A ZX Spectrum game by Massimiliano Arca, entirely coded in Boriel ZX Basic for the ByteManiacos Concurso Basic 2023.

**What ParaZXland is and what it is not**

ParaZXland is a computer game for the ZX Spectrum 48k and its clones. It is the ideal evolution of PARA2023, a small game written entirely in authentic ZX Spectrum BASIC, submitted to the 10-liner context, where it ranked in the 17th position, being the second ZX Spectrum game in the "PUR-80" category.

ParaZXland does not share anything with PARA2023 though, as it was developed in Boriel ZX Basic and not in the original ZX Spectrum BASIC. While PARA2023 was contained in 10 lines, ParaZXland consists of almost 2000 lines of code. Its development was carried out on the Visual Studio Code framework. In order to participate to the ByteManiacos Concurso Basic 2023, no ASM content has been used in any form ("*Sólo se admite usar la última versión del compilador para el caso de ZX Basic (Boriel). Versión estable 1.16.4. No se puede usar ASM, bajo ningún concepto, sólo las ventajas que el compilador permita*"). The inclusion of any supporting ZXB routines that implicitly use ASM has also been avoided. The programme is therefore pure Boriel ZX Basic. For compatibility reasons, the sound is very simple (achieved with the BEEP and OUT keywords), so that it can be reproduced in full on an un-expanded ZX Spectrum 48k. ParaZXland can be directly compiled from the command line:

```
zxbc -t -B -O 2 -a --org 28000 -o ParaZXland.tap ParaZXland.bas
```

Finally, for the sake of completeness, an original intro screen created with ZXPaintBrush has been added to the TAP. The intro screen is also provided as a JPG file. The TZX file can be converted into a real cassette version (available from the author): an artwork for the cassette cover is also provided in JPG format for those who want to make their own.

The content of the submission is therefore:

| | |
|---|---|
| pZXl.tap | (the game in TAP format, with loader and intro screen) |
| pZXl.z80 | (the game in Z80 format, with intro screen) |
| pZXl.tzx | (the game in TZX format, with loader and intro screen) |
| ParaZXland.bas | (the self-consistent Boriel ZX Basic code) |
| ParaZXland.tap | (the game as generated by ZXBC, without loader) |
| ParaZXland.scr | (intro screen in SCR format) |
| ParaZXland.jpg | (intro screen in JPG format) |
| ParaZXland_cover.jpg | (cassette cover) |
| ParaZXland.pdf | (this text as formatted PDF file) |
| pZXl.txt | (this text as plain unformatted TXT file) |

To put the game on an SD card, to run it on ZX Spectrum Next and clones, ZX-UNO and derivatives, or directly on a ZX Spectrum equipped with an SD card reader, only the pZXl.tap, pZXl.z80, and pZXl.txt files are needed.

**The Story**

In an unlikely mission, a volunteer parachutist has been chosen to save the world by dropping from an unstable helicopter onto a tiny island where the ultra-secret codes to save the world from destruction have been hidden. The island is protected by flying anti-grav bombs, which would kill our hero on contact. Strong oceanic winds also make the descent difficult.

Our hero will have to take off from the helicopter, avoid explosions, contrast the wind, and finally land on the tiny island.

In the unlikely event of success, the mission will be repeated, but with increased difficulty, more scaring types of bombs and – of course – more points! Thankfully, extra points are also distributed in the sky under the form of red hearts and additional lives are available on increasing the score:

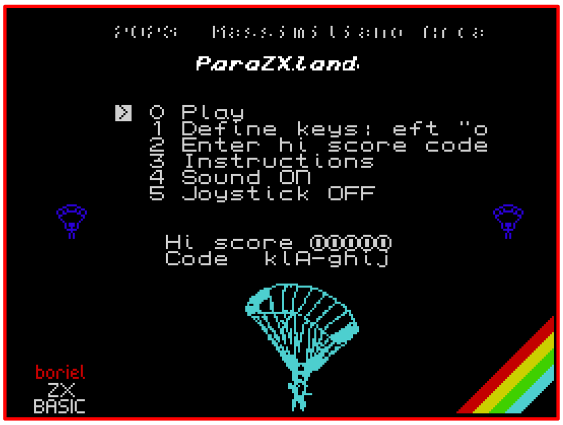
| Score | Event | Points |
|---|---|---|
| 0–10000 | Descent increase | +10 |

| | | |
|---|---|---|
| | Heart collision | +15 |
| | Landing | +1000 |
| 10000–29999 | Descent increase | +30 |
| | Heart collision | +50 |
| | Landing | +2000 |
| >30000 | Descent increase | +50 |
| | Heart collision | +50 |
| | Landing | +3000 |

- Extra life at 20000 and 50000 points.
- After a successful landing the number of mines increases by 4.
- Bomb types change at 10000 and 30000 points.

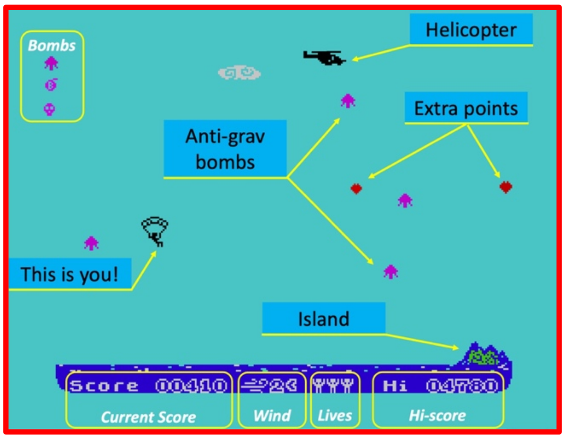
## The game

The opening screen allows to start a new game (option 0), redefine the keys (option 1; by default: "O" for left, "P" for right, "Q" for pause), enter a previous high score (option 2), view the instructions (option 3), turn the sound on and off (option 4), and enable a Kempston joystick (option 5; the combination of up and the fire button serves as a pause key).



The hi-score is associated with an encrypted 8-digit code (in the format ###-####) that can be entered by the user to retrieve a previous game.

During the game, the current score, hi-score, wind speed and direction are displayed on the bottom line of the screen.



## The structure of the programme

Taking advantage of the Boriel ZX Basic features, the program is structured in functions and subroutines. Each of these is extensively commented to improve readability.

In general, a limited use of Boolean operators on the same line (nested IF ... THEN are preferred) has been used in the main game cycle to increase the execution speed.

The graphics are done either by using UDGs (also prepared thanks to the ZX Graphics Editor) or by directly drawing the character point by point (**drawpara** routine for the artwork and **paraman** during the game; this is very fast in Boriel ZX Basic) to allow a pixel-by-pixel positioning of the main character, whose design depends on the game situation. Different fonts are used in the game. The helicopter – entering at the beginning of each game – is right-scrolled by a properly-prepared original 24-byte 2-bit scrolling routine using the bit-shift operation included in Boriel ZX Basic, adapted to work on a 24 bit (4 byte) sequence. The same approach is used for slowly moving the cloud in the background.

As mentioned above, Hi-Score values are associated with an 8-digit code (format ###-####, where # is an alphanumeric character). The conversion is done by the **codify** and **decodify** functions. The former function converts a number into an encrypted string, where characters 1 to 5 are calculated from the original digits of the number; character 6 is a random character used to encrypt the numbers; character 7 is a control code calculated from characters 1 to 5 (implicitly including the encrypting character); character 8 is "-" and finally the encrypted string is inverted to give the final form. The **decodify** function decodes a string to recover the original number. Note that a random character is used as the key to encrypt/decrypt the number, so different codes can be associated with the same score, but a single score is associated with each encrypted string of characters.

To enter the string, a cursor-based input routine (**asktext**) has been prepared which accepts characters between code 33 (!) and 126 (~). The routine limits the number of characters and allows the use of the delete and enter keys to cancel the last character entered or to accept the text.

During the game, the keyboard and joystick (if enabled) are constantly monitored by means of the LastK system variable and the $1F I/O port, respectively. Please note that on an authentic ZX Spectrum 48k, if the Kempston joystick is enabled without a working interface, unexpected keyboard issues can occasionally occur.

Finally, the physical model of the parachute landing. Due to the presence of the parachute, which results in a linear vertical speed, no gravitational effect has been introduced. However, the parachute itself induces a periodic oscillation in the horizontal position of the character, which is obtained with the trigonometric COSine function in dependence of the vertical position. The horizontal position is also affected by the wind (whose strength and direction can be found in the bottom line), which makes the descent even more difficult. The position of the parachutist is indeed determined by the lateral acceleration.

On the hardware side, ParaZXland was successfully tested on authentic vintage ZX Spectrum 48k computers, N-GO (fully compatible with the ZX Spectrum Next), ZX Uno, and a variety of emulators running on Windows, macOS, and even MS-DOS.

## Distribution

This game is completely free and can be unreservedly distributed, provided that it is not modified in any way and that the relevant files are accompanied by this text.

**Acknowledgements**

José Rodriguez, aka *Boriel*, for the immense gift of ZX Basic to the community of ZX Spectrum enthusiasts; David Saphier aka *em00k* for setting up NextBuild; *DuefectuCorp* for the ZX Graphic Editor; Damien Guard aka *damieng* for the exterminate collection of fonts available on his web site.

**Final notes**

The short tunes in the game are copyright-free and royalty-free. They are freely inspired by Beethoven symphony no. 5 and the Cucaracha motif. Finally, the author guarantees that no parachutist has been harmed during the development of this game.