a negative number, tangent of pi/2, logarithm of zero or a negative number, and overflow conditions like 2↑300. *SpeedCalc* detects these errors and displays *ERROR* in the cell of an illegal calculation.

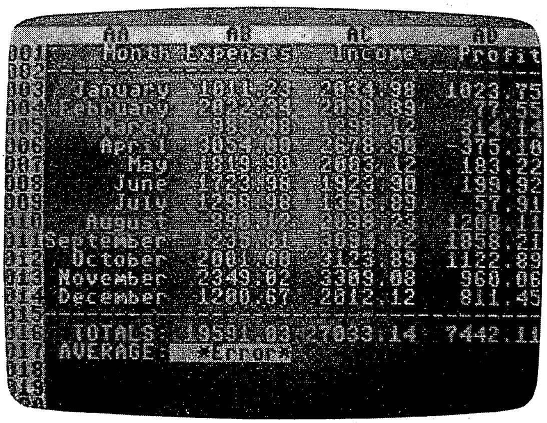
Editing the Sheet

Editing is a very important spreadsheet function. The simplest way to change what a cell contains is to move to it and start typing. The old data in that cell is replaced by whatever you enter. For instance, to replace the contents of cell AA1 with the number 456, move to that cell, type 456, and press RETURN or exit with a cursor key.

Press CTRL-B (think of *blank*) to erase what's in the current cell. You can also clear a cell by typing a space and pressing RETURN, but this uses some memory. A truly blank cell doesn't use any memory.

To erase everything in the sheet, press SHIFT–CLR/HOME. Before carrying out this drastic operation, *SpeedCalc* asks you to confirm it by pressing Y or N.

In some cases, only a minor change is needed. *Edit mode* lets you change the data in a cell without retyping the entire entry. To activate edit mode, move to the desired cell and press CTRL-E. In this mode, the up/down cursor key is disabled, and the left/right cursor key moves within the input buffer. Erase unwanted characters with the DEL key. Typing in edit mode inserts new characters in the line: Everything to the right of the new character moves right one space (unless the buffer is already full). Since the cursor keys have a different function in edit mode, you cannot use them to end the input. Press RETURN to enter the new data and escape from edit mode.



```
          AA        AB        AC        AD
001     Month  Expenses    Income    Profit
002----------------------------------------
003   January   1011.23   2034.98   1023.75
004  February   2022.34   2099.89     77.55
005     March    983.98   1298.12    314.14
006     April   3054.08   2678.90   -375.18
007       May   1819.90   2003.12    183.22
008      June   1723.98   1923.90    199.92
009      July   1298.98   1356.89     57.91
010    August    890.12   2098.23   1208.11
011 September   1235.81   3094.02   1858.21
012   October   2001.00   3123.89   1122.89
013  November   2349.02   3309.08    960.06
014  December   1200.67   2012.12    811.45
015----------------------------------------
016   TOTALS:  19591.03  27033.14   7442.11
017  AVERAGE:   *Error*
018
019
```

Typical SpeedCalc *error message.*

As you may have learned already, *SpeedCalc* displays *ERROR* in a cell when you enter an erroneous formula. The usual cause is that you have made a typing error in that cell, or the formula refers to text or an empty cell. A line of asterisks (*********) signals that a number is too large to be printed in the cell. Though these messages appear in the cell area, no data is lost. You may move to the affected cell, view its contents in the input buffer, and make whatever correction is needed.

Recalculation

The recalculation feature is the very core of *SpeedCalc.* As you know, entering or editing a piece of data causes *SpeedCalc* to perform a calculation and put the result in the cell under the cursor. In most cases, the new data relates to data in other cells, so you'll ultimately want to recalculate the entire spreadsheet as well. This can be done in two different ways: manually or automatically.

To recalculate the spreadsheet manually, press the back-arrow key (←) at the upper left of the keyboard. *SpeedCalc* begins at AA1 and recalculates every cell that contains data, placing fresh results wherever needed. *SpeedCalc* displays the message RECALCULATING while it's busy.

If you switch to automatic recalculation mode, *SpeedCalc* automatically recalculates the entire spreadsheet each time you enter new data or edit what exists. When you press CTRL-R, *SpeedCalc* changes the recalculation status and displays it at the top of the screen. If automatic recalculation was turned off before, it is now on (and vice versa). If you aren't sure which mode you're in, press SHIFT–CTRL-R; *SpeedCalc* displays the recalculation mode without changing it.

Automatic recalculation can be fun to watch in a large spreadsheet. Every time you make a change, new results ripple all the way down the screen. However, the more data your spreadsheet contains, the longer it takes to update the entire sheet. For this reason, you may want to turn automatic recalculation off most of the time, recalculating with the back-arrow key whenever you need to view results.

One problem with recalculation arises from the order in which cells are calculated. Because only one cell can be calculated at a time, you must sometimes recalculate the entire spreadsheet two or three times to get correct results in every cell (this is common to all spreadsheet programs). For instance, say you have a formula in AA1 which refers to a formula in AB15. When *SpeedCalc* calculates AA1, it must use the existing data from AB15—which is probably out of date since the formula in AB15 hasn't been recalculated yet. To avoid this problem, you should always press the back-arrow key two or three times before printing a spreadsheet or saving it to disk.

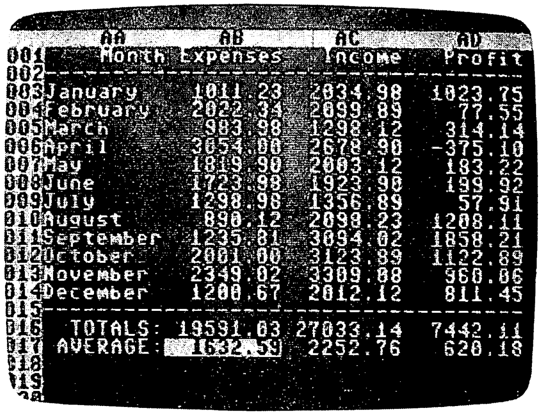
SpeedCalc offers a number of other features. Before experimenting with them, you should spend some time typing in a hypothetical spreadsheet—perhaps a fictitious yearly budget—to become thoroughly familiar with the basic commands covered so far. Most importantly, create formulas, using all the operators in different combinations. Try doing things that you know will cause errors. Then correct the errors in edit mode, and so on. It takes a thorough grasp of the fundamentals to get the most out of *SpeedCalc*'s advanced features.

Change Type and Format

The default (normal) format for numeric data is flush right with rounding to two decimal places. In other words, the number is displayed in the rightmost part of the cell, with two numbers after the decimal point. Text and formula results are also displayed flush right. *SpeedCalc* offers several commands for changing cell formats.

Change format (CTRL-F). This command changes the location of data in the cell and the number of decimal places. When you press CTRL-F, *SpeedCalc* displays the question "FORMAT: Left, Center, or Right justify?" in the command line. Press L, C, or R to move the data to the left, center, or right of the cell.

Change number of decimal places (CTRL-D). This command lets you specify the number of digits displayed after the decimal point. The default value is 2, convenient for dollar amounts, but you may change it to anything from 0 through 15. If you choose zero decimal places, any number in that cell is rounded off to the nearest integer (whole number). A setting of 15 is special: The number in that cell is not rounded off at all. Instead, *SpeedCalc* displays the number exactly as you entered it or as it was calculated from a formula. Watch out for one feature of CTRL-D: It also resets the cell to right justification, so you may need to change this with CTRL-F.



```
          AA       AB        AC         AD
001    Month Expenses    Income     Profit
002-----------------------------------------
003January    1011.23   2034.98    1023.75
004February   2022.34   2099.89      77.55
005March       983.98   1298.12     314.14
006April      3054.00   2678.90    -375.10
007May        1819.90   2003.12     183.22
008June       1723.98   1923.90     199.92
009July       1298.98   1356.89      57.91
010August      890.12   2098.23    1208.11
011September  1235.81   3094.02    1858.21
012October    2001.00   3123.89    1122.89
013November   2349.02   3309.08     960.06
014December   1200.67   2012.12     811.45
015-----------------------------------------
016  TOTALS: 19591.03  27033.14    7442.11
017 AVERAGE:  1632.59   2252.76     620.18
018
019
020
```

SpeedCalc *allows you to change the location of data within a cell. Here, the months have been left-justified.*

Width (CTRL-W). The width command changes the width of an entire column of cells. Move the cursor to any cell in the desired column, then press CTRL-W. When *SpeedCalc* displays the prompt "Width:", you should respond with a number from 4 through 36. The entire screen is redrawn to accommodate the new format, and may look very different depending on what value you choose. For instance, if you increase a column's width, the rightmost column of the former display may disappear. *SpeedCalc* displays only as many complete columns as it can fit on the screen. If you decrease the width of a column, you may see asterisks where numbers used to be (indicating the cell is now too small to display the entire number). To get rid of the asterisks, expand the column as necessary.

Global format (SHIFT-CTRL-F). This is the same as the ordinary format command, but operates globally, changing every cell in the sheet instead of just one. To alert you to the difference, *SpeedCalc* changes the color of the command line to blue.

Global width (SHIFT-CTRL-W). This is a global version of the width command. The command line turns light green to signal the difference. Every column in the sheet changes to the designated width.

Global decimal (SHIFT-CTRL-D). This command lets you change the number of decimals displayed for the entire sheet. The default for the sheet is two decimal places. Every cell changes to the designated setting, and the new setting becomes the default for future entries.

Macroediting

After typing in a large spreadsheet, you may decide to make a major change. You may want to add new data somewhere in the middle, delete a section, or move a group of cells from one location to another. *SpeedCalc's* macroediting (large-scale editing) commands simplify such operations, affecting an entire block of cells at once. A *block* is simply a group of cells connected in rectangular fashion: You can define it as a single cell, a row or column, or any rectangular area within the spreadsheet.

There are two ways in which macro commands can work: *verbatim* or *relative*. To take a simple example, say that cell AA2 contains the formula =aa1*5, and you want to move its contents to cell AB2. When this is done in verbatim mode, AB2 contains an exact copy of what was in AA2 (=aa1*5). Note that the cell name used in the formula does not change; the formula still refers to AA1. If you perform the same operation in relative mode, the cell name in the formula is adjusted to fit the new location. In this case, AB2 would contain the formula =ab1*5.

Copy (CTRL-C). The copy command copies a block of cells into a different location without disturbing the original cells. Place the cursor on the upper-left corner of the block you want to copy, then press CTRL-C. *SpeedCalc* changes the command line to purple and prompts you to move the cursor to the lower-right corner of the block you want to copy. Once the cursor is in place, press RETURN. Now *SpeedCalc* prompts you to move the cursor to the place where you want to put the block. This is the upper-left corner of the new position. Once the cursor is there, press RETURN again. The new data replaces whatever was contained in the designated cells. Note that if you define an impossible block (for instance, moving the cursor to the upper-left of the original position, rather than below and to the right), *SpeedCalc* does not copy any data. You can use this trick to escape from Copy if you press CTRL-C accidentally. Another escape is to press RETURN twice while the cursor remains on the original cell.

Move (CTRL-M). This command works like a copy, but it fills the original cells with blanks. Though *SpeedCalc* has no insert command, you can use this command to make space for new data in the middle of a spreadsheet. Simply move everything below the insertion point down as far as you need. To cancel this command, press RETURN twice while the cursor is on the same cell.

Relative Copy (SHIFT-CTRL-C). This form of the copy command adjusts the cell names used in formulas within the copied block (see explanation above).

Relative Move (SHIFT-CTRL-M). This is the relative form of the move command. Cell names in formulas are adjusted to reflect the move.

Memory Management

SpeedCalc leaves 10,752 bytes of memory (10.5K) available for data. As noted earlier, *SpeedCalc* lets you spread your data over a much larger number of cells than you can actually fill with data. The extra space is provided to give you full control over the final format of the spreadsheet—for example, you could have a 15 × 150 spreadsheet—and to leave some elbow room for move and copy operations.

Because memory is limited, you should keep careful track of how much is free while using the program. Press CTRL-A to display the amount of free memory. We suggest limiting your spreadsheets to 1296 cells (equivalent to 36 rows × 36 columns). If you have filled nearly all of free memory, you may have to break the spreadsheet into two smaller sheets.

Although *SpeedCalc* checks the amount of available memory and displays an error message if you run out of memory, you should be careful not to exhaust free memory. Any move or copy operation in process will be aborted if sufficient memory is not available.

Disk Operations

SpeedCalc has four disk commands which allow you to save a spreadsheet to disk, load it, display the disk directory, and send commands to the disk drive. The directory command is the simplest to use. Press CTRL-4 (think of the dollar sign, as in LOAD "$",8 to list the directory from BASIC). The screen clears and the directory is displayed. Press RETURN to return to the normal screen. You may pause the directory display with the space bar.

To save a spreadsheet to disk, press the f8 function key (SHIFT-f7). *SpeedCalc* prints "SAVE:" on the command line, followed by an underline cursor. Enter a valid Commodore filename and press RETURN. (If you change your mind and decide not to save anything, press RETURN without typing a filename.) The disk drive spins for a few moments, then *SpeedCalc* prints the drive status in the command line. The message "00,OK,00, 00" means there were no errors.

To load a saved file from disk, press the f7 key. Again, *SpeedCalc* prompts you to enter the filename and displays the disk status when the operation is complete. *SpeedCalc* files are saved as PRG (program) file types, but do not load as normal program files. *SpeedCalc* uses special header bytes to identify a *SpeedCalc* file. If you try to load anything other than a valid *SpeedCalc* file, you'll see the message NOT A SPEEDCALC FILE.

You can send Commodore disk commands to the drive with CTRL-↑—press CTRL and the ↑ (up-arrow) key together. *SpeedCalc* prompts you to enter a disk command. The CTRL-↑ command works much like the Commodore Wedge utility. If you press RETURN without typing a command, *SpeedCalc* displays the drive status and sends no command. You need not enclose the command in quotation marks or type ,8 after it. For example, press CTRL-↑, then enter I0 to initialize a disk. Consult your disk drive manual for more information about Commodore disk commands.

Printing

SpeedCalc lets you print data to three different devices: to the screen for previewing output, to a printer for permanent documentation, or to a disk file for integrating the data with another program.

To preview your spreadsheet on the screen, place the cursor below and to the right of the section of the sheet you wish to view, press SHIFT–CTRL-P, then press S (screen output) when prompted. Naturally, the display will look odd if your sheet is wider than 40 columns. Think of each pair of 40-column lines as one 80-column printed line.

To print a hardcopy of the spreadsheet, press CTRL-P. If your printer is configured like most, this should produce a satisfactory printout. This command sends output to the printer as device number 4 with a secondary address of 7 (uppercase/lowercase on most systems). *Before using this command, you must position the cursor below and to the right of the block of* *cells you wish to print.* The upper-left corner of the printout starts at cell AA1. The entire width you define by this position is used. Therefore, don't try to print overly wide spreadsheets that won't fit on the paper. If you want to print a spreadsheet wider than 80 columns, many printers have a condensed mode that lets you fit 132 characters on a line. You can set this by switching an internal DIP switch, or by sending a CHR$ code from BASIC before running *SpeedCalc.* Many printers respond to this command for condensed mode: OPEN 4,4: PRINT#4,CHR$(15):CLOSE 4.

To send output to a printer with a device number other than 4 or a secondary address other than 7, enter SHIFT–CTRL-P, then enter the device number and secondary address when prompted. During a printout, you can pause the output by pressing SHIFT or SHIFT LOCK. The screen border turns white and printing ceases until you release SHIFT. Press RUN/STOP to abort printing.

You can also print *SpeedCalc* data to a disk file for use with terminal programs, databases, or word processors (including *SpeedScript).* Select the D option after pressing CTRL–SHIFT-P, then enter the filename you wish the new file to have. The data is saved as a program (PRG) file of that name. The disk file is an exact Commodore ASCII image of what would go to the printer.

Note that *printing* to disk creates a different file than does *saving* to disk. You should *save* files that you wish to reload into *SpeedCalc* and *print* files that you wish to convert for *SpeedScript* or other programs. While you may pause this operation with SHIFT as with printer output, *do not use RUN/STOP to abort printing to disk.* This may create a "poison" (unclosed) file which can be safely removed only by validating the disk.

Using *SpeedCalc* Data with *SpeedScript*

SpeedCalc sends data to the printer in simple, plain-vanilla form. That may be fine for personal use, but if you're creating a document for others to view, you may want special features such as boldface, underlining, and so on. Since *SpeedScript* already offers a way to access these features (and many more), no attempt has been made to include them in *SpeedCalc.* All that's needed is to use the *SpeedScript* File Converter program to convert *SpeedCalc* files into a form that *SpeedScript* can load.

Then you can edit the file with *SpeedScript* as you would any other document—inserting printer control codes, reformatting the text, merging it with other text, and so on.

To change a *SpeedCalc* spreadsheet into a file that *SpeedScript* can read, you'll need to do the following:

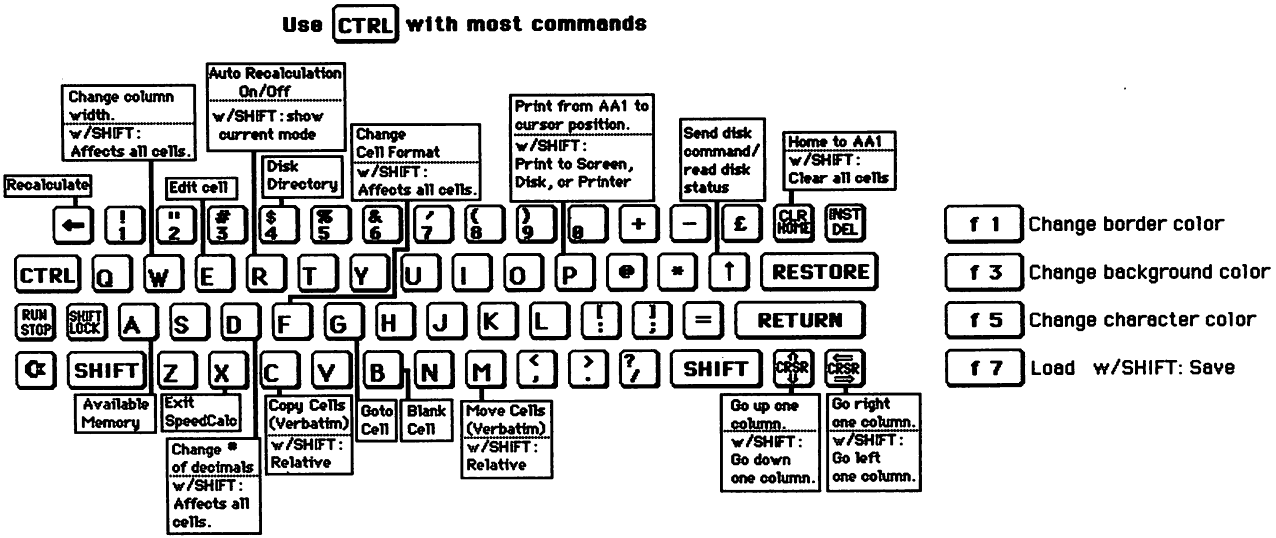
1. After creating and *saving* a spreadsheet with *SpeedCalc, print* it to disk as described above (*be sure the cursor is to the right and below the data you want*).
2. Exit *SpeedCalc,* then load and run the *SpeedScript* File Converter program (stored on the disk with this book with the name SSFC). The program prompts you to enter the input filename (enter the filename of the *SpeedCalc* file you printed to disk). Then it asks you to enter the output filename (the name of the *SpeedScript* file you want to create; of course, this name should be different from the first). Press D for disk. Finally, you should select option 3, Commodore ASCII to SpeedScript.
3. After the File Converter has finished, load and run *SpeedScript,* then load the new *SpeedScript* file as you would any *SpeedScript* document. The data appears on the screen, ready to be edited in any way you wish.

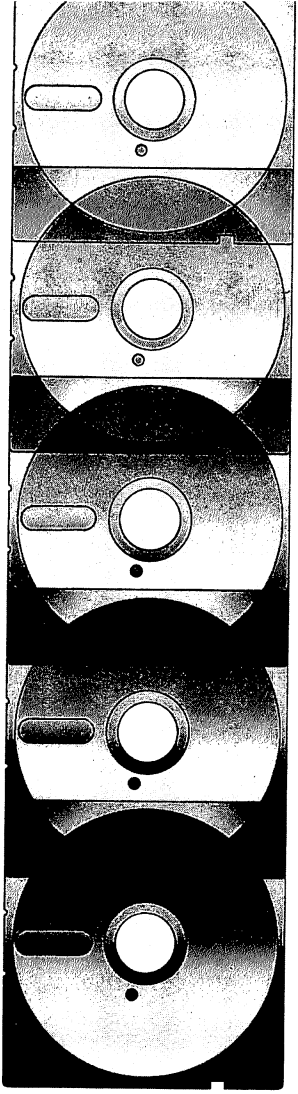
Following this chapter are clip-out charts for *SpeedCalc* commands and keyboard reference.

Table 3-1. *SpeedCalc* Commands

Command	Action
CTRL-A	Available memory check
CTRL-B	Blank (erase) current cell
CTRL-C	Copy block verbatim
CTRL-D	Set number of decimals
CTRL-E	Edit current cell
CTRL-F	Change cell format
CTRL-G	Goto selected cell
CTRL-M	Move block verbatim
CTRL-P	Print sheet
CTRL-R	Turn on/off auto recalculation
CTRL-W	Change column width
CTRL-X	Exit *SpeedCalc*
CTRL-4	Disk directory
CTRL-↑	Send disk command
CLR/HOME	Home cursor
SHIFT–CTRL-C	Copy block relative
SHIFT–CTRL-D	Change decimal mode for all cells
SHIFT–CTRL-M	Move block relative
SHIFT–CTRL-P	Print to screen, disk, or printer
SHIFT–CTRL-R	Display current recalculation mode
SHIFT–CTRL-W	Change width of all columns
SHIFT–CLR/HOME	Erase entire sheet
f1	Change border color
f3	Change background color
f5	Change character color
f7	Load *SpeedCalc* file
f8	Save *SpeedCalc* file
←	Recalculate sheet

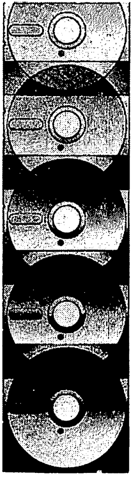
Figure 3-1. *SpeedCalc* Keyboard Reference


Use CTRL with most commands
Recalculate
Change column width. w/SHIFT: Affects all cells.
Edit cell
Auto Recalculation On/Off w/SHIFT: show current mode
Disk Directory
Change Cell Format w/SHIFT: Affects all cells.
Print from AA1 to cursor position. w/SHIFT: Print to Screen, Disk, or Printer
Send disk command/ read disk status
Home to AA1 w/SHIFT: Clear all cells
Available Memory
Exit SpeedCalc
Change # of decimals w/SHIFT: Affects all cells.
Copy Cells (Verbatim) w/SHIFT: Relative
Goto Cell
Blank Cell
Move Cells (Verbatim) w/SHIFT: Relative
Go up one column. w/SHIFT: Go down one column.
Go right one column. w/SHIFT: Go left one column.
f 1 Change border color
f 3 Change background color
f 5 Change character color
f 7 Load w/SHIFT: Save



Chapter 4
Mini-Filer

Kevin Martin



Mini-Filer

Databases are powerful tools for many applications. They can manage any information you need to store and retrieve. "Mini-Filer," while not a full-featured database, allows you to easily create and manage any number of files by providing various single-key commands for creating, adding, editing, searching, and printing out.

Because Mini-Filer is a general-purpose file manager, it can be used for many applications. It's not designed to fulfill a specific purpose, so it's general enough to handle many different types of files. For example, you could create a name and address file (name, street, city, state, zip code, and phone number), a file for your garden (plant name, gestation, watering/feeding needs, and harvest time), and a file for your library (title, author, publication date, publisher, and subject).

Each grouping of information within a file is a *record*. For example, *John Doe, 123 Main Street, Anywhere, USA 11111, 888-999-9999* could be one record within an address file. When you enter a new name, address, and so on, you're adding another record. In Mini-Filer, there's no restriction as to the number of records (outside the normal limitations of computer memory and disk space), but exercise common sense when creating a file. Remember, the longer your file, the longer the search time to find a record. And if the record size is fairly small, you'll be able to fit more records into available memory. Finally, each record contains one or more *fields*. In the examples above, name, street, and city are fields within the address file, and author and publisher are fields in the library file. The fields can be *alphanumeric*, meaning any characters are allowed, or *numeric*, containing numbers only.

Think of the structure as a recipe card box. The entire collection of cards is the file. Each card is a record inside the file. And each line on the card is a field.

If you see how all this information is organized, you can see how useful a file manager like Mini-Filer can be. There are a multitude of applications. Aside from those noted above, you can create a file for home inventory, collections (cards, coins, stamps, record albums, and so on), gifts, recipes, Christmas cards, and mailing labels, to name a few.

A Few Steps

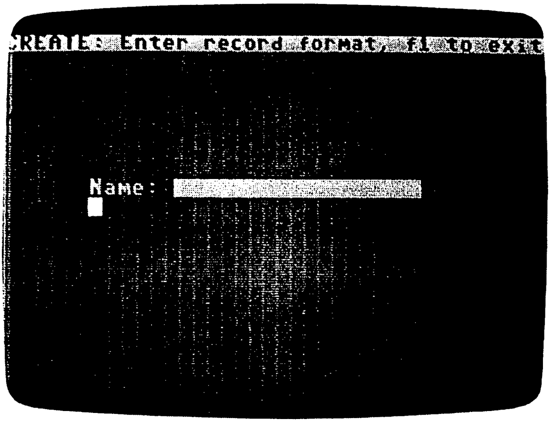
Load Mini-Filer from the menu or directly by entering LOAD"MINI-FILER",8 and pressing RETURN; once the prompt appears, enter RUN and press RETURN.

Using Mini-Filer requires three basic steps: create a file structure, enter records, and save these records to disk for later use. Mini-Filer also allows you to print out a single record or an entire file at the stroke of a single key. A printer is not required for use with the program, however.

Creating a File

After running Mini-Filer, you'll see a gray screen with a command line at the top which includes a series of letters, each of which represents a command: A (add or enter), C (create), F (find), P (print a record), R (print a file), f7 (load a file from disk), f8 (save a file to disk), CLR (clear data from memory), up (cursor up), and down (cursor down). For a complete list with more detailed explanations, see Table 4-1 at the end of the chapter. To get started, press C to create a file. You'll notice that the command line has changed to read "CREATE: Enter record format, f1 to exit." (If you have a file already in memory, this option is not allowed.) You'll notice the cursor at the top-left corner of the screen, beneath the command line. Think of the screen as a blank file card on which you can write anywhere. Using the cursor keys, move the cursor to where you wish to begin. Any location on the screen is acceptable, but note that each record is restricted to one screen of information.

First, type a label that describes the information in the field that will follow. Determine the maximum number of characters you'll need (up to 128 characters per field). For example, in creating a name and address file, you might type *Name:* and decide that 18 characters would be enough for even the longest name you'd enter. Next, press SHIFT and the space bar to allocate the number of spaces for the field length. In our example, you would hold down the SHIFT key and press the space bar 18 times.



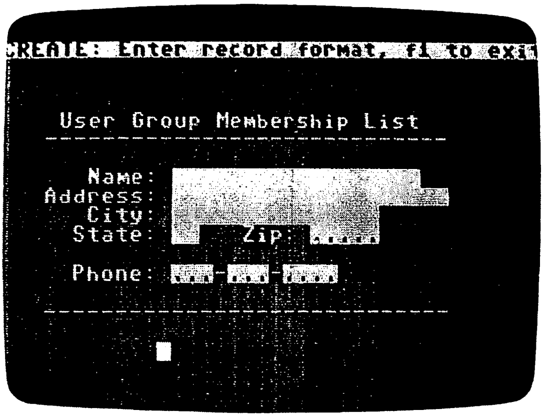
```
REATE: Enter record format, f1 to exit

    Name:
```

Creating a new field.

If you've made any typing errors, use the INSerT/DELete key to edit. Press RETURN when you've finished defining the length of a field. (In our example, press RETURN after the eighteenth space has been entered.) The cursor will then jump to the beginning of the line below. Next, move the cursor to where you want to type the name of the next field label. Repeat the procedure just described until you've completed a record.

If you plan to use a numeric field (like a phone number or zip code), press the Commodore key and the space bar rather than a shifted space. (A numeric field appears as reverse periods; an alphanumeric field—letters and/or numbers—appears as reverse spaces.)

The command line at the top of the screen reminds you to press f1 to exit the format creation mode. Now you're ready to enter data. The original command line will reappear. Press A (for adding or editing a record). This puts you in Add mode. You'll see "ADD Record" appear in the command line, and the cursor will immediately go to the first space in the first field you've defined. Now begin entering information. After typing in the data for one field, press RETURN. The cursor will automatically jump to the first space in the next field. If you reach the last available space in the field and type a character, the cursor will jump to the first space in the next field. The INSerT/DELete key may be used in case of typing errors. If you've defined a field as numeric only (appearing as reverse periods), it will accept only numeric characters. Continue entering information until all fields in the record have been entered. Pressing RETURN when you're at the last character of the last field will return the cursor to the first character of the first field. At this point, you should proofread the record just entered. If you've discovered a typing error, press RETURN until you're at the beginning of the field that contains an error and move to the unwanted character with the cursor-right key. After making the correction, press RETURN.



```
CREATE: Enter record format, f1 to exit

  User Group Membership List
 ------------------------------

     Name:
  Address:
     City:
    State:       Zip:

    Phone:    -   -
 ------------------------------
```

A typical name and address file record.

To proceed to the next record, press f7. You'll see the formatted fields with the cursor in the first position of the first field. After completing the record, you can proceed to enter another by pressing f7, or you may exit the Add mode by pressing f1. The latter choice will return the original command line. At this point, you can move through all records by pressing the cursor-up key to read the previous record, or the cursor-down key to move to the next record. When the screen shows the format field—the fields with no entries—you're at the last position in the file. At this point, if you wish to enter a new record, press A and type in new data. If you wish to edit a record, press A while the record you wish to edit is on the screen.

Searching for Data

To search for any information, press F (for Find). You're then prompted to enter the characters you wish to search for. After entering them, press RETURN. Mini-Filer will instantly find the requested information and display the record which contains it. The original command line is then returned. To search for any subsequent occurrences, press SHIFT-F. The search will continue from the next record, and so on.

If the information requested is not in the file, the command line will display "Not Found." (When searching, be sure to enter the word you're looking for with the exact spelling, including capital and lowercase letters—*Spanish* will not be found if you enter *spanish.)* Also, Mini-Filer will find a string of data within a field. For example, if you're searching for an area code—which is a portion of the phone number field—you would type the three-digit code and Mini-Filer would find it.

Saving and Loading

From the main command line, you can load or save a file at any time. To save a file, press f8. You'll be prompted for a filename. After typing one in, press RETURN, and the file will be saved to disk as a program file with the name you specified.

To load a file from disk, press f7. You'll be prompted for a filename. After entering one, press RETURN and Mini-Filer will load the file from disk. After the file has loaded, the first record in the file will be displayed. Press any key to get the main command line.

Should you mistakenly press f7 or f8 and get a LOAD or SAVE prompt, press RETURN to get back to the main command line.

Printing Options

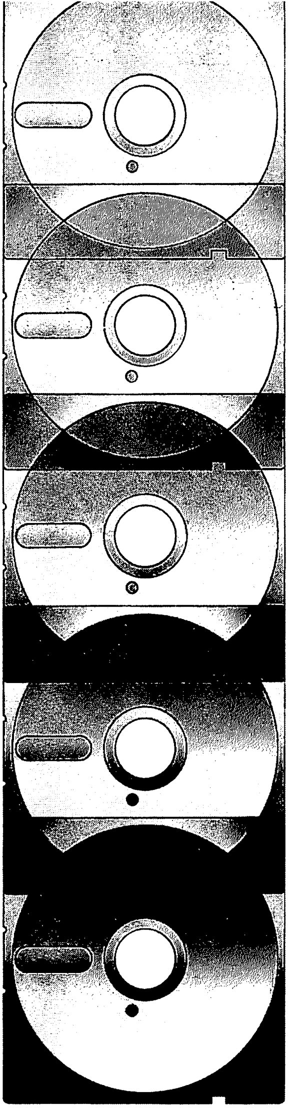
To print the current record, the one displayed, press P. This prints *only* the current record. To print the entire file, press R (for Report). The printed characters will appear just as they do on the screen.

Erasing a File

At some point, you may wish to erase the data in a file from memory and start over. To do this, press CLR (SHIFT–CLR/HOME). You're asked "Are You Sure?" Press Y if you wish to erase it or N if you'd rather reconsider. CLR is also useful if you plan to work with more than one file in a single sitting. You create the first, save to disk, and then press CLR. You can now load (or create) the second file.

Table 4-1. Mini-Filer Commands

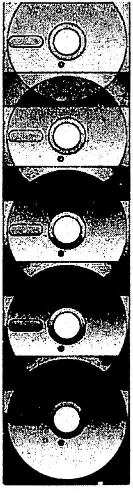
A	Add or edit data
C	Create a database file
F	Find a field (any string of characters)
SHIFT-F	Continue searching for next occurrence (may be used only after F)
P	Print the record displayed
R	Print the file
f7	Add a record (in Add mode); load a file from disk (from main command line)
f8	Save a file to disk
CLR	Clear all file data from memory
up (cursor key)	Move the cursor up (from Create mode) or move to the previous record (from main command line)
down (cursor key)	Move the cursor down (from Create mode) or move to the next record (from main command line)



Chapter 5

Sidplayer

Craig Chamberlain



Music and the Sidplayer

Of all the special chips in the Commodore 64, none is more devoted to one specific purpose than the SID chip. This chip, more formally known as the Sound Interface Device, is solely responsible for producing music and sound effects. It has many features for controlling sound and has been described as a complete synthesizer on a chip. The only problem is that the SID chip is very complicated. As a result, it's not fully understood by most people and often is not fully used.

Enter "Sidplayer," a music playing and editing utility for the Commodore 64. This music system is designed to make it easy to use the SID chip so that you can realize its full potential. Using Sidplayer, you can make your Commodore 64 produce music of extremely high quality.

To fully appreciate the capabilities of Sidplayer, it's first necessary to have an understanding of how electronic music works.

Electronic Music

When your ear hears a sound, it is actually detecting vibrations. The rate of vibration is called the *frequency* and determines the *pitch* of a sound. In a musical instrument, a metal string, reed, stretched membrane, or air in a tube is what vibrates. The player usually has a method for changing the pitch.

But sounds are not so simple. There are many different kinds of vibrations. When viewed with an oscilloscope, vibrations have another characteristic, called *waveform*. Square, triangle, and sawtooth are common waveforms. The waveform helps distinguish the sound produced by one instrument, such as a flute, from the sound produced by another instrument, such as a violin.

There's just one more essential characteristic remaining—volume. As a string is plucked or air is blown, the volume changes over a short period of time. This pattern of changing volume levels is called an *envelope,* and is usually divided into four parts called the *attack, decay, sustain,* and *release.* In the first three parts, the volume rises to a peak level and then falls to a sustain level. When the note is released, the volume fades away to silence.

The frequency, waveform, and envelope are all essential parts of a note. A sequence of notes creates music. Electronic music is merely a method of producing these qualities of sound by electronic means. A device which does this is called a *synthesizer.* Theoretically, it's possible for a synthesizer to imitate any musical instrument or to produce sounds never heard before.

The Sidplayer

The SID chip contains three oscillators. Each oscillator acts as one "voice" and can produce a tone in a range of eight octaves, using one of four basic waveforms. The tone is passed through an envelope generator which regulates the volume of that voice. All three voices are then combined into one audio signal, which is controlled by the master volume and sent to the television or monitor speaker.

Figure 5-1. Producing Electronic Sounds

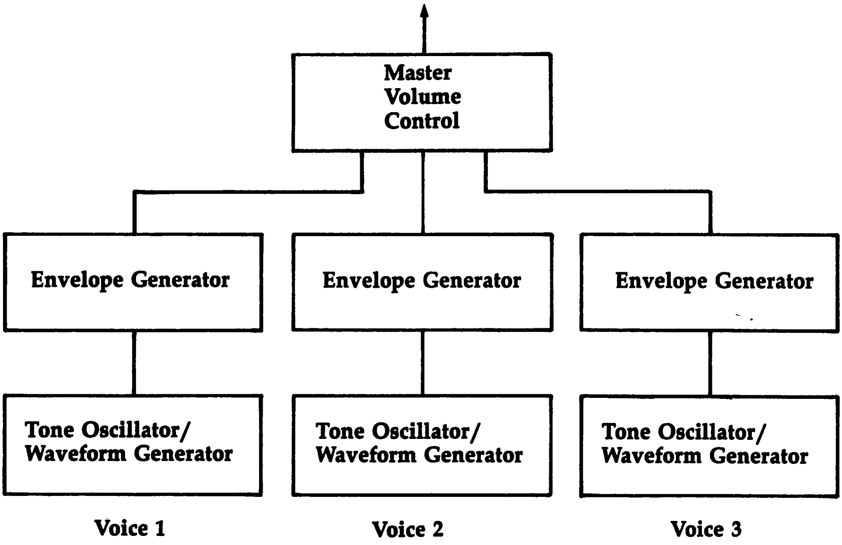

Master Volume Control
Envelope Generator
Envelope Generator
Envelope Generator
Tone Oscillator/ Waveform Generator
Tone Oscillator/ Waveform Generator
Tone Oscillator/ Waveform Generator
Voice 1
Voice 2
Voice 3

Figure 5-1 is a simplified description of the actual design of the SID chip, but it does serve our purposes for the moment. Advanced features not mentioned include a filter and options for synchronization and ring modulation.

Sidplayer fully supports all of the features built into the SID chip. As many as three voices can be played at the same time, each with its own pitch, waveform, and envelope. Sidplayer also supports the filter and all of the related filter controls, plus the synchronization and ring-modulation options.

To further extend the power of the SID chip, there are additional features provided by software control, such as vibrato, portamento, transposing, automatic filtering, and much more.

The most important thing, however, is not that Sidplayer has all these features, but that it makes them so easy to use. The music system includes a carefully designed editing program which lets you edit all three voices. Notes can be entered from keyboard or by joystick and are played as they are entered for immediate feedback. Special commands are available to select things like waveform and envelope settings.

Finally, all music created with Sidplayer can be merged with your own BASIC programs. The music will even play while the program is running. This opens up many possibilities, including animated screen displays that change in time with the music.

Some demonstration songs have been provided cn the disk so that you can hear just what Sidplayer can do. From the menu select SID PLAYER.

Let the Music Play

The first song is Bach's "Two-Part Invention in A Minor." This is the theme music used by Commodore in their television ads.

The second song is, "March of the Wooden Soldiers." It appears on the Sidplayer program screen as WSOLDIER. The final demonstration song is an original composition written using the Sidplayer Editor. It's called ETAL.

When you run SID PLAYER, it displays the message TUNING INSTRUMENTS while it initializes and loads the SID.OBJ file. It also lists a directory of all music files on the disk in three columns. In response to the prompt YOUR REQUEST?, you should type the name of the song you want to play. The song loads, the full title and credit information print, and the song starts playing. It will go something like this:

```
YOUR REQUEST? COMMODORE

TWO PART INVENTION #13
J.S. BACH
COURTESY CRAIG CHAMBERLAIN
```

When the song is finished, you'll be prompted for another selection. You can enter another song name to load and play another tune.

To make a song stop playing before its end, hit any key. PLAYER cancels the current song and prompts you for a new selection. To end the program, press RUN/STOP–RESTORE.

The demonstration tunes cover a variety of music, from classical to modern. The first three songs, COMMODORE, WSOLDIER, and ETAL will give you an idea of what Sidplayer can do.

Fundamentals of Music Theory

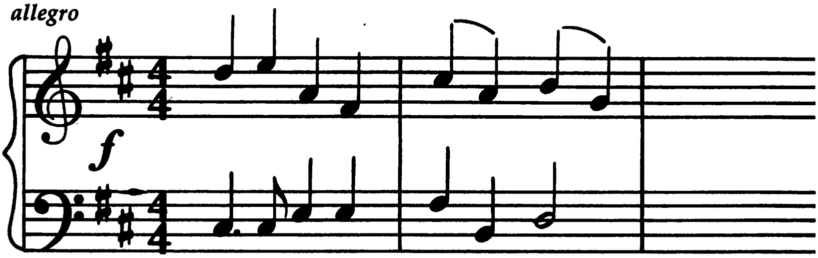
Sidplayer is a lot more than just a music playing program. It's also a complete system for entering and editing music. The Player is accompanied by an Editor which is so easy to use that you don't even have to know how to read sheet music. You may want to skip this section for now and come back to it later. This section offers the rudiments of elementary music theory—it presents the fundamental concepts that will help you get started.

Notation. If a piece is good, the melody will stick in your mind, and you may find yourself humming the tune long after it's finished playing. Sometimes a song can be so good you can't get it out of your mind. By hearing the song, you've learned it and can play it yourself. Songs such as those sung in native American ceremonies have been passed from generation to generation in just this way. As songs get longer and more complex, however, this method of communicating a song becomes less reliable. This is when it's necessary to make a permanent copy of the song on paper, which is the purpose of sheet music. Today, orchestras can faithfully reproduce the great symphonies of Beethoven. These symphonies have survived for nearly two centuries only because they were written down.

To express music on paper, a special form of notation has been developed. This notation is capable of describing every facet of a piece of music, from the order in which to play the notes to such specifics as the style in which they are to be played.

Each group of five horizontal lines is called a staff. At the left edge of each staff is a clef symbol. The clef symbol for the top staff indicates that the staff is a *treble* clef. The bottom staff uses a different clef symbol and is called a *bass* clef. Together, the two staves form a *grand staff,* which is most often used for displaying notes. Figure 5-2 illustrates a grand staff.

Figure 5-2. Grand Staff



In the following text, the different characteristics of notes are introduced one at a time. As each characteristic is discussed, the method for expressing it in notation is also shown. Admittedly, music theory is an extremely complicated subject. What follows is only a simplified explanation of the essential concepts and isn't intended as a complete treatment. Once you understand what's presented here, however, you should be able to read a simple piece of sheet music.

Pitch. When an object is vibrating, its vibrations pass through the air and are detected by your ear as sound. Frequency is the measure of the number of vibrations per unit of time. The most common method of specifying a frequency is in terms of vibrations per second. Frequency is measured in units of *hertz,* abbreviated *Hz.*

The frequency of a sound is interpreted by your ear as a *pitch.* Faster rates of vibration produce higher pitches. Usually, the smaller an instrument, the higher the pitch it can produce. A piccolo can produce a very high pitch, whereas a tuba produces a very low pitch.

Although the human ear can detect a wide range of frequencies, only frequencies occurring at specific intervals are commonly used in music. Let's start with one of these pitches and label it C. This pitch has a frequency of 261.63 Hz, or 261.63 vibrations per second. The sequence of pitches continues, with pitches at the following intervals being named D, E, F, G, A, and B.

B	493.88 Hz
A	440.00 Hz
G	392.00 Hz
F	349.23 Hz
E	329.63 Hz
D	293.66 Hz
C	261.63 Hz (Start here)

When you listen to the sequence of pitches in order, they form a *scale,* but the scale will seem incomplete. One final note, after the B, is needed to complete the scale. This note happens to be another C, related to the earlier C, but at a higher pitch. (The actual mathematical relationship is that the new C occurs at 523.25 Hz, exactly twice the frequency of the first.) It doesn't stop here, though. There's another D after the new C, and a second E after the new D, and so on. In fact, the scale repeats several times, both above and below the original C.

D	1174.70 Hz
C	1046.50 Hz
B	987.77 Hz
A	880.00 Hz
G	783.99 Hz
F	698.46 Hz
E	659.26 Hz
D	587.33 Hz
C	523.25 Hz
B	493.88 Hz
A	440.00 Hz
G	392.00 Hz
F	349.23 Hz
E	329.63 Hz
D	293.66 Hz
C	261.63 Hz (Original C)
B	246.94 Hz
A	220.00 Hz
G	196.00 Hz

F	174.61 Hz
E	164.81 Hz
D	146.83 Hz
C3	130.81 Hz
B	123.47 Hz

The scale repeats with each C. By examining one sequence, from one C to the next, you can see that it consists of eight pitches. Collectively, these eight pitches are called an *octave*. To distinguish this set of pitches from the next, the first set is said to occur one octave lower than the second.

Just as the different pitches in an octave are labeled, so are the different octaves. However, instead of using a letter of the alphabet, a number is used. The piano key for the original C is found at about the middle of the keyboard. This C is called *middle C*, and begins octave 4. Other octaves are numbered relative to the octave containing middle C. The octave immediately above octave 4 is octave 5. The octaves which are of the most use musically are octaves 1–7.

In music notation, the pitch value of a note is represented by its vertical position when drawn on a staff. Thus, C5 (C of the fifth octave) is indicated by placing the note between the second and third lines of the treble staff. The next higher pitch, D5, is indicated by placing the note above the position for C5, except this time the note is placed on the line. For the entire grand staff, the positions for all notes alternate between being on a staff line or between staff lines. See Figure 5-3 for an illustration.

Middle C is a special case. The staff line for C4 is placed halfway between the treble and bass staves. The pitches around middle C must take this variation into account. The separation of the two staves creates some space used for messages and special symbols which give additional information to the performer.

Another special situation is when a note is so high or low in pitch that it goes off the grand staff. In such instances, additional staff lines, called leger lines (see Figure 5-4), are added. The pitch of notes drawn on leger lines is still determined in the normal way, by counting staff lines and seeing whether the note is placed on or between lines.

Figure 5-3. Grand Staff

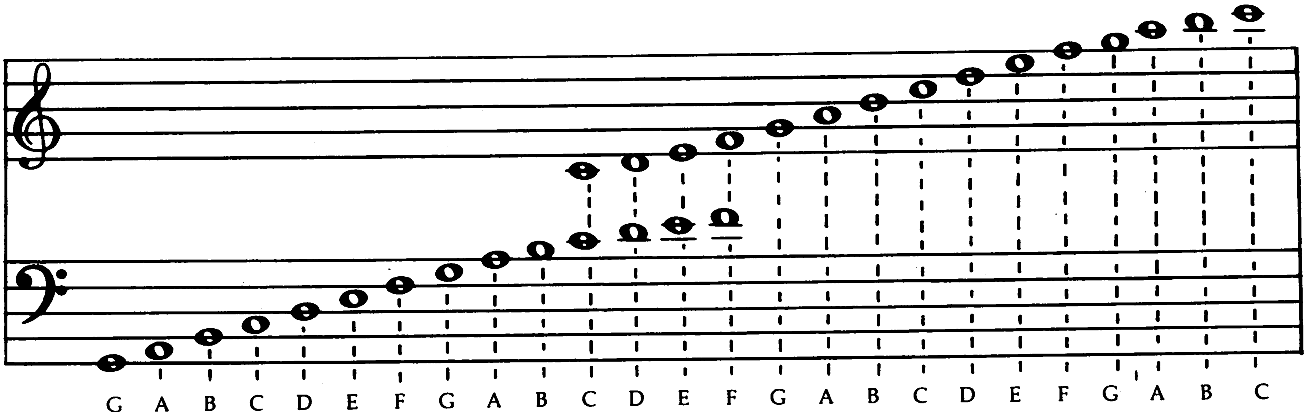
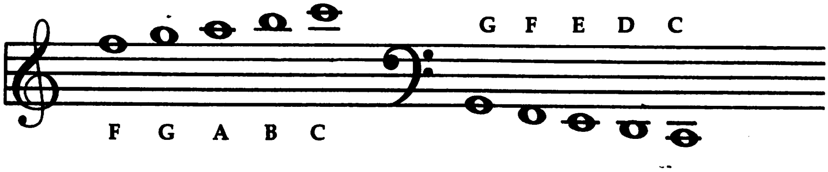


Figure 5-4. Leger Lines



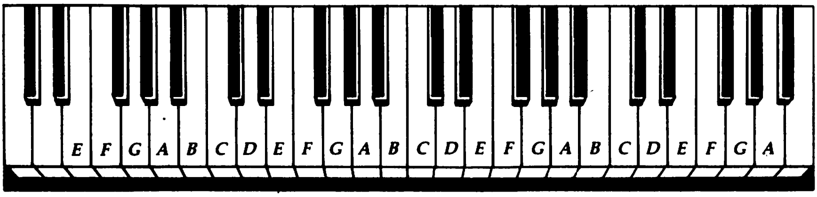
By using the grand staff and leger lines, eight octaves (octaves 0–7) can be displayed.

Sharps and flats. Eight octaves, each containing 7 different pitches, would seem to make a total of 56 pitches. Actually, there are some intermediate pitches between some of these notes. These are called sharps and flats.

	C	
	B	
A-sharp		B-flat
	A	
G-sharp		A-flat
	G	
F-sharp		G-flat
	F	
	E	
D-sharp		E-flat
	D	
C-sharp		D-flat
	C	

A note is sharp if its pitch is a half step above normal. A note is flat if the pitch is a half step below the normal pitch. Notes that are not sharp nor flat are said to be *natural.* Figure 5-5 shows some of the natural notes on the piano keyboard.

Figure 5-5. Piano Keyboard

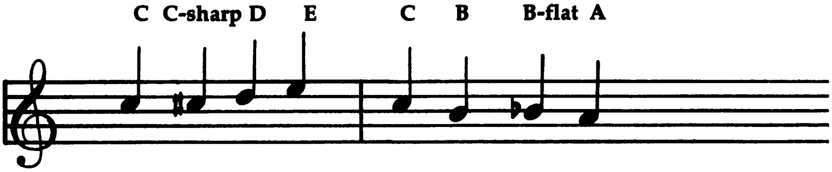


Two important observations should be made. First, every sharp note is equivalent to a flat note. C-sharp and D-flat both denote the same pitch. The difference lies in the viewpoint, whether the intermediate pitch is a half step above C or a half step below D.

Thus far, we've been using the words *sharp* and *flat* for accidental pitches. Another way to indicate that a note is sharp or flat is to use a special symbol. The symbol for a sharp note looks like a slanted pound sign (♯), while the symbol for a flat note looks something like a lowercase letter *B* (♭). The natural symbol is normally not used in front of natural notes.

To show that a note on the grand staff is a sharp or flat note, the appropriate accidental symbol is placed just before the note. The C-sharp and B-flat notes in Figure 5-6 are signified with these symbols. Sharps and flats indicated in this way last only one measure.

Figure 5-6. Accidentals



Including the sharps and flats, one octave consists of 12 different pitches. With eight octaves, the total is now 96 different pitches. Most songs use only notes that come from this palette of 96 pitches.

Key signatures. Just because there are 96 pitches available, does that mean that each one will be used in a song? No, a song may not play in every octave, nor every note within a particular octave. Perhaps it uses just a subset of the 12 pitches within one octave. The selection of notes is determined by the *key* in which the music is written.

The topic of pitch was introduced by starting with a C scale. This is a sequence of notes, starting on C, that continue for one octave. Let's examine the relationship of these notes to the 12 in the entire octave. If the distance between each of the 12 pitches is called a *half step*, the sequence of notes forming the C scale is determined by the following steps: whole, whole, half, whole, whole, whole, half—where a whole step equals two half steps.

C
B
A♯ /B♭
A
G♯ /A♭
G
F♯ /G♭
F
E
D♯ /E♭
D
C♯ /D♭
C

Now apply that sequence of steps again, but this time start the scale at note A.

A
G♯
F♯
E
D
C♯
B
A (Start here)

This scale contains three sharp notes, as opposed to the earlier scale which contained none. The sharp notes replaced their natural counterparts. This scale is said to be written in the *key of A*. A song written in the key of A will normally use only this set of pitches in each octave. This means that we're back to a situation where we have to deal with only seven pitches per octave.

You can start a scale on any note, and for every starting note, there is a different combination of sharp or flat notes used. Here's another example, this time using flats:

B♭
A
G
F
E♭
D
C
B♭ (Start here)

This is the key of B-flat. The notes were determined by using the sequence of half and whole steps given earlier. The key of B-flat contains two flat notes, B-flat and E-flat. The notes B-natural and E-natural will not normally be used by a song written in the key of B-flat.

Table 5-1 is a complete listing of all the major keys. The keys with less than five sharps or flats are the ones used most often.

Table 5-1. Keys

Key	Notes								Sharps/Flats	
C	C	D	E	F	G	A	B	C	0	
G	G	A	B	C	D	E	F♯	G	1♯	(F♯)
D	D	E	F♯	G	A	B	C♯	D	2♯	(F♯, C♯)
A	A	B	C♯	D	E	F♯	G♯	A	3♯	(F♯, C♯, G♯)
E	E	F♯	F♯	A	B	C♯	D♯	E	4♯	(F♯, C♯, G♯, D♯)
B	B	C♯	D♯	E	F♯	G♯	A♯	B	5♯	(F♯, C♯, G♯, D♯, A♯)
F♯	F♯	G♯	A♯	B	C♯	D♯	E♯	F♯	6♯	(F♯, C♯, G♯, D♯, A♯, E♯)
C♯	C♯	D♯	E♯	F♯	G♯	A♯	B♯	C♯	7♯	(F♯, C♯, G♯, D♯, A♯, E♯, B♯)
F	F	G	A	B♭	C	D	E	F	1♭	(B♭)
B♭	B♭	C	D	E♭	F	G	A	B♭	2♭	(B♭, E♭)
E♭	E♭	F	G	A♭	B♭	C	D	E♭	3♭	(B♭, E♭, A♭)
A♭	A♭	B♭	C	D	E♭	F	G	A♭	4♭	(B♭, E♭, A♭, D♭)
D♭	D♭	E♭	F	G♭	A♭	B♭	C	D♭	5♭	(B♭, E♭, A♭, D♭, G♭)
G♭	G♭	A♭	B♭	C♭	D♭	E♭	F	G♭	6♭	(B♭, E♭, A♭, D♭, G♭, C♭)
C♭	C♭	D♭	E♭	F♭	G♭	A♭	B♭	C♭	7♭	(B♭, E♭, A♭, D♭, G♭, C♭, F♭)

If you study Table 5-1 carefully, you'll notice some patterns. For example, each key which contains sharp notes contains F#. The key of G has F# as its only sharp note. The key of D keeps the F#, but adds C#. Each successive key adds one more sharp note, while retaining all the other sharp notes from before. This pattern works in the same way for keys containing flat notes, starting with the note Bb.

Most of the time you can determine the key in which a piece of music is written by counting the number of sharp or flat symbols near the clef symbols on the grand staff. If no sharp or flat symbols appear there, the music is written in the key of C. If one sharp symbol is displayed, the piece is written in the key of G. Two sharp symbols mean that the key of D is to be used, and so on. Likewise, one flat symbol indicates the key of F, two indicate the key of B-flat, on up to seven flat symbols, which indicate the key of C-flat.

Just as the number of sharp or flat symbols is important, so is their position. The sharp symbol for F# is always placed on the line that designates note F. Furthermore, when a sharp symbol is put next to the clef symbol, it has the effect of automatically placing a sharp symbol in front of every note on that line. A sharp symbol on line F means that all notes placed on the grand staff in F positions are to be played as F-sharps. Of course, the same is true when flats are used. A flat symbol placed near the clef on the line for B means that all B notes should be played as B-flats.

Sharp and flat symbols placed after a clef symbol are called a *key signature*. The use of a key signature saves a lot of work when writing music, because it's no longer necessary to write a sharp or flat symbol in front of every note that needs one.

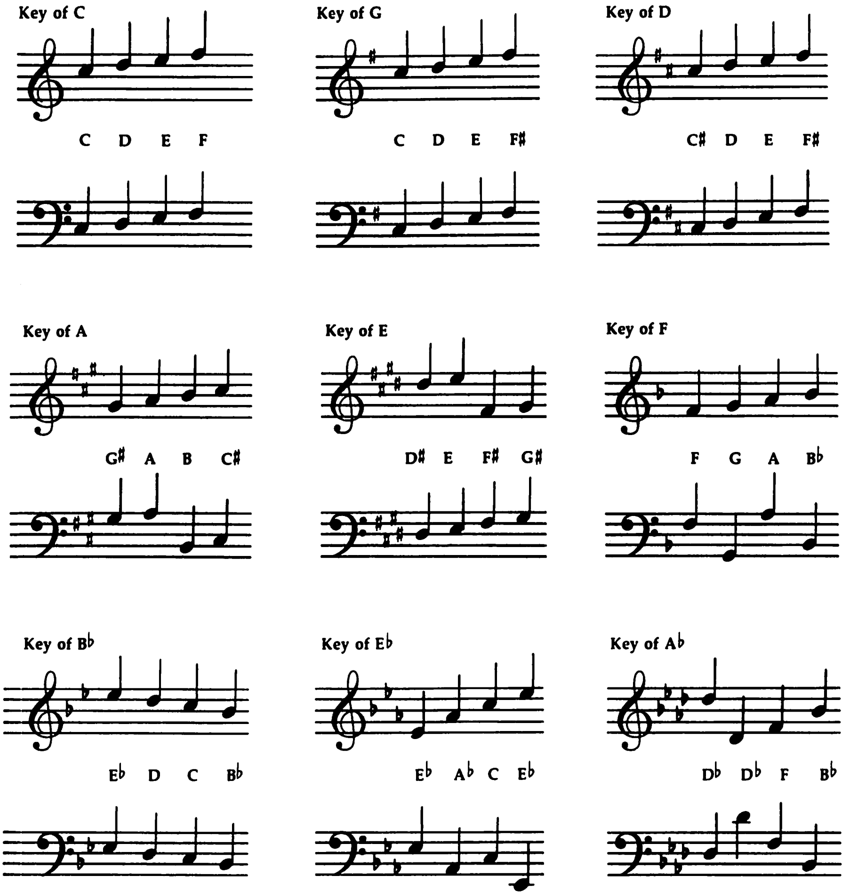
Figure 5-7 contains some examples of key signatures. Since all keys that contain sharps contain F#, all of these keys have a sharp symbol at the F position. Each successive key adds a sharp symbol at a new position while retaining all the old ones. Also notice that a sharp or flat on one line affects not only the notes on that line, but the corresponding notes in the octaves above and below as well.

Duration. The vertical position of a note on the grand staff determines its pitch. The horizontal direction of the staff indicates time. A sequence of notes is played in order from left to right, just as text is read from left to right. By putting the pitches together in a pleasing order, you'll create a melody, the basis for a song.

Pitch, however, is only one major characteristic of a note. Another important quality of a note is its duration. In a song, notes are not always played at the rate of one note every beat. Sometimes a note may be played for two beats. Other times, two notes might be played within the span of one beat, meaning that each note is half a beat long. Thus, every note on the staff is going to have to specify not only its pitch, but also its duration in terms of beats.

The duration of a note is indicated by its shape. The standard note you've been using thus far is formally called a *quarter note,* and is drawn with a stem and a filled-in oval at the bottom. If we assume a quarter note plays for a duration of one beat, then twice that length, two beats, is indicated by a *half note,* which looks like a quarter note except that the oval is not filled in. Twice the length of a half note is a *whole note,* which plays for four beats and looks like a half note without a stem.

Figure 5-7. Key Signatures

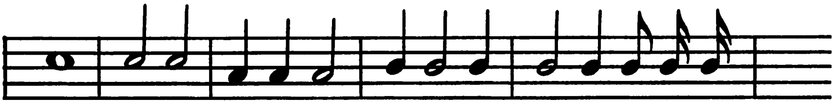


In the other direction, for durations less than one beat, the symbol for a quarter note is used, but flags are added at the top of the stem. An *eighth note* plays for half a beat and has one flag. A *sixteenth note* has two flags. Four sixteenth notes are equal in duration to one quarter note. Thirty-second and sixty-fourth notes do exist, but they're not used very often. Figure 5-8 shows these notes and how they're written. Take a moment to look over it.

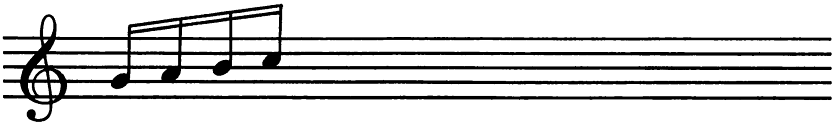
Figure 5-8. Duration

4 beats	Whole note	𝅝	1/2 beat	Eighth note	♪
2 beats	Half note	𝅗𝅥	1/4 beat	Sixteenth note	𝅘𝅥𝅯
1 beat	Quarter note	♩	1/8 beat	Thirty-second note	𝅘𝅥𝅰

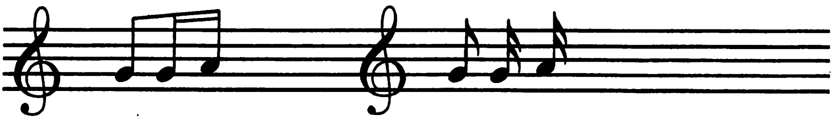
The following combinations are all equivalent in duration to one whole note.



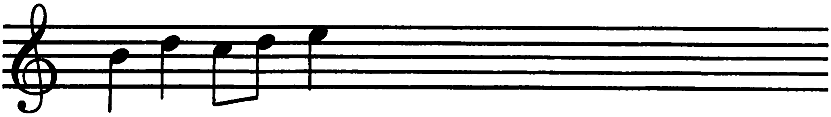
One convention in displaying durations less than one beat is to combine notes of equal duration in sets. Two eighth notes can be drawn by extending the flag from the first one to the top of the stem of the second one. This can also be done with sixteenth notes, except two lines connect the tops of the stems, because sixteenth notes have two flags.



Notes of different durations can be combined. The notes shown on the left are the same as those shown on the right.



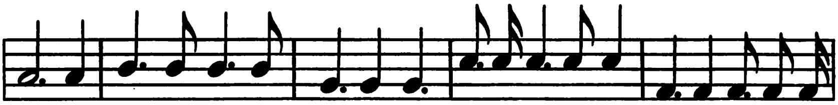
Notes can even be drawn upside down. This is done only when the notes would appear near the top of a staff. The oval portion of the note stays in the same place, so the pitch is not affected. Drawing a note upside down does not affect its duration either.



Dotted notes. With just a few different durations, it's possible to create a variety of different rhythms. But there are still some durations that cannot be expressed using only the notes you've seen so far. For example, how do you show that a note should be played for three beats? Situations like this require the use of dotted notes.

When a dot is placed after a note, it means that the note should be played for one and a half times the normal duration. Given a dotted half note, the half note portion is two beats, and half of that is one beat, for a total of three. A dotted whole note plays for six beats (four beats for the whole note and two beats for the dot). And a dotted quarter note? That plays for one and a half beats.

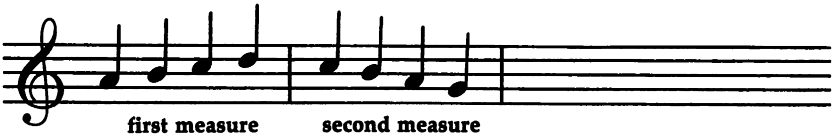
Using the dot, here are some more note combinations which total four beats.



Notice that the dot always appears to the right of the note. If you see a dot placed above or below a note, it has a different meaning and does not affect the note's duration. These dot placements are explained a bit later.

Measures. A song is just a long sequence of notes of different pitches and durations. To make it easier to deal with pitches, they're separated into groups called octaves. Likewise, to make it easier to work with a sequence of notes, the notes are often divided into groups called measures, with each measure consisting of the same number of beats. A common number of beats per measure is four.

In sheet music, a measure is formed by placing a vertical line called a *bar* between each group of notes on the staff.



Measures are used mainly for organization and reference. It's much easier to refer to a notє as being the second note in the twenty-third measure than it is to refer to the one hundred forty-seventh note.

Each measure must have the same total duration. Since this total duration is often four beats, or one whole note, you can see why the note for one beat is called a quarter note.

Tempo. You've seen that the length of a note is expressed in beats, and that notes can be organized into groups called measures, which all have the same number of beats. The question is, how long is a beat?

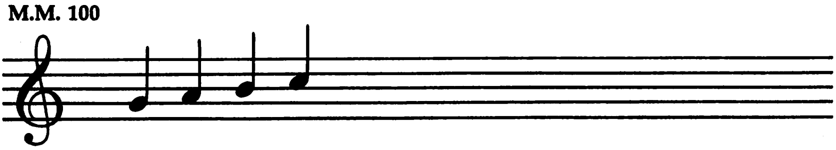
A beat is a unit of time. The shorter the amount of time for each beat, the faster they'll occur. If the time is longer, the beats won't occur as often.

The rate at which the beats occur is called the *tempo*. The faster the tempo, the more quickly the notes are played. At a slower tempo, each beat lasts for a longer amount of time. Another way to look at it is to say that for a fixed amount of time, such as one minute, there will be more beats at a fast tempo than there will be at a slow tempo.

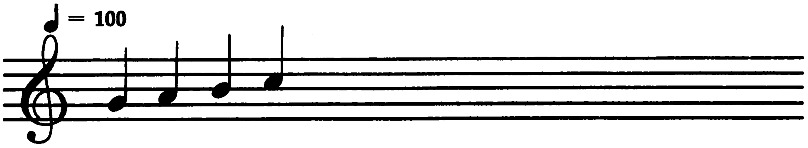
The relationships of quarter notes to half notes and other notes still hold; a half note will always be twice as long as a quarter note. It's just the actual time lengths that change.

The standard method of measuring a tempo is to specify the beats per minute. An average tempo is about 100 beats per minute. A tempo of 60 means that there will be 1 beat every second, which is rather slow. A tempo of 150 is more than 2 beats every second, which is relatively fast.

The tempo is a very important part of a song. A beautiful melody can be ruined if it's played too fast or too slow. Therefore, sheet music usually indicates the proper tempo. At the top of the sheet music you should find the letters *M.M.*, followed by a number. This number indicates the beats per minute, and defines the tempo that should be used in playing the song.



Another way to show the tempo is to replace the M.M. with a quarter note and an equal sign. The quarter note is used because it represents one beat. Either way works just as well.



Using a number for the tempo is convenient because it can be used when setting a metronome. But there's another method of specifying the tempo. Often, you'll see words such as *adagio* or *allegro* on sheet music. What do they mean? Just like numbers, they specify tempo. Here's a list of most of the tempo terminology, in order from slowest to fastest:

lento	Very slow
grave	Slow, solemn
largo	Broad
adagio	Leisurely
andante	Walking
andantino	A little faster walk
moderato	Moderate
allegretto	Rather fast
allegro	Fast
vivace	Lively
presto	Very fast
prestissimo	As fast as possible

These values may be modified by one of the following words:

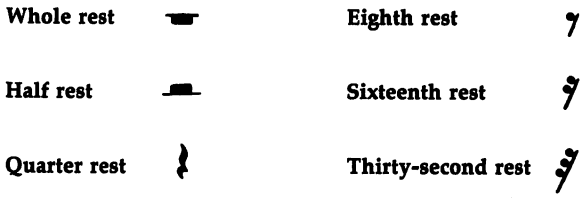
molto	Very
meno	Less
piu	More

The tempo notation *moderato* indicates medium speed, which roughly corresponds to M.M. 100.

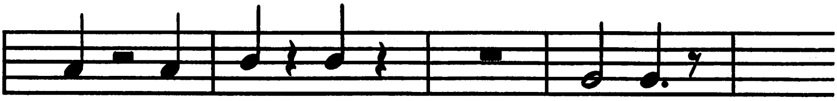
Rests. Pitch and duration are the two most important parts of a note. There's something similar to a note, however, which has only duration and no pitch. It's called a rest. For the specified amount of time, no tone is produced.

There's a different rest symbol for each duration. Because the idea of pitch does not apply, the vertical position of a rest does not matter, so it's usually placed in the middle of the staff. A whole rest is drawn as a small block placed right below the second staff line. A half rest looks the same except that the block is placed on top of the third staff line. A quarter rest is a symbol that defies description. Eighth, sixteenth, and thirty-second rests are all drawn as slanted stems with the proper number of flags to the left of the stem. Figure 5-9 shows the rest symbols.

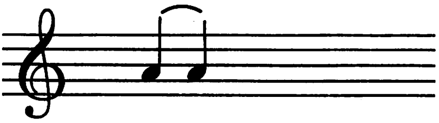
Figure 5-9. Rests



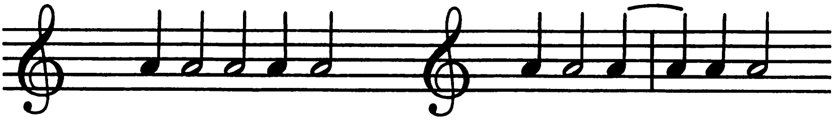
Here are some combinations of notes and rests. They've been divided into measures to show that each group has a total duration of four beats.



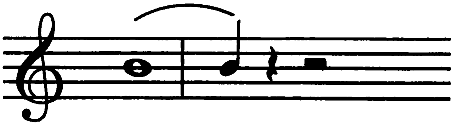
Ties and slurs. Another special symbol is the *tie* symbol. Two notes are tied together when connected by a symbol that looks like a curved line or arc. The tie means that the two notes are to be played together as one long note, with no break in volume. Thus, two quarter notes tied together will play just like a half note.



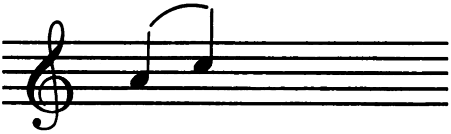
The reason for using ties is that the effect of a tie can extend across one measure or beat into another. In the following two sequences of notes, each sequence sounds the same when played, but the first one cannot be divided into measures.



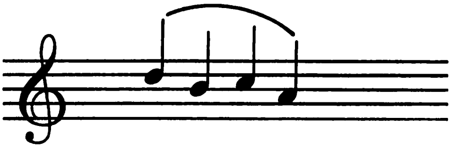
The next example demonstrates the use of a tie to create a note five beats long.



Another application of the tie symbol is to connect notes of different pitches. In this case, the tie is called a *slur* and may be used within a measure as well as between two measures. Playing two quarter notes slurred together is like playing a half note that changes its pitch halfway through playing.



Sometimes a very long tie symbol is used over a long stretch of notes. This produces a smooth, legato effect when the notes are played.



Volume. Yet another major characteristic of a note is its volume. Some parts of a song can be emphasized if they're played loudly. Other parts may be subdued by being played quietly. The level of loudness or softness of a piece of music is referred to as *dynamics.*

Dynamics are indicated on sheet music by letters which appear between the two staves of the grand staff. These letters are listed below, in order from loudest to softest.

fff	fortississimo
ff	fortissimo
f	forte
mf	mezzo forte
mp	mezzo piano
p	piano
pp	pianissimo
ppp	pianississimo

These volume levels range from very very loud (fff) to very very soft (ppp). Extremes such as ffff or pppp are not used very often.

The term *dynamics* should not be confused with the concept of an envelope, which describes the changes in volume as an individual note is played.

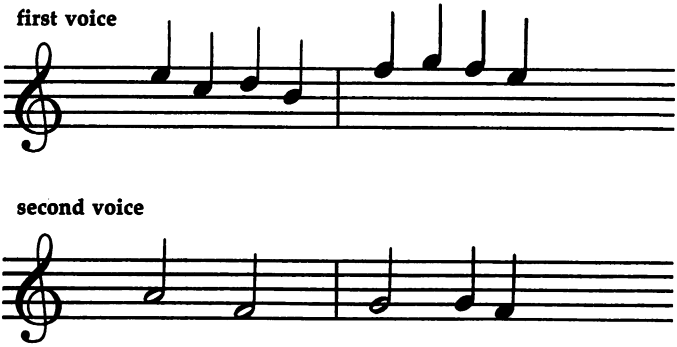
Multiple voices. You've seen the essential characteristics of individual notes and how notes can be combined into groups called measures. The music can then be sung or played on an instrument. The next step is to have several voices or instruments playing at the same time.

A set of notes for one singer or one instrument is generally referred to as one *voice*. With two voices playing simultaneously, one voice can play a melody while the other voice plays a bass part to give a little more body to the song. If a third voice is added, it can be used for harmony or for percussion effects, like drums or cymbals.

Each voice is independent of the others and can play its own notes of different pitches and durations. This brings up only one problem; there must be a way of keeping the voices synchronized. They should start together and end together.

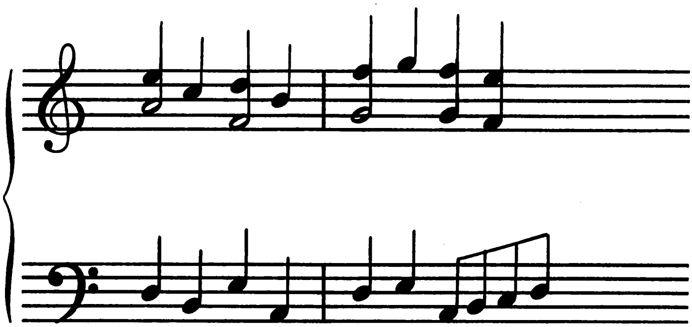
Fortunately, the concept of tempo and the use of measures solve this problem. The voices may be independent, but one thing they must have in common is the tempo. The tempo establishes a beat which all voices can follow. The notes in each voice are divided into measures. Then, even though the durations within a measure may differ for each voice, at least the voices will always be on the same measure at any given instant. Figure 5-10 shows this concept.

Figure 5-10. Multiple Voices, Example 1



The horizontal direction of the grand staff corresponds to time. Because multiple voices are synchronized according to tempo, it's possible to represent more than one voice on just one grand staff. Within each measure, the notes for all of the voices are drawn. It's a rather simple matter to determine which notes go with which voice. Usually, the topmost notes are for the first voice, the notes below those are for the next voice, and so on, with the bottommost notes assigned to the last voice.

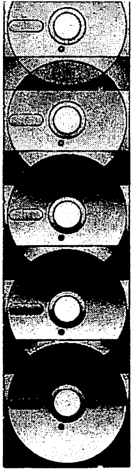
Figure 5-11. Multiple Voices, Example 2



Summary

- Notes are shown on a grand staff which consists of the treble and bass staves. Each staff has its own clef symbol.
- The most important characteristics of a note are its pitch and duration.
- Pitches occur in groups called octaves. Each octave contains the pitches labeled C, D, E, F, G, A, and B, plus five intermediate pitches called sharps and flats.
- Only some of the pitches in each octave are used, depending on the key in which the music is written.
- Pitch is indicated by the vertical position of a note on the grand staff. The note may be drawn on a staff line or between staff lines.
- An accidental (a sharp or flat symbol) may be placed immediately before the note to indicate that the note is a sharp or a flat.
- A key signature is indicated by placing sharp or flat symbols near the clef symbol.
- Durations are specified in terms of beats. Common units of duration are whole note, half note, quarter note, and so on, down to sixty-fourth note.
- Durations are indicated by the shapes of the notes.
- Placing a dot after the note means that its duration should be one and a half times normal.
- Notes are organized into groups of equal total duration called measures.
- Measures are indicated by a single vertical line, called a bar, that crosses the staff lines.
- The rate at which the beats occur is called the tempo and is measured in terms of beats per minute.
- The tempo is indicated by an M.M. marking at the beginning of the music.
- A rest is similar to a note in that it plays for a certain duration, but it produces no tone so it has no pitch.
- Rests are indicated by special symbols, one for each duration.
- An arc-type symbol connecting two notes is used to indicate a tie or slur.
- The general volume level of a piece of music is specified by dynamics.
- Dynamics are indicated by letters that appear between the two staves.

- The notes for one singer or one instrument form one voice.
- Several voices can be represented on the same grand staff.
- When played, multiple voices stay synchronized because they share the same tempo.



LOAD "EDITOR",8

loads "editor" & "editor.obj" files

The Editor

You've been introduced to the Sidplayer, which plays songs. You've even seen some of the elements of music. But in order to play your own songs or tunes, you have to create them.

That's what the ~~Sid Editor is~~ for. It's ~~used to enter, edit~~, and debug up to three voices of a song. The program contains features which make music editing easy. Since there are so many features and the program can appear rather intimidating the first time you use it, be assured that you'll see exactly how each feature works.

Using the Sid Editor

To use the Editor, ~~plug a joystick into port~~ 2 and select the Sid Editor from the menu. The EDITOR.OBJ file automatically loads and the screen blanks while the program initializes. (If you want to copy EDITOR to another disk, you must also copy the file EDITOR.OBJ, the machine language portion of EDITOR.) The Sid Editor is a graphically complex program, using redefined characters, sprites, and raster scan interrupts, so it takes awhile to initialize. When the program is ready, it displays a main menu.

1) PLAY MUSIC
2) EDIT MUSIC
3) LOAD MUSIC FILE
4) SAVE MUSIC FILE
5) DISK DIRECTORY or
5) (NOT AVAILABLE)
6) QUIT

EDIT Music

Press the 2 key to ~~select EDIT~~, and ~~then the 1 key to choose voice~~ 1. The display changes to show the editing screen.

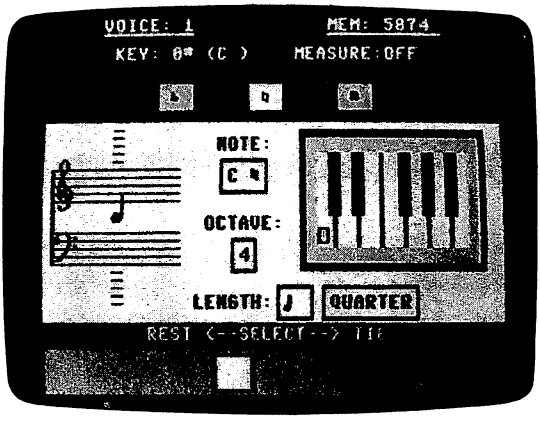
The first thing you'll notice is that it's divided into different levels. The top level shows the current voice number and the amount of free memory remaining. The next level displays the current key signature and tells whether the measure feature is on or off. The level below that has one of the three accidental symbols highlighted.

The next level is the main level, where notes are selected. The main level displays the current pitch in three different forms: as a note on the grand staff, as a piano key, and as a letter and octave number. The current duration is indicated by a symbol and a word. The level below this main level is used for entering rests and ties. The bottom level contains a box in which entered notes appear.

In the main level, this is the method for using the joystick:

1. Push the stick up or down to change the current pitch.
2. Push the stick left or right to change the current duration.
3. Press and release the joystick button to enter a note.

While you're in the main level, pushing the stick up or down changes the pitch. The quarter note on the grand staff moves, the next key on the piano is indicated, and the written display changes. A full eight octaves are available by pushing the joystick.



VOICE: 1 MEM: 5874
KEY: 0* (C) MEASURE:OFF
NOTE: C
OCTAVE: 4
LENGTH: QUARTER
REST <--SELECT--> TIE

The Sidplayer Editor.

Take a moment to enter a few notes. Just select a pitch and press the joystick button. A note appears in the box in the bottom level and then scrolls to the left. The note looks like a quarter note, the current duration, and has the note name and octave number written above it. As you enter more notes, each one displays in the box—all notes scroll to the left to make room for the next.

If you press the button without changing the pitch, the entered note is set at the same pitch as the previous note.

To change the duration, push the stick left or right. Notice that all durations also have a dot option, except the thirty-second note. Now press the button. The note which shows in the box is in the current pitch, but it has a new duration.

For the moment, ignore the ~~durations~~ marked ~~UTILITY and ABS SET~~.

Now that you've entered a few notes, you might like to hear your musical creation. ~~Return to the main menu by pressing the function key f1.~~ Then press the 1 key for PLAY, followed by the RETURN key to play the voice. When the voice is through, the program waits for a new selection from the main menu.

To continue editing, press the 2 key to select EDIT and then the 1 key to choose voice 1 again. The display switches back to the editing screen, and the notes that you've entered appear in the same position as before.

As notes are entered, the previous notes are scrolled to the left. Since the ~~bottom level can display only a few notes~~, notes have to scroll off the screen. Sometimes you'll want to scroll those notes back onto the screen to review them. ~~Scrolling can be done manually by using the two cursor keys.~~

Press the cursor key on the left (ignore the up and down markings). All the notes scroll to the right. The note which scrolled off the screen appears at the left edge, and the most recently entered note moves back into the box. Every time you press this cursor key, the notes scroll one position.

As the notes move, the current pitch and duration change to reflect each note which appears. You can scroll as far as you want. Scrolling stops when you come to the beginning of the voice.

Press the cursor key on the right to scroll in the opposite direction. Scrolling stops at the end of the voice.

Replacing notes. It's easy to correct a mistake made while entering notes. Use the cursor keys to scroll the notes until the one which needs to be changed appears in the box. Select the correct pitch and duration and press the button. The old note is replaced with the new one. You can then scroll back to the end of the voice and continue entering notes.

Insert. If you miss a note and need to insert one, scroll the notes until the insertion point is reached. The note in the box will be to the *right* of the inserted note. Next, press SHIFT–INST/DEL. The note in the box and all notes to its right scroll to the right, creating a blank in which you can enter a note.

If necessary, you can insert several blanks. The only thing you cannot do while blanks exist is scroll or return to main menu. You must fill in all of them to make the voice complete before adding new notes at the end.

Delete. To get rid of an extra note or blank, just press the DEL key (the unshifted INST/DEL key). The note or blank in the box is deleted, and the notes to its right scroll to the left to fill in the gap.

When deleting notes, remember that the keyboard is buffered for up to ten keystrokes. If you press the DEL key a second time, before the Editor is finished deleting the first note, a second note will also be deleted.

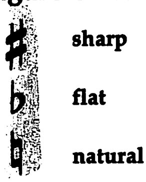
Clear. The clear feature is used when you want to delete all the notes from the current note to the end of the voice. This allows you to delete several notes at once. Press SHIFT–CLR/HOME. Since clearing can be disastrous when done accidentally, there's a confirmation prompt. Press the Y key to erase the note currently in the box and all the notes to its right, or hit the N key to cancel the clear.

By scrolling to the beginning of the voice and using SHIFT–CLR/HOME, you can erase the entire voice.

Moving to the beginning or end of the voice. To move to the beginning or end of a voice, use the function keys f2 and f4, respectively. Pressing SHIFT-f1 will immediately take you to the beginning of the voice. Likewise, pressing SHIFT-f3 takes you to the end of the voice.

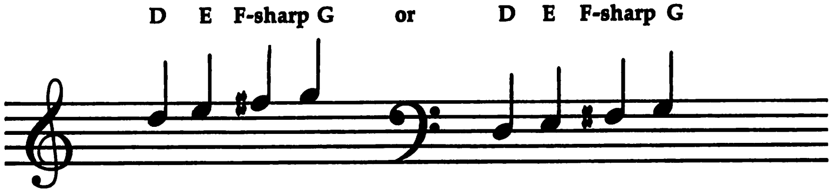
Accidentals. Figure 5-12 shows three symbols, called *accidentals,* that are often found in sheet music.

Figure 5-12. Accidentals



When you find one of these symbols placed before a note in the sheet music, it means that the pitch of the note should be adjusted up or down slightly. Say, for instance, that you find an F note with a sharp in front of it, something like Figure 5-13.

Figure 5-13. F-sharp



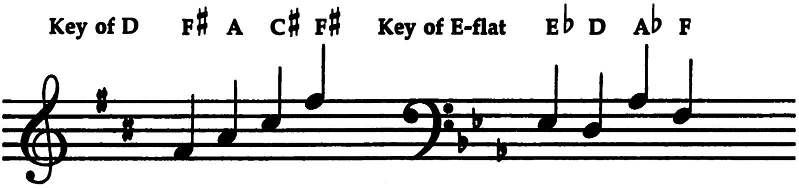
This means that you should enter the note with the pitch F-sharp instead of just an F natural. To do so, use the joystick to move to pitch F, then press the + key (it's on the top row of the keyboard) to select the sharp before you press the button to enter the note.

A sharp increases the pitch of a note. The pitch F-sharp is halfway between pitches F and G. A flat, on the other hand, decreases a note's pitch. To make a pitch flat, press the — key (right beside the + key). If you see a flat symbol before a note, you must move to the pitch, hit the — key, then press the button to enter the note.

To cancel a sharp or flat, press the English pound sign (£) to select the natural pitch.

Key signature. Sometimes you'll find one or more sharp or flat symbols at the left edge of the staff, next to the clef symbol, as in Figure 5-14.

Figure 5-14. Key Signatures



These symbols form the *key signature*. The number of sharps or flats in the key signature determines the key in which the music is written. As shown in Figure 5-14, the key of D has two sharps, and the key of E-flat has three flats.

The presence of a sharp or flat in a key signature means that all notes at that position on the staff should be treated as if they had a sharp or flat symbol in front of them. For example, a sharp appearing at the position for F in a key signature means that all F notes, in all octaves, should be entered as F-sharps.

Since having to remember which notes should have a sharp or flat can make note entry much more difficult, the Editor has a feature to help. All you have to do is tell the Editor how many sharps or flats are in the key signature, and the Editor automatically selects the appropriate pitches for you.

Look at the level near the top of the screen which displays the number *0* and the letter C. The number tells how many sharps or flats are in the current key, and the letter gives the name of the key. The default key is C, which has no sharps or flats.

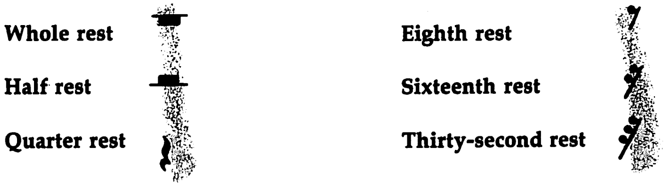
Let's say that the key signature shows two sharp symbols, at the positions for F and C. Press SHIFT-+ (plus key) *twice* to select this key signature. The display changes to show two sharps in the key of D. Now when you push the joystick, the pitches F and C are automatically sharped.

To select a key with more sharps, press SHIFT-+ as necessary. To reduce the number of sharps or to select a key with flats, press SHIFT-— (minus key). Keep pressing until the desired number of sharps or flats is displayed.

If the key has been set and you come across an accidental symbol in front of a note, that accidental overrides what was set by the key signature. You'll have to use the +, —, or £ key to change the pitch for that note.

Rest. A voice does not have to play notes constantly. A rest tells a voice how long it should be silent. Rests have duration but no pitch. Different rest symbols are used for different durations. Figure 5-15 illustrates these symbols.

Figure 5-15. Rests



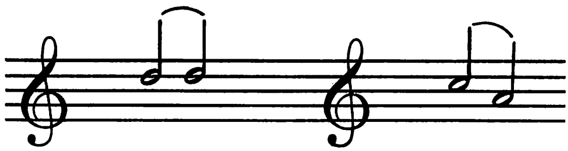
To select a rest, press the R key. The note name in the main level alters to read (R), and the word *REST* in the level below the main level displays in reverse letters. All notes now entered will be rests.

The Editor cannot display the duration symbols for rests. Rests of different durations can be selected by pushing the joystick left or right, but the Editor will still display the duration symbols for normal notes.

The rest mode stays in effect until canceled. To cancel it, either press the R key again, or push the stick up or down to change the pitch.

Tie/slur. Two notes are tied or slurred when they're connected by a symbol which looks like an arc. When the symbol joins two notes of the same pitch, it's called a tie. If the two notes have different pitches, the symbol is called a slur. Notes tied or slurred are played with no break in volume.

Figure 5-16. Ties and Slurs



A tie or slur is selected by pressing the slash (/) key (near the right on the bottom row of the keyboard). The word *TIE* in the level below changes to reversed letters. The next note entered is followed by a curved symbol, indicating a tie or slur.

Unlike the rest mode, the tie/slur mode is automatically canceled after a note is entered. To cancel it without entering a note, press the / key a second time.

Special Options

In addition to notes, the Editor lets you enter commands to control things like tempo, volume, waveform, and envelope. Since quite a variety of commands are supported, there's no room to display the choices on the editing screen, so a separate screen is used. Press function key f3 to switch from the editing screen to the special option screen.

The commands are organized by headings like TEMPO and WAVEFORM. Each command has its own three-letter abbreviation. One of the commands is displayed inside a box, which you can move by pushing the joystick. If the box is moved off the end of one row or column, it wraps around to the other side.

The window in the upper-right corner of the screen displays the full name of the current command. This display changes as the box is moved from one command to another.

To enter a command, first press the joystick button. The window then shows the number range for the selected command. All commands require that a number be entered from the keyboard. Once a number is typed and the RETURN key is pressed, the command is actually entered. The command name and the number show up in the bottom row, the command and the previous notes scroll to the left, and the display returns to the editing screen. If you want to enter another command, you have to press f3 again.

There will be times when you'll want to enter two or more commands—having to return to the special option screen each time can be inconvenient. Fortunately, there's a way to enter a command without having the display switch back to the editing screen. After you press the button and type the number, press SHIFT-RETURN instead of RETURN. The command is entered, but the screen doesn't change.

If you accidentally press the button on the wrong command, press only RETURN without typing a number. The command is not entered, and you'll be able to choose another.

Commands can be scrolled, inserted, and deleted, just like notes. The INST/DEL key can also be used while you're in the special options screen.

To return to the editing screen without entering a command, press the f1 key.

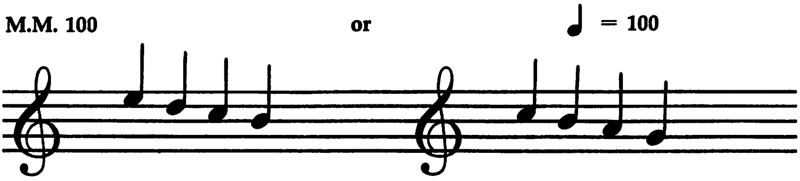
We'll discuss just three commands here. The other commands are for advanced applications and are explained in later chapters.

Tempo. The tempo determines how quickly the notes are played; it is normally specified at the beginning of a song by one of the two methods shown in Figure 5-17.

The TEM command, used to set the tempo, is usually placed at the beginning of a voice. The number part of the command is the desired tempo value. In the preceding example, this number was 100.

There are some restrictions regarding which tempo values can be used. Sidplayer supports only a limited number of

Figure 5-17. Set Tempo



tempo values, and of those some do not permit the use of certain durations. Not all tempo values support sixteenth or thirty-second notes or dotted whole notes. Table 5-2 lists all the available tempo values and their restrictions.

Table 5-2. Tempo Values and Restrictions

Tempo	Restrictions
900	No thirty-second notes
600	No sixteenth or thirty-second notes
450	None
360	No sixteenth or thirty-second notes
300	No thirty-second notes
257	No sixteenth or thirty-second notes
225	None
200	No sixteenth or thirty-second notes
180	No thirty-second notes
163	No sixteenth or thirty-second notes
150	None
138	No sixteenth or thirty-second notes
128	No thirty-second notes
120	No sixteenth or thirty-second notes
112	None
105	No sixteenth or thirty-second notes
100	No thirty-second notes
94	No sixteenth or thirty-second notes
90	None
85	No sixteenth or thirty-second notes
81	No thirty-second notes
78	No sixteenth or thirty-second notes
75	None
72	No sixteenth or thirty-second notes
69	No thirty-second notes
66	No sixteenth or thirty-second notes
64	None
62	No sixteenth or thirty-second notes
60	No thirty-second notes
58	No sixteenth or thirty-second notes
56	None

Those tempos which do not support a sixteenth note do not support a dotted eighth note. Likewise, those tempos which do not support thirty-second notes do not support dotted sixteenth notes.

Tempo values 81 and below do not support dotted whole notes, but this is no major problem. A dotted whole note can be simulated by tying a half note to the whole note.

When you type the number for the TEM command, the Editor uses the closest available tempo value. For example, if you type 160, the command TEM 163 is actually entered because tempo 160 is not available.

In most cases, it doesn't matter if the exact tempo is not available. There's little difference between M.M. 160 and M.M. 163. The only complication would be if the song used sixteenth or thirty-second notes, which are not supported in M.M. 163. If that was the case, the tempo value 150 would have to be used.

If you try to use durations like sixteenth notes in a tempo which does not support them, the Editor prints the error message ILLEGAL DURATION when the voice is played. If this should happen, press any key to acknowledge the error, and then change the tempo to one that *does* support the duration.

Sometimes a word like *adagio* or *allegro* is used in place of a number. In these situations, you have to choose the appropriate tempo value based on the information given in the previous article.

If the sheet music does not specify any tempo at all, just use whichever tempo sounds best to you. If you don't enter a TEM command, M.M. 100 is used.

Volume. Like tempo, the general volume level (dynamics) of a song is also usually indicated at its beginning. The level is specified with the letters *p* for piano (soft) and *f* for forte (loud).

The master volume of the SID chip can range from 0 through 15, with 15 being the loudest and 0 being off. This volume level is set by the VOL command.

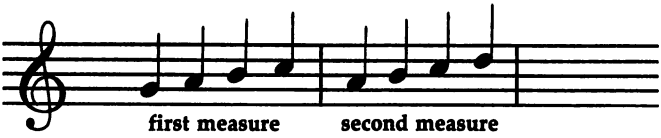
It's recommended that you not use volume levels above 12, because they can cause notes to distort. If the sheet music does not specify a volume, do not enter a VOL command. The default volume level of 8 (mf) will then be used.

Here are the suggested volume levels for various dynamics markings.

Dynamic	Volume
fff	12
ff	11
f	10
mf	8
mp	7
p	5
pp	4
ppp	3

Measures. Notes are organized into groups called measures. Measures help the performer keep in time with other players and are convenient when trying to locate a particular note in a song.

Figure 5-18. Measures



To help keep your place in a voice, measure markers are available with the MS# command. This command is entered with a number ranging from 0 through 999.

Measures in sheet music usually are not numbered, so you'll have to number them yourself. Start at the first measure and write the number *1* above it. Number the next measure *2*, and so on, for the length of the song. Then, before you enter the notes for each measure, enter the appropriate measure marker.

Measure markers have no effect when a voice is played; they are used strictly for editing purposes. The use of measure markers is optional but recommended, especially in longer songs.

To enter measure numbers from the editing screen, press the function key f7. The Editor enters the next measure marker, using the number one greater than the previous marker. If the last measure entered was 3, pressing f7 enters

MS# 4. This allows you to enter measure markers without going to the special option screen and without typing numbers.

The usual procedure when entering a sequence of measures is to go to the special option screen to set the starting measure number and then to use f7 to enter successive measure markers.

The real advantage in using measure markers is that the Editor can search for a specific measure marker and move to that point in the voice. This is like moving to the beginning or end except that now you can quickly move to any location in the voice.

To search for a measure, press the f5 key when in the editing screen. The free memory display changes to a prompt asking for a measure number. Enter the number. The Editor searches the voice from the beginning until it finds the requested measure. If the measure is found, the Editor moves to that marker. If the Editor does not find the measure before it reaches the end of the voice, the Editor simply moves to that position.

If you hit the f5 key accidentally and don't notice the measure number prompt, the joystick and keyboard may seem to be locked up. To recover, just press the RETURN key to cancel the search.

Function Key Summary

At this point it might be good to review the uses of the function keys.

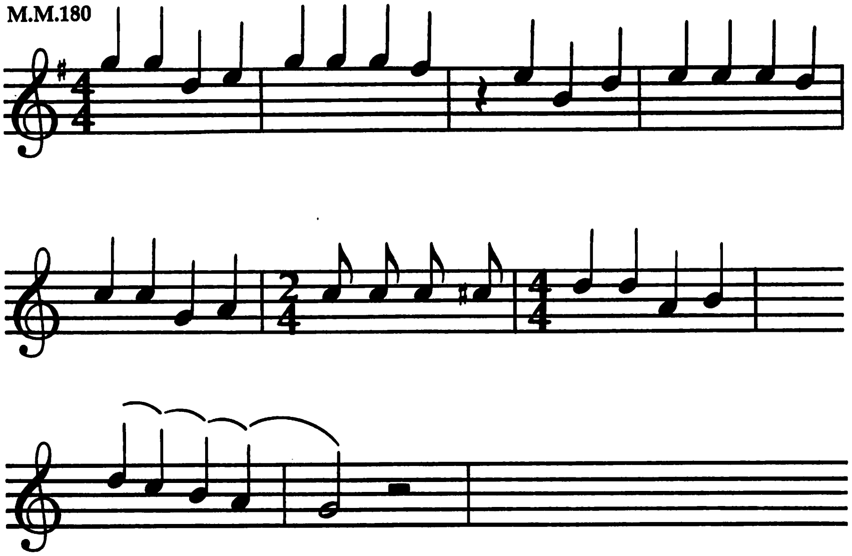
f1	If in editing screen, return to main menu. If in special options screen, return to editing screen.
f3	Go to special option screen.
f5	Search for measure.
f7	Enter next measure marker.
f2	Move to beginning of voice.
f4	Move to end of voice.

(Note: Figure 5-20, at the end of this section, is a cut-out that you can place over the function keys.)

Entering a Melody

Now that you know something about how the Editor works, how about trying to enter a short song? Just follow this step-by-step example, and you'll have one. Use the sheet music given in Figure 5-19.

Figure 5-19. Blues



Make sure the Editor is on your screen. Erase the entire voice by moving to the beginning (function key f2) and using SHIFT–CLR/HOME.

The sheet music indicates a tempo of M.M. 180. Go to the special option screen (function key f3) and move to the TEM command. Press the joystick button and type 180, but do not press RETURN. Since you're going to enter a second command, press SHIFT–RETURN instead. This prevents the display from returning to the editing screen when the command is entered.

No volume level is specified, so ignore setting the volume with a VOL command. When the voice is played, the default level of 8 will be used.

You're almost ready to enter the first measure, so move down to the MS# command (the fastest way to get there is to push the joystick up and wrap around to the bottom), press the button, and enter the number 1. You can press RETURN this time, because you won't be entering any more commands.

One last thing you need to do before entering notes is set the key. The song is written in the key of G, shown by the key signature displaying one sharp. Select the key of G by pushing SHIFT-+ until the key signature display indicates one sharp.

Select the quarter note duration and enter the first four notes. They are G5, G5, D5, and E5. You've just entered the first measure.

Press the function key f7 to enter the command MS# 2 (beginning the next measure). Type in the next four notes. You'll notice that because of the current key, the F-sharp is automatically selected for the last note instead of F-natural.

Measure 3 begins with a quarter rest. After you press f7 for the next measure marker, press the R key to turn on the rest mode. The rest will be entered when you press the button. The rest mode will still be on, though it's canceled when you move down to enter the note E5.

Measures 4 and 5 are pretty straightforward. Measure 6, however, is different. The notes are eighth notes, not quarter notes, so you'll have to change the duration. The last of these eighth notes has a sharp symbol by it, changing the pitch for that note from C-natural to C-sharp, so you'll have to press the + key to select the sharp.

You may also notice that this measure plays for only two beats instead of the usual four beats. The fraction 2/4 shows a change in time signature, an advanced topic which is explained later.

Measure 7 and the following measures are back in 4/4 time. Remember to change the duration back to a quarter note.

The notes in measure 8 are followed by ties, so press the / key to turn on the tie mode before you enter each note.

The last measure, the ninth, should be easy since it contains only one note. Once this note has been entered, the voice is ready to be played.

Error Checking

Sometimes you may make mistakes when entering a song. The Editor can find some, like the use of a sixteenth note in a tempo that does not support sixteenth notes, but it cannot find others, such as wrong pitches. Fortunately, the Editor has some features that make it easy to track down mistakes and correct them.

Autostop. If you're playing a voice and the Editor stops with the ILLEGAL DURATION error, the Editor points to the note which caused the error. This means that when you switch to the editing screen, the notes move so that the one with the illegal duration appears in the box.

Manual stop. Playing can also be stopped at any time by pressing the f7 key. When you then go to the editing screen, the note in the box will be the one playing at the moment you hit the key. This feature can be convenient when you're playing a song and you hear a "bad" note.

If you press any key other than f7 while a song is playing, the playing stops, but the Editor leaves the voice alone and doesn't move to the note being played.

Fast forward. Another feature is a fast-forward mode, which can be useful when you're entering a long song. Perhaps the beginning of the song plays fine, but later parts need work. When you choose PLAY from the main menu, playing always starts at the beginning of the song. There's no way to start somewhere in the middle.

With the fast-forward mode, you still have to start at the beginning, but you can make it play much faster. If you hold the joystick button down while playing a song, the song plays at three times the normal speed. Playing will return to normal as soon as you release the button.

Slow down. A slow mode is also available. If you let go of the button and push the stick in any direction while playing a song, it slows down to one-third normal speed. Use this feature when you want to listen carefully to a sequence of complicated notes.

Saving the Music

Select this item from the main menu when you want to save a music file to disk. It's a good practice to save your work frequently.

The Editor asks if you want to change the old text lines. The text lines are displayed in the window at the bottom of the main screen. If you've just entered a new song, there will be no text lines, so press the Y key to enter new ones.

The text lines are used to give the title of the song, to identify the composer, and to credit the person who entered the song. Up to four lines of text are allowed. The standard format for the text lines is one or two lines for the title, one line for the composer, and one line for the acknowledgment.

Enter the lines one at a time. Each can hold up to 32 characters. If you don't need all four lines, press RETURN to enter a blank line.

After the text lines have been entered, the Editor requests a filename. Enter a filename up to 12 characters long, then press RETURN. Do not include the .MUS extension as part of the filename. Be sure to choose a filename not already in use (you can check the disk by returning to the Editor menu and selecting option 5 for the directory).

The saving begins once the filename has been entered. When the SAVE is completed, the Editor clears the main menu and waits for your next instruction.

Disk drive errors such as FILE EXISTS and DISK FULL are not reported by the Editor. Always check the red drive light after a SAVE to make sure that it isn't flashing. The only error that the program does report is DEVICE NOT PRESENT, which can happen if the drive is turned off. If this error occurs, press any key to acknowledge it, turn the drive on, and try again. This time, however, you won't have to change the text lines.

To cancel a SAVE, press only RETURN in response to the filename prompt.

Loading a Song

This main menu item is used to load a music file from disk into memory. Type the filename of the desired file (excluding the .MUS extension), and press RETURN.

Note that loading a song erases the one in memory. If you want to preserve the song currently in memory, save it to disk before loading a new one.

If you're using disk and the requested file is not on the disk, the Editor will report the FILE NOT FOUND error.

After the song is loaded, the text lines in the window are updated. The Editor is ready for you to select another item.

If you accidentally choose LOAD, don't type a filename and just press RETURN. The song in memory will not be erased.

What's on the Disk?

This prints a disk directory on the screen. The names of all files with the .MUS extension on the disk are listed in two columns. Printing the directory is useful when you want to check whether a filename is already in use or if you want to see how many blocks are free.

Using All Three Voices

You've seen how to enter, edit, and debug one complete voice. But, as you know, Sidplayer can play up to three voices at the same time. To access the other two voices, press the 2 or 3 key after choosing EDIT from the main menu.

The voices are completely independent. Editing done on one voice does not affect the others. For instance, you can add some notes to voice 2 and completely erase voice 3, but when you return to voice 1, you'll find it unchanged.

When it comes to playing, however, the voices are not entirely independent. Multiple voices must share the same tempo and master volume. The TEM and VOL commands can be used on any voice, but they affect all three. If the tempo is set on voice 1, voices 2 and 3 use that same tempo. It's not possible for one voice to play at one tempo while another voice plays at another. The same idea applies to the master volume.

When you choose PLAY from the main menu and press RETURN, the Editor plays all three voices. If one of the voices runs out of notes before the others, that voice stops but the others continue.

It's possible to play individual voices. Instead of pressing RETURN after choosing PLAY, press the Y or N key for each voice. The Editor plays only those voices which were selected by pressing the Y key.

One problem with playing individual voices involves tempo. If the TEM command is used only on voice 1, and voice 1 is not played, the tempo will not be set for the other voices.

When you save or load a file, all three voices are saved or loaded, even if some of the voices are empty.

To clear all three voices at once, press the 2 key in the main menu to select EDIT, then press SHIFT–CLR/HOME.

On the *Complete 64* disk is an enhanced version of the blues rhythm given earlier. Voices 2 and 3 are used to add bass and percussion parts. You may wish to load this song into the Editor and examine the individual voices. Voice 1 should be identical to the voice in the earlier demonstration. Voices 2 and 3 use advanced commands explained in later chapters.

Keyboard Note Entry

The Editor has been designed to use the keyboard as well as a joystick for note entry. If you prefer to use the keyboard, or if you just don't have a joystick, follow these instructions.

- Press a letter key from A through G. The current note name will change to that letter.
- To change the octave, press a digit key from 0 through 7.
- The duration can be set by pressing one of the following keys:

W	Whole
H	Half
Q	Quarter
8	Eighth (E is used for a note name)
S	Sixteenth
T	Thirty-second

- Press the period (.) key if you want a dotted note. Press the key again to cancel the dot.
- Press the RETURN key to enter a note. This is like pressing the joystick button.
- In the special options screen, use the letters *I*, *J*, *K*, and *M* to move the box up, left, right, or down. Press RETURN to select a command.

Complete Joystick Editing

If you prefer to use only the joystick to edit, you can. Many of the things done with the keyboard can also be done with the joystick.

Accidentals, rests, and ties can be selected; the joystick can also be used to change the current key and to scroll the notes. But before any of these things can be done, you must first change the levels.

To move from the main level to a different level, press the joystick button and hold it down while you push the stick up or down. The different levels of the screen are highlighted in order. Release the button to stop on a particular level.

Once you're in a level and have released the button, push the joystick left or right to change whatever that particular level controls. For example, if you end up on the level which displays the current key, all the keys will be shown when you push the stick left or right. When you come to the key that you want, press the button to return to the main level.

On the accidental level, the flat, natural, and sharp symbols can be selected one at a time. Wraparound is supported, so if the current accidental is flat, pushing the stick left will switch to sharp. Again, press the button to return to the main level.

The level for selecting a rest or tie works a little differently. As soon as you push the joystick left (for rest) or right (for tie), that feature is selected and the Editor automatically returns to the main level. To turn off the rest or tie mode, move to this level and select rest or tie again.

Move to the bottom level and push the stick left or right to scroll the notes. Be aware that keyboard items, including the INST/DEL key, do not work when you're in this level. Press the button to return to the main level.

Leaving the Editor: QUIT

When you're done with the Editor, choose QUIT from the main menu. The prompt CONFIRM? appears. Press the Y key to return to BASIC, or press N to continue editing.

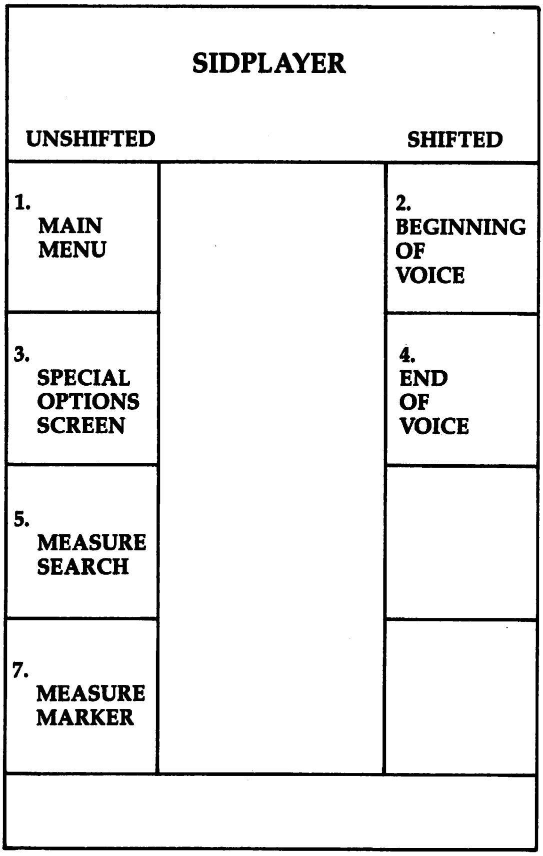
The RUN/STOP key has been disabled for your protection, so you cannot press this key to stop the program. Pressing RUN/STOP–RESTORE will not work either.

A final word of caution: Remember to save your work before you choose QUIT.

This concludes the description of the Sidplayer Editor. You should now be able to enter, edit, and debug a simple piece of sheet music. Music stores and libraries offer a large selection of classical and contemporary sheet music that will give you plenty of practice.

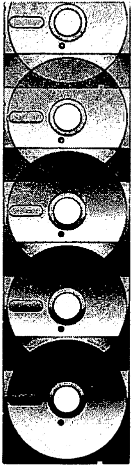
Once you feel comfortable with the Editor, you may wish to move on to the advanced features explained in the following chapters.

Figure 5-20. Function Key Cutout



SIDPLAYER

UNSHIFTED	SHIFTED
1. MAIN MENU	2. BEGINNING OF VOICE
3. SPECIAL OPTIONS SCREEN	4. END OF VOICE
5. MEASURE SEARCH	
7. MEASURE MARKER	



Making Music

Waveforms

The *timbre* of a sound is what distinguishes a middle C played on a saxophone from a middle C played on a cello. The main thing that controls a sound's timbre is the type of vibration which produces the sound. There are a few basic types of vibrations, or *waveforms.* These types are named according to their shape when viewed with an oscilloscope.

Triangle waves (Figure 5-21) produce soft, mellow tones. The flute is an example of an instrument which produces triangle waves.

Figure 5-21. Triangle

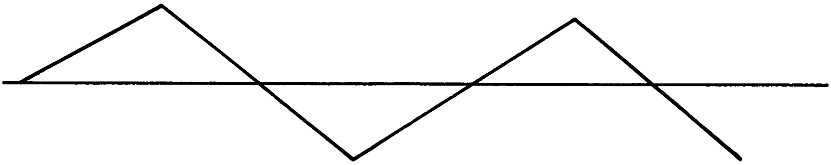


Figure 5-22 is a waveform for a bright, buzzy tone. Brass instruments produce waves that are basically *sawtooth* waves. (This is sometimes called a *ramp* waveform.)

Figure 5-22. Sawtooth

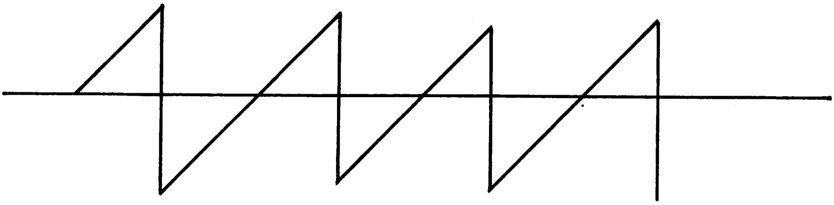
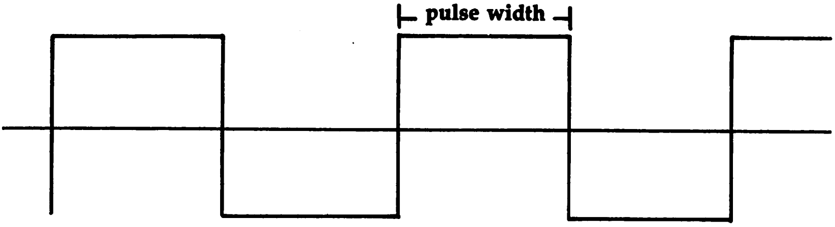


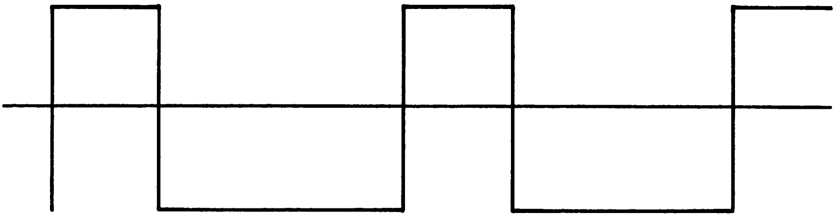
Figure 5-23 illustrates a *square* wave. This waveform sounds rich and hollow, and can be heard from the clarinet.

Figure 5-23. Square



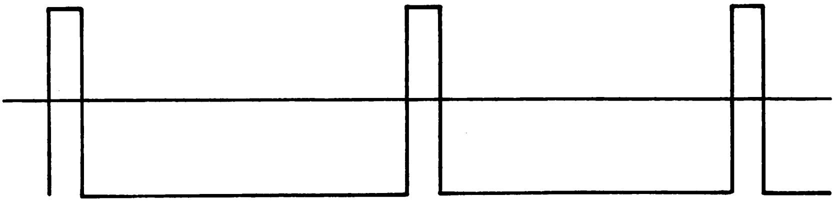
The *pulse* wave alternates between high and low states. The amount of time during one cycle that the wave is high is called the *width,* or *duty cycle,* of the wave. Square waves have a pulse width of 50 percent. When the width is reduced, the wave becomes more rectangular. The waveform in Figure 5-24 might be produced by an oboe or bassoon.

Figure 5-24. Rectangular



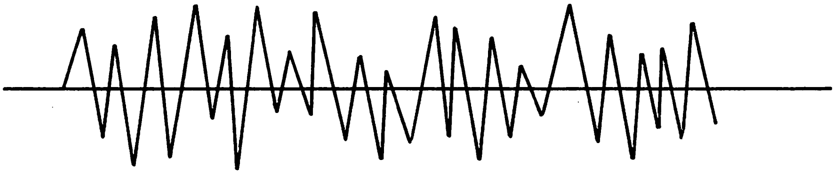
Pulse waves with a very narrow pulse width sound thin and reedy. Pulse waves with widths greater than 50 percent sound just like pulse waves with widths less than 50 percent. For example, a pulse wave with a 40 percent width sounds the same as a pulse wave with a 60 percent pulse width.

Figure 5-25. More Rectangular



Another type of waveform has no definite shape because it's completely random. It's called *noise* (Figure 5-26) because it is the waveform associated with white noise, the sound you hear when a television station goes off the air. Noise is useful for producing percussion effects, such as snare drums.

Figure 5-26. Noise



Another useful waveform, the sine wave, unfortunately is not supported by the SID chip.

The waveforms introduced here are just the basic types. Actual instruments produce more complicated waveforms that may not conform exactly to one of these types.

Setting the waveform. The special option screen in the Editor has a row of commands which pertain to waveforms. Three of these commands are introduced in this chapter. The first command, labeled WAV, is used to select the waveform for a voice.

When you press the joystick button to choose the WAV command, the window in the upper-right corner displays some extra information. The desired waveform has to be specified in the form of a number, so the window shows you the number for each waveform.

0	**N**	**Noise**
1	**T**	**Triangle**
2	**S**	**Sawtooth**
4	**P**	**Pulse**

After you type the number and press RETURN, the command which appears in the box displays the letter instead of the number.

Each of the three voices can have its own waveform. Try changing waveforms. Load BLUES from the Editor and play voice 1 with various waveforms.

Pulse width. If you choose the pulse waveform, you may also want to use the P-W command to set the pulse width.

The numeric range for this command is from 0 through 4095, with 2048 being a perfect square wave. Values smaller or larger than 2048 produce rectangular waves.

As you approach the limit of 0 or 4095, the waves become so narrow that the volume begins to decrease. The pulse wave is inaudible when the pulse width is set to 0 or 4095. In most cases, only width values from 100 through 4000 are used.

Try playing a voice with different numbers for the P-W command to hear the effects of the various widths. Just as each voice has its own waveform, each voice also has its own pulse width. Changing the pulse width for one voice will not affect the others.

Pulse width sweeping. An advanced feature of Sidplayer is that it can change the pulse width during a note. When this feature is turned on, the pulse width starts at the specified value, but increases or decreases for the duration of the note. The width is then reset back to the specified value at the beginning of the next note.

The effect of pulse width sweeping is to add a sense of motion to the sound. A common way to use sweeping is to set the pulse width at 2048 and have it increase slowly so that the sweeping is barely noticeable. Another technique is to set the pulse width at 1000 and have it increase rather quickly. You'll hear the sound go from a rectangular wave to a square wave as the width reaches 2048, and then back to a rectangular wave as the width continues to increase.

The P-S command controls pulse width sweeping. Values from 1 through 127 turn on the sweeping. The larger the value, the faster the width increases. Values from −1 through −127 do the same thing, except they cause the width to decrease. Try 10 for starters. Using the number 0 with the P-S command turns off the sweeping.

If the pulse width is allowed to sweep past 4095, it wraps around to 0. The same thing happens if the width is decreasing and sweeps past 0. When the width wraps around, the effect on the tone is quite noticeable and usually isn't desirable. In most cases the wraparound can be avoided by changing the values for the pulse width, the sweep rate, or the direction.

One interesting use of pulse width wraparound is to set the sweep rate at 127, the maximum value. This produces a raspy tone.

If one note is tied to another note, the pulse width is not reset when the second note starts playing. The sweeping continues with no break.

When you turn sweeping off with the command P-S 0, the pulse width won't be reset to the specified value at the beginning of the next note. If you want the pulse width to be reset, use the P-W command to set the width again.

Here are the default waveform settings.

WAV	**P**
P-W	**2048**
P-S	**0**

This gives you a square wave with no sweeping. That's the type of waveform used if none of the waveform commands is placed at the beginning of a voice.

Waveform demonstrations. On the *Complete 64* disk are two waveform demonstration pieces. The first, FSONATINA, uses triangle, sawtooth, and pulse wave settings. The second, GSONATINA, uses only the pulse wave, but with different pulse widths, plus the use of sweeping.

Envelopes

Dynamics describe the general volume of a song, as set by the master volume, but do not describe the changes in volume which occur while an individual note is playing. These changes in volume over the course of a note are referred to as the *envelope* of the note.

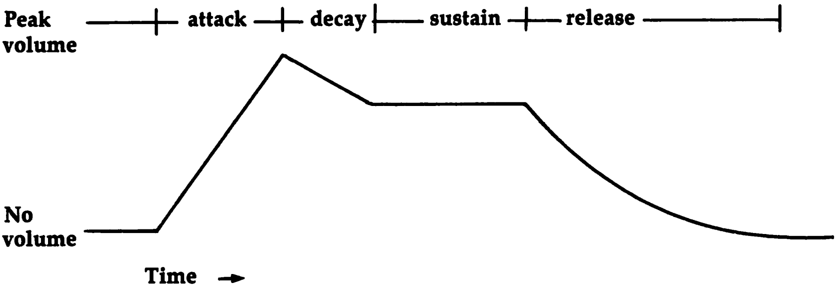
When a note first starts playing, the volume must increase from no volume to the peak volume established by the dynamics. The speed at which the volume rises is called the *attack* rate.

The volume then diminishes slightly until it reaches an intermediate volume level called the *sustain* level. The rate at which the volume falls from the peak level to the sustain level is called the *decay* rate.

Toward the end of a note, the note will be released, and the volume will begin to fade away, at a speed called the *release* rate.

A good way to understand the four stages of the envelope is to graph them (Figure 5-27).

Figure 5-27. ADSR Envelope



This standard envelope is sometimes also called an *ADSR envelope,* because of the attack, decay, sustain, and release sequence.

Not all instruments have the same ADSR values. Each instrument has its own characteristic envelope. In electronic music, the ability to control the envelope of a voice lets you more closely approximate a particular instrument.

There are two basic types of envelopes. One type is used for sustaining instruments. These include string instruments which are played with a bow and wind instruments.

The other type is for nonsustaining instruments. String instruments which are plucked and percussion instruments such as drums are examples of nonsustaining instruments.

Sustaining envelopes. For an example of a sustaining envelope, let's consider a person playing a violin. At the beginning of a note, the player has to dig in with the bow to start the string vibrating. This is the attack.

Once the sound has started, the player does not have to apply quite so much pressure to the string, and the volume is reduced a little. This is the decay part of the envelope.

As the player continues to bow, a constant volume level is maintained. This is the sustain level.

At the end of the note, the player stops the bow, but the string continues to vibrate for a moment until the sound fades away completely. The instant when the player stops the bow is called the release, and the rate at which the volume fades away is the release rate.

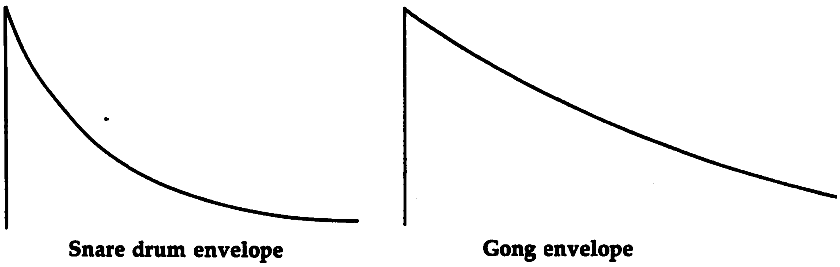
The whole process works similarly for a wind instrument, such as a flute. The player has to blow with a little extra force to start the air vibrating and then eases off slightly. The air continues to vibrate for a moment after the release, when the player stops blowing.

Nonsustaining envelopes. Nonsustaining instruments have completely different envelopes. The most important characteristic of these instruments is that they are struck. The instrument is hit or plucked once for each note. No continual force is applied, so the volume is never sustained.

Most percussion instruments have nonsustaining envelopes. These instruments include drums, bells, and others like wood blocks. They usually have very fast attack rates. The decay rate varies from one instrument to another. The sound of a snare drum decays rather quickly, but the sound of a gong takes a long time to decay. Since there is no sustain, there is no sustain part to the envelope, and the envelope can be called an *ADR envelope.*

The two graphs in Figure 5-28 illustrate nonsustaining envelopes.

Figure 5-28. ADR Envelope



Setting the envelope. The special option screen has five commands that are used to configure the envelope.

The **ATK** command lets you choose one of 16 attack rates. The rates are numbered 0–15. At rate 0, the attack takes just a fraction of a second (two milliseconds to be precise), while at rate 15, the attack takes eight seconds.

The **DCY** command is used to set the decay rate. Again, the rates are numbered 0–15, with 0 being the fastest, but the range is a little different. The decay takes 6 milliseconds at rate 0, and 24 seconds at rate 15.

Use the **SUS** command to set the sustain level. If the sustain level is set to 0, the volume decays to no volume. If the level is set at 15, the volume doesn't decay at all and stays at the peak volume set by the master volume control. Values between 0 and 15 correspond to evenly spaced volume levels between 0 and peak volume. When you're using a nonsustaining envelope, the sustain level should be set to 0.

The **RLS** command sets the release rate. The rates, numbered 0–15, are the same as those for the DCY command. If you're using a nonsustaining envelope, the release rate should be set to the same value as the decay rate.

The release point. The remaining command is **PNT,** which is used to set the release point. In order to understand how to use this command, it's first necessary to know how Sidplayer handles durations.

Sidplayer deals with note durations in terms of time units called *jiffies.* One jiffy lasts about 1/60 second. Table 5-3 gives the jiffy length of each note duration in each tempo.

The number of jiffies for a whole note is different for each tempo. This number is repeatedly cut in half to give durations like half note, quarter note, and so on. After a certain point, some numbers cannot be evenly divided by two, which is why some tempos do not support sixteenth or thirty-second notes.

The PNT command determines how many jiffies from the end of a note the volume should be released. For an example, let's say that the release point is set at 6 and the current tempo is M.M. 100. A quarter note in this tempo is 36 jiffies long. For the first 30 jiffies, the volume goes through the attack, decay, and sustain phases of the envelope. Then the voice will be released, and for the last 6 jiffies, the volume falls from the sustain level to no volume.

The range of the PNT command is 0–255, but usually only very small values are used. Values like 3, 4, and 5 work best. It's preferable not to set the release point too high, or notes of short duration won't be heard. For instance, in M.M. 100, an eighth note is 18 jiffies long. If the release point was set at 20 jiffies, an eighth note would be released as soon as it started playing. The volume would never have a chance to rise, and the note would not be heard.

For best results, always make sure that the release point is less than the duration of the shortest note in the song.

If you're using a nonsustaining envelope, the PNT command must be used to set the release point at 1.

Table 5-3. Tempo

M.M.	W	H	Q	E	S	32	64
900	16	8	4	2	1	–	–
600	24	12	6	3	–	–	–
450	32	16	8	4	2	1	–
360	40	20	10	5	–	–	–
300	48	24	12	6	3	–	–
257	56	28	14	7	–	–	–
225	64	32	16	8	4	2	1
200	72	36	18	9	–	–	–
180	80	40	20	10	5	–	–
163	88	44	22	11	–	–	–
150	96	48	24	12	6	3	–
138	104	52	26	13	–	–	–
128	112	56	28	14	7	–	–
120	120	60	30	15	–	–	–
112	128	64	32	16	8	4	2
105	136	68	34	17	–	–	–
100	144	72	36	18	9	–	–
94	152	76	38	19	–	–	–
90	160	80	40	20	10	5	–
85	168	84	42	21	–	–	–
81	176	88	44	22	11	–	–
78	184	92	46	23	–	–	–
75	192	96	48	24	12	6	3
72	200	100	50	25	–	–	–
69	208	104	52	26	13	–	–
66	216	108	54	27	–	–	–
64	224	112	56	28	14	7	–
62	232	116	58	29	–	–	–
60	240	120	60	30	15	–	–
58	248	124	62	31	–	–	–
56	256	128	64	32	16	8	4

Here are the default values for the envelope. This configuration produces an organ effect.

ATK	**2**
DCY	**0**
SUS	**15**
RLS	**5**
PNT	**4**

If some default settings are satisfactory, but others are not, you have to change only the ones that need new values.

Also, remember that you can change the envelope at any point in a song. For example, a voice may briefly switch to a nonsustaining envelope to play a few notes and then switch back.

Each voice can have its own envelope, but it's recommended that the attack rate be nearly the same for all three voices.

Experiment with different envelopes. Load BLUES from the *Complete 64* disk and try different envelope settings on the first voice.

Sometimes in sheet music you may see what appears to be a very long tie symbol spread over several notes. This symbol indicates that the notes are to be played smoothly, in one breath or one bow movement. The term *legato* describes this type of effect.

A sequence of notes can be played in a legato style by using the command PNT 0. When the release point is set to 0, notes are never released. This saves you from having to put a tie on each note.

Another style of playing is *staccato,* which is the opposite of legato. A dot placed above or below a note in sheet music means that the note should be played in a quick, light, choppy manner.

One way to produce this effect on Sidplayer is to set the release point as large as possible. A more reliable method is to switch to a nonsustaining envelope.

A nonsustaining envelope is also a good choice when you want to play a series of short, fast notes. The release point usually has to be set very small to play these notes, and if a sustaining envelope is used, the notes may sound too legato. Using a nonsustaining envelope insures that the notes will be played distinctly and will not be run together.

Envelope Example

The file T&V on the *Complete 64* disk is the song "Theme and Variation" by Beethoven. It uses a variety of envelopes. Load T&V into the Editor, select Play Music, and press RETURN to hear all three voices.

The Filters

Sometimes it's not enough to set the waveform and envelope of a voice in order to imitate the sound of a particular instrument. It may also be necessary to control the harmonic content of the voice by using the filter.

When an oscillator generates a tone, it produces not only the requested pitch, but some harmonics as well. *Harmonics* are frequencies related to the main pitch. The first harmonic is the frequency of the main pitch and is also called the *fundamental frequency.* The second harmonic has a frequency twice the fundamental frequency. The frequency of the third harmonic is three times that of the fundamental frequency, and so on.

Because the volume of the fundamental frequency is always greater than the volume of the other harmonics, the main pitch detected by your ear is that of the fundamental frequency. The harmonics, however, give the tone its *timbre.*

Since each instrument has its own characteristic timbre, the ability to control the harmonic content of a voice can be helpful in emulating a particular instrument. The filter is used to remove selected frequencies from a tone. This enables you to emulate a whole new variety of instruments.

The SID chip's filter has three main control parameters that must be set before the filter can be used. These parameters are the *mode,* the *cutoff frequency,* and the *resonance.*

Filter mode. The filter mode determines which types of frequencies are removed from a tone. The most commonly used mode is the low-pass mode. This mode allows only the frequencies below a certain frequency, called the cutoff frequency, to pass through the filter. Any frequencies above the cutoff are *attenuated* (greatly reduced in volume) so that they are seemingly removed from the tone.

The low-pass filter mode (Figure 5-29) produces full-bodied tones. The opposite of the low-pass mode is the high-pass mode (Figure 5-30), in which frequencies below the cutoff are suppressed, while frequencies above the cutoff are passed through unaltered.

The high-pass mode causes tones to sound tinny or buzzy.

Figure 5-29. Low-Pass Filters

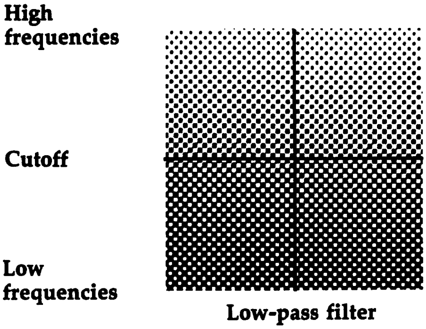
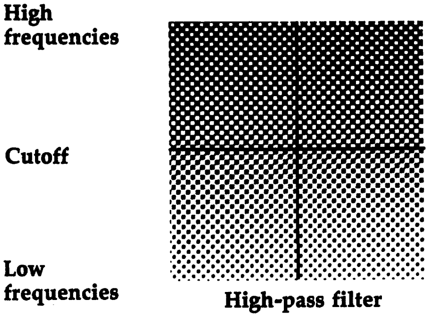
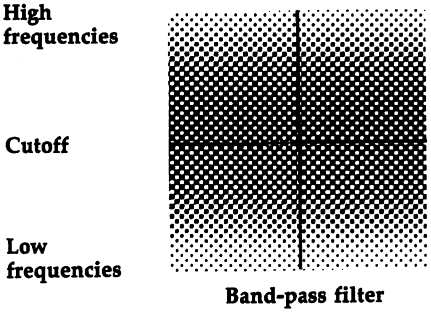


Figure 5-30. High-Pass Filters



One other filter mode is the band-pass mode (Figure 5-31). In this mode, only the frequencies near the cutoff are passed through the filter. All other frequencies are attenuated. The band-pass mode produces thin, open tones.

Figure 5-31. Band-Pass Filters



Use the F-M command to set the filter mode. This command works similarly to the WAV command. When you press the joystick button for F-M, the window displays a number for each filter mode.

1	**L**	**Low pass**
2	**B**	**Band pass**
4	**H**	**High pass**

Type the number of the desired mode and press RETURN to enter the command. The letter indicating the filter mode appears as part of the command.

You can even select combinations of filter modes. Just add the appropriate numbers together. For example, choosing 5 sets the filter for both low- and high-pass modes. This combination is also called the *band reject,* or *notch,* mode.

Filter cutoff. The cutoff frequency of the filter acts as a dividing line. In low-pass mode, for example, frequencies above the cutoff are attenuated, while frequencies below the cutoff are passed through unaltered. But consider what happens when the pitches of notes in a song fall right around the cutoff frequency. Notes having pitches below the cutoff are played correctly, but notes with pitches above the cutoff are not heard.

What you need is a way to set the cutoff higher or lower. The ideal setting is to place the cutoff right above the highest note to be played. That way, all of the fundamental frequencies will be below the cutoff, so all of the notes will be heard. Most of the harmonics will still be above the cutoff, so they'll be removed from the tone.

The F-C command is used to set the filter cutoff. The range of this command is 0–255, with 0 the lowest and 255 the highest. The standard practice is to start with a value between 0 and 255, play the voice, and then move the cutoff up or down as necessary.

Resonance. The effect of resonance is to produce a sharper tone by emphasizing, or *peaking,* the frequencies in the tone that are close to the cutoff. Actually, the resonance control of the SID chip is not very effective and acts mainly as a way to control the volume of the voice being passed through the filter.

Use the RES command to set the resonance level. The number for this command ranges from 0 (no resonance) through 15 (maximum resonance). In most cases the value 15 is best, but sometimes you may want to use a smaller value to produce a muted effect.

Passing a voice through the filter. Once you've configured the filter by using the F-M, F-C, and RES commands, use the FLT command to indicate that the voice should be passed through the filter. This command works a little differently from the others. Unlike the previous commands which support a number range, the FLT command gives you a simple choice of either yes (1) or no (0). Enter 1 to indicate that you want the voice to be passed through the filter.

Here's an example showing how you might use the filter. Start with the following commands:

```
F-M  L
F-C  255
RES  15
FLT  Y
```

Next, enter a few notes. A simple scale is sufficient. When you play the voice, it should play normally. In low-pass mode with the filter cutoff set at the maximum value, all frequencies are passed through the filter.

Now, lower the cutoff by reducing the number for the F-C command, and play the voice again. Do this a few times, each time reducing the F-C command number by about 20 or so. The effects of filtering should gradually become more noticeable. To stop the filtering, enter the FLT command with the number 0 to indicate that you don't want the voice to be passed through the filter. You don't have to change the mode, cutoff, or resonance setting.

```
FLT  N
```

If you want the voice to be passed through the filter later in the song, enter the FLT command with the number 1. The earlier filter settings will still be in effect, but can be changed if necessary.

Autofilter. Trying to find a proper setting for the filter cutoff can often be inconvenient and time-consuming. As an alternative to the F-C command, Sidplayer offers a special feature called the *autofilter mode.* When the autofilter mode is turned on, the filter cutoff is automatically set according to the pitch of each note. Since the cutoff is calculated for each note, all notes, high and low, produce the same filtering effect.

To turn on the autofilter mode, select the command AUT and enter the number 1. Do this instead of entering the F-C command.

```
F-M  B
AUT  1
RES  15
FLT  Y
```

You may also specify that an offset should be added to the cutoff. When you enter the number for the AUT command, choose any number from 1 through 127 or from −1 through −127. Different offset values give different filtering effects. The autofilter mode is turned off by the command AUT 0.

When using the autofilter feature, it's important that you turn it off whenever you stop passing the voice through the filter.

```
FLT  N
AUT  0
```

Filter sweep. Just as pulse width sweeping changes the pulse width during a note, filter sweeping lets you increase or decrease the cutoff frequency while a note is playing.

Use the F-S command to turn on filter sweeping. The number for this command controls the sweep rate and direction, and ranges from −127 through 127. Values of 1 through 127 make the cutoff sweep upward. The larger the number, the faster the sweep. Values of −1 through −127 sweep downward. Zero turns off the sweeping.

In most cases, the best results are obtained by using small numbers for the sweep rate, such as values 1–10. If the cutoff is swept too far, it will wrap around, but this is not as useful as it is with pulse width sweeping.

Be sure to turn the filter sweeping off when you stop passing the voice through the filter.

```
FLT  N
F-S  0
```

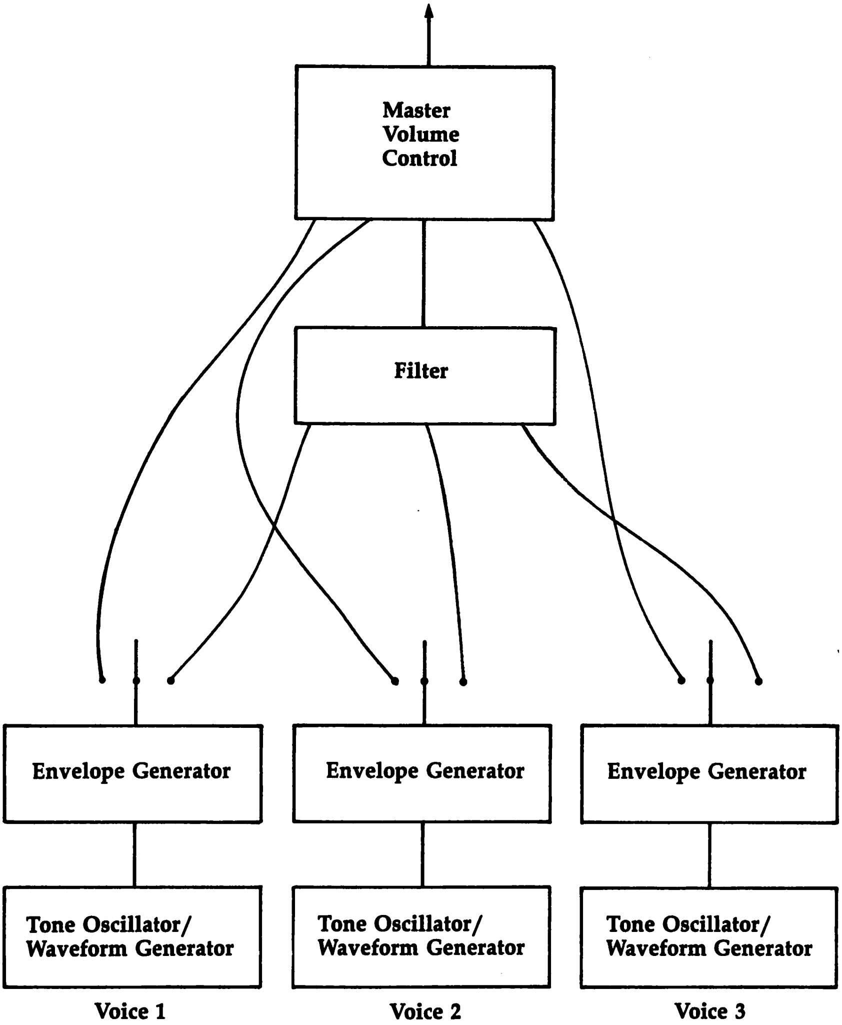
Or if using the autofilter mode:

```
FLT  N
AUT  0
F-S  0
```

Restrictions. While the SID chip has three oscillators, and can therefore play three voices, it has only one filter. There's not a separate filter for each voice. Figure 5-32 should help you understand the arrangement. The illustration is the same as Figure 5-1 except that it has been modified to include the filter.

As shown by Figure 5-32, any voice can be passed through the filter, but only one voice should be routed through the filter at one time. Passing two or three voices through the filter at the same time can overpower it, causing it to produce an unpleasant buzzy noise.

Figure 5-32. Filtering Sound


Master Volume Control
Filter
Envelope Generator
Envelope Generator
Envelope Generator
Tone Oscillator/ Waveform Generator
Tone Oscillator/ Waveform Generator
Tone Oscillator/ Waveform Generator
Voice 1
Voice 2
Voice 3

The major drawback to using the filter is that there is extreme variance in filtering effects from one computer to another. The same filter settings can produce completely different results on different Commodore 64s. On some computers, especially the older ones, a voice can be almost inaudible when it is passed through the filter, no matter what settings are in effect. The variance lies not only in the SID chip, but in the support circuitry as well, so there's no easy way to fix the problem. If you have one of the earliest versions of the Commodore 64, you may not be able to use the filter.

Three Demonstrations

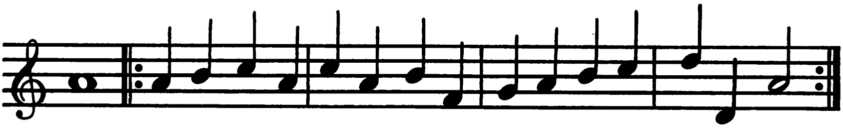
Three demonstration songs have been provided on the *Complete 64* disk. BRASS is a theme played by a trumpet, a trombone, then a tuba. The other two demonstrations emulate a koto and a sitar.

Repetition

As you listen to a song, you can sometimes hear a group of measures played more than once. Repetition occurs often in music, especially in contemporary songs.

The simplest form of repetition is when a group of measures is repeated immediately after being played. So that the repeated measures do not have to be written twice, special symbols are used in sheet music to indicate when a sequence of measures should be repeated. The symbol that marks the beginning of a repeat consists of two vertical lines followed by two dots (Figure 5-33). The end of a repeat is marked by a symbol that looks almost the same, except the dots come before the double lines instead of after them.

Figure 5-33. Repeats



When the music is played and the first repeat symbol is reached, the playing continues as usual. Upon reaching the end of the repeat, however, playing does not continue to the next measure. Instead, playing jumps back to the measure which had the beginning repeat symbol and continues from that point. When playing comes to the end repeat mark the second time around, it's ignored and playing continues with the next measure.

Normally, a sequence of notes is repeated only once. There are, however, cases where a voice may repeat several times, such as in a bass line.

Sidplayer has two commands to support repeats. The beginning of a repeat should be marked by the command HED, which stands for *repeat head.* The number for this command determines how many times the sequence of notes is to be played. In a standard repeat, the notes are repeated once, meaning that the sequence is played a total of two times, so you would normally type the number 2. Then you would press RETURN to enter the command and continue entering notes.

At the end of the repeat, enter the TAL command *(repeat tail).* There's no data value for this command, so just enter the number 0.

When the voice is played, the sequence of notes between the HED and TAL commands plays the designated number of times.

The number entered for the HED command must fall in the range 0–255. The number 1 means that the sequence should be played only once, which would seem to make the repeat structure unnecessary. A value of 1 is used only when you're developing a piece of music. If you keep replaying a song to listen for bad notes at the end, you don't want to wait for repeats earlier in the song, so use 1 at first (just remember to change it to the correct value before you save the final version of the song).

Using 0 with HED makes the notes repeat forever. This feature may be useful in some game applications, but should not be used in regular music, because the song will never end.

Repeats cannot be nested. Whenever you have a HED command, it has to be later followed by a TAL command before another HED can be used. It's okay for a voice to contain several repeats; you just cannot have a repeat inside a repeat. Each voice can have its own repeat, however, so each voice can repeat independently of the others.

When repeats are used properly, there should be one repeat end for every repeat beginning. If a voice is playing and a TAL command is encountered with no previous HED command, playing will repeat forever back to the most recent HED command. If no HED command has been used at all on the current voice, playing will repeat forever back to the beginning of the voice.

Phrases. Occasionally, you'll find that a repeat has a first and a second ending. This means that one set of notes should be played at the end of the sequence the first time through, and a different set should be played the second time. The simple repeat structure of the HED and TAL commands cannot handle this—in this case, you have to use phrases.

If you consider that a repeat is a loop, then a phrase is like a subroutine. A phrase allows the same sequence of notes to be played at different places in the music. The first time the phrase is played, the beginning and end are remembered by Sidplayer. This is called *defining the phrase.* Later in the music, when the notes have to be played again, playing can be made to jump back to the beginning of the phrase by the use of a single command. This is known as *calling the phrase.* When the end of the phrase is reached, playing continues with the rest of the song.

It's important to understand that there are some differences between Sidplayer phrases and BASIC subroutines. In BASIC, a subroutine is usually put at the end of a program. Every time the subroutine has to be executed, it's called by the GOSUB statement. With Sidplayer, the notes and commands that form the phrase are placed in the song at the first instance where the phrase must be played. After the phrase has played once, and is thereby defined, it can be played again by use of the phrase call.

To define a phrase, first enter the DEF command. This command needs a number in the range 0–15. For now, just enter the command with 0, then enter the notes which form the phrase.

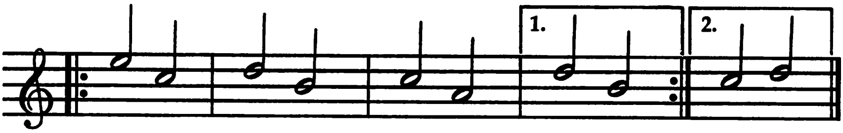
After the last note in the phrase, enter the command END. Like the TAL command, the END command has no data value, so it should be entered by typing the 0 and pressing RETURN.

When playing reaches the END command, the notes have played once, the definition is complete, and the phrase is ready for calling.

To call the phrase, all you have to do is enter the CAL command with the number 0. This one command takes the place of all the notes in the phrase.

Figure 5-34 demonstrates how phrase calling works.

Figure 5-34. Phrases



Begin by entering the command DEF 0, followed by the notes up to but not including the first ending. Now enter the END command. The notes for the first ending come next. At this point, playing is supposed to repeat back to the beginning. Instead of entering all those notes again, just enter the command CAL 0. Finally, enter the notes for the second ending.

When the voice plays, the phrase plays the first time. The DEF and END commands have no effect. The notes for the first ending then play. The command CAL 0 sends playing back to the first note after the command DEF 0 to play the phrase again. When the END command is reached, playing continues with the first note after the CAL 0 command.

A phrase can be called more than once. A phrase is still defined after it has been called, so it can be called as many times as necessary.

Phrases have many uses besides handling repeats with different endings. In fact, phrases are so useful that quite often one is not enough. Don't worry—Sidplayer can remember up to 16 independent phrases. When you enter the DEF command, the number from 0 through 15 identifies which phrase is being defined. That particular phrase can be called later by using the same phrase number with the CAL command.

If, while a song is playing, a CAL command tries to call a phrase that has not been defined earlier in the music, the Editor stops with the UNDEFINED PHRASE CALL error.

The 16 phrases are shared among the three voices. For example, even though phrase 7 may be defined on voice 1, it can also be called on voice 2 or 3. This means that playing can temporarily jump to another voice.

This can cause a problem when you are playing individual voices. For example, if you play only voices 2 and 3, and voice 2 contains a call defined in voice 1, playing will stop with the UNDEFINED PHRASE CALL error.

It's even possible for one phrase to call another. Phrase calls can be nested to a limit of four levels on each voice. If you try to exceed the limit, the Editor stops and reports the STACK OVERFLOW error. You can also define one phrase inside another. Be aware that a phrase definition counts as one nesting level. The only thing a phrase definition cannot do is call itself. If the definition of phrase 3 directly or indirectly contains a call to phrase 3, an infinite loop results. Playing will eventually stop with a STACK OVERFLOW error.

Phrases can be redefined. If a phrase is no longer needed, the phrase number can be used in the DEF command of a new phrase. This lets you use more than 16 phrases during the course of a song.

The STACK UNDERFLOW error occurs if DEF and END commands are not properly matched and playing comes across an END command with no previous DEF command.

Da capo, dal segno, and coda. Repeats are not the only kind of repetition. Other forms include *da capo* and *dal segno.* Da capo is indicated in sheet music by the letters *D.C.;* it means that the playing should jump back to the beginning of the voice and continue from there, this time ignoring all repeats. The playing may be stopped before the end of the song by the use of the word *fine.*

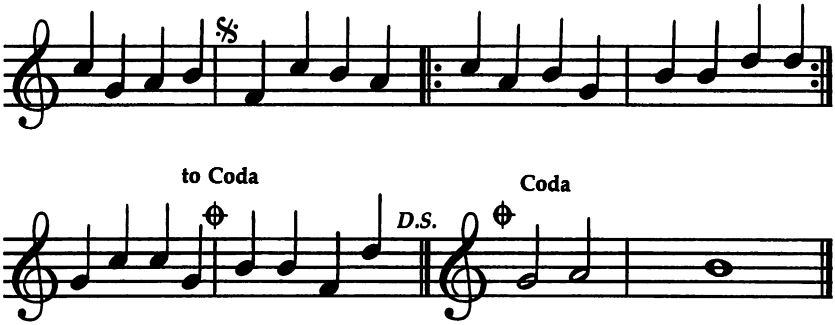
Dal segno, identified by the letters *D.S.,* means that playing should jump to the measure marked by a special sign. This sign looks like a slash with dots on either side, passing through a fancy letter *S.* Playing continues from this point and stops either at the end of the song or at a *fine,* whichever comes first.

There's one other symbol often encountered when da capo or dal segno has been used. After playing has jumped back to the beginning of the voice or to a particular measure, you may encounter the message *To Coda,* followed by a coda symbol. The coda symbol looks like a letter *O* with a cross passing through it. This means that playing is going to jump to another place again, but this time, instead of jumping back, the playing skips ahead. At the end of the sheet music you should find some measures labeled *Coda,* with the coda symbol shown again. Playing jumps to the first of these measures and continues to the end of the song.

Through the clever use of phrase calling, Sidplayer can handle these advanced forms of repetition. The example in Figure 5-35, though condensed and not necessarily typical, uses repeats, dal segno, and a coda. Below is the order in which the measures would be played.

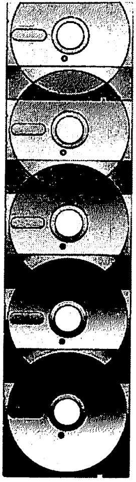
1. The first two measures are played (the sign is ignored).
2. The next two measures are played and then repeated.
3. The following two measures are played (the coda symbol is ignored).
4. At the D.S., playing goes back to the measure which had the sign (the second measure) and continues from there.
5. This time, the measures in the repeat are played only once.
6. When playing reaches the coda symbol, playing jumps to the coda (the last two measures) and then ends.

Figure 5-35. Repeats, D.S., and Coda



Phrases are also handy if you frequently need to alternate between two or more voice settings. A phrase can consist of only commands and no notes. Just be careful not to make a voice too complex. Too many commands, whether in a phrase or not, can cause the CLOBBER error in the fast-forward play mode. The CLOBBER error means that processing on one voice took too much time.

One practice to avoid is defining several phrases at the beginning of a song before any notes are played. This increases the risk of a CLOBBER error. Remember, you should not define a phrase until the first time it's needed and then call it for all later times.



Sophisticated Sounds

Music can be defined in many ways, but one way is to say that it consists of change—changing pitches, changing durations, and changing volumes. The first part of this chapter introduced these characteristics of music on a simple level.

This article reexamines these same characteristics, but from a broader viewpoint. For example, the volume level may not stay the same throughout a song. It might change at different places in the music. Or the tempo may change. Even the key may change during a song. Changes like these may not happen in a simple piece of music, but they certainly do occur in longer, more sophisticated works.

Now that you have some experience using the Sidplayer, it's time to look at these more advanced aspects of music. The purpose of this chapter is to cover all the remaining elements of notation commonly found in sheet music. When you finish with these descriptions and explanations, you'll be able to enter almost any piece of music with the Sidplayer and Editor.

Tempo changes. There are two kinds of tempo changes. In the first kind, the tempo changes abruptly, perhaps from a slow to a fast speed. This is most often found at the beginning of a movement or major part of a piece of music.

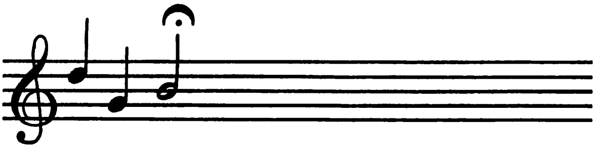
In the second kind of tempo change, the tempo increases or decreases gradually. This type of change is marked by the words *accelerando* and *ritardando,* which respectively mean that the tempo should start getting faster or slower. For instance, if the tempo is currently M.M. 100, and you see *ritard,* you might change the tempo to M.M. 94, then change it to M.M. 90 a few notes later.

When making such tempo changes, be sure that the voices stay synchronized. Make tempo changes only at a point in the music where all three voices are starting a new note. Usually, the beginning of a measure is a suitable place to change the tempo.

(Incidentally, if the tempo is ever indicated by a half note followed by an equal sign instead of a quarter note and equal sign, double the tempo value when entering the TEM command.)

The symbol used to change the tempo briefly is called a *fermata;* it looks like a narrow semicircle with a dot below it. A fermata can be placed over a particular note or rest to extend its duration. Glance at Figure 5-36 for a sample. This is sometimes referred to as a hold.

Figure 5-36. Fermata

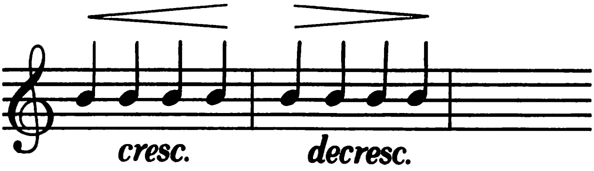


The easiest way to handle a fermata is to enter the corresponding note with a longer duration than written.

Dynamics. Volume changes are the most common type of change in music. As with tempo, the dynamics can change immediately or gradually. Immediate changes are indicated in the normal way, using letter combinations such as *p* and *fff.* Gradual changes are marked by the words *crescendo* and *decrescendo,* which respectively mean that the volume should gradually increase or decrease.

Gradual volume changes can also be denoted by symbols placed above notes, as in Figure 5-37.

Figure 5-37. Crescendo and Decrescendo



Use the VOL command for abrupt volume changes, when the master volume is to be changed by more than one level. For gradual changes, however, you may prefer to use the BMP command. This command is used to bump the master volume up or down one level.

When you press the joystick button while on the BMP command in the special option screen, the window indicates that 0 should be entered to bump the volume up one level, or that 1 should be entered to bump the volume down one level. After you press the RETURN key, the command is displayed with the name *BMP* and the letters *UP* or *DWN.*

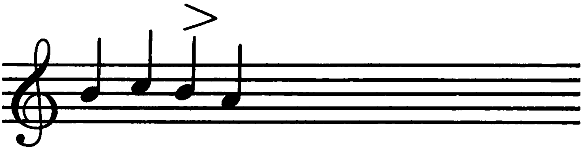
To produce the effect of a crescendo or decrescendo, bump the volume up or down every few measures. The BMP command is also useful in repeat loops at the end of a song, where the volume has to fade out.

A BMP UP command does not change the master volume if the volume is at 15, the highest level. Likewise, a BMP DWN command has no effect if the volume is already 0.

Remember that the VOL command should be used only on one voice. This is also true of the BMP command.

An *accent mark* is the symbol used to change the volume of one note. It looks like a greater-than sign (Figure 5-38). When placed above a note, it means that the note should be played just a little louder than the others.

Figure 5-38. Accent Mark



Since there is no volume control for individual voices, accents are hard to simulate. You might set the sustain level a little higher or sustain the note a little longer by using a smaller release point. Insert these changes before the note to be accented. After the note, reset the sustain level or release point to the previous values.

Key changes. Usually, the clef symbols are drawn at the beginning of each staff. They're followed by sharp or flat signs that indicate the key in which the music is to be played. These symbols specify the key signature.

A piece of music does not have to use just one key for an entire song. If a key signature appears somewhere in the middle of the sheet music, it's indicating a change in the key. Any previous sharps or flats are canceled, and only the sharps or flats specified by the new key signature are to be used.

Sometimes you'll see natural signs appearing in a key signature when there is a key change. These natural signs are often used when changing keys to show that the sharps and flats of the previous key no longer apply.

Since key changes are not as common as tempo or volume changes, they can be easy to miss when you're reading sheet music. A good suggestion is to look over the music before entering notes so that you'll be expecting a key change if there is one.

Time signatures. At the beginning of sheet music, right after the key signature, you'll often find a fraction such as 4/4. The top number tells how many beats there are per measure, and the bottom number tells how many beats there are per whole note. Together, these two numbers define the *time signature.*

Until now, you've used only *4/4 time,* in which each measure contains four beats, and a whole note is four beats long.

When using different time signatures, there may be more than or fewer than four beats in every measure, and a whole note may not always be four beats long.

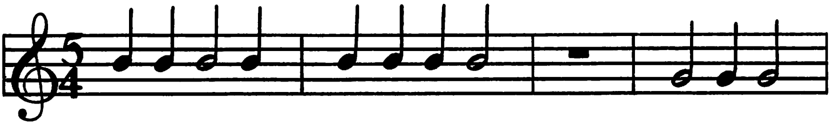
One of the more common time signatures is 3/4 time. This is just like 4/4 time, except there are only three beats in every measure. Waltzes are always written in 3/4 time.

Changing the number of beats in a measure really does not affect Sidplayer. As long as you follow the sheet music, there should be no problem. The important thing is that each measure have the same number of beats.

A common practice in time signatures other than 4/4 is to use the whole rest symbol as a *measure rest* symbol to indicate a full measure of rest. For example, if a song is written in 5/4 time and you see a whole rest, the sheet music is *not* wrong. It means that the rest should last for five beats.

The five-beat rest in Figure 5-39 would be entered as a whole rest and a quarter rest. In other time signatures, a measure rest may last a different number of beats.

Figure 5-39. Measure Rest



When the bottom number is 4, a whole note plays for four beats, so one-fourth of that duration, a quarter note, plays for one beat. If the bottom number is 2, however, a whole note is only two beats long, and a half note is one beat long. The quarter note plays for only half a beat. When the number on the bottom is 8, a quarter note plays for two beats, and the note for one beat is now an eighth note. The tempo stays the same; the number of beats per minute remains unchanged. It's the number of beats per note that's different.

Sidplayer is designed to expect a whole note always to be four beats so that a quarter note is always one beat. There's no way to change the number of beats for these standard durations. Time signatures which have a number other than 4 on the bottom can be used indirectly, however, by fooling Sidplayer into thinking that a whole note is longer or shorter than it actually is.

Consider the time signature 2/2, in which a whole note is two beats, compared with four beats in 2/4. A whole note is seemingly reduced to half its normal duration. This can be achieved on Sidplayer by doubling the tempo. At faster tempo selections, more beats per minute means that each beat takes less time, so whole notes are shorter. Therefore, when the tempo is M.M. 100, you can simulate 2/2 time by actually using M.M. 200. To use a time signature where the bottom number is 8, as in 3/8, the tempo should be cut in half, making whole notes play twice as long as normal.

There's another way to show a time signature without using numbers. The letter C placed where the time signature belongs indicates 4/4 time. The C stands for *common time.* If the C has a vertical line passing through it, it indicates 2/2 time, also known as *cut time.*

One last word about time signatures—they can change while a song is playing. Such changes are indicated by double bars followed by a new fraction or symbol.

Accidentals. In any octave, there are 12 different pitches, including naturals, sharps, and flats. Earlier, you saw that a song will use only 7 of these pitches, according to the current key. Let's retract that now. Once in a while a song may have to play a note using a pitch not in the key. The blues piece is written in the key of G, meaning that only one note is sharp (F-sharp), but the melody line had to play a C-sharp at one instance.

Special exceptions like this are handled by placing an accidental sign immediately before the note that is to be sharp or flat. This accidental overrides the current key signature for that particular pitch.

Furthermore, the effect of the accidental sign holds true for all following notes of the same pitch. A sharp sign placed in front of a C affects not only that C note, but any other C notes in the same octave which may come later. The changed accidental is not permanent, though, and is canceled at the next measure.

An accidental sign used in this way affects only the designated pitch. All other pitches remain the same.

You already know how to change a natural pitch into a sharp or flat. What about the other direction—removing a sharp or flat from a pitch to make it natural? This can be done by using a natural sign. Placed in front of a note, it cancels the sharp or flat for all following notes of the same pitch, but only within the current measure.

A common mistake is forgetting that an accidental on one note also affects later notes of that pitch in the same measure. To help with this problem, the Editor supports a measure feature.

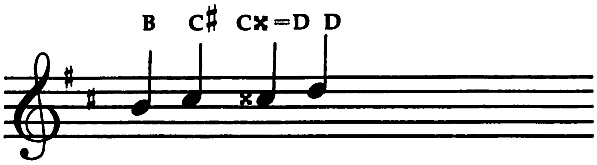
The key signature level shows whether the measure feature is on or off. Press the M key to turn this feature on. Now, whenever you enter a note with an accidental, the Editor remembers that the accidental is in effect. If you enter more notes and then move back to that pitch, the accidental is automatically selected.

Since accidentals affect only those pitches in one measure, accidentals remembered by the Editor are cleared whenever a measure marker is entered.

To turn off the measure feature, press the M key again (it acts as a toggle switch). This feature can also be selected with the joystick. Push the stick up or down while in the key signature level.

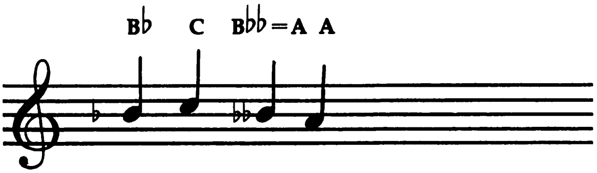
There's one more possibility. Although they're not used often, it's possible to have double sharps or double flats. Changing a natural note into a sharp note is done by increasing the pitch one half step. If this is done on a note already sharp, the pitch is bumped up another half step. The symbol to indicate a double sharp looks something like a letter X (Figure 5-40).

Figure 5-40. Double Sharps



Two flat signs are used to indicate a double flat (Figure 5-41).

Figure 5-41. Double Flats



The double sharp and double flat signs, along with the sharp, natural, and flat signs, give a total of five different accidental signs.

The Editor does not support double sharps and double flats. When you see a note with a double sharp or flat, you must calculate the pitch that should be entered. For example, if you encounter a C-double-sharp, you should enter a D-natural.

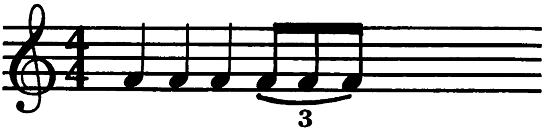
Double dots. A dot placed after a note means that the duration should be increased by one-half. Adding a second dot means that the value of the first dot should be increased by one-half. Thus, a double-dotted half note is equal in duration to a half note plus a quarter note plus an eighth note. Double dots can be used on other durations as well.

Because double-dotted notes are very rare, they're not directly supported by Sidplayer. You can simulate them, however, by entering the appropriate single-dotted note and tying it to another note of the same pitch. For example, a double-dotted C4 half note could be simulated by entering a dotted C4 half note tied to a C4 eighth note.

Triplets. All the durations, from whole note to thirty-second note, are based on the number 2. These durations allow notes to be played for one, two, or four beats, half a beat, and so on. This system works very well except that it's difficult to handle durations based on the number 3. To play a note for three beats, a dotted half note can be used. But how do you play a note for one-third of a beat? Using dots will not help there. Instead, you need to use a new kind of duration called a *triplet.*

A triplet (Figure 5-42) consists of three notes played for the amount of time normally allotted to two notes. An eighth triplet is equal in duration to two eighth notes. Because two eighth notes form one beat, each of the three notes in a triplet is one-third beat long.

Figure 5-42. Triplets



A sixteenth triplet means that the amount of time used for two sixteenth notes, or one eighth note, is to be divided into three equal parts. Each of the three notes in the triplet plays for that duration. Triplets based on other durations, such as quarter notes, are also possible. Triplets are always written with the number 3 above or below them.

To support the less common durations, Sidplayer has a *utility duration* which can be set to last any amount of time. If you want to play an eighth triplet, for example, all you have to do is set the utility duration to play for the appropriate number of jiffies, then enter the notes of the triplet using the duration marked UTILITY. Push the joystick left or right, or press the U key to select this duration.

The UTL command is used to set the utility duration. The range of this command is 0–255 jiffies, with 0 meaning 256. To calculate the number of jiffies for a particular duration, refer to Table 5-3 in the previous article. If you want an eighth triplet, find the jiffy count for a quarter note in the current tempo and divide that number by 3. Enter the result for the UTL command.

Once the utility duration has been set by the UTL command on one voice, it can be used for entering notes and rests on all three voices.

The length of the utility duration stays the same until changed by the UTL command. Since the calculation of the utility duration depends on the current tempo, the utility duration may have to be changed if the tempo ever changes.

Some tempos do not support the use of eighth or sixteenth triplets. In M.M. 128, for example, a quarter note plays for 28 jiffies. The number 28 is not evenly divisible by 3, so a different tempo has to be used.

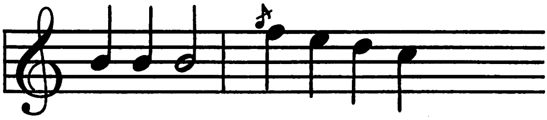
If only a few triplets are used in the music you're translating, try this. To play an eighth triplet in M.M. 128, set the utility duration to 9 jiffies and enter the first two notes of the triplet. Then set the utility duration to 10 jiffies and enter the remaining note. The total duration of the three notes will be 28 jiffies.

This trick of changing the utility duration can also be used to play sixteenth or thirty-second notes in tempos which do not normally support them. An eighth note in M.M. 120 plays for 15 jiffies. To play a pair of sixteenth notes in this tempo, set the utility duration to 7 jiffies for the first note and to 8 jiffies for the second note.

The default value of the utility duration is 12 jiffies, which is the length of an eighth triplet in M.M. 100.

Grace notes. If you see a note written a lot smaller than all the other notes, it's a grace note. A grace note is played very quickly, just long enough to be heard. Refer to Figure 5-43 for an example of a grace note.

Figure 5-43. Grace Note



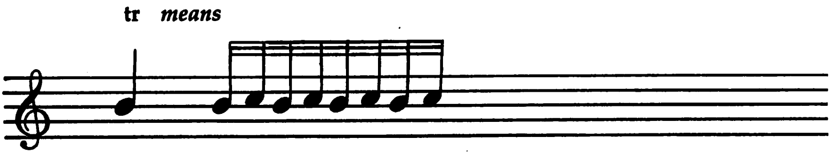
To enter a grace note, enter the preceding note with a slightly shorter amount of time, then enter the grace note with the remaining duration. Generally, the utility duration must be used to do this.

In Figure 5-43, the grace note is preceded by a half note. Let's say that the current tempo is M.M. 100, in which case a half note plays for 72 jiffies. The grace note could be entered by setting the utility duration to 68 jiffies and entering the half note with the utility duration, then setting the utility duration to 4 jiffies and entering the grace note. The total duration of the two notes (68 + 4 jiffies) will be correct.

There's no definite rule about how many jiffies should be used for a grace note. The grace note in the example could have been played for 6 jiffies, in which case the previous note would have played for 66 jiffies.

Trills. The letters *tr* above a note are used to indicate a trill (Figure 5-44). This means that the note should be played with the pitch rapidly alternating between the designated pitch and the next higher pitch in the current key.

Figure 5-44. Trill



Repeat loops can be useful in entering trills. To enter the trill shown in the example, a pair of thirty-second notes, in the pitches F and G, could be entered between a HED and TAL. The loop would be repeated four times.

Portamento and Vibrato

To make the SID chip produce a pitch, a frequency number has to be POKEd into the chip's frequency registers. The frequency numbers for all eight octaves can be found in Appendix M of the *Commodore 64 User's Guide* (the manual that came with your Commodore 64). Every time a new note is played, Sidplayer examines the pitch of the note and POKEs the corresponding frequency number into the SID chip frequency registers.

The frequency numbers are rather large. For example, the number for middle C is 4291. There's also a large gap between frequency numbers for each half step. The number for C-sharp is 4547, quite a jump from 4291.

All the numbers between 4291 and 4547 make the SID chip produce pitches between C and C-sharp. Normally, these pitches are never played. If Sidplayer plays a C followed by a C-sharp, the frequency number changes immediately from 4291 to 4547. But if the frequency number were to run through all the intermediate values before reaching 4547, the pitch would make a smooth transition from the C to the C-sharp. The pitch would *glide* from one note to the other.

The effect is called *glissando,* or *portamento* in synthesizer terminology. Gliding can be done between any two pitches, and can go up or down. The trombone is one instrument which does this naturally.

Sidplayer supports a portamento option for each voice. To turn on portamento, enter the POR command with a number greater than zero. Then, as each note is played, the pitch will glide up or down from the previous note until it reaches the new pitch.

The POR command number controls the glide rate. The larger the number, the more quickly the pitch glides from one note to the next.

An important characteristic of the rate number is that larger rate numbers must be used for higher pitches. If you examine the frequency numbers, you'll notice that the difference between the numbers increases as the pitch increases. To be precise, the difference doubles with each octave.

In order to get the same glide effect in each octave, the rate should be doubled or halved as necessary. For instance, if the glide rate is 100 when playing notes in octave 4, use a glide rate of 200 for notes in octave 5, or a rate of 50 for notes in octave 3.

The portamento feature can be turned off by entering the command POR 0.

Usually, gliding is done only a few times in a song. When done continually, the result can be rather comical, as illustrated by the song ALBUMLEAF on the *Complete 64* disk.

The effect of *vibrato* is to make the pitch waver slightly. The pitch repeatedly increases and then decreases by a small amount as each note plays. When done properly, the slight but steady fluctuation in pitch is barely noticeable, but it makes a tone sound more natural and alive.

Two commands are needed to control vibrato. The VDP command sets the vibrato depth. The number for this command ranges from 0 through 255. The larger the number, the more pronounced the vibrato effect. The most commonly used values are 1–50. Like the glide rate, the depth number should be doubled for each higher octave and halved for each lower octave.

The VRT command specifies the vibrato rate, or how quickly the tone alternates between increasing and decreasing pitches. The number ranges from 0 through 255, with 0 meaning 256. Smaller rate numbers produce faster vibrato rates. The values 1–4 are used most often.

There are no default vibrato settings, so the first time you want to turn vibrato on, both commands should be entered.

To turn vibrato off, use the command VDP 0. You don't have to do anything to the rate when you turn vibrato off. To turn vibrato back on, just enter the VDP command again. The previous rate remains in effect until a new rate is specified by another VRT command.

K.C.O, also included on the disk, demonstrates a brief but pleasant example of how vibrato can add a nice touch to a song.

Detuning

The DTN command is used to detune a voice. Detuning is accomplished by adding a constant number, specified by the DTN command, to the frequency number of each note played.

Detuning a single voice is not very useful. All it does is make the voice play slightly out of tune. When used with two voices, however, with both voices playing the same notes, but one slightly detuned, the result is a "chorus" effect. This effect was used in the Commodore demonstration song and has some interesting applications.

What's the easiest way to make two voices play the exact same notes? The answer is to use phrases. A phrase defined on voice 1 can be called simultaneously on voice 2 or 3. You don't have to wait for playing to reach the END command before calling the phrase; the phrase is available as soon as Sidplayer has processed the DEF command.

This means that you can define the phrase on voice 1 and call it on voice 2 after having set the detune value.

```
Voice 1: DEF 0 Play Notes END
Voice 2: DTN 50 CAL 0
```

Either voice can be detuned. The DTN command can be put either before the DEF 0 or before the CAL 0. Just make sure that the DTN command is not included in the phrase definition. You don't want both voices detuned.

When entering the DTN command, use a number from 1 through 2047 to tune the voice a little sharp. The larger the value, the more the voice will be detuned. Values around 50 work best. Enter a number from −2047 through −1 to tune the voice flat. For most applications, the direction makes no difference.

As with the glide rate and vibrato depth, larger values have to be used for higher pitches.

Enter the DTN 0 to turn the detuning off.

The only drawback to detuning is that it takes two voices, so only one voice is left free. The effect can sometimes be worth it, though. PIPERS on the *Complete 64* disk demonstrates just how realistic a sound can be created with your Commodore 64.

Transposing

Detuning works by adjusting the pitch of each note so that it's slightly sharp or flat. Transposing also changes the pitch of each note, but in quite a different way.

When a voice is transposed, the pitch of each note is shifted up or down a designated number of half steps. For example, if a voice is being transposed up one half step, all notes entered as C play as C-sharps, all C-sharps play as D, every D plays as D-sharp, and so on. If the voice is being transposed down one half step, D notes play as C-sharps, C-sharps as C-naturals, C-naturals become B-naturals, and so on. You can check these alterations by tracing the transposing using Table 5-4.

Table 5-4. Transposing

C
B
A♯/B♭
A
G♯/A♭
G
F♯/G♭
F
E
D♯/E♭
D
C♯/D♭
C

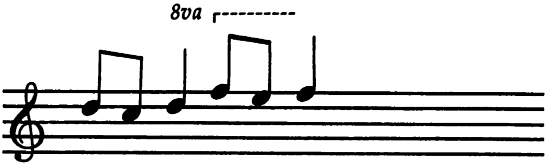
Transposing can be done by more than 1 half step. In a voice being transposed up by 7 half steps, for example, a note entered as C will play using the pitch for G. Since there are 12 half steps in an octave, transposing up or down by 12 half steps makes a voice play an octave higher or lower.

The TPS command is used to specify the number of half steps a voice should be transposed. Enter this command with a number in the range of 1 through 95 to transpose a voice up by 1 through 95 half steps. Enter −1 through −95 to transpose the voice down. Enter 0 to turn transposing off.

Transposing applies to all notes equally, so unlike some of the previous commands, the value does not need to be changed for higher or lower pitches.

There are many useful applications for the TPS command. For instance, you may sometimes see a dashed line appearing above or below a sequence of notes.

Figure 5-45. Octave Offset



The dashed line, along with the message *8va* or *8vaba,* means that the notes should be played one octave higher or lower than written. This is done because writing the notes with the correct pitches would require too many leger lines, and the notes would be difficult to read.

The easiest way to handle an octave offset is to enter the notes as written and then insert a TPS 12 or TPS −12 command at the beginning of the offset.

An interesting effect is created by having two voices play the same note, while one of the voices is transposed up one or two octaves. This produces a rich, warm tone. The effect was used in the Commodore demonstration song. Another possibility is to use transposing with detuning. While one voice plays a series of notes, the other plays the same notes, but is detuned and transposed up one or two octaves. Tunes CALLIOPE and BISTRO on the *Complete 64* disk use this technique to imitate a calliope and an accordion.

Synchronization

In the previous section, it was mentioned that a rich, warm tone can be created when two voices play the same notes and the second voice is transposed up by 12 or 24 half steps. But if you try this technique with the second voice transposed up by other numbers of half steps, like 6, 11, or 13, the resulting tone sounds less than desirable.

The reason that the pleasant tone is produced only when the second voice is transposed up an octave or two is because the voices become synchronized. Remember that the frequency of a note is doubled when it's played an octave higher. When two voices play the same notes, one an octave higher, the direct relationship between the frequencies causes the tones to be synchronized.

The tone produced by two synchronized voices sounds rich and warm because it contains more harmonics than usual. The second voice augments the harmonics of the first one. This method of adding harmonics to a tone is called *additive synthesis.* (Filtering is called *subtractive synthesis* because the filter removes harmonics from a tone.)

Synchronization occurs naturally at intervals of 12 half steps, but it can be made to occur at any half-step interval if the synchronization mode of the SID chip is turned on. The advantage to using the synchronization mode is that different half-step intervals produce different harmonic patterns. Synchronization, therefore, lets you produce many new types of tones.

The SNC command is used to control the synchronization mode. Enter 1 (yes) to turn the mode on.

A specific procedure must be followed in order to synchronize two voices. Let's say that you want to synchronize voices 1 and 2. First, define voice 1 to be a phrase. On voice 2, you should enter the SNC YES command and the TPS command. Choose any number of half steps, such as 8. Then call the phrase that was defined on voice 1.

DEF	**on voice 1**
SNC	**on voice 2**
TPS	**on voice 2**
CAL	**on voice 2**

Play the two voices several times, each time transposing voice 2 by a different number of half steps, to hear the various types of tones that can be created. Each waveform produces different effects when synchronized, so you might also try changing the waveform.

Both voices should have the same envelope, but they do not have to have the same waveform. For example, you might use a square wave on voice 1 and a triangle wave on voice 2.

To synchronize voices 1 and 3, use this procedure:

SNC on voice 1
TPS on voice 1
DEF on voice 1
CAL on voice 3

Use this procedure to synchronize voices 2 and 3:

DEF on voice 2
SNC on voice 3
TPS on voice 3
CAL on voice 3

It's very important that you use only these procedures. Synchronization will not work correctly if the SNC or TPS command is used on the wrong voice. Also, only positive transpose values should be used. Negative values do not seem to be very useful.

When you want to stop using the synchronization mode and return to normal playing, enter the SNC command with the value 0 (no) to turn the mode off. Do this on the voice which earlier turned the mode on. You may also want to enter a TPS 0 command to cancel the transposing.

On the *Complete 64* disk, TPI#14 uses different half-step intervals and different waveforms to demonstrate some of the effects possible with synchronization.

Ring Modulation

A ring modulator takes two frequencies and produces two new frequencies in their place. The new frequencies are the sum and difference of the originals. For example, if a ring modulator is given frequencies of 200 and 300 Hz, it will produce the frequencies of 100 and 500 Hz, but not 200 or 300 Hz.

Sometimes it can be difficult to discern the pitch of a ring-modulated tone. The two frequencies may be interpreted by your ear as a single pitch, but the frequency corresponding to that pitch is not one of the two produced by the ring modulation, so it does not really exist.

Ring-modulated tones are useful for creating percussion effects. Bells, chimes, steel drums, and various instruments made of metal, wood, or glass can be approximated.

When used simultaneously with synchronization, a whole new set of tone colors (timbres) is available.

To use ring modulation, a definite procedure must be followed. Two voices must play the same sequence of notes, just as when you are using the synchronization mode. With ring modulation, however, only voices 1 and 3 can be used.

Voice 1 is used to supply the waveform and envelope of the ring-modulated tone, plus one of the two frequencies needed for the process. The first thing voice 1 should do is select the triangle waveform. This waveform must be used in order for ring modulation to work.

Since ring-modulated tones are used mainly for percussion effects, a nonsustaining envelope should be selected.

Finally, enter the command RNG with the value for yes to turn ring modulation on. Voice 1 is now ready to start playing notes. These notes should be put in a phrase definition so that they can be called by voice 3.

Voice 3 is used only to supply the second frequency needed for ring modulation, so the waveform and envelope of voice 3 do not have to be set. In fact, it's best to turn off the output from this voice completely, by using the 3-O (3 OFF) command. Enter this command with the value for yes at the beginning of voice 3. The frequencies of notes played by voice 3 are still used in the ring-modulation process, but the output from voice 3 itself is not heard. Voice 3 can now call the phrase defined in voice 1.

Here's an example sequence of commands that could be entered to use ring modulation.

Voice 1

```
WAV T
ATK 0
DCY 9
SUS 0
RLS 9
PNT 1
RNG Y
DEF
```

Voice 3

```
3-O Y
CAL
```

As with synchronization, different tone types can be produced by using the TPS command. With ring modulation, however, transposing one voice can cause the resulting pitch to be out of tune, so it's often necessary to transpose the other voice as well to bring the pitch back in tune.

Table 5-5 is a chart ordered by increasing half-step intervals. For each interval, the appropriate transpose values for voices 1 and 3 are given.

Table 5-5. Transpose Values for Use with Ring Modulation

Halfstep Offset	V1 TPS	V3 TPS	Highest Note
−20	7	−13	E7
−19	7	−12	E7
−18	7	−11	E7
−17	-	-	
−16	6	−10	F7
−15	6	−9	F7
−14	6	−8	F7
−13	-	-	
−12	0	−12	B7
−11	-	-	
−10	2	−8	A7
−9	4	−5	G7
−8	2	−6	A7
−7	7	0	E7
−6	10	4	C♯ 7
−5	0	−5	B7
−4	2	−2	A7
−3	-	-	
−2	-	-	
−1	-	-	
0	0	0	B7
1	−1	0	B7
2	-	-	
3	-	-	
4	0	4	G7
5	−5	0	B7
6	4	10	C♯ 7
7	0	7	E7
8	-	-	
9	−5	4	G7
10	4	14	A6
11	-	-	

12	0	12	B6
13	–	–	
14	−4	10	C♯ 7
15	−5	10	C♯ 7
16	0	16	G6
17	−5	12	B6
18	1	19	E6
19	0	19	E6
20	–	–	
21	–	–	
22	−2	20	E♭ 6
23	4	27	G♯ 5
24	0	24	B5
25	–	–	
26	0	26	A5
27	1	28	G5
28	0	28	G5
29	–	–	
30	−5	25	B♭ 5
31	0	31	E5
32	−5	27	G♯ 5
33	1	34	C♯ 5
34	0	34	C♯ 5
35	0	35	C5
36	0	36	B4
37	0	37	B♭ 4
38	0	38	A4
39	0	39	G♯ 4
40	0	40	G4

The transpose values for some of these intervals are close approximations and don't produce a distinct pitch, which is why they're best used for percussion effects. Those intervals for which no satisfactory transpose values could be found have been marked with a dash (–).

When the transpose value for a voice is 0, no TPS command has to be entered for that voice.

The rightmost column tells you the highest note that can be played using the given transpose values. For example, when the 9 half-step interval is used, only notes up to G7 should be used. Notes above G7 won't play properly.

You can transpose the tone up or down one or more octaves by adding a multiple of 12 half steps to each transpose value. Let's say that you are using the 26 half-step interval, in which the transpose values are 0 and 26. Adding −12 half steps gives the transpose values −12 and 14.

Be aware that the effect of ring modulation may vary from one octave to another. Notes played in one octave may sound quite different when played with the same half-step interval in a different octave.

By now you might be wondering just what ring-modulated tones sound like. HOLST (on the *Complete 64* disk) is a demonstration that uses the 26 half-step interval.

Only a few of the half-step intervals listed in Table 5-5 produce very pleasant or useful sounds. Too often, the sound produced by one interval includes an undesirable high-pitched tone. But if you pass the ring-modulated tone through the filter, with the filter mode set to low pass, the squeal can be eliminated. Most of the intervals listed in the table then become useful.

Use the filter on voice 1 only. Here are some example settings.

F-M L
AUT 1
RES 15
FLT Y

Try using different filter modes, including combinations. Also try different cutoff settings on the AUT command. The resonance value set by the RES command can be adjusted to take some of the "bite" out of a tone.

Since there are so many variable factors, some experimentation will be needed. This can be worthwhile, however, because you'll discover some timbres that you've never heard before from the SID chip. Some intervals to try are the intervals with −7, 28, and 38 half steps.

One remaining possibility is to use synchronization and ring modulation at the same time. This technique yields yet another set of timbres.

To use both modes simultaneously, follow this procedure.

Voice 1
WAV T
SNC Y
RNG Y
TPS
DEF

Voice 3
3-O Y
CAL

The triangle waveform is still required, but now any envelope can be used. Also, every half-step interval produces a usable tone, so you can disregard Table 5-5. The transposing always stays in tune, so it has to be done only on voice 1. As with synchronization, however, negative transpose values are not very useful.

To turn ring modulation off, enter the command RNG N on voice 1. Enter the command 3-O N on voice 3 to reenable the output of that voice. The command TPS 0 should be entered on those voices that were being transposed. If you were also using the filter, enter the commands FLT N and AUT 0 on voice 1. Or, if you were using synchronization, enter the command SNC N on voice 1.

Whether used alone, with the filter, or with synchronization, the ring-modulation feature of the SID chip can produce some very interesting sounds. Listen to the piece on the disk called PROMENADE; it is another example of the techniques.

Advanced Techniques

This section explores some new uses for the commands introduced in the previous chapters. The techniques suggested here should give you an idea of just what's possible with Sidplayer.

Waveform. Probably the most overlooked waveform is the noise waveform. Although this waveform does not have a definite pitch quality, its character does change as the pitch is changed. High pitches produce a hissing sound, whereas low pitches create more of a rumble.

An interesting use of the noise waveform is to play white-noise notes with different pitches to create a repeating percussion line. The sequence of notes should use short durations and should repeat every one or two measures. With just a little effort, you can create a fancy rhythm that adds a nice touch to a song.

The character of the noise waveform can be altered significantly by passing it through the filter. Try this with the filter set for the high-pass mode.

As for the other three waveforms, one possibility is to use two waveforms at the same time on the same voice, with the objective of creating new types of sounds. Unfortunately, only the combination of triangle and pulse waves is audible.

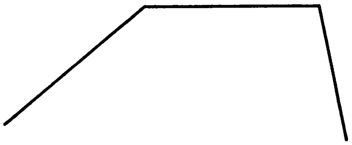
To select the triangle and pulse wave combination, enter the WAV command with the number 5 (1 for triangle plus 4 for pulse). This waveform sounds a lot like the pulse wave with the pulse width set to about 200, but with more of a buzz.

When a voice is playing with the triangle/pulse combination, the pulse width acts as a volume control for the voice. Setting the pulse width above 2048 makes the voice inaudible; smaller values make the voice sound louder.

Envelope. Thus far, you've concentrated on emulating conventional instruments. A synthesizer, however, is capable of producing any kind of sound, natural or unnatural.

One way to make a voice sound unnatural is to use a backward envelope (Figure 5-46). Such an envelope has a slow attack rate and a very fast release rate.

Figure 5-46. Backward Envelope



If you've ever heard a recording of a song played backward, you've heard the backward envelope. This type of envelope is not characteristic of any normal instrument.

A major drawback to the SID chip is that there's no master volume control for each voice. To some extent, the sustain level can be used as a voice volume control. Select sustain levels less than 15 to make a voice a little quieter. You may also want to adjust the decay rate so that the volume does not drop from the peak to the sustain level too rapidly.

One other technique is to experiment with the attack rate and release point so that each note is released before the volume hits the peak level (before the attack phase is complete). In order for this to work, the release rate must be set so that each note fades to complete silence before the next note begins. As you might guess, this isn't an easy technique to use.

Filter. Pin 5 of the monitor jack can be used to send an audio signal to the SID chip. This input signal is mixed with the output from the three voices before being sent to the master volume control.

If you want to pass the external audio signal through the filter, use the F-X command. This command works just like the FLT command and accepts a yes/no data value.

Since using an external audio input is an advanced application of the SID chip, you'll rarely use the F-X command.

Portamento. When the portamento feature is turned on, the pitch glides from one note to the next. The starting pitch of the glide is the pitch of the previous note, and the ending pitch is the pitch of the new note. For example, if a G note is played after a C, the pitch will start at C and end at G.

The *absolute set pitch* command can be used to change the starting pitch of a glide. In the earlier example, a command to absolutely set the pitch at E could have been placed between the notes C and G. When the G was then played, the pitch would have glided from E to G, instead of from C to G.

Unlike the other commands, the absolute set pitch command is entered from the editing screen. To absolutely set the pitch, select the name, accidental, and octave of the pitch, change the duration so that it reads ABS SET, and press the joystick button.

Although this command is entered like a note, it is still a command because it has no duration. The command is entered like a note because the editing screen offers a convenient method of specifying a pitch.

One application of this command is to create pitch-bending effects. To bend the pitch of a note, turn portamento on, set the pitch one half step below the pitch of the note, play the note, and turn portamento off. Another application is discussed later, in the section on synchronization.

The absolute set pitch command has no effect on playing when the portamento feature is turned off.

Vibrato. An advanced technique to use with vibrato is to change the depth of the vibrato during a note.

Let's say that a voice is playing a whole note. Break the whole note into four quarter notes. Enter the first quarter note with a tie, and follow it with a VDP command to set the vibrato depth. Do this again for the next two quarter notes, each time using a larger depth value. Enter the last quarter note without a tie.

When played, the tied quarter notes sound like a whole note, and the different VDP commands make the vibrato effect deepen gradually. The result is a more natural vibrato effect.

Remember to reset the vibrato depth for the next note.

Detuning. Sidplayer can play a note using any frequency, including frequencies between the normal half steps. Just choose a note with a pitch near the desired frequency, and use the DTN command to set a frequency offset value. Refer to Appendix M of the *Commodore 64 User's Guide* to find the frequency number for each pitch.

As an example, let's say that you want to play a note with a frequency of 500 Hz. Use the following formula to calculate the corresponding frequency number:

Frequency number = Frequency/0.06097

The frequency number for 500 Hz is 500/0.06097, or 8201. The pitch with the closest frequency is B4, which has the frequency number 8101. Therefore, to play a note at 500 Hz, all you have to do is enter the command DTN 100 followed by the note B4.

Transposing. By using the TPS command, it's possible to play notes in a ninth octave. If you play notes in octave 0 after entering the command TPS −12, the notes will actually play in octave −1.

It's very hard to hear the pitch of notes in octave 0, and even harder to hear notes in octave −1, so this technique is probably useful only for creating special effects.

Unfortunately, this technique won't work with notes above octave 7. If a note is transposed into octave 8, it's still played in octave 7.

Another possibility is to play a phrase in different keys by entering the TPS command with the correct number of half steps before each phrase call.

Synchronization. When two voices are synchronized by the sync mode, a rather unusual effect can be created by using portamento on the voice which adds the harmonics.

One technique is to enter an absolute set pitch command before each note. The command should set the pitch one octave below the pitch of the following note so that the pitch glides up one full octave when the note is played. This technique was used in the song ETAL.

Future expansion. The AUX command is reserved for possible future expansion. At present, this command is ignored by Sidplayer.

Uncommon instruments. Sidplayer can emulate instruments which are no longer commonly used. The tune COURANTE, the next file on the disk, imitates instruments that were popular in the late Renaissance period.

Sound effects. Sidplayer can be used to produce sound effects as well as music. Enjoy the effect created with white noise at the end of the file JOKE from the *Complete 64* disk.

Phonetics. Certain combinations of frequencies are associated with different vowel sounds, as demonstrated in YOY on the disk.

More than three voices. The most serious limitation of the SID chip is that it supports only three voices. Many classical music pieces can be played with three voices, but most of today's songs need at least four.

Sometimes when a song has to play four notes, one of the notes can be eliminated without significantly affecting the music. When a note in the bass clef is also played a couple of octaves higher in the treble clef, the treble clef note can be dropped. If the treble clef contains notes which do not seem to be part of the melody, try deleting them. When it's not obvious which note should be dropped, experiment.

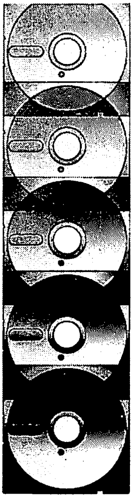
Contemporary sheet music is often written with two treble staves and one bass staff. The top treble staff usually has just the notes for the melody, and the other one has the melody notes plus extra notes for chords. If you enter the top treble staff and the bass staff, you'll still have one voice free for percussion effects.

A clever technique is to have one voice play both a bass part and percussion part. Bass notes can be played on each beat, and percussion effects, such as snare drum strikes, can be played on the off beat. To make this as easy as possible, define a phrase which selects a nonsustaining envelope with the noise waveform, plays a note, and then resets the envelope and waveform. To enter the voice, just enter bass notes separated by phrase calls.

Instrument parameters. Table 5-6 lists suggested values for emulating different instruments. Filter values may have to be adjusted for your particular computer. This list is by no means complete; many other instrument settings are possible and can be found by experimentation.

Table 5-6. Sidplayer Instrument Parameters

Instrument	ATK	DCY	SUS	RLS	PNT	WAV	P-W	P-S	F-M	AUT	RES	FLT	F-S	F-C	SNC	RNG	VDP	VRT	TPS	DTN
DEFAULT	2	0	15	5	4	P	2048	0		0	0	N	0		N	N	0	0	0	
VIOLIN	0	0			4	P	800										30	2	12	
CELLO	5	0	12	4	4	P	2048										20	2	−12	
BASS (pluck)	1	10	0	10	1	P	1800		L	−10	9	Y							−12	
HARP	0	9	0	0	1	T			LB	−1	15	Y								
BANJO	1	8	0	0	1	P	100		B	3	12	Y	5							
MANDOLIN	1	8	0	0	1	P	500		H	−20	12	Y								
BAZOOKI	1	10	0	0	1	P	1000		H	−20	12	Y	5							
LUTE	0	8	0	0	1	T			L	−40	15	Y								
KOTO	0	8	0	0	1	P	300		L	−10	15	Y	20							
SITAR	2	10	12	12	7	P	100	10	H	−20	15	Y	5							
FLUTE	3	0	14	4	1	T														
PICCOLO	3	0	14	4	1	T													12	
RECORDER—ALTO	3	0	14	4	1	T														
CLARINET	3	0	14			P	2048													
OBOE	7	0	15	2	1	P	700		B	−20	15	Y								
BASSOON	7	8	14	4	1	P	700		B	−20	15	Y							−12	
BAGPIPE	2	0	15	5	1	P	200													70
HARMONICA	5	0	8	4	4	P	2048										20	2		
ACCORDION	2	0	15	4	4	S														70
SHAWM	2	2	14	2	1	P	100		B	5	15	Y								
TRUMPET	5	0	15	4	1	S			L	20	15	Y	1							
TROMBONE	9	0	15	4	1	S			L	1	15	Y								
TUBA	9	0	15	4	1	S			L	10	15	Y								
HARPSICHORD 4′	0	12	8	9	8	P	100													
HARPSICHORD 8′	0	12	8	9	8	P	300													
ORGAN: FLUTE	3	0	14	4	1	T														
ORGAN: PRINCIPAL	3	2	10	4	1	P	2048													
ORGAN: REED	3	0	14	4	1	P	200													
ORGAN: TRUMPET	3	0	15	4	1	S														
CALLIOPE	0	12	13	5	4	P	2048												12	70
FINGER DRUM	0	9	0	0	1	T														
BELLS	0	10	0	10	1	T										Y			7,16	
CHIMES	0	10	0	10	16	P	2048													70
WOOD BLOCK	0	2	0			T														



Music and Your BASIC Programs

Merging Sidplayer with BASIC Programs

Run the player program, and while it's playing a song, press the RUN/STOP key. The music still plays, even though the program has stopped. Now type LIST. The music continues as the program is listed on the screen. Enter a few more statements, such as POKEs to change the screen colors, and then enter the CONT command to make Sidplayer resume. Perhaps not the most exciting demonstration you've seen, but it does illustrate something important.

Sidplayer has been designed so that it can play music while BASIC executes commands, statements, or even a whole program. Every 1/60 second, BASIC processing is temporarily set aside and Sidplayer is allowed to process the music. When the music processing is done, BASIC resumes. Since this happens 60 times a second, the continual interruption of BASIC is too fast to be noticeable. BASIC and the Sidplayer appear to be executing simultaneously.

Although this technique causes BASIC processing to run a little slower than normal, it does make it easy to add music to games, adventures, or educational programs. And that can enhance your own BASIC programs.

Load and play procedure. To demonstrate how to merge Sidplayer with a BASIC program, let's begin with a simplified player program. The program listed on the menu called SID.BAS (this is both the name that appears on the menu and the actual filename on the disk) contains only the statements necessary to load and play a song, and can be readily merged with another program.

When adding this to your own program, all of the lines before line 57000 can be renumbered to start at any place in the program. The lines starting at 57000, however, must not be renumbered.

The program begins with the standard assignments for using tape or disk I/O and assigns values to variables to be used with SYS.

```
120 DN=8:SA=780:SX=781:SY=782:SP=783
```

The next step is to load the Sidplayer machine language into memory.

```
130 GOSUB 57000
```

This subroutine loads the SID.OBJ file, and also assigns the variables SS, HK, PL, and DP. These four variables are used later in the program. Be sure to have the proper disk in the drive.

Before you can load a song into memory, the music filename and the LOAD address must be set. Assign the name of the music file to the variable F$. Do not include the .MUS extension as part of the filename.

```
200 F$="COMMODORE" . . .
```

Line 200 continues by calculating the LOAD address.

```
200 . . . :LA=PEEK(49)+256*PEEK(50)+1000 . . .
```

The LOAD address is the address of free memory, plus 1000 bytes for a safety margin. This is the address in memory where the music file will be stored.

The final statement on line 200 calls the subroutine at line 57500 to actually load the song. Again, be sure the proper disk is in the drive. If you assigned F$ as above for a demonstration, make sure that the COMMODORE.MUS file is on that disk.

```
200 . . . :GOSUB 57500
```

Besides loading the requested music file, the subroutine at 57500 also assigns the variables LO and HI to be the low and high bytes of the LOAD address. These variables are used later in the program.

For clarity, here is line 200 as it appears in the program SID.BAS:

```
200 F$="COMMODORE":LA=PEEK(49)+256*PEEK(50)+1000:
    GOSUB 57500
```

At this point, everything has been loaded into memory. There's still a little bit of preparation which must be done before the playing can begin.

```
210 SYS HK
```

This statement calls the Sidplayer HOOK routine, which installs Sidplayer as part of the normal interrupt processing that's done every 1/60 second.

```
220 POKE SX,LO:POKE SY,HI:SYS PL
```

The PLAY (SYS PL) routine tells Sidplayer where the song begins in memory and sets all the default values, such as tempo and volume. It also returns the address of the text lines in locations SX and SY. It's not necessary to print the text lines in order to play the song, so the next two program lines (lines 230 and 240) are optional. If you *do* want to print the text lines, the program should start displaying characters at the returned address and stop when it reaches a CHR$(0).

The song is now ready to start playing. The following statement makes the playing begin:

```
250 POKE SS,7
```

Location SS is the Sidplayer status value. The last three bits of this location control voice processing.

Voice	SS Bit	POKE Value
1	0	1
2	1	2
3	2	4

When a bit is set, the corresponding voice plays. Playing for that voice stops when the bit is cleared, or set to 0. An individual voice can be played by POKEing location SS with the value 1, 2, or 4. To play all three voices, POKE SS with 1+2+4, or 7.

Playing continues as long as the three bits are set. Playing can be stopped at any time by POKEing SS with a 0. The bits are automatically cleared by Sidplayer when it reaches the end of the song or if an error occurs.

SID.BAS simply loops until the three bits in SS are cleared when the end of the song is reached, but anything can be done during this time. The program could draw pictures on the screen, change the colors, or move sprites.

After a song is done, it can be replayed if you like. Just call the PLAY routine and set the three status bits again. Use this line:

```
POKE SX,LO:POKE SY,HI:SYS PL:POKE SS,7:REM REPLAY
```

When playing is finished, call the DROP routine to remove Sidplayer from the interrupt processing.

```
270 SYS DP
```

The DROP routine undoes everything done by the HOOK routine and restores the interrupt processing to normal.

You must be careful in using the HOOK and DROP routines. Do not call the HOOK routine if Sidplayer is already installed. Also, do not call the DROP routine if Sidplayer has already been removed or has not yet been installed. The correct order is HOOK, PLAY, then DROP. Calling these routines in the wrong order can cause the computer to crash.

While a song is playing, BASIC is free to execute any statements and do any processing you wish. The only restrictions concern using string variables and disk I/O. If string variables are assigned frequently, the program should periodically call the free memory function, as in K=FRE(0), to reorganize free memory. Otherwise, the string data may interfere with the free memory used for storing the song.

The processing that's done every 1/60 second is handled differently when the computer is communicating with the disk drive. Therefore, disk files should not be accessed while a song is playing.

The flag command. Sometimes it would be convenient if a BASIC program could determine which part of a song Sidplayer is currently playing. Such information would be helpful in synchronizing screen displays to the music. What's needed is a method of communication between Sidplayer and the BASIC program.

The BASIC program can control Sidplayer by setting or clearing the three status bits in location SS. For communication from Sidplayer to the BASIC program, the FLG command is available. This command is entered in the Editor with a number from 0 through 255. When playing reaches the command, Sidplayer POKEs the number into location SS+1, the *flag* location. The BASIC program can monitor this location to watch for specific values and change the screen accordingly.

Several FLG commands may be used in a song. The general procedure is to wait for the value in the flag location to change, update the screen, wait for the next value, and so on. To detect a change in the flag location, a few different methods can be used.

The first method is to wait for a specific value to be POKEd into the flag location, as in

```
400 IF PEEK(SS+1)<>2 GOTO 400
```

This method requires a direct correspondence between the flag values in the music and the values being checked in the program. Usually, the flag commands will use incrementing numbers, so the BASIC program would first watch for the value 1, then for the value 2, and so on.

Another method is to disregard the value in the flag location and just wait for it to change. This can be done in two ways:

```
400 P=PEEK(SS+1)
410 IF PEEK(SS+1)=P GOTO 410
```

or

```
400 POKE SS+1,0
410 IF PEEK(SS+1)=0 GOTO 410
```

The advantage of this method is that you don't have to remember the exact flag values used in the music.

One other method is to use the WAIT statement:

```
400 WAIT SS+1,1:REM WAIT FOR ODD NUMBER
```

or

```
400 WAIT SS+1,1,1:REM WAIT FOR EVEN NUMBER
```

The advantage of using the WAIT statement is that you don't have to use a whole program line. Other statements can be placed after the WAIT. The only drawback is that the flag values must alternate between even and odd numbers.

To show how to merge music with a BASIC program and how to use the FLG command, we've provided a demonstration program. Select SIDDEMO from the menu. It is a BASIC program which uses colorful screen displays and sprites while the music plays. SIDDEMO uses the music file SCIPIO.MUS. This song contains FLG commands on voice 1.

The halt command. With the FLG command, the BASIC program waits until Sidplayer reaches a certain point in the music. An alternative is to have the playing stop and wait until the program is ready for the music to continue. This is possible with the HLT command.

HLT is used to stop the playing on a particular voice. When processed, this command clears the corresponding status bit and sets the frequency to 0 to reduce the background noise of the SID chip. To make the playing continue, the BASIC program can simply set the status bit again.

Multiple songs. Once a song is through playing, it's a simple matter to load and play another song. Just assign the name of the new song to F$, recalculate the LOAD address and assign it to LA, and call the subroutine at 57500. To play the song, follow the standard HOOK, PLAY, and DROP procedure.

Another possibility is to hold more than one song in memory at the same time. The only limit to the number of songs that can be stored in memory simultaneously is the amount of free memory available. When the subroutine at 57500 loads a song, it changes the value of LA to the address of the first free byte after the song, which is the address where the next song should be loaded. This makes it easy to load one song after another.

Load the first song in the usual way:

```
F$="SONG1":LA=PEEK(49)+256*PEEK(50)+1000:GOSUB 57500
```

Remember the values of LO and HI for later use:

```
L1=LO:H1=HI
```

To load the second song, assign the new filename to F$, but don't change the value of LA. Call the subroutine at 57500, and remember the new values of LO and HI:

```
F$="SONG2":GOSUB 57500:L2=LO:H2=HI
```

This procedure can be repeated as many times as necessary. To play one of the songs, just POKE the appropriate low and high byte values into locations SX and SY before calling the PLAY routine:

```
400 SYS HK:REM INSTALL SIDPLAYER
410 POKE SX,L1:POKE SY,H1:SYS PL:POKE SS,7:REM START
    SONG1
420 IF PEEK(SS)AND7 GOTO 420:REM WAIT UNTIL SONG1
    ENDS
430 POKE SX,L2:POKE SY,H2:SYS PL:POKE SS,7:REM PLAY
    SONG2
440 IF PEEK(SS)AND7 GOTO 440:REM WAIT UNTIL SONG2
    ENDS
450 SYS DP:REM REMOVE SIDPLAYER
```

Compatibility. Sidplayer uses memory from location 49152 through 51199, and zero page addresses 251 through 255. Sidplayer should be compatible with other utilities, such as machine language routines for bitmapped graphics drawing or sprite animation, as long as they don't use these memory locations.

Since the bitmapped graphics routines published in this book use some of the same memory as Sidplayer, there would seem to be a problem. Fortunately, it's possible to relocate a portion of Sidplayer to free memory so that there's no conflict.

To load and relocate the Sidplayer machine language, calculate the LOAD address, assign it to the variable LA, and call the subroutine at 57100. Do this instead of calling the subroutine at 57000.

```
200 LA=PEEK(49)+256*PEEK(50)+1000:GOSUB 57100
```

The subroutine loads SID.OBJ into memory, then moves a portion of it to the specified address in free memory. After the relocation, only memory from 49152 through 49663 is used by Sidplayer.

The subroutine at 57100 also sets LA to the address of the first free byte after the Sidplayer code. Therefore, the LOAD address should not be changed when the song is loaded.

```
210 F$="SONG":GOSUB 57500
```

The song is now ready to be played.

If you're using bitmapped graphics shapes, the procedure is just a little more complicated. The normal procedure to load a shape file is to assign the name of the shape file to F$, calculate the LOAD address and assign it to LA, and call the routine at 56500. When using shapes with Sidplayer, the same procedure should be used, except the value of LA should *not* be changed.

```
220 F$="SHAPES":GOSUB 56500
```

Utility Programs

The *Complete 64* disk includes four utility programs to help with song debugging and music file management.

Lister. The bottom level of the editing screen can display only a few notes at a time. Sometimes it would be helpful to see more notes at one time. This would make it easier to search for a particular note or command, such as a phrase definition, and would also make it easier to find mistakes, such as a measure with the wrong number of beats.

"Sid Lister" (filename LISTER) is used to list the notes and commands of the three voices in a music file. Although the notes are not displayed in the grand staff format, the program does let you see one or two full measures at a time when the listing is shown on the screen. The listing can also be sent to a printer for permanent reference.

Once run, the program first asks for an output device number. Enter 3 for the screen or 4 for the printer. For now, just press the RETURN key to choose 3, the default value displayed in parentheses.

The next prompt asks for the name of the file to be listed. As always, do not include the .MUS extension as part of the filename. A good song to examine for purposes of explanation is COMMODORE.MUS.

Lister then requests a starting voice number. A listing usually begins with voice 1, but sometimes you may want to skip voice 1 or voice 2. Enter the number of the first voice you want to list, or just press the RETURN key to start at voice 1. The program loads the requested file and begins the listing.

Each note is listed by its name, accidental, octave number, and duration.

C S 4 Q (C-sharp, octave 4, quarter note)

The first letter specifies the note name; it can be a letter A–G or the letter R for rest. If the note is sharp or flat, S or F is displayed after the note name. The octave number, 0–7, comes next, followed by a letter for the duration. The duration can be W, H, Q, E, S, T, or U. Respectively, these letters stand for whole, half, quarter, eighth, sixteenth, and thirty-second notes; U represents the utility duration.

If the note is dotted, a D shows after the duration. If the note is tied, the note listing ends with a T.

C S 4 Q D T (C-sharp, octave 4, quarter note, dotted, and tied)

Commands are listed with their full names. If a command has a data value, such as a number or yes/no indication, the value is placed after the command name.

TEMPO 90
WAVEFORM T

Measure numbers are displayed at the left edge of the listing, separate from the notes and commands. This makes it easy to find a particular measure. At the end of each measure, the total jiffy count for that measure is printed. This number is affected by the tempo and the time signature. For example, COMMODORE.MUS is written in 4/4 time and is played in M.M. 90. If you refer to Table 5-3, four beats at tempo 90 give a total of 160 jiffies. That's why the measure duration for each measure in COMMODORE.MUS is 160 jiffies.

If you're listing a song and one of the measures has a jiffy count different from the others, it could indicate that the measure doesn't have the correct total number of beats. Either a duration is wrong, a note is missing, or there's an extra note.

The measure duration feature won't work properly when repeats or phrases are used. Also, the total number of jiffies for a measure may change from one voice to another if the TEM command is not used on each voice.

If the listing goes by too fast, you can make the screen scroll slower by pressing the CTRL key. If that's still too fast, or if you want to stop the listing, just hit the RUN/STOP key. Enter CONT to make the listing continue.

To send the listing to the printer, enter 4 in response to the OUTPUT DEVICE prompt. Make sure that the printer is connected and turned on.

If you have an RS-232 printer instead of a parallel printer, you may have to change the OPEN statement in line 805 of the Lister.

Merge. The Editor has a limited amount of memory available for editing a song. Some songs may be just too long for it to handle. The solution is to edit the song in sections and then merge the sections together.

"Sid Merge" (filename MERGE.SID) is used to combine two or more music files into one larger file. This file can then be played by the Player. Another use of Merge is to copy a music file from one disk to another disk.

The program begins by asking for a file to load. After you enter the filename and the song is loaded, the program requests the name of the file to be appended to the first file. When you enter this second filename, the program reads each voice in the file and adds it to the end of each voice in the first song.

The program then asks for the name of another file to be appended. You can append as many files as you like. When you don't want to append any more files, just press RETURN. The program finally asks for a filename to use in saving the composite file.

To copy a music file from one disk to another disk, use Merge to load the file you want to move, do not append any other files, and then save the file to the new disk. This is more convenient than using the Editor to copy files.

Extract. "Extract" (filename EXTRACT.SID) loads a music file, asks for starting and ending measure numbers, extracts the measures in the specified range, and saves them as a new file. The original file retains all its measures. Extract does not remove measures from the original—it only pulls out specified measures and places them in a new file.

When you run the program, the first prompt asks for the name of the file to load. After you enter the filename, the program loads the song.

The program requests a starting measure number. Type the number of the first measure that should be extracted. If you want the extraction to start at the beginning of the file instead of at a later measure, just press RETURN.

When prompted for the ending measure number, type the number of the measure at which the extraction should stop. For example, if you want to extract measures 26–50, inclusive, the ending measure should be 51. Press RETURN by itself if you want the extraction to go to the end of the voice.

The program displays the message PROCESSING VOICE 1 while it searches for the starting measure marker. If the measure marker is not found in the voice, the program stops with the message ERROR:STARTING MEASURE NOT FOUND.

Having found the starting measure marker, the program continues to search until it comes to the ending measure marker or the end of the voice. If the searching reaches the end of the voice, the program prints the warning NOTE:RAN TO END OF VOICE.

The program repeats this procedure for voices 2 and 3.

Finally, the program asks for the filename to use in saving the new music file.

One use of the Extract program is to break a large music file into parts. Another application is to copy a sequence of notes in a song to the end of the song. For example, a song may repeat a chorus, but the chorus might be played a little differently the second time. Repeats or phrases cannot be used if there are minor changes when the notes are repeated. Instead of entering all the notes in the chorus again, just extract the notes which form the chorus, use Merge to merge them to the end of the song, and make the necessary changes.

Cross File Merge. The last utility is a cross file merger (CROSS FILE MERGE). This program lets you construct a new music file from the voices of different music files. Voice 1 of the new file can come from one file, voice 2 from another file, and voice 3 from yet another.

The program begins by constructing voice 1. It asks for the name of the file which contains the voice to be used for voice 1. The program then asks which of the three voices in that file is to be used. For instance, if the first voice of the file being created is to come from voice 3 of the song COMMODORE.MUS, here's how you would respond:

```
FILE FOR VOICE 1 ? COMMODORE
USE WHICH VOICE ? 3
```

The program loads the requested voice from the specified file. This may take awhile. The procedure is then repeated for voices 2 and 3.

After all the voices have been loaded, the program requires one more filename to use when saving the new file.

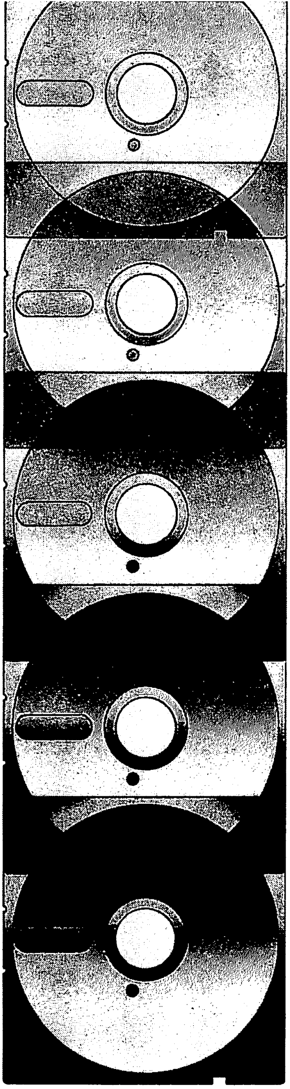
The "Cross File Merge" program can be used to change the order of the voices in a file. This is sometimes necessary if you start a song and later decide to use the synchronization mode or ring modulation. As an example, let's say that you want to switch voices 2 and 3 in the song MINUET. Here's how you would reply to the program prompts:

```
FILE FOR VOICE 1 ? MINUET
USE WHICH VOICE ? 1

FILE FOR VOICE 2 ? MINUET
USE WHICH VOICE ? 3

FILE FOR VOICE 3 ? MINUET
USE WHICH VOICE ? 2
```

Remember to use a different filename to save the new file so that you won't get a FILE EXISTS error.

More Music

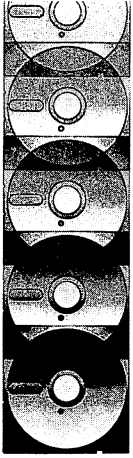
In addition to the music discussed in this chapter, your *Complete 64* disk includes some additional songs that were created using the Sidplayer Editor. Use the Sid Player program or the directory option of the Editor to see a list of all the Sidplayer files on the disk.



Chapter 6

UltraFont +

Charles Brannon

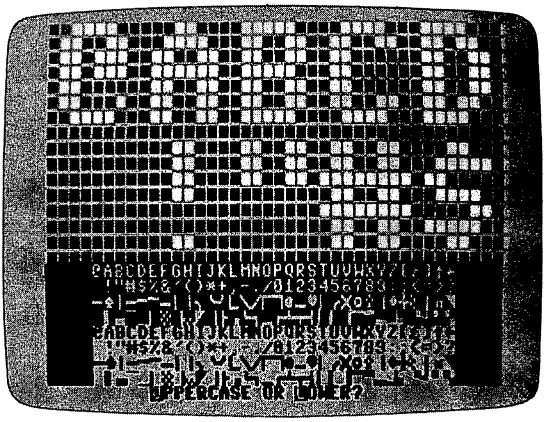


Ultrafont + Character Editor

Anyone who has used graph paper to plot out characters, then tediously converted the rows into decimal numbers can appreciate a character editor. Instead of drawing and erasing on paper, you can draw your characters freehand with a joystick. "Ultrafont +" has been written to offer almost every conceivable aid to help you design whole character sets. And any character set created using Ultrafont + can be used with *SpeedScript* (see the instruction for using special fonts with *SpeedScript* in Chapter 2).

Getting Started

The easiest way to load Ultrafont is to select it from the menu. But if you prefer you can enter LOAD "UF",8,1" and press RETURN. Once the READY prompt appears, type NEW, press RETURN and then type SYS 49152, and press RETURN.



```
@ABCDEFGHIJKLMNOPQRSTUVWXYZ[£]↑←
 !"#$%&'()*+,-./0123456789:;<=>?
@ABCDEFGHIJKLMNOPQRSTUVWXYZ[£]↑←
 !"#$%&'()*+,-./0123456789:;<=>?
UPPERCASE OR LOWER?
```

The Ultrafont + edit screen. Press L to edit lowercase characters or U to edit the uppercase/graphics characters.

The Display

At the bottom of the screen are eight lines of characters. These are the 256 characters you can customize, arranged in eight 32-character rows. A flashing square rests on the *at* sign (@), the home position of the character set. Above the eight rows is the main grid, a blown-up view of ten characters. The bottom row of the screen is reserved for messages. The first time you SYS to Ultrafont +, you'll be asked whether you want to edit the uppercase/graphics character set or the lowercase set.

About the Grid

The grid is like a large window on the character set. You see the first five characters and the five beneath them. A large red cursor shows you which character you're currently editing, and a smaller flashing square is the cursor you use to set and clear pixels in order to draw a character.

Moving Around

You can use the cursor keys (up, down, left, right) to move the large red cursor to any character you want to edit. If you move to a character not on the large grid (out of the window), the window automatically scrolls to make the character appear. You can also look at the bottom of the screen to move the larger cursor, since the flashing square on the character set moves with the main grid.

The CLR/HOME key moves the small cursor to the upper-left corner of the screen. If you press it twice, it takes you back to the top of the character set—to @.

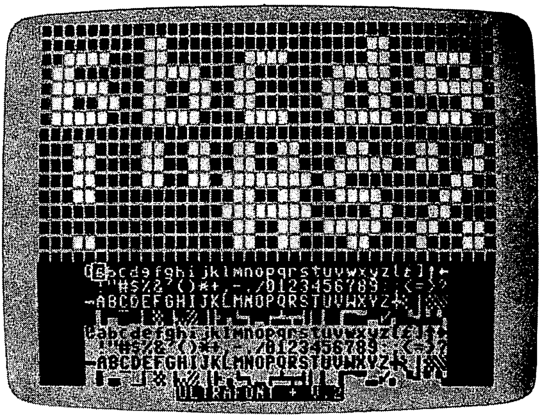
A joystick plugged into port 2 moves the small cursor within the grid. If you move the cursor out of the current character, the red cursor jumps to the next character in whatever direction you want to move. The display at the bottom adjusts, and the grid scrolls as necessary. This means that you can ignore the traditional boundaries between characters and draw shapes as big as the entire character set—256 × 64 pixels (a pixel is a picture element, or dot). You still edit one character at a time or make a shape within a 2 × 2 box of characters. There is no wraparound for the cursor in the bottom section of the screen. When it hits an edge, it will go no farther in that direction.

The joystick's fire button is used to set and clear points. If you press it when the cursor is resting on a solid square, the pixel is turned off. If the square is currently off, it's turned on. Holding down the button while you move the joystick keeps you in the same drawing mode. If you set a point, you will continue to draw as you move. If you clear a point, you can move around and erase points all over the screen.

If the drawing cursor is too fast or too slow, just press V to set the cursor velocity. Answer the prompt with a speed from 0 (slow) through 9 (too fast for practical use).

Manipulations

There are several functions that affect the current character (where the red box is). You can rotate, shift, mirror, reverse, erase, replace, and copy characters. The best way to learn is to play with the functions. It's really a lot of fun. The following keys control each function.



Just press the M key to create a mirror image of a character.

Function Keys

f1 Scrolls character right. All pixels move right. The rightmost column of pixels wraps around to the left.

f2 Scrolls character left. Wraparound is like f1.

f3 Scrolls character down. All pixels move down. The last row of pixels wraps around to the top.

f4 Scrolls character up. Wraparound is like f3.

R Rotate. Rotates the character 90 degrees. Press twice to flip the character upside down.

M Mirror. Creates a mirror image of the character, left to right.

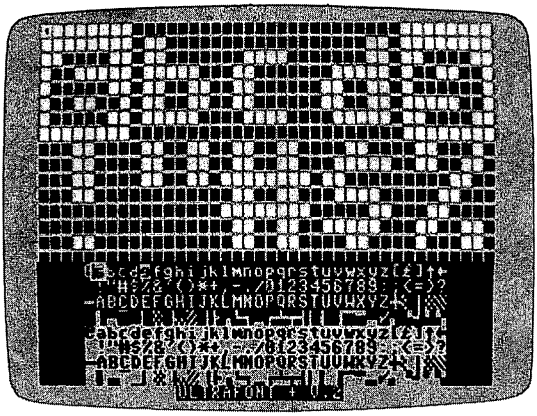
CLR (SHIFT-CLR/HOME) Erases the current character.

CTRL-R or **CTRL-9** Reverses the character. All set dots are clear, and all empty dots are set. The bottom half of the character set is the reversed image of the top half.

CTRL-back arrow (←) Copies upper half of the character set, reverses it, and places it in the lower half. This way, you have to redraw only the normal characters, then use CTRL–back arrow to create the reverse set.

F Fix. Use this if you want to restore the normal pattern for the character. If you've redefined *A* and press F while the red cursor is on the character, the Commodore pattern for *A* will be copied back from ROM.

T Type. This lets you try out your character set. The screen clears, with a copy of the character set provided for reference. You can type and move the cursor around, just as in BASIC. This is handy for envisioning sample screens and fitting together multiple-character shapes. Press the RUN/STOP key to exit from Type and return to Ultrafont +.



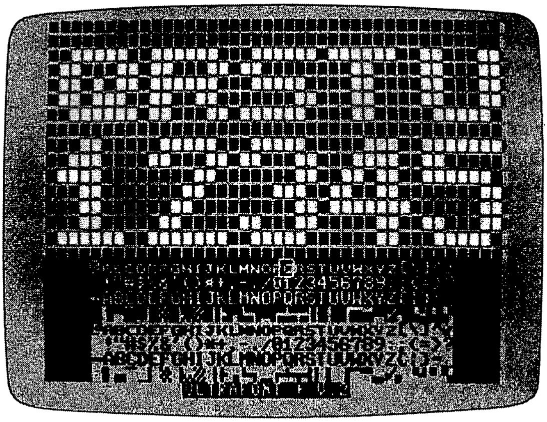
Rotating characters is easy; just press the R key.

Saving and Loading Character Sets

To save your creation to disk, press S, then D for disk. When requested, enter the filename, up to 16 characters. Don't use the 0: prefix if you're using a disk drive (it's added for you). The screen clears, displays the appropriate messages, and then returns to the editing screen if there are no errors. If there *are* errors, such as the disk being full, Ultrafont + will read the disk error message and display it at the bottom of the screen.

Press a key after you've read the message and try to correct the cause of the error before you save again.

To load a character set previously saved, press L and answer the TAPE OR DISK message. Enter the filename. After the LOAD, you'll be returned to the editing screen; a glance is all it takes to see that the set is loaded. Once again, if you're using disk, the error message will be displayed. Press a key to return to editing so that you can try again.



The Smallcaps set.

Copying and Moving Characters

You can copy one character to another with function keys f7 and f8. When you press f7, the current character flashes briefly, then is copied into a buffer. Ultrafont + remembers that character pattern. You can position the cursor where you want to copy the character before pressing f8. The memorized character replaces the character the cursor is resting on. You can also use the buffer as a fail-safe device. Before you begin to edit a character you've already worked on, press f7 to store it safely away. That way, if you accidentally wipe it out or otherwise garble the character, you can press f8 to bring back your earlier version.

Creating DATA Statements

A very useful command, CTRL-D, allows you to create DATA statements for whatever characters you've defined. Ultrafont + doesn't make DATA statements for all the characters, just the ones you've changed. After you press CTRL-D, Ultrafont + adds the DATA statements to the end of whatever program you have in BASIC memory. If there is no program, the DATA statements exist alone.

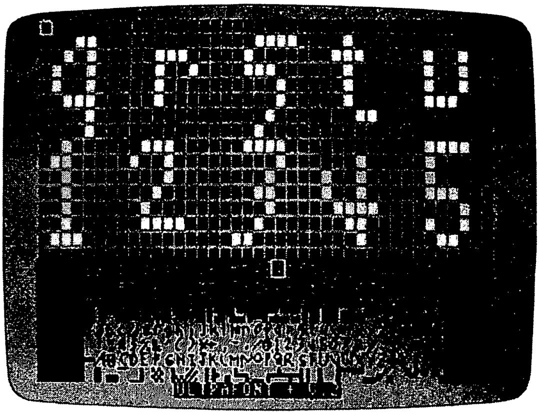
You can load Ultrafont +, enter NEW to reset some BASIC pointers, load a program you're working on, then SYS 49152 to Ultrafont + to add DATA to the end of the program. The DATA statements always start at line 63000, so you may want to renumber them. If you press CTRL-D twice, another set of DATA statements will be appended, also numbered from line numbers 63000 and up. Since the keys repeat if held down, just tap CTRL-D. If you hold it down, you may find that a hundred DATA statements have been created. See the notes at the end of this article for more details on using DATA statements in your own programs.

Exiting Ultrafont +

After you create the DATA, you'll still be in Ultrafont +. If you want to exit to see the DATA statements or go on to other things, press CTRL-X. The screen will reset to the normal colors and you'll see the READY prompt. If you've made DATA, a LIST dramatically reveals it. It's best to enter the command CLR to make sure BASIC is initialized properly after creating DATA statements. One thing to watch out for: *Don't use RUN/STOP-RESTORE to exit Ultrafont +*. The program moves screen memory from the default area at address 1024, and the RUN/STOP–RESTORE combination does not reset the operating system pointers to screen memory. If you do press it, you won't be able to see what you're typing. To fix it, blindly type POKE 648,4 or SYS 49152 to reenter Ultrafont + so that you can exit properly.

Reentering Ultrafont +

To restart Ultrafont + within the program, press SHIFT–RUN/STOP. After you've exited to BASIC, you can rerun Ultrafont + with SYS 49152. You'll see the character set you were working on previously, along with the message USE ROM SET? (Y/N). Usually, Ultrafont + will copy the ROM character patterns into RAM where you can change them. If you press N, however, the set you were previously working on is untouched. Press any other key, like RETURN, to reset the characters to the ROM standard. You can copy either the uppercase/graphics set from ROM or the lowercase set.



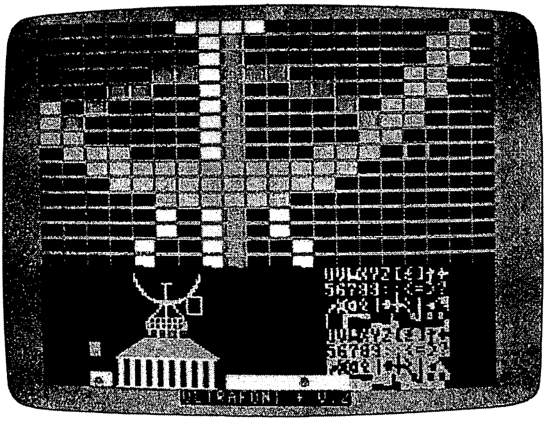
The Scrawl set.

A Whole New World of Multicolor

You're not finished yet. There's yet another mode of operation within Ultrafont +, the multicolor mode. In multicolor mode, any character can contain up to four colors (one has to be used for the background) simultaneously. Multicolor changes the way the computer interprets character patterns. Instead of a one bit representing a solid pixel and a zero representing a blank, the eight bits are organized as four *pairs* of bits. Each pair can represent four possibilities: 00, 01, 10, and 11. Each pair is also a number in decimal from 0 through 3, and represents one of the four colors.

Ultrafont + makes multicolor easy. You don't have to keep track of bit pairs any more than you have to convert binary to decimal. Just press the f5 key. Presto—the whole screen changes. The normal characters are rather unrecognizable, and the drawing cursor is twice as wide (since eight bits have been reduced to four pixel pairs, making each dot twice as wide). You have only four dots horizontally per character, but you can easily combine several characters to form larger shapes.

Multicolor redefines the way the joystick and fire button work. The fire button always lays down a colored rectangle in the color you're currently working with. That color is shown in the center of the drawing cursor. Press a number key 1, 2, 3, or 4 to choose different colors to draw with. The number of the key is one more than the bit pattern, so color 1 is bit pattern 00, and color 4 is bit pattern 11. When you first SYS to Ultrafont +, the four colors show up distinctly on a color TV or monitor.



The multicolor Dish.set. Using the multicolor mode can help you design figures for games.

You can easily change the colors. Just hold down SHIFT and press the appropriate number key to change that number's color. You will see the message PRESS COLOR KEY. Now press one of the color keys from CTRL-1 through CTRL-8, or from Commodore-1 through Commodore-8. Hold down the CTRL or Commodore key as you do this. Instantly, that color, and everything previously drawn in that color, is changed.

Three of the colors (including 1, the background color) can be any of the 16 colors. But because of the way multicolor works, color 4 (represented by bit pattern 11, or 3 in decimal) can only be one of the eight CONTROL colors. Assigning it one of the Commodore logo colors just picks the color shown on the face of the color key. Incidentally, it's the color of bit pattern 3 (color 4) that changes according to the character color as set in color memory. The other colors are programmed in multicolor registers 1 and 2 (locations 53282 and 53283), so all characters share these two colors. When you want to vary a certain color without affecting the rest of the characters, you'll want to draw it in color 4.

Some of the commands in the multicolor mode aren't as useful as others. You have to press f1 and f2 twice to shift a character, since they only shift one bit, which causes all the colors to change. You can use CTRL-R or CTRL-9 (Reverse) to reverse all the colors (color 1 becomes color 4, color 2 becomes color 3, color 3 becomes color 2, and color 4 becomes color 1). R (Rotate) changes all the colors and is rather useless unless you press it twice just to turn the characters upside down. M (Mirror) works as it did before except that colors 2 and 3 are switched. And you can still copy characters by using f7 and f8 (see above).

Two multicolor character sets are included on the *Complete 64* disk: Alien.set and Dish.set.

Returning to Normal

You can switch instantly back to the normal character mode by pressing f6. If you were drawing in multicolor, you can see the bit patterns that make up each color. Multicolor characters look just as strange in normal mode as normal characters look in multicolor.

If you changed colors in the multicolor mode, some of the colors in the normal mode may have been altered. You can change these colors just as you did in multicolor mode. Press SHIFT-1 to change the color of the empty pixels and SHIFT-2 to change the color of the *on* pixels. Use SHIFT-4 to change the color of the eight rows of characters.

Notes: Using the DATA Statements

The DATA statements are created from lines 63000 and up, as many as necessary. Each line of data has nine numbers. The first number is the internal code of the character (the code you use when POKEing to the screen). It represents an offset into the table of character patterns. The eight bytes that follow are the decimal numbers for the eight bytes needed to define any character. Here's a sample program to read them and display them:

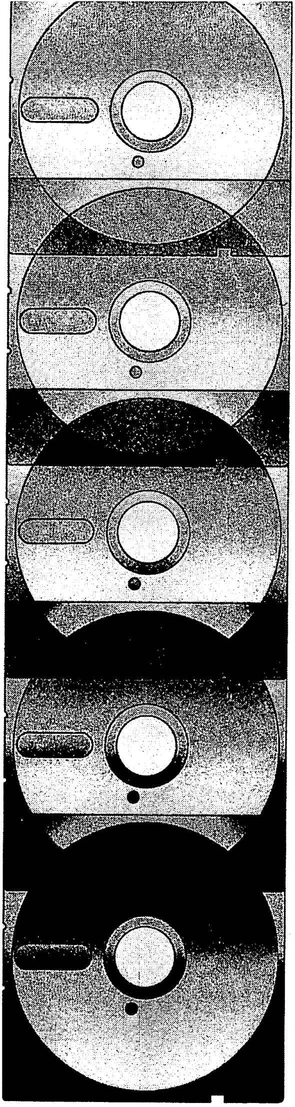
```
10 POKE 56,48 : CLR
50 READ A : IF A=-1 THEN 70
60 FOR I=0 TO 7 : READ B
63 POKE 12288 + A * 8 + I, B
66 NEXT : GOTO 50
70 PRINT CHR$(147);"{10 DOWN}" : REM TEN CURSOR
   DOWNS
80 FOR I = 0 TO 7 : FOR J = 0 TO 31
83 POKE 1028 + J + I * 40, I * 32 + J
86 POKE 55300 + J + I * 40, 1 : NEXT : NEXT
90 POKE 53272, (PEEK (53272) AND 240) OR 12
95 END
```

You'll also need to add the following line to the end of your DATA statements:

```
63999 DATA -1
```

Quick Reference: Ultrafont + Commands

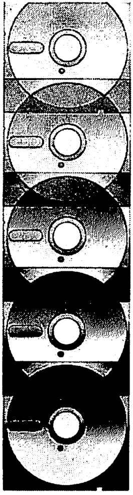
Cursor keys	Move to next character.
HOME (CLR/HOME)	Moves the cursor to upper-left corner. Press twice to go back to start.
V	Cursor velocity; answer from 0 (slow) through 9 (fast).
f1	Scrolls right with wraparound.
f2 (SHIFT-f1)	Scrolls left.
f3	Scrolls down.
f4 (SHIFT-f3)	Scrolls up.
R	Rotates 90 degrees; press twice to invert.
M	Mirror image.
CLR (SHIFT-CLR/HOME)	Erases current character.
CTRL-R, CTRL-9	Reverse pixels.
CTRL-back arrow (←), CTRL-F	Copy first four rows of characters, reversed, to bottom four.
F	Fix characters from ROM pattern.
L	Load. Tape or Disk, Filename.
S	Save; Tape or Disk, Filename.
T	Typing mode: RUN/STOP to exit.
f7	Memorizes character (keep).
f8 (SHIFT-f7)	Recalls character (put).
f5	Switches to multicolor character mode.
f6 (SHIFT-f5)	Returns to normal character mode.
CTRL-D	Makes DATA statements.
SHIFT-RUN/STOP	Restarts Ultrafont +.
CTRL-X	Exits Ultrafont + to BASIC.



Chapter 7

High-Resolution Sketchpad

Chris Metcalf



High-Resolution Sketchpad

The magic words *high resolution* were part of what prompted me to buy a Commodore 64. No doubt you too were influenced by the idea of having a 320 × 200 dot map of picture elements on the screen, a total of 16 colors to be spread about on the screen, and the ability to mix up to four colors within each 8 × 8 pixel area.

Unfortunately, it is very difficult to employ these powerful features. The Commodore 64 lacks BASIC commands for high resolution, but does have a pair of high-resolution bitmapping modes with great potential. The only difficulty is in accessing them from BASIC.

BASIC provides only minimal control over the graphics. A series of POKEs are needed even to bring up the high-resolution graphics screen, then further POKEs are needed to clear the graphics page for use. Once this has been accomplished, more POKEs are necessary to plot points on the screen and set their colors. This process is slow, tedious, and difficult.

High-Resolution Graphics

The Commodore 64 permits a bitmapping screen to be located at any of eight 8K areas in memory. The sketchpad program uses locations 40960–48959 ($A000–$BF3F) for this screen. The color data is stored elsewhere in memory. In the standard high-resolution mode, color can come from any 1K block in the same 16K area of memory as the bitmap screen. Sketchpad uses the area from 35840 through 36839 ($8C00–$8FE7) for this floating color memory. In multicolor bitmap mode, further memory is needed to support the additional colors, and this color memory is fixed at locations 55296–56296 ($D800–$DB87).

On the 64, the high-resolution screen resembles 1000 programmable characters in its format. The first byte of the screen defines the eight pixels at the beginning of the top line. The following seven bytes define the first eight pixels of each following line. However, the next group of eight bytes is located not below but to the right of the initial eight. After 40 groups of eight bytes (the equivalent of a line of programmable characters), the sequence repeats for the next eight pixel lines.

In standard high-resolution mode, both background color and pixel color are defined by the selectable 1K of color memory. The most significant nybble (four bits, half a byte) defines the color of all the pixels within one 8 × 8 pixel group (one "character"). The least significant nybble defines the background color in the same area (seen when a bit is zero).

Multicolor mode allows multiple colors within one 8 × 8 pixel area by assigning one of four colors to each possible combination of two bits. However, the result is that it takes both bits to define a single pixel. Each bit pair takes its color from the corresponding byte on the floating color screen, the fixed screen, or the background color register as follows:

Bit pair	Source of its color code
00	Background color register (53280,$D021)
01	High nybble of floating color memory
10	Low nybble of floating color memory
11	Fixed color memory

As in standard high-resolution mode, the color memory provides separate color information for each group of 8 × 8 pixels. However, unlike standard high-resolution mode, all 00 bits are set from the register at 53281.

However, "High-Resolution Sketchpad" allows you to ignore most of these details. You should, however, understand why you cannot plot too many colors together. New colors simply change the color of all the appropriate pixels within each 8 × 8 area.

Loading Sketchpad

You can load Sketchpad in one of three ways. The simpliest method is to select High-Resolution Sketchpad from the menu. Using the menu saves you the trouble of remembering the proper commands needed—the menu does all the work.

If you prefer not to use the menu program, you can load and run the Sketchpad loader program by entering

```
LOAD"SKETCH.LOAD",8
```

and pressing RETURN. Then type **RUN** and press RETURN.

The third method involves a number of steps. First, enter this line and press RETURN:

```
LOAD "SKETCH.OBJ",8,1
```

Once the READY prompt returns, enter **NEW** and press RETURN. Finally enter this line and press RETURN:

SYS 36864

Getting Started

Once Sketchpad has been loaded and run, you'll be greeted with the following:

HIRES SKETCHPAD – BY CHRIS METCALF
MULTICOLOR MODE? N

The cursor will blink on the N. At this point, enter either Y or N to determine whether you will use standard or multicolor bitmapping during the program run. Standard mode provides better resolution for more intricate designs, while multicolor is more useful for less detailed or more colorful displays.

Simple Graphics with the Sketchpad

Once you press RETURN, the bitmap screen should come up. A small turtle sprite in the center of the screen indicates where you are plotting. The first time you enter the program after turning on the computer, the display will be covered with random pixels and colors. *Press SHIFT–CLR/HOME to clear out the screen.* At any time you may press CTRL–left arrow (←) to leave the program.

The program has been designed so that either joystick or keyboard can control the turtle plotter. Joystick users can move the turtle with the joystick in control port 2, and can control various modes with the fire button. However, a number of keys have been defined for moving the turtle as well. The square of keys with Q, E, Z, and C as its corner points will steer the turtle in all eight directions.

	up/left	**up**	**up/right**	
	Q	**W**	**E**	
left	**A**		**D**	**right**
	Z	**X**	**C**	
	down/left	**down**	**down/right**	

The S key at the center of the square is used to return the turtle to its starting position at the center of the screen, and the HOME key puts the turtle at the top-left corner of the screen.

THe first thing to experiment with is simple plotting.

Press the space bar (or SHIFT–space bar) or the fire button on the joystick to enter plot mode. A dot will appear in the center of the turtle. Now you will draw a line wherever you go. To stop plotting and just move about, hit the space bar or the fire button again.

When you first enter the program, the turtle will be the same color that you were typing in before (the character color). To change this color, use the CTRL or Commodore key with the numbers 1–8. The CTRL colors are shown on the front of the numeric keys. The color you are using is plotted with every point you plot; therefore, if you try to plot in an 8 × 8 pixel area previously in a different color, the color of the pixel area will change to your plot color.

Multicolor Mode

This problem can be reduced by using multicolor mode. When you begin the program, enter Y for multicolor. You will see that the pixels you plot are in fact larger. However, you can now intermix colors freely. Each of the three types of plotting (bit pairs 01, 10, and 11) and the erase mode (00) are represented by the function keys. The f1 key corresponds to 11, f3 to 10, and f5 to 01. When the program begins, you begin in f5. While using any one plot type, you are constrained by the same problem with colors that affects standard bitmap mode. However, the coloring of each of the three types is completely independent, so by changing among f1, f3, and f5, you can plot without affecting the colors in different plot types. The f7 key will put you in erase mode.

In standard bitmap mode, the same function keys can be used. In either mode, the plus and minus keys (and their shifted equivalents), which correspond to f5 and f7, respectively, can also be used. Normal plotting in standard hi-res is in f5 mode, the mode you begin in. The f1 key has been set to yield f5 when pressed in standard bitmap mode. The f7 mode has the same effect as in multicolor—it erases pixels without affecting the color of neighboring pixels. The f3 mode does not plot any pixels—instead, it changes the background color within each 8 × 8 square, without altering the pixel plot colors. Plotting in this mode can be a good way to familiarize yourself with the 8 × 8 pixel color setup.

Special Features

Changing the border and background colors can also be done from within the program. However, if you are in standard bitmap mode, the bitmap background color will not change until you press SHIFT–CLR/HOME. Border and background colors are changed with the joystick or the direction keys. To enter the color change mode, press the up-arrow (↑) key. Moving the joystick left and right or using the corresponding keys on the keyboard will change the border color. To change the background color (this will be immediately apparent only in multicolor mode), move the joystick up and down, or use the keys. To break out of this mode, press the fire button or any key other than those in the direction keys, and you will return to the main loop.

Moving by steps is another feature of this program. When you begin the program, you move one pixel at a time. However, whenever you press a number key or its shifted equivalent, you will begin to move that many pixels at a time. For example, if you want to do double-spaced plotting, press the 2 key; to move eight pixels at a time, press the 8 key. The same feature works in multicolor mode, but, because of the double-width pixels, odd numbers give somewhat peculiar results.

More Advanced Graphics Modes

More powerful options are available with the shifted function keys. The first option, known as the draw-from mode, is turned on and off with f2. When you press f2, the start point for the line-draw routine is assigned to your location. Now, as you move around, you will see a line connecting your turtle to the start point you have selected. This rubber-band line does not change the pixels around it. However, it does change the colors if you are in any of the plotting modes. Only in f7 or minus mode will no colors be plotted to the rubber-band line as you move about. Once the line is in a position that suits you, press the SHIFT key or the fire button, and a real line will be drawn in the color and plot mode you are using.

As you continue to move about and draw lines, the start point will remain where you initially assigned it. This allows you to create intricate abstract works simply by setting a spacing of three or four (or whatever you like), and moving around while holding down the SHIFT key or the fire button.

The SHIFT LOCK key can also be used. However, the space bar will still toggle on and off the simple plotting beneath the turtle. To terminate the draw-from mode, hit f2 again. Then, to assign your position as the new start point, press f2 yet again. Note that since the SHIFT logo will draw a line, it is often helpful to use the Commodore logo key for normally shifted characters (for example, SHIFT–CLR/HOME, SHIFT-f8) since it yields the same results.

The second mode is selected with the f4 key. This is the draw-to mode, which is very similar to draw-from. However, in this mode, every time you press the fire button or the SHIFT key, the line is drawn and the line-draw start point is automatically assigned to your current position. This provides the same effect as the Atari DRAWTO command. The draw-to mode allows you to draw figures more easily. Note that if you are in f4 mode and select f2, f4 will be canceled and replaced by f2. The reverse is also true.

The third line-draw mode (f6) is useful primarily for making shaded figures. When you're in this mode, every time you press the SHIFT key or fire button, a line will be drawn to the right in the mode and color you are in until it encounters another pixel or the right-hand edge. This mode has no rubber-band effect.

You can also select where the draw-right will stop. Normally, the line will stop when it encounters a pixel in the same mode, so it would erase a filled-in area to the right or draw right in f3 mode until it encountered an f3 bit pair, and so forth, depending on your mode. However, the asterisk (*) will toggle a variation. If you press the asterisk key once after beginning the program, the draw mode will search for any *on* pixel. Thus, you can draw right in f1 mode and stop at an f3-mode pixel, creating a border of a different color. Pressing this key again will return you to the initial fill-to-same mode.

Fill

One of the most powerful features of the program is called when you press the f8 key. This key activates a fill. This function will fill in any area bounded by pixels or fill to the edges of the screen. This feature is also dependent on the asterisk to know what to fill to. Normally, it fills to any pixel of the same type, but it can be toggled to fill to any on pixel, thus allowing differently colored borders in multicolor mode. The fill can and will escape from any shape in which there is a hole in the border, but it does not slip between diagonally separated pixels.

The Status Line

All of these modes are somewhat difficult to hold in mind. What with four plot modes, a plot/no-plot option, three kinds of lines, and a fill type (asterisk) toggle, things can get confused. This is especially true since fill-right has no rubber band, since plot-minus and no-plot appear the same, and since the multicolor plots are indistinguishable when in the same color. To help keep them all straight, a status line can be toggled on at the bottom of the screen by pressing and holding down the RETURN key.

The status line consists of four parts. The first indicates the mode you are in (f1, f3, f5, or f7). The second indicates whether your plotting is on or off (plotting or just moving about). The third displays the type of line-draw mode you are in (OFF, FROM, TO, or LINE), and the fourth tells the status of the asterisk mode (SAME is what you begin in; ANY means stop filling at any on pixel).

Input/Output for Sketchpad

The program is provided with a feature for loading and saving all the data that makes up the hi-res image. To access this feature, press the @ *(at)* key. The program will ask whether you wish to Load or Save (note that only the first letter is significant). Any other answer will abort the process. Then you must specify the device number. The Datassette is 1, and disk drives can be either 8 (as most are) or 9. (Device 2, the RS-232 channel, can also be used, but modifications to the machine language will be necessary to include sending baud rate and other parameters.) No other devices are permitted. Finally, you will have to provide the name. If no name is given, the process will terminate. Now the turtle sprite will disappear for the duration of the Load or Save. When the process is finished, the turtle will return.

Disk input/output is simple. Specify L or S, 8, and the name. Make sure a disk is in the drive, and, most importantly, turned on and plugged in. *If the drive is not ready, the program will lock up.* In this case, RUN/STOP–RESTORE is all that can recover the program. No suffixes are necessary for disk Saves or Loads, but any prefixes you wish (such as "0:" or "@0:") will have to be included in the name. When the disk drive is finished, the error channel is read and displayed for about two seconds. Normally, a "00,OK,00,00" is returned. Here are some common errors:

62, FILE NOT FOUND—Loading a nonexistent file.
63, FILE EXISTS—Save under another name or with "@0:".
64, FILE TYPE MISMATCH—Saving with "@0:" over a program file.
72, DISK FULL—Get a new disk or scratch some files.
74, DRIVE NOT READY—The drive door is open; save with "0:".

Any other error (particularly 21) indicates a disk malfunction of some sort. Refer to your 1541 manual.

The high-resolution information is saved in a unique format. The first two bytes saved are the border and background colors. This is followed by the floating color screen data (1000 bytes), the fixed color screen data (another 1000 bytes), and, finally, the high-resolution screen. The screen is saved by a data-compaction technique. All nonzero bytes are output normally, but a zero flags a special mode: The next two bytes are the address of the following nonzero byte in low-byte/high-byte format. This allows the program to clear the intervening space quickly and load only the relevant picture data.

Sample Pictures

Included on the *Complete 64* disk are five sample pictures. Two of these pictures were created in the multicolor mode of Sketchpad: HOUSE1.MLT and HOUSE2.MLT. The other pictures are SNOWFLAKE.HIR, PATTERN.HIR, and GATE.HIR.

Load/Save Subroutine

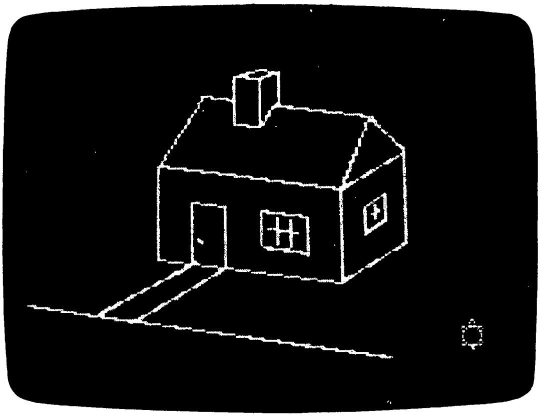
Also on the disk, but not appearing on the menu, is the file LOAD-SAVE SUB. This program is a subroutine to allow you to integrate Sketchpad designs into your own programs. The subroutine comes in three main parts: the data loader, the subroutine itself (at line 50000), and the machine language data. The data loader goes at the beginning of your program and simply reads the DATA statements into memory from 51676 through 51967 ($C9DC–$CAFF). The subroutine processes your request and calls the machine language.

To use the subroutine, load LS with either load or save (load=0, save=1), DV with the device (8 for disk), and NM$ with the name of the file. Then GOSUB 50000. The BASIC subroutine is not, however, necessary; the machine language can be called on its own. To do so, POKE 2 with 0 for load or 1 for save. Then OPEN the appropriate type of file:

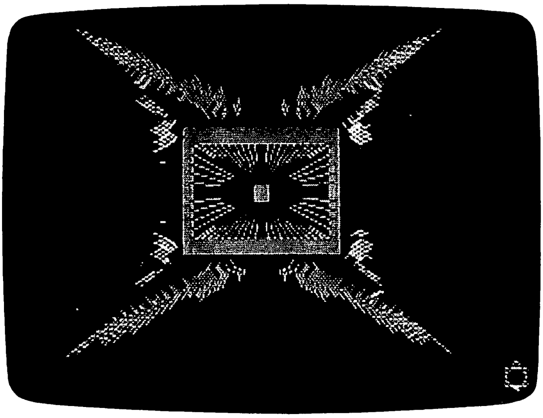
disk load: OPEN 1,8,2,"filename"
disk save: OPEN 1,8,2,"filename, S,W"

Finally, SYS 51676. For example, to load a picture ("DESIGN3") from disk:

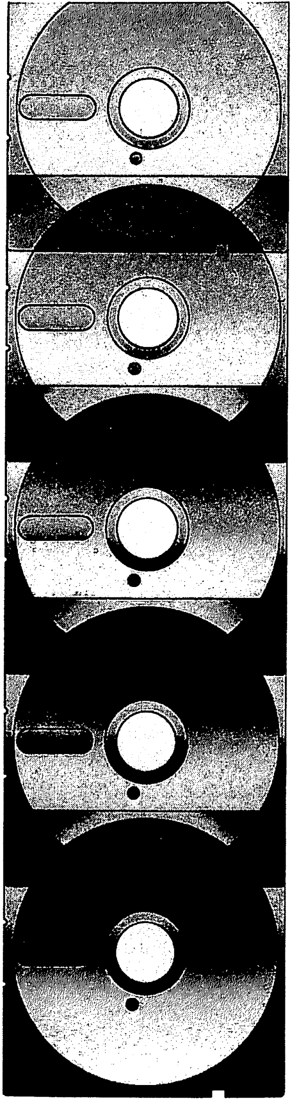
POKE 2,0: OPEN 1,8,2,"DESIGN3":SYS 51676



A simple house created in multicolor mode of Sketchpad. It can be loaded from disk; filename HOUSE1.MLT.



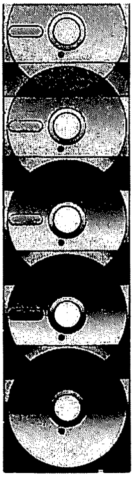
With some practice, creating interesting designs becomes easy with Sketchpad.



Chapter 8

Plus/Term

Gregg Peele



Plus/Term

Telecommunications for Your Commodore 64

There's a lot more out there than you think.

Sure, you can use your home computer to play games, write letters, and balance your budget. But *telecomputing*—communicating with other computers over the phone lines—can let you do even more. "Plus/Term" and a modem will turn your home computer into a full-fledged communications terminal. You can link up with the personal computers of your friends, the mainframe computer at work or at school, electronic bulletin boards, online databases, and commercial information services such as CompuServe, The Source, and Dow Jones News/Retrieval.

Plus/Term is designed to make your computer emulate an *asynchronous ASCII terminal.* An asynchronous terminal does not require special timing (sync) characters and doesn't require the receiving terminal to operate in step with your system. ASCII is a standard character code that computers use to understand each other. Most microcomputer communications are asynchronous and in ASCII.

Entering Terminal Mode

Once you have your modem connected and turned on, you can either load Plus/Term from the menu or enter

LOAD "PLUS/TERM",8

and press RETURN.

When you run Plus/Term, it will load the machine language section and then ask you to type YES if you have a Mitey Mo or HESmodem model; type NO if you have some other modem.

Next, you'll need to enter a *baud rate.* The baud rate, more properly known as the *data transfer rate,* is the speed in bits per second (bps) at which a modem communicates. Inexpensive modems—including most of those sold for the Commodore 64—can transmit and receive information at speeds ranging from 110 to 300 bps (about 10 to 30 characters per second). More expensive modems can usually be switched to 1200 bps to speed up communications by a factor of four.

Remember that the computer at the other end of the phone line must be transmitting and receiving at the same speed as yours. You can't send and receive at 1200 bps if the other computer has only a 300 bps modem. Many electronic bulletin boards and commercial information services are capable of communicating at both 300 and 1200 bps. Some even recognize your baud rate when you first sign on and will adjust themselves accordingly. But you still have to set Plus/Term for the proper baud rate each time you run the program.

After you specify the baud rate, Plus/Term enters terminal mode. A blinking underline cursor will appear at the upper-left corner of the screen.

To see how terminal mode works, try typing these two sentences:

This is a test of the Plus/Term software. It is designed to provide lots of options for use with many systems.

Notice how Plus/Term handles words that are typed at the end of a line. Rather than splitting words, Plus/Term moves the entire word to the next line. This feature, known as *word-wrap,* makes text easier to read. It's found on most word processors.

You can turn word-wrap on or off at any time by calling up a menu. All the main features in Plus/Term are controlled from a main menu and its submenus. When you're in terminal mode, you can flip the screen to the main menu whenever you want by pressing the f7 special function key.

Plus/Term Main Menu

Here's what it looks like:

```
PARAMETERS
1. WORD LENGTH
2. STOP BITS
3. BAUD RATE
4. PARITY
5. DUPLEX
6. DEFINE KEY
7. TERMINAL OPTIONS
8. RETURN TO TERMINAL
9. RETURN TO BASIC
```

The first five options control communications parameters. Plus/Term defaults to these settings:

8-bit word length
1 stop bit
300 baud
No parity
Half duplex

Always set your modem to full duplex while using Plus/Term. That way you can control the duplex setting with the program. To change Plus/Term from half duplex to full duplex, press the 5 key to select option 5 from the main menu (you'll want to change to full duplex whenever calling a BBS or an information service). The menu goes away and a submenu appears. Again, press the appropriate key to select the option you want (half or full duplex); your choice will be highlighted in reverse video. Press RETURN to go back to the main menu.

All the other options in Plus/Term work the same way. Just select an option from the main menu and another menu will usually appear.

If you change baud rates from within Plus/Term, all other parameters will appear on the various submenus to have reverted to their default values. The actual parameter settings will not have changed, just the values shown in the menus. Thus, if you change the baud rate after changing other parameters, the menus will no longer reflect the actual settings of the other parameters. For this reason you should always change the baud rate before changing any other parameters. Usually, you need to set the baud rate only when first running the program.

Never press RUN/STOP–RESTORE when the menu is on the screen. If you do, the program halts. You can restore it by typing POKE 648,4 and pressing RETURN, but you'll have to type blindly because the operating system thinks the screen is at a different location.

Terminal Mode Commands

You can return to terminal mode from the main menu by selecting option 8 (option 9 exits Plus/Term to BASIC). Notice that the text you left on the screen is still there.

Other keys besides f7 execute commands in terminal mode. You must press three keys simultaneously: Hold down SHIFT and CTRL, then press the appropriate command key (this sequence makes it possible for you to send special control codes in terminal mode by pressing CTRL and a character key). Here are the terminal mode commands:

CTRL–SHIFT-B	Change background color
CTRL–SHIFT-F	Change border color
CTRL–SHIFT-K	Change text color
CTRL–SHIFT-O	Open buffer (start storing characters in memory)
CTRL–SHIFT-C	Close buffer (stop storing characters in memory)
CTRL–SHIFT-S	Save buffer contents on disk
CTRL–SHIFT-L	Load buffer contents from disk
CTRL–SHIFT-Z	Zero (erase) buffer

Notice that some of these commands involve the manipulation of a *buffer.* A buffer is an area in memory set aside to store data. If the buffer is open (CTRL–SHIFT-O), all the characters sent and received by your computer are stored in memory. The contents of the buffer can then be saved or sent to a printer. This feature lets you *download* (receive) text files and programs from remote computers. You can also *upload* (send) files to other computers. We'll discuss these procedures in a moment.

The buffer is 30,720 bytes long (about 30K). The file you plan to upload or download must fit in the buffer unless you handle it in pieces to be assembled later. If the buffer fills up when you're downloading, a screen prompt asks SAVE BUFFER, YES OR NO? *If you answer no, the contents of the buffer are erased.* Press CTRL-Q to continue your conversation with the other system.

Terminal Options

Option 7 on the main menu, TERMINAL OPTIONS, controls some of the most useful features of Plus/Term. Here's what the terminal options submenu looks like:

1. **DIRECTORY (DISK)**
2. **COMM TO COMM**
3. **COMM TO ASCII**
4. **WORD WRAP**
5. **NORMAL SCREEN**
6. **PRINT BUFFER**
7. **LINEFEED WITH RETURN**

Option 1 (DIRECTORY) simply lets you call a disk directory without leaving Plus/Term.

Options 2 and 3 (COMM TO COMM and COMM TO ASCII) toggle between each other. Option 3 is the default setting. Plus/Term normally translates Commodore codes into ASCII and vice versa. If you select option 2, Plus/Term stops converting Commodore codes to ASCII. This is useful when communicating with other Commodore systems or when transferring files, as we'll see in a moment. Option 3 also lets you save or load key redefinitions with a disk drive. After you've customized the keys as described below, select option 3 and follow the screen prompts.

Options 4 and 5 (WORD WRAP and NORMAL SCREEN) also toggle back and forth to turn word-wrap on or off.

Option 6 (PRINT BUFFER) lets you access the most powerful features of Plus/Term. You can print the contents of the buffer on the screen or a printer, or send it through the RS-232 port (which may be connected to your modem or a serial printer). Whenever you're printing the buffer, you can pause the action by pressing the SHIFT key, slow it by pressing CTRL, or stop it by pressing the space bar. Since some printers may enter graphics mode if certain character codes are sent, you may need to modify the OPEN statement in line 1900 to lock your printer into text mode.

Option 7 (LINEFEED WITH RETURN) lets you disable the linefeed character that normally accompanies the RETURN character. Normally, when you press RETURN, two things happen: The cursor jumps to the left side of the screen and also moves down a line. The downward cursor movement is a linefeed. Some remote computers automatically send a linefeed when they receive a RETURN, so the extra linefeed is unnecessary. If option 7 is highlighted, the RETURN character will include a linefeed. Selecting this option toggles linefeeds on and off.

Customizing Plus/Term

We've already mentioned key redefinition briefly. This is a feature usually found only on the better terminal programs, and it requires some explanation. Basically, it lets you customize Plus/Term for communicating with a specific remote computer. To use this feature, select option 6 on the main menu.

Here's why it's important: When Plus/Term is in normal ASCII mode, all the characters you type are translated into the standard ASCII codes before they are sent over the phone line. This assumes that the other computer also is sending and receiving the same ASCII codes. But some computers occasionally depart from ASCII. For instance, some systems use ASCII code 127 as a delete character, while pressing the INST/DEL key on a Commodore 64 generates ASCII code 20. The result will be a failure to communicate.

Plus/Term lets you redefine any key on the keyboard to send out any ASCII code you want. To redefine the INST/DEL key to send the ASCII code 127 that the other computer expects instead of the ASCII code 20, you'd first select option 6 from the main menu. The following submenu appears:

CHANGE VALUES
1. GOING OUT
2. COMING IN

Since you want to change the value you're sending out, select option 1. A screen prompt asks you to press the key you want to redefine; press INST/DEL. Plus/Term tells you that the key currently sends an ASCII 20 and asks you to type in the code you want. Type 127 and press RETURN. The main menu reappears.

It's that easy. Now the INST/DEL key sends an ASCII 127 instead of 20.

Two-Way Translating

You're only half done, though. When Plus/Term sends the 127 over the phone line, the other computer will accept it as a delete key, all right. But then the remote computer echoes the code back to *your* computer (we won't get into the technical reasons). Plus/Term knows that it's supposed to send a 127 instead of a 20, but it doesn't know how to translate the 127 coming back into the 20 that your computer recognizes as a delete key. Instead of deleting characters on your screen, pressing INST/DEL would make back arrows appear.

The solution, as you may have guessed, is to customize Plus/Term further so that it translates the key in both directions. Select option 6 from the main menu again, then choose option 2 on the submenu (COMING IN). Now you can match the incoming code with the appropriate Commodore code.

When the program asks you which code you want to change, type 127. When it asks you for the new value, type 20. Pressing RETURN brings you back to the main menu.

Don't forget that the function keys (except for f7, which calls the main menu) can also be redefined. If you want to save the new definitions so that you don't have to repeat the process each time you run Plus/Term, select option 3 from the terminal options menu as described above.

Uploading and Downloading

As we mentioned, one of Plus/Term's most powerful features is the ability to exchange files with other computers.

Downloading is as simple as opening the buffer (CTRL–SHIFT-O in terminal mode), clearing it out if necessary (CTRL–SHIFT-Z), and closing it when you've received everything you want (CTRL–SHIFT-C). Then you can save the buffer by pressing CTRL–SHIFT-S, or print it out from the terminal options menu.

Frequently, when you attempt to download a program from a bulletin board or commercial information service, you will be asked whether you wish to use an *error-checking protocol.* You should specify that no protocol is to be used, since Plus/Term does not support error checking when downloading or uploading; it simply sends and receives files as continuous streams of text. Unless you have a particularly bad phone connection, this will usually be sufficient to transfer the desired information successfully.

To upload a file, reverse the process. Clear the buffer if necessary by pressing CTRL–SHIFT-Z; load the file you wish to transmit by pressing CTRL–SHIFT-L; go to the terminal options menu and select option 6 (PRINT BUFFER); and send the file to the RS-232 port, where your modem is connected.

However, there are some complications—imposed by the computer, not the program. Transferring text files is easy. Usually, they're already stored in ASCII format by the word processor, and they can be loaded into another word processor after the transfer is complete. But BASIC program files present a problem.

Most computers, including Commodores, can store programs in two formats: ASCII and *tokenized.* Tokenized files are abbreviated versions of ASCII files. A program must be tokenized before it will run on a 64. Unfortunately, the 64 lacks a command to load an ASCII file back into the computer and convert it to a tokenized file. After you transfer a program you won't be able to run it.

The Tokenizer Solution

To overcome this limitation, we've included a short tokenizer utility (Program 3) which converts ASCII files into tokenized files. To convert a tokenized BASIC program file into an ASCII file for uploading, use the following procedure. First, before running Plus/Term, load the program you wish to transfer into the computer. Then type the following and press RETURN:

OPEN 8,8,8,"0:*filename*,P,W":CMD8:LIST

(Replace *filename* with a filename that is different from the BASIC program that you just loaded.)

When the cursor returns, type the following and press RETURN:

PRINT#8:CLOSE8

The program has now been converted into a Commodore ASCII file which can be uploaded.

Sending a File

Now follow these steps:

1. Load and run Plus/Term and enter terminal mode.
2. Zero (erase) the buffer by pressing CTRL–SHIFT-Z.
3. Load your file into the buffer (press CTRL–SHIFT-L). A screen prompt will ask you to enter the filename, and another prompt will ask if you want to load.
4. When the disk stops whirring and the red busy light goes off, close the buffer (press CTRL–SHIFT-C).
5. Establish your communications link with the remote computer (you could have done this during step 1, if desired). Notify the person at the other end of the line that you're ready to send the program file. The other system must be set to receive Commodore ASCII. If the other person has Plus/Term, both of you should adjust your parameters for COMM TO COMM (selection 2 on the terminal options menu described above). The person at the other end should then open and zero his or her buffer (CTRL–SHIFT-O and CTRL–SHIFT-Z).

6. Now press the f7 key to exit terminal mode and reach the main menu. Select option 7 to call up the terminal options menu. Choose option 6 (PRINT BUFFER) and send the file through the RS-232 port (where your modem is connected). After awhile, you'll be asked to press any key to continue. After you press a key, the transfer is complete.

As the file is transmitted, it is listed on the screen of the remote computer. When the word READY appears, the upload is finished. The buffer can then be closed (CTRL–SHIFT-C) and saved (CTRL–SHIFT-S). If the file is a BASIC program, remember to use the Tokenizer utility to convert it from ASCII into a tokenized file before running it.

Using the Tokenizer

To use the Tokenizer, select it from the menu. You'll soon see a prompt telling you to enter a line such as this:

SYS 828,"0:*filename*"

Be sure to substitute the *filename* for the file you wish to tokenize.

The file should list on your screen and end with a SYNTAX ERROR message. This is normal; ignore the error. If the uploading/downloading process has been successful, you will have a ready-to-run BASIC program in memory that you can save on disk.

Index

COMPUTE! Books

Ask your retailer for these **COMPUTE! Books** or order directly from **COMPUTE!**.

Call toll free (in US) **800-346-6767** (in NY 212-887-8525) or write COMPUTE! Books, P.O. Box 5038, F.D.R. Station, New York, NY 10150.

Quantity	Title	Price*	Total
______	SpeedScript: The Word Processor for the Commodore 64 and VIC-20 (94-9)	**$ 9.95**	______
______	Commodore SpeedScript Book Disk	**$12.95**	______
______	128 Machine Language for Beginners (033-5)	**$16.95**	______
______	COMPUTE!'s First Book of the Commodore 128 (059-9)	**$14.95**	______
______	COMPUTE!'s Commodore 64/128 Collection (97-3)	**$12.95**	______
______	All About the Commodore 64, Volume Two (45-0)	**$16.95**	______
______	All About the Commodore 64, Volume One (40-X)	**$12.95**	______
______	Programming the Commodore 64: The Definitive Guide (50-7)	**$24.95**	______
______	COMPUTE!'s Data File Handler for the Commodore 64 (86-8)	**$12.95**	______
______	COMPUTE!'s Kids and the Commodore 128 (032)	**$14.95**	______
______	Kids and the Commodore 64 (77-9)	**$12.95**	______
______	COMPUTE!'s Commodore Collection, Volume 1 (55-8)	**$12.95**	______
______	COMPUTE!'s Commodore Collection, Volume 2 (70-1)	**$12.95**	______
______	COMPUTE!'s Personal Accounting Manager for the Commodore 64 (014-9)	**$12.95**	______
______	COMPUTE!'s VIC-20 and Commodore 64 Tool Kit: BASIC (32-9)	**$16.95**	______
______	COMPUTE!'s VIC-20 and Commodore 64 Tool Kit: Kernal (33-7)	**$16.95**	______
______	COMPUTE!'s Telecomputing on the Commodore 64 (009)	**$12.95**	______
______	COMPUTE!'s VIC-20 Collection (007)	**$12.95**	______
______	Programming the VIC (52-3)	**$24.95**	______
______	VIC Games for Kids (35-3)	**$12.95**	______
______	Mapping the VIC (24-8)	**$14.95**	______
______	COMPUTE!'s 128 Programmer's Guide (0319)	**$16.95**	______
______	The Complete 64 (Book/Disk Combination) (0629)	**$29.95**	______

*Add $2.00 per book for shipping and handling.
Outside US add $5.00 air mail or $2.00 surface mail.

NC residents add 4.5% sales tax ________
Shipping & handling: $2.00/book ________
Total payment ________

All orders must be prepaid (check, charge, or money order).
All payments must be in US funds.
□ Payment enclosed.
Charge □ Visa □ MasterCard □ American Express

Acct. No.______________________ Exp. Date________
(Required)

Name______________________________________

Address____________________________________

City__________________ State ________ Zip________

*Allow 4–5 weeks for delivery.
Prices and availability subject to change.
Current catalog available upon request.

COMPUTE!'s

The Complete 64

Directions: Turn on your disk drive, monitor or TV, and computer. (Commodore 128 users should type **GO 64** and press RETURN.) Insert disk into drive and type **LOAD "MENU",8** and press RETURN.

The Complete 64

Turn your Commodore 64 or 128 into a power machine with this collection of seven extraordinary applications programs. Play music, write letters, or keep a budget with your computer and *The Complete 64*. All programs and supporting files are on disk, making *The Complete 64* an even more impressive package.

- ***SpeedScript.*** A particularly efficient, full-featured word processing program. Five custom character sets are included.
- ***SpeedCalc.*** A spreadsheet program that rivals commercial software. And spreadsheets created with *SpeedCalc* can be loaded into a *SpeedScript* document.
- ***Mini-Filer.*** An easy-to-use and flexible file manager.
- ***Sidplayer.*** The popular music editor and player. Includes music on disk that you can immediately listen to and modify.
- ***Ultrafont +.*** Easily create your own custom character sets for use with *SpeedScript* or with your own BASIC programs.
- ***High-Resolution Sketchpad.*** For the artist in you. Create detailed pictures on the high-resolution or multicolor screen.
- ***Plus/Term.*** Connect a modem to your 64 or 128 and communicate with computers anywhere in the world with this sophisticated terminal program.

All these applications and much more are included on the disk. Just load the easy-to-use menu program and make your selection—absolutely no programming experience is necessary. Clear and concise directions make the applications even more convenient to use.

The Complete 64 has what you need to turn your Commodore 64 or 128 (in 64 mode) into a more powerful, useful tool than you ever thought possible.

ISBN 0-87455-062-9

Includes a program disk. System requirements: Commodore 64 or 128, Commodore 1541 or 1571 disk drive, color or monochrome monitor or television. Printer optional.

COMPUTE!'s The Complete 64

COMPUTE! Books