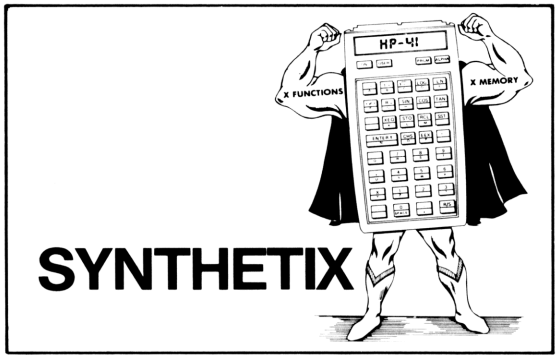
SKWIDBC

THE HP-41 BARCODE GENERATION ROM

A JOINT VENTURE OF

SKWID INK



SKWIDBC
USER'S MANUAL

SYNTHETIX
P.O. Box 1080
Berkeley, CA 94701-1080
U.S.A.

ACKNOWLEDGEMENT

SKWID INK would like to thank Gary Friedman for suggesting the idea of LaserJet barcode and for offering the use of his father's LaserJet printer. We would also like to thank Dr. George Friedman for allowing us to use his LaserJet through many revisions of SKWIDBC.

Thanks also go to Clifford Stern who wrote the COMPEND function and for writing a COMPILE function upon which the SKWIDBC COMPILE function is loosely based.

DISCLAIMER

TABLE OF CONTENTS

INTRODUCTION 1

USING THE SKWIDBC FUNCTIONS

- ABC and AABC 5
- DIREXBC 8
- PROGBC and PPROGBC 10
- PROGRW and PPROGRW 13
- COMPOFF 16
- PAPERKB 17
- COMPEND 20
- COMPILE 21
- ACTEXT 22
- XTOALFT 23
- XTOART 24
- CHKSUM 25
- MAKEBC 25
- FRMFEED 34
- XBC 34
- REGBC and REGBCX 36
- COLBC 39

SELF TEST EXAMPLE PROGRAM 41

BARCODE STRUCTURE 43

APPENDIX A: SKWIDBC Error Messages 49

APPENDIX B: XROM Numbers 53

APPENDIX C: Interfacing a LaserJet series printer to the HP-41 55

APPENDIX D: Troubleshooting Tips 63

APPENDIX E: Barcode for Example Programs 65

INTRODUCTION

SKWIDBC has been developed to enable users of the HP-41 Wand to print their own barcode. Prior to the introduction of SKWIDBC, generating barcode for use by the Wand has been an expensive or difficult task which required extra equipment costing thousands of dollars. With SKWIDBC, you may now produce top-quality barcode on the HP-IL ThinkJet printer or the HP LaserJet, including the LaserJet Plus and Series II. The functions provided in SKWIDBC allow you to produce any and all of the allowable types of barcode which the Wand will read.

The functions presented in SKWIDBC have many advantages over previous programs written to print barcode. First and foremost they are the *fastest* barcode printing programs ever written to print HP-41 barcode on an HP ThinkJet or LaserJet. Second, no space in main memory is used by these programs. This allows you to print barcode even though the memory of the calculator is full of programs and data. Third, previous barcode printing programs required the use of at least two ROMs in addition to the HP-IL module, forcing you to remove other modules that you may have normally kept in your HP-41.

We hope that you enjoy the mix of functions provided in SKWIDBC and have fun on your journeys through the manual.

Many of the SKWIDBC functions operate slightly differently, depending on whether you are using a ThinkJet or a LaserJet series printer. In this manual, the use of each function is described first assuming that you are useing a ThinkJet. Then the differences in operation for the LaserJet are described. All the LaserJet printers work the same with SKWIDBC.

SKWIDBC+ Upgrade

SKWIDBC+ is an alternate version of SKWIDBC designed especially for people who have access to a LaserJet Plus (or Series II). It prints barcode roughly 15 times faster than SKWIDBC on the LaserJet Plus or Series II. A full page of program barcode prints in 25 to 30 seconds. You can

distinguish SKWIDBC from SKWIDBC+ by their headers in Catalog 2: the header for SKWIDBC+ has a + in it; the one for SKWIDBC does not.

You may upgrade from SKWIDBC to SKWIDBC+ by sending your SKWIDBC module to SYNTHETIX and paying the price difference. See the order blank at the back of this manual.

But before you decide to upgrade, you should be aware that you will give up one SKWIDBC feature: SKWIDBC+ will not work with the original HP LaserJet printers. These printers, which are no longer manufactured by HP, do not support the downloadable font feature on which SKWIDBC+'s high-speed operation is based.

Installing the SKWIDBC module in the HP-41

To insert the SKWIDBC module into the HP-41:

1.) Ensure that the calculator is turned off.
2.) Remove one of the port caps from an empty port (or another module if all ports are being used).
3.) Insert the SKWIDBC module into the empty port.
4.) Make sure that you do not have another module with XROM ID number 08 plugged in. The only such module manufactured by HP is the Stress Analysis Pac.

Now you are ready to use SKWIDBC.

NOTE: If you execute CATalog 2 with your SKWIDBC or SKWIDBC+ module present, you will probably see an entire duplicate set of functions. This is an artifact of the interface chip used in your module. (The other chip in the module is a CMOS EPROM, which is erasable and reprogrammable using special equipment.) The phantom duplicate function entries in Catalog 2 will not disturb the normal functioning of your HP-41, including its finding and executing SKWIDBC functions.

Executing a SKWIDBC function

For those of you who have little or no experience using the HP-41 calculator, when the instruction "execute ABC" is given you should do the following steps:

1.) Press the XEQ key (to the right of the yellow shift key).
2.) Press the ALPHA key (upper right just below the display). Now the blue letters on the slanted part of the keys are in effect.
3.) Key in the appropriate letters -- in this case, A B C.
4.) Hit the ALPHA key again.

That's all there is to executing a function by name from the keyboard.

The HP82153A Wand

Since its introduction in 1980, two versions of the Wand have been released by Hewlett-Packard. The first, which was in production until 1981, is Revision 1E. During a CAT 2 its header will be -WAND 1E-. It has several bugs. The other revision displays its header as -WAND 1F- during Catalog 2. Most of the bugs from the earlier revision were fixed and three new barcode types were added. The barcode produced by the REGBC and REGBCX functions is one of these new types and cannot be read by a Revision 1E wand.

CAUTION

If you are using the ThinkJet printer with SKWIDBC, you must **clear flag 0.** If you are using a LaserJet printer, **set flag 0.**

ERROR MESSAGES

If you get the error message: NO IL PR. ROM when you are using SKWIDBC with the ThinkJet, this means either that the HP-IL module is not plugged into the calculator or that the switch on the back of the module is set to Disable. The switch on the HP-IL module must be set to Enable for any of the barcode functions to work.

The error message NO THINKJET means that there is no ThinkJet printer in the loop. With flag 0 clear, SKWIDBC expects to see an HP-IL ThinkJet. If this device is not present, you will see the error message NO THINKJET. If you happen to be using either an RS-232 ThinkJet through an HP-IL RS-232 interface or a parallel ThinkJet printer via an HP82166A parallel converter, you should make either the RS-232 interface or the parallel converter the selected device and execute the HP-IL module's MANIO function to override this HP-IL printer existence check.

The error message NO INTERFACE means that there is no interface device, such as an RS-232 serial interface or an HP82166A parallel converter, in the loop. Since there is no HP-IL LaserJet printer, an HP-IL Module and an interface device are required to connect the LaserJet printer to the HP-41.

The message TRANSMIT ERR means that either that the loop is broken or that one of the devices on the loop is not turned on.

It is possible that the number of graphic bytes needed to print the row of barcode is greater than the width of the graphics area on a ThinkJet. If this occurs the LINE TOO LG. error message will be generated. Other error messages specific to each function will be discussed in the description of the corresponding function.

Throughout the manual we shall assume that you have an HP-IL module plugged into the HP-41 with the switch on the back in the Enable position. It is also assumed that there is a working HP-IL ThinkJet printer somewhere in the loop. You should also be in AUTOIO mode. For tips on connecting a LaserJet series printer to HP-IL, see Appendix C. Remember to clear flag 0 for a ThinkJet, and set flag 0 for a LaserJet.

USING THE SKWIDBC FUNCTIONS

The rest of this manual describes each function in turn -- the purpose of the function, the inputs required, and the results. ThinkJet operation is described first, then any differences that apply to the LaserJet printers.

If you have SKWIDBC+, all the functions work identically except that SKWIDBC+ does not support the original LaserJet printer. The LaserJet Plus and LaserJet Series II output is dramatically faster than for SKWIDBC, but the usage is identical. There are no differences between SKWIDBC and SKWIDBC+ for ThinkJet operation.

ABC and AABC

The barcode produced by the ABC (Alpha Bar Code) and AABC (Alpha Append Bar Code) functions allows you to read up to 14 characters into the alpha register. ABC prints Alpha replace BarCode, while AABC prints Alpha Append BarCode. To use either of these functions, place the character string in the ALPHA register, then execute ABC or AABC. The printed output will show the alpha characters surrounded by quotes above a row of barcode. With AABC the word APPEND precedes the printed character string. This allows you to easily distinguish between the two different types of alpha barcode.

LaserJet note: The LaserJet printers do not immediately output a page after you execute ABC or AABC. That would be a waste of paper in most cases, because you may want to print more barcodes on the same page. If you need to see the ABC or AABC barcode immediately, you should execute SKWIDBC's FRMFEED (form feed) function to eject the page.

When you scan a barcode produced by ABC or AABC, you will be placed into ALPHA mode and the characters will be placed into the alpha register if you are in run mode. If you are in PRGM mode, the characters will be input as a program step. If the barcode row was printed by AABC, the characters will be appended to the alpha register in run mode, or entered as an append alpha string in PRGM mode. Unfortunately, the revision 1E Wand fails to supply the necssary append symbol for AABC barcode in PRGM mode, so you get an alpha replace text line instead of an alpha append instruction.

Now let's do some examples.

EXAMPLE 1

Print the characters "-SKWIDBC" as a line of barcode. To do this:

1.) Place the message "-SKWIDBC" in alpha.
2.) Clear flag 0 (ThinkJet) and execute ABC.

In about 30 seconds (ThinkJet) the barcode row will be finished printing. The characters which are represented by the row of barcode are printed above the row, enclosed in quotes. You may now scan the barcode with your wand to verify it.

EXAMPLE 2

For our second example, we shall use the AABC function to create the message " IS THE GREATEST" as a row of alpha append barcode.

1.) Place the message " IS THE GREATEST" in alpha.
2.) Execute AABC.

Oh, no! The message ALPHA > 14 came up in the display. This is because you may only have up to 14 characters in alpha when executing ABC and AABC. If you count the number of characters currently in alpha you will see there are 16, which is too many. Let's try it again with " IS GREAT" in alpha.

3.) Place the message " IS GREAT" in alpha.
4.) Execute AABC.

The word APPEND followed by " IS GREAT" is printed above the barcode row to tell you this row will be appended to the alpha register when read in. Now scan the first row of barcode which was printed. The message "-SKWIDBC" will be placed in the alpha register and the calculator will be placed in alpha mode. Whatever was in alpha before you scanned this row will be replaced with this message. Now scan the second barcode row. " IS GREAT" will be appended to whatever is in alpha.

Because of a bug in the Wand software, an alpha string in which all of the first six characters have an ASCII value of 15 (hexadecimal 0F) or less will crash the calculator when scanned. (This is a "soft" crash that can be recovered from using either the normal ON/ENTER key reset procedure or brief removal of the batteries.) The crash will also happen if all of the characters 7 through 12 or 13 through 14 have ASCII values of 15 or less. Because of this bug, SKWIDBC checks to ensure that the alpha string will generate a usable barcode.

Two related Wand bugs should be mentioned. Characters with ASCII values of 15 or less will be dropped if you scan the barcode in run (non-PRGM) mode. Also, characters with ASCII values of 240 or greater will give improper results (but not crashes) when scanned in run mode.

EXAMPLE 3

Place the angle character (ASCII value 13) into alpha. Now exit alpha and place 1 in X and execute XTOART. Now execute AABC. The DATA ERROR message was generated since both alpha characters had ASCII values of less than 16.

EXAMPLE 4

Let's print barcode for the string "A∢B", which is the character A followed by the angle character and a B. Place A, angle, B in alpha and execute ABC.

You will get the text printed above the barcode row. No error was generated even though the angle character has a value of less than 16, since there were other characters in alpha which had ASCII values greater than 16.

Usage Summary for ABC and AABC

Inputs:

X register: none

Alpha register: up to 14 characters, which you wish to be represented by the printed barcode row.

LaserJet differences: Flag 0 must be set.

DIREXBC (nonprogrammable)

This is the DIRect EXecution BarCode function. It allows you to print a function such as STO 59 or XEQ "PLG" as a row of barcode. When the row is scanned the function will be executed if you are in run mode or inserted as a program step if you are in PRGM mode.

There are several types of instructions which are not allowed for use as direct execution barcode. These are: one byte labels (numeric LBL's from 00 to 14), numbers, END instructions, alpha labels, numeric XEQ's and GTO's, and alpha characters. (For alpha characters use ABC and AABC.)

In order to use the DIREXBC function, there must be a program in the main memory of the calculator with a LBL "Q" instruction. The LBL "Q" need not be the first line of the program. The instruction you wish to have printed in the barcode row must be present as the next instruction after the LBL "Q". We shall do an example using the "+" instruction. Here is how you would set up the LBL "Q" section to print direct execution barcode for the + instruction.

EXAMPLE 5

```
01 LBL "Q"
02 +
03 END
```

Now you can execute DIREXBC while in program mode. The function is nonprogrammable, so it won't be inserted as a program step. The instruction name is printed above the row of barcode.

Try scanning the barcode. In run mode, the + function will be executed. In PRGM mode, a + will be inserted as a program step.

If you place one of the illegal instructions mentioned above as the step following LBL "Q", then the ILLEGAL INST. message will be displayed by DIREXBC. If there is no program with a LBL "Q" in main memory then the NO LBL Q error message comes up.

Let's try placing an XEQ 01 instruction into barcode.

EXAMPLE 6

Key in XEQ 01 as the step immediately after the LBL "Q".

```
01 LBL "Q"
02 XEQ 01
03 END
```

Now execute DIREXBC and the ILLEGAL INST. message will appear, since numeric XEQ's are not allowed in direct execution barcodes.

Note for Synthetic Programmers (skip this note if you don't know what Synthetic Programming is): DIREXBC will not allow synthetic instructions such as RCL M, X<> c, or STO IND J, because these barcodes will not be interpreted correctly by the Wand. (The Wand is not as liberal as the HP-41 itself about support of synthetic instructions.) If one of these instructions is the one immediately after LBL "Q" the ILLEGAL INST. message is given.

Usage Summary for DIREXBC

Inputs:

X register: none

Alpha register: none

Other: There must be a LBL "Q" in main memory. The instruction to be printed must be the program step directly after LBL "Q".

LaserJet differences: Flag 0 must be set.

PROGBC and PPROGBC

These are two functions for printing program barcode. PROGBC (PROGram Bar Code) prints regular program barcode and PPROGBC (Private PROGram Bar Code) is used if you want the program to be private when it is read from the barcode.

Before you use these functions, you must place in X the number of program bytes you want to have printed in each row of barcode. Also, the name of the program to be printed must be in ALPHA. The number of program bytes printed per row may be between 1 and 12 for the ThinkJet (flag 0 clear) and between 1 and 13 for the LaserJet (flag 0 set). If the number in the X register is outside the allowed range then the DATA ERROR X message will be displayed. If the program named in alpha does not exist, then the NONEXISTENT message comes up. The program for which barcode is to be printed must be in the main memory of the calculator or the "ROM" error is generated. If the program you are trying to put into barcode is private, the "PRIVATE" error will be displayed. If you try to barcode an MCODE function the "NONEXISTENT" error message is generated.

When you execute either PROGBC or PPROGBC, the program to be printed should have a nonpermanent END (not the .END.) as its last step. If the .END. is the last step of the program, then an END will automatically be inserted. The program will then be PACKED, with the PACKING message displayed.

The next step is compiling. A brief explanation is in order. As part of its normal operation, the HP-41 is able to store jump distances for most GTO and XEQ instructions. The first time you execute one of the GTO or XEQ instructions, the HP-41 searches for the matching LBL, then saves the jump distance and direction if possible. This process is called compiling.

SKWIDBC has a built-in function that automatically computes and saves the jump distance and direction for all the GTOs and XEQs in your program. This provides you the fastest version of the program when you scan in the barcode.

During the compiling step, the jump distances will be placed into all numeric GTOs and XEQs. During compiling, which can take over a minute for a

long program, the message COMPILING will be placed into the display.

If you have a GTO to a numeric LBL from 00 to 14 and the jump distance is greater than 111 bytes, the compiling routine will attempt to convert the corresponding GTO instruction into a synthetic three-byte GTO, so that the jump distance can be compiled. If there is not enough room to insert the extra byte then the PACKING, TRY AGAIN error is generated. If the insert is successful then the PACKING message will come up again. Next COMPILING will again be displayed and so on until the whole program is compiled. If a GTO or XEQ does not have a corresponding LBL to jump to then the error message NO LBL XX , where XX is the number of the missing LBL, will be displayed and the program pointer will be placed at the offending GTO instruction so you may view it when you go into program mode.

After the program is compiled, printing starts. First the message PROGRAM: (or PRIVATE PROGRAM:) is printed followed, by the name of the program. On the same line, the number of registers necessary to read in the whole program is printed, along with the number of bytes in the program and the number of rows needed to print the program. The PRINT RW. 01 message is displayed. When row two is being printed, the message is updated to a PRINT RW. 02. The displayed row number is incremented as each barcode row is printed.

After the first 20 rows are printed, a form feed is given. The program name is printed at the top of the page, followed by the 21st row. A new page is started for every 20 rows until all of the barcode is printed.

If you wish to terminate PROGBC or PPROGBC in the middle of printing barcode, you may do so by pressing either the R/S (halt) or the ON (halt and turn off) key. The printing will stop as soon as the current row of barcode is finished. Other keys have no effect.

Note: If you are using a ThinkJet, the mode is changed to bold and 8 lines per inch. When the program terminates, the ThinkJet is set back to normal printing and 6 lines per inch character printing. If these are not your normal settings, you may need to write a short program to send an escape sequence to restore your preferred settings.

Let's do an example. We shall print barcode for the PRPLOT program, from the HP-IL printer ROM, at 12 program bytes per row.

EXAMPLE 7

Place 12 in the X register and the characters "PRPLOT" in ALPHA. Now execute PROGBC. The ROM error message will be displayed since "PRPLOT" is still in ROM. We must use the COPY function to copy it from ROM into main memory. Execute COPY and hit the ALPHA key when the single prompt appears. Then fill in "PRPLOT" and hit the alpha key again. You must have at least 79 unused registers to copy "PRPLOT" into RAM. Reduce the SIZE or use CLP to clear one or more programs if necessary.

Now let's try again. Place 12 in X and "PRPLOT" into alpha. Now execute PROGBC and SKWIDBC will display PACKING, then COMPILING. It seems that the PRPLOT program has 3 two-byte GTO's which cannot be compiled since the jump distance is too far. So the PACKING message will come up three more times followed by the COMPILING message. The program will wind up being 540 bytes long and will take exactly 45 rows of barcode to print. The time to print the whole program is approximately 27 minutes. The time would have been less except that the compiling took longer than usual because three GTO's had to be expanded. With SKWIDBC+ and a LaserJet Plus or compatible printer, the total time is 2 minutes and 18 seconds.

Usage Summary for PROGBC and PPROGBC

The use of PPROGBC is the same as for PROGBC except that when the barcodes printed by this function are read, the program will be private.

CAUTION: Use FRMFEED before PROGBC or PPROGBC, or manually make sure that the printing starts at the top of the page. Otherwise the first 20 rows may not fit on the first page.

Inputs:

X register: number of program bytes per row of barcode. From 1 to 12 for the ThinkJet, 1 to 13 for the LaserJet.

Alpha register: name of the program to be printed. If alpha is empty, then the current program is printed.

LaserJet differences: Flag 0 must be set. The number of program bytes per row (the number input to X) may be from 1 to 13. The last page of barcode is ejected even if 20 rows of barcode were not printed on it.

PROGRW and PPROGRW

PROGRW prints PROGram barcode starting from the RoW specified in Y. Similarly, PPROGRW prints Private PROGram barcode starting from RoW Y. Except for the additional input in Y, the use of these programs is exactly the same as the PROGBC and PPROGBC functions. Place the name of the program in alpha, the number of program bytes per barcode row in X, and the number of the barcode row at which you want printing to start in Y. This allows you to start barcoding a program from any row. If the ThinkJet printhead ran out of ink in the middle of printing, or the batteries went dead, you don't have to start all over.

Note: In order to use the barcode rows printed by these functions with rows barcoded by PROGBC or PPROGBC, the number of program bytes per barcode row MUST be the same for both sets of barcode. If this is not the case then you will not be able to append the barcode printed by PROGRW/PPROGRW to the rows printed with PROGBC/PPROGBC. The barcodes from PROGRW cannot be appended to the ones printed with PPROGBC. If you barcode a program as private then the rows which are to be appended must also be of the private type, that is, you must use PPROGRW. If the original barcode was done with PROGBC then PROGRW must be used to append rows.

The barcode for the specified program is printed starting with the row specified in Y until the end of the program. After 20 rows have been barcoded, the paper is advanced to the next page and and the program name is printed followed by the next row and so on until the end of the program is reached.

Because the functions which print program barcode print 20 barcode rows per page and then advance the paper it is imperative that the page you start with is at the top of form. (Use FRMFEED first if you have a LaserJet and you're not sure whether anything has been printed on the page yet.)

On the ThinkJet you must start at the top of the page, because otherwise you might get a barcode row which lies on the perforation between two pages.

The possible error messages are the same as with PROGBC/PPROGBC, with one addition. The extra error message is "DATA ERROR Y" if the row number in Y is greater than the number of barcode rows needed to print the program.

EXAMPLE 8

For this example, we shall print barcode for the PRPLOT program beginning with the 23rd row. If the PRPLOT program is no longer in the main memory of the HP-41, copy it back into main memory as was done during Example 7. Place the name of the program ("PRPLOT") in alpha, then place 23 in X, press ENTER↑, and put 12 in X. Now you may execute PROGRW. PROGRW will PACK the program and place a nonpermanent END on the program if it does not already have one. Then COMPILING will be displayed. If you are using the PRPLOT program which was already compiled in the last example then no GTO's will have to be expanded and COMPILING will only be displayed once. However, if you copied PRPLOT from ROM again PROGRW will go through the same procedure as PROGBC and expand the three GTO's which cannot be compiled.

At this juncture the PRINT RW. nn message is displayed. The row number nn starts at 01 and is quickly incremented until the designated row number is reached. Printing is then begun and continues until the end of the PRPLOT program is reached.

If you scan the first 22 rows printed when the PROGBC function was used, you may then scan the rest of the rows from either those printed by PROGBC or the ones just printed with PROGRW. The two are identical. Now let's try another example.

EXAMPLE 9

Place "PRPLOT" in alpha, 50 in Y and 12 in X. The program will be packed and then compiled. Then the PRINT RW. nn message is displayed, with nn counting up from 01 very quickly.

Oh no! What happened? The program displays DATA ERROR Y. This occurred because there are only 45 rows of barcode and we wanted to start printing at the 50th row. Since, there is no 50th row, the error message was given.

Usage Summary for PROGRW and PPROGRW

Inputs:

X register: the number of program bytes per row, from 1 to 12 for the ThinkJet, from 1 to 13 for the LaserJet.

Y register: the row at which to start printing barcode. If Y=0, then row 1 is assumed.

Alpha register: the name of the program to be barcoded. If alpha is empty, then the current program is printed.

LaserJet differences: Flag 0 must be set. The number of program bytes per row (the number in X) may be from 1 to 13. The last page of barcode is ejected even if 20 rows of barcode were not printed on it.

CAUTION: PROGBC, PPROGBC, PROGRW, and PPROGRW can be used in a program, but not in a subroutine. This is because the user subroutine stack gets wiped out during PACKING when the program to be barcoded is printed. For example, you might expect the routine listed below to barcode the programs "PRPLOT" and "CDE", but it won't work because execution does not return from line 11.

```
01*LBL "ABC"
02 "PRPLOT"
03 XEQ 01
04 "CDE"
05 XEQ 01
06 BEEP
07 RTN
08*LBL 01
09 12
10 PROGBC
11 END
```

The above example would barcode the "PRPLOT" program and then stop at line 11. The subroutine stack is wiped out by the PROGBC function at line 10. The example below shows the correct procedure:

```
01*LBL "ABC"
02 "PRPLOT"
03 12
04 PROGBC
05 "CDE"
06 PROGBC
07 BEEP
08 END
```

COMPOFF

The COMPile OFF function is for use with the functions which print program barcode. If you have a program to be barcoded which is already compiled, or if you don't wish to compile it, then just execute COMPOFF first and this part of the process will be skipped. We could have saved about 2.5 minutes by doing this with PRPLOT. The program barcode functions will still ensure that the permanent END (the .END.) is not the last step of the program. After this is done program memory is packed and barcoding is started.

The COMPOFF function sets flags 42 and 43 simultaneously to tell SKWIDBC's program barcode functions not to compile a program. These are the two flags which determine your trig mode: degrees, radians, or gradians. In degrees mode both flags are clear, in radians mode flag 43 is set, and gradians sets flag 42. Under normal conditions, both flags will never be set. When COMPOFF sets both flags, radians mode is activated and the RAD annunciator is turned on. To re-enable compiling of programs by the program barcoding functions, just execute the DEG, RAD, or GRAD functions to restore normal settings of flags 42 and 43.

Usage Summary for COMPOFF

Inputs:

X register: none

Alpha register: none

Other: Sets flags 42 and 43 and turns on the RAD annunciator.

LaserJet differences: none

PAPERKB (nonprogrammable)

The PAPERKB function prints PAPER KeyBoard barcode. Paper Keyboard barcode allows you to use the Wand barcodes as the keyboard of the calculator. The operation of PAPERKB is much the same as DIREXBC.

Just as for DIREXBC, there must be a program in the main memory of the calculator with a LBL "Q" instruction. The LBL "Q" need not be the first line of the program. The instruction you wish to have printed in the barcode row must be present as the next instruction after the LBL "Q". If this instruction is a two byte instruction (such as STO 99), only the first byte will be barcoded. Whenever an instruction requiring a suffix is barcoded by PAPERKB, scanning the barcode will give the function name with the appropriate prompts. The only two byte functions allowed by PAPERKB are XROM functions. These are functions which execute a function or program in a plug-in eXternal ROM module.

With PAPERKB, all of the functions keyable from the keyboard are available except XEQ IND, GTO IND, and alpha characters whose ASCII value is greater than 127.

EXAMPLE 10

For this example key in the following program steps.

```
01*LBL "Q"
02 "A"
03 END
```

Now execute PAPERKB. A short barcode will be printed with the designation A (not in quotes) above it. When you scan this barcode, the HP-41 will be placed into alpha mode and an A will be placed into alpha (or inserted as a program instruction if you are in program mode). If you are in the middle of keying in alpha characters, the A will be appended and a prompt will be appended after the A.

EXAMPLE 11

Now place the PRA (PRint Alpha) function as the step after the LBL "Q".

```
01*LBL "Q"
02 PRA
03 END
```

Execute PAPERKB. If you scan this barcode the PRA function will be executed or inserted into a program if you are in program mode.

If the instruction after the LBL "Q" is a numeric LBL from 00 to 14, then the resulting paper keyboard function is one of the HP-41's built-in nonprogrammable functions. These functions and their corresponding labels are listed in the table below.

The asterisks in the table indicate that the function name is not correctly printed above the barcode. You can get around this problem by using other SKWIDBC functions. The exact procedure will be given in the description of MAKEBC.

Nonprogrammable function	Corresponding LBL
GTO ..	LBL 00*
DEL	LBL 01
COPY	LBL 02
CLP	LBL 03
R/S	LBL 04*
SIZE	LBL 05
BST	LBL 06
SST	LBL 07
ON	LBL 08
PACK	LBL 09
<- (delete)	LBL 10*
ALPHA ON/OFF	LBL 11*
PRGM ON/OFF	LBL 12*
USER ON/OFF	LBL 13*
ASN	LBL 14

Let's try the nonprogrammable function SIZE as our next PAPERKB example.

EXAMPLE 12

The SIZE function corresponds to LBL 05 in the preceding table. Place LBL 05 as the step immediately after LBL "Q".

```
01*LBL "Q"
02*LBL 05
03 END
```

Now execute PAPERKB. SIZE will be printed above the barcode (not LBL 05). When this barcode is scanned the SIZE function is executed, giving the familiar 3 prompts in display.

Numeric paper keyboard barcodes may be scanned in much the same as alpha barcodes. These numbers are single digits and may be used anytime you would normally press a number key. For this next example we shall make a paper keyboard barcode for the digit 1.

EXAMPLE 13

Place the number 1 as the step immediately after the LBL "Q".

```
01*LBL "Q"
02 1
03 END
```

Now execute PAPERKB. A 1 will be printed above a short barcode row. Now if you scan the SIZE barcode, which was printed in Example 12, then scan the 1 barcode three times, you will get a size of 111. The numeric barcode filled in the prompts for the SIZE.

Usage Summary for PAPERKB

Inputs:
X register: none
Alpha register: none
Other: Place the instruction to be barcoded after a LBL "Q".

LaserJet differences: Flag 0 must be set.

COMPEND (nonprogrammable)

The COMPiled END function attaches a packed, compiled END to the last program in main memory. This allows you to read in a program with the wand, card reader, cassette drive, or from extended memory without risking loss of the compiled jump distances for all of the GTOs and XEQs.

After you have read in the program, immediately execute COMPEND. "Immediately" in this context means before you PACK, GTO . . , turn the calculator off, or exit PRGM mode. Any of these actions would trigger decompiling. Once the new packed, compiled END is in place, it will prevent the GTO's and XEQ's from being decompiled.

EXAMPLE 14

In this example we shall read in the PRPLOT program printed in Example 7 and then place a packed, compiled END on it. First read in the program from barcode. Since the GTO's and XEQ's were compiled right before the barcode was printed, they will be still compiled when PRPLOT is read back in. Now execute COMPEND to attach the protective packed, compiled END.

Usage Summary for COMPEND

Inputs:
X register: none
Alpha register: none

LaserJet differences: none

COMPILE (nonprogrammable)

As part of its normal operation, the HP-41 is able to store jump distances for most GTO and XEQ instructions. The first time you execute one of the GTO or XEQ instructions, the HP-41 searches for the matching LBL, then saves the jump distance and direction if possible. This process is called compiling.

The COMPILE function computes all the required jump distances and stores them in the numeric GTO and XEQ instructions. To do this yourself, you would have to SST each GTO or XEQ in run mode, or run the program if you are sure each of these instructions will be executed. But the COMPILE function actually does more -- it converts two-byte (short-range) GTO instructions to three-byte (long-range) form where necessary.

To use the COMPILE function, just execute COMPILE. A single prompt will appear. Press the alpha key and fill in the name of the program. Press alpha again and COMPILE will start working.

The first thing COMPILE does is make sure that there is a nonpermanent END on the program. If the .END. is the last step of the program, an END is

added and the program is packed. Then the program is compiled and the COMPILING message is displayed.

Compile will change a two-byte GTO (suffixes 00 to 14) to a synthetic three-byte GTO using the same LBL if the jump distance is more than 111 bytes. In this case the program must be re-packed (because an insertion took place) and compiling starts over. When COMPILE is finished a TONE 7 is sounded.

Usage Summary for COMPILE

Inputs:

X register: none

Alpha register: none

Other: At the prompt hit the alpha key and fill in the name of the program and then hit alpha again.

LaserJet differences: none

ACTEXT

The ACcumulate TEXT function is part of SKWIDBC's custom barcode toolkit. ACTEXT allows you to custom-label your barcodes. It takes the characters in alpha and copies them into the buffer of the device to which you are printing barcode.

Since there are no ThinkJet or LaserJet characters which correspond to the angle, not equal, and append characters, other character sequences have been substituted for these characters. They are, $\blacktriangleleft$ for angle, $\neq$ for not equal, and $\blacktriangleright$ for the append character. Characters which have an ASCII value of less than 32 or greater than 127 are shown as ⊛ (asterisk inside an O).

An example of the use of ACTEXT will be given in the description of the MAKEBC function.

Usage Summary for ACTEXT

Inputs:

X register: none

Alpha register: the characters to be accumulated to the printer.

LaserJet differences: Flag 0 must be set.

If you are using a ThinkJet, the text is in bold and at 8 lines per inch. If you are using a LaserJet, the left margin is set to 10. The printers are left in these modes when ACTEXT is finished printing.

XTOALFT

The X TO Alpha LeFT function is part of SKWIDBC's custom barcode toolkit. XTOALFT appends the ASCII equivalent of the number in X to the left end of the alpha register. Only numeric inputs are allowed, and only the integer part is used. LASTX is not updated. An alpha string in X causes the ALPHA DATA error message. Numbers with an integer part greater than 255 give the DATA ERROR message.

To use XTOALFT, just place a number from 0 to 255 in X and execute XTOALFT. The ASCII equivalent of the number will be appended as the first character in alpha. If the alpha register was full (had 24 characters) then the rightmost character will be deleted and the number in X will then become the leftmost character. No warning tone will be sounded.

An example of the usage of XTOALFT is given in the MAKEBC section.

Usage Summary for XTOALFT

Inputs:

X register: number from 0 to 255

Alpha register: none

LaserJet differences: none

XTOART

The X TO Alpha RighT function is part of SKWIDBC's custom barcode toolkit. XTOART works exactly the same as the XTOA function in the HP-41CX and in HP's Extended Functions module. XTOART appends the ASCII equivalent of the number in X to the right end of the alpha register. Only the integer part of the input is used, and LASTX is not updated. Numbers with an integer part greater than 255 give the DATA ERROR message. Unlike XTOALFT, alpha string inputs are allowed. If X contains an alpha string, the string is appended to the right end of alpha.

To use XTOART, just place either an alpha string of up to 6 characters or a number from 0 to 255 in X and execute XTOALFT. For alpha input, the string from X will be appended to the right end of alpha. For numeric input, the ASCII equivalent of the number will be appended as the last character in alpha. If the alpha register was full (had 24 characters) then the leftmost character will be deleted. Alpha inputs can cause up to 6 characters to be lost from the left end of alpha. In either case, no warning tone will be sounded.

An example of the usage of XTOART is given in the MAKEBC section.

Usage Summary for XTOART

Inputs:

X register: number from 0 to 255, or an alpha string

Alpha register: none

LaserJet differences: none

CHKSUM

The CHecKSUM function is part of SKWIDBC's custom barcode toolkit. Most types of barcode require a checksum byte to be present. This byte is used by the wand to confirm that all the bytes were read correcctly.

CHKSUM computes the checksum for the string of bytes in alpha, then appends

it as the leftmost character in alpha. The checksum covers all characters in alpha and is of the end-around carry type. To calculate a checksum like this, SKWIDBC adds the ASCII values of all the characters in alpha, then takes the result MOD 255. If the answer is 0, then 255 is substituted.

CHKSUM will only calculate a checksum if there are less than 16 characters in alpha. If there are more characters present, the ALPHA > 15 error is generated. This constraint is imposed because barcodes with more than 16 infomation bytes cannot be scanned by the HP-41's Wand. Also, if alpha is empty an ALPHA EMPTY error comes up.

An example of the usage of CHKSUM will be given in the next section.

Usage Summary for CHKSUM

Inputs:

X register: none

Alpha register: from one to fifteen characters.

LaserJet differences: none

MAKEBC

The MAKE BarCode function is the main function in SKWIDBC's custom barcode toolkit. MAKEBC allows you to make any barcode you want, from a byte string in alpha.

WARNING: Unlike SKWIDBC's other barcode printing functions, MAKEBC doesn't do a lot of fancy checking to see that the result will be a usable barcode. In fact, one of the examples given below actually causes MEMORY LOST when scanned. (Don't worry, we will warn you.) So be very careful with MAKEBC.

MAKEBC takes your bytes from alpha and prints them as a barcode row. If there are more than 16 characters in alpha the ALPHA > 16 error message is given. If alpha is empty then the ALPHA EMPTY error message is generated.

When you use MAKEBC, you should use ACTEXT first to label your custom barcode. First accumulate the text which is to be printed above the barcode row. Then execute ACTEXT. Nothing will be printed yet, but the text is in the buffer. Next build your barcode byte sequence in alpha, using the XTOART, CHKSUM, and XTOALFT functions. Next execute MAKEBC to print the barcode. MAKEBC will send a carriage return and linefeed first to separate the text from the barcode row.

MAKEBC was put in SKWIDBC to allow you to print the few usable barcodes that the other functions in SKWIDBC do not provide for. When used in conjunction with ACTEXT, XTOART, CHKSUM, and XTOALFT you could actually produce any barcode done by the other functions in SKWIDBC, although quite a bit of work might be required.

The primary category of functions which require MAKEBC is nonprogrammable built-in functions with suffixes. This includes barcodes to set a specific SIZE, make non-synthetic key assignments, and copy programs. The following program shows the combined use of the SKWIDBC functions to form barcodes to assign functions or programs to keys, set a specific size, delete a specified number of program lines, clear a program, or copy a program.

All of the barcodes produced by the following program are of the direct execution type. This means (see the section of this manual on Barcode Structure) that the leftmost byte of the barcode is the checksum and that the second byte from the left is 64 decimal (hex 40). The byte immediately following the type byte is the function byte: SIZE, ASN, etc. To compute the byte values for these functions just take the corresponding LBL from the PAPERKB section and add 1. Thus the byte value for SIZE is 6, ASN is 15, and so on. The documentation accompanying the program below provides instructions on how to use each subprogram.

01*LBL "ASNBC" To produce barcode which assigns a key, just place the name of the program or function to be assigned in alpha and the row/column keycode (10 times row number + column number, negative for a shifted key) in X. Then execute this subprogram. The keycode is the same one which appears when you use ASN manually and press and hold the key to be assigned. If the alpha register is empty, the

	resulting barcode will cancel the key assignment to the specified key, the same as if you manually executed ASN ALPHA ALPHA.
02 ASTO 00	Save the six leftmost characters of the key assignment to register 00.
03 ASHF	Shift them off.
04 ASTO 01	Save the last character of the assignment if there is one. No check is made to see if the function or program actually exists.
05 "ASN "	Put ASN into alpha for use in labeling the barcode.
06 ARCL 00	Get back the name of the function or program.
07 ARCL 01	
08 "↦ "	Append a space.
09 STO 02	Store the keycode in register 02.
10 RCLFLAG	Recall the settings of flags 0 to 43. This step requires Extended Functions, and is only necessary if you want to preserve your flag settings.
11 FIX 0	Set display mode to 0 fractional digits.
12 CF 29	We don't want the decimal point either.
13 ARCL Y	Recall the keycode to alpha.
14 STOFLAG	Restore the flag settings. (Delete this line if you deleted the RCLFLAG at line 10.)
15 ACTEXT	Accumulate alpha to the print buffer. This will be the heading for the barcode.
16 X<>Y	Put the keycode back to X.
	Lines 17-38 convert the decimal row/column keycode 10r+c into a hexadecimal code 16r+c, plus hex 80 for a shifted key.
17 10	Divide the keycode by 10 to get the row number to the integer part of X.
18 /	
19 INT	Isolate the row number.
20 0	
21 X<>Y	
22 16	Multiply the row number by 16 to put it into hex format.
23 *	
24 128	
25 X<Y?	Was the row number 9 or greater?

26 SF 99	If so, give the error message NONEXISTENT.
27 RDN	Put the row number back in X.
28 X<0?	Is this for a shifted key?
29 R↑	If so, put the 128 back in X and subtract it from the row number (we are actually increasing the row number by 8).
30 -	
31 ABS	Take the absolute value.
32 RCL 02	Recall the keycode.
33 ABS	Take its absolute value.
34 10	Isolate the column number.
35 MOD	
36 X=0?	Is the column number 0?
37 SF 99	If so, give the error message NONEXISTENT.
38 +	This result is the hexadecimal code 16r+c
39 CLA	Clear alpha to start with a clean slate.
40 ARCL 00	Get the function name.
41 ARCL 01	
42 XTOALFT	Place the keycode byte on the left end of alpha. If you get a "DATA ERROR" here it's because you wanted a shifted keycode with a row of greater than 8.
43 15	Place the ASN function code in X.
44 GTO 01	Jump to LBL 01, where the function code is placed at the left end of alpha, then a checksum is calculated and the barcode is printed.
45*LBL "SIZEBC"	To produce SIZE barcode, just place the number of data registers desired in X, then execute this subprogram.
46 "SIZE "	Place the function name in alpha.
47 6	Store the function code in register 00.
48 STO 00	
49 RDN	Bring the number of data registers back to X.
50 319	Place the maximum number of data registers in X.
51*LBL 02	Entry point to be used by "DELBC".
52 X<Y?	Is the requested size to great?
53 SF 99	If so, give the error message NONEXISTENT.
54 X<>Y	Bring the requested size back to X.
55 RCLFLAG	Recall the current flag settings to X.

56 FIX 0	Set display mode to eliminate any fractional portion.
57 CF 29	Set display mode to eliminate the decimal point.
58 ARCL Y	Append the requested size to alpha.
59 STOFLAG	Restore the previous flag settings.
60 X<>Y	Bring the requested size back to X.
61 ACTEXT	Accumulate the barcode title from alpha.
62 STO Y	Store the requested size in Y.
63 256	Here we shall change the size into a hex number by isolating the lower 2 digits of the hexcode using MOD.
64 MOD	This is the code for the upper hex digits.
65 X<>Y	Bring the original number back to X.
66 LASTX	Put 256 back in X.
67 /	Isolate the upper hex digits.
68 CLA	Clear alpha to start with a clean slate.
69 RCL 00	Get the function number.
70 XTOART	Place it in alpha.
71 RDN	Bring the upper hex digits to X.
72 XTOART	Place them to the right of the function code.
73 X<>Y	Get the lower digits of the size.
74 XTOART	Put them in alpha.
75 GTO 04	Go calculate the checksum and print the barcode.
76*LBL "DELBC"	To produce barcode for the function DEL nnn, place the number nnn of lines to delete in X and execute this subprogram.
77 "DEL "	Place "DEL " in alpha.
78 2	Put the delete function code in register 00.
79 STO 00	
80 RDN	Bring the number of lines to delete into X.
81 256	
82 X↑2	Put the maximum number of lines in X (65535).
83 1	
84 -	
85 GTO 02	Continue as in the SIZEBC section.
86*LBL "CLPBC"	To produce barcode for the function CLP "pname" to clear the program pname, just place the program name in alpha and execute this subprogram. No check is made to see if

	the program exists before the barcode is printed.
87 4	Put the clear program function number in X.
88 ASTO 00	Store the first six characters of the program name in 00.
89 ASHF	Remove these 6 characters.
90 ASTO 01	Store the seventh character if there is one.
91 "CLP "	Place "CLP " to alpha.
92 GTO 03	Jump to the entry point in the COPYBC section.
93*LBL "COPYBC"	To produce barcode for the function COPY "pname" (to copy the program pname from a plug-in ROM), just place the name of the program in alpha and execute this subprogram.
94 3	Put the function number for copy program in X.
95 ASTO 00	Store the first six characters of the program name in 00.
96 ASHF	Remove the 6 characters.
97 ASTO 01	Store the 7th character if there is one.
98 "COPY "	Place "COPY " in alpha.
99*LBL 03	Entry point for CLPBC.
100 ARCL 00	Append the program name to alpha.
101 ARCL 01	
102 ACTEXT	Copy the barcode description from alpha to the printer.
103 CLA	Clear alpha to start with a clean slate.
104 ARCL 00	Place the program name in alpha again.
105 ARCL 01	
106*LBL 01	Entry point for ASNBC.
107 XTOALFT	Append the function code to the left end of alpha.
108*LBL 04	Entry point for SIZEBC, DELBC.
109 64	Put the type code for direct execution barcode in X.
110 XTOALFT	Append it to the left end of alpha.
111 CHKSUM	Calculate a checksum of the bytes and append it at the left end of alpha.
112 MAKEBC	Print the alpha bytes as barcode.
113 END	All done!!

Barcode for this program is provided in Appendix E.

Here is another demonstration program. This one will allow you to print barcode for any alpha string, even if it has all ASCII characters of value less than 16.

```
01*LBL "ALPHABC"
02 ALENG        Get the length of the alpha string.
03 ACTEXT       Accumulate alpha to the printer.
04 112          Place 70 Hex into X (this is 112 in decimal).
05 +            Add in the number of characters.
06 XTOALFT      Place the type and number of characters to alpha.
07 CHKSUM       Calculate the checksum.
08 MAKEBC       Print the barcode.
09 END          End of program.
```

To use ALPHABC, place the characters to be barcoded in alpha and execute ALPHABC. You will get the barcode, titled with the unquoted alpha string.

Now let's do some examples.

EXAMPLE 15

We shall create a barcode which assigns the ABC function to the LN key. The keycode for the LN key is 15 (row 1 and column 5). Place 15 in X and ABC in alpha. Now execute ASNBC. The barcode will be printed with "ASN ABC 15" printed above it. Now scan in the barcode and ABC will be assigned to the LN key.

If you wish to make an assignment to a shifted key just put a negative number in X. For example, -44 corresponds to the CLX/A key (the shifted backarrow key).

EXAMPLE 16

Frequently a barcoded program requires a certain number of data registers to operate. It is quite convenient to include a barcode which will set the correct size (number of data registers) along with the program barcode so the user does not have to search the instructions for the correct size. The ASNBC program requires a SIZE of 3. To make this easier, we shall print a

direct execution size barcode to give us 3 data registers. Just place 3 in X and execute SIZEBC. The description SIZE 3 will be printed above the barcode.

To test this barcode, set the size of your HP-41 to 100 by executing SIZE and filling in 100. Now scan the SIZE 3 barcode and try to recall register 05. You will get NONEXISTENT . If you try to create a barcode for setting a size greater than 319 then you will get a NONEXISTENT error at line 53 since the HP-41 only has 319 data registers.

EXAMPLE 17

If you are in the habit of deleting lines from programs you may want to print barcode for the function DEL nnn for various values of nnn. Put the value of nnn in X, then execute DELBC. The barcode will be printed with the designation DEL followed by a space and then the number nnn.

A DEL 1999 barcode may be useful if you are always reading in programs from a tape and then deleting the program from a certain step to the end. The value 1999 is safe, since the END cannot be deleted. To create this barcode place 1999 in X and execute DELBC. Now whenever you need to delete the rest of a program just scan this barcode.

EXAMPLE 18

If you are always clearing a particular program from memory, you may want to have a CLP "pname" barcode.

As an example, you are always clearing a program called TEMP. To print this function in barcode, just put TEMP into alpha and execute CLPBC. Now whenever you wish to clear TEMP, just scan the barcode and it will be cleared and packed.

EXAMPLE 19

If you have a favorite program that you are always copying out of a ROM module into RAM you can print the function COPY "pname" in direct execution barcode by using the COPYBC program. Just place the name of the ROM program

in alpha and execute COPYBC. Now whenever you need to copy the program into your machine scan the barcode and it will be done.

EXAMPLE 20

Here is the MEMORY LOST barcode example promised earlier. When this barcode is scanned in program mode with the 1F version of the Wand, a MEMORY LOST will occur after a dramatic pause. This shows what can happen if you decide to use MAKEBC to create unknown barcodes.

1.) Place "MEMORY LOST" in alpha and execute ACTEXT.
2.) Clear alpha and place 106 (hexadecimal 6A) in X.
3.) Execute XTOART.
4.) Execute XTOART again.
5.) Execute CHKSUM.

The barcode bytes are now in alpha. Now execute MAKEBC. This barcode produced is of the numeric type. With either version of the Wand, it will place 6 in the X register if you are in run mode. If you have the 1E version of the Wand, it will also insert a 6 as a program step in PRGM mode. However, if you scan this barcode in PRGM mode with the 1F version of the Wand, the calculator will sound a tone (to signify the row was correctly read in) and then the calculator will pause before displaying the dreaded MEMORY LOST.

Usage Summary for MAKEBC

Inputs:
X register: none
Alpha register: from 1 to 16 bytes to be barcoded

LaserJet differences: Flag 0 must be set.

FRMFEED

The FoRM FEED function sends the form feed character (hexadecimal 0C) to the device on which barcode printing occurs. If flag 0 is clear, it will be sent to the ThinkJet. With flag 0 set the form feed goes to the LaserJet.

The FRMFEED function was included to allow you to eject a page of paper from the printer directly from the HP-41 keyboard, either manually or under program control. It is most useful when printing several barcode functions which each only use one row. After barcoding 20 or so of these you may use FRMFEED to start at the top of the next page so a barcode row will not be printed half on the bottom of one page and half on the top of the next page. (This would normally be a problem only with the ThinkJet.)

Usage Summary for FRMFEED

Inputs:

X register: none

Alpha register: none

LaserJet differences: Flag 0 must be set.

XBC

The X register BarCode function produces numeric barcode if X is numeric or alpha replace barcode (the same barcode produced by ABC) if X contains an alpha string.

If X contains an alpha string and all of the characters have an ASCII value of less than 16 then the DATA ERROR message comes up. (This case corresponds to an alpha replace barcode that will crash the HP-41.) Otherwise the alpha string is printed above the barcode and surrounded by quotes just as with ABC.

For numeric input, the number is printed above the barcode row. The current display setting (FIX, SCI, ENG) is ignored when the number is printed. So the SCIentific 9 setting with 9.9900000 02 in X will be printed as 999 above

the barcode row. Similarly with the number 12.64785 in X and a FIX 2 setting the printout would be 12.64785 not, 12.65. XBC assumes you want all of X barcoded. If not, use the HP-41's RND (round) function first.

Fractional numbers (those less than one and greater than 0) are always printed (and barcoded) in scientific notation. So 0.0987098 will be barcoded as 9.887098E-2.

The radix (integer/fraction separator) printed above the barcode will match the status of Flag 28. If flag 28 is set, the decimal point is the radix; if flag 28 is clear, the comma is the radix (European number format).

EXAMPLE 21

To print the number -8518.296 in barcode, put this number in X and execute XBC. The number will be printed above the barcode.

EXAMPLE 22

Suppose you want to print barcode for all of the HP-41's data registers. The following program uses XBC to accomplish this task. (Delete line 02 and input the SIZE if you don't have Extended Functions.)

```
01*LBL"REGTOBC  Name of the program.
02 SIZE?        Extended function to get the current SIZE.
03 1            Subtract 1 from this.
04 -
05 1 E3         Divide this by 1000.
06 /
```

We now have a counter of the form bbb.eee in X, where bbb is the first register (00) and eee is the last register.

```
07 SIGN         Place the counter into L.
08*LBL 00       Start the loop to print barcode.
09 RCL IND L    Recall the data to X.
10 XBC          Print the data as barcode.
```

The DATA ERROR message could be generated at line 10 if one of the registers has a null alpha string in it.

11 ISG L	Increment the counter to the next register and skip the next step if we are done.
12 GTO 00	Start the loop over.
13 END	All done.

The barcode generated by this program will be of the numeric type and/or the alpha replace variety if there is alpha data in the registers.

The two functions REGBC and REGBCX have been provided to accomplish much the same thing as this "REGTOBC" program. The only differences are the speed (much faster) and the fact that the sequenced barcode type is not readable by the 1E version of the wand.

Usage Summary for XBC

Inputs:

X register: data to be barcoded
Alpha register: none
Flag 28: set or clear for selection of point or comma as radix

LaserJet differences: Flag 0 must be set.

REGBC and REGBCX

The REGister BarCode and REGister BarCode by X functions allow you to easily place the contents of data storage registers into barcode. The barcode type produced by REGBC and REGBCX is sequenced numeric or sequenced alpha replace (if the register contains an alpha string). This barcode cannot be read by the 1E version of the Wand, and the error message W:TYPE ERR will result.

The advantage of sequenced type barcode is that when you use the WNDDTX function in the Wand, the barcode rows must be read in a specified order. This makes it easy to transfer data as barcode to a friend and have him read

it in the correct order.

The use of REGBC is the same as for the PRREG function in the printer module. Just execute REGBC and all of the data registers will be barcoded. If there are no data registers the NONEXISTENT error message comes up. The REGBCX function works like PRREGX in that an input of the type bbb.eee, where bbb is the first register and eee is the ending register, is required. Numbers to the right of the eee field are ignored. If either the bbb or eee register does not exist, then the NONEXISTENT error message is generated. If bbb is greater than eee, then only the bbb register is barcoded.

If one of the registers contains an alpha string whose characters all have an ASCII value of less than 16, this register is skipped and the sequence number which goes along with that register is also skipped. When you scan the resulting barcodes, you should press the SST key to step over the missing sequence number.

When the barcodes are printed, the contents of the register are printed above the barcode row preceded by "SEQ #nn" where the nn is the number in the sequence. Sequence numbering starts at zero and continues until the functions are finished printing. If you want to skip a sequenced barcode row during WNDDTX just hit the SST key to get to the next row. During barcoding the word "REGISTER " is placed into the display followed by the number of the register which is currently being printed.

EXAMPLE 23

We shall barcode the contents of all the data registers. First set the number of data registers to 150 using the SIZE function. Use the following program to fill the registers with data.

```
01*LBL "FILL"
02 .149
03*LBL 01
04 STO IND X
05 ISG X
06 GTO 01
07 END
```

Now enter alpha mode and key in SKWID then ASTO this in register 12. Now clear alpha and ASTO the empty contents of alpha in register 16. Now execute REGBC and watch it print. The first 16 data registers will be printed. The seventeenth (number 16) will be skipped since it contains all characters whose ASCII value is less than 16. After register 19 is printed, a form feed is automatically sent to the printer and barcoding continues on the top of the next page. This will continue until all 150 registers have been barcoded (seven and one half pages). If you wish to stop the printing before it is done, just hit either the R/S or ON key and barcoding will stop when the row which is currently being printed is finished.

If you don't want to barcode the data in all the registers, you may use the REGBCX function to just print part of the data. Barcoding need not start at register zero or end at the last register.

EXAMPLE 24

Let's barcode the contents of registers 15 to 69. To do this place 15.069 in the X register and execute REGBCX. Printing will be done the same as with REGBC, at 20 rows per page, skipping registers which have all of their alpha characters less than 16. Sequencing starts at zero even if the starting register is not zero.

Usage Summary for REGBC and REGBCX

Inputs:

X register: none for REGBC
bbb.eee for REGBCX, where bbb is start register and eee is the ending register.

Alpha register: none

Flag 28: set or clear for selection of point or comma as radix

LaserJet differences: Flag 0 must be set.

COLBC

The COLumn BarCode function allows you to print numeric integer barcodes on labels. With the ThinkJet printer the labels are arranged on a page two across by twelve down. This label paper is sold by Hewlett Packard; the part number is HP51630L. You get about 100 fanfold sheets with tractor feed holes.

For the LaserJet we have found label paper which may be fed through the paper bin, just like regular paper. Most ordinary label papers must be fed through the LaserJet manual feed slot. The label sheet is 8 1/2 inches by 11 inches and has three labels across by eleven down. Each label is 2 3/4 inches wide by 1 inch high. These labels are not manufactured by Hewlett Packard. They come in 100-sheet packages and are available from Imaging Products, 12696 Rockhaven Road, Chesterland, OH 44026 USA, phone number (216) 285-2813.

The inputs for COLBC are a high number in Y and the lower integer in X. COLBC will print the barcodes for all of the numbers from X to Y. So for example, if 15 is placed in Y and 4 in X, then the barcodes for 4, 5, on up to 15 will be printed on the labels. If you make a mistake and reverse the X and Y inputs, COLBC will give you the error message X GT.Y (X is greater than Y).

When using a LaserJet series printer, the last page of barcode is ejected from the printer even if all of the labels are not used. The label stock should be placed face down in the paper tray when using a LaserJet or LaserJet Plus. If a LaserJet Series II is used, the label stock should be placed face up in the paper tray.

When you are using the ThinkJet printer, about 3/16 of an inch of the first label you are going to print on should be showing. This means you will have to waste the first row of labels on the sheet so they may be placed under the rollers.

EXAMPLE 25

This example is for ThinkJet users. In this example we shall print the barcodes for the numbers 23 to 34. First clear flag 00, then place 34 in X and press the ENTER key. Now place 23 in X and execute COLBC. The numbers will be printed above the barcodes. If the last row has only one barcode to print, it will be printed on the left label and the right one will be left blank.

EXAMPLE 26

This example is for LaserJet users. To use COLBC with the LaserJet you must replace the regular paper in the paper bin with the label sheets. When using a LaserJet or LaserJet Plus, the labels should be placed facing down in the paper tray. With the LaserJet series II the labels should be placed face up. Place 45 in X and press ENTER↑. Then place 13 in X. Execute COLBC.

After the first sheet is ejected, you will notice that the last row of labels is not used. This is caused by the LaserJet not being able to print far enough down on the label.

Usage Summary for COLBC

Inputs:

X register: first integer to be printed
Y register: last integer to be printed
Alpha register: none

LaserJet differences: Flag 0 must be set.

SELF TEST EXAMPLE PROGRAM

Below is a self test program which makes use of most of the functions in SKWIDBC. If you want to check out all of the different kinds of barcode produced by SKWIDBC, just execute BCTEST and wait for the program to finish. Remember that flag 0 must be clear if you are using a ThinkJet and set for LaserJet use. The output for BCTEST will have the barcode type printed above the barcode row and then an example of the barcode.

Barcode for this program is given in Appendix E. If you decide to key the program in, you will need to assign DIREXBC and PAPERKB to keys. Turn off the HP-41, remove the SKWIDBC Module, turn the calculator back on, then use the assigned keys to enter lines 17 and 20. Then turn off the HP-41, reinsert SKWIDBC, turn on, and key in the rest of the program.

Program step	Description
01*LBL "BCTEST"	Name of program
02 CF 17	Ensure that OUTA (line 47) adds CR/LF.
03 "ALPHA REPLACE"	Put the barcode type into alpha.
04 XEQ 01	The LBL 01 subroutine accumulates alpha to the printer and then appends " BARCODE" to this. A carriage return and line feed are then sent.
05 "HEWLETT"	Place HEWLETT in alpha.
06*LBL"Q"	LBL "Q" for DIREXBC and PAPERKB.
07 ABC	Print alpha replace barcode for HEWLETT.
08 "ALPHA APPEND"	Alpha append barcode type.
09 XEQ 01	Accumulate this to the printer.
10 "PACKARD"	Place PACKARD in alpha.
11 38	Place the "&" symbol to the left of alpha.
12 XTOALFT	
13 AABC	Print the alpha append barcode.
14 "DIRECT EXECUTIO"	Put DIRECT EXECUTIO in alpha.
15 "┼N"	Append the N.
16 XEQ 01	Accumulate this to the printer.
17 DIREXBC	Print direct execution barcode of ABC.
18 "PAPER KEYBOARD"	Put PAPER KEYBOARD in alpha.

19 XEQ 01	Accumulate this to the printer.
20 PAPERKB	Print ABC as a paper keyboard barcode.
21 "SEQUENCED"	Sequenced barcode.
22 ASTO 02	Store SEQUEN in register 2.
23 XEQ 01	Accumulate this to the printer.
24 .01	Barcode registers 0 to 10.
25 REGBCX	Barcode the registers.
26 "NUMERIC"	Say we will print numeric barcode.
27 XEQ 01	Accumulate this to the printer.
28 12.696345	Number to barcode.
29 XBC	Barcode the number.
30 FRMFEED	Go to the top of the next page.
31 "BCTEST"	Name of this program for PROGBC.
32 12	Print 12 program bytes per line. Could use the sequence 12, FS? 00, 13 to allow 13 bytes if printing on a LaserJet.
33 PROGBC	Barcode the program.
34 "COLUMN"	We will now print some column barcode.
35 XEQ 01	Accumulate this to alpha.
36 13	Make 13 the high number.
37 ENTER↑	
38 0	0 will be the lower limit.
39 COLBC	Barcode in columns.
40 RTN	End of barcoding.
41*LBL 01	LBL 01 subroutine.
42 10	
43 XTOALFT	Add a line feed character at the left of alpha.
44 13	
45 XTOALFT	Add a carriage return at the left of alpha.
46 "⟩ BARCODE"	Append " BARCODE" to alpha.
47 OUTA	Print CR/LF (type) BARCODE CR/LF. The second CR/LF is because flag 17 is clear (line 02). **<u>CAUTION</u>: OUTA will normally send data to your printer. If it doesn't, either remove other HP-IL devices or place the proper Device ID in alpha, and SELECT the device.**
48 END	End of subroutine.

HP-41 BARCODE STRUCTURE

This section is especially for those of you who want to poke around with MAKEBC, or who need to print HP-41 barcode directly from a computer. If you don't want to do these things, skip this section. If you need more information than is provided here, consult HP's book "Creating Your Own Barcode".

HP-41 barcode is a series of wide and narrow bars separated by spaces. The wide bars (each equivalent to a binary 1) are twice as thick as the thin bars (binary 0). The spaces are the same width as the narrow bars. The bars are coded into a binary format. So the byte F2 would be 11110010 with the 1's being wide bars and the 0's the narrow bars.

The barcode row always begins with two narrow bars called the start bits. The last two bars of every row are a wide and then narrow bar called the stop bits.

Barcode rows may have up to 16 bytes (128 bars plus 2 start bars and 2 stop bars, for a total of 132 bars). For all barcode types, except paper keyboard (type 5), the first eight bars after the start bits are the checksum. The checksum is of the end-around carry type. It is calculated by adding all of the bytes, except the checksum, and taking the total MOD 255 to get the actual checksum. If this answer is 0, then 255 is substituted. The next 4 bars are the barcode type which may be from 1 to F (15). Type zero is not allowed. The rest of the barcode row is data which depends on the barcode type. Below is a description of how the barcode rows are structured.

Type	Description
1	Program barcode - The first eight bars are the checksum. The next 4 bars are always set to 0001. Bars 13 to 16 are the (row number - 1) MOD 16. Therefore the first row number is (1 - 1) mod 16 which equals 0. The next 4 bars are the number of program bytes at the beginning of the row which are part of an instruction from the preceding row.

This is provided in case an instruction is split between two rows.
Bars 21 to 24 are the number of bytes at the end of the row which combine with bytes at the beginning of the next row to form an instruction.
The rest of the barcode row is the program bytes. There may be from 1 to 13 program bytes per row.

2 Private Program barcode - This is exactly the same as type 1 except that the 4 type bars are set to 0010.

3 Type 3 barcode cannot be read in by the wand.

4 Direct Execution - The first eight bars are the checksum.
The next 8 bars are always 01000000 (40 hex).
The rest of the barcode is the instruction which is to be executed. These bytes are the same as the ones in program memory i.e., STO 99 would be 10010001 01100101 (91 65 in hex). The only exception to this is alpha GTOs and XEQs. With these instructions the second byte is left out. So instead of 1D F3 41 42 43 for GTO "ABC" you would place 1D 41 42 43 as the bytes in the barcode row.

5 Type 5 is Paper Keyboard barcode. Paper keyboard barcode is any barcode row which is either 1 or 2 bytes long. The one-byte barcodes are all numeric entry barcodes. These are formatted as follows: Bars 5 to 8 are the number in binary, i.e., 3 would be 0011. Bars 1 to 4 are the mirror image of bars 5 to 8, so 0011 would be 1100 and the barcode row becomes 11000011. If bars 5 to 8 are 1010, then the decimal point is the input. 1011 denotes the EEX key, 1100 denotes the CHS key, 1101 denotes the backarrow key, and 1110 denotes the R/S key. A value of 1111 in bars 5 to 8 is considered illegal.
Barcodes which are two bytes cover the rest of the instructions which can be keyed in from the keyboard. Bars 1 to 4 of these barcodes are a four bit end-around checksum. This is computed the same as the eight bit checksum for other barcode types except only 4 bits at a time are added instead of eight, and the modulus is 15 instead of 255. Bars 5 to 8 are the type and the eight bars are the instruction data. Below is a table for all of the possible types.

Nybble	1	2	3	4
Programmable	Checksum	0 000	Function	
Alpha characters	Checksum	0 001	2 times upper four bits of character	Lower 4 bits of character
Indirect	1010	0 010	1000	0000
Not used		0 011		
Non-programmable	Checksum	0 100	0000	0000 to 1111
XROM's	Checksum	1 Lower 3 bits of 1st byte	second byte of function	

For example the STO instruction would be coded as A0 91, where the A is the checksum, 0 is nybble 2 and 91 is the function code for STO.

With alpha characters, the four most significant bits of the character are multiplied by two. This is the reason characters greater than 127 (7F in hex) cannot be placed into barcode, since the upper bit is lost during multiplication. For the "A" character, the barcode is A1 81. The character code for "A" is 41, but the 4 is multiplied by two.

The indirect barcode is the only way to simulate the shift key to get IND postfixes.

Non-programmable functions just place the function number in nybble 4 and calculate the appropriate checksum.

The XROM functions have the leftmost bit of nybble two set to 1. The other 3 bits are the three least significant bits of the first byte of the XROM function code. Nybbles 3 and 4 are the second byte of the XROM function code.

Type Description

6 Numeric - The first eight bars are the checksum and the next 4 bars constitute the type which is always 0110 (6). The rest of the barcode row is data. Each number is contained in 4 bits and is set to its decimal equivalent. So a 0001 would place a 1 as part of the number and so on. The decimal point is represented by 1011. The minus sign is 1101 and the exponent is 1110. 1010 is a filler nybble in case there are an even number of numeric characters. However this filler byte must be at the beginning, before any numbers, or you will get a MEMORY LOST with the 1F version of the Wand in PRGM mode.

For example the number -123.765 E3 is barcoded as 6A D1 23 B7 65 E3. The first 6 is the type. The A is a filler since we must have an odd number of nybbles. If the number is only one digit long, for example 2, then two filler nybbles must be placed before the number. To barcode the number 2, the bytes would be 6A A2.

7 Alpha replace - The first eight bars are the checksum. The next four are the type, which is always 0111. The next four bars are the number of alpha characters in the barcode. The rest of the barcode row is the data containing the alpha characters.

8 Alpha append barcode - This is exactly the same as alpha replace barcode except that the type bars are changed to 1000.

The following three types of barcode are only readable by the 1F version of the Wand. When using the WNDDTX function, you may mix the data types as long as the sequence number is the next one in line.

9 Numeric sequenced data - The first 8 bars are the checksum and the next 4 are the type (1001). The next 12 bars are a sequence number which may be from 0 (0000 00000000) to 4095 (1111 11111111). Then comes the numeric data, which is of the same format as with type 6 (numeric) barcode.

10 Alpha replace sequenced - The first eight bars are the checksum and the next 4 are the type which is 1010. Then there is a 12 bit sequence number format the same as above. Then there can be up to 13 alpha characters.

11 Alpha append sequenced - Same as type 10 except for bars 8 to 11 which become 1011.

Barcode types 12 to 15 are not used by the wand and will give a "W:TYPE ERR" if read in with the wand.

APPENDIX A

SKWIDBC Error messages

Message	Functions	Cause
NO IL PR. ROM	All barcode functions	HP-IL module not plugged in, or switch on back of module set to disable.
NO THINKJET	All barcode functions	No ThinkJet Printer in the loop.
NO INTERFACE	All barcode functions	No interface class device (i.e. an HP-IL to RS-232 interface) in the loop.
ALPHA EMPTY	ABC AABC CHKSUM MAKEBC	No characters in the alpha register.
ALPHA > 14	ABC AABC	More than 14 characters in the alpha register.
ALPHA > 15	CHKSUM	More than 15 characters in the alpha register.
ALPHA > 16	MAKEBC	More than 16 characters in the alpha register.
LINE TOO LG.	All barcode functions	The number of graphics bytes to print a barcode row is more than will fit on the graphics area of the ThinkJet.
NO LBL Q	DIREXBC PAPERKB	No program in memory with a LBL "Q".
ILLEGAL INST.	DIREXBC PAPERKB	Instruction which cannot be printed by the barcode type.

ILL. CHR.	PAPERKB	The step after the LBL "Q" is an alpha string and its 1st character is greater than 127 ASCII.
NONEXISTENT	All	The module is not plugged in.
	PROGBC PPROGBC PROGRW PPROGRW	The program named in alpha is not present in the calculator.
	COMPILE	The Program name keyed in at the prompt is not in the calculator.
	REGBC	There are no data registers.
	REGBCX	At least one of the registers specified does not exist.
ROM	PROGBC PPROGBC PROGRW PPROGRW COMPILE	The program named in alpha is in a plug-in module.
PRIVATE	PROGBC PPROGBC PROGRW PPROGRW	The named program is private.
DATA ERROR X	PROGBC PPROGBC PROGRW PPROGRW	The number in X is greater than 12 (flag 00 clear) or 13 (flag 00 set).
	COLBC	X is greater than 999,999,999 (flag 00 clear) or 9,999,999,999 (flag 00 set).
DATA ERROR Y	PROGRW PPROGRW	The row number to start printing at is greater than the number of barcode rows needed to print the program.
	COLBC	Y is greater than 999,999,999 (flag 00 clear) or 9,999,999,999 (flag 00 set).

PACKING/ TRY AGAIN	PROGBC PPROGBC PROGRW PPROGRW COMPEND COMPILE	There is not enough room in memory to place a nonpermanent END on the program.
NO LBL nn	COMPILE	There is no LBL corresponding to the GTO or XEQ. The label number is nn.
DATA ERROR	XTOALFT XTOART	X is greater than 255.
	ABC AABC XBC	All of a group of 6 characters in alpha have ASCII values of less than 16. There is an alpha string in X and all of the characters have an ASCII value of less than 16.
X GT. Y	COLBC	The number in X is greater than the one in Y.

APPENDIX B

XROM Numbers

This is a list of the XROM numbers which correspond to each function. These numbers will show up if the SKWIDBC module is removed from the HP-41 and one of the SKWIDBC functions is in a program or assigned to a key.

Function	XROM number
-SKWIDBC	08,00
ABC	08,01
AABC	08,02
ACTEXT	08,03
CHKSUM	08,04
COLBC	08,05
*COMPEND	08,06
COMPOFF	08,07
*COMPILE	08.08
*DIREXBC	08.09
FRMFEED	08.10
MAKEBC	08,11
*PAPERKB	08,12
PPROGBC	08,13
PPROGRW	08,14
PROGBC	08,15
PROGRW	08,16
REGBC	08,17
REGBCX	08,18
XBC	08,19
XTOALFT	08,20
XTOART	08,21

The asterisks indicate non-programmable functions.

APPENDIX C

Interfacing a LaserJet series printer to the HP-41

There are four ways in which you can connect the HP-41 to a LaserJet printer. All of them use HP-IL and therefore require an HP-IL module. The first interfacing method is an HP-IL to RS-232C interface (HP part number HP82164A), HP-IL to GPIO interface (HP82165A), the HP-IL converter (HP82166A), and the HP-IL link card for IBM PC and compatibles (HP82973A). The HP-IL to RS-232C interface is also manufactured by several other companies, and these interfaces are compatible with the one made by Hewlett-Packard. Although the HP-IL converter is no longer made, Software Operations and Systems is developing an interface which replaces this part with some significant enhancements.

Interfacing the LaserJet using an HP-IL to RS-232C interface

The original LaserJet came with only an RS-232C interface, so if this is what you are using then no change of the interface is necessary. The dual-interface LaserJet+ printer requires you to take the back plate off the printer in order to select the RS-232C interface for communications. If you left the switches in their default settings, then the RS-232C interface is the one which is being used. All of the other factory settings also match the default settings of the HP-IL to RS-232C interface. All that has to be done is connect the interface to the LaserJet with an RS-232C cable.

For those of you whose printer is not in the factory setting, the following switch settings are needed (see page F-4 of the LaserJet Operator's Reference Manual for the location of these switches).

Switch number	on/off
1	off
2	on
3	off
4	on
5	off
6	off (on for LaserJet+)
7	off
8	on

This gives the following configuration:

Start bits	1
Data bits	8
Stop bits	1
Parity	None
Baud rate	9600

The cable should be a "straight-through" cable, with all 25 wires on one end going to the same pins on the other end. One end of the cable should have a male 25 pin D-shaped DB-25 connector and the other end should be a female DB-25 connector.

The LaserJet Series II printer can be configured from the keyboard on the front of the printer. See the description of printer configuration in either the "Getting Started" book for the Series II or the User's Manual for how to set the configuration parameters to those listed above. The cable used is the same as for the LaserJet and LaserJet+.

The HP-IL to RS-232C interface has the configuration listed above as its default setting, so no extra setup is necessary. However there is a jumper inside the interface which determines configuration of the output. This is the Data Terminal Equipment (DTE) and Data Communications Equipment (DCE) jumper. It should be set to DCE. See page 21 of the HP82164A Owner's Manual.

Other RS-232 Interface Devices

There are several companies other than Hewlett-Packard which manufacture a direct replacement for the HP82164A HP-IL to RS-232 interface. Below is a listing of all the known products and a description of any differences.

A company called Hand Held Products makes an HP-IL to RS-232 interface which is functionally identical to the HP's. It comes in the same box as the HP RS-232 interface and the exterior layout is the same. However they have added two very nice features. The first is that the interface runs off of

Ni-Cad batteries. This allows the unit to run for up to 15 hours without being recharged. The recharger is the standard HP charger (part number 82059D). The second additional feature is the addition of an 8190 byte buffer. The HP interface has 193 bytes of buffer space (84 transmit, 109 receive).

This interface may be obtain from: Hand Held Products, Inc.
8008 Corporate Center Drive
Charlotte, NC 28211
Phone (704) 541-1380

Firmware Corporation makes two versions of the RS-232 interface. The first is powered by an AC wall transformer and the second may be run from batteries. The two units are otherwise identical. The transmit and receive buffers are each 500 bytes.

For more information about these interfaces write to:

Firmware Corporation
605 N.W. Fifth St. Suite 2A
Corvallis, OR 97330
Phone (503) 753-9314

Corvallis MicroTechnology (CMT) makes an HP-IL to RS-232 interface which is functionally identical to HP's in every respect (including the buffer size). The only difference is that 9 pin DB connector is used instead of the 25 pin one. The pinout for this connector is the same as for serial ports on IBM PC AT computers. You may also buy this interface in the same box as their recently introduced HP-IL RAM disc. The whole package is a little bit larger than the HP-41.

For more information on this product contact CMT at:

Corvallis MicroTechnology
895 N.W. Grant Ave.
Corvallis, OR 97330
Phone (503) 752-5456

An interface which is in the design stage as this manual is being written may be manufactured by Software Operations and Systems. It will consist of

a board in the same case as Hewlett-Packard's RS-232 interface. You will be able to plug modules into this board to make it become different types of interfaces. It will come with a module which emulates the HP82165A GPIO interface. You may also purchase a module to make the device an RS-232 interface (the 9 pin variety) and one which will have the device emulate an IBM PC parallel printer port. With this last interface module you could use a readily available IBM PC parallel printer cable to hook the interface to the LaserJet.

To obtain more information on this device write to:

Software Operations and Systems
1850 East 17th St. Suite 102
Santa Ana, CA 92701

Interfacing the LaserJet using an HP-IL to GPIO interface

To enable the use of HP-IL with devices which have a parallel or "Centronics" interface, Hewlett-Packard developed the HP-IL to GPIO interface (HP82165A). You may only use this if you have a LaserJet+ or LaserJet series II printer. To connect the interface to the LaserJet a cable must be constructed with a DB-25 connector on the end which connects to the interface and a 34-pin male Centronics connector for the end which connects to the LaserJet. Below is the cable diagram.

```
  GPIO                                  LaserJet

       pin #                                pin #

RDY1   1 ---------------------------- 11   BUSY
DAVO   14 --------------------------- 1    STROBE
DAC1   2 ---+
GND    21 --------------------------- 19   GND
                                  +----- 20   GND
                                  +----- 21   GND
                                  +----- 22   GND
                                  +----- 23   GND
                                  +----- 24   GND
                                  +----- 25   GND
                                  +----- 26   GND
                                  +----- 27   GND
                                  +----- 28   GND
                                  +----- 29   GND
                                  +----- 30   GND
DA7    25 --------------------------- 9    DATA8
DA6    24 --------------------------- 8    DATA7
DA5    23 --------------------------- 7    DATA6
DA4    22 --------------------------- 6    DATA5
DA3    8 ---------------------------- 5    DATA4
DA2    7 ---------------------------- 4    DATA3
DA1    6 ---------------------------- 3    DATA2
DA0    5 ---------------------------- 2    DATA1
```

Interfacing to the HP82166A converter

A discontinued (though very useful) HP-IL interface device is the HP82166A converter. It does much the same job as the GPIO interface except that it is in a smaller package and it must be powered either by the device it is driving or some other external source. We recommend powering the interface with a set of 4 AA batteries, either rechargeable or disposable.

To connect this interface you need a 34-pin female "header" connector on one

end and a 34-pin male Centronics connector on the other end to attach to the LaserJet. Below is the necessary wiring diagram.

Converter			LaserJet	
RDY1	6	----------	11	BUSY
DAVO	5	----------	1	STROBE
DA7	25	----------	9	DATA8
DA6	24	----------	8	DATA7
DA5	23	----------	7	DATA6
DA4	22	----------	6	DATA5
DA3	8	----------	5	DATA4
DA2	7	----------	4	DATA3
DA1	6	----------	3	DATA2
DA0	5	----------	2	DATA1
DCLO	34	----------	31	RESET

Using the HP-IL Interface card

The HP-IL interface card (also known as the "Link card") is a plug-in peripheral card for IBM PC's and compatibles. PC software for its use is provided with the card. However, Hewlett-Packard and third parties have written other software drivers for the Link card. We shall describe the use of two of the extra cost pieces of software as well as the standard one.

When you install the link card, the switches on the card should be set as follows:

Switch number	on/off
1	off
2	off
3	off
4	on
5	off
6	on
7	on
8	on

This gives a peripheral address of 1700. The leftmost digit of the address is the binary equivalent of switches 1 to 4. The 7 is set by switches 5 to 8. If you already have another peripheral at this address, then change the address to one which will not interfere with another card. For instance you could change the address to 1600 by turning switch 8 off. If you do change the address, then you should replace the number 1700 in all of the following examples with the new address.

Using the HP-IL Link card with the standard software

Place the HP-IL Link disc into the A disc drive. Now type CS80 1700 and press return. The display should come up asking you to press either the 1, 2, or 3 key. You should press the 2 key. Now just ensure that the LaserJet is ready to go.

In order to use this configuration with the SKWIDBC software, the Link card must be the selected device and you must be in MANIO mode.

Using the Link card with the HP-IL Link software

In order to allow the use of the Link card with a greater variety of HP-IL computers, Hewlett-Packard came out with a new software driver called HP-IL Link (part number HP82477). To use this software just insert the disc into disc drive A and type HPILLINK. If you changed the address of your Link card when you installed it, you must type in HPILLINK /AXXXX where XXXX is the address of the interface card. Now press the F5 key to toggle the print feature on. To access the printer you will have to select the Link card and set the HP-41 to MANIO mode. To get into this state, execute AUTOIO, place PCIBM into alpha and execute FINDID. Now execute the SELECT function and then the MANIO instruction. You must set flag 00. Now you are already to start printing barcode.

Using the Link card with LINK by Southern Software

The LINK program by Southern Software is by far the best software driver for the HP-IL Interface card if you are using an HP-41, 71, or 75. If you changed the address of the Link card see page 4 of the LINK User's Manual for how to change the address in the LINK program. To use LINK insert the software disc into disc drive A and type LINK. You will be presented with a menu of the following choices:

D - Disc drive (such as 9114A or 82161A cassette drive)
P - Printer (uses the PC printer)
V - Video display
M - MS-DOS device
E - End program

"D" allows you to emulate an HP-IL mass storage device of up to 8 megabytes.

"P" redirects all data sent to the interface to the PC's current printer device. The setup for this option is the same as for the last two software drivers, except that you should put DOPRN in alpha, the interface card must be the selected device, and the HP-41 must be in MANIO mode.

"V" redirects all data to the video display. This is an emulation of the Hewlett-Packard video display with some enhancements. These include color (if you have a color monitor) and inverted characters.

"M" allows you to direct all data to any device or file on the PC. To use this option press M. You will now be asked what device or file you wish to access. If the LaserJet is connected to the PC via a parallel port then type in LPT1 at the prompt. If you are using the RS-232C interface on the LaserJet type in COM1. Now hit return and you are ready to go. There is no need to place the HP-41 into MANIO.

To exit the program, just press F10, then "E" when the main menu appears.

The LINK program may be obtained from: Southern Software, 215 Hawthorne, Houston, TX 77006 USA Phone (713) 522-6220 from 8 AM to 7PM Central time, Monday through Saturday.

APPENDIX D

Troubleshooting Tips

This section will give hints to eliminate problems you may encounter while using SKWIDBC.

Problem:
You get very large bars when using a LaserJet printer.
Solution:
Set flag 0. This error can only occur if you are in Manual I/O mode.

Problem:
You get very thin bars when using ThinkJet.
Solution:
Clear Flag 0. This error can only occur if you are in Manual I/O mode.

Problem:
You get the incorrect text character font when using a LaserJet.
Solution:
There are nondefault fonts selected. Push the on-line button to take the printer off line. Then press and hold the reset button until the reset code (07) is displayed. Press the on-line button again to put the printer on-line. If this seems too complicated, you can simply turn the printer off and back on.

Problem:
You get the error message LINE TOO LG. in the middle of printing a program.
Solution:
One of the barcode rows is too long to fit in the graphics area of the printer. Decrease the number of bytes per row (the number in X). If a 10 is placed in X this error will never occur. This can only occur while using a ThinkJet.

Problem:
The LINE TOO LG. error occurs during ABC or AABC.
Solution:
Split the alpha characters into two separate barcode rows. Make the second

an alpha append barcode. By doing this you will have to scan both rows to get the whole message into alpha.

Problem:

You get the message TRANSMIT ERR whenever a SKWIDBC function is used.

Solutions:

1. The HP-IL loop could be broken. Check to insure that the loop is not broken.
2. Are all of the devices on the loop turned on? The ThinkJet should have a red light in the upper left corner lit. With the HP82166A converter, the batteries could be low.
3. Is the printer out of paper? If so, this error will occur.
4. Is flag 33 set? If so, flip to Appendix E and read in the barcode for the CF33 program. Then execute CF33. Check that flag 33 is clear. Then scan the CLP CF33 barcode and try using the barcode functions again. Note that the CF33 program uses synthetic instructions. (Get a copy of "HP-41 Synthetic Programming Made Easy" if you want to learn what synthetic instructions are and what they are good for.)

```
01*LBL "CF33"
02 RCL d
03 STO M
04 "┼●●●●"          hex F5 7F 00 00 00 01
05 X<> M
06 X<> d
07 CF 01
08 X<> d
09 X<> M
10 "┼***"
11 X<> N
12 STO d
13 CLX
14 CLA
15 END
```

APPENDIX E

Barcode for Programs

XEQ "CF33"

CLP CF33

PROGRAM "CF33" 6 REGS/42 BYTES/4 ROWS

ROW 1 LINES (1-4)

ROW 2 LINES (4-8)

ROW 3 LINES (9-14)

ROW 4 LINES (15-15)

```
XEQ "ASNBC"
```


```
PROGRAM "ASNBC"  34 REGS/238 BYTES/19 ROWS
ROW 1  LINES (1-4)
```


```
ROW 2  LINES (4-8)
```


```
ROW 3  LINES (9-15)
```


```
ROW 4  LINES (16-24)
```


```
ROW 5  LINES (25-35)
```


```
ROW 6  LINES (36-43)
```


```
ROW 7  LINES (44-45)
```


```
ROW 8  LINES (46-51)
```


```
ROW 9  LINES (52-59)
```


```
ROW 10  LINES (59-67)
```


```
ROW 11  LINES (68-76)
```


```
ROW 12  LINES (76-77)
```


```
ROW 13  LINES (78-86)
```


```
ROW 14  LINES (86-90)
```


```
ROW 15  LINES (91-93)
```


```
ROW 16  LINES (93-98)
```


```
ROW 17  LINES (98-104)
```


```
ROW 18  LINES (105-112)
```


```
ROW 19  LINES (112-113)
```


```
XEQ "BCTEST"
```

```
PROGRAM "BCTEST"  31 REGS/214 BYTES/17 ROWS
ROW 1  LINES (1-3)
ROW 2  LINES (3-3)
ROW 3  LINES (4-6)
ROW 4  LINES (6-8)
ROW 5  LINES (8-10)
ROW 6  LINES (10-14)
ROW 7  LINES (14-15)
ROW 8  LINES (15-18)
ROW 9  LINES (18-20)
ROW 10  LINES (20-22)
ROW 11  LINES (23-26)
ROW 12  LINES (26-28)
ROW 13  LINES (28-31)
ROW 14  LINES (32-35)
ROW 15  LINES (35-43)
ROW 16  LINES (44-46)
ROW 17  LINES (46-48)
```

WARRANTY INFORMATION

Your SKWIDBC Module is warranted for 90 days from your date of purchase against hardware problems (physical and electrical defects). If you suspect that your module is defective, call SYNTHETIX (if you can) to discuss the problem. Weekends are best. Then pack the module carefully and send it, together with a copy of your receipt showing the date of purchase, to:

SYNTHETIX
P.O. Box 1080
Berkeley, CA 94701-1080 U.S.A.
Phone (415) 339-0601

Please include an addressed return envelope or an address label for your replacement module.

SKWIDBC has been thoroughly tested, and we do not expect that you will encounter any software problems. Nevertheless, software "bugs" are not covered by this warranty. If you find any serious bugs, we may, at our option, modify the SKWIDBC software and provide module updates at nominal cost. If you mail in your Registration Form (page 75), you will automatically be advised of the availability of these updates. We will also inform you of updates to this manual.

INDEX

AABC 5, 8, 41, 53
ABC 5, 8, 41, 53
ACTEXT 22, 26, 27, 29, 30, 31, 33, 53
ALPHABC 31
ASNBC 26, 31
AUTOIO 4

BCTEST 41

CHKSUM 24, 26, 30, 31, 33, 53
CLP 12, 29, 32
CLPBC 29, 32
COLBC 39, 42, 53
COMPEND 20, 53
Compiling 10, 11, 13, 21
COMPILE 21, 53
COMPOFF 16, 53
  Re-enabling compiling 16
COPYBC 30, 32

Decompiling 20
DELBC 29, 32
DIREXBC 8, 41, 53

Error Messages:
  ALPHA > 14 6, 49
  ALPHA > 15 25, 49
  ALPHA > 16 25, 49
  ALPHA DATA 23
  ALPHA EMPTY 25, 49
  DATA ERROR 7, 23, 24, 34, 35, 51
  DATA ERROR X 10, 50
  DATA ERROR Y 13, 14, 50
  ILL. CHR. 50
  ILLEGAL INST. 8, 9, 49

LINE TOO LG. 4, 49, 63
NO IL PR. ROM 3, 49
NO INTERFACE 4, 49
NO LBL ## 11, 51
NO LBL Q 8, 49
NO THINKJET 4, 49
NONEXISTENT 10, 37, 50
PRIVATE 10, 50
ROM 10, 11, 50
TRANSMITT ERR 4, 49, 63
X GT. Y 39, 51

FILL 37
FRMFEED 5, 11, 13, 34, 42, 53

HP-IL module 4

Interfaces:
HP-IL to RS-232 4, 55
HP-IL to Parallel 4, 58, 59
HP-IL LINK card 60,

Labels 39, 40
LaserJet 1, 2, 3, 4, 5, 7, 13
LaserJet+ 1, 2, 3, 4, 5, 12
LBL "Q" 8, 9, 17, 18, 20
LINK card software:
HP-IL LINK 61
Southern Software LINK 61
Standard 61

MAKEBC 18, 22, 23, 24, 25, 30, 31, 33, 43, 53
MANIO 4

Non-programmable functions 8, 17, 18, 20, 53

PACKING 12, 51
PAPERKB 17, 41, 53

PPROGBC 10, 13, 15, 53
PPROGRW 13, 15, 53
PRINT RW. ## 11, 13
PROGBC 10, 13, 15, 42, 53
PROGRW 13, 15, 53

RAD 16, 17
REGBC 36, 53
REGBCX 36, 41, 53
REGTOBC 35

SIZE 11, 19, 26, 31, 37
SIZEBC 28, 32
SKWIDBC
  CATalog 2 2
  Executing a function 2
  Installing 2
SKWIDBC+ 1, 5, 12

ThinkJet 1, 2, 3, 4, 5, 6, 11, 13

WAND:
  Bug in software 7
  Version 1E 3, 5, 33
  Version 1F 3, 33
  Function WNDDTX 37

XBC 34, 35, 42, 53
XROM 17, 53
XTOALFT 23, 26, 28, 30, 31, 42, 53
XTOART 24, 26, 29, 30, 33, 53

ALSO BY SKWID INK

If you enjoy your SKWIDBC Barcode Generation Module, you may be interested in another module created by SKWID INK.

The **EXTENDED IL ROM** helps you:

1) get the best use of wide carriage printers including the ThinkJet and LaserJet, and
2) manage files on your HP-IL disk drive or other mass storage device.

Since you have a SKWIDBC Barcode Generation Module already, you probably have access to a ThinkJet or LaserJet printer. The Extended IL ROM will allow you to more fully utilize the capabilities of this printer with the HP-41.

For example, the Extended IL ROM contains an easy-to-use Multiple Column Print Program function that can print a program listing in several columns on a single sheet of paper. All formatting is automatically done. You define the width of each column, and the Extended IL ROM does the rest. Even synthetic instructions are handled correctly.

Also included is a set of functions which allow you to create HP-IL "macros". You can save an alpha string in an I/O buffer and then send out the entire string by placing its "name" (a single character) in alpha.

The mass storage section allows you to access mass storage devices, including RAM disks, of up to 1 megabyte in size. All of the file creation functions in the HP-41's HP-IL Module have been rewritten to remove all known bugs.

Three new types of mass storage files have been added. Using the Extended IL ROM, you may now save all of the HP-41's memory, both main and extended, to a tape or disc. (This is advantageous for those of you who tend to get MEMORY LOST with a full calculator). Extended memory may also be saved by itself, and I/O buffers may be saved to tape or disc.

The directory function has been updated to include these three new file types. It also places the volume label as the first line of a printed listing (for easy identification) and lists the number of unused directory entries and unused records at the end of the listing.

Many functions which are totally new have also been included. NAME MEDium allows you to place a volume label on a disc or tape. Another function tells you how much room is left on the medium. The directory size may be obtained and the number of directory entries currently used.

If you are using a disc drive, tape drive, RAM disk, or 80-column printer with your HP-41, you need the Extended IL ROM.

LaserJet Barcode Service

If you are using SKWIDBC with a ThinkJet printer, you may occasionally want to have some of your programs or data barcoded on a LaserJet. See Appendix E of this manual for an example of what LaserJet barcode looks like. SKWID INK offers a LaserJet barcode service at reasonable prices. Send your programs and/or data to:

SKWID INK
P.O. Box 3103
Tustin, CA 92681 USA

For programs the cost is $5.00 per page or partial page. Unless otherwise specified, there will be 260 program bytes per page (20 rows at 13 bytes per row). Each program starts on a new page. For barcode types other than program, write to SKWID INK for a price quote.

REGISTRATION FOR UPDATES

If you would like to be informed of important updates and new products, please fill out this form and mail it to us.

Name _____

Company _____

Address _____

_____

_____

Phone _____

Which Module did you purchase? _____

Are you interested in having your SKWIDBC upgraded to SKWIDBC+? _____

Are you interested in a combined 8K module containing SKWIDBC (or SKWIDBC+) together with the EXTENDED IL ROM? _____

Are you interested in having your SKWIDBC or SKWIDBC+ combined with another 4K module you own into a single 8K module? _____

If so, which other module? _____

Is there any other information you would like to receive? _____

_____

Please mail your completed form to:

SYNTHETIX

P.O. Box 1080

Berkeley, CA 94701-1080 U.S.A.

Phone (415) 339-0601

ORDER BLANK

	Price per copy	Qty	Amount
For HP-71'S			
HP-71 Basic Made Easy, by Joseph Horn	$18.95	___	_____
For HP-71'S & HP-41'S			
Control the World with HP-IL, by Gary Friedman	$24.95	___	_____
For HP-41'S			
HP-41 Advanced Programming Tips, by A. McCornack & K. Jarett	$20.95	___	_____
HP-41 M-Code for Beginners, by Ken Emery	$24.95	___	_____
Inside the HP-41, by Jean-Daniel Dodin	$12.95	___	_____
Extend Your HP-41, by W. Mier-Jędrzejowicz	$29.95	___	_____
HP-41 Extended Functions Made Easy, by Keith Jarett	$16.95	___	_____
HP-41 Synthetic Programming Made Easy, by Keith Jarett (Includes one Quick Reference Card)	$16.95	___	_____
Quick Reference Card for Synthetic Programming (QRC)	$2.00	___	_____
Synthetic Quick Reference Guide (SQRG)	$5.95	___	_____
For HP-10C, 11C, 15C, 16C, AND HP-41			
ENTER (Reverse Polish Notation Made Easy), by J.Dodin	$4.95	___	_____
Humor			
It's Amazing How These Things Can Simplify Your Life: The Harold Guide to Computer Literacy	$4.95	___	_____
ROM Modules			
SKWIDBC Barcode Generation Module by Skwid Ink	$199.95	___	_____
SKWIDBC+ Barcode Generation Module by Skwid Ink	$249.95	___	_____
SKWIDBC upgrade to SKWIDBC+ (must return SKWIDBC module)	$50.00	___	_____
Extended IL ROM option for SKWIDBC, SKWIDBC+, or upgrade	$89.95	___	_____
AECROM by Redshift Software	$99.00	___	_____
Sales tax (California orders only, 6% or 7%)			_____

Shipping (circle one line below)	1st book	Add'l books
within USA, book rate (4th class)	$1.50	$0.50
USA 48 states, United Parcel Service	$2.50	$1.00
USA, Canada, air mail	$3.00	$1.50
elsewhere, book rate (6 to 8 week wait)	$2.00	$1.00

elsewhere, air mail $12.05 for **Extend Your HP-41**, $6.05 for others

Free shipping for **ENTER** and **It's Amazing...** with purchase of any other book

Free shipping for ROM's, QRC's, and SQRG's

Shipping total $_____

TOTAL DUE $_____

Checks must be in U.S. funds, and payable through a U.S. bank.

Name _____

Address _____

_____

City _____ State _____ Zipcode _____

Country _____

Mail to:

SYNTHETIX
P.O. Box 1080
Berkeley
CA, 94701-1080
USA

Phone: (415) 339-0601 7 days

ORDER BLANK

	Price per copy	Qty	Amount
For HP-71'S			
HP-71 Basic Made Easy, by Joseph Horn	$18.95	___	_____
For HP-71'S & HP-41'S			
Control the World with HP-IL, by Gary Friedman	$24.95	___	_____
For HP-41'S			
HP-41 Advanced Programming Tips, by A. McCornack & K. Jarett	$20.95	___	_____
HP-41 M-Code for Beginners, by Ken Emery	$24.95	___	_____
Inside the HP-41, by Jean-Daniel Dodin	$12.95	___	_____
Extend Your HP-41, by W. Mier-Jędrzejowicz	$29.95	___	_____
HP-41 Extended Functions Made Easy, by Keith Jarett	$16.95	___	_____
HP-41 Synthetic Programming Made Easy, by Keith Jarett (Includes one Quick Reference Card)	$16.95	___	_____
Quick Reference Card for Synthetic Programming (QRC)	$2.00	___	_____
Synthetic Quick Reference Guide (SQRG)	$5.95	___	_____
For HP-10C, 11C, 15C, 16C, AND HP-41			
ENTER (Reverse Polish Notation Made Easy), by J.Dodin	$4.95	___	_____
Humor			
It's Amazing How These Things Can Simplify Your Life: The Harold Guide to Computer Literacy	$4.95	___	_____
ROM Modules			
SKWIDBC Barcode Generation Module by Skwid Ink	$199.95	___	_____
SKWIDBC+ Barcode Generation Module by Skwid Ink	$249.95	___	_____
SKWIDBC upgrade to SKWIDBC+ (must return SKWIDBC module)	$50.00	___	_____
Extended IL ROM option for SKWIDBC, SKWIDBC+, or upgrade	$89.95	___	_____
AECROM by Redshift Software	$99.00	___	_____
Sales tax (California orders only, 6% or 7%)			_____

Shipping (circle one line below)	1st book	Add'l books
within USA, book rate (4th class)	$1.50	$0.50
USA 48 states, United Parcel Service	$2.50	$1.00
USA, Canada, air mail	$3.00	$1.50
elsewhere, book rate (6 to 8 week wait)	$2.00	$1.00
elsewhere, air mail	$12.05 for **Extend Your HP-41**, $6.05 for others	

Free shipping for **ENTER** and **It's Amazing...** with purchase of any other book
Free shipping for ROM's, QRC's, and SQRG's

Shipping total $_____

TOTAL DUE $_____

Checks must be in U.S. funds, and payable through a U.S. bank.

Name _____
Address _____
_____
City _____ State _____ Zipcode _____
Country _____

Mail to:

SYNTHETIX
P.O. Box 1080
Berkeley
CA, 94701-1080
USA

Phone: (415) 339-0601 7 days