SNA Format (Snapshot)

The SNA format is one of the most widely used for snapshots of the ZX Spectrum. Its popularity is due to its simplicity and extended support in emulators. However, it has a known limitation: two bytes of memory can become corrupted because the program counter (PC) is temporarily stored on the stack during the snapshot saving process.

This issue arises because the **PC** register value is "pushed" onto the stack, overwriting two bytes at the stack pointer (**SP**) position. While this usually does not affect the saved program, there can be cases where the stack space is fully in use, which could cause memory corruption. A suggested solution to mitigate this problem is to replace the corrupted bytes with zeros and then increment the stack pointer.

When a snapshot is restored, the program resumes its execution with a **RETN** command, returning control to the point where the snapshot was saved. The **IFF2** register indicates the state of interrupt enablement; when active, it means interrupts were enabled at the time of capture.

SNA File Structure (48K)

Offset	Size	Description
0	1 byte	**I** Register
1	8 bytes	Alternate registers: **HL'**, **DE'**, **BC'**, **AF'**
9	10 bytes	Main registers: **HL**, **DE**, **BC**, **IY**, **IX**
19	1 byte	Interrupt register (bit 2 contains **IFF2**, 1 = enabled, 0 = disabled)
20	1 byte	**R** Register
21	4 bytes	**AF**, **SP** Registers
25	1 byte	Interrupt mode (**IM**) (0 = IM0, 1 = IM1, 2 = IM2)
26	1 byte	Border color (0 to 7)
27	49152 bytes	RAM dump from address **16384** to **65535**

Total size: 49179 bytes

SNA Format (128K)

The 128K SNA format is an extension of the 48K format, including additional RAM banks. It also fixes the issue related to the **PC** register, which is now stored in an additional variable instead of being pushed onto the stack.

Offset	Size	Description
0	27 bytes	SNA Header (identical to the 48K format)

Offset	Size	Description
27	16 KB	RAM Bank 5 (paged at address 0xC000)
16411	16 KB	RAM Bank 2
32795	16 KB	Currently paged RAM bank
49179	2 bytes	**PC** Register
49181	1 byte	**0x7FFD** Port state (paging control)
49182	1 byte	**TR-DOS** ROM paged (1) or not (0)
49183	16 KB	Remaining RAM banks in ascending order (0, 1, 3, 4, 6, 7)

Total size: 131103 or 147487 bytes, depending on the number of banks saved.

Additional Details:

- The third saved RAM bank is always the currently paged bank, even if it is Bank 5 or 2, meaning the same bank can appear twice in the snapshot.
- The remaining banks are saved in ascending order. For example, if Bank 4 is paged at the time of capture, the file will contain Banks 5, 2, and 4, followed by Banks 0, 1, 3, 6, and 7. Conversely, if Bank 5 is paged at the time of capture, the file will contain Banks 5, 2, and 5, followed by Banks 0, 1, 3, 4, 6, and 7.

Cases Where ROM is Included

In a new variant of the SNA format for **48K**, if the file size is **65563 bytes** instead of the standard **49179 bytes**, it means the snapshot includes a 16KB dump of the ZX Spectrum's ROM. This extra ROM block is stored immediately after the header and before the RAM dump from address **16384** to **65535**. (This variant was introduced by Juan José Ponteprino in the ESPectrum project.)

In this case, the structure is as follows:

Offset	Size	Description
0	27 bytes	SNA Header
27	16 KB	ROM dump
16411	48 KB	RAM dump from address 0x4000 to 0xFFFF

Total size: 65563 bytes

SP Format

Overview

The SP format is used to store programs in the ZX Spectrum's memory. This format includes information about the Z80 processor state and registers, as well as the state of memory and other relevant settings.

SP File Structure

The SP file is structured as follows:

Offset	Size	Description
0	2 bytes	"SP" (0x53, 0x50) Signature.
2	1 word	Program length in bytes (usually 49152 bytes).
4	1 word	Initial program position (usually position 16384).
6	1 word	Z80 BC register.
8	1 word	Z80 DE register.
10	1 word	Z80 HL register.
12	1 word	Z80 AF register.
14	1 word	Z80 IX register.
16	1 word	Z80 IY register.
18	1 word	Z80 BC' register.
20	1 word	Z80 DE' register.
22	1 word	Z80 HL' register.
24	1 word	Z80 AF' register.

Offset	Size	Description
26	1 byte	Z80 R (refresh) register.
27	1 byte	Z80 I (interrupt) register.
28	1 word	Z80 SP register.
30	1 word	Z80 PC register.
32	1 word	Reserved for future use, always 0.
34	1 byte	Border color at startup.
35	1 byte	Reserved for future use, always 0.
36	1 word	Status word encoded by bits. Format:
		Bit: 15-8: Reserved for future use.
		Bit: 7-6: Reserved for internal use, always 0.
		Bit: 5: Flash state: 0 - INK ink, PAPER paper; 1 - PAPER ink, INK paper.
		Bit: 4: Pending interrupt execution.
		Bit: 3: Reserved for future use.
		Bit: 2: IFF2 flip-flop (internal use).
		Bit: 1: Interrupt mode: 0=IM1; 1=IM2.
		Bit: 0: IFF1 flip-flop (interrupt state): 0 - Interrupts disabled (DI); 1 - Interrupts enabled (EI).

Cases Where ROM is Included

If the SP file includes the ROM, both the "program length" field (offset 2) and the "Initial program position" field (offset 4) are set to 0.

Z80 File Format

The .z80 format is undoubtedly the most widely supported by emulators on all platforms. .z80 files are snapshots of memory; they contain an image of the ZX Spectrum's memory contents at a particular point in time. As a result, they cannot be used to recreate the original tape from a snapshot file but are loaded almost instantly.

The .z80 format was originally developed by Gerton Lunter for his Z80 emulator, and three versions of the format are used, as saved in Z80 versions 1.45 (and earlier), 2.x, and 3.x (and later). For easier notation, they will be referred to as versions 1, 2, and 3 of the format, respectively. Several extensions to the .z80 format have also been made by other emulators.

Version 1 of the .z80 format can only save 48K snapshots and has the following header:

Offset	Size	Description
0	1 byte	**A** Register
1	1 byte	**F** Register
2	1 word	**BC** register pair (LSB first, i.e., C first)
4	1 word	**HL** register pair
6	1 word	Program counter
8	1 word	Stack pointer
10	1 byte	**I** Register (interrupt register)
11	1 byte	**R** Register (refresh register) (bit 7 is not significant!)
12	1 byte	Flags for **R** Register
		Bit 0: Bit 7 of the **R** register
		Bit 1-3: Border color
		Bit 4: 1=SamRom Basic enabled
		Bit 5: 1=Compressed data block
		Bit 6-7: No significance
13	1 word	**DE** register pair
15	1 word	**BC'** register pair
17	1 word	**DE'** register pair
19	1 word	**HL'** register pair
21	1 byte	**A'** Register
22	1 byte	**F'** Register

Offset	Size	Description
23	1 word	**IY** Register (again LSB first)
25	1 word	**IX** Register
27	1 byte	Interrupt flip-flop, 0=DI, otherwise EI
28	1 byte	**IFF2** (not particularly important...)
29	1 byte	Bit 0-1: Interrupt mode (0, 1, or 2)
		Bit 2 : 1=Issue 2 emulation
		Bit 3 : 1=Double interrupt frequency
		Bit 4-5: 1=High video synchronization
		3=Low video synchronization
		0,2=Normal
		Bit 6-7: 0=Cursor/Protek/AGF joystick
		1=Kempston joystick
		2=Left Sinclair 2 joystick (or user-defined for .z80 version 3 files)
		3=Right Sinclair 2 joystick

For compatibility, if byte 12 equals 255, it should be considered as 1.

After this 30-byte header block, the Spectrum's 48K of memory follows in compressed format (if bit 5 of byte 12 is set). The compression method is very simple: it replaces repetitions of at least five equal bytes with a four-byte code ED ED xx yy, which means "byte yy repeated xx times." Only sequences of at least 5 bytes are encoded. The exception is sequences consisting of ED; if found, even two EDs are encoded as ED ED 02 ED. Finally, any byte directly following a single ED is not taken as part of a block, for example, ED 6*00 is not encoded as ED ED ED 06 00 but as ED 00 ED ED 05 00. The block ends with an end marker, 00 ED ED 00.

Versions 2 and 3 of .z80 files begin with the same 30-byte header as version 1 files. However, bits 4 and 5 of the flag byte no longer have meaning, and the program counter (bytes 6 and 7) is set to zero to signal a version 2 or 3 file.

After the first 30 bytes, an additional header follows:

Offset	Size	Description
* 30	1 word	Length of additional header block (see below)
* 32	1 word	Program counter
* 34	1 byte	Hardware mode (see below)
* 35	1 byte	If in SamRam mode, bitwise state of 74ls259.
		For example, bit 6=1 after an OUT 31,13 (=2*6+1)

Offset	Size	Description
		If in 128 mode, contains the last OUT to 0x7ffd
		If in Timex mode, contains the last OUT to 0xf4
* 36	1 byte	Contains 0xff if the Interface I ROM is paged
		If in Timex mode, contains the last OUT to 0xff
* 37	1 byte	Bit 0: 1 if **R** register emulation is enabled
		Bit 1: 1 if **LDIR** emulation is enabled
		Bit 2: AY sound is in use, even on 48K machines
		Bit 6: (if bit 2 is set) Fuller Audio Box emulation
		Bit 7: Modify hardware (see below)
* 38	1 byte	Last OUT to port 0xfffd (sound chip register number)
* 39	16 bytes	Sound chip register contents
55	1 word	Lower **T** state counter
57	1 byte	Upper **T** state counter
58	1 byte	Spectator (QL emulation) flag byte
		Ignored by Z80 when loading, zero when saving
59	1 byte	0xff if the MGT ROM is paged
60	1 byte	0xff if the Multiface ROM is paged. Must always be 0.
61	1 byte	0xff if 0-8191 is ROM, 0 if RAM
62	1 byte	0xff if 8192-16383 is ROM, 0 if RAM
63	5 words	5 x User-defined joystick key mappings
73	5 words	5 x ASCII values: keys corresponding to the mappings above
83	1 byte	MGT type: 0=Disciple+Epson, 1=Disciple+HP, 16=Plus D
84	1 byte	Disciple inhibit button state: 0=out, 0xff=in
85	1 byte	Disciple inhibit flag: 0=ROM pageable, 0xff=no
** 86	1 byte	Last OUT to port 0x1ffd

The word value at position 30 is 23 for version 2 files, and 54 or 55 for version 3; the fields marked '*' are those present in the version 2 header. The final byte (marked '**') is only present if the value at position 30 is 55.

In general, the fields have the same meaning in version 2 and version 3 files, except for byte 34:

Value	Meaning in v2	Meaning in v3

Value	Meaning in v2	Meaning in v3
0	48k	48k
1	48k + Interface	48k + Interface
2	SamRam	SamRam
3	128k	48k + MGT
4	128k + Interface	128k
5	-	128k + Interface
6	-	128k + MGT

(Curiously, documentation for Z80 versions 3.00 to 3.02 had the entries for 'SamRam' and '48k + MGT' swapped in the second column of the above table; additionally, bytes 61 and 62 of the format were incorrectly documented until version 3.04. Snapshots produced by earlier versions of Z80 followed the above; only the documentation was wrong).

Other emulators have extended the .z80 format to support more machine types:

Value	Meaning
7	Spectrum +3
8	[Misused by some XZX-Pro versions to indicate a +3]
9	Pentagon (128K)
10	Scorpion (256K)
11	Didaktik-Kompakt
12	Spectrum +2
13	Spectrum +2A
14	TC2048
15	TC2068
128	TS2068

While most emulators using these extensions write version 3 files, some write version 2 files, so it's probably best to assume that any of these values may appear in version 2 or version 3 files.

If bit 7 of byte 37 is set, the hardware types are slightly modified: any 48K machine becomes a 16K machine, any 128K machine becomes a +2, and any +3 machine becomes a +2A.

The high **T** state counter counts upwards in modulo 4. Just after the ULA generates its interrupt once every 20 ms, the counter is 3 and increments by one every 5 ms of emulated time. In these intervals of 1/200 second, the lower **T** state counter counts down from 17471 to 0 (17726 in 128K modes), making a total of 69888 (70908) **T** states per frame.

The 5 ASCII words (the high byte is always 0) at offsets 73-82 are the keys corresponding to joystick directions: left, right, down, up, and fire, respectively. Shift, Symbol Shift, Enter, and Space are denoted by [,], /, , respectively. The ASCII values are only used to display the joystick keys; the information in the 5 keyboard mapping words determines which key is actually pressed (and should correspond to the ASCII values). The low byte is in the range 0-7 and determines the keyboard row. The high byte is a mask byte and determines the column. Enter, for example, is stored as 0x0106 (row 6 and column 1), and 'g' as 0x1001 (row 1 and column 4).

Byte 60 must be zero because the contents of the Multiface RAM are not saved in the snapshot file. If the Multiface was paged when the snapshot was saved, the emulated program will likely crash when reloaded.

Bytes 61 and 62 are a function of the other flags, such as byte 34, 59, 60, and 83.

Next comes a series of memory blocks, each containing the compressed data of a 16K block. Compression is performed according to the old scheme, except the end marker is now absent. The structure of a memory block is:

Byte	Length	Description
0	2	Length of compressed data (not including this 3-byte header)
		If length = 0xffff, the data is 16384 bytes long and uncompressed
2	1	Page number of the block
3	[0]	Data

Pages are numbered depending on the hardware mode as follows:

Page	In 48K mode	In 128K mode	In SamRam mode
0	48K ROM	Basic ROM	48K ROM
1	Interface I ROM, Disciple or Plus D, depending on configuration.	(same as 48K)	(same as 48K)
2	-	Reset ROM	SamRam ROM (basic)
3	-	Page 0	SamRam ROM (monitor,..)
4	8000-bfff	Page 1	Normal 8000-bfff
5	c000-ffff	Page 2	Normal c000-ffff
6	-	Page 3	Shadow 8000-bfff
7	-	Page 4	Shadow c000-ffff
8	4000-7fff	Page 5	4000-7fff
9	-	Page 6	-
10	-	Page 7	-

Page	In 48K mode	In 128K mode	In SamRam mode
11	Multiface ROM	Multiface ROM	-

In 48K mode, pages 4, 5, and 8 are saved. In SamRam mode, pages 4 to 8 are saved. In 128K mode, all pages from 3 to 10 are saved. Pentagon snapshots are very similar to 128K snapshots, while Scorpion snapshots save all 16 pages of RAM in pages 3 to 18. There is no end marker.

Cases Where ROM is Included

In 48K mode, although not common, it is possible to save page 0 (48K ROM).

References

- World of Spectrum. Formats Reference. Retrieved from https://worldofspectrum.org/faq/reference/formats.htm.
- Pedro Gimeno's Spectrum Emulator Documentation.
- Lunter, G. (1988). Z80 Emulator Documentation.