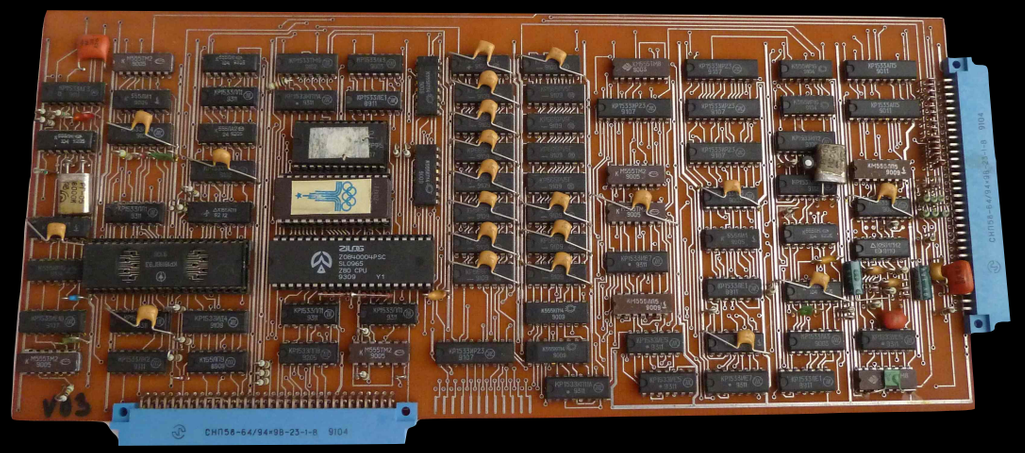
# Starting development for the ZX Spectrum


tg: @errorsoft

# Specifications
# ZX Spectrum (Pentagon 128)

- CPU: z80, 3.5 MHz, 8 bit
- RAM: 128 kb
- ROM: 16(64) kb



- OS: TR-DOS (Disk OS, 640kb floppy disks)
- MUSIC: AY (3 channel)
- GRAPHICS: 256x192 px, 15 colors
- Number of ticker per frame: 71680

**Про такты: http://hypr.ru/blog/773.html**

# Z80


- 8-bit CISC CPU
- Designed in 1976
- 158 instructions
- 16 bit address bus, 64 kb addressable memory

# Memory Map

| | |
|---|---|
| **#0000** | ROM BASIC/BASIC128 |
| **#4000** | Screen (6144 bytes) |
| **#5800** | Attributes (768 bytes) |
| **#5B00** | Printer buffer<br>(256 bytes), if you don't need 128ROM, you can use |
| **#5C00** | BASIC system variables |
| **#5CB6…#5D26** | TR-DOS system variables |
| **#5D3B/#5CCB** | Start of BASIC program (depending on whether TR-DOS is initialized or not) |
| **#5E3C** | Beginning of BASIC program when working with TR-DOS with disk (TR-DOS uses 257 bytes for the sector buffer) |
| **…** | **Bottom of the stack (the stack grows to meet the BASIC** |
| **~#6100…#FFFF** | *****Your Program*****<br>It is better not to touch the memory before this address if the operability of TR-DOS and BASIC is important to you |

**Lower memory**

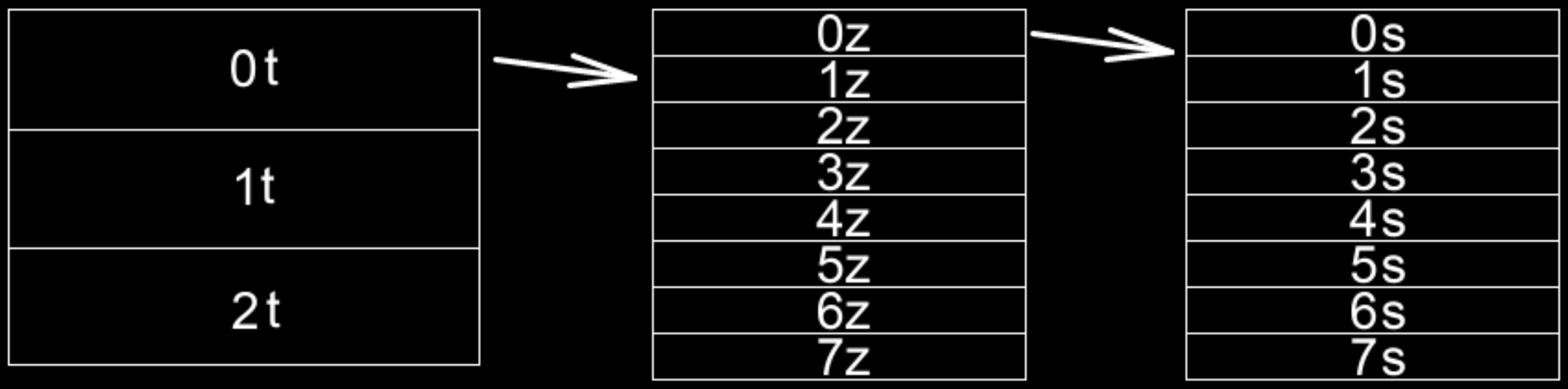
**Upper memory**

**To keep BASIC operational, the IY register must be # 5C3A**

# Screen structure

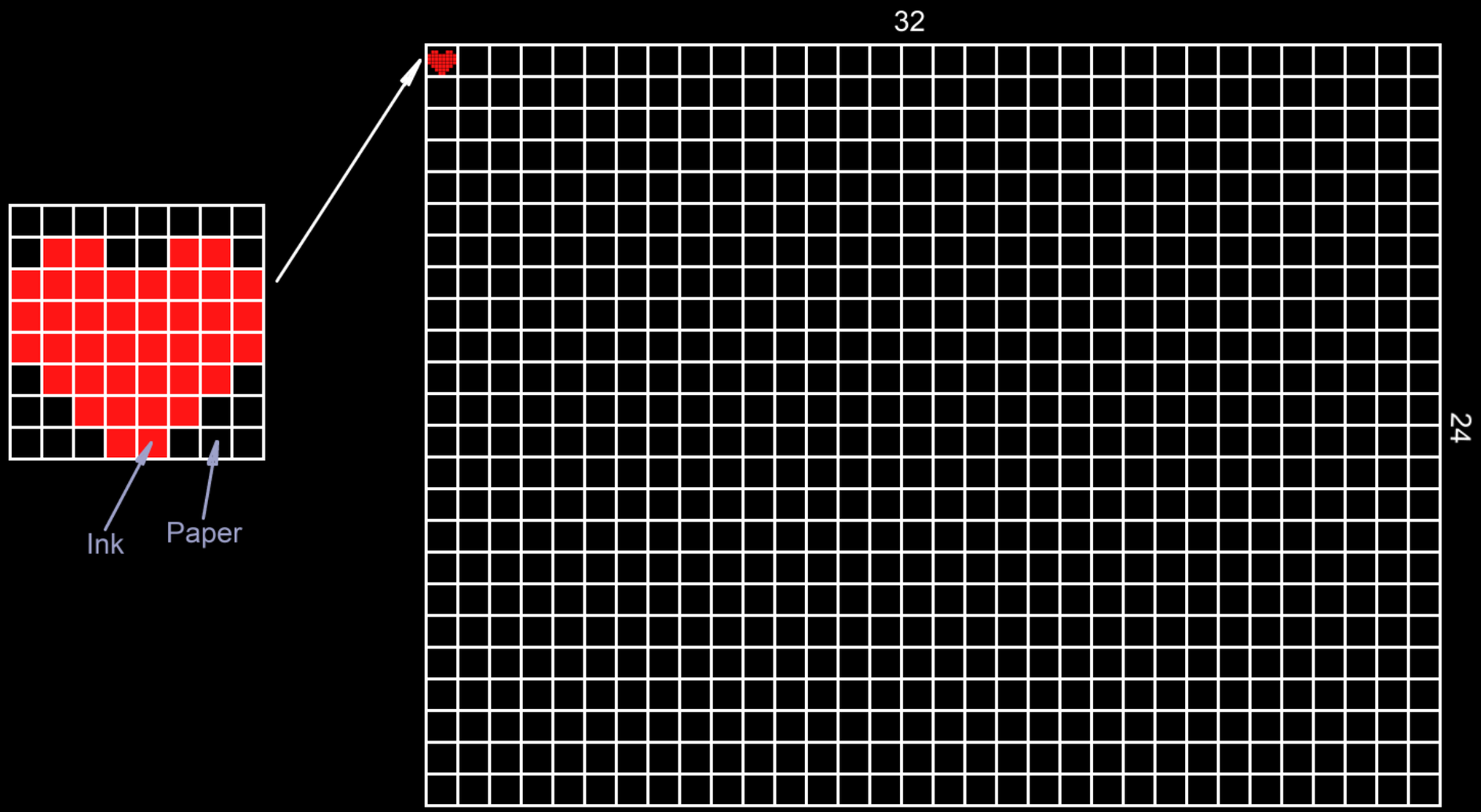
```
 +------H------+ +------L------+
-+-------------+-+-------------+-
|0|1|0|t|t|s|s|s|z|z|z|x|x|x|x|x|
-------+-+-+---+-+---+-+-------+-
       +-+ +---+ +---+ +-------+
        1    2     3       4
```

1(t)  Third number 0..2
2(s)  Offset inside familiarity 0..7
3(z)  Row number 0..7
4(x)  Column number

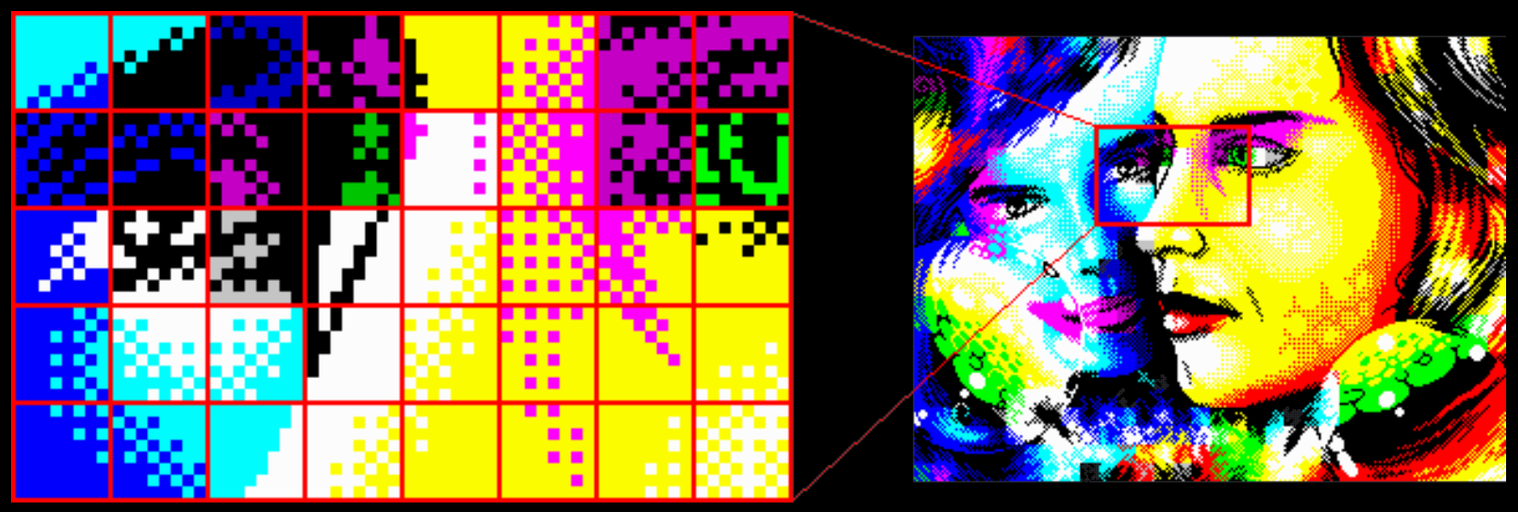




https://zxpress.ru/article.php?id=18217  http://hype.retroscene.org/blog/dev/130.html

# Colors




| B0 | B1 | Номер | Название |
|---|---|---|---|
| | | 0 | Black |
| | | 1 | Blue |
| | | 2 | Red |
| | | 3 | Magenta |
| | | 4 | Green |
| | | 5 | Cyan |
| | | 6 | Yellow |
| | | 7 | White |

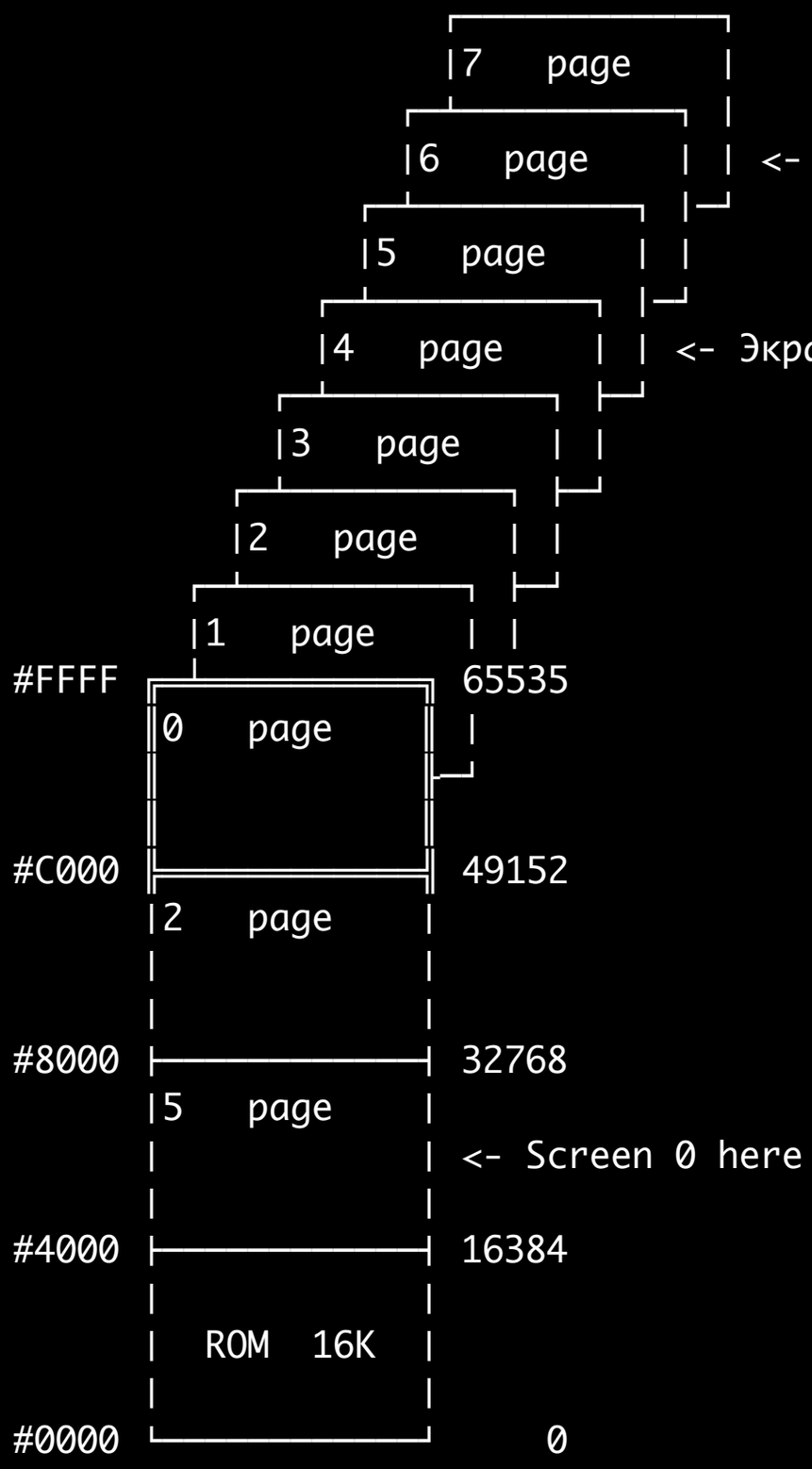


**D0..D2** - ink color (INK)
**D3..D5** - paper color (PAPER)
**D6** - luminance bit (BRIGHT)
**D7** - flashing bit (FLASH)

# Port #FE

| Reading | Writing |
| --- | --- |
| **D0-D4** - display the state of a certain half-row of the ZX Spectrum keyboard. Half-row ports - #7FFE, #BFFE, #DFFE, #EFFE, #F7FE, #FBFE, #FDFE и #FEFE.<br><br>**D6** - displays the status of the tape input.<br><br>**D5, D7** - unused | **D0-D2** - determine the color of the border.<br><br>**D3** - controls the state of the recording output of the MIC tape recorder.<br><br>**D4** - controls the internal speaker (beeper).<br><br>**D5-D7** - unused. |

# Switching RAM pages / screens




**#7FFD - port that controls the display of pages**

**D0-D2** - page number of RAM connected to the upper 16 KB of memory (from address # C000)

**D3** - select the displayed video page.
0 - page in bank 5, 1 - in bank 7.**(default 0)**

**D4** - ROM page number.
0 - BASIC128, 1 - BASIC48. **(commonly used 1)**

**D5** - extended memory inhibit (48K latch).
**(we need 0)**
When the bit is set, extended memory management will not be possible until the computer is reset.

**Never use port #FD!**

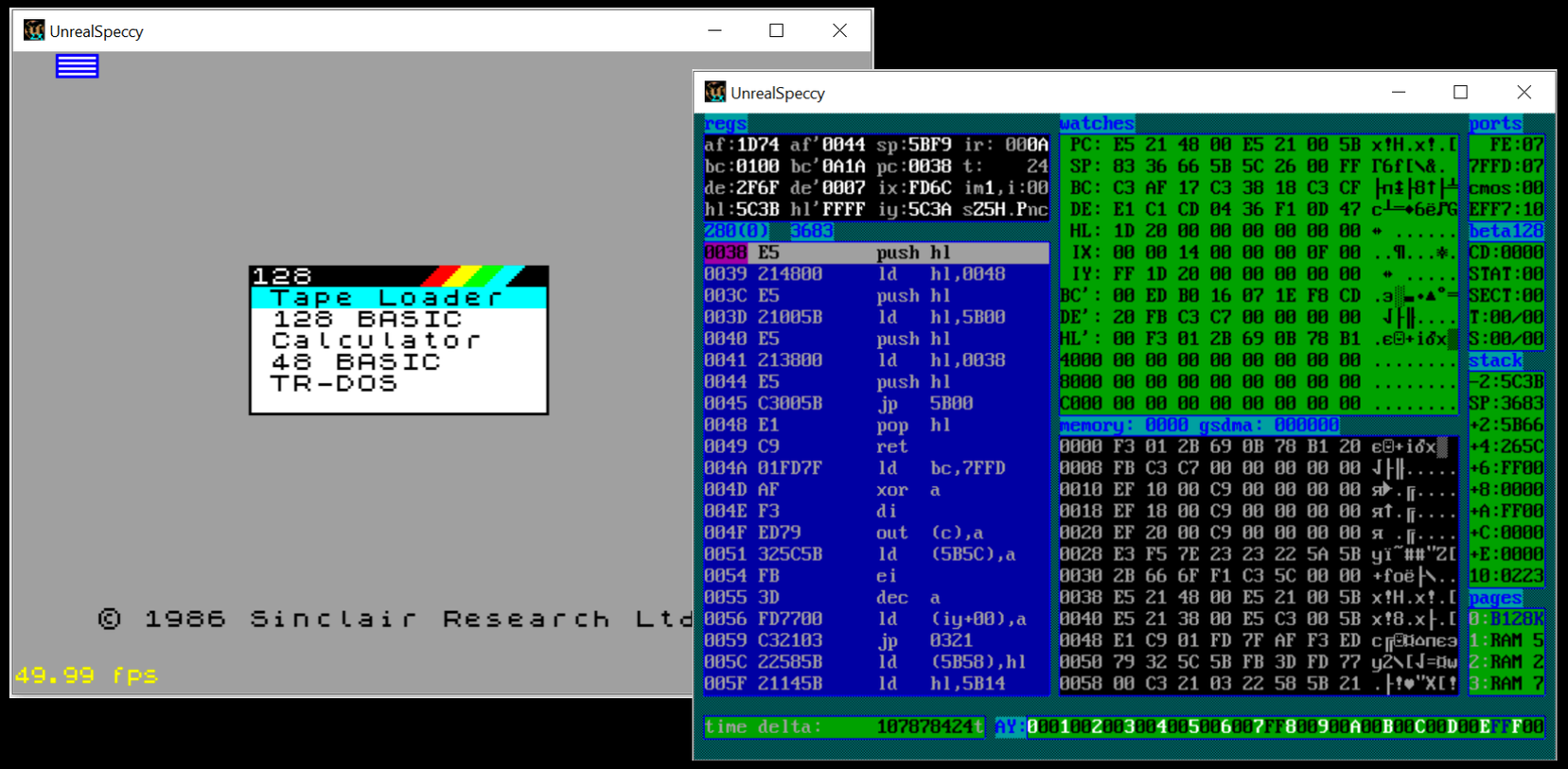
# AY


- #FFFD - register address AY-3-8910
- #BFFD - data register AY-3-8910

# Useful software and development tools

# Unreal Speccy

## Spectrum emulator (Pentagon)




Running: unreal <file_name>

The emulator is able to display custom labels from the user.l file when pressing Ctrl + L. (Try Ctrl + J)

The user.l file must be in the folder with the emulator

http://dlcorp.nedopc.com/viewforum.php?f=8

# sjAsmPlus

## Cross assembler z80

```
SjASMPlus Z80 Cross-Assembler v1.14.3 (https://github.com/z00m128/sjasmplus)
Pass 1 complete (0 errors)
Pass 2 complete (0 errors)
> code size: 14
Pass 3 complete
Errors: 0, warnings: 0, compiled: 35 lines, work time: 0.031 seconds
```

Running: sjasmplus <asm_file>.asm --nofakes

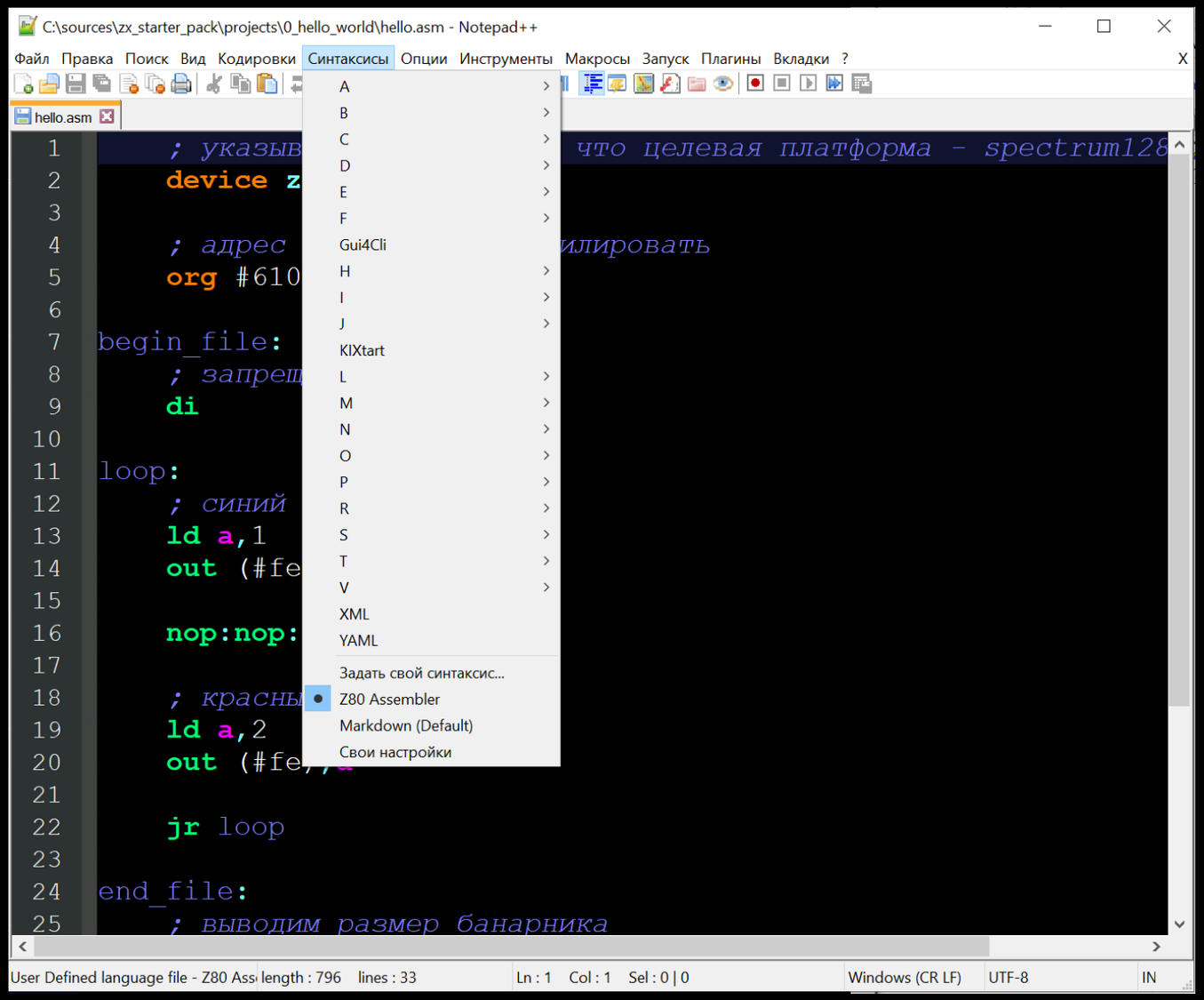
*Parameter --nofakes prohibits fake instructions such as LD HL,DE => LD H,D: LD L,E*

***I highly recommend not using fake instructions!***

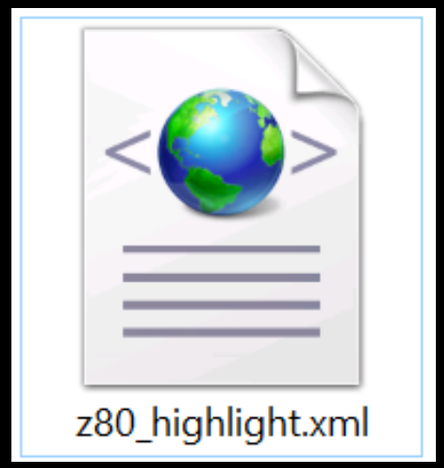
https://github.com/z00m128/sjasmplus

# Notepad++

## Editor



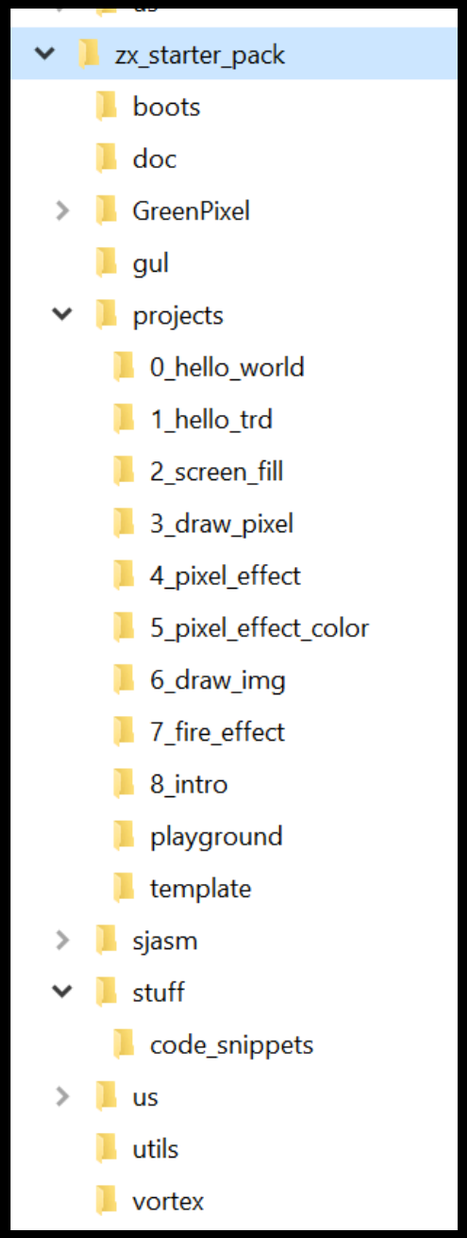

+

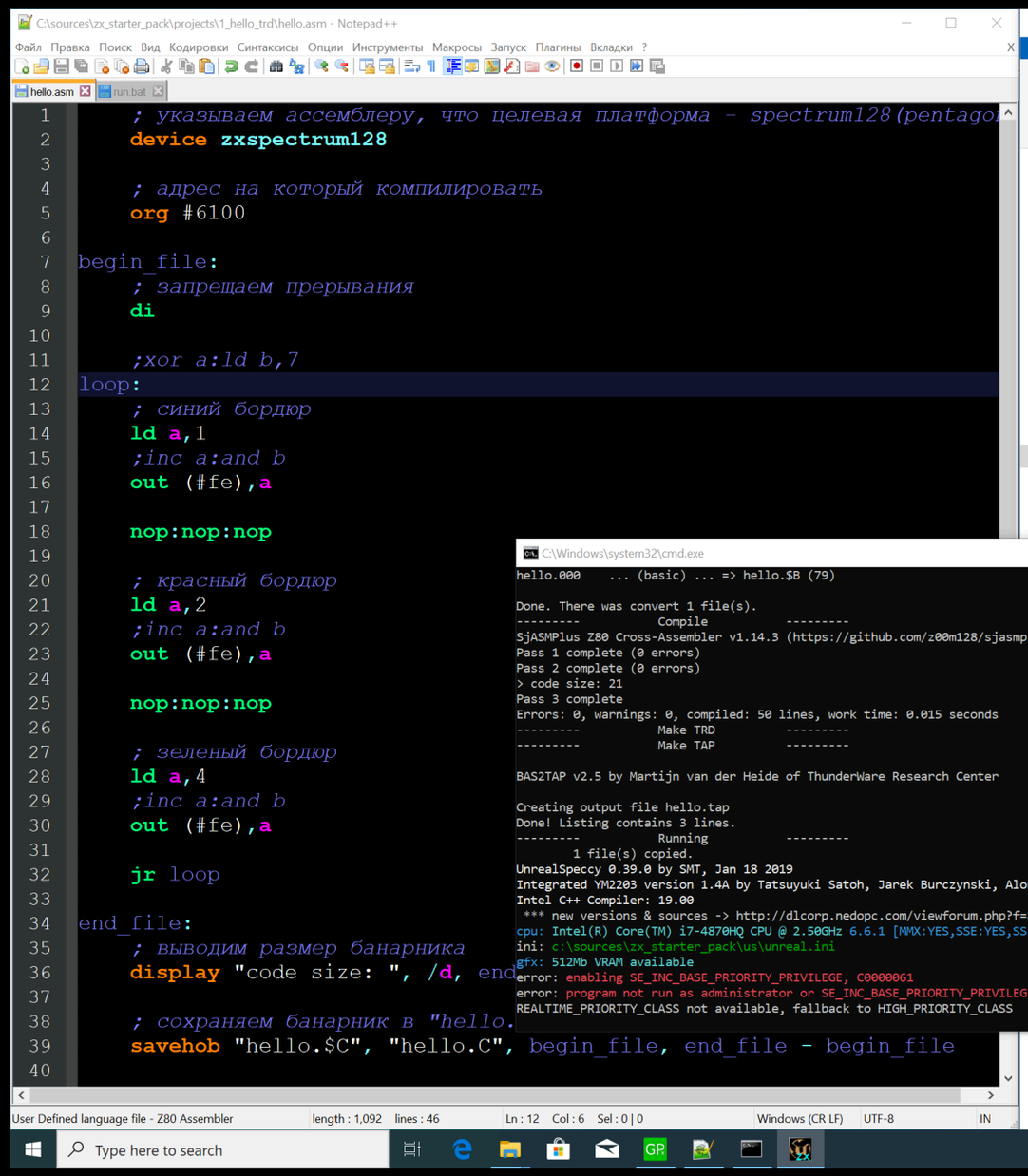


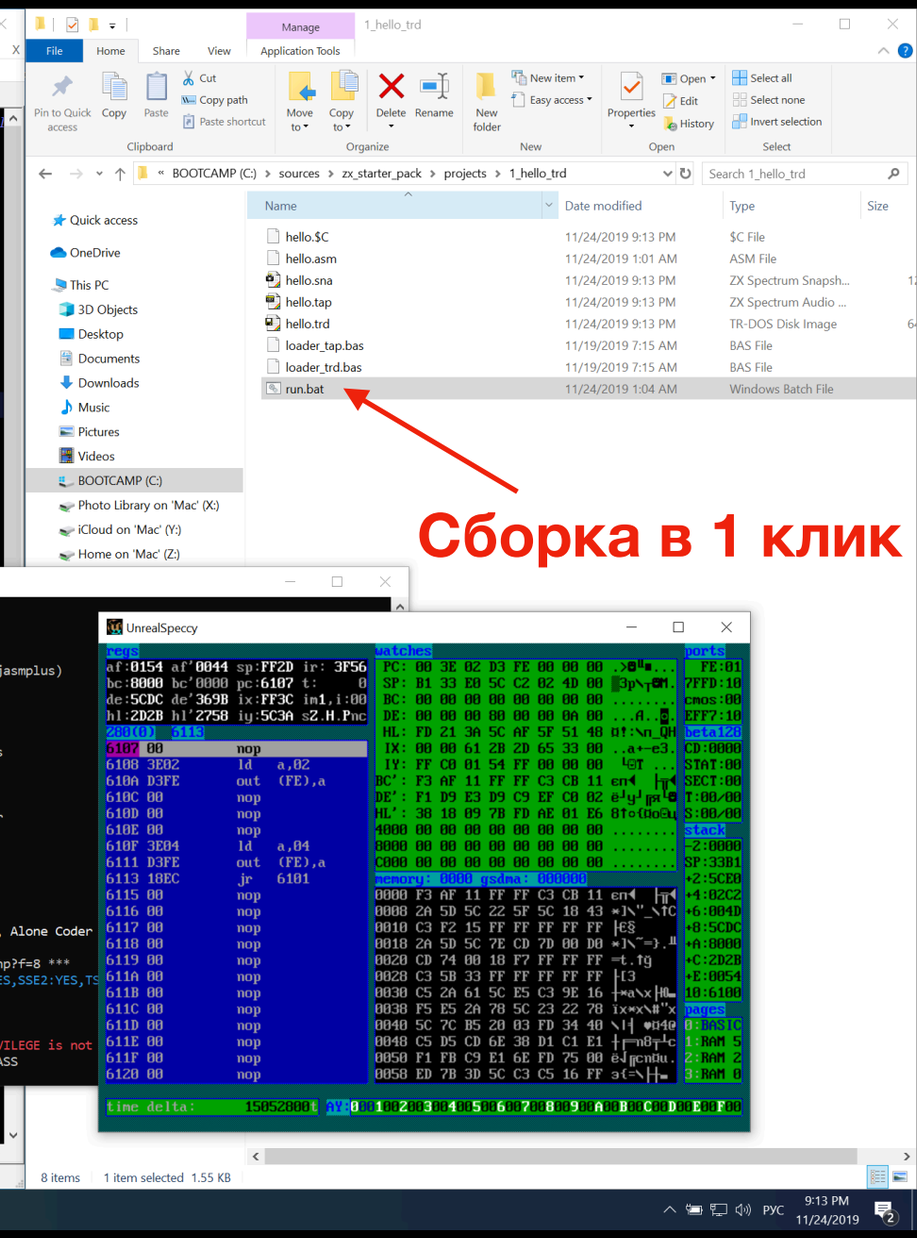

**Z80 asm custom highlighting**

https://notepad-plus-plus.org

# Working environment










Сборка в 1 клик

# Simple script to build an sna file

```
@echo off

rem Params
set name=hello
```

Project name

```
echo ---------              Compile             ---------
..\..\sjasm\sjasmplus hello.asm --nofakes

echo ---------              Running             ---------
rem Copy labels to emulator
copy "user.l" "..\..\us\"
del user.l

..\..\us\unreal %name%.sna
```

Compile necessary files

We copy the labels, for easy debugging

Launching the emulator, with execution created sjAsm sna file

# Simple project template

```
    ; tell the assembler target platform is -spectrum128 (pentagon)
    device zxspectrum128

    ; the address to compile to
    org #6100

begin_file:
    ; YOUR CODE
    di:halt

end_file:
    ; displaying the size of the banner
    display "code size: ", /d, end_file - begin_file

    ; save sna (state snapshot) file
    savesna "hello.sna", begin_file

    ; save labels
    labelslist "user.l"
```

**These two commands stop the CPU**
**di - disable interrupts**
**halt - wait for interrupt**

**We save the sna file,**
**to run in the emulator**

**Save labels for easy debugging**

# Registers Z80

| **Main registers** | | **Alternative registers** | |
|---|---|---|---|
| A<br>Accumulator | F<br>Flags | A’<br>Accumulator | F’<br>Flags |
| B | C | B’ | C’ |
| D | E | D’ | E’ |
| H | L | H’ | L’ |

BC, DE, HL - register pairs

| **Special registers** | |
|---|---|
| IX (16 bit) | Index register |
| IY (16 bit) | Index register |
| SP (16 bit) | Stack pointer |
| PC (16 bit) | Program counter |
| I | Interrupt vector |
| R | DRAM refresh counter |

IXL, IXH / IYL, IYH  - halves of index registers

# Registers / Memory Access

**EXX** - exchange BC,DE,HL and BC`,DE`,HL`

**EX AF, AF** - exchange AF and AF`

**EX DE, HL** - exchange register pairs DE and HL

**LD R1, R2** - load register value R2 to register R1

**LD R1, N** - load value N to register R1

**LD RR, NN** - load value NN load value into register pair RR

**LD (RR), R1** - load into memory, at the address in the register pair RR, register value R1

**LD R1, (RR)** - load into register R1 the value from memory, at the address in the register pair RR

**LD R,(NN)** - load into register A / RR a value from memory, at address NNNN

**LD (NN),R** - load into memory, at NNNN address, register value A/RR

# Logical / arithmetic operations

**OR R1** - OR operation between A and R1

**AND R1** - AND operation between A and R1

**XOR R1** - XOR operation between A and R1

**OR N/AND N/XOR N** - logical operation between A and immediate value N

**CPL** - invert A

**ADD R1** - add R1 and A

**SUB R1** - subtract R1 from A

**ADD N/SUB N** - add / subtract the immediate value of N

**CP N/R1** - compare A with value N / register R1 (SUB, without saving the result)

**INC R1** - increase register R1 by 1

**DEC R1** - decrease register R1 by 1

# Shifts

**SRL reg - 0 >>**

**SRA reg - 0/1 >> (arithmetic)**

**RR reg - C >>**

**SLA reg - << 0**

**SLI reg - << 1**

**RL reg - << C**

**Optimization:**

SLA A (is replaced by ADD A, A )

SLA L:RL H (is replaced by ADD HL, HL )

RL L:RL H (to be replaced by ADC HL, HL)

**RLCA reg - [<<] - cyclic shift**

**RRCA reg - [>>] - cyclic shift**

WORK FASTER THAN OTHER SHIFTS:

RLA - << C

**RRA - C >>**

# Jump

**JP NN** - jump to address NN

**JP <NZ,NC,PO,P,Z,PE,C,M>, NN** - jump to address NN, depending on the condition

**JR N** - relative jumps +127…-128

**JR <NZ,NC,Z,C>, N** - relative jump + 127 ... -128, depending on the condition

**DJNZ N** - loop, decrements B, if B is not 0, jump relative to N

| Condition | Flag | Description |
|---|---|---|
| **Z / NZ(не)** | **Z** | **The result of the operation is 0** |
| **C / NC(не)** | **C** | **A carry has occurred (e.g. 120-130)** |
| P / M | S | The result of the operation is positive / negative (7th bit) |
| PO / PE | P/V | Parity (number of bits!) / overflow (sign changed) |

# Subroutines / interrupts

**CALL NN** - subroutine call at address NN

**RET** - return

**CALL <NZ,NC,PO,P,Z,PE,C,M>, NN** - conditional subroutine call at address NN

**RET <NZ,NC,PO,P,Z,PE,C,M>** - conditional return

**DI** - disable interrupts

**EI** - enable interrupts

**HALT** - wait for interruption => **DI:HALT** - leads to hang

# I / O ports

**IN A,(N)** - input value from port N to register A

**OUT (N), A** - pin A to port N

**OUT (C), A** - output to the port, in the BC register of the value A.

**Example:**

```
LD A, #10
LD BC, #7FFD
OUT (C), A
```

**Always use port # 7FFD, never use port #FD!**

# Release creation

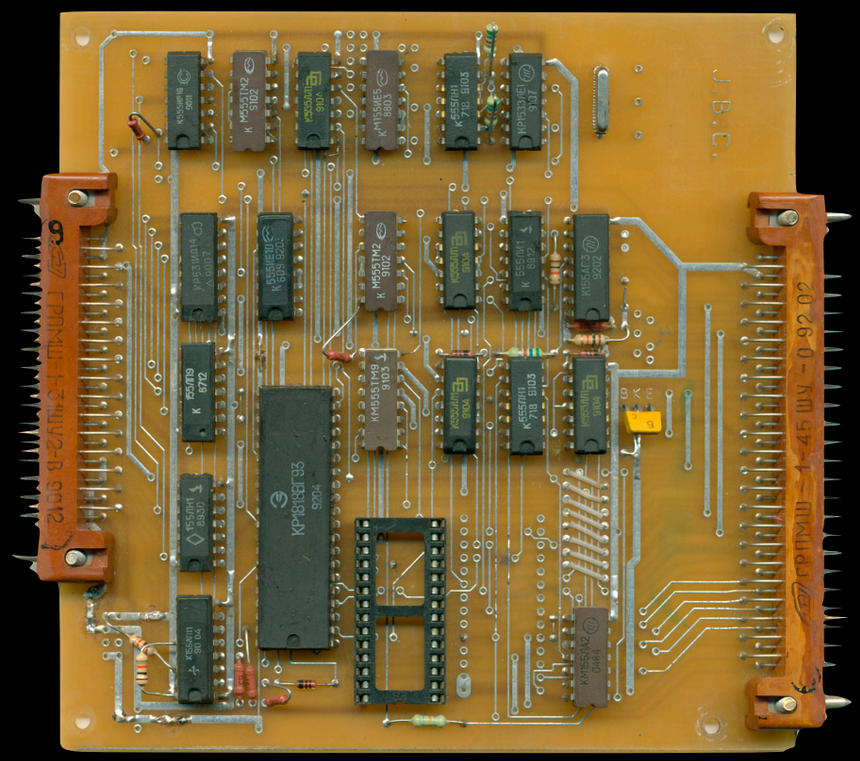

# Beta 128 Disk Interface


By Technology Research Ltd

Start of production 1984/1987

Support for 640kb floppy disks

It was widespread in the USSR and the CIS countries, and is extremely unknown in England.



(c) Batareikin

# tr-dos

**Start to tr-dos:**
**RANDOMIZE(T) USR(Ex+L) 15616**


LIST(K) - list files

RUN(R) - launching BOOT-a

RUN "(Ss+P)name"(Ss+P) - running a BASIC file named «name»

RUN "(Ss+P)name"(Ss+P) CODE(Ex+I) - running a BIN file named «name»

RUN "(Ss+P)name"(Ss+P) CODE(Ex+I) 24832 - running a BIN file named «name» and address

# bas2tap

Utility to convert text file to tap file with basic

```
Command: bas2tap -s<…> -a<…> -c in_file.bas
                 out_file.tap
```

**Parameters:**

-s - File name (displayed during upload)

-a - Start line

-c - Ignore case

https://github.com/andybalaam/bas2tap

# Convert BASIC file, tap file, Hobeta format

```
tapto0 -f in_name.tap
0tohob <filename in tap>.000
```

**Hobet file** - 1 file from image .trd (floppy disks)

**File formats:**

*.$C - Code block

*.$B - Basic program

**https://sourceforge.net/projects/zxspectrumutils/**

# trdtool

## Floppy disk image utility (trd)

**Creating an empty trd file:**

**trdtool # <trd_name>.trd**

**Adding hobeta file to trd:**

`trdtool + <trd_name>.trd <hobeta_name>.$?`

http://shiru.untergrund.net/software.shtml

# taptool

## Utility for working with tape images

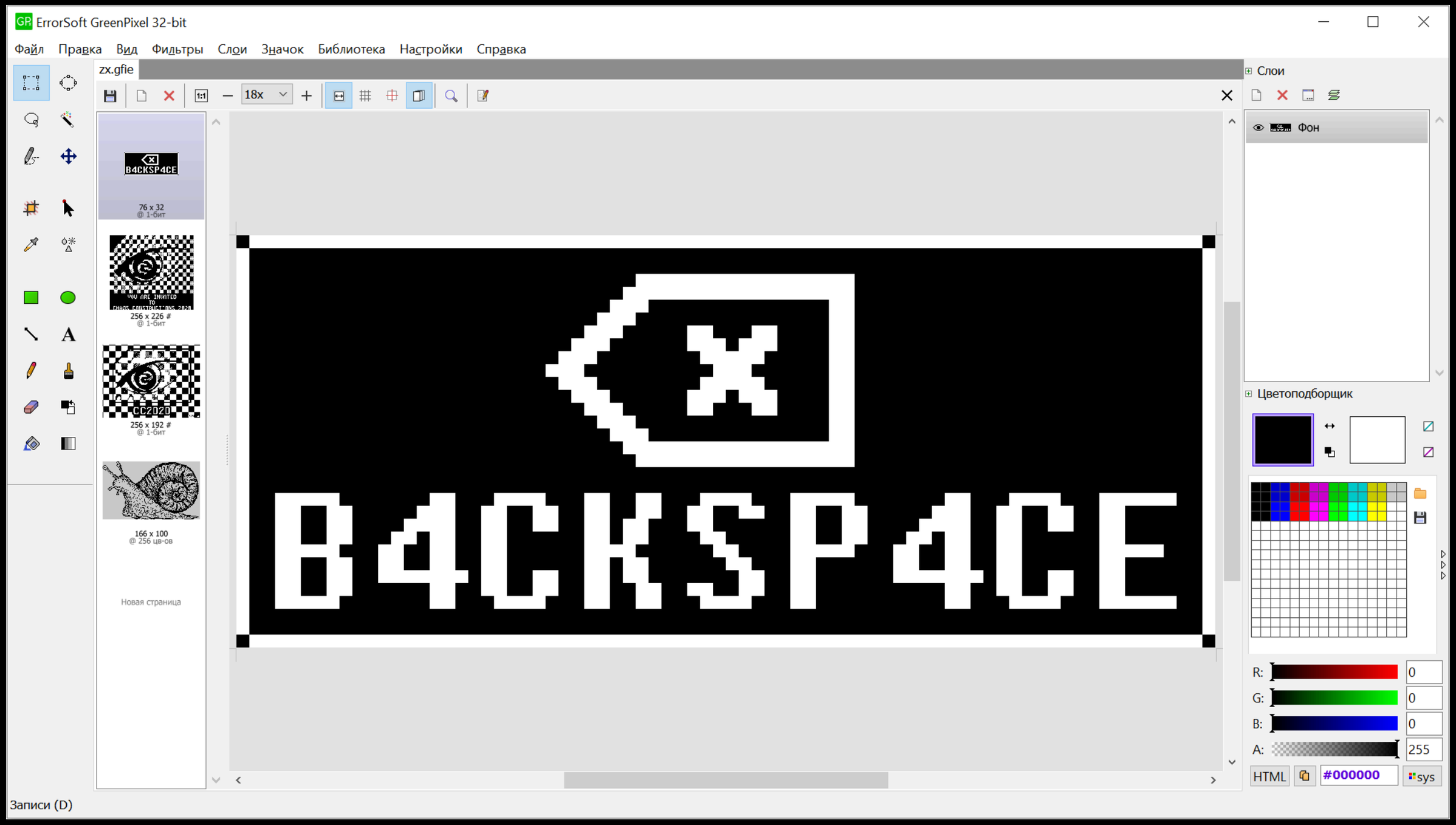
**Adding hobeta to tape image:**

```
taptool +$ <tap_name>.tap <hobeta_name>.$?
```

https://zx-pk.ru/threads/19326-taptool-utilita-dlya-dobavleniya-blokov-v-tap-fajl.html

# GreenPixel

## Pixel art editor




https://github.com/errorcalc/GreenPixel

# gul

## Utility to convert png file to bin file

```
Running: gul -src <png_name>.png -dst <bin_name>.bin -bg 0,0,0
```

**Parameters:**

< -src Source PNG/BMP image > - in image

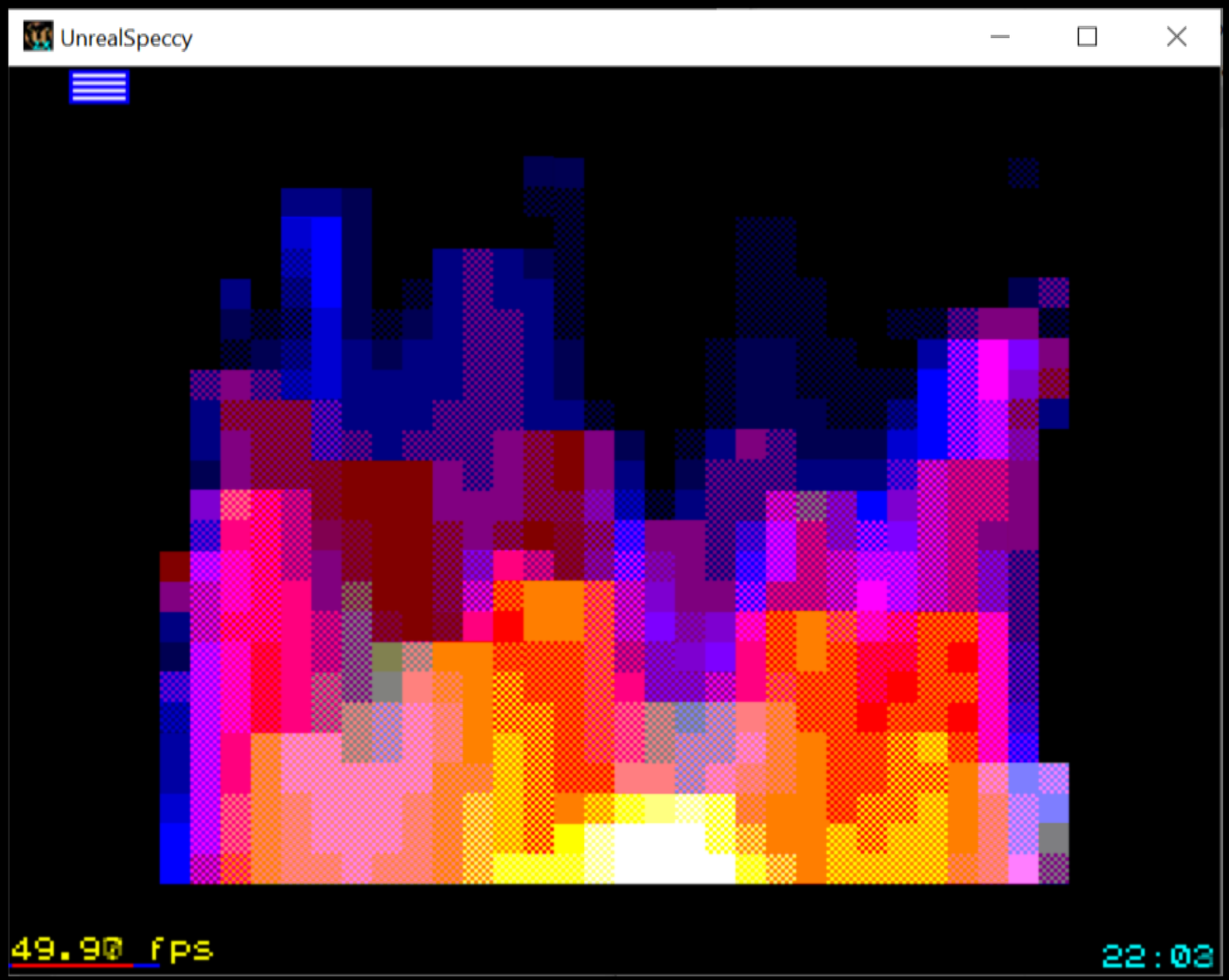
< -dst Destination BIN file > - out bin

[ -bg RED,GREEN,BLUE ] - background color (0,0,0 by default), other colors are foreground color

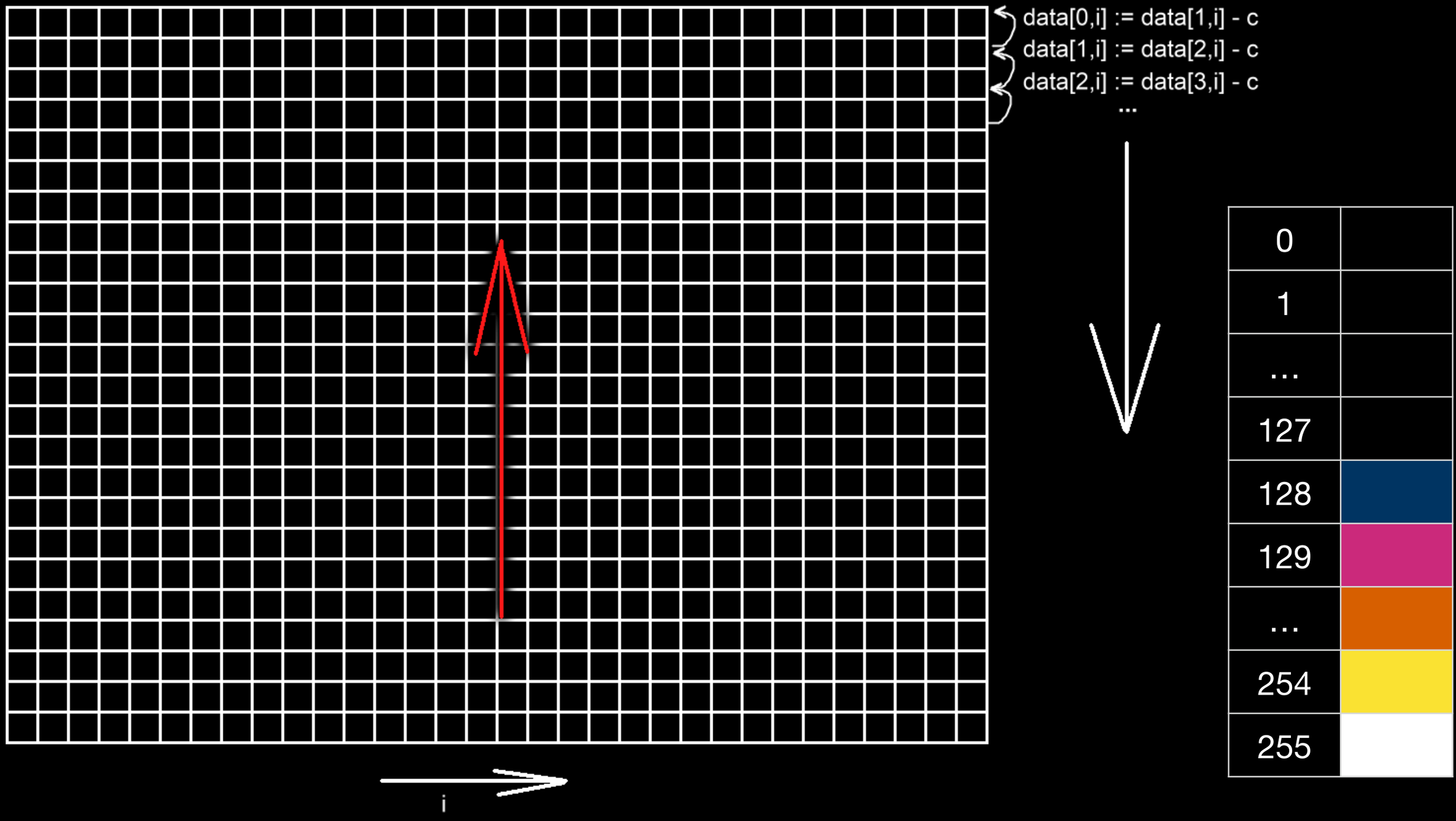
[ -dev 0...255 ] - What deviation from bg can be determined as the background color (20 by default)

**https://github.com/errorcalc/gul**

# Fire effect

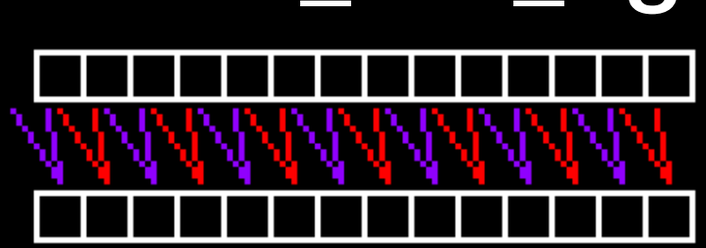


# calc_fire


data[0,i] := data[1,i] - c
data[1,i] := data[2,i] - c
data[2,i] := data[3,i] - c
...
i
0
1
...
127
128
129
...
254
255

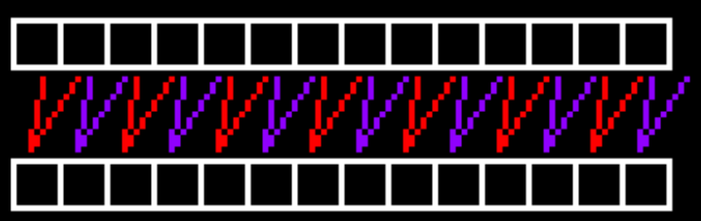
# smoth_line_left/smoth_line_right

## smoth_line_right




[de] := [hl] / 2 + [hl-1] / 2

## smoth_line_left




[de] := [hl] / 2 + [hl+1] / 2

# im2

I=#9E

#9E | #XX

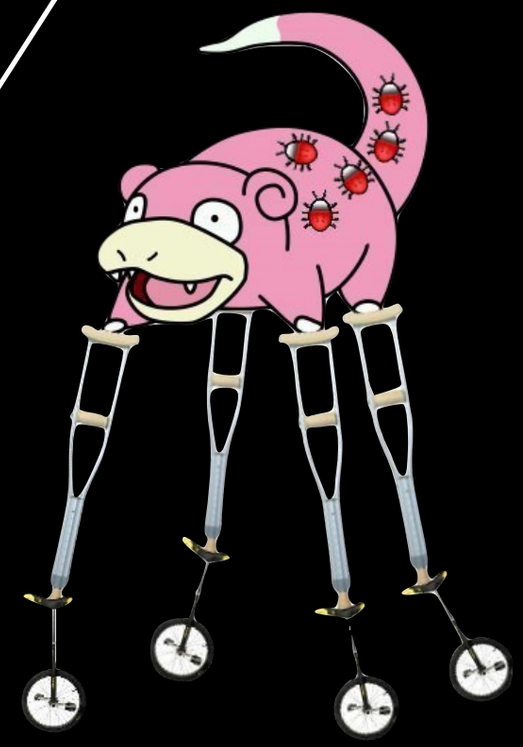
Data Bus Garbage

**Im2 table**

| | |
|---|---|
| #9E00 | #87 |
| #9E01 | #87 |
| #9E02 | #87 |
| #9E03 | #87 |
| #9E04 | #87 |
| #9E05 | #87 |
| #9E06 | #87 |
| … | … |
| #9EFF | #87 |
| #9F00 | #87 |

**Handler address**

#8787



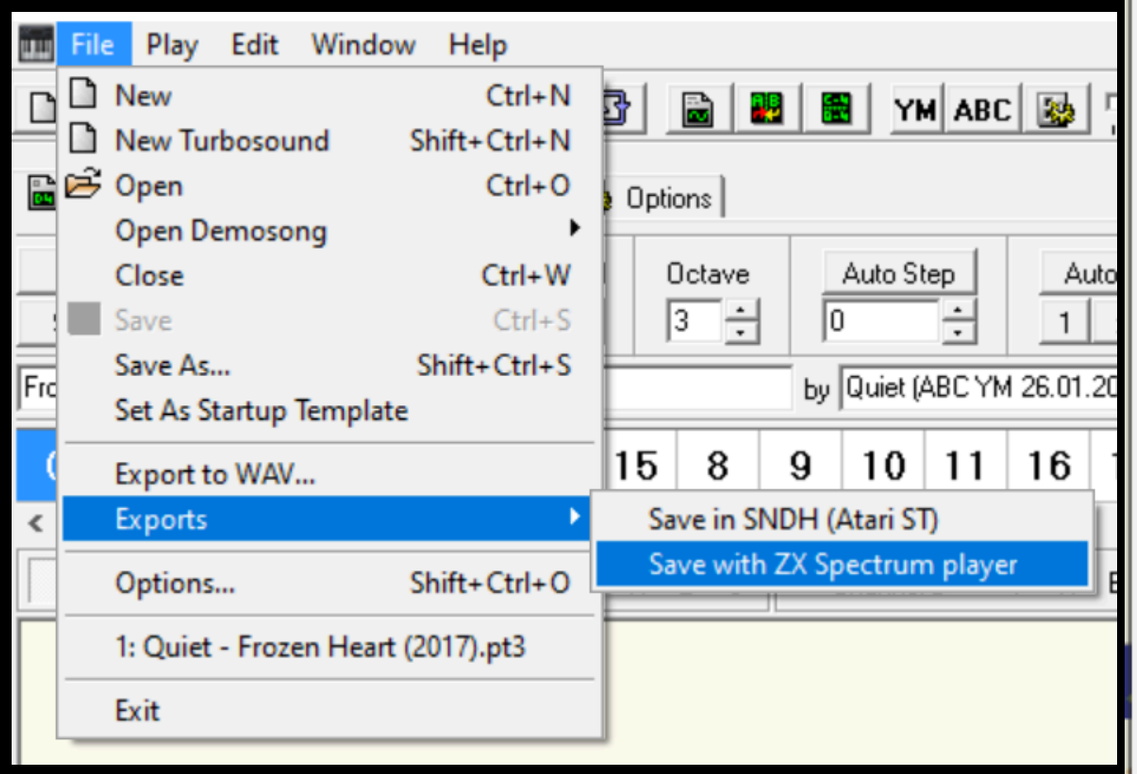
# Music

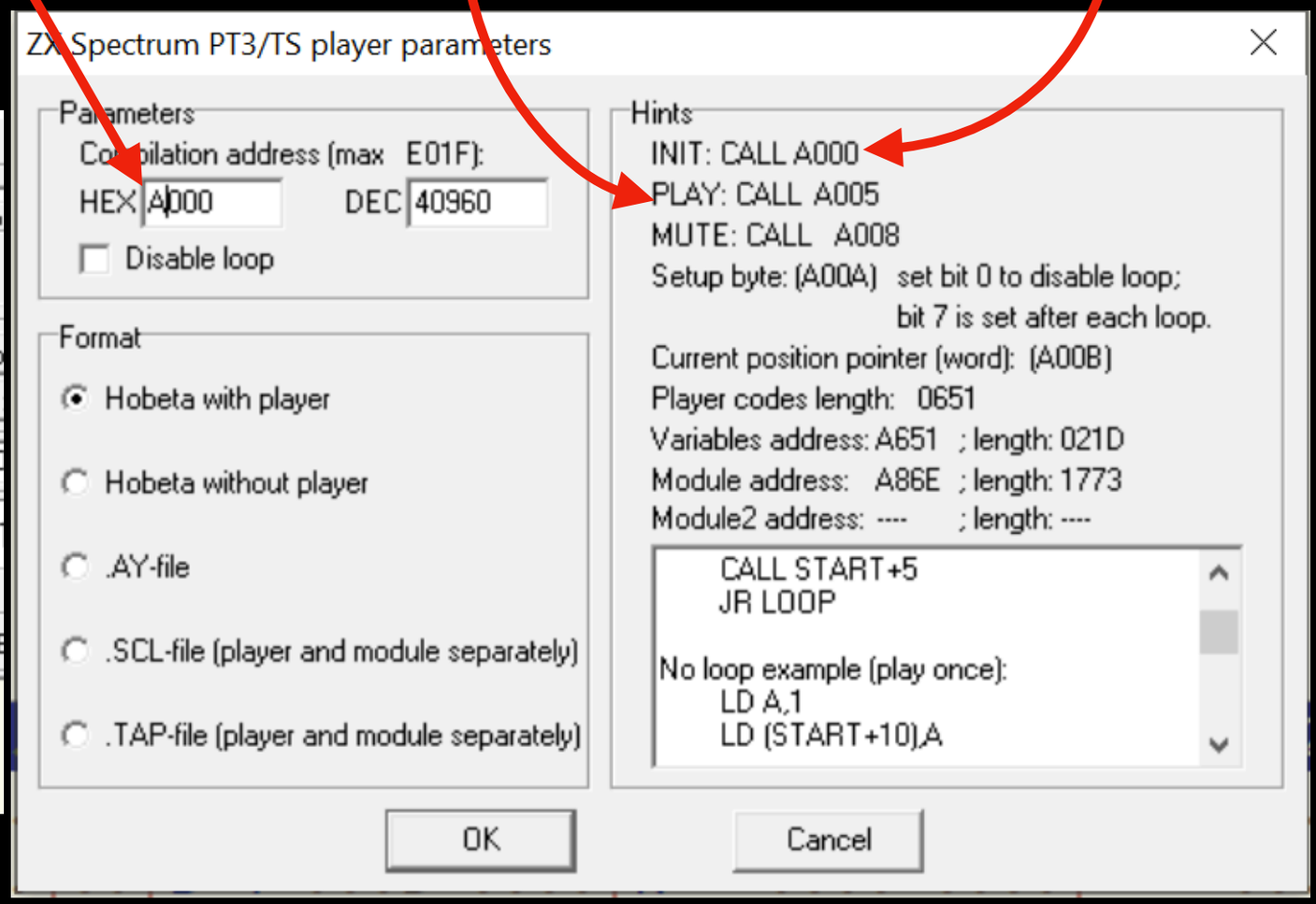
```
    call #A000; init
    ei

loop:
    halt
    call #A005; играем
    jp loop

    org #A000
    incbob "Quiet.$c"
```

Which address compile player

Call for playing

Call for Initialization

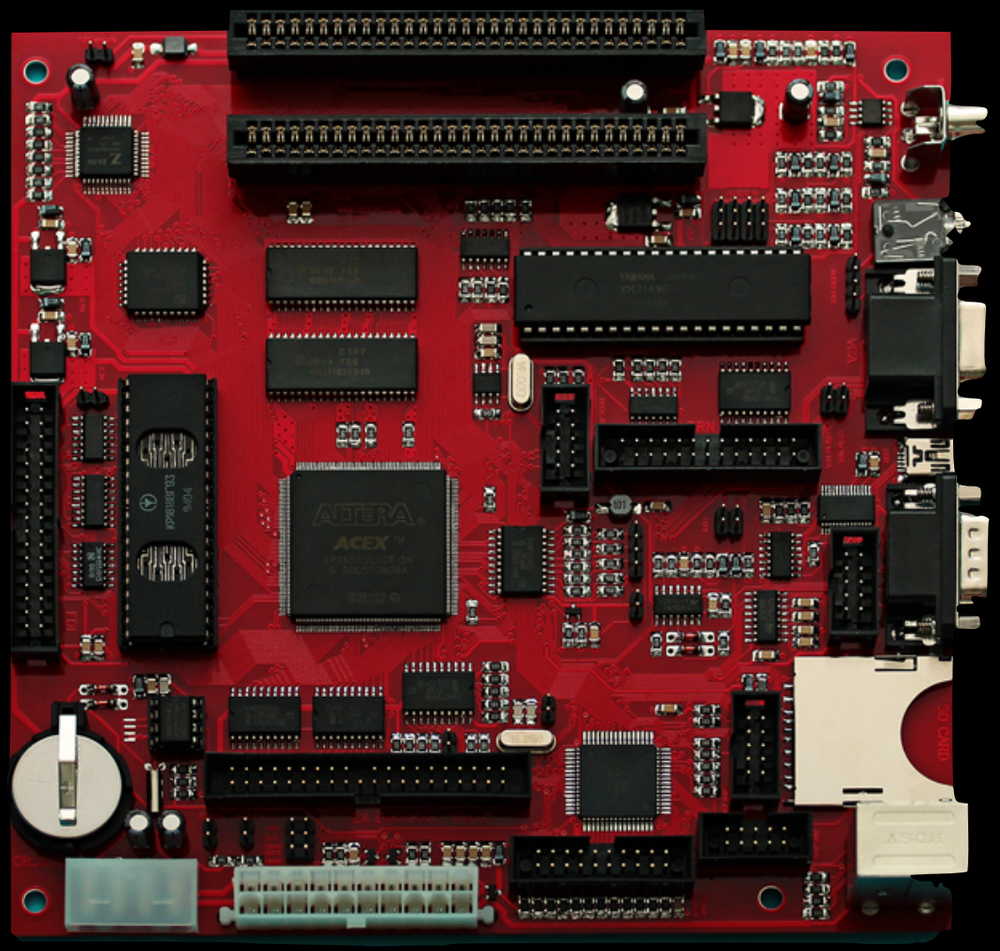


# LICENSE

**"BEER DEMOSCENE LICENSE v1"**
***** error, github.com/errorcalc *****
**You can do whatever you want with this stuff.**
**Any liability and risks lie with you.**
**If we meet some day, and you think this**
**stuff is worth it, you can buy us a beer in return.**
**The author will be happy if you mention author in your prod.**

# Where can I get real?



**ZX Evolution**

**Information:** http://www.nedopc.com/zxevo/zxevo.php

**Buy:** http://tetroid.nedopc.com

# Literature

## Description Z80:

1. "Complete description of the Z80 microprocessor command system" Alexey Asemov (Alex / AT), Advanced Technologies, 2000-2005

2. "CENTRAL PROCESSOR Z80CPU" MINSK UKIK "CENTER" 1991, PDF version by Deny (Denisenko D.A.)2007

**Books are available in the "ZX Spectrum Starter Pack" repository**

# Links

**Interactive Z80 Command Table:**
**http://clrhome.org/table/**

**Math on the Z80:**
http://map.grauw.nl/sources/external/z80bits.html
http://z80-heaven.wikidot.com/math
http://map.grauw.nl/articles/mult_div_shifts.php
http://zxpress.ru/article.php?id=11746

**16-bit shifts on the Z80:**
http://www.chilliant.com/z80shift.html

**Lots of articles about ZX Spectrum programming:**
http://zxpress.ru/tag.php?id=13

**Demos:**
https://github.com/ChaosConstructions

**Archive of books, software, games, demos:**
https://vtrd.in

**Demo archive:**
https://zxaaa.net

# Russian-speaking communities

**Flashback Blog:**
http://hypr.ru

**Great forum about ZX Spectrum:**
https://zx-pk.ru

**ZX Spectrum art:**
https://zxart.ee

**Telegram communities dedicated to coding for ZX Spectrum:**
https://t.me/zxcoding
https://t.me/speccy_code
https://t.me/z80_q11

# Contacts

- tg: @errorsoft
- site: errorsoft.org
- habr: habr.com/users/Error1024/
- vk: vk.com/errorcitizen
- github: github.com/errorcalc

**Downloads/issue/pull req.:** github.com/errorcalc/zx_starter_pack

**Downloads:** github.com/ChaosConstructions/lessons