

Входим в разработку под ZX Spectrum



tg: @errorsoft



Характеристики ZX Spectrum (Pentagon 128)

- CPU: z80, 3.5 МГц, 8 бит
- RAM: 128 кб
- ROM: 16(64) кб
- OS: TR-DOS (Дисковая ОС, дискеты на 640 кб)
- MUSIC: AY (3 канала)
- GRAPHICS: 256x192 px, 15 цветов
- Количество тактов на 1 фрейм: 71680



Про такты: <http://hypr.ru/blog/773.html>

Z80



- 8-битный CISC процессор
- Разработан в 1976 году
- 158 инструкций
- 16 разрядная шина адреса, 64 кб адресуемой памяти



Распределение памяти

#0000	ПЗУ BASIC/BASIC128	Нижняя память
#4000	Экран (6144 байта)	
#5800	Атрибуты (768 байта)	
#5B00	Буфер принтера (256 байт), если не нужен 128ROM, можно использовать	
#5C00	Системные переменные BASIC	
#5CB6...#5D26	Системные переменные TR-DOS	
#5D3B/#5CCB	Начало BASIC программы (в зависимости от того, проинициализирован tr-dos или нет)	
#5E3C	Начало BASIC программы, при работе tr-dos с диском (tr-dos использует 257 байт под буфер сектора)	
...	Дно стека (стек растет на встречу BASIC программе)	Верхняя память
~#6100...#FFFF	***Ваша программа*** До этого адреса память лучше не трогать, если вам важна работоспособность tr-dos и basic	

Для сохранения работоспособности BASIC в регистре IY должно быть #5C3A

Структура экрана

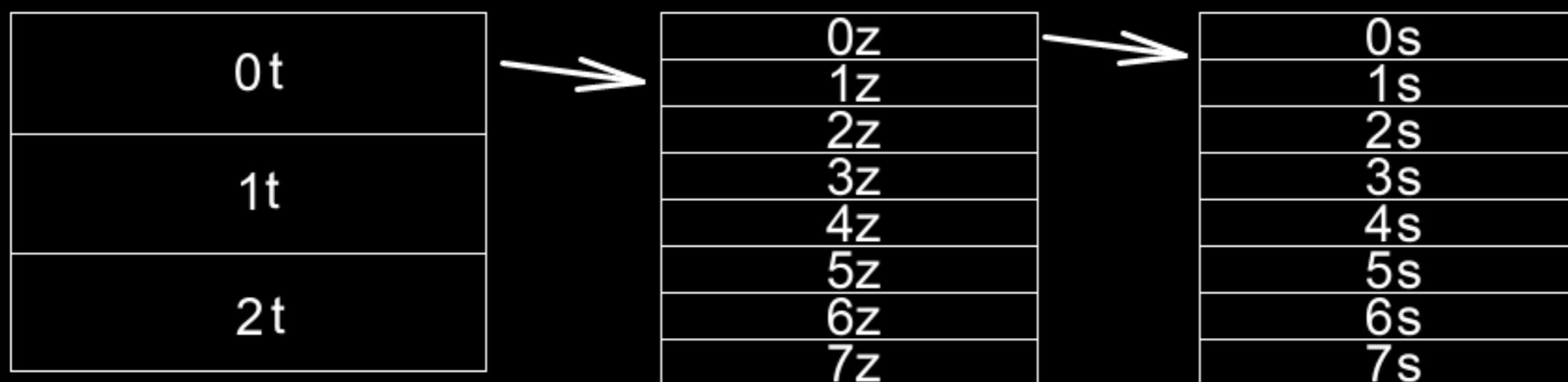
	H		L	
-	-	-	-	-
	0 1 0 t t s s z z x x x x			
-	-	-	-	-
	++	---+	---	-----+
	1	2	3	4

256



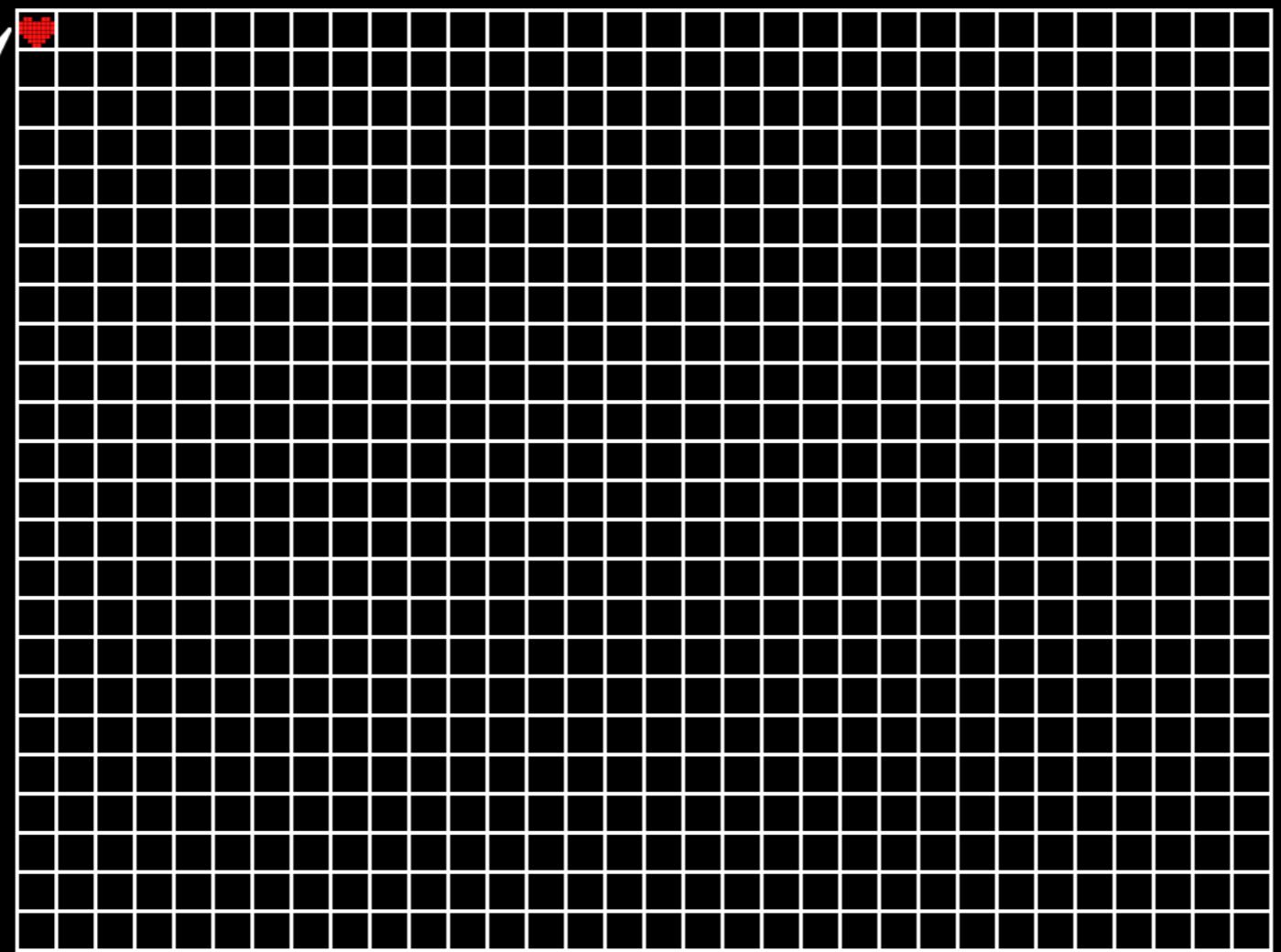
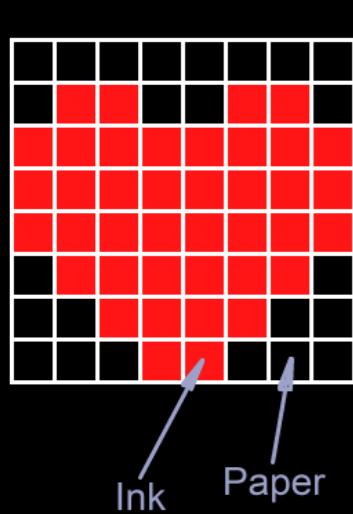
192

- 1(t) Номер трети 0..2
- 2(s) Смещение внутри знакоместа 0..7
- 3(z) Номер ряда 0..7
- 4(x) Номер столбца

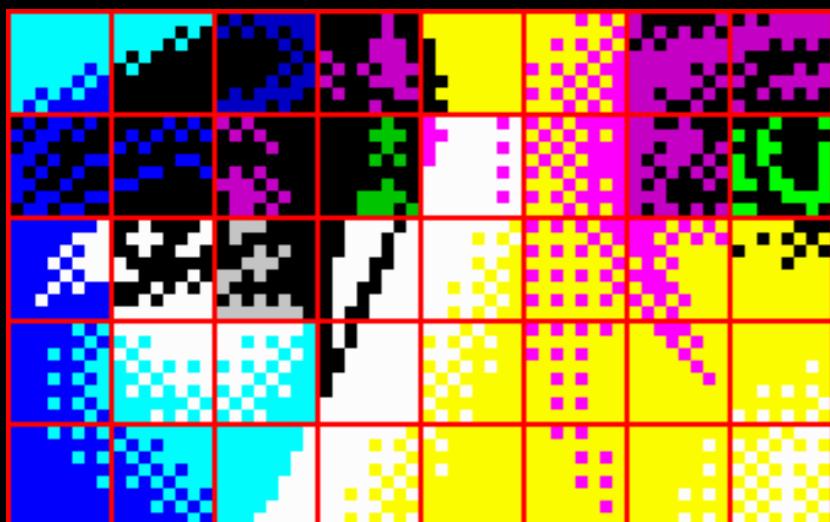


Цвета

32



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



- D0..D2 - цвет "чернил" (INK)
- D3..D5 - цвет "бумаги" (PAPER)
- D6 - бит яркости (BRIGHT)
- D7 - бит мерцания (FLASH)

Порт #FE

Чтение	Запись
<p>D0-D4 - отображают состояние определённого полуряда клавиатуры ZX Spectrum. Порты полурядов - #7FFE, #BFFE, #DFFE, #EFFE, #F7FE, #FBFE, #FDDE и #FEFE.</p> <p>D6 - отображает состояние магнитофонного входа.</p> <p>D5, D7 - обычно не используются.</p>	<p>D0-D2 - определяют цвет бордюра.</p> <p>D3 - управляет состоянием выхода записи на магнитофон MIC.</p> <p>D4 - управляет внутренним динамиком (бипером).</p> <p>D5-D7 - обычно не используются.</p>

Переключение страниц ОЗУ/экранов



#7FFD - порт управляющий отображением страниц

D0-D2 - номер страницы ОЗУ, подключенной в верхние 16 КБ памяти (с адреса #C000)

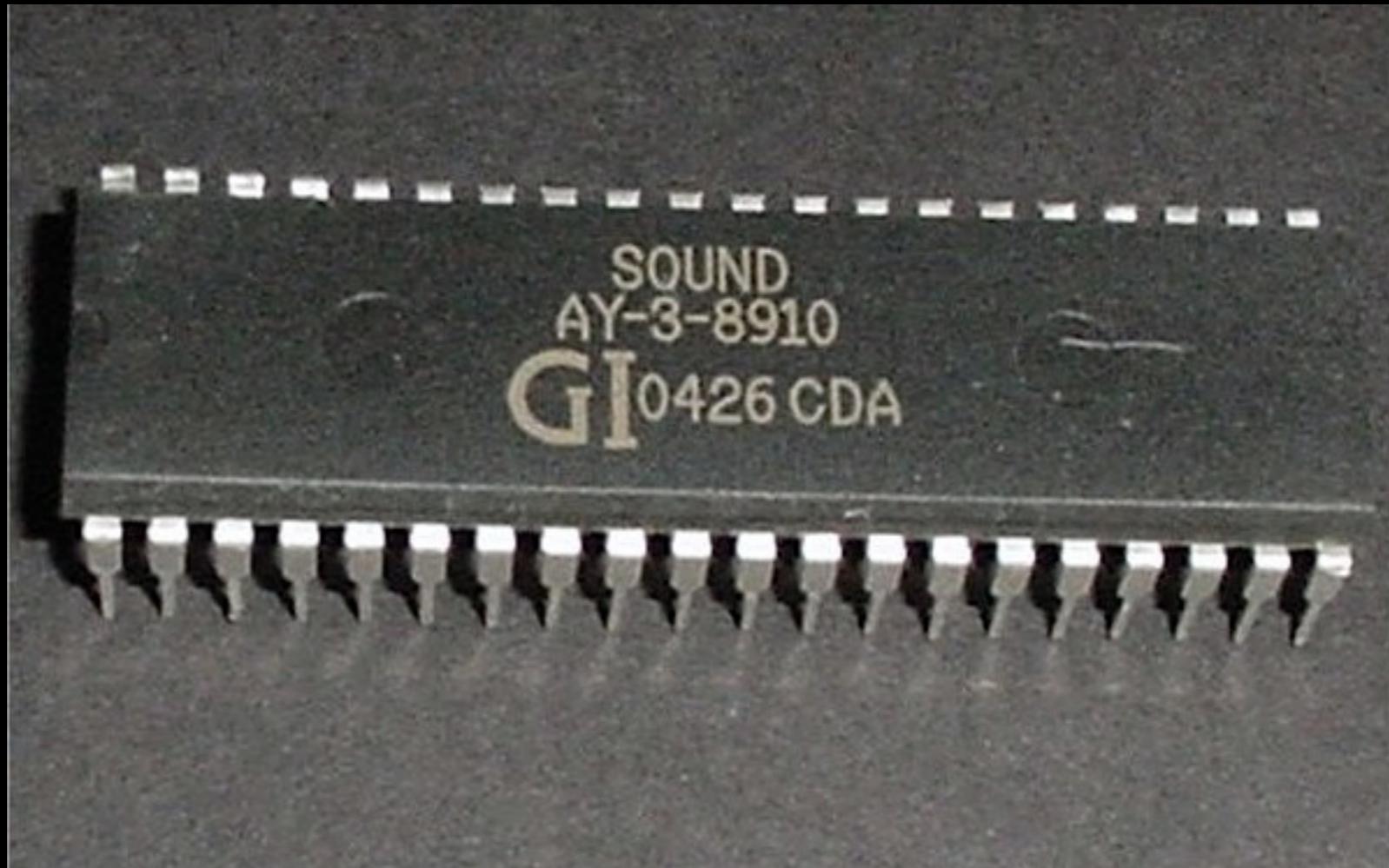
D3 - выбор отображаемой видеостраницы.
0 - страница в банке 5, 1 - в банке 7. **(по умолчанию 0)**

D4 - номер страницы ПЗУ.
0 - BASIC128, 1 - BASIC48. **(обычно используется 1)**

D5 - запрет расширенной памяти (48K защёлка).
(нам надо 0)
При установке бита управление расширенной памятью будет невозможно до сброса компьютера.

Никогда не используйте порт #FD!

AY



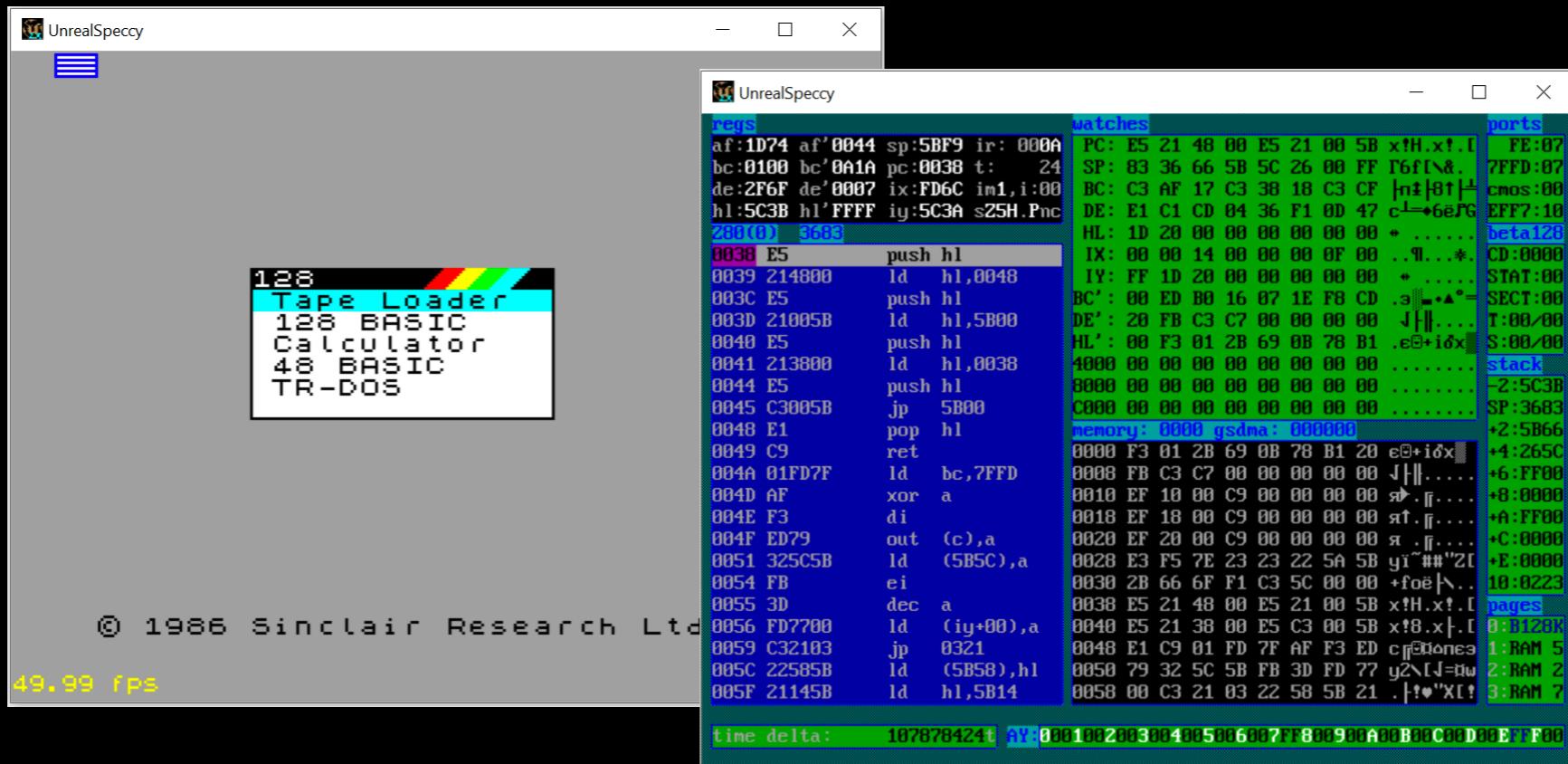
- #FFF D - регистр адреса AY-3-8910
- #BFF D - регистр данных AY-3-8910

Используемый софт и рабочая среда



Unreal Speccy

Эмулятор спектрума (Pentagon)



Запуск: unreal <file_name>

Эмулятор умеет отображать пользовательские метки из файла user.l, при нажатии Ctrl+L.(попробуйте Ctrl+J)

Файл user.l должен лежать в папке с эмулятором



sjAsmPlus

Кросс-ассемблер z80

```
Compiling
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
```

Запуск: sjasmplus <asm_file>.asm --nofakes

*Параметр --nofakes запрещает фейковые инструкции
такие как LD HL,DE => LD H,D: LD L,E*

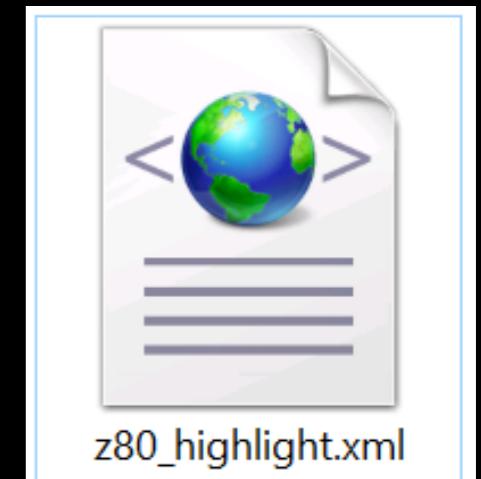
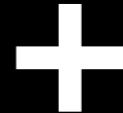
***Крайне рекомендую не использовать фейковые
инструкции!***

Notepad++

Редактор

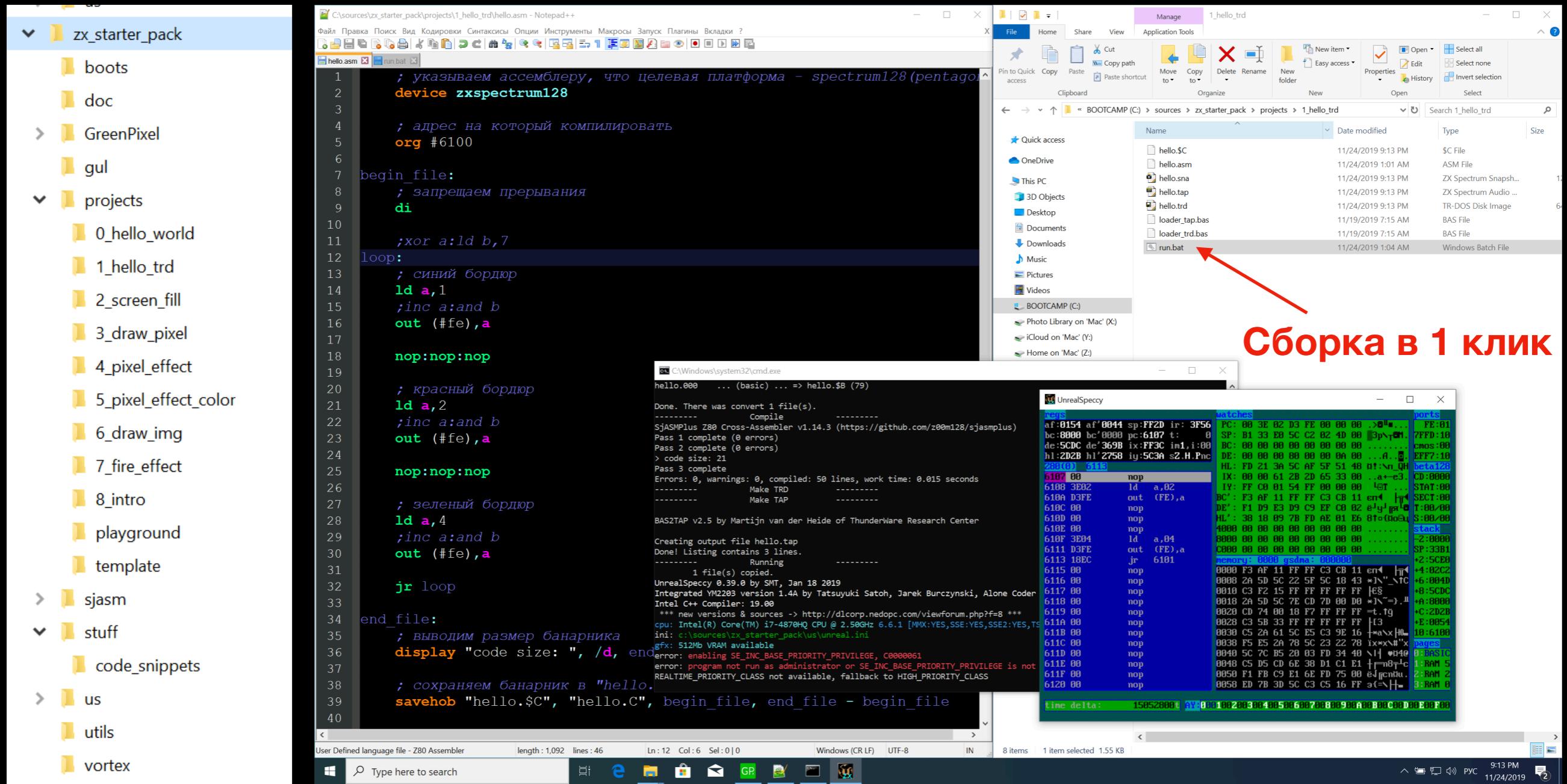
The screenshot shows the Notepad++ interface with a syntax highlighting dropdown menu open over a Z80 assembly code file named 'hello.asm'. The menu lists various syntax definitions, including 'Z80 Assembler' (selected), 'Markdown (Default)', and 'Свой настройки'. The code itself includes directives like .device z, .org #610, and .begin_file, along with assembly instructions like ld a,1 and out (#fe).

```
C:\sources\zx_starter_pack\projects\0_hello_world\hello.asm - Notepad++
Файл Правка Поиск Вид Кодировкі Синтаксисы Опции Инструменты Макросы Запуск Плагины Вкладки ?
hello.asm
1 ; указыв
2 device z
3 ; адрес
4 org #610
5
6 begin_file:
7 ; запрещ
8 di
9
10 loop:
11 ; синий
12 ld a,1
13 out (#fe)
14
15 nop:nop:
16
17 ; красны
18 ld a,2
19 out (#fe)
20
21 jr loop
22
23 end_file:
24 ; выводим размер банарника
25
User Defined language file - Z80 Ass length : 796 lines : 33 Ln : 1 Col : 1 Sel : 0 | 0 Windows (CR LF) UTF-8 IN
```



Кастомная подсветка z80 asm

Рабочая среда



Простейший скрипт для сборки sna файла

```
@echo off
```

```
rem Params
```

```
set name=hello
```

Имя проекта

```
echo -----
```

Compile

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



Компиляция
необходимых
файлов

```
echo -----
```

Running

```
rem Copy labels to emulator
```

```
copy "user.l" "..\\..\\us\\"
```

```
del user.l
```

Копируем метки,
для удобной отладки

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

Запуск эмулятора,
с исполнением
созданного sjAsm sna файла

Простейший шаблон проекта

; указываем ассемблеру, что целевая платформа - spectrum128(pentagon)
device zxpectrum128

; адрес на который компилировать
org #6100

begin_file:

; ВАШ КОД
di :halt ←————

Две этих команды останавливают CPU
di - отключение прерываний
halt - ожидание прерывания

end_file:

; выводим размер бинарника
display "code size: ", /d, end_file - begin_file

; сохраняем sna(снапшот состояния) файл
savesna "hello.sna", begin_file ←————

Сохраняем sna файл,
для запуска в эмуляторе

; сохраняем метки
labelslist "user.l" ←————

Сохраняем метки, для удобства отладки

Регистры Z80

Основные регистры		Альтернативные регистры	
A Аккумулятор	F Флаги	A' Аккумулятор	F' Флаги
B	C	B'	C'
D	E	D'	E'
H	L	H'	L'

BC, DE, HL - регистровые пары

Специальные регистры	
IX (16 bit)	Индексный регистр
IY (16 bit)	Индексный регистр
SP (16 bit)	Указатель стека
PC (16 bit)	Program counter
I	Вектор прерывания
R	Регенерация памяти

IXL, IXH / IYL, IYH - половники индексных регистров

Регистры/доступ к памяти

EXX - обменять BC,DE,HL и BC`,DE`,HL`

EX AF, AF - обменять AF и AF`

EX DE, HL - обменять регистровые пары DE и HL

LD R1, R2 - загрузить значение регистра R2 в регистр R1

LD R1, N - загрузить значение N в регистр R1

LD RR, NN - загрузить значение NN в регистровую пару RR

LD (RR), R1 - загрузить в память, по адресу в регистровой паре RR, значение из регистра R1

LD R1, (RR) - загрузить в регистр R1 значение из памяти, по адресу в регистровой паре RR

LD R,(NN) - загрузить в регистр A/RR значение из памяти, по адресу NNNN

LD (NN),R - загрузить в память, по адресу NNNN, значение регистра A/RR

Логические/арифметические операции

OR R1 - операция OR между A и R1

AND R1 - операция AND между A и R1

XOR R1 - операция XOR между A и R1

OR N/AND N/XOR N - логическая операция между A и непосредственным значением N

CPL - инвертировать A

ADD R1 - сложить R1 и A

SUB R1 - вычесть R1 из A

ADD N/SUB N - сложить/вычесть непосредственное значение N

CP N/R1 - сравнить A со значением N/регистром R1 (SUB, без сохранения результата)

INC R1 - увеличить регистр R1 на 1

DEC R1 - уменьшить регистр R1 на 1

Сдвиги

SRL reg - 0 >>

Оптимизация:

SLA A (заменяется на ADD A,A)

SRA reg - 0/1 >> (арифметический)

SLA L:RL H (заменяется на ADD HL,HL)

RR reg - C >>

RL L:RL H (заменяется на ADC HL,HL)

SLA reg - << 0

SLI reg - << 1

RL reg - << C

RLCA reg - [<<] - циклический сдвиг

RRCA reg - [>>] - циклический сдвиг

РАБОТАЮТ БЫСТРЕЕ ЧЕМ ОСТАЛЬНЫЕ СДВИГИ:

RLA - << C

RRA - C >>

Переходы

JP NN - перейти по адресу NN

JP <NZ,NC,PO,PZ,PE,C,M>, NN - перейти по адресу NN, в зависимости от условия

JR N - относительный переход +127...-128

JR <NZ,NC,Z,C>, N - относительный переход +127...-128, в зависимости от условия

DJNZ N - цикл, уменьшает В, если В не равно 0, относительный переход по N

Условие	Флаг	Описание
Z / NZ(не)	Z	Результат операции равен 0
C / NC(не)	C	Произошел перенос (например 120-130)
P / M	S	Результат операции положительный / отрицательный (7й бит)
PO / PE	P/V	Четность(кол-во бит!) / переполнение(поменялся знак)

Подпрограммы/прерывания

CALL NN - вызов подпрограммы по адресу NN

RET - возврат

CALL <NZ,NC,PO,PZ,PE,C,M>, NN - условный вызов подпрограммы по адресу NN

RET <NZ,NC,PO,PZ,PE,C,M> - условный возврат

DI - запретить прерывания

EI - разрешить прерывания

HALT - ожидать прерывание => **DI:HALT** - приводит к зависанию

Порты ввода-вывода

IN A,(N) - ввод значения из порта N в регистр A

OUT (N), A - вывод A впорт N

OUT (C), A - вывод в порт, в регистре BC значения A.

Пример:

LD A, #10

LD BC, #7FFD

OUT (C), A

Всегда используйте порт #7FFD,
никогда не используйте порт #FD!



Создание релиза



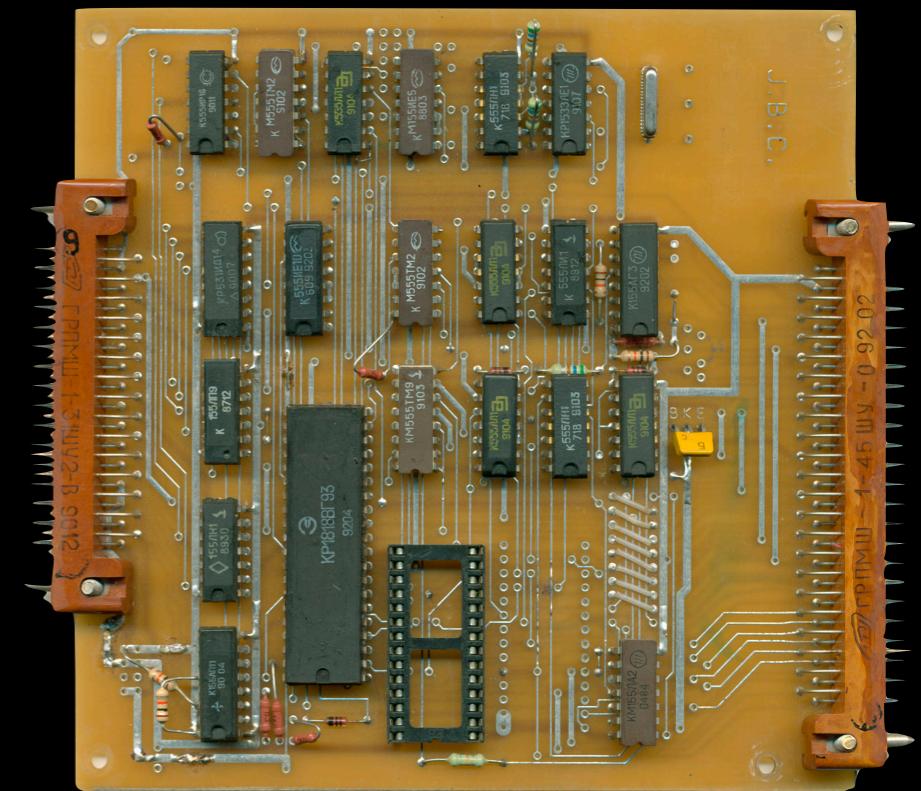
Beta 128 Disk Interface

By Technology Research Ltd

Начало выпуска 1984/1987 год

Поддержка дисков на 640кб

Был широко распространен в СССР и странах СНГ, и крайне малоизвестен на родине в Англии.



(c) Batareikin

tr-dos

Вход в tr-dos:

RANDOMIZE(T) USR(Ex+L) 15616

LIST(K) - вывод списка файлов

RUN(R) - запуск BOOT-а

RUN "(Ss+P)name"(Ss+P) - запуск BASIC файла с именем «name»

RUN "(Ss+P)name"(Ss+P) CODE(Ex+I) - запуск BIN файла с именем «name»

RUN "(Ss+P)name"(Ss+P) CODE(Ex+I) 24832 - запуск BIN файла с именем «name» и адресом



bas2tap

**Утилита для преобразования текстового файла в
tap файл с basic**

Запуск: bas2tap -s<...> -a<...> -c in_file.bas
out_file.tap

Параметры:

- s - Имя файла (отображается при загрузке)
- а - Стока запуска
- с -忽орировать регистр

Преобразование BASIC файла, в tap файле, в Hobeta формат

```
tapto0 -f in_name.tap  
0tohob <имя файла в tap>.000
```

Hobet файл - 1 файл из образа .trd (дискеты)

Форматы файлов:

*.\$C - Кодовый блок

*.\$B - Basic программа

trdtool

Утилита для работы с образом дискеты(trd)

Создание пустого trd файла:

```
trdtool # <trd_name>.trd
```

Добавление hobeta файла в trd:

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

taptool

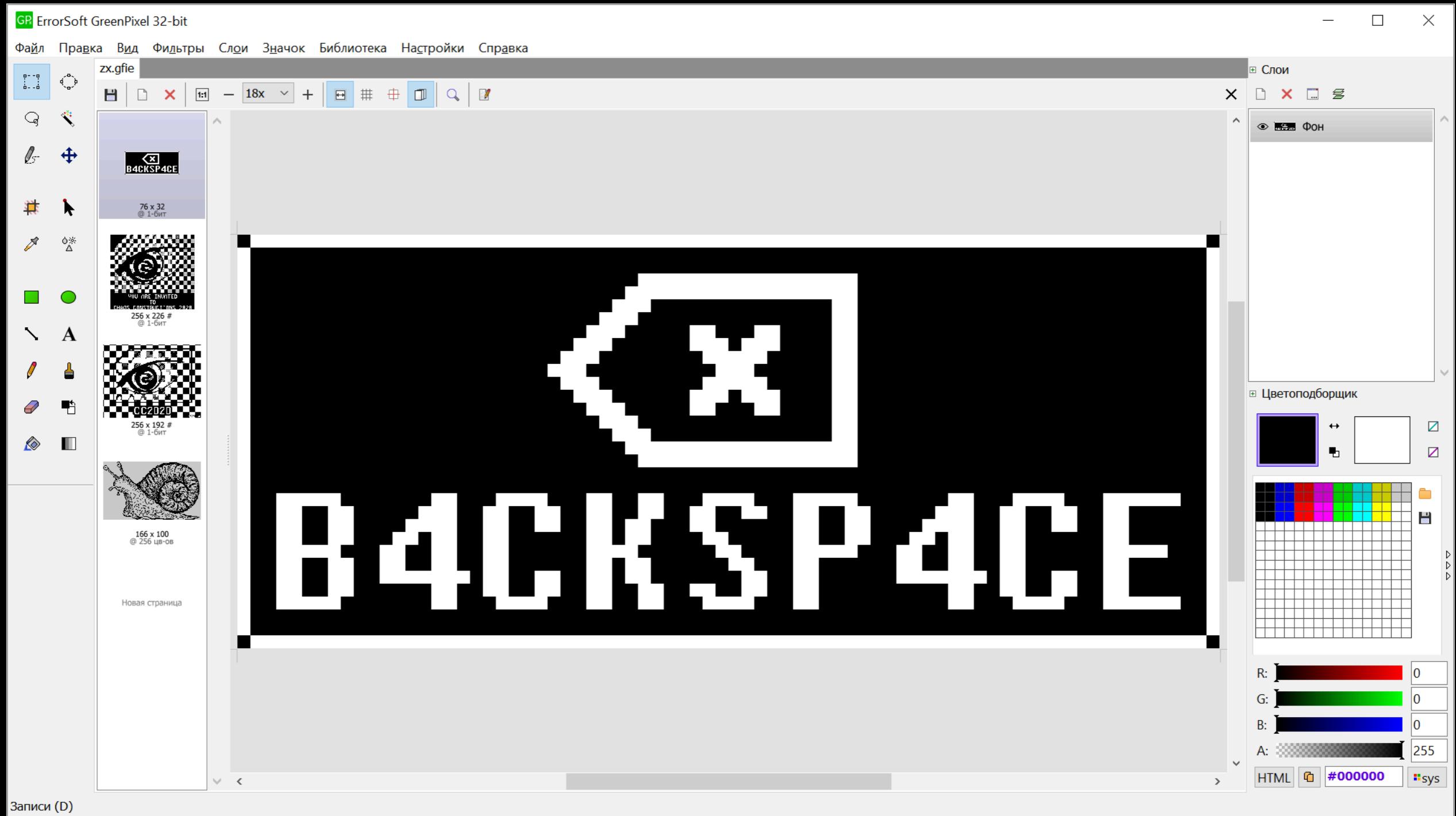
Утилита для работы с образом ленты(tap)

Добавление hobeta файла в tap:

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

GreenPixel

Редактор пиксельной графики



<https://github.com/errorcalc/GreenPixel>

gul

Утилита для преобразования png файла в bin файл

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

Параметры:

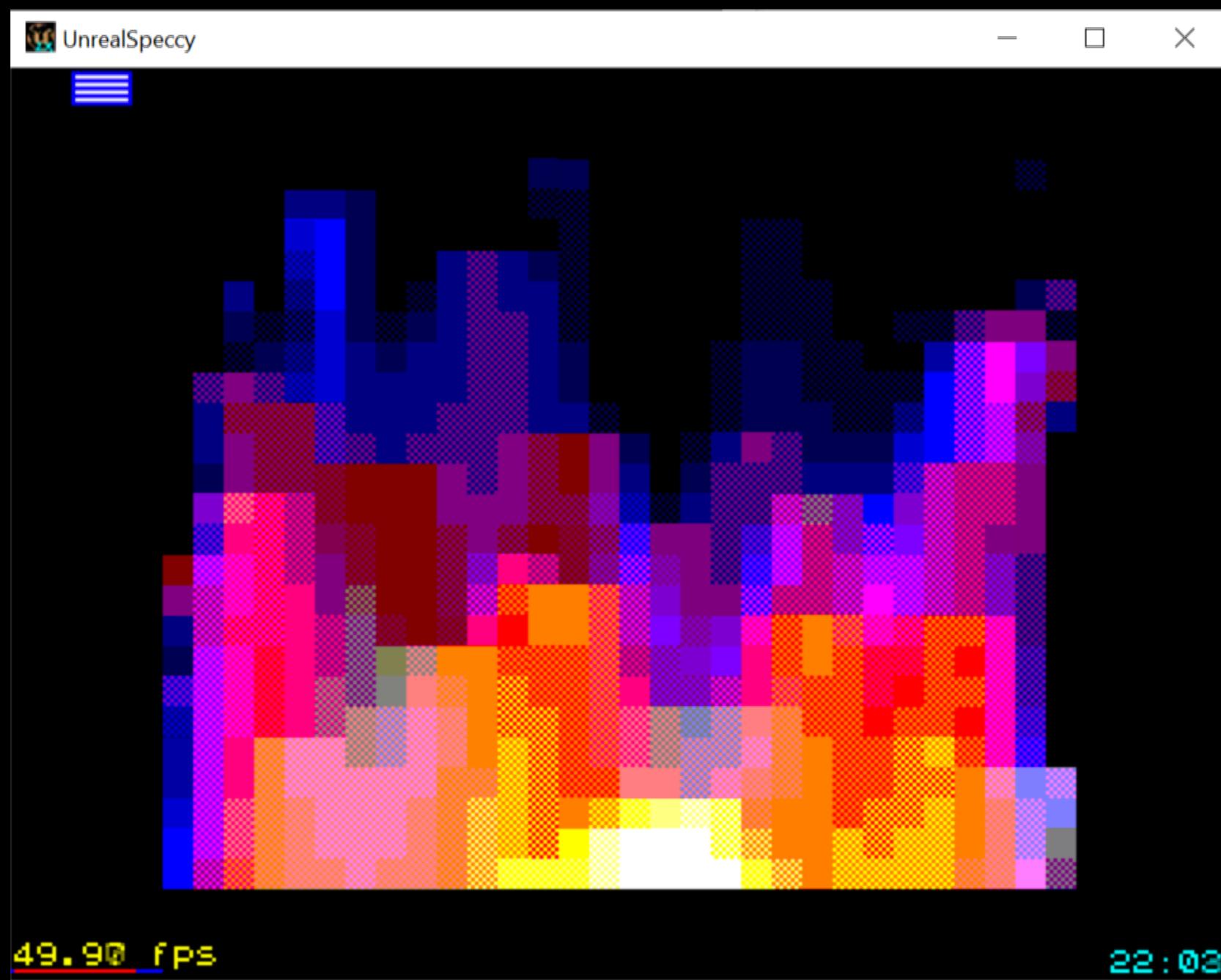
`<-src Source PNG/BMP image>` - in image

`<-dst Destination BIN file>` - out bin

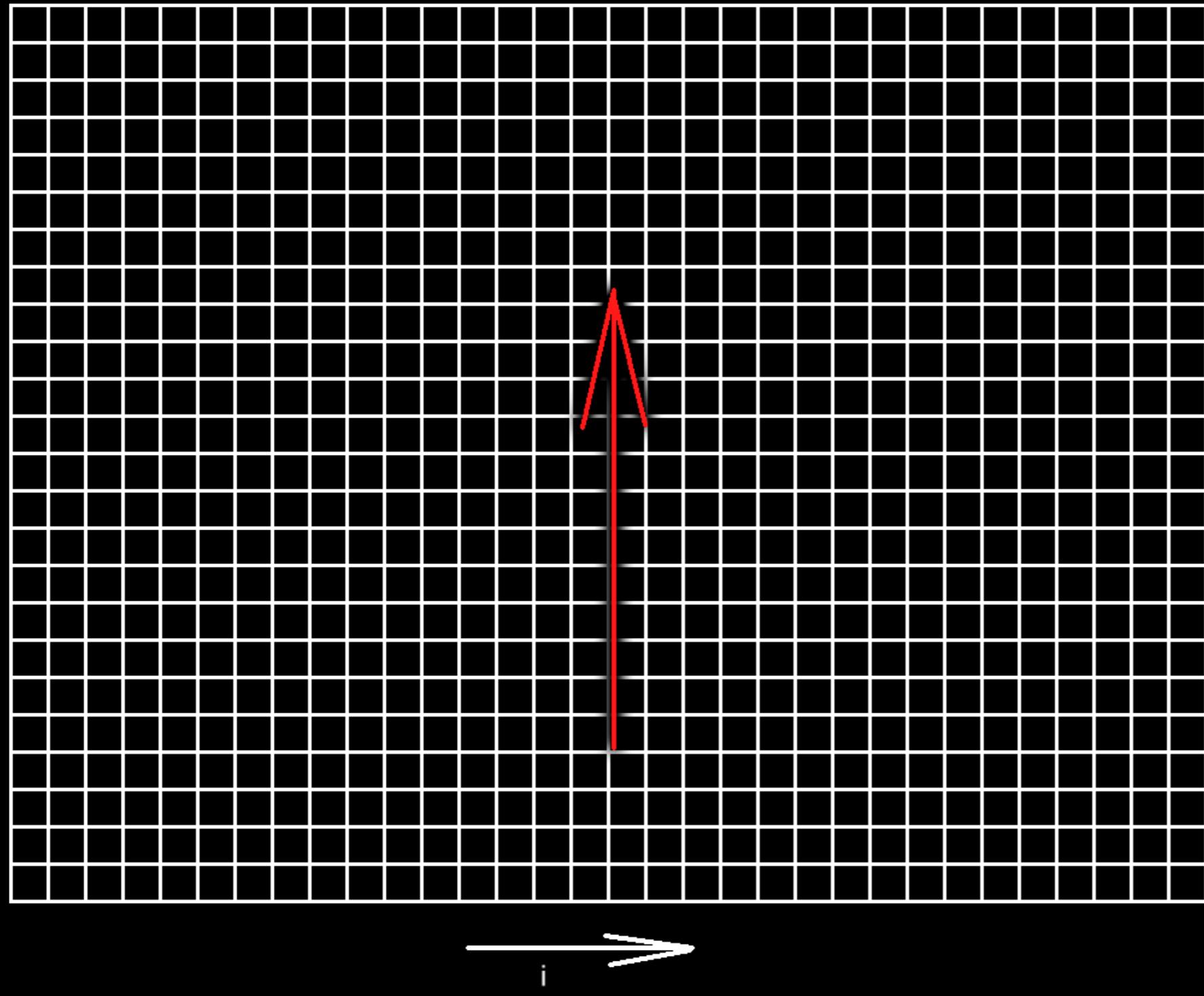
[`-bg RED,GREEN,BLUE`] - background color (0,0,0 by default), other colors are foreground color

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

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
...



0	
1	
...	
127	
128	
129	
...	
254	
255	

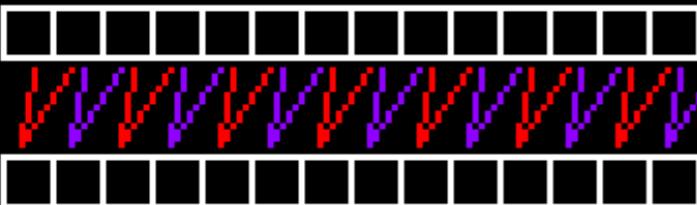
smooth_line_left/smooth_line_right

smooth_line_right



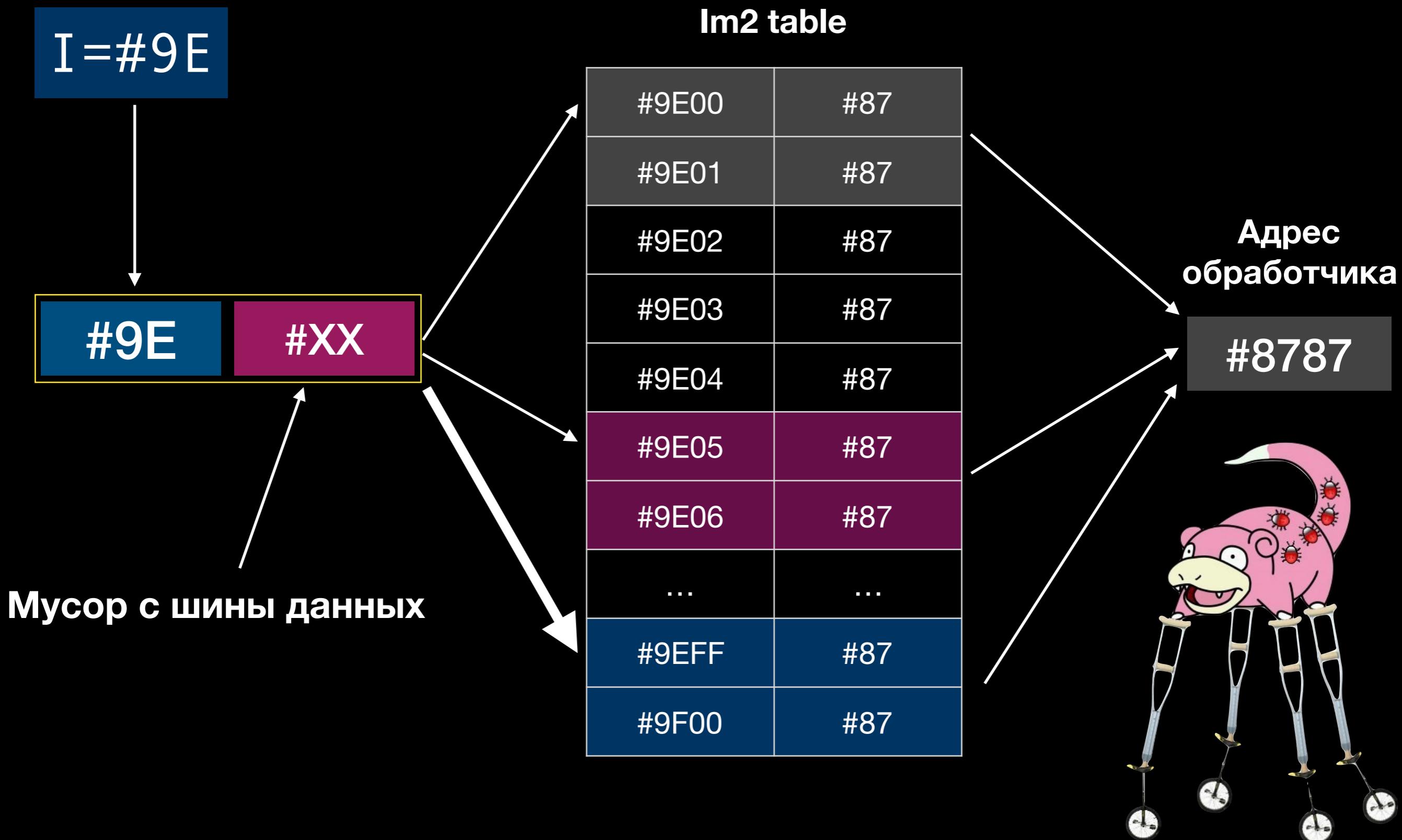
hl [de] := [hl] / 2 + [hl-1] / 2
de

smooth_line_left



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

im2



Музыка

```
call #A000; init
ei
```

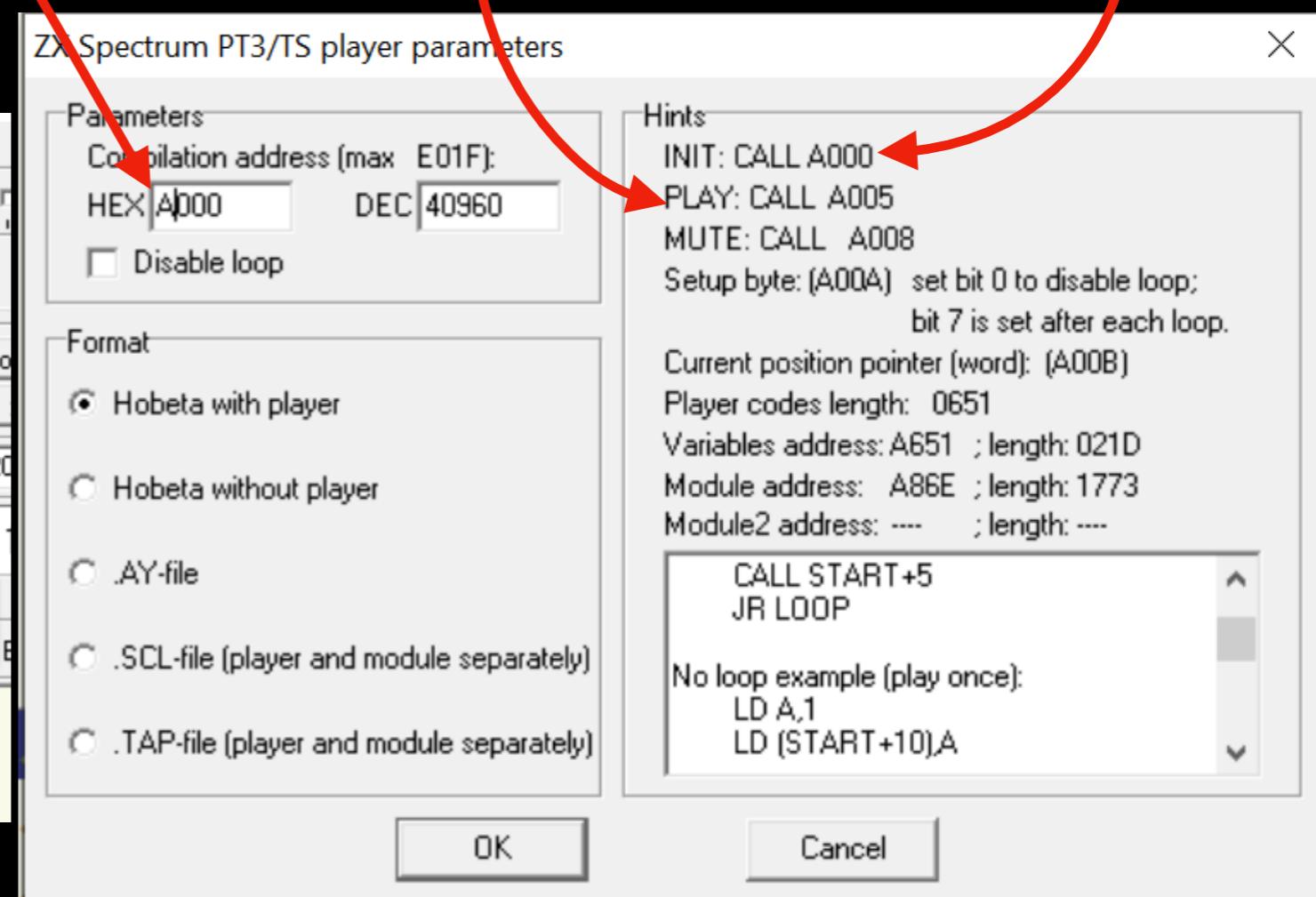
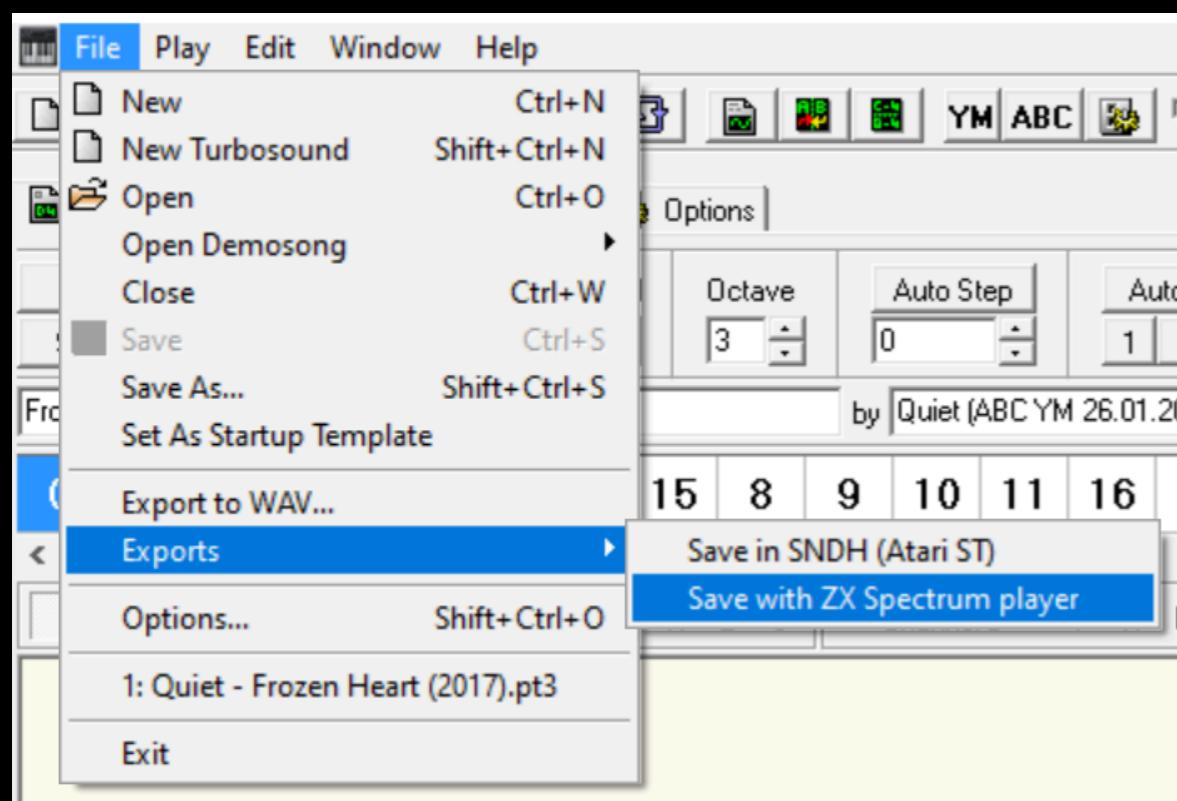
```
loop:
halt
call #A005; играем
jp loop

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

На какой адрес компилировать плеер

Вызов для проигрывания

Вызов для Инициализации



LICENSE

"BEER DEMOSCENE LICENSE v1"

***** error, github.com/errorcalc *****

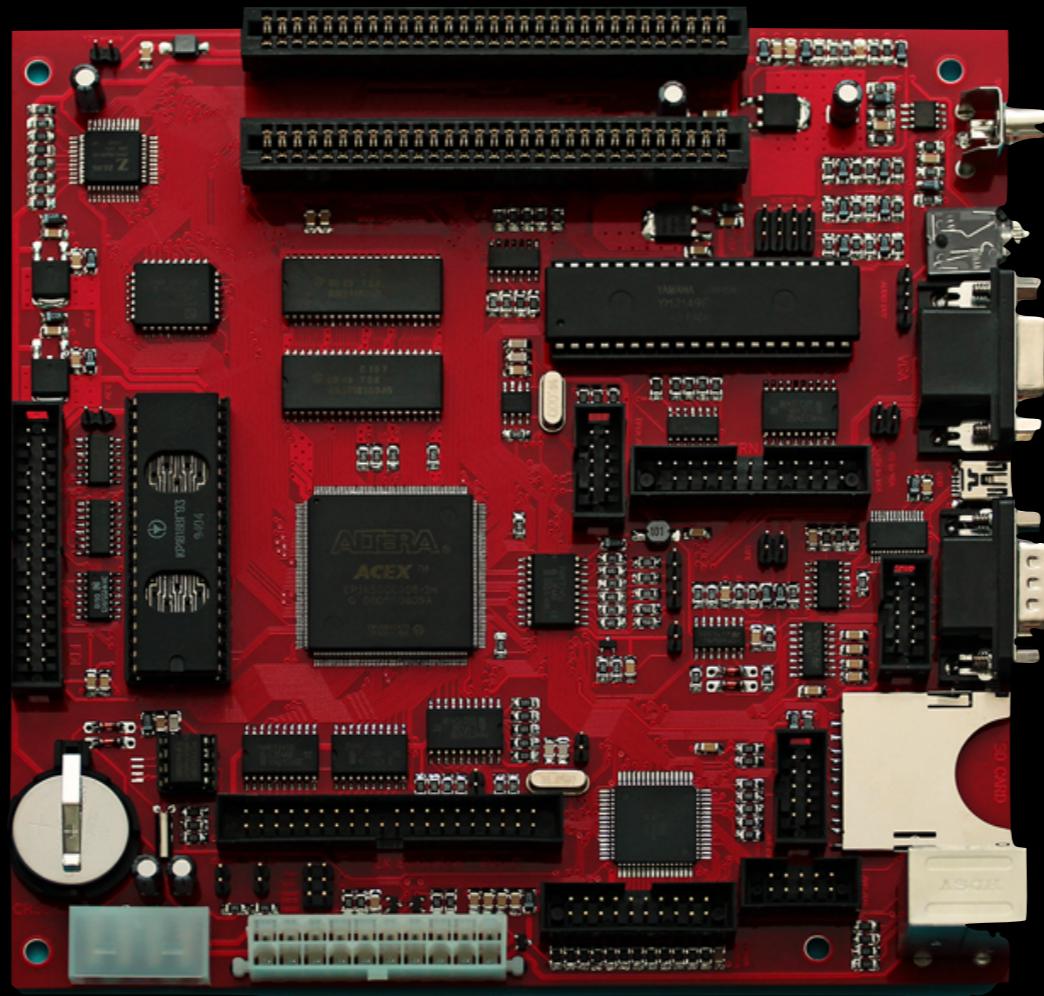
You can do whatever you want with this stuff.

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.

Где взять реал?



ZX Evolution



Информация: <http://www.nedopc.com/zxevo/zxevo.php>

Купить: <http://tetroid.nedopc.com>

Литература

Описание Z80:

1. «Полное описание системы команд микропроцессора Z80» Алексей Асемов (Alex/AT), Advanced Technologies, 2000-2005
2. «ЦЕНТРАЛЬНЫЙ ПРОЦЕССОР Z80CPU» МИНСК УКИК "ЦЕНТР" 1991, PDF version by Deny (Денисенко Д.А.) 2007

Книги доступны в репозитории «ZX Spectrum Starter Pack»

Ссылки

Интерактивная таблица команд Z80:

<http://clrhome.org/table/>

Математика на 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-ти битные сдвиги на Z80:

<http://www.chilliant.com/z80shift.html>

Куча статей о программировании ZX Spectrum:

<http://zxpress.ru/tag.php?id=13>

Сорцы демок:

<https://github.com/ChaosConstructions>

Архив книг, софта, игр, демок:

<https://vtrd.in>

Архив демок:

<https://zxaara.net>

Русскоязычные сообщества

Блог ретросцены:
<http://hypr.ru>

Большой форум о ZX Spectrum:
<https://zx-pk.ru>

ZX Spectrum арт:
<https://zxart.ee>

Telegram сообщества посвященные кодингу под ZX Spectrum:
<https://t.me/zxcoding>
https://t.me/speccy_code
https://t.me/z80_q11

Контакты

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

Скачать/issue/pullrequest: github.com/errorcalc/zx_starter_pack

Скачать: github.com/ChaosConstructions/lessons