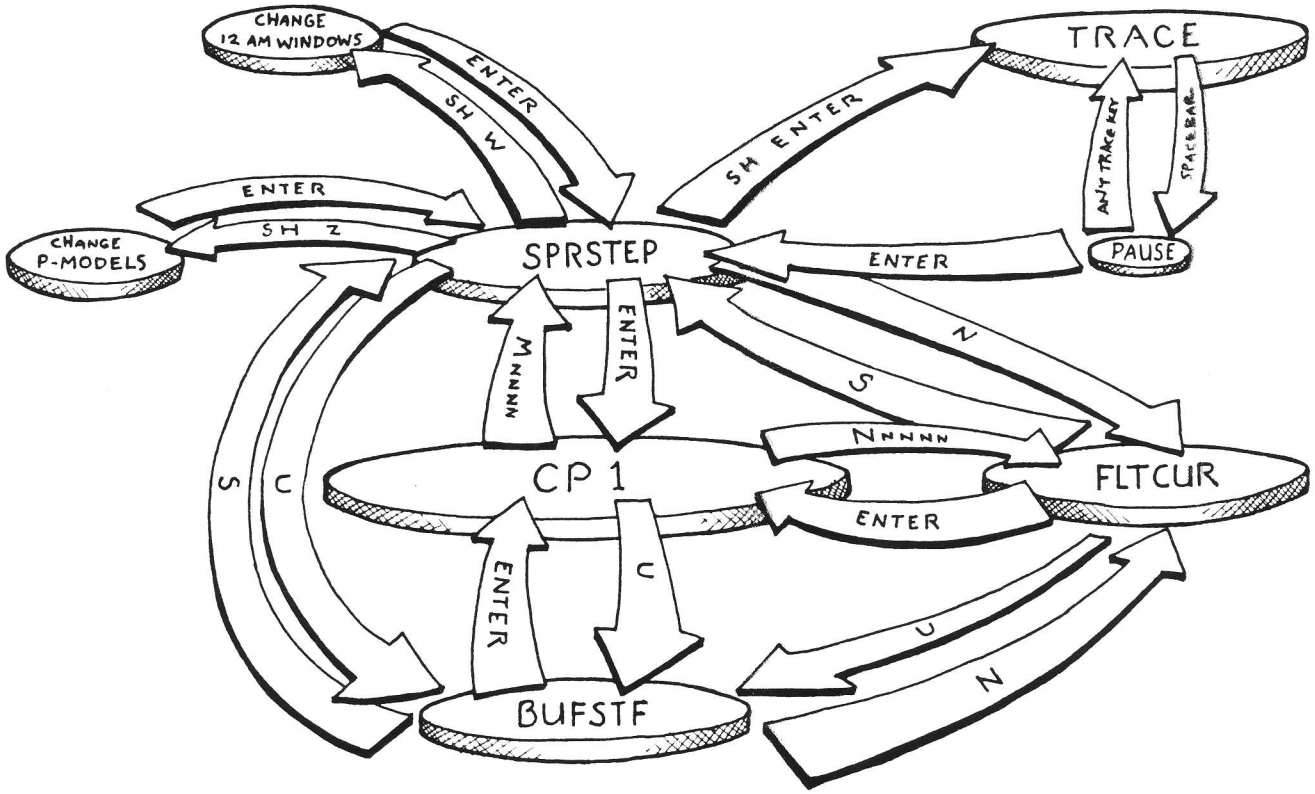
STRETCH SuperStep



ALGORIX

(415) 387-3131

Box 11721 San Francisco, CA 94101

STRETCH SuperStep
Z80 Simulator/byte editor for TRS-80 Model I/III

ESSENTIAL INFORMATION

1. Loading and making a backup.

   i) Put product diskette in drive 0 and press the reset button. Press the appropriate key to load STR32/CMD, STR48/CMD or DEMO/CIM. -MINOS- (c) 1982 SOUTHERN SOFTWARE.

   ii) Now put a SYSTEM diskette in drive 0 and press ~~(SH) BREAK~~ RESET to boot the DOS from STR32/CMD or STR48/CMD. DEMO/CIM returns automatically to DOS.

   iii) Use the DUMP command to make a copy on your SYSTEM diskette. Consult your DOS manual for the correct syntax. The programs are located as follows:

STR32/CMD	start=7700H, end=8FFFH,	tra=8BF6H
STR48/CMD	start=8700H, end=FFFFH,	tra=CBF6H
DEMO/CIM	start=7000H, end=70FFH,	tra=8BF6

, so make sure STR32 is loaded first.

For example, to save STR32 on Mod I, we type

```
DUMP STR32/CMD (START=X'7700',END=X'8FFF',TRA=X'8BF6')    (enter)
```

To save STR48 on Mod III, we type

```
DUMP STR48/CMD (START=0B700,END=0FFFF,TRA=0CBF6)
```

TABLE OF CONTENTS

Stretch SuperStep Operating Instructions

HISTORY: Stretch SuperStep is the current version of a design base that began in 1978 with TSTEP, a module that linked with Radio Shack's monitor TBUG and provided a single-stepping display of the Z80 registers, including a bit-expansion of the F (flag) register and the top of the stack. Details of the display format shared by the Software CPU (tm) series of CPU simulators made their first appearance in TSTEP, followed in 1979 by EMU 02, a 6502 simulator, and in 1980 by Super STEP and EMU 09, a 6809 simulator. Super STEP was a complete rewrite of TSTEP that added a disassembler and TRACE capability to the original single-stepper as well introducing a new feature to the display format, the Intelligent RAM Window. Later in 1980 the W/IT/T extension to Super STEP provided more RAM windows, extended the TRACE controls and optionally displayed instruction times. EMU 09 was never published, but the display format showed up later in 6809 native code with COCOBUG, a 6809 monitor for the TRS-80 Color Computer, first appearing in January 1981. Stretch SuperStep is the third major revision of Super STEP. Added is a full screen byte oriented editor and a special buffer with viewing controls, also certain keys are now autorepeating and there is printer control. Another program in the EMU series is currently under preparation.

DESIGN GOALS: The Software CPU series is meant to provide a set of detailed instruction-level CPU simulators running on the TRS-80 Model I/III. These simulators cover their target systems using a common display format and command structure. They allow the user to write, debug and "execute" machine language programs in the code of the target system without actually having the CPU in question. Developed code is entirely compatible with the real microprocessor. Stretch SuperStep is a case where the target CPU and the host CPU are the same.

Operationally, there are two simple goals: first, to show all active machine elements instruction-by-instruction in one display screen and second, to be able to examine and/or modify anything shown with as few keystrokes as possible. In the primary simulation mode (single-step/TRACE) this is carried out by using a split screen. The left side of the screen scrolls and displays the instruction stream in the form of a memory location, its contents byte and the disassembled op code. The right side of the screen displays a Z80 Programming Model with registers, flag expansion and stack, also the Intelligent RAM Window.

This side of the screen doesn't scroll. The P-Model is animated by the scrolling instruction stream, the registers, flags and stack all operate in response to the disassembled instructions. Further information is shown by the Intelligent RAM Window. The IW is activated by any instruction referencing memory, either directly or indirectly, it shows the environment around the referenced location.

Thus we have the instruction stream, the CPU architecture and referenced memory displayed at once; this comprises the programmer's imaginative overhead. Actually, everything is displayed twice at once; an optional format shows the P-Models at both entry and exit of the current instruction for before/after comparison of the effects on the CPU and memory. Moreover, any displayed element of RAM or CPU architecture can be changed by the user on the spot with a minimum of keystrokes, so it becomes quite easy to program experimentally. This is handy whether or not you believe computer science is an experimental science.

FEATURES OF THIS PROGRAM: Stretch SuperStep has features beyond the other programs in the Software CPU series. Since the simulators operate at the byte-by-byte instruction level, also included is a full screen byte oriented editor operable in hex or ASCII modes, with over 30 commands, a floating cursor and autorepeat. From this control point it's easy to view, write and edit machine language programs directly in RAM. The disassembler is available in this editing mode.

A user-defined buffer area (typically 12K-24K in size) with viewing controls is the third major functional segment in Stretch SuperStep. The buffer is stuffed in various combinable formats: disassembled listings in straight-line or program-flow order, hex/ASCII displays, etc. and a window with viewing controls is opened over the material. This lets you freeze pieces of code (12K will hold about 256 disassembled lines) for examination or reference. More specialized stuffs load the buffer with selected instructions in a given area of memory. You may isolate all instances of program flow affecting instructions (JP, CALL, etc.) or all RAM affecting instructions (LD nnnn, ADD (HL), etc.), or all comparisons (CP A, FFH, CP (HL) etc.) or any particular set of instructions you may be interested in. Why? This can help you when you are trying to make semantic sense out of some alien machine language program or during debugging. The idea is that you can take a skeleton view of a program at the instruction level.

CONTROL POINTS

Stretch SuperStep is organized into four functional segments, each with its own characteristic screen displays and key commands. The segments are:

1) Control Point 1 (CP1). This is the primary control point and can always be returned to from the other control points by keying "ENTER". There are about 30 key commands available under CP1, having to do with formatting the display, defining and loading the buffer, formatting the printer and other housekeeping details, doing tape I/O, breakpointing/jumping, and entering the three other segments etc. etc.

2) FLTCUR. This is the 2-D byte oriented editor; about 20 keys control the cursor and the display of material. A sub-segment displays memory in ASCII.

3) SPRSTP. Simultaneously displays memory and registers. Memory appears as a 1-D scrolling strip at the left of the screen; most of the FLTCUR editing commands are available for use on the scrolling area, which has an alternative mode that suppresses the scrolling and displays only two memory location (normally up to 16 locations are shown). Registers are shown in the non-scrolling area at the right screen; there are several formats showing a P-Model and a set of RAM windows, two P-Models (before/after) or nothing at all, for use with subject programs that need the screen. The Single-step and TRACE modes activate the scrolling strip and the P-Models. TRACE itself is a sub-mode with about 20 commands available.

4) BUFSTF. Works in conjunction with CP1. The user defines the buffer area (e.g. C000-EFFF) and directs loading of the buffer under CP1. The 10 BUFSTF keys are used to control a window over the buffer for viewing large amounts of code in a convenient way.

Screen pictures can be made under FLTCUR, SPRSTP and BUFSTF. Any of the modules can be reached from any other module; SPRSTP can be entered from FLTCUR or BUFSTF by keying "S", FLTCUR from SPRSTP or BUFSTF by keying "N", BUFSTF from SPRSTP or FLTCUR by keying "U". CP1 is reached from any module by keying "ENTER". The commands available under these control points are initiated by single keystrokes (sometimes parameters are required) in aid of minimizing number of keystrokes needed. The disadvantage to this scheme is that some key commands are rather arbitrary, meaning the user must refer to the summary card often at first. With a little use, however, the key commands become familiar and operating the program will be easy.

```
                        AF'= 0000  0000 =BC'  . 42E9 54 54
                        DE'= 0000  0000 =HL'  . 42EA 43 43
. STRETCH SUPER STEP                          . 42EB 48 48
(C)1982 ALLEN GELDER     AF= 0000  0000 =BC   . 42EC 20 20
                         DE= 0000  0000 =HL   . 42ED 43 43
                                              . 42EE 4D 4D
                         IX= 0000  0000 =IY   . 42EF 44 44
                         SP= 7680  FFFF =PC   . 42F0 96 96
.                            SZ-H-PNC    */] .  9126 00 00
                             00000000        .  9127 00 00
                            P NZ PO NC       .  9128 00 00
                          7680 FF  FFFE 00   .  9129 00 00
                          7681 FF  FFFF 00    . 3800 00 00
                          7682 FF  0000 F3    . 3801 00 00
                          7683 FF  0001 AF   .  3840 00 00
                          7684 FF  0002 C3    . 3880 00 00
```

Fig. 1. Control Point 1.

```
7000 .3E.41 32 A0 3C FF FF FF FF FF FF FF FF FF FF FF   .7000.
7010  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ......
7020  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
7030  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
7040  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
7050  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
7060  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
7070  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
7080  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
7090  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
70A0  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
70B0  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
70C0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
      -0--1--2--3--4--5--6--7--8--9--A--B--C--D--E--F
```

Fig. 2. A FLTCUR screen.

```
                        AF'= 0000  0000 =BC'    0000  0000
                        DE'= 0000  0000 =HL'    0000  0000
. STRETCH SUPER STEP
(C)1982 ALLEN GELDER     AF= 4100  0000 =BC     0000  0000
                         DE= 0000  0000 =HL     0000  0000

                         IX= 0000  0000 =IY     0000  0000
                         SP= 7680  7001 =PC     7680  7002
. M 7000 3E  LD A,41H        SZ-H-PNC    * ]    SZ-H-PNC
7001 41                      00000000           00000000
7002 32                     P NZ PO NC         P NZ PO NC
                          7680 FF  FFFE 00  7680 FF  FFFE 00
                          7681 FF  FFFF 00  7681 FF  FFFF 00
                          7682 FF  0000 F3  7682 FF  0000 F3
                          7683 FF  0001 AF  7683 FF  0001 AF
                          7684 FF  0002 C3  7684 FF  0002 C3
```

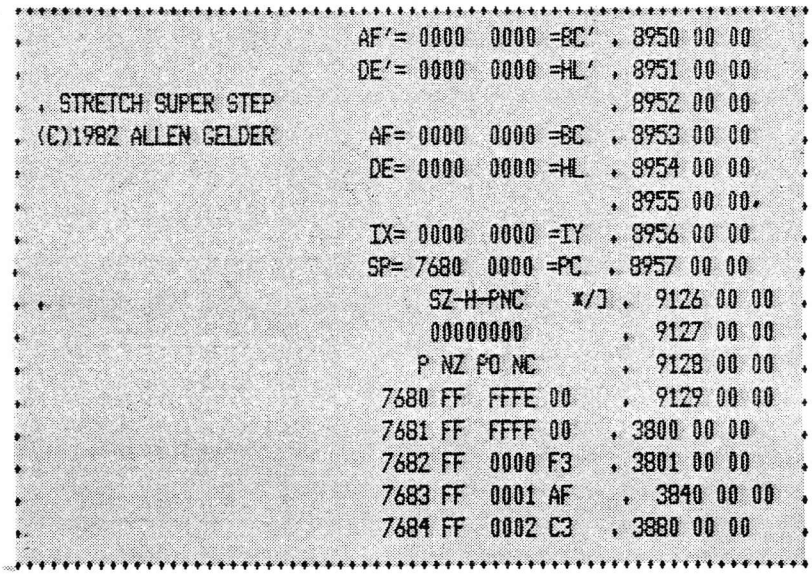
Fig. 3. A SPRSTP screen.

```
0083  36 3A      LD (HL),3AH    018B  12        LD (DE),A
0086  70         LD (HL),B      01AD  32 99 40  LD (4099),A
0088  36 2C      LD (HL),2CH    01B8  77        LD (HL),A
008B  22 A7 40   LD (40A7),HL   01BF  22 21 41  LD (4121),HL
0096  36 C3      LD (HL),C3H    01C4  32 AF 40  LD (40AF),A
0099  73         LD (HL),E      01D5  32 AB 40  LD (40AB),A
009B  72         LD (HL),D
00A1  36 C9      LD (HL),C9H
00AB  70         LD (HL),B
00CF  77         LD (HL),A
00D1  70         LD (HL),B
00E2  77         LD (HL),A
00E4  70         LD (HL),B
00F2  22 B1 40   LD (40B1),HL
00F6  22 A0 40   LD (40A0),HL
0182  12         LD (DE),A
```

Fig. 4. A BUFSTF screen.

CONTROL POINT 1 (CP1)

Control Point 1 is the nexus control point; the other control points can be reached from CP1, and CP1 can be reached from any of the other control points by pressing the "ENTER" key.



	KEY		KEY FUNCTION
1.	S		Clears the screen, brings up the copyright notice and screen display.
2.	M nnnn	(see SPRSTP)	Opens SPRSTP control point over location nnnn.
3.	N nnnn	(see FLTCUR)	Opens FLTCUR control point over location nnnn.
	Breakpoint/Jump commands		
4.	Bm nnnn		Set breakpoint m (m=1,2,3,4) at location nnnn. B5 sets 76 HALT.
5.	(SH) Bm		Clear breakpoint m. (SH) B5 sets 76 NO HALT.
6.	?		Display all breakpoints.
7.	J nnnn		Jump (transfer real-time program flow) to location nnnn.
	Display Formatting commands		
8.	X	(see SPRSTP)	Alternates left screen scrolling between full scrolling and reduced (2-line) scrolling. Line locations are found in A51C-1D and A51E-1F.
9.	Z	(see SPRSTP)	Alternates right screen format between the P-model/RAM Window format and a clear right screen.

CP1 (cont)

10. W (see SPRSTP) — Alternates extreme right screen between RAM Window and secondary P-Model display.

11. T (see SPRSTP) — Brings up Instruction Time display. Mod I is on 1.774 mhz basis, Mod III is on 2.028 mhz basis.

12. (SH) Z — Change P-Model register. Opens a cursor over AF register. The user may enter a hex byte (00-FF), move the cursor using ← , → or exit by keying "ENTER".

13. (SH) W — Change RAM Window positioning. Opens a cursor over the top RAM Window. The user may enter a hex location (0000-FFFF), move the cursor using ← , → , or exit by keying "ENTER"

14. X — Controls CALL execution (real-time/single-stepped). See SPRSTP.

15. / (see SPRSTP) — Deactivates simulator, which means that Single-step and TRACE modes operate in straight-line order. Otherwise simulator proceeds in program-flow order.

Buffer Control (see BUFSTF)

16. U — Opens BUFSTF control window (after defining the buffer and loading it as below).

17. (SH) U mmmm nnnn — Defines BUFSTF buffer from mmmm to nnnn, eg U C000 EFFF .

18. ; mmmm nnnn — Load buffer with hex memory from mmmm to nnnn (or until buffer is filled).

19. + mmmm nnnn — Load buffer with hex/ASCII memory from mmmm to nnnn (or full buffer).

20. - mmmm nnnn — Load buffer with disassembled listing from mmmm to nnnn (or full buffer). Note that the order of the listing (straight-line or program-flow order) depends on "/" key (see below).

21. ! mmmm nnnn — Load buffer with all occurances of memory-affecting instructions found between mmmm and nnnn. SL or PF order depending on "/".

22. " mmmm nnnn — Load buffer with all occurances of comparison instructions. SL or PF order depending on "/".

23. # mmmm nnnn — Load buffer with all occurances of program-flow affecting instructions. SL or PF order depending on "/".

24. $ mmmm nnnn
aa bb cc dd — Load buffer with all occurances of instructions with leading bytes aa, bb, cc or dd (00-FF). SL or PF order depending on "/".

CP1 (cont)

Tape I/O Mod I only

27. P mmmm nnnn aaaa XXXXXX — Write SYSTEM tape of material with start address mmmm, end address nnnn, entry point aaaa and file name XXXXXX.

28. L XXXXXX mmmm nnnn aaaa — Read SYSTEM tape into RAM. Parameters are read from tape.

29. (SH) P mmmm nnnn aaaa XXXXXX — Write fast tape. This is not a SYSTEM readable tape.

30. (SH) L XXXXXX mmmm nnnn aaaa — Read fast tape. Parameters are read from the tape.

Printer control

31. Z mmmm nnnn — Print hex memory from mmmm to nnnn in 20 X 8 format.

32. & mmmm nnnn — Print hex memory from mmmm to nnnn in 20 X 20 format (1K).

33. (— Print one page of buffer.

34. ' — Turn printer echo on/off. When on the "'" appears in the status panel and the SPRSTP commands Single-step and TRACE will be echoed by the printer.

Other commands

35. F — Find byte that's in the A register, start looking at HL (load A and HL using the (SH) Z command). Keep decremented count in BC This is exactly the CPIR format. The Intelligent RAM Window shows the envirnment around the found byte.

FLTCUR screens

```
          . 0002  C3 74 06     JP 0674          .
          ..................................
. 0000  F3 AF C3 74 06.C3.00 40 C3 00 40 E1 E9 C3 9F 06   .@@@@.
. 0010  C3 03 40 C5 06 01 18 2E C3 06 40 C5 06 02 18 26   ......
. 0020  C3 09 40 C5 06 04 18 1E C3 0C 40 11 15 40 18 E3
. 0030  C3 0F 40 11 1D 40 18 E3 C3 12 40 11 25 40 18 DB
. 0040  C3 D9 05 C9 00 00 C3 C2 03 CD 2B 00 B7 CD 1B F9
. 0050  0D 0D 1F 1F 01 01 5B 1B 0A 1A 0B 1B 09 19 20 20
. 0060  0B 78 B1 20 FB C9 31 00 06 3A EC 37 3C FE 02 D2
. 0070  00 00 C3 CC 06 11 80 40 21 F7 1B 01 27 00 ED B0
. 0080  21 E5 41 36 3A 23 70 23 36 2C 23 22 A7 40 11 2D
. 0090  01 06 1C 21 52 41 36 C3 23 73 23 72 23 10 F7 06
. 00A0  15 36 C9 23 23 23 10 F9 21 E8 42 70 31 F8 41 CD
. 00B0  BF 1B CD C9 01 21 05 01 CD A7 2B CD B3 1B 38 F5
. 00C0  D7 B7 20 12 21 4C 43 23 7C B5 28 1B 7E 47 2F 77
.       -0--1--2--3--4--5--6--7--8--9--A--B--C--D--E--F
```

Fig. 1. Disassembly, using SPACEBAR.

```
          .IMM  3CA0  20 46 46 20 46 46 20 46  .
          ..................................
. 7000  3E 41 32.A0.3C FF FF FF FF FF FF FF FF FF FF FF   .7003.
. 7010  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ......
. 7020  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
. 7030  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
. 7040  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
. 7050  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
. 7060  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
. 7070  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
. 7080  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
. 7090  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
. 70A0  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
. 70B0  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
. 70C0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
.       -0--1--2--3--4--5--6--7--8--9--A--B--C--D--E--F
```

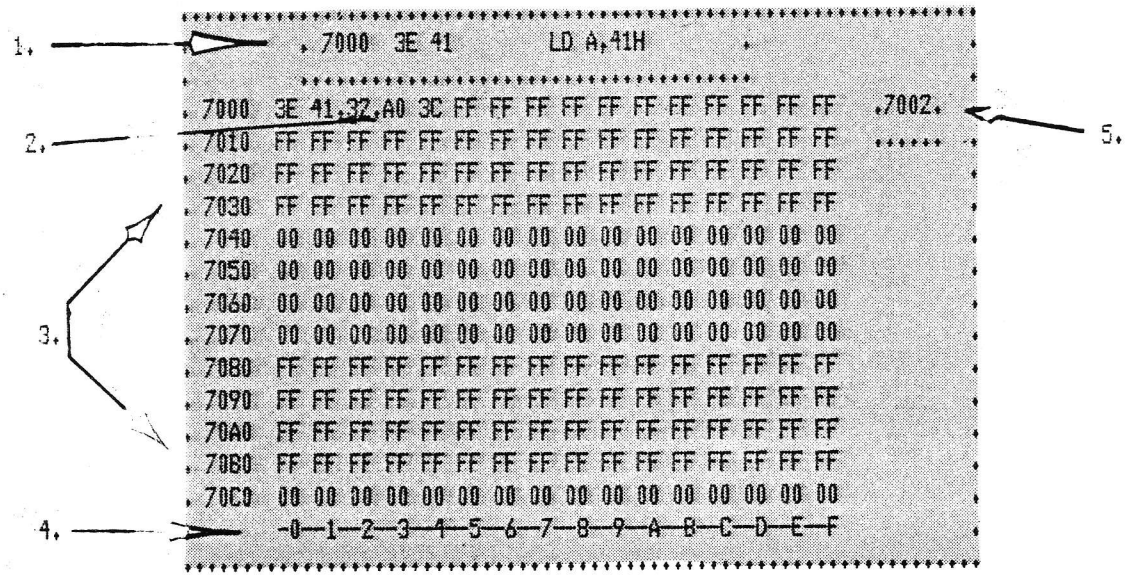
Fig. 3. Immediate addressing display, using the "I" key.

```
          .REL  7002  32 A0 3C 3D 10 FA C9 FF  .
          ..................................
. 7000  3E 41 32 A0 3C 3D 10.FA.C9 FF FF FF FF FF FF FF   .7007.
. 7010  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ......
. 7020  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
. 7030  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
. 7040  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
. 7050  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
. 7060  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
. 7070  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
. 7080  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
. 7090  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
. 70A0  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
. 70B0  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
. 70C0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
.       -0--1--2--3--4--5--6--7--8--9--A--B--C--D--E--F
```

Fig. 2. Relative addressing display, using the "R" key.

```
          .0100 . (.  M E M O R Y   S I Z E.
          ..................................
. 0080  21 E5 41 36 3A 23 70 23 36 2C 23 22 A7 40 11 2D
. 0090  01 06 1C 21 52 41 36 C3 23 73 23 72 23 10 F7 06
. 00A0  15 36 C9 23 23 23 10 F9 21 E8 42 70 31 F8 41 CD
. 00B0  BF 1B CD C9 01 21 05 01 CD A7 2B CD B3 1B 38 F5
. 00C0  D7 B7 20 12 21 4C 43 23 7C B5 28 1B 7E 47 2F 77
. 00D0  BE 70 2B F3 18 11 CD 5A 1E B7 C2 97 19 EB 2B 3E
. 00E0  BF 46 77 BE 70 20 CE 2B 11 14 44 DF DA 7A 19 11
. 00F0  CE FF 22 B1 40 19 22 A0 40 CD 4D 1B 21 11 01 CD
. 0100 .A7.28 C3 19 1A 4D 45 4D 4F 52 59 20 53 49 5A 45   .0100.
. 0110  00 52 41 44 49 4F 20 53 48 41 43 4B 20 4C 45 56   ......
. 0120  45 4C 20 49 49 20 42 41 53 49 43 0D 00 1E 2C C3
. 0130  A2 19 D7 AF 01 3E 80 01 3E 01 F5 CF 2B CD 1C 2B
. 0140  FE 80 D2 4A 1E F5 CF 2C CD 1C 2B FE 30 D2 4A 1E
.       -0--1--2--3--4--5--6--7--8--9--A--B--C--D--E--F
```

Fig. 4. ASCII line display, using the "/" key.



FLTCUR DISPLAY

1. Display line, activated by SPACEBAR (disassemble location under the cursor).

2. Cursor. Move it around with the arrow keys.

3. Memory display, 13 X 16 = 208 bytes.

4. Column low-digit.

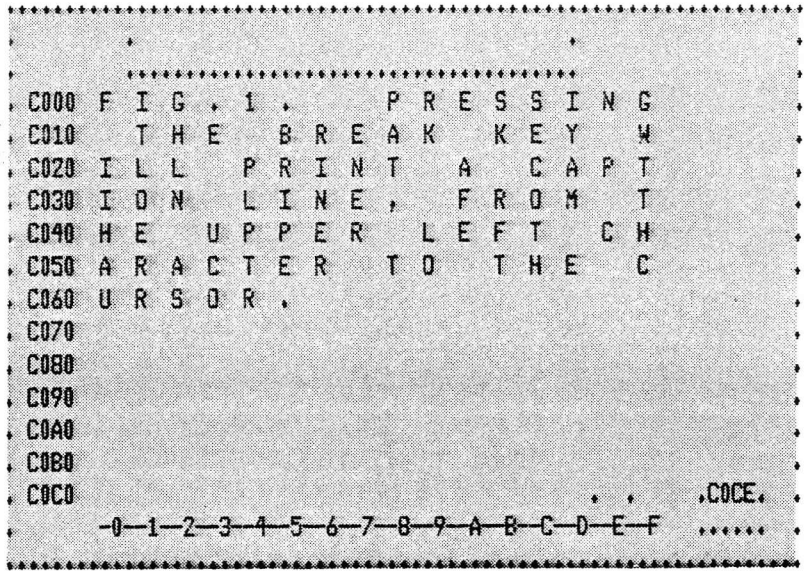
5. Current cursor location.

FLTCUR is entered from CP1 by keying N nnnn and from SPRSTP or BUFSTF by keying N . You may enter hex bytes 00-FF into the location under the cursor.

	KEY	KEY FUNCTION
	Getting around.	
1.	ENTER	Returns control to Control Point 1.
2.	S	Transfer control to SPRSTP over current cursor position.
3.	U	Transfer control to BUFSTF.
4.	(SH) @	Switch to ASCII cursor (see below).
	Cursor control (* denotes autorepeating key)	
5.	↑↓←→ arrow keys	* Move cursor over the material.
6.	(SH) ↓↑←→ arrow keys	* Move the material under the cursor.
7.	(SH) 0 (zero)	* Fill location under cursor with 00.
8.	M nnnn	Reposition the material so that the current line starts at nnnn.
9.	(SH) R	Relative reposition. The byte under the cursor is taken as a twos complement displacement and the material is repositioned relative to this offset. For following instructions using the Z80 Relative Addressing mode.
10.	(SH) I	Immediate reposition. The byte under the cursor is taken as the low byte of a two byte address and the material is repositioned with the cursor over this location. For following instructions using the Z80 Immediate Addressing mode.
11.	Y	Reference reposition. Whenever (SH) R or (SH) I are used the original cursor location is saved for reference. When "Y" is keyed the material is repositioned to this reference location.
	Activating the display line.	
12.	R	Display eight bytes at Relative location.
13.	I	Display eight bytes at Immediate location.
14.	/	Display the current line in ASCII.
15.	SPACE BAR	Disassemble the current instruction.

ASCII cursor

ASCII cursor is entered from FLTCUR by keying (SH) @. Keys pressed under this mode are entered into the location under the cursor as ASCII values. Be careful, you can wipe out program material quite easily since almost every key is alive.

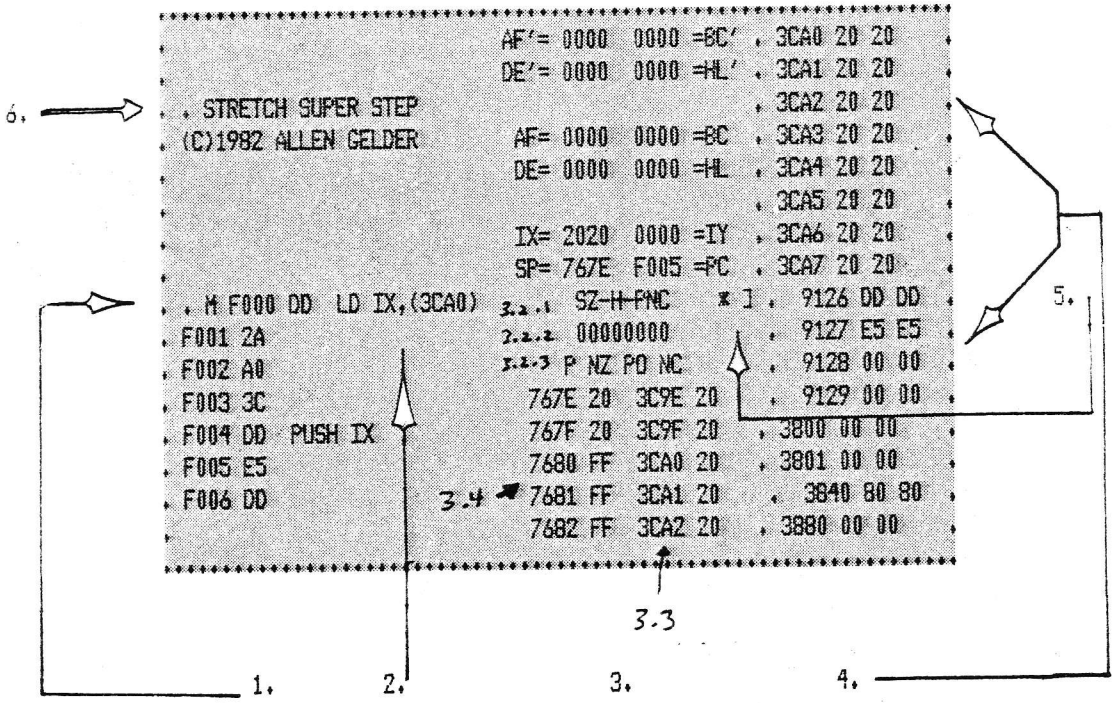


	KEY	KEY FUNCTION
	Getting around.	
1.	ENTER	Returns control to Control Point 1.
2.	(SH) @	Returns control to FLTCUR.
	Cursor control (* denotes autorepeating key)	
3.	↓↑ ← → arrow keys	* Move cursor over the material.
4.	(SH) ↑ (SH) ↓	* Move the material up, down one line.
5.	(SH) ENTER nnnn	Reposition the material so that the current lines starts at nnnn.
	Comment line to printer	
6.	(SH) BREAK	Prints a line up to 80 characters long. The line starts at the upper left of the screen and continues to the cursor. A crude but effective way to comment screen pix, etc. A good scratchpad area is the BUFSTF buffer (see CP1, BUFSTF).

The most important SPRSTP key commands are:

1.	ENTER	Return to CP1.
5.	00-FF	Enter hex byte.
7.	↓	Advance memory without change.
.	M nnnn	Display memory at nnnn.
19.	(SH) Z	Open cursor over Z80 model.
26.	/	Simulator bar to shut off simulation.
28.	SPACEBAR	Single-step/disassemble.

29. (SH) ENTER Start TRACE. To pause trace key SPACEBAR. To cease TRACE then key ENTER.



SPRSTP DISPLAY

1. Memory location and contents byte

2. Assembly mnemonic

3. Z80 Model
   3.1 CPU registers
   3.2 F (flag) register expansion
      3.2.1 Header (Sign, Zero, not used, Half-carry, not used, Parity/Overflow, Add-Subtract, Carry)
      3.2.2 Bit status
      3.2.3 Assembly mnemonic
   3.3 Intelligent Ram Window
   3.4 Stack
4. Ram Windows
   4.1 Memory location
   4.2 Current contents byte
   4.3 Previous contents byte (prior to most recent use of SPACEBAR Single-step or (SH) ENTER TRACE

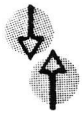
8-location Window
4-location Window
2-location Window
1-location Windows

5. Status Panel
   5.1 x CALL/RST instructions executed in real-time
   blank CALL/RST instructions simulated instruction-by-instruction
   5.2 / Z80 models inactive, straight-line disassembly
   blank Z80 models active, simulation takes place in program-flow order
   5.3 Intelligent RAM Window active
   5.4 . Printer on. Single-step and TRACE activity routed to printer
   blank Printer off

6. Copyright notice

SPRSTP

SPRSTP is entered from CP1 by keying M nnnn. The user may key in hex bytes (00-FF) under this control point. It is very powerful because it has byte-editing (1-D) and a detailed simulator available at any given keystroke. (* denotes autorepeating keys).

	KEY	KEY FUNCTION
	Getting around.	
1.	ENTER	Return control to CP1.
2.	N	Transfer control to FLTCUR.
3.	U	Transfer control to BUFSTF.
4.	P	Print screen picture.
	M nnnn	Display nnnn, contents.
	Editing	
5.	00-FF	Enter hex byte.
6.	(SH) 0	* Write 00 at current location.
7.		* Advance to the next location without changing the current byte.
8.		* Backspace. Advance to previous location without change.
9.	' (key)	Enter the ASCII equivalent of (key).
10.	(SH) 	* Display the current byte in ASCII, advance to next location.
11.	(SH)	* ASCII Backspace. Display byte in ASCII and advance to previous location.
12.	R	Relative display. The current byte is interpreted as the displacement byte of a Relative Addressing Mode instruction. The effective address (and contents byte) are displayed.
13.	I	Immediate display. The current byte is interpreted as the high byte of a two byte address. This location and its contents are displayed.
14.	(SH) R	Relative Space. Advances memory to the Relative location (and loads Reference Space).
15.	(SH) I	Immediate Space. Advanes memory to the Immediate location (and loads Reference Space).
16.	Y	Reference Space. Advances memory to Reference location, ie the site of the most recent (SH) R or (SH) I keying.

KEY	KEY FUNCTION

The following two keys are used to insert and delete bytes. As currently set up, the user must place a string delimiter after the last byte of the string to be moved. The delimiter consists of two FF's, e.g.

```
7000 01
7001 02
7002 03
7003 04
7004 FF
7005 FF
```

If (SH) > or (SH) < were keyed at 7000 the string would be defined as extending fron 7000-7005

The string is defined from the current memory location down (higher in memory) to the FFFF delimiter.

17. (SH) > — Insert byte. Moves string down one space and rotates the byte overwritten by the leading FF into the opened location.

18. (SH) < — Delete byte. Moves string up one space, leaves an FF at the foot of the string.

Formatting the Z80 models

19. (SH) Z — Opens a cursor over the Z80 model, you can then enter a byte (00-FF), move the cursor with → right arrow or ← left arrow. Return to SPRSTP by keying ENTER.

20. (SH) W — Opens a cursor over the RAM Window display at the right of the screen, you may enter addresses byte by byte, move the cursor → right arrow or ← left arrow. Return to SPRSTP by keying ENTER.

21. W — Alternates the RAM Window display at screen right with a secondary unlabeled Z80 model. This model runs one step behind the labeled model during simulation (single-step or TRACE) to entry conditions prior to the action of the current instruction.

22. Z — Clears the right screen and suppresses further display of the Z80 model setup (although note well that simulation will still place under Single-step and TRACE.) . The next time you press "Z" you will get the models back.

23. T — Turns on the instruction-time display. During simulation (single-step or TRACE) the time each instruction takes is posted in seconds, also a total is accumulated and display as the ET. Instruction times are given on a 1.774 mhz clock basis. The 2.028 mhz clock times are 87.5% of the displayed values. Also, every instruction simulated will be logged to a user-defined buffer for subsequent analysis (see V, (SH) V below). Also COUNTM.

24. (SH) T — Zeros the instruction-time display.

25. CLEAR — Clears the screen.

	KEY	KEY FUNCTION
Simulator control		
26.	/	Simulator bar. Use of this key controls whether or not the Z80 models are active during Single-step or TRACE. When "/" appears in the status panel it means that the Z80 models are not active and that disassembly will proceed in straight-line order. Pressing "/" will invoke the Z80 models (and the "/" will disappear from the status panel). Now the Z80 models are active during Single-step and TRACE and disassembly/simulation will follow program-flow order. Press "/" again to return to straight-line order.
27.	x	CALL control. Pressing x will cause instructions in the CALL goup to be executed directly in real-time. Note that the subject routine must return normally for this to work; should a new return address be put onto the stack by the called routine then control will be seized by the subject program. (Return to STR32 by resetting, SYSTEM x? /35830, to STR48 by resetting, SYSTEM x? /52214). Pressing x again will turn out the x in the status panel—CALLs will be simulatively executed instruction-by-instruction.
28.	SPACE BAR	x Single-steps current instruction. The disassembler is active on the first byte of the instruction; the Z80 models are active on the last byte of the instruction. SPACEBAR Single-step is the most important commmand in the program. Remember that if the "/" simulator bar appears in the status panel then only the dissassembler will be active, and that in straight-line order.
29.	(SH) ENTER	TRACE (automatic single-stepping). Correct use of TRACE on a subject program can be a lot of fun; the effect is that the original program is running, slowly, but interesting machine state information is displayed at the same time. The user may pause the TRACE by keying SPACEBAR and change display formats or communicate with the Z80 models to formally influence the subject program. The Z80 instruction 76 HALT is employed as a breakpoint if B5 has been set (see CP1, Bn Breakpoints). Return to SPRSTP is accomplished by the key sequence SPACEBAR,ENTER.

The TRACE keys are available after the user pauses the TRACE by keying SPACEBAR. Except for ENTER, each key returns to TRACE.

TRACE KEY	TRACE KEY FUNCTION
SPACE BAR	Pauses the TRACE and awaits a key command, as listed below. Subsequent pushes of SPACEBAR work as follow: a firm push will single-step the instruction, a quick push will return control to TRACE.

Now you may key any of:

ENTER	Halt the TRACE and return control to SPRSTP.
2	Fast TRACE speed.
1	Slow TRACE speed

	TRACE KEY	TRACE KEY FUNCTION
4.	M	Sets F register bit 7 (sign flag minus)
5.	P	Resets register bit 7 (sign flag plus)
6.	(SH) Z	Sets register bit 6 (zero flag).
7.	NZ	Resets register bit 6 (zero flag).
8.	E	Sets register bit 2 (overflow flag).
9.	O	Resets register bit 2 (overflow flag).
10.	C	Sets register bit 0 (carry flag).
11.	NC	Resets register bit 0 (carry flag).
12.	D	Loads B register with 01 to fall out of DJNZ loops.
13.	/	Skip the next instruction (it's still disassembled).
14.	x	Controls CALL execution as in SPRSTP. If x appears in the status panel then CALLS take place in real-time, if not CALLs will be executed single-step-wise.
15.	Z	Controls right screen format (Z80 models or transparent screen).
16.	W	Controls RAM Window/secondary Z80 model display.
17.	X	Controls the size of the scrolling area at the left screen-full scrolling or 2-line scrolling into screen locations stored in A51C-1D and A51E-1F.
18.	T	Controls instruction timer/counter.
19.	CLEAR	Clears the screen.
	any other key	Trace execution continues unchanged.

BUFSTF

Use of BUFSTF takes place in several steps:

1. Define the buffer while in Control Point 1, using (SH) U mmmm nnnn, where mmmm is the start address of the buffer and nnnn is the end address of the buffer, e.g. (SH) U C000 EFFF will clear C000-EFFF for subsequent use.

2. Load the buffer while in Control Point 1. The formats are described below. Buffer format I is used to make reference copies of memory you are working on, viewing disassembled listings in straight-line or program-flow order, printing listings, etc. The - mmmm nnnn, ; mmmm nnnn and + mmmm nnnn keys are used. Buffer Format II is used for analyzing programs for content. You may isolate instances of certain instructions (see below).

3. Open a window over the buffer by keying "U". Now the arrow keys will let you browse over the material in the buffer. You may print a page of the buffer by keying "(".

BUFFER FORMAT I

```
7000 3E LD A,41H          7010 FF  RST 38
7001 41                   7011 FF  RST 38
7002 32 LD (3CA0),A       7012 FF  RST 38
7003 A0                   7013 FF  RST 38
7004 3C                   7014 FF  RST 38
7005 FF  RST 38           7015 FF  RST 38
7006 FF  RST 38           7016 FF  RST 38
7007 FF  RST 38           7017 FF  RST 38
7008 FF  RST 38           7018 FF  RST 38
7009 FF  RST 38           7019 FF  RST 38
700A FF  RST 38           701A FF  RST 38
700B FF  RST 38           701B FF  RST 38
700C FF  RST 38           701C FF  RST 38
700D FF  RST 38           701D FF  RST 38
700E FF  RST 38           701E FF  RST 38
700F FF  RST 38           701F FF  RST 38
```

Fig. 1. Disassembled (- mmmm nnnn).

```
0105 4D M                 0115 4F O
0106 45 E                 0116 20
0107 4D M                 0117 53 S
0108 4F O                 0118 48 H
0109 52 R                 0119 41 A
010A 59 Y                 011A 43 C
010B 20                   011B 4B K
010C 53 S                 011C 20
010D 49 I                 011D 4C L
010E 5A Z                 011E 45 E
010F 45 E                 011F 56 V
0110 00 .                 0120 45 E
0111 52 R                 0121 4C L
0112 41 A                 0122 20
0113 44 D                 0123 49 I
0114 49 I                 0124 49 I
```

Fig. 2. Hex listing (; mmmm nnnn).

```
0105 4D M                 0115 4F O
0106 45 E                 0116 20
0107 4D M                 0117 53 S
0108 4F O                 0118 48 H
0109 52 R                 0119 41 A
010A 59 Y                 011A 43 C
010B 20                   011B 4B K
010C 53 S                 011C 20
010D 49 I                 011D 4C L
010E 5A Z                 011E 45 E
010F 45 E                 011F 56 V
0110 00 .                 0120 45 E
0111 52 R                 0121 4C L
0112 41 A                 0122 20
0113 44 D                 0123 49 I
0114 49 I                 0124 49 I
```

Fig. 3. Hex ASCII listing (+ mmmm nnnn).

BUFFER FORMAT II

```
0003 36 3A      LD (HL),3AH   0189 12        LD (DE),A
0006 78         LD (HL),B     018A 32 99 40  LD (4099),A
0008 36 2C      LD (HL),2CH   018D 77        LD (HL),A
000B ZZ A7 40   LD (40A7),HL  018F ZZ 21 41  LD (4121),HL
0096 36 C3      LD (HL),C3H   01C4 32 AF 40  LD (40AF),A
0099 73         LD (HL),E     01C5 32 AB 40  LD (40AB),A
009B 72         LD (HL),D
00A1 36 C9      LD (HL),C9H
00AB 70         LD (HL),B
00CF 77         LD (HL),A
00D1 78         LD (HL),B
00E2 77         LD (HL),A
00E4 78         LD (HL),B
00F2 ZZ B1 40   LD (40B1),HL
00F6 ZZ A8 40   LD (40A8),HL
0102 12         LD (DE),A
```

Fig. 4. RAM-affecting instructions (! mmmm nnnn).

```
0060 FE 02      CP 02H        03D0 B8        CP B
0004 BE         CP (HL)       03D1 FE 02     CP 02H
00E3 BE         CP (HL)       0412 FE 68     CP 68H
0140 FE 80      CP 80H        042F FE 3C     CP 3CH
0148 FE 30      CP 30H        0453 FE 01     CP 01H
0257 CB 10      RL B          0468 FE 20     CP 20H
029B FE A5      CP A5H        046D FE 80     CP 80H
02CA FE 2F      CP 2FH        0471 FE 40     CP 40H
02D4 FE 55      CP 55H        0477 FE 28     CP 28H
02E1 BE         CP (HL)       04A6 FE C0     CP C0H
02ED FE 78      CP 78H        04AA FE 08     CP 08H
02F1 FE 3C      CP 3CH        04AE FE 0A     CP 0AH
030A B9         CP C          0511 FE 0E     CP 0EH
03A2 FE BC      CP BCH        0517 FE 0F     CP 0FH
03A6 FE 0A      CP 0AH        051B FE 17     CP 17H
03AD FE 0D      CP 0DH        051F FE 18     CP 18H
```

Fig. 5. Comparison instructions (" mmmm nnnn).

```
0002 C3 74 06   JP 0674       003E 18 D8     JR 0018
0005 C3 00 40   JP 4000       0040 C3 D9 05  JP 05D9
0008 C3 00 40   JP 4000       0043 C9        RET
000C E9         JP (HL)       0046 C3 C2 03  JP 03C2
000D C3 9F 06   JP 069F       0049 CD ZB 00  CALL 002B
0010 C3 03 40   JP 4003       004D C0        RET NZ
0016 18 2E      JR 0046       004E 18 F9     JR 0049
0018 C3 06 40   JP 4006       0058 18 09     JR 0066
001E 18 26      JR 0046       005E 20 20     JR NZ,0080
0020 C3 09 40   JP 4009       0063 20 FB     JR NZ,0060
0026 18 1E      JR 0046       0065 C9        RET
0028 C3 0C 40   JP 400C       006F DZ 00 00  JP NC,0000
002E 18 E3      JR 0013       0072 C3 CC 06  JP 06CC
0030 C3 0F 40   JP 400F       0090 10 F7     DJNZ 0096
0036 18 E3      JR 001B       00A6 10 F9     DJNZ 00A1
0038 C3 12 40   JP 4012       00AF CD 0F 1B  CALL 1B0F
```

Fig. 6. Program-flow affecting instructions (# mmmm nnnn).

```
007E ED B0      LDIR          0407 DD 7E 05  LD A,(IX+05)
0103 ED 5F      LD A,R        040D DD 72 05  LD (IX+05),D
0257 CB 10      RL B          0492 DD 75 03  LD (IX+03),L
03C3 DD E5      PUSH IX       0495 DD 74 04  LD (IX+04),H
03C6 DD E1      POP IX        049A DD 7E 05  LD A,(IX+05)
03D6 DD 6E 01   LD L,(IX+01)  04B9 DD 77 05  LD (IX+05),A
03D9 DD 66 02   LD H,(IX+02)  055E ED B8     LDIR
03DE DD E1      POP IX        059A DD B6 03  OR (IX+03)
03F4 CB 01      RLC C         059F DD 7E 03  LD A,(IX+03)
0407 CB 01      RLC C         05A2 DD 96 04  SUB A,(IX+04)
0416 CB 08      RRC B         05C1 DD 34 04  INC (IX+04)
0435 CB 08      RRC B         05C4 DD 7E 04  LD A,(IX+04)
043E CB 08      RRC B         05C7 DD BE 03  CP (IX+03)
0458 DD 6E 03   LD L,(IX+03)  05CC DD 36 04 00 LD (IX+04),00H
045B DD 66 04   LD H,(IX+04)  067F ED B8     LDIR
0460 DD 7E 05   LD A,(IX+05)  0682 CB 46     BIT 0,(HL)
```

Fig. 7. User choice ($ mmmm nnnn) instructions.
aa bb cc dd

BUFSTF is entered from CP1 by keying "U" (after first defining the buffer location and loading it while under CP1, see below). Some of the load formats can be merged. This helps prepare listing for page printing. Another use of BUFSTF is to establish a reference copy of code areas being modified.

	KEY	KEY FUNCTION
1.	ENTER	Returns control to Control Point 1.
2.	S	Transfers control to SFRSTP.
3.	N	Transfers control to FLTCUR.
4.	↑↓←→ arrow keys	* Move material for viewing. ↑ and ↓ arrows scroll by line, ← and → arrows move material left or right by the half-screen.
5.	P	Print screen picture.

Related keys used under CP1

6.	(SH) U nnnn mmmm	Defines the location of the BUFSTF buffer with start at nnnn and and ending at mmmm. Typical buffer locations might be (48K) C000-EFFF, (32K) 5400-73FF. Smaller (or larger) are OK too. The buffer locations will be blanked (20H).

Buffer format I. Used to make listings for reference or printing (Using "P" to print a screen picture or "(" to print a buffer page. The three formats can be combined.

7.	- mmmm nnnn	Load buffer with disassembled listing of the code found between mmmm and nnnn. The order is straight-line if "/" is on, program flow order is followed.
8.	; mmmm nnnn	Load buffer with hex listing of the code found between mmmm and nnnn. (Straight-line order).
9.	+ mmmm nnnn	Load buffer with hex/ASCII listing of the code found between mmmm and nnnn. (Straight-line order).

The above three keys are combinable in the following sense. After the particular key function is complete a prompt comes up. You may respond with one of these keys and another address range, or you may key "ENTER" to return to CP1. Example:

(SH) U C000 EFFF	Defines buffer from C000-EFFF.
- 0080 00FF	Writes a disassembled listing of 0080-00FF.
; 0100 010F	Writes hex listing of 0100-010f.
+ 0110 014F	Writes hex/ASCII listing of 0110-014F.
- 0150 01FF	Writes disassembled listing of 0150-.... (buffer full).

Buffer format II. Used to isolate instances of specified instructions and log to buffer for skeleton program views within the given range. Note that this can be straight-line or program-flow order, depending on the condition of "/". These functions are combinable; a prompt comes up after the function is finished and the user may enter a new command (!, ", # or $) or exit back to CP1 by keying "ENTER".

	KEY	KEY FUNCTION
10.	! mmmm nnnn	Load buffer with all instances of memory-affecting instructions found between mmmm and nnnn. The following instructions are included:
11.	" mmmm nnnn	Load buffer with all instances of comparison instructions found between mmmm and nnnn. The instructions included are:
12.	# mmmm nnnn	Load buffer with all instances of program-flow affecting instructions found between mmmm and nnnn, including:
13.	$ mmmm nnnn aa bb cc dd	Load buffer with all instances of aa, bb, cc, and dd (hex bytes (hex bytes 00-FF) found between mmmm and nnnn. For instance: $ 0000 37FF DD FD ED CB will locate and instructions with DD, FD, ED and CB as leading leading bytes.

Note that Buffer format I and Buffer format II are not inter-combinable; they store the material in different ways. To force Buffer format I key "_" "ENTER", to force Buffer format II key "$" "ENTER".

Setting up the Z80 models, formatting the screen, etc.

Q. How do I change the Z80 model?

A. In CP1 or SPRSTP, key (SH) Z. This opens a blob-cursor in the Z80 model (over the A register). You can enter byte values or move the cursor around by using → right arrow or ← left arrow. When you're done key ENTER to get back to CP1 or SPRSTP. (Also, the Z80 model changes when instructions are being simulated. More below.)

Q. How do I change the RAM Window setup?

A. Same as the Z80 model, only key (SH) W to get the cursor opened over the RAM Windows. Exit by keying ENTER when you've finished.

Q. What other screen formats are available?

A. You can replace the RAM Window display (at the extreme right of the screen) with a second Z80 model. Actually, this is an echo of the first Z80 model (the one that has labeled registers) that runs one instruction behind when the simulation mode is active (Single-step or TRACE). Another format is a clear right screen (except for the status panel in the upper right). You get a clear right screen by keying "Z" under CP1 or SPRSTP. The next time you key "Z" the Z80 models are returned.

Q. Why two Z80 models, again?

A. Because the secondary model preserves the status prior to the execution of the instruction you have simulated. You see everything twice at once, exit/entry, after/before. A permanent running backstep.

Q. How can I change the scrolling at the left of the screen?

A. Press "X" when in CP1. This will set the reduced scrolling mode so that two locations will scroll. You can set the place on the screen where you want each line by changing A51C-1D and A51E-1F. To get back to full scrolling key "X" again while in CP1.

Examining and changing memory.

Q. How do I examine memory?

A. There are three main ways:

(1) From Control Point 1 (CP1) key M nnnn (where nnnn is a 4-digit hex address, such as 0000, FFFF, 1A2B, etc.) to enter the SPRSTP control point. As soon as you type in the location its contents byte will be displayed. Press ↓ down arrow to see the next location and byte. Press ↑ up arrow to see the previous location and byte.

(2) From CP1 key N nnnn. This gets you the FLTCUR display and 13 lines of memory. Press (SH)↑ up arrow or (SH)↓ down arrow to scroll the display. (You can also scroll sideways using (SH)→ right arrow or (SH)← left arrow.)

(3) Set up the BUFSTF buffer and load it (better see the BUFSTF section about this method.)

Q. How about in ASCII.

A. After you're in SPRSTP (M nnnn ..) key (SH) ↓ down arrow or (SH) ↑ up arrow and the ASCII is displayed. In FLTCUR (N nnnn ..) key (SH) @ and the whole display goes ASCII. You can change it back to hex by keying (SH) @ again.

Q. How do I change the bytes?

A. In SPRSTP just enter a byte (00-FF); after you've entered it the display will scroll and you're ready for the next location. To enter in ASCII press the ' and then any key. The ASCII value of the key will be entered. For instance 'A will load the location with the hex byte 41. To change memory in the FLTCUR mode just enter the byte; it will be loaded into the location under the cursor. If you've pressed (SH) @ and are in the ASCII mode your keystrokes will be entered in ASCII.

Q. How about some simulator examples?

A. OK, here are three.

I. Single-stepping a ROM routine. First we set up the simulator from CP1.

	YOU PRESS ("." is prompt)	REASON
1.	. B5	Enables the 76-HALT breakpoint to halt the simulator during single-step/TRACE. Check by keying / .
2.	. M 7200 CD	Enter the bytes under SPRSTP or FLTCUR.
	7201 2B	
	7202 00	
	7203 FE	
	7204 0D	
	7205 20	
	7206 F9	
	7207 76	
3.	. M 7200 CD CALL 002B	Now press SPACEBAR to single-step or (SH) ENTER
	7201 2B	begin the TRACE. This is a ROM keyboard-scan
	7202 00	routine that emerges with the A register
	7203 FE CP 0DH	containing the ASCII value of the key pressed.
	7204 0D	Experiment with the TRACE features, including the
	7205 20 JR NZ, 7200	* operation that controls the CALL execution.
	7206 F9	Remember that to halt the TRACE you must key
	7200 CD CALL 002B	SPACEBAR, then key ENTER. In this example you
	etc.	can also halt the TRACE by keying ENTER (0D).
	etc.	

II. Single-stepping DEMO/CIM, a simple graphics loop. Load DEMO/CIM from DOS READY, then load STR32/CMD or STR48/CMD. To successfully TRACE or single-step DEMO/CIM we will have to use either reduced scrolling or reduced display. This is because DEMO/CIM conflicts with STRETCH in using the screen.

1.	. X	Set reduced scrolling from CP1.
2.	. M 70E0 CD CALL 7060	Use single-step or TRACE with * enabled.
	70E1 00	Halt the TRACE by using SPACEBAR and then (SH)
	etc.	ENTER.

Now we go back to full scrolling and reduce the display. A different DEMO/CIM entry point will reposition the DEMO material.

3.	. X	Restore full scrolling
4.	. Z	Now right screen is transparent.
5.	. M 70F0 CD CALL 7068	
	70F1 68	
	70F2 70	
	70F3 18 JR 706B	
	70F4 F6	
	etc.	

III. Single-stepping the BASIC ROM. This activity is fraught with difficulties caused by the fact that many ROM subroutines don't return "normally", that is, to the location following the address of the CALLing routine. Because of this construction, use of the * (direct execution of subroutines) can cause control to pass to BASIC. When this happens use *? SYSTEM *? /35830 (STR32/CMD) or *? /52214 (STR48/CMD). However, much of BASIC can be used with * enabled and without loss of control. You may write, LIST and RUN programs without leaving the simulator. This can be quite useful if you are interested in how BASIC works. To the simulator the BASIC ROM is just another machine language program.

	YOU PRESS ("." is prompt character)	REASON
1. From DOS READY		
	LOAD STR32/CMD	We have to simulate Level II BASIC because disk operations return control to BASIC.
2.	BASIC2	Or use the Reset button while holding down the BREAK key.
3.	*? SYSTEM *? /35830	Enter STRETCH SuperStep. Or *? /52214 for STR48/CMD.
4.	. X	Set reduced scrolling.
5.	. M 06CC 01 LD BC, 1A18H 06CD 18 06CE 1A 06CF C3 JP 19AE etc.	Make sure * is enabled and turn on TRACE by keying (SH) ENTER. Use the fast TRACE speed by keying SPACEBAR, then 2 . NOTE: Mod III use 1997 as entry point.

What should happen is the eventual appearance of the READY prompt. BASIC is now waiting in a subroutine for your keyboard input. We write a simple program:

6.	READY AUTO 10 GOSUB30 20 END 30 CLS 40 ?CHR$(95);"<" 50 ?"YDAER" 60 RETURN	The Z80 Models should click madly whenever you press ENTER. Note that the 8-byte Window is positioned over the start of the BASIC text.
7.	LIST	
8.	RUN	

NOTES

1. The most information packed screen in Stretch SuperStep is in the Single-step/TRACE mode of SPRSTP. You get to see the instruction stream (disassembled), the CPU status including the top of the stack and the flag configuration, and also any memory that the current instruction might be referencing (Intelligent RAM Window). Depending on how the right screen is set up, you can continously monitor up to 16 memory locations or preserve CPU entry conditions in a second Z80 model.

2. The most transparent screen is in SPRSTP, with all the models suppressed and the instruction stream (left side of the screen) in the most reduced scrolling mode--the two scrolling locations superimposed. All the simulation can be taking place with only one 27 space line being overwritten. This means you can simulate subject programs that use the screen effectively. Note also that the TRACE mode is fastest under these conditions.

3. The most dangerous key functions are probably in SPRSTP, the (SH)< and (SH)> (Insert/Delete byte). IF you forget to place the FFFF string delimiter you could move unexpected RAM areas lying above the string, including Stretch SuperStep itself. A real corrupter. Another dangerous mode is the (SH)@ version of FLTCUR, where almost every keystroke causes the ASCII value of the key to be immediately written into the location under the cursor. If that location is not supposed to receive that value, too bad. This is different than when entering bytes; you have a safety net in the fact that bytes require two keystrokes to be entered.

4. There are shoals in the simulator mode too. Like when you sail into LDIR (Load, Increment and Repeat) instructions with the BC (count) register unsecured. Or those Register Indirect instructions, where an unloaded register might shoot a hole in your program, or even Stretch SuperStep itself. In fact, any instruction which affects RAM can affect the WRONG RAM.

5. Things a subject program might do to vex simulation: Change its subroutine return addresses inside the subroutine. If the simulator is set to execute subroutines in real-time (the (SH)X set) then control will

pass directly to the subject program. To get back to Stretch SuperStep you must hit RESET and BREAK at the same time and use SYSTEM *? /35830.

Note that when you next enter SPRSTP (M nnnn ..) the simulation will want to take off--be ready to hit the SPACEBAR (pause) and then ENTER (halt) and make sure that nnnn doesn't harbor a dangerous instruction.

Another potential bomber is when a subject program tests memory by XORing or in some way changing successive locations. If this testing passes through Stretch SuperStep then by the pigeon-hole principle it will corrupt some byte the moment the real-time PC is pointing to that location. This will almost always be catastrophic to the simulation.