HUMBUG ®
68000 Version

by Peter A. Stark

All Star-K computer programs are licensed on an "as is" basis without warranty.

Star-K Software Systems Corporation shall have no liability or responsibility to customer or any other person or entity with respect to any liability, loss or damage caused or alleged to be caused directly or indirectly by computer equipment or programs sold by Star-K, including but not limited to any interruption of service, loss of business or anticipatory profits or consequential damages resulting from the use or operation of such computer or computer programs.

Good data processing procedure dictates that the user test the program, run and test sample sets of data, and run the system in parallel with the system previously in use for a period of time adequate to insure that results of operation of the computer or program are satisfactory.

SOFTWARE LICENSE

A. Star-K Software Systems Corp. grants to customer a non-exclusive, paid up license to use on customer's computer the Star-K computer software received. Title to the media on which the software is recorded (cassette and/or disk) or stored (ROM) is transferred to customer, but not title to the software.

B. In consideration of this license, customer shall not reproduce copies of Star-K software except to reproduce the number of copies required for use on customer's computer and shall include the copyright notice on all copies of software reproduced in whole or in part.

C. The provisions of this software license (paragraphs A through C) shall also be applicable to third parties purchasing such software from customer.

NOTES

SK*DOS and HUMBUG are registered trademarks of Star-K Software Systems Corp. Whenever used in this manual, the term FLEX is a trademark of Technical Systems Consultants Inc.

6809 SK*DOS was formerly known as STAR-DOS.

We provide support for SK*DOS users via the Star-K Software BBS at 914-241-3307. This computerized system operates 24 hours a day at 300 and 1200 baud, 8 bits, no parity, 1 stop bit. Feel free to call the BBS at any time - it is a popular medium for interchanging ideas, opinions, and programs among the many users of Star-K software and hardware.

This is revision 1.06 of the manual, last revised on May 25, 1991.

Please save the HUMBUG serial number, printed below; it will enable us to provide better service to you should you need upgrades or assistance. The number is

HUM -

CONTENTS

INTRODUCTION

This manual provides information primarily for the PT68K-2 and -4 versions of HUMBUG/68K, although it may also be used for other 68K versions. In that case, your HUMBUG may not be exactly like the version described herein; it may have more - or fewer - functions or commands. You may, however, use the HE (help) command to determine which commands are present in your version.

HUMBUG is a necessary part of your computer, because it contains the very first instructions performed when the computer is first turned on. It performs the basic initialization of the system which is required to allow it to work, and becomes the 'monitor' which oversees the direct operation of the hardware - at least, until such time as you load SK*DOS.

HUMBUG is supplied on two EPROMs which plug into the two 28-pin sockets on the system board, labelled U20 and U27. On the PT68K-2, the EPROM labelled 'lower' or '1' plugs into U27, while the EPROM labelled 'upper' or '0' plugs into U20. On the PT68K-2, the 'lower' EPROM plugs into U26, and the 'upper' EPROM plugs into U27. When inserting the EPROMS, make sure not to bend any pins, and orient the notches (which identify pin 1) as shown on the printed circuit board.

Before turning on the computer, make sure to position the computer jumpers for the appropriate EPROM type – 27128, 27256, or 27512. The computer will not work if you fail to do this.

STARTING THE SYSTEM

In most systems, HUMBUG is operated through a serial terminal. On the PT68K-2 and -4, however, you may use either a serial terminal (plugged into J22), or a PC-style video card (plugged into any of the six PC-style connectors) and PC-style keyboard (plugged into J9.) The video card can be either a monochrome (or monochrome/graphics) card or Color Graphics (CGA) card; support for EGA and VGA cards will be available soon.

When you start the system, you may see nothing on the display until you press the RETURN key (which may be labelled RETURN, ENTER, CR, or just with a strange arrow which looks like <-'; we will refer to it as "RETURN" in the rest of this manual.) HUMBUG is waiting to see which keyboard you intend to use (either a serial terminal or a PC-compatible keyboard may be used), and (if a serial keyboard) what baud rate your terminal is set to, and will adapt itself to it. If you are using a serial terminal, then you may have to press RETURN two or three times before you get a display.

Once HUMBUG identifies your I/O equipment, it will display the message

```
HUMBUG (R) Ver. x.x Copyright (C) 1986-
1990 by Peter A. Stark
*_
```

The asterisk (*) is the prompt telling you that HUMBUG is now waiting for your command, and the underline (_) is the cursor (which may appear different on different terminals.) You may type the command HE for a listing of the allowed commands, which are described in the next section.

HUMBUG COMMANDS

HUMBUG responds to two-letter commands from the keyboard. For example, the HE command prints a 'help' message which gives a short listing of available commands.

Many of the HUMBUG commands, such as memory dump commands, require a starting and ending address for proper operation. These commands prompt for this pair of addresses with a FROM ... TO ... You may enter addresses in free form, and need not enter initial zeroes. For example, address $00000123 could be entered as 00000123, or 0123, or just 123. (Each number entered into HUMBUG **must** be followed by a space to indicate that it is finished.) The pair of addresses entered by the user is called the "FT" pair, and is stored for possible reuse later.

Any valid hex address is acceptable as a response to the FROM ... TO ... prompt. If a RETURN is entered instead of the FROM address, then the current command will use the last FT pair previously entered. Any other character (except a space) will cancel the execution of the command and return to HUMBUG command entry mode.

HUMBUG obeys the following two-character commands:

AD - Formatted ASCII dump. The specified area of memory is dumped to the output device, sixteen bytes to a line. Each line is identified with its starting address. ASCII codes of 7E, 7F, and 00-1F are printed as a period, and the most significant bit (parity bit) is ignored.

AI - ASCII Input. The AI command allows the direct input of ASCII data from the keyboard into any area of memory. All text following the AI is inserted into the memory area defined by the FT pair. If the memory area set aside is too small to hold all the text entered, or if the text is not properly stored (due to nonexistent or defective memory), either you will get a BUS ERROR message, or your screen will start outputting the word ERROR immediately after the last possible character has been stored. The only way to get out of the AI mode is by the control-S/RETURN combination, or by pushing RESET. When this is done, the FT pair will be changed to reflect the amount of memory actually filled by the AI, so that a following AO or HD command would output exactly the same data as entered by AI.

AO - ASCII Output. Following this command, the contents of the memory area defined by the FT pair is output to the screen. This is normally used to output ASCII text. The AI-AO combination is primarily intended for testing.

BA - Basic. If available, this command takes you into the Basic interpreter, which is described separately below.

BP - Print Breakpoints. HUMBUG allows up to four breakpoints to be set at the same time. The BP command prints out the addresses of the current breakpoints, and the operation codes of the instructions at those breakpoints, so that the user does not forget their locations.

BR - Breakpoint set/reset. The four possible breakpoints are numbered 1 through 4, and can be individually set or reset. When the system is first turned on, all breakpoints are erased; subsequent RESETs do not erase the current breakpoints. The typical BR command has the following form:

```
BR NUMBER: n ADDRESS: addr
```

where the computer's responses are underlined. *n* is the number of the breakpoint you wish to set or reset; *addr* is the new address of that breakpoint. Entering a new address, or hitting RETURN or any invalid entry for *addr*, will cancel the old breakpoint number n.

CO - Continue. After a breakpoint is encountered in a program, or after a single-step execution, the program being tested may be continued with the CO command. After a breakpoint, the breakpoint should be removed with the BR command before hitting CO; otherwise the break will be executed again and the program will not go on.

CS - Checksum. This command prints a 16-bit checksum of the memory area defined by the FT pair. This is primarily intended to check whether a program or data has been properly loaded, or whether it has been changed.

FD - Boot SK*DOS from a floppy disk. You must use the IS command to do an initial floppy disk setup before you can use the FD command for the first time. If you have trouble booting your floppy disk, recheck the IS command to make sure your system is properly set up.

FI - Find. FI will print out all addresses in the area of memory defined by the FT pair which contain a specified one- through five-byte constant. The typical command sequence is

```
FI HOW MANY BYTES? n FIND WHAT? d...d
FROM addr TO addr
```

where computer responses are underlined. *n* is the number of bytes to be found, *d...d* are 2 through 10 hex digits representing the 1 through 5 bytes to be found, and *addr* are the two FT addresses specifying the address range to be searched.

FM - Fill memory. This command allows a specified area of memory, defined by the FT pair, to be filled with a specified byte.

HA - Hex and ASCII dump. Combination of the HD and AD commands, which prints the contents of memory in both hex and ASCII.

HD - Hex Dump. Prints a hexadecimal dump of the area of memory defined by the FT pair. Sixteen bytes are printed per line, with each line preceded by the address.

HE - Help. Prints a listing of all HUMBUG commands.

IS - Initial Setup (introduced on HUMBUG version 2.0). On the PT68K-4, HUMBUG supports two different floppy disk controllers - the WD1772 (same as the PT68K-2), and the WD37C65B. Before you can boot a floppy disk with the FD command (on either the PT68K-2 or -4 system), you must therefore tell HUMBUG (a) which controller(s) you have, (b) which one you wish to boot from, and (c) what kind of disk drives you are using. If your system has a battery-backed up RAM, this information will be permanently stored. Thus you need only enter it the very first time you boot the system, or if you change the controller or disk drives at a later date. BUT NOTE: If you change the setup, SK*DOS will not use the new values until you reboot. ALSO: If you have a PT68K-2 computer, but attempt to run a PT68K-4 version of SK*DOS and have trouble using your floppy disk, make sure that location FL0C (at $FF0BE9) does

not have a $37 which is telling SK*DOS to use a 37C65 controller which you do not have.

JS - Jump to System program. This command jumps to the address specified after the JS, with the CPU in system (supervisor) state.

JU - Jump to User program. This command is similar to JS, but enters the program with the CPU in user state.

LO - Load S1 - S9 Motorola binary format from main keyboard.

MC - Memory compare. This command compares two specified memory areas byte-by-byte, and prints out memory contents for each byte which is different in the two areas. Prompts ask for the FT pair for the first area, and for the starting area of the second.

ME - Memory examine and change. This command allows you to examine the contents of memory on a byte-by-byte basis, and enter new data if desired. When you type in ME, followed by the address to be examined, HUMBUG will display the current contents of that address and wait. You may now type in one of the following:

a space to go to the next byte
an up arrow to go back to the previous byte
anything else to quit

MO - Move memory. This command allows the contents of the memory area specified by the FT pair to be moved to another memory area. Memory data can be moved to higher or lower addresses, and the new area can overlap the original area. Moving is done in the correct way so that no data is lost even on overlaps.

MS - Memory Store. This command is similar to ME, but stores data without first reading it out, and without verifying that it was properly stored. It is used primarily for storing data into I/O registers.

MT - Memory Test. Does a simple memory test on the memory area defined by the FT pair. If memory is OK, it prints a plus sign and returns to HUMBUG. If memory is bad, it prints the address of the bad location, a hex number representing the bad bit, and the actual contents of that location at the time it failed the test. (This is a non-destructive test of memory since the previous contents of each location are restored. If, however, a memory test is done of I/O locations, it is possible that false I/O operation may occur, or that I/O devices may not be properly initialized.)

RC - Register Change. Allows you to modify the contents of the CPU registers displayed by the RE (Register Examine) command. When you type RC, HUMBUG will respond with REG:, which you should answer with the code for the register to be changed. HUMBUG then enters the ME mode at the location where that register is stored. You may then examine or change the register contents, as desired. Note, however, that register A7 cannot be directly changed. Instead, you must change either US (user stack) or SS (system stack).

RD - Return to SK*DOS. This command returns to SK*DOS. Caution - do not use the RD command unless SK*DOS has already been booted and run.

RE - Register Examine displays the current CPU registers. The Data and Address registers are displayed first, with the remaining registers below. An RE printout is automatically performed following a single-step or upon encountering any breakpoint. Note that the display for A7 depends on which state is currently reflected in the status register; changing the current state will also generally change the A7 display.

SS - single-step. Perform the next instruction of the program being tested. SS uses the register contents printed by the RE command; hence the SS command cannot be used to start single-stepping until after a prior breakpoint or single-step has been performed. When an SS is performed, HUMBUG prints out several lines: the first line prints out the address of the instruction to be performed, while the other lines print out the RE dump after the instruction has been performed.

ST - Start single-stepping. Since SS cannot be performed until after a breakpoint or previous single step, the special ST command is included to perform an initial single-step if the breakpoint is not used. ST prompts for the address of the first instruction to be single-stepped, and then executes it in exactly the same way as the SS instruction.

WA, WB, or WD - Winchester disk boot. These commands are used to boot a Winchester disk, if available. WA and WB are used for a WD1002A-WX1 controller (WA boots from the first hard drive, WB from the second if you have one), while WD would be used for a WD1002-HDO controller.

X1 and X2 - These are auxiliary commands, included in HUMBUG to allow the insertion of additional commands by users.

!! - Monitor reset command. HUMBUG does not normally erase breakpoints except at the first power up; other resets omit this step. The !! command does a complete reset, exactly the same as at power up. In general, this is a command which will not be commonly used, and hence has been assigned a non-standard command code.

BASIC

In most systems, Basic (or UBASIC) is a disk-resident command intended for use with SK*DOS. On the PT68K-2 and -4, Basic is also present in the HUMBUG ROM. In that case, typing HUMBUG's BA command brings you into Basic. There is not much to say about it, since anyone familiar with Basic will be able to use it quite easily.

Basic obeys the following commands:

DATA	DIM	END	FOR	GOSUB	GOTO
IF	INPUT	LET	LIST	MON	NEW
NEXT	ON	POKE	PRINT	READ	REM
RESTORE	RETURN	RUN	SOUND	STOP.	

and has the following functions:

ABS	CHR$	INT	PEEK	RND	SGN
TAB					

All of the above are fairly standard, but there are a few special commands:

MON	returns to HUMBUG
LET	is optional
PEEK(decimal address)	returns contents of address
POKE dec.address, byte	stores byte into address
SOUND length,pitch	sounds note through the speaker

Decimal addresses in the above can be 0 through 16777215, but note that trying to PEEK or POKE a nonexistent address may lead to a BUS ERROR message and return to HUMBUG.

The length and pitch in SOUND are limited to 1 through 255. The length of the tone is proportional to the length specifier; the frequency of the note is inversely proportional to the pitch specifier (i.e., doubling the pitch specifier makes the tone go down one octave.)

Although this Basic allows string constants (as in PRINT "HI"), it does not allow string variables (such as A$). Only one statement per line is allowed, the arithmetic is floating point BCD with nine significant digits, and no exponentiation is allowed. Variable names can be A through Z, and A0 through Z9, but array names can only be A() through Z(); one or two dimensions of 1 through 255 are allowed, but subscripts start with 1, not 0.

HUMBUG I/O CONTROL

Any time that HUMBUG is looking for commands, or any time that INEEE or OUTEEE are called, it checks the keyboard for a control-S break character arriving from the keyboard. When a control-S is detected, HUMBUG rings the bell (control-G) and halts all current I/O.

When I/O is halted, HUMBUG waits for one more character which is used for controlling monitor ports. This control character can be one of the following:

Carriage Return (CR) - this cancels the current program and forces a return to the monitor.

W - turns the wait (pause) mode on and off. When the wait mode is on, output will stop every 15 lines to allow it to be read on a CRT terminal.

R - turns output to the serial (RS-232C) control port on or off.

B - turns output to the monochrome (Black and white) video board on or off.

C - turns output to the color video board on or off. Even when you use a serial terminal, HUMBUG is simultaneously outputting to the video board if you have one. The B or C options allow this to be controlled. If the two video cards do not conflict, you may use one of each, though HUMBUG will default to output via the monochrome card.

P - turns output on or off for a serial printer connected to port B of the 1st DUART (U29), at J21.

F - toggles between fast or slow output to a CGA video board, if any. HUMBUG defaults to the fast display when it starts, but on some CGA boards this may produce an unacceptable amount of flicker or snow on the screen; in that case, toggle to the slow option to reduce the snow.

Any other character is ignored and output continues.

The control-S/RETURN combination allows many stuck programs to be aborted without reaching for the RESET button. But there are two additional key combinations which provide greater control when a PC-style keyboard is used:

Pressing and holding Control (or CTRL) and then SCROLL LOCK (which is also labelled BREAK on most PC-style keyboards) aborts the current program and returns to SK*DOS, if it is loaded into memory. Otherwise it returns to HUMBUG.

Pressing and holding Control (or CTRL), Alternate (or ALT) and then Delete (or DEL) aborts the current program and returns to HUMBUG.

In order to prevent aborting a program in the middle of a disk operation, with the resultant possibility of corrupting a disk or its directory, the control-break and control-alt-delete commands are enabled only while executing a user program, and are disabled whenever SK*DOS is being called for a disk operation. You may therefore occasionally have to repeat the key sequence several times before the program actually halts.

MEMORY USAGE

The precise memory addresses used by HUMBUG depend on the version; the following discussion covers the version used in the PT68K-2 and -4 computer. If your HUMBUG was supplied as source code on a disk, then you may assign your own memory addresses with appropriate ORG or EQU statements; if supplied in EPROM, then the addresses are fixed; in that case, a separate list of addresses is supplied at the back of this manual.

HUMBUG uses the following memory locations:

0000 - 00BF	Exception vectors
00C0 - 0103	Entry point JMP vectors
0104 - 0107	Hard reset flag
0800 - 09FF	Used while booting SK*DOS
F80000 - F87FFF	HUMBUG EPROMs
FF0000 - FF0BE7	Static RAM used by Basic only
FF0BE8 - FF0FEF	Static RAM used by HUMBUG, its stack, etc.

Hence HUMBUG does not use any of the memory also used by SK*DOS. Moreover, the RAM used by Basic is unused at other times, so you may use it for writing short programs (except that the 512 bytes from $FF09E8 through $FF0BE7 is used when booting SK*DOS from a hard disk). Note, however, that SK*DOS uses parts of HUMBUG for console I/O, and so the upper part of the static RAM cannot be used when SK*DOS is running.

HUMBUG ENTRY POINTS

Most versions of HUMBUG contain a standardized set of entry points which may be used by user programs. In the PT68K-2 and -4 computer, HUMBUG uses the following entry points:

00C0	F800C0	COLDST	Cold start
00C6	F800C6	WARMST	Warm start
00CC	F800CC	INEEE	Input 7-bit char to D5, echo (current device)
00D2	F800D2	INCH7	Input 7-bit char to D5, no echo
00D8	F800D8	INCH8	Input 8-bit char to D5, no echo
00DE	F800DE	INCHEK	Check input device status, return NZ if has char
00E4	F800E4	OUTCHX	Output char from D5 to current device, no control-S check
00EA	F800EA	OUTEEE	Output char to current device, control-S check
00F0	F800F0	OUTCHM	Output char to serial port (DUART 1A)
00F6	F800F6	OUTCHP	Output char to serial printer (DUART 1B)
00FC	F800FC	OUTCHB	Output char to monochrome video board
0102	F80102	PSTRNG	print string pointed to by A4
0108	F80108	OUT4HS	Output 4 hex digits from D4
010E	F8010E	OUT8HS	Output 8 hex digits from D4
0114	F80114	OUTCHC	Output char to color video board
011A	F8011A	INCH16	Input 16-bit extended char to D5, no echo
0120	F80120	CHEK16	Check extended char status, return NZ if has char

The above routines have two addresses, since the F80xxx JMP vectors are copied into lower RAM at 0xxx. SK*DOS uses several of these routines, but uses the vectors at 0xxx so that user-written programs could be substituted.

USEFUL HUMBUG VARIABLES

The following are some useful variables in HUMBUG:

ADDR	BYTES	NAME	DEFAULT	EXPLANATION
FF0C00	4	PORADM	00FE0000	Address of main serial port
FF0C04	4	PORADP	00FE0020	Address of 2nd (printer) serial port
FF0C0C	1	PORECH	FF	Port echo flag: 0=no, else yes
FF0C0D	1	HARDWE		Available hardware descriptor
FF0C7A	1	STATSI		Input port status (see below)
FF0C7B	1	STATSO		Output port status (see below)
FF0C7C	1	CATTR	1E	Color Video board attribute
FF0C7D	1	MATTR	07	Monochrome Video board attribute
FF0C7E	1	CCURV		Color cursor vertical position (0-$18)
FF0C7F	1	CCURH		Color cursor horiz position (0-$4F)
FF0C80	1	MCURV		Mono cursor vertical position (0-$18)
FF0C81	1	MCURH		Mono cursor horiz position (0-$4F)
FF0C82	1	MCURSZ	$B	Mono cursor size
FF0C83	1	CCURSZ	6	Color cursor size
FF0C84	1	PAUCTR		Line counter for pause
FF0C9C	1	KBS		Keyboard status byte (see below)
FF0C9D	1	KBA		Keyboard active byte (see below)
FF0C9E	1	INQFLG		PC keyboard ready flag (0=not ready)
FF0C9F	1	INQCHR		Last char rcvd from PC keyboard
FF0CA0	4	T15VEC		Trap 15 vector
FF0CA4	4	BERVEC		Bus error vector
FF0CA8	2	EXTCHR		Extended 16-bit char

			from keyboard
FF0CAA	1	EXTFLG	Extended character flag (0=not ready)
FF0BE9	1	FL0C	Controller type for drive F0
FF0BEB	1	FL0D	Drive type for drive F0
FF0BED	1	FL1C	Controller type for F1
FF0BEF	1	FL1D	Drive type for drive F1
FF0BF1	1	FL2C	Controller type for F2
FF0BF3	1	FL2D	Drive type for drive F2
FF0BF5	1	FL3C	Controller type for F3
FF0BF7	1	FL3D	Drive type for drive F3
FF0BF9	1	FL4C	Controller type for F4
FF0BFB	1	FL4D	Drive type for drive F4
FF0BFD	1	FL5C	Controller type for F5
FF0BFF	1	FL5D	Drive type for drive F5

The HARDWE byte indicates what I/O equipment HUMBUG has found installed:

BIT	INSTALLED DEVICE
7	Monochrome video board
6	Color (CGA) video board
5	EGA video board
4	VGA video board
3	PC-style expansion slots
2	DUART 2
1	DUART 1
0	68230 Parallel Interface IC

The status bytes tell which I/O devices are currently active; a bit is 1 if active, 0 if not active. It is possible to have several output devices (but not input devices) active at one time. For example, when the input status byte is 01100000, both the serial terminal output and the monochrome video board are on. This is a common situation when HUMBUG detects a monochrome video board while you are using a serial terminal. STATSO, the output status byte, definitions are:

BIT	CURRENTLY ACTIVE OUTPUT DEVICE
7	Serial printer on DUART 1 port B
6	Serial terminal output on DUART 1 port A
5	Monochrome video board
4	Color (CGA) video board
3	EGA video board
2	VGA video board
1	reserved
0	Pause mode flag

STATSI, the input status byte, definitions are:

BIT	CURRENTLY ACTIVE INPUT DEVICE
7-2	reserved
1	PC-style keyboard in use
0	serial keyboard on DUART 1 port A

The KBA and KBS bytes are associated with the PC keyboard and its interrupt service routine. The KBA byte tells us which of two lock keys is currently depressed:

BIT	KEY CURRENTLY DEPRESSED
6	Caps lock key
5	Num lock key
	other bits are currently not used

The KBS byte tells us what the current keyboard mode is:

BIT	CURRENT KEYBOARD MODE
6	Caps lock mode is on
5	Num lock mode is on
2	Control key is depressed
0	Shift key is depressed
	other bits are currently not used

The two attribute bytes define the color and/or intensity of the characters used by the video boards, using the standard bit notation as described in the video board documentation. For example, 07 for the monochrome board defines a dim character without underlining or blinking; the $1E for the color board defines a high intensity yellow character on a blue background without blinking.

The twelve FL0C through FL5D bytes specify the controller and drive type for up to six floppy disk drives (a maximum of four on a 1772 controller, and two on a 37C65 controller.) FL0C through FL5C specify the controller type for the six drives normally called F0 through F5 by SK*DOS:

VALUE	CONTROLLER
00	None
17	1772
37	37C65
other	1772 (default with older versions of HUMBUG)

FL0D through FL5D specify the drive types for those six drives. These bytes are split into two 4-bit nibbles: The left four bits (representing the numbers 0 through 3 for the 1772, or 0 through 1 for the 37C65) give the physical drive number the drive is selected as, either through jumpers on the drive or through a twist in the drive cable. The right four bits specify the drive type as follows:

VALUE	DRIVE TYPE
0	None
1	360K (standard 40-track 300 rpm)
2	720K (standard 80-track 300 rpm)
3	1.1 meg (80-track 360 rpm "high-density")
4	1.4 meg (80-track 300 rpm 3-1/2" "high-density")
5	1 meg 8" (special interface cable required)

NOTE: If you have fewer drives than a controller allows, then you should number them starting with the lowest number. For example, if you have only two drives in a 1772 controller, then they must be numbered 0 and 1; neither drive can be drive

2 or 3. In particular, SK*DOS will only boot from the first drive on a controller, so there must be a drive F0.

You must use the IS command before booting from a floppy disk with the FD command, so HUMBUG knows which controller and drive to boot from.

EXTENDED 16-BIT CHARACTERS

If a PC-compatible keyboard is used, then HUMBUG (versions 1.0 and above) may return a 16-bit character code as well as an 8-bit code; this allows various combinations of control, function, alternate, and cursor keys to be used. This process works as follows:

For either a serial or PC-compatible keyboard, when INCH7 or INCH8 are called, only normal alphanumeric keys return a character in D5.B; if a function or special key is pressed, it is ignored, and HUMBUG continues to wait for a normal character. INCHEK is then the status function which may be checked to determine whether such a character is available.

INCH16, however, may be called when it is desired to sense special keys as well as normal keys (in which case CHEK16 is the corresponding status function call.) If a serial keyboard is in use, then INCH16 defaults to the normal INCH8 function. If a PC-compatible keyboard is in use, however, then INCH returns a code for every key on the keyboard, as well as unique codes for combinations which include control, alternate, or function keys. These codes are extended 16-bit codes, returned in D5.W.

For normal (i.e., alphanumeric characters and standard control) characters, the left eight bits of the code are zeroes, and the right eight bits are the standard ASCII codes for that key, with the exception that bit 7 is 1 if the ALT key is depressed, 0 otherwise. Non-standard keys (such as function or cursor keys) as well as combinations of such keys return a code whose left eight bits are non-zero.

The following table shows the codes generated by each key:

KBD KEY#	KEY	(1) NORM	(2) SHIFT	(3) CNTRL	(4) NUMLCK	
1	ESCAPE	1B/001B	1B/001B	1B/001B		*
2	1 !	31/0031	21/0021	--/0431		*
3	2 @	32/0032	40/0040	++/0100		*
4	3 #	33/0033	23/0023	--/0433		*
5	4 $	34/0034	24/0024	--/0434		*
6	5 %	35/0035	25/0025	--/0435		*
7	6 ^	36/0036	5E/005E	1E/001E		*
8	7 &	37/0037	26/0026	--/0437		*
9	8 *	38/0038	2A/002A	--/0438		*
10	9 (	39/0039	28/0028	--/0439		*
11	0)	30/0030	29/0029	--/0430		*
12	- _	2D/002D	5F/005F	1F/001F		*
13	= +	3D/003D	2B/002B	--/043D		*
14	BACKSPACE	08/0008	08/0008	7F/007F		*
15	TAB	09/0009	--/021F	--/031F		*
16	q Q	71/0071	51/0051	11/0011		*
17	w W	77/0077	57/0057	17/0017		*
18	e E	65/0065	45/0045	05/0005		*
19	r R	72/0072	52/0052	12/0012		*
20	t T	74/0074	54/0054	14/0014		*
21	y Y	79/0079	59/0059	19/0019		*
22	u U	75/0075	55/0055	15/0015		*
23	i I	69/0069	49/0049	09/0009		*
24	o O	6F/006F	4F/004F	0F/000F		*
25	p P	70/0070	50/0050	10/0010		*
26	[{	7B/007B	5B/005B	1B/001B		*
27	] }	7D/007D	5D/005D	1D/001D		*
28	RETURN	0D/000D	0D/000D	0D/000D		*
29	CONTROL	--/----	--/----	--/----	--/----	
30	a A	61/0061	41/0041	01/0001		*
31	s S	73/0073	53/0053	13/0013		*
32	d D	64/0064	44/0044	04/0004		*
33	f F	66/0066	46/0046	06/0006		*
34	g G	67/0067	47/0047	07/0007		*
35	h H	68/0068	48/0048	08/0008		*
36	j J	6A/006A	4A/004A	0A/000A		*
37	k K	6B/006B	4B/004B	0B/000B		*
38	l L	6C/006C	4C/004C	0C/000C		*
39	; :	3B/003B	3A/003A	--/043B		*
40	' "	27/0027	22/0022	--/0427		*
41	` ~	60/0060	7E/007E	--/0460		*
42	LEFT SHIFT	--/----	--/----	--/----	--/----	
43	\ |	5C/005C	7C/007C	1C/001C		*
44	z Z	7A/007A	5A/005A	1A/001A		*
45	x X	78/0078	58/0058	18/0018		*
46	c C	63/0063	43/0043	03/0003		*
47	v V	76/0076	56/0056	16/0016		*
48	b B	62/0062	42/0042	02/0002		*
49	n N	6E/006E	4E/004E	0E/000E		*
50	m M	6D/006D	4D/004D	0D/000D		*
51	, <	2C/002C	3C/003C	--/042C		*
52	. >	2E/002E	3E/003E	--/042E		*
53	/ ?	2F/002F	3F/003F	--/042F		*
54	RIGHT SHIFT	--/----	--/----	--/----	--/----	
55	PRT SCR *	2A/011D	2A/002A	--/031D	2A/002A	
56	ALT	--/----	--/----	--/----	--/----	
57	SPACE	20/0020	20/0020	20/0020		*
58	CAPS LOCK	--/----	--/----	--/----	--/----	
59	F1	--/0101	--/0201	--/0301		*
60	F2	--/0102	--/0202	--/0302		*
61	F3	--/0103	--/0203	--/0303		*
62	F4	--/0104	--/0204	--/0304		*
63	F5	--/0105	--/0205	--/0305		*
64	F6	--/0106	--/0206	--/0306		*
65	F7	--/0107	--/0207	--/0307		*
66	F8	--/0108	--/0208	--/0308		*
67	F9	--/0109	--/0209	--/0309		*
68	F10	--/010A	--/020A	--/030A		*
69	NUM LOCK	--/----	--/----	--/----	--/----	
70	SCRLL LOCK	--/011E	--/021E	--/031E		*
71	HOME	--/0120	37/0037	--/0320	37/0037	
72	UP ARROW	0B/0125	38/0038	--/0325	38/0038	
73	PG UP	--/0123	39/0039	--/0323	39/0039	
74	GREY MINUS	2D/002D	2D/002D	--/032D	2D/002D	
75	LEFT ARROW	08/0128	34/0034	--/0328	34/0034	
76	5	--/0127	35/0035	--/0327	35/0035	
77	RT ARROW	09/0126	36/0036	--/0326	36/0036	
78	GREY PLUS	2B/002B	2B/002B	--/033B	2B/002B	
79	END	--/0122	31/0031	--/0322	31/0031	
80	DN ARROW	0A/0127	32/0032	--/0327	32/0032	
81	PG DN	--/0124	33/0033	--/0324	33/0033	
82	INSERT	--/012A	30/0030	--/032A	30/0030	
83	DEL	--/012B	2E/002E	--/032B	2E/002E	

NOTES:

1. All codes in columns (1) through (4) are hex numbers.

2. The notation AA/BBCC means that the code AA is generated using the normal character input routine (INCH8 in HUMBUG; GETCH, INNOEC, or ICNTRL function 0002 in SK*DOS), and BBCC is generated using ICNTRL (in HUMBUG, or ICNTRL function 0005 in SK*DOS) #

3. If BB is 00, then CC is the standard ASCII code for that key, and is equal to AA. With some exceptions, BB codes of 01 stand for unshifted characters, 02 stand for shifted characters, 03 stand for control characters, and 04 stand for characters which do not fit any of the above groups.

4. -- or ---- means that no key code is generated for that key.

5. Items labelled * are not affected by the NUM LOCK key; their key codes are indicated in the other three columns at all times.

6. CAPS LOCK affects only the alpha keys A-Z. For these keys, it reverses the meanings of columns (1) and (2).

7. The ALTernate key adds $80 (or $0080) to all codes shown.

8. The precedence is (a) ALT affects all codes, (b) NUM LOCK codes are not affected by SHIFT or CONTROL, (c) CONTROL is not affected by SHIFT.

9. The ++ code for control-@ generates no code in HUMBUG version 1.0, and generates a null (00) in versions 1.1 and later. 10. The first column codes for PRT SCR, UP ARROW, LEFT ARROW, RIGHT ARROW, and DOWN ARROW were changed from -- to the values listed above in HUMBUG version 1.7.