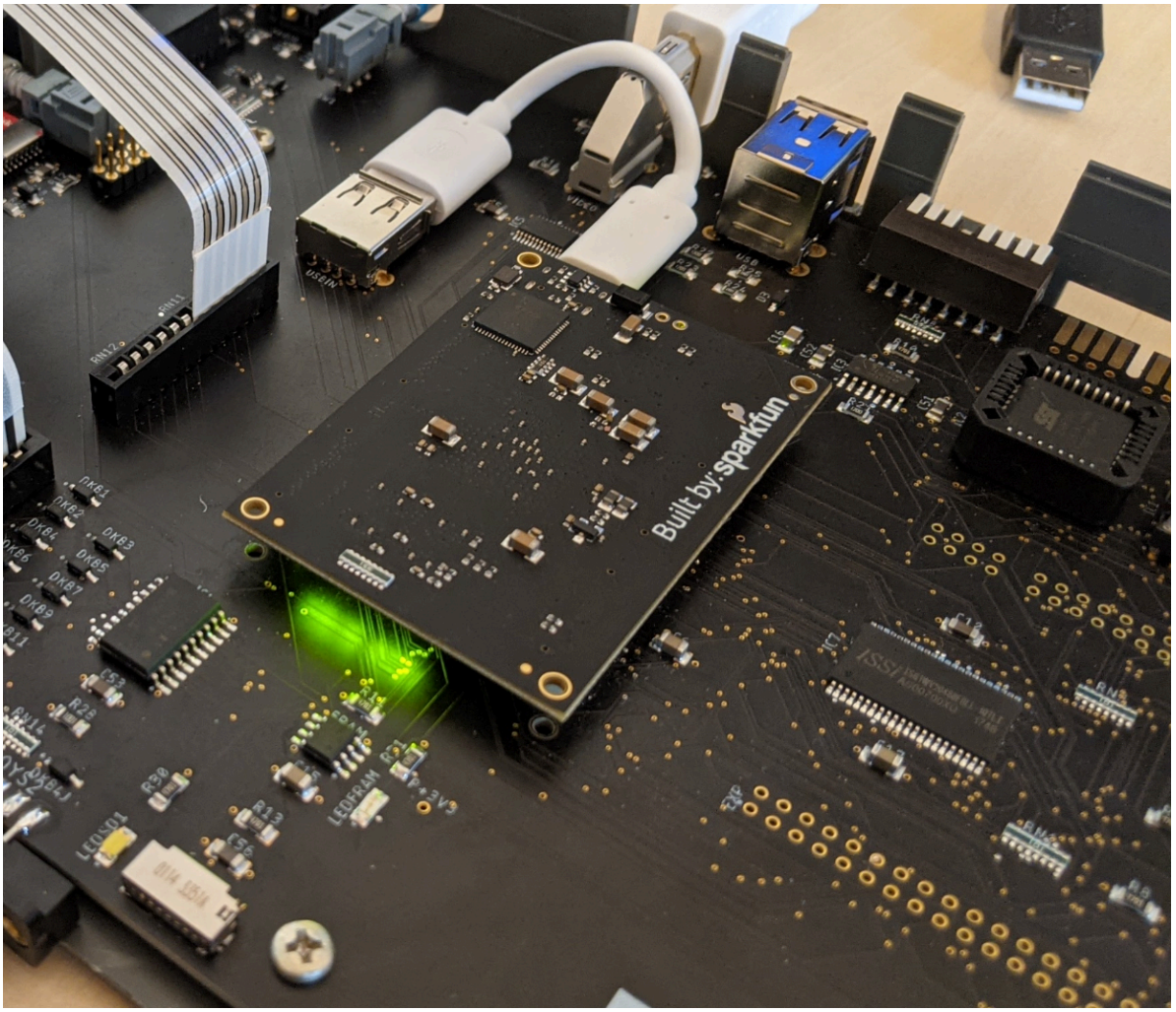
Programmer's Reference

LMN128 / Jan Kucera



version 1.28 / updated 2025-11-17

Quick Help

A

AA-SRAM e_zxi_038
ABC see TurboSound
ACB see TurboSound
Areas (FlashROM) e_zxi_005
attributes
see ULA, Timex, HRC, HRXC
AY see TurboSound

B

beam line (HGFX) g_zxi_01E
beeper e_zxi_002
BigBorder g_zxi_002
border
indexed see HGFX
buffer erase (UART) g_zxi_030

C

chunky
clock
counter e_zxi_0F3-0F8
realtime see RTC
colour table see HGFX
coprocessor see LCU
Covox see MonsterBlaster
CPU see Z80
CRT effect g_zxi_009 e_zxi_01E

D

Device
select #
Didaktik Gama
machine e_zxi_000
divIDE see DivSD
divMMC see DivSD
divRAM see DivSD
divSD e_zxi_001
slots order e_zxi_02D
DMA
= Z80-DMA, FastDMA, SoundDMA
DVI e_zxi_01F

E

eZXi see ZXi hardware
esxDOS see SD, rescue
expansion pack A see AA-SRAM
extra
button e_zxi_020
memory see URAM

F

FastDMA
memory registers g_zxi_041
mode g_zxi_040
FILL
FlashROM
areas e_zxi_005
esxDOS e_zxi_02D
free mem see URAM
FPGA
core time stamp e_zxi_0F9-0FE
flash control g_zxi_0FF
ID e_zxi_01D u_zxi_0FF
frequency modulation
see TurboSound FM
full mode see URAM

G

gamepad (Sega) see joystick
General Sound
GigaScreen g_zxi_008 e_zxi_01D
graphics extra g_zxi_001
gZXi see ZXi hardware

H

HAM
HiColor see Timex graphics modes
HiRes
see HRC, HRXC
see Timex, Pentagon 1024
HRC g_zxi_001
HRXC g_zxi_001
HGFX g_zxi_001
beam line g_zxi_01E
colour table g_zxi_023
general settings g_zxi_020
indexed border g_zxi_020
line interrupt g_zxi_01C
linear memory g_zxi_020
memory registers g_zxi_022
status g_zxi_02F
VRAM g_zxi_021

I

ID see FPGA
interface ID e_zxi_0FF

J

joystick
Kempston e_zxi_003
Sinclair e_zxi_003n
key emulated e_zxi_003
Sega e_zxi_01A

K

Kempston joystick e_zxi_003
keyboard
PS/2 e_zxi_004 u_zxi_004
KeyColour g_zxi_004
KeyLayer g_zxi_003
K-Mouse e_zxi_004

L

LCU
LIFO e_zxi_02F
line interrupt (HGFX) g_zxi_01C

M

machine = memory + timing
see ZX Spectrum, Pentagon, Didaktik
MB02+ e_zxi_001
memory
see AA-SRAM, URAM

memory registers see HGFX
FastDMA g_zxi_041
SoundDMA g_zxi_049
MonsterBlaster e_zxi_002
mouse see K-Mouse
PS/2 u_zxi_009

N

NTSC
frequency see SID

O

overclock see Z80

P

page 6 see VRAM
PAL
frequency (= SID)
palette
see TrueColour, ULA+

Pentagon 128K
machine e_zxi_000
Pentagon 512K
machine e_zxi_000
Pentagon 1024
machine e_zxi_000
speed e_zxi_032
Pentagon 4096
machine e_zxi_000
planar
protect mode (URAM) g_zxi_010
PS/2 see keyboard, mouse

Q

P
port 255 see Timex, SAA 1099

R
Radastan u_zxi_040
see ULA+
realtime clock see RTC
registers = memory registers
rescue (esxDOS) e_zxi_02D
ROM see FlashRAM
RTC e_zxi_02B
time zone e_zxi_03E

S
SAA 1099 e_zxi_002
SafeNMI g_zxi_080
sample
position (SoundDMA) g_zxi_04A
scancode see keyboard
Sega controller see joystick
SD see divSD, Z-Controller
SID e_zxi_002
frequency e_zxi_019
model e_zxi_019
soundchip
see SAA, SID, TurboSound (FM)
SoundDMA
memory registers g_zxi_049
mode g_zxi_048
sample position g_zxi_04A
speed
see Pentagon 1024
see UART, Z80, ZXUno
stack see LIFO
stereo see TurboSound
SounDrive see MonsterBlaster

T
tape
input e_zxi_002
time (zone) see RTC
Timex
graphic modes e_zxi_010
TC2128
TC2144
transparency
see KeyLayer
TrueColour
HGFX table g_zxi_023
TurboSound e_zxi_002
chip model (AY/YM) e_zxi_018
stereo mode (ACB/ABC) e_zxi_018
TurboSound FM e_zxi_002
TV effect see CRT effect

U
UART
buffer erase g_zxi_030
speed g_zxi_030
ULA
audio e_zxi_002
ULA+ e_zxi_011
ultimate
see MB03+, URAM
URAM g_zxi_000
free mem g_zxi_011
full mode g_zxi_010
protect mode g_zxi_010

V
video RAM = VRAM
VRAM (HGFX) g_zxi_021
VRAM
linear see HGFX
mixing see GigaScreen
page6 g_zxi_002

W

X

Y
YM see TurboSound

Z
Z80
overclock e_zxi_031
speed e_zxi_030
Z80-DMA e_zxi_02E
Z-Controller e_zxi_02C
ZXi #
ZX Spectrum 48K
machine e_zxi_000
ZX Spectrum 128K
machine e_zxi_000
ZX Spectrum +2A
machine e_zxi_000
ZX Spectrum SE
ZX Uno
speed e_zxi_032 u_zxi_00B
see keyboard, mouse (PS/2)
see Timex, Radastan

numbers
1099 see SAA
6581 see SID
8x1 see Timex, HiColor
8580 see SID

Content

Types of eLeMeNt ZX hardware ports **9**
ZXi 16-bits ports - general range 9
eLeMeNt range 9
ZX Uno range 9
Programming example 9
Other ports 9
General ZXi ports **10**
g_zxi_000 - Device select 10
g_zxi_001 - Extra graphics modes 10
g_zxi_002 - Advanced video parameters 10
g_zxi_003 - KeyLayer mode 10
g_zxi_004 - KeyLayer mode additional parameters 10
g_zxi_008 - GigaScreen (old TV/Monitor sum colours effect) 10
g_zxi_009 - CRT/TV effect 10
g_zxi_010 - Ultimate layout type activate/Extra memory mode - reflects g_zxi_000 10
g_zxi_011 - RAM in Ultimate mode - low byte - reflects g_zxi_000 11
g_zxi_012 - RAM in Ultimate mode - high byte - reflects g_zxi_000 11
g_zxi_01C - HGFX line interrupt Y position - low byte - default 0 11
g_zxi_01D - HGFX line interrupt Y position - high byte - default 0 11
g_zxi_01E - HGFX beam Y line position - low byte - reflects g_zxi_000 11
g_zxi_01F - HGFX beam Y line position - high byte - reflects g_zxi_000 11
g_zxi_020 - HGFX - general settings 11
g_zxi_021 - HGFX - high byte of Videoram area 11
g_zxi_022 - HGFX - high byte of Registers area 11
g_zxi_023 - HGFX - high byte of Indexed colour table area 11
g_zxi_02F - HGFX status 11
g_zxi_030 - UART speed - read, UART FIFO buffer erase - write 11
g_zxi_040 - FastDMA - mode 11
g_zxi_041 - FastDMA - high byte of Registers area 12
g_zxi_048 - SoundDMA - mode 12
g_zxi_049 - SoundDMA - high byte of Registers area 12
g_zxi_04A - SoundDMA - channel #A high byte of current sample position 12
g_zxi_080 - SafeNMI 12
g_zxi_081 - SafeNMI - low byte 12
g_zxi_082 - SafeNMI - high byte 12
g_zxi_0FF - FPGA Core FLASH control 12
eLeMeNt ZXi ports **13**
e_zxi_000 - Machine - memory/timing mode 13
e_zxi_001 - Storage/Memory Interface 13
e_zxi_002 - Audio output 13
e_zxi_003 - Joystick 13
e_zxi_004 - Mouse/Keyboard 13
e_zxi_00F - Flash memory (ROM Areas) - eLeMeNt ZX v2 only 13
e_zxi_010 - Timex graphics modes 14

e_zxi_011 - ULAplus 14
e_zxi_018 - TurboSound FM 14
e_zxi_019 - SID 14
e_zxi_01A - Joystick/DE9 connector 14
e_zxi_01D - GigaScreen (old TV/Monitor colours sum effect) 14
e_zxi_01E - CRT/TV effect 14
e_zxi_01F - Digital Video Interface - sync and other parameters 14
e_zxi_020 - EXTRA button - short press 14
e_zxi_021 - EXTRA button - long press 14
e_zxi_02B - RTC 15
e_zxi_02C - Z-Controller SD 15
e_zxi_02D - divSD 15
e_zxi_02E - DMA 15
e_zxi_02F - IO LIFO stack 15
e_zxi_030 - CPU speed 15
e_zxi_031 - CPU OverClock mode 15
e_zxi_032 - CPU speed alternative change 16
e_zxi_038 - Expansion pack A 16
e_zxi_039 - Expansion pack B - UART(WiFi) and second PS2 16
e_zxi_03E - RTC Time Zone 16
e_zxi_03F - Diagnostic - for internal development only - can be changed 16
e_zxi_0EF - Interface FPGA ID 16
e_zxi_0F3-0F8 - Real ZX Spectrum Clock Counter 16
e_zxi_0F9-0FF - Interface ID and Core Time Stamp 16
ZX Uno ZXi ports 17
u_zxi_004 - SCANCODE 17
u_zxi_005 - KEYSTAT 17
u_zxi_009 - MOUSEDATA 17
u_zxi_00B - SCANDBLCTRL 17
u_zxi_040 - RADASCTRL 17
u_zxi_041 - RADASOFFSET 17
u_zxi_042 - RADASPADDING 17
u_zxi_043 - RADASPALBANK 18
u_zxi_0C6 - UARTDATA 18
u_zxi_0C7 - UARTSTAT 18
u_zxi_0FF - COREID 18
Other important ports 19
Z80-DMA 19
SaveINT 19
I/O port-based LIFO stack 19
SafeNMI 19
LCU - LMN Coprocessor Unit 20
Addition: 20
Multiplication: 20
Division: 20
How to process signed numbers? 20

Kempston Mouse with wheel **21**
Default (master) mouse 21
Second (slave) mouse 21
Kempston and Sega joystick **21**
External PC Keyboard **22**
ZX Spectrum 128K memory (16K banked) **23**
128K memory 23
+2A/B/+3 memory 23
256K to 4MB memory (16K ZX128 banked) **23**
Pentagon 512KB memory **23**
Pentagon 1024 v2.2 memory **23**
Pentagon 4096 memory **24**
Combined memory variants 24
Didaktik Gama 80KB memory **25**
divSD memory **25**
MB02+ memory **25**
Extra memory **26**
Ultimate RAM paging on eLeMeNt ZX 26
Ultimate RAM paging on MB03+ 26
Fast 8-bits paging ports 26
Memory parameters (port h53) 27
Memory type layout (bits 4-7) 27
Protected Ultimate RAM 27
Shared types of memory 27
Flash memory (ROM Areas) 28
Default ROM Area organisation per 16KB blocks 28
Switching ROM Areas on the eLeMeNt v1 28
FRAM 28
Audio expansions (soundchips) **29**
AY Soundchips/TurboSound FM 29
Second AY/YM 29
ZXi mirrored ports for TSFM 29
SAA1099 29
ZXi assigned ports 29
SAM Coupe ports 29
Sound interface device (SID) 29
SID registers 29
SID parameters 29
More SIDs 30
Audio expansions (sample based) **30**
MonsterBlaster 30
General Sound (GS) 30
ULAplus palette **31**
Register port 31
Groups 31
Palette Group 31

Mode Group 31
Extra Mode Group 31
Data port 31
TrueColour palette 32
Advanced 128K videoRAM 33
Timex Graphics 33
HiColor mode 33
HiRes mode 33
Pentagon 1024 Monochrome HiRes 33
Four VRAMs (ZX Spectrum SE) 33
Radastan graphics mode 34
HRC & HRXC graphic modes 34
HiResColour (HRC) 34
HiResindeXedColour (HRXC) 34
KeyLayer Mode 35
KeyLayer in ZX and HRC modes 35
KeyLayer in HRXC mode 35
Changing a color palette per line 35
HGFX 36
HGFX memory registers 36
h00 - Video parameters 36
h01 - Planar Mode 36
h02 - Advanced commands 36
h03 - IndexColour 36
h04 - PlanarMask 37
h05 to h0C - Offsets 37
h0D - AdvancedGraphicsFeature 37
h0E - SelPaperIndex 37
h0F - FillPlannarMask 37
h10 to h17 - PlannarWorkingWindow 37
HGFX Colour table layout 37
HGFX in chunky format 38
How does planar graphics work? 38
FastDMA 39
FastDMA registers 39
h00 - FastDMA - Parameters (byte) 39
h01 - FastDMA - Block length of transferred data (word) 39
h03 - FastDMA - Source - address (word) 39
h05 - FastDMA - Source - batch length (word) 39
h07 - FastDMA - Source - skip length (word) 39
h09 - FastDMA - Destination - address (word) 39
h0B - FastDMA - Destination - batch length (word) 39
h0D - FastDMA - Destination - skip length (word) 39
SoundDMA 40
SoundDMA registers: 40
h00 - SoundDMA Channel #A - Parameters (byte) 40

h01 - SoundDMA Channel #A - Volume Left (byte) 40
h02 - SoundDMA Channel #A - Volume Right (byte) 40
h03 - SoundDMA Channel #A - Start address (3-bytes) 40
h06 - SoundDMA Channel #A - Loop address (3-bytes) 40
h09 - SoundDMA Channel #A - Length (3-bytes) 40
h0C - SoundDMA Channel #A - Period (2 or 3-bytes) 40
h1x to h5x - SoundDMA Channels #B to #F 40
Sample rates 41
SD cards interfaces 42
Z-Controller 42
divSD 42
Serial and Wi-Fi communication 43
ZXi ports used 43
RX and TX ports 43
Wi-Fi ESP module 44
Real Time Clock 44
Annex I - Reserved ports 45
Annex II - Source codes examples 46
HGFX PAL546 viewer 46

Notes:

Types of eLeMeNt ZX hardware ports

The port layout was done in accordance with the **ZXi specification**: https://zxdesign.itch.io/zxi

All ZXi and other new eLeMeNt ZX ports are writable and readable.

Note: A bit with a prefix means the signal is active in 0. The signal without a prefix is active in 1.

ZXi 16-bits ports - general range

common with other devices like the ***MB03+ Ultimate*** interface

31803 **h7C3B** port0 - select port column **hx00 - hxFF**
32059 **h7D3B** port1 = column1/row1 = g_zxi_000 to g_zxi_0FF

eLeMeNt range

30779 **h783B** port0 - select port column **hx00 - hxFF**
31035 **h793B** port1 = column1/row1 = e_zxi_000 to e_zxi_0FF

ZX Uno range

64571 **hFC3B** port0 - select port column **hx00 - hxFF**
64827 **hFD3B** port1 = column1/row1 = u_zxi_000 to u_zxi_0FF

Programming example

```
ld bc,h783B          //ZXi portselect
ld a,2               //Audio
out (c),a
ld bc,h793B
ld a,%11111110       //disable ULA sound
out (c),a
```

Other ports

Fast 8bit extra memory ports
Old write-only ports made readable (where possible)
Reserved ports - in the Annex of this manual

General ZXi ports

g_zxi_000 - Device select

bit0 = 0-external/1-internal select device (for features like Ultimate memory mode)

g_zxi_001 - Extra graphics modes

00 - none
01 - HiResColour(HRC) - 512x192 pixels / 64x48 attributes / Spectrum attributes
02 - HiResindeXedColour(HRXC) - 512x192 pixels / 64x48 attributes / indeXedColours
03 - HGFX 1 - 256x192 pixels x 256 colour - two buffers - ZX/linear memory layout
04 - HGFX 2 - 512x192 pixels x 256 colour - one buffer - linear memory layout
05 - HGFX 3 - 512x384 pixels x 16 colour - one buffer - linear memory layout
06 - HGFX 4 - 720x546 pixels x 4 colour - one buffer - linear memory layout (Chunky mode only)

Colours are selected from TrueColour palette (RGB)

g_zxi_002 - Advanced video parameters

bit0 = 1 - activate first video buffer to page 6 instead of page 5
bit1 = 1 - enable BigBorder in HGFX modes (obsolete -don't use it)
bit2 = 1 - RGB or 0 - BGR palette order for HRXC and HGFX modes

g_zxi_003 - KeyLayer mode

00 - none
01 - KeyLayer - a key colour in the first screen is transparent for the second one (and vice versa), using ZX128 VRAM0 and VRAM1)

g_zxi_004 - KeyLayer mode additional parameters

Key Layer only - value of Key colour 0-15 for ZX Screen and HiResColour
value of Key colour 0-31 for HiResindeXedColour

g_zxi_008 - GigaScreen (old TV/Monitor sum colours effect)

00 - none (disabled)
01 - GigaScreen mode0: 2 VRAMs always mixed
02 - GigaScreen mode1: two video frames mix
03 - GigaScreen mode2: auto detect mode1

g_zxi_009 - CRT/TV effect

00 - none (disabled)
01 - 1/2
02 - 1/4
03 - 1/8
04 - 1/16
05 - 1/32
06 - 1/64
07 - 1/128

g_zxi_010 - Ultimate layout type activate/Extra memory mode - reflects g_zxi_000

00 - none
01 - Ultimate Full mode - all RAM/FLASH/FRAM access (dev. purposes only)
02 - Ultimate Protect mode - access to user RAM only. System regions as DivSD, ZX, MB02+ is protected

g_zxi_011 - RAM in Ultimate mode - low byte - reflects g_zxi_000

read only - number of free 8KB banks - low byte

g_zxi_012 - RAM in Ultimate mode - high byte - reflects g_zxi_000

read only - number of free 8KB banks - high byte

g_zxi_01C - HGFX line interrupt Y position - low byte - default 0

g_zxi_01D - HGFX line interrupt Y position - high byte - default 0

g_zxi_01E - HGFX beam Y line position - low byte - reflects g_zxi_000

read only - low byte

g_zxi_01F - HGFX beam Y line position - high byte - reflects g_zxi_000

read only - high byte

g_zxi_020 - HGFX - general settings

bit0 = memory layout: 0 - standard ZX; 1 - linear
bit1 = border colour: 0 - standard ZX; 1 - indexed (index 0-7)
bit2 = ZX transparency: 0 - enable; 1 - disable
bit7 = write to HGFX plannar videoram, registers & colour table:
0 - enable; 1 - disable

If index colour no. 0 is used, then an attributed pixel from the ZX-screen will be displayed at the place of the HGFX pixel.

g_zxi_021 - HGFX - high byte of Videoram area

h40 - default

g_zxi_022 - HGFX - high byte of Registers area

h5B - default

g_zxi_023 - HGFX - high byte of Indexed colour table area

start address for a table of 768 bytes, 256 positions
h5C - default

g_zxi_02F - HGFX status

bit0 = 0 - ready; 1 - busy (e.g. erase/copy full buffer)

g_zxi_030 - UART speed - read, UART FIFO buffer erase - write

read gets last UART speed written to port h143B, write h01 causes fast FIFO buffer erase
see also Serial and Wi-Fi communication

g_zxi_040 - FastDMA - mode

bit0 = 0-disable/1-enable FastDMA
bit1 = 0-disable/1-enable second sets of registers
bit2-7 = reserved, must be 0

g_zxi_041 - FastDMA - high byte of Registers area

h5D - default

g_zxi_048 - SoundDMA - mode

bit0 = 0-disable/1-enable SoundDMA
bit1-6 = reserved, must be 0
bit7 = 0-enable; 1-disable write to registers

g_zxi_049 - SoundDMA - high byte of Registers area

h5E - default

g_zxi_04A - SoundDMA - channel #A high byte of current sample position

read only

g_zxi_080 - SafeNMI

bit0 = 0-disable/1-enable RAM Write protection
bit2-6 = reserved, must be 0
bit7 = 0-enable/1-disable push NMI button

g_zxi_081 - SafeNMI - low byte

read only

g_zxi_082 - SafeNMI - high byte

read only

g_zxi_0FF - FPGA Core FLASH control

internal use only

eLeMeNt ZXi ports

e_zxi_000 - Machine - memory/timing mode

```
00 - ZX Spectrum 48
01 - ZX Spectrum 128+2
02 - reserved
03 - ZX Spectrum 128+2A
04 - reserved
05 - Pentagon 128
06 - Pentagon 512
07 - Pentagon 1024 v2.2
08 - Pentagon 4096 (incl. KAY/Profi/
Scorpion/Phoenix/ATM1 memory)
09 - Didaktik Gama 80
```

Every machine uses a corresponding part of ROM Area (16K, 32K or 64K ZX ROMs). For switching whole ROM Areas, see e_zxi_00F.

e_zxi_001 - Storage/Memory Interface

```
00 - none
01 - DivSD (512KB divMMC/IDE)
02 - DivSD+MB (512KB divMMC/IDE & MB02+)
03 - SD only
FE - Rescue mode
FF - obsolete - 
```
don't use it! (former Ultimate mode, use **g-zxi_010** instead)

e_zxi_002 - Audio output

```
bit0 = enable ULA audio (beeper)
bit1 = enable TurboSound FM
bit2 = enable MonsterBlaster (SounDrive, etc.)
bit3 = enable SID (C64 sound chip)
bit4 = enable SAA1099 (SamCoupe sound chip)
bit5 = enable TAPE input
```

e_zxi_003 - Joystick

```
00 - none
01 - Kempston
02 - Sinclair Left
03 - Sinclair Right
04 - Keyboard (OPQA+Space+MN+BHR)
```

e_zxi_004 - Mouse/Keyboard

```
00 - none
01 - K-Mouse Master (external keyboard disabled)
02 - K-Mouse Slave (external keyboard disabled)
03 - External Keyboard
04 - External Keyboard + K-Mouse Master
05 - K-Mouse Master + Slave (external keyboard disabled)
```

e_zxi_00F - Flash memory (ROM Areas) - *eLeMeNt ZX v2 only*

```
00 - Area 0
01 - Area 1
02 - Area 2
03 - Area 3
```

Note: For the *eLeMent ZX v1*, see mechanical DIP-switching, in the *User Manual*.

e_zxi_010 - Timex graphics modes

bit0 = enable Timex gfx mode
bit1 = enable read port hFF (does not work with some ZX Spectrum games, e.g. Arkanoid)

e_zxi_011 - ULAplus

bit0 = enable ULAplus
bit1 = enable Timex mode select by ULAplus regs

e_zxi_018 - TurboSound FM

bit0 = select TSFM chip model: 0-AY, 1-YM
bit1 = select TSFM Stereo mode: 0-ACB, 1-ABC

e_zxi_019 - SID

bit0 = select SID chip model: 0-6581, 1-8580
bit1 = select SID frequency: 0-PAL, 1-NTSC

Note: This port is for SetUp only! In applications, use ZXi SID Control registers **h6C3B** and **h6D3B.**

e_zxi_01A - Joystick/DE9 connector

00 - standard 8-bits era
01 - Sega controller (3-8 buttons)
02 - UART MB03+ ports
03 - UART ZX128-AY ports

e_zxi_01D - GigaScreen (old TV/Monitor colours sum effect)

00 - none (disabled)
01 - GigaScreen mode0: 2 VRAMs always mixed
02 - GigaScreen mode1: two video frames mix
03 - GigaScreen mode2: auto detect mode1

e_zxi_01E - CRT/TV effect

00 - none (disabled)
01 - 1/2
02 - 1/4
03 - 1/8
04 - 1/16
05 - 1/32
06 - 1/64
07 - 1/128

e_zxi_01F - Digital Video Interface - sync and other parameters

bit0 = enable legacy DVI - use on older monitors
bit1 = enable INT frame synchronization (for Ultimate timing - works only on some monitors)

e_zxi_020 - EXTRA button - short press

00 - none (only hold CPU)
01 - CPU speed
02 - Machine
03 - GigaScreen
04 - Warm reset
05 - Joystick/Gamepad interface mode

e_zxi_021 - EXTRA button - long press

00 - none (disabled)
01 - Spectrum menu

In the “Spectrum Menu” you see the speccy-rainbow in the border and you can press a key:

“1” - CPU speed 3.5 MHz
“2” - CPU speed 7 MHz
“3” - CPU speed 14 MHz
“4” - CPU speed 20 MHz
“5” - CPU OverClock 22MHz
“6” - CPU OverClock 28-MHz - 1T delay in ROM
“7” - CPU OverClock 28MHz - RAM only software
“8” - CPU OverClock 30MHz - RAM only software

e_zxi_02B - RTC

```
00 - none (disabled)
01 - FPGA internal - batteryless
```

e_zxi_02C - Z-Controller SD

```
00 - none (disabled)
01 - on first SD slot
02 - on second SD slot
```

e_zxi_02D - divSD

bit0 = select DOS rom: 0-FPGA(rescue esxDOS), 1-FLASH
bit1 = swap order first/second SD
bit7 = keep active over other memory mode

e_zxi_02E - DMA

bit0 = enable

e_zxi_02F - IO LIFO stack

bit0 = enable LIFO

e_zxi_030 - CPU speed

```
00 - 3.5 MHz
01 - 7 MHz
02 - 14 MHz
03 - 20 MHz
04 - OverClock mode 22MHz
05 - OverClock mode 28MHz - 1T delay in ROM area
06 - OverClock mode 28MHz - no delay - for RAM only software
07 - OverClock mode 30MHz - 1T delay - for RAM only software
```

e_zxi_031 - CPU OverClock mode

bit0 = enable OverClock

OverClock mode is intended for an experimental use only, because the system with an overclocked CPU may not work properly. You can try this mode to play games and view demos. It is NOT recommended for software that uses system-critical functions such as writing to an SD card. In this case, data stored on the SD card may be destroyed.

To set the Overclock mode, use the external PC keyboard or the Spectrum menu. Speeds of OverClock modes 06 and 07 (without 1T-state delay) are too high for the internal FLASH memory, therefore you must apply it on software which does not use a ROM.

e_zxi_032 - CPU speed alternative change

bit0 = enable ZX Uno speed change
bit1 = enable Pentagon 1024 v2.2 turbo

e_zxi_038 - Expansion pack A

00 - none (disabled)
01 - Memory 2048KB extension (AA-SRAM)

e_zxi_039 - Expansion pack B - UART(WiFi) and second PS2

00 - UART - internal USB-UART (MB03+ mode) / PS2
01 - UART MB03+ / PS2
02 - UART ZXUNO / PS2
03 - UART ZX128-AY / PS2 disabled

e_zxi_03E - RTC Time Zone

00 - GMT+0	05 - GMT+5	hFB - GMT-5
01 - GMT+1(Zlin)	..	hFC - GMT-4
02 - GMT+2	hF8 - GMT-8	hFD - GMT-3
03 - GMT+3	hF9 - GMT-7	hFE - GMT-2
04 - GMT+4	hFA - GMT-6	hFF - GMT-1

e_zxi_03F - Diagnostic - for internal development only - can be changed

00 - none (disabled)	04 - Status Info
01 - Video sync.	FD - Border under screen
02 - Audio level	FE - Video Test Text Pattern
03 - CPU speed	FF - Video Test Colour Moving Pattern

e_zxi_0EF - Interface FPGA ID

- zxi_0EF - FPGA ID: 00-Au,01-AuPlus,02-RetroCoreX

e_zxi_0F3-0F8 - Real ZX Spectrum Clock Counter

Size of this Time Counter is 48 bits.

- zxi_0F3	- zxi_0F6
- zxi_0F4	- zxi_0F7
- zxi_0F5	- zxi_0F8

Registers are locked when read occurred on reg zxi_0F3 to allow correct reading.

e_zxi_0F9-0FF - Interface ID and Core Time Stamp

- zxi_0F9 - seconds	- zxi_0FC - days
- zxi_0FA - minutes	- zxi_0FD - months
- zxi_0FB - hours	- zxi_0FE - years

- zxi_0FF - Interface ID: hE1 for e-eLeMeNt ZX and 1-board version v1.x

ZX Uno ZXi ports

u_zxi_004 - SCANCODE

Gives the value of the last scancode generated by the keyboard. Use Scan code set 2.
More info: https://wiki.osdev.org/PS/2_Keyboard (use Scan code set 2.)

u_zxi_005 - KEYSTAT

Several bits that indicate whether or not there is a new key pressed, or released, and whether this is an extended or normal key.

bit 0 = PEN	value 1 indicates new data is ready to be read in the SCANCODE register after reading KEYSTAT, this bit becomes 0
bit 3 = ERR	value 1 indicates that the transmission from the PS/2 port had errors
bit 7 = BSY	value 1 indicates that the transmission from/to the PS/2 port is busy

u_zxi_009 - MOUSEDATA

Used to read/send direct commands to the PS/2 mouse. For example: to initialize the mouse, the value hF4 must be sent to this register.

u_zxi_00B - SCANDBLCTRL

bit 0-5 = not used
bit 6-7 = **TURBO** (ZX Uno Z80-CPU speed)

00	3.5 MHz
01	7 MHz
10	14 MHz
11	20 MHz (the highest speed of the real Z80-CPU in the eLeMeNt ZX)

u_zxi_040 - RADASCTRL

bit 2-7 = not used
bit 0-1 = value **b11** to turn radastan on

u_zxi_041 - RADASOFFSET

Contains the number of bytes that must be added to the base address of the screen (h4000h, h6000, hC000 or hE000 depending on the configuration) to obtain the address where the first two pixels are located in Radastan mode.

It is a 14 bit register. To write a value, the 8 least significant bits are written first, followed immediately by the 8 most significant bits (the 2 most significant bits of this value are ignored).

If the offset value is such that scanning the screen memory to create the image reaches the end of a 16KB page, the scan will continue from the beginning of that same page.

64 steps will produce **verticall scroll**.

u_zxi_042 - RADASPADDING

Contains the number of bytes (+64) that a scanline occupies in Radastan mode.

If this register is 0, the length in bytes of a scanline is 64 bytes (128 pixels).

If this record is 4, the length of a scanline is 68 bytes (136 pixels).

If it is 255, the length of a scanline is 64 + 255 = 319 bytes, or 638 pixels.

If the offset value is such that scanning the screen memory to create the image reaches the end of a 16KB page, the scan will continue from the beginning of that same page.

Example: write 1 to that register, it means 65 bytes, enabled to read only 64 bytes every 65 to paint the screen, leaving 1 byte hidden/unused per line. That allows **horizontal scroll**.

u_zxi_043 - RADASPALBANK

Set which section of the ULAplus palette will be used to define the colours in Radastanian mode, and how the border will behave.

bit 3-7 = not used

bit 2 = BOR3

Indicates whether the border colour will be 0 to 7 (0) or 8 to 15 (1).
It can be considered as the bit 3 of the border colour in Radastan mode.

bit 0-1 = RADPALQUARTER

Set a section of the ULAplus palette to use for the radastanian mode.

00 to use the input section 0 to 15
01 for inputs 16 to 31
10 for inputs 32 to 47
11 for inputs 48 to 63

u_zxi_0C6 - UARTDATA

u_zxi_0C7 - UARTSTAT

u_zxi_0FF - COREID

Each read operation provides the next ASCII character in the string containing this text "**eLeMeNt zx**" identification. When the chain ends, subsequent reads emit bytes with the value 0 (at least one of them is emitted) until the chain starts again. This pointer returns to 0 automatically after a reset. The delivered characters that are part of the string are printable standard ASCII (codes 32-127).

ZX-Uno hardware ports are fully described on webpages
https://www.zxuno.com/wiki/index.php/ZX_Spectrum#Nuevos_registros_E.2FS_para_control_de_ZX-Uno
and
http://uto.speccy.org/zxunofaq_en.html#programming

Other important ports

Some Speccy, MB02+ and divIDE/MMC/SD ports **are readable** where possible, e.g.:

23	**h17**	MB02+ memory
227	**hE3**	divSD/IDE/divMMC - control
231	**hE7**	divSD/MMC - select device
235	**hEB**	divSD - data
254	**hFE**	ULA - readable at 40701 **h9EFD**
255	**hFF**	Timex - readable at 40957 **h9FFD** (see Timex Graphics)

8189	**h1FFD**	+2A/+3 and Pentagon memory
32765	**h7FFD**	128/+2/+2A/+3 and 256K+ memory
57341	**hDFFD**	256K+ memory (Profi)
65021	**hFDFD**	256K+ memory (ATM)

Bits 3 and 4 of the port h1FFD are not implemented, they return the last written value.
Ports h2FFD and h3FFD return hFF.

Z80-DMA

Provides fast and direct access to memory or port.

11	**h0B**	this DMA is MB02 compatible

Transfers linear data blocks (max. 64kB) in modes:

- memory to memory
- memory to port
- port to memory
- port to port

A detailed documentation is on the webpage https://velesoft.speccy.cz/data-gear.htm
See also FastDMA, SoundDMA and SaveINT.

SaveINT

All DMA transfers in the eLeMeNt hardware are complemented by the SaveINT feature. If an INT interrupt occurs during a DMA transfer, it will not be discarded. The INT is executed immediately after the end of the DMA transfer.

I/O port-based LIFO stack

intended for new esxDOS syscalls and others, e.g. NMI menu

43	**h2B**	circular 32 bytes buffer

SafeNMI

SafeNMI is an enhancement that will help especially in NMI menus and Memory Snappers. It protects memory and ensures error-free processing of the return address. Thus, it prevents the stack from being overwritten.

LCU - LMN Coprocessor Unit

This is a versatile calculator, a hardware multiplier and divider was designed by Busy soft.
The implemented mathematical expression is: `Z = ( A * B + C ) / D`
LCU memory map: `h2000 to h201F (h0000 to h001F)`

Input members - set by the program

`h2000 [4]` ... member A (to be added)
`h2004 [4]` ... member B (to be added)
`h2008 [4]` ... member C (to be added to the multiplication result)
`h200C [4]` ... member D is a divisor

Intermediate result of multiplication - set by the program or by the FPGA

`h2010 [8]` ... a dividend = intermediate result A*B+C as a member for division

Division results - set by the FPGA

`h2018 [4]` ... division result - quotient
`h201C [4]` ... division result - remainder

Extended command

`h2020 [1]` … h00 - reset all input to defaults

In brackets [ ] is the width of the data (in bytes). In order to compute the whole mathematical expression it is necessary to fill all members A, B, C and D and the result will appear in the next 16 bytes.

Addition:
set A=1
set members B and C
output: result of addition of B and C numbers

Multiplication:
set members A and B
set C=0
output: result of multiplication of A and B number

Division:
set a dividend into cells of the intermediate result
set D=divisor
output: division results in quotient and reminder

How to process signed numbers?

For multiplications and divisions, all negative values must first be converted to positive (this is the negation of all bits + the increment) and the sign of the result is then the XOR of the signs of the operands.

When calculating X+C (where X is the result of A*B), it is first necessary to determine whether the absolute value of the negative C is greater than X, and if not, C does not need to be converted. If it is greater, the result of A*B+ will be negative and must be converted to positive and after then the division must be performed. It's a lot of testing, negations, increments, but they are all trivial operations at the level of addition/subtraction, which are quickly done.

Kempston Mouse with wheel

Default (master) mouse

```
64479  hFBDF    X-axis (horizontal) position
65503  hFFDF    Y-axis (vertical) position
64223  hFADF    buttons
     bit 0 =    zero if right button pressed
     bit 1 =    zero if left button pressed
     bit 2 =    zero if middle button pressed
     bit 3 =    unused (originally: 4th button)
     bits 4-7 = mouse wheel position (b1111 defaults)
```

This port returns 255 if the mouse is stationary and no buttons are pressed.

```
65247  hFEDF    detection of connected master K-mouse, h80 if active
```

Second (slave) mouse

```
15327  h3BDF    X-axis (horizontal) position
16351  h3FDF    Y-axis (vertical) position
15071  h3ADF    buttons
     bit 0 =    zero if right button pressed
     bit 1 =    zero if left button pressed
     bit 2 =    zero if middle button pressed
     bit 3 =    unused (originally: 4th button)
     bits 4-7 = mouse wheel position (b1111 defaults)
16095  h3EDF    detection of connected slave K-mouse, h80 if active
```

In order to operate a K-Mouse implemented in the eLeMeNt ZX, an USB mouse has to support a PS/2 protocol. The list of the compatible mice is in the file “eLeMeNt ZX peripherals - notes” in https://sites.google.com/view/elementzx/docs

Kempston and Sega joystick

```
31  h1F
     bit 0 = right          bit 4 = button1
     bit 1 = left           bit 5 = button2
     bit 2 = down           bit 6 = Sega Controller
     bit 3 = up             bit 7 = is always 0
```

By setting the e_zxi_01A (Joystick/GamePad) we say what game controller is inserted into the 9pin D-SUB (Canon) connector: standard **Atari joystick** with up to 2 buttons or **Sega gamepad** with 3-8 buttons.

The e_zxi_003 (Joystick) sets how the controller is emulated in the hardware (some of the legacy joysticks like Kempston or Sinclair or mapping to a keyboard).

External PC Keyboard

A PC keyboard with USB socket and compatible with PS/2 standard or a keyboard with a PS/2 socket and a PS/2 to USB adapter can be attached to the eLeMeNt's upper USB port.

Scancodes generated by an external keyboard are readable at **u_zxi_004 - SCANCODE** and **u_zxi_005 - KEYSTAT** ports.

The **ZX Spectrum keyboard** is mapped as follows:

Caps Shift	..……..	Shift	**Up**	*CS+7* ……..	Up
Symbol Shift	……..	Ctrl	**Right**	*CS+8* ……..	Right
Extend mode	*CS+SS* ……	Tab	**Graph**	*CS+9* ……..	Insert
Edit	*CS+1* ……..	accent (`)	**Delete**	*CS+0* ……..	BackSpace
Caps Lock	*CS+2* ……..	CapsLock		*CS+SS+0* …..	Delete
TrueVideo	*CS+3* ……..	PageUp	**Break**	*CS+Space* ….	Escape
InvVideo	*CS+4* ……..	PageDown	**<=**	*SS+q* ……..	Home
Left	*CS+5* ……..	Left	**>=**	*SS+e* ……..	End
Down	*CS+6* ……..	Down	**:** (colon)	*SS+z* ……..…	\ (backslash)

Additional control keys are:

Win + F1	CPU 3.5 MHz
Win + F2	CPU 7 MHz
Win + F3	CPU 14 MHz
Win + F4	CPU 20 MHz
Win + F5	CPU 22 MHz (overclock)
Win + F6	CPU 28 MHz (overclock, 1T-state delay in ROM area)
Win + F7	CPU 28 MHz (overclock, for RAM based SW only)
Win + F8	CPU 30 MHz (overclock, for RAM based SW only)
Win + F11	EXTRA button
Win + F12	NMI button
Win + d	DVI mode
Win + g	Gigascreen mode
Win + p *or* ***Pause***	pause
Win + Alt + d	diagnostic and debug info
Win + Alt + b	border ZXS diagnostic visible under the screen
Win + Alt + F1	ZX Spectrum 48K
Win + Alt + F2	ZX Spectrum 128K / +2
Win + Alt + F3	ZX Spectrum +2A
Win + Alt + F4	Pentagon 128
Win + Alt + F5	Pentagon 512
Win + Alt + F6	Pentagon 1024 v2.2
Win + Alt + F7	Pentagon 4096
Ctrl + Alt + Delete	reset
Ctrl + Alt + Space	reset - useful for esxdos reboot, press ***Ctrl + Alt + Space*** keep ***Space*** pressed

The list of the compatible keyboards is in the file "eLeMeNt ZX peripherals - notes":
https://sites.google.com/view/elementzx/docs

ZX Spectrum 128K memory (16K banked)

128K memory

The shadow screen is in the bank 7. The normal bank 5 is always between h4000 and h7fff.

32765 **h7FFD**

bits 0-2 = RAM page (0-7) at hC000
bit 3 = normal (0) or shadow (1) screen to be displayed
bit 4 = 0 for the 128K ROM; 1 for the 48K BASIC
bit 5 = if set, memory paging is disabled (until computer reset)

+2A/B/+3 memory

8189 **h1FFD**

bit 0 = paging: 0 = normal, 1 = all-ram
bit 1 = in normal mode: ignored; all-ram: low bit
bit 2 = in normal mode: high bit of ROM selection; all-ram: high bit

ROM 0	128K editor, system, self-test
ROM 1	128K syntax checker
ROM 2	+3DOS
ROM 3	48K BASIC

bits 3-4 = not implemented (originally: FDD and printer)

All-ram: bits 2 and 1 of h1FFD set four RAM banks at 0, h4000, h8000 and hC000.

00	0 - 1 - 2 - 3	10	4 - 5 - 6 - 3
01	4 - 5 - 6 - 7	11	4 - 7 - 6 - 3

Note: If the system is in +2A/B/+3 all-ram mode, esxDOS ROM mapping is disabled.

256K to 4MB memory (16K ZX128 banked)

Pentagon 512KB memory

32765 **h7FFD**

bit 6-7 = 4. and 5. bit for 512K

Pentagon 1024 v2.2 memory

32765 **h7FFD**

bit 5 = 6. bit of RAM paging for 1024K (bit 2 of hEFF7 must be 0)
if bit 2 of port hEFF7 is set then this bit is used for a paging disable!
bit 6-7 = 4. and 5. bit for 512K

61431 **hEFF7**

bit 2 = 1 for 128K, 0 enables 1MB memory
bit 3 = 1 disable ROM and connect RAM page 0 at h0000-h3FFF

Pentagon 4096 memory

This memory expansion was mainly taken from https://github.com/rdacomp/Pentagon-4096. It offers several popular enhanced ZX128+ memory paging systems used in east-european clones, from 256K up to 2048K. And more, 4096 KB, with the 2 MB AA-SRAM memory expansion pack.

Paging system	2 MB eLeMeNt ZX RAM	2 MB eL. + 2 MB AA-SRAM
Scorpion	256K	512K
Profi, ATM1, Kay, Phoenix	512K	1024K

Combined paging systems		
Pentagon +Scorpion/Profi/ATM1/Kay	2048K	4096K

8189 **h1FFD**
bit 0 = 1 disable ROM and connect RAM page 0 at h0000-h3FFF
bit 4 = 4. bit for 256K **Scorpion** 256K
bit 7 = 5. bit for 512K **Kay** 512K

32765 **h7FFD**
bit 6-7 = 6. and 7. bit **Phoenix** 1024K, **Kay** 1024K

57341 **hDFFD** **Profi**
bit 0-1 = 4. and 5. bit for 512K
bit 2 = 6. bit for 1024K *AA-SRAM only*

65021 **hFDFD** **ATM**
bit 0-1 = 4. and 5. bit for 512K
bit 2 = 6. bit for 1024K *AA-SRAM only*

Combined memory variants

Pentagon + Kay /Scorpion	h7FFD bits 6-7	512K
	h1FFD bits 4 and 7	2048K
	h7FFD bit 5	4096K
Pentagon + Profi	h7FFD bits 6-7	512K
	hDFFD bits 0-2	4096K
Pentagon + ATM1	h7FFD bits 6-7	512K
	hFDFD bits 0-2	4096K

Didaktik Gama 80KB memory

This model contains an extra memory, which is divided into two pages of 32 KB, from address 32768.
For paging purposes, the port A (port 95) and control port of the i8255 are implemented.
The usual way of paging in BASIC is using the OUT 127,x command, where x is either 0 or 1.

divSD memory

For full info, see the *DivIDE programming model* at https://divide.cz/files/info/pgm_model.txt

```
119     h77     divRAM control register (originally DivIDE control register)
          bit 0 to 5 = memory bank (8K * 64)
          bit 6      = MAPRAM
          bit 7      = CONMEM
```

MB02+ memory

```
 23     h17     MB02+ memory     bit 0 to 4 = memory bank (16K * 32)
```

Extra memory

There are three sizes of the Ultimate RAM (URAM) implemented in the eLeMeNt hardware. The basic eLeMeNt ZX holds 2 MB URAM. Next 2 MB adds a memory expansion module for the eZX (named AA-SRAM), expanding the total amount of the Ultimate RAM in the computer to 4 MB. The MB03+ Ultimate interface has got 16 MB of URAM.

Having attached the MB03+ to the eZX computer, you can work with their both URAMs. Before using the Ultimate RAM on the eLeMeNt ZX computer, switch to internal, not the external MB03+ memory. Use the **g_zxi_000 - Device select** for the eZX (internal) or MB (external) URAM. Using **g_zxi_000** has an effect even if the devices are not connected together; on the computer itself or on the interface itself, this causes their URAM to be disconnected.

It is suggested to use the Ultimate Protect mode via **g_zxi_010**. See the chapter *"Protected Ultimate RAM"***.** Check a number of free (available) Ultimate RAM pages with **g_zxi_011** and **g_zxi_012** registers.

Ultimate RAM paging on eLeMeNt ZX

The memory paging mechanism of the eLeMeNt ZX computer works with **four 16 KB Zones** (4 parts of the Z80-CPU direct accessible memory from h0000 to hFFFF). Every Zone is split into **two** paging **8 KB Areas**. This makes 8 selectable Areas (or 4 pairs of Area0 + Area1). Every Area you can assign to any 8 KB page from the Ultimate RAM space. This means 256 x 8 KB pages on the basic eZX computer, 512 x 8 KB pages on the 4 MB machine (with the AA-SRAM expansion). You can work with **special RAM types** (LCU, Flash) in **the Zone0 only**.

Ultimate RAM paging on MB03+

The MB03+ URAM consist of 2048 x 8 KB pages. Please note the MB03+ is an external device and can page the URAM **only in the Zone0**. The MB03+ cannot place any part of its URAM to upper parts (from addr. 16384) of the standard computer memory, because of the lack of signals on the ZX Spectrum expansion bus.

Zone0 (eZX, MB03)	h0000 - h3FFF	**Area0**	h0000 - h1FFF
		Area1	h2000 - h3FFF
Zone1 (eZX)	h4000 - h7FFF	**Area0**	h4000 - h5FFF
		Area1	h6000 - h7FFF
Zone2 (eZX)	h8000 - hBFFF	**Area0**	h8000 - h9FFF
		Area1	hA000 - hBFFF
Zone3 (eZX)	hC000 - hFFFF	**Area0**	hC000 - hDFFF
		Area1	hE000 - hFFFF

Fast 8-bits paging ports

```
007  h07  low  byte of Page in Area0 h0000-h1FFF
039  h27  high byte of Page in Area0 h0000-h1FFF
071  h47  low  byte of page in Area1 h2000-h3FFF
103  h67  high byte of page in Area1 h2000-h3FFF
083  h53  memory parameters
115  h73  memory Zone (00-03)
```

Switching through the port **h73** redirects to the given Zone. All what is written to registers h07, h27, h47, h67 and h53 then, applies to this new Zone.

It is necessary to first set the zone (port 73h), then other ports can be set in any order. After turning on the computer, the zone is set to 0. If we only work with zone 0, it is sufficient to change areas; it is not necessary to always set the zone repeatedly.

Memory parameters (port h53)

bit0 = enable write Area0
bit1 = enable write Area1
bit2 = enable one instruction delay ports 07,h27,h47,h67 – *not ready*
bit3 = enable MAPPER – *not ready*
bit4-7 = memory type layout

Memory type layout (bits 4-7)

0 - disable
1 - reserved
2 - **RAM** Area0 & **RAM** Area1
3 - reserved
4 – RAM Area0 & MAPPER config memory in Area1 - zone 0 only – *not ready*
5 - reserved
6 - **FLASH** Area0 & **FLASH** Area1 - zone 0 only
7 - reserved
8 – **HGFX** Area0 & **HGFX** Area1 (Chunky mode) – respect OffsetY and PlanarMask
9 – **RAM** Area0 & **HGFX** Area1 (Chunky mode) – respect OffsetY and PlanarMask
A – **LCU** Area0 & **RAM** in Area1 - zone 0 only
B - reserved
C – **RAM** Area0 & **LCU** in Area1 - zone 0 only
D - reserved
E - **RAM** Area0 & **FRAM** setup memory in Area1 - zone 0 only
F - reserved

On the eLeMeNt ZX, URAM (and also HGFX graphics buffers) are pageable in all Zones and Areas. Use memory type layouts no. **0, 2, 8** and **9** also in the Zone1 - Zone3.

Protected Ultimate RAM

Ultimate protect mode (**g_zxi_010**) ensures the divRAM (divSD 512 KB RAM), MB02+ 512 KB RAM and all pages of the ZX memory (128 KB, 256 KB, 512 KB, 1024 KB, 2048 KB) are not mixed with URAM pages.

Here is an example of Ultimate Protected mode on the eZX computer with a Pentagon 128 (8 x 16 KB pages) and divRAM (512 KB) attached: set g_zxi_010 to 02 (URAM protect), set e_zxi_000 (eg. 05 = Pentagon 128) and set e_zxi_001 to 01 (divSD on). Read g_zxi_011 and g_zxi_012 to check a number of free Ultimate RAM pages.

ZX RAM	divRAM	URAM Protect on eZX 2 MB	URAM Protect on eZX 4 MB (AA-SRAM)
48 KB - 512 KB	512 KB	1024 KB … 128 x 8 KB pages	3072 KB … 384 x 8 KB pages
1024 KB	512 KB	512 KB … 64 x 8 KB pages	2048 KB … 256 x 8 KB pages

Shared types of memory

512 KB DivSD RAM is shared with 512 KB MB02+ RAM. Be carefull when using the 4 MB ZX memory (Pentagon-4096 on the eLeMeNt ZX) with the divSD because there is no Protect mode and divRAM is shared in a part of the 4 MB.

Note: Different type of memory are HGFX graphics buffers, available in the eLeMeNt memory paging as **chunky** areas. They use their own FPGA-located memory, named **BRAM**.

Flash memory (ROM Areas)

Default size is 512 KB (alternatively it is possible use smaller 256 KB or 128 KB chip).
512 KB is divided into 4 ROM Areas, selectable by **e_zxi_00F** (values 0 to 4)**.**
Note: Don't use ROM Area 0 for your personal roms to avoid reflash an area by wrong data!

Default ROM Area organisation per 16KB blocks

0 ROM0	ROM0
1 ROM1	ROM1
2 ROM2	ROM2
3 ROM3	ROM3
4 esxDOS	esxDOS
5	*reserved*
6 Rescue	Rescue system
7 SetUp	eLeMeNt SetUp

Switching ROM Areas on the eLeMeNt v1

On the eLeMeNt ZX board v1, ROM Areas can be switched by DIP 7 and DIP 8, on the back side of the board. 0 means UP and 1 DOWN.

```
DIP8  DIP7
 0     0    - backup of ZX128/+2 roms
 0     1    - backup of ZX128+2A roms
 1     0    - backup of ZX48 roms
 1     1    - free (SE Basic, DiagRom, etc.)
```

FRAM

8 KB FRAM can be used for system internal purposes as BIOS/Setup (holds setting of hw features that are loaded on startup) or system counter (statistics of use).

h2000-h23FF	SetUp/Bios area
h2400-h27FF	User Area 01 - LNX
h2800-h3BFF	reserved
h3C00-h3FFF	System counter

Audio expansions (soundchips)

Audio devices can be turned on/of via the e_zxi_002.

AY Soundchips/TurboSound FM

65533 **hFFFD**
write: select an AY/YM register 0-14
read the value of the selected register
49149 **hBFFD**
write to the selected register

Second AY/YM

`OUT 65533,254` to switch to the 2nd AY chip (all the AY registers will be redirected on the 2nd AY chip)
`OUT 65533,255` to get back to the first AY chip

ZXi mirrored ports for TSFM

32315 **h7E3B**
32571 **h7F3B**

SAA1099

ZXi assigned ports

28219 **h6E3B** data
28475 **h6F3B** register

SAM Coupe ports

255 **h0FF** data (write-only)
511 **h1FF** register (write-only)

Sound interface device (SID)

27707 **h6C3B** select SID Control Register (ZXi assigned port)
27963 **h6D3B** read/write selected SID Control Register (ZXi assigned port)

SID registers

h00-h1C = original SID registers
h1F = SID type and frequency
h20 = more SIDs

SID parameters

The eLeMeNt ZX and MB03+ FPGA implementation can switch various SID types.

h1F = select SID parameters
bit 0 = SID type: 0-6581; 1-8580
bit 1 = frequency:
0-PAL 1-NTSC (985248Hz for PAL C64, 1022730Hz for NTSC C64)
bits 2-7 = always 0

More SIDs

Currently only one SID is implemented.
Reserved: **h20** = multiple SIDs (optional)

Audio expansions (sample based)

Audio devices can be turned on/of via the e_zxi_002.
See also the Sound DMA.

MonsterBlaster

SounDrive, COVOX and CS/CZ DA Convertor compatible

15	**h0F**	Soundrive LeftA, Covox Left
31	**h1F**	Czech A, Soundrive LeftB
63	**h3F**	Czech B
95	**h5F**	Czech C, Soundrive RightD
79	**h4F**	Soundrive RightC, Covox Right

all ports are write only

General Sound (GS)

The eLeMeNt ZX version of the GS has got a 20 MHz "virtual Z80" CPU, 16KB ROM, 2MB RAM and four 8bit sound channels.

For details, see the General Sound Programming Manual by STINGER & CYRAX, translated by iliks/Hugi.

Note: The GS is available only in the FPGA core for the **AU+ board,** with a bigger Artix-7 chip.

ULAplus palette

ULAplus™ is controlled by two ports, the register port and the data port.

Register port

Port **hBF3B** is the register port (write only).
The byte output is interpreted as follows:

bits 0-5 = Select the register sub-group
bits 6-7 = Select the register group

Groups

Following groups are available:

```
00    Palette group
01    Mode
10    Extra (HRC and HRXC)
```

Palette Group

When this group is selected, the sub-group determines the entry in the palette table (0-63).

Mode Group

The sub-group is (optionally) used to mirror the video functionality of Timex port FF and the framebuffer select of port 7FFD as follows:

bits 0-2 = Screen Mode

```
000   screen 0 (bank 5)          100   screen 0 (bank 7)
001   screen 1 (bank 5)          101   screen 1 (bank 7)
010   hi-colour (bank 5)         011   hi-colour (bank 7)
110   hi-res (bank 5)            111   hi-res (bank 7)
```

bits 3-5 = Screen Colour in Hi-Res Mode (foreground/background)

```
000   0 / 7                      100   4 / 3
001   1 / 6                      101   5 / 2
010   2 / 5                      110   6 / 1
011   3 / 4                      111   7 / 0
```

Extra Mode Group

10 *This is a proposal. Not approved yet.*
bits 0-5 = Screen Mode (the same as the **g_zxi_001** - Extra graphics modes)

```
00    none - disable
01    HiResColour (HRC)
02    HiResindeXedColour
```

Data port

Port **hFF3B** is the data port (read/write).
When the palette group is selected, a written byte will describe the colour.
When the mode group is selected, a byte output will be interpreted as follows:

bit 0: ULAplus™ palette on (1) or off (0)

Reading from port hFF3B returns the last data byte written to the currently selected register. This can be used to read back the current palette or determine if palette mode is active. Implementations that support the Timex video modes use port **hFF** as the primary means to set the video mode, as per the Timex machines. It is left to the individual implementations to determine if reading the port returns the previous write or the floating bus.

Note: Modes 2 to 255 are reserved for future use. Groups 10 and 11 are reserved for future use.

TrueColour palette

TrueColour palette is stored in **Indexed Colour Table**, per 3 bytes, in **B-G-R** or **R-G-B** format.
See g_zxi_002 (advanced video parameters).

The TrueColour is used in the HRXC graphic mode and all HGFX modes.

The colour value holds a 24-bit (3 bytes) number, the highest byte is stored first (RGB).

```
hC000 - hFF - Red
hC001 - h00 - Green
hC002 - hDC - Blue
```

In BGR format is the lowest RGB byte stored first.

```
hC000 - hDC - Blue
hC001 - h00 - Green
hC002 - hFF - Red
```

The BGR storage is similar to how the Z80 CPU stores two-byte numbers:

```
h8000 LD BC, h4000
```

is stored as:

```
h8001 - h00
h8002 - h40
```

Format of the RGB table:

h02 - red	h01 - green	h00 - blue	RGB values of indexed colour h00
h05 - red	h04 - green	h03 - blue	RGB values of indexed colour h01
..	..	..	..

Format of the BGR table:

h00 - blue	h01 - green	h02 - red	RGB values of indexed colour h00
h03 - blue	h04 - green	h05 - red	RGB values of indexed colour h01
..	..	..	..

Advanced 128K videoRAM

Through **g_zxi_002** you can activate the first video buffer in the RAM page no.6 instead of the original RAM page no.5. First, you need to set this ZXi port in order to make and display screen from the page 6. After that, you can start to write data to this page (not before!) and such a data will be displayed then.

Timex Graphics

255 **hFF**

bits 0-2 = Mode

000	ZX Spectrum classic screen in area 0
001	ZX Spectrum classic screen in area 1 (2nd videoram)
010	HiColor
110	HiRes

bits 3-5 = PAPER & INK for HiRes

000	Black on white	100	Green on magenta
001	Blue on yellow	101	Cyan on Red
010	Red on cyan	110	Yellow on blue
011	Magenta on green	111	White on black

bit 6 unused (originally - disable timer interrupt)
bit 7 unused (originally - Timex MMU bank select)

This Timex port is readable at: 40957 **h9FFD**
See also e_zxi_010 - Timex graphics modes.

HiColor mode

256x192 pixels, with an attribute per each 8x1 pixels block
Pixels located at h4000 (area 0), attributes at h6000 (area 1). Both areas take h1800 bytes.

HiRes mode

512x192 pixels, 2 colours (one INK and one PAPER colour with active BRIGHT for the whole screen)
HiRes also uses areas h4000 and h6000, alternating each one per each 8pixels(1char)-column.

Pentagon 1024 Monochrome HiRes

61431 **hEFF7**

bit 1 = 1 for HiRes 512*192 (originally B&W, colours can be set via Timex hFF)

Four VRAMs (ZX Spectrum SE)

In addition to the original Timex screen modes, two separate video areas of the ZX Spectrum 128K can be accessed. This is done by using bit 3 of the port 0x7ffd. This gives the ***ZX Spectrum SE*** videosystem with a total of 27K of RAM which can be used for up to four standard screen areas or two HiRes or HiColour screens, from h4000 and h6000 or from hC000 or hE000.

Radastan graphics mode

Radastan graphics is controlled via ZX-Uno ZXi ports.

- 128x96 pixels, no attributes, linear memory layout
- each pixel uses 4 bits of memory, nibbles in a byte: high (bit 4-7) is a left pixel, low (bit 0-3) is a right pixel
- two videorams for buffer switching
- 16 colours, 4 subpalettes
- screen offset, scanline control (e.g. for scrolling)

Memory layout:

6144 B	pixels	h4000-h57FF (hC000-hD7FF)

See also full programming info on the webpage https://zxuno.com/forum/viewtopic.php?t=4075#p28018

HRC & HRXC graphic modes

HR(X)C graphics is controlled via **g_zxi_001** - Extra graphics modes.

HiResColour (HRC)

- 512x192 pixels and 64x48 attributes
- classic ZX Spectrum attributes format
- linear memory layout
- two videorams (ZX128 like buffer switching, per page 7)
- Keylayer and Gigascreen supported

Memory layout:

12 KB	pixels	h4000-h6FFF (hC000-hEFFF)
3 KB	attributes	h7000-h7BFF (hF000-hFBFF)

HiResindeXedColour (HRXC)

- 512x192 pixels and 64x48 attributes
- indexed colours attributes (16 unique INK & 16 unique PAPER colours in Index table)
- linear memory layout
- two videorams (ZX128 like buffer switching, per page 7)
- Keylayer supported

Memory layout:

12 KB	pixels	h4000-h6FFF (hC000-hEFFF)
3 KB	attributes	h7000-h7BFF (hF000-hFBFF)
96 B	Indexed Colour Table	h7C00-h7C5F (hFC00-hFC5F)

Because HRXC colours do not have bright and flash attributes, a PAPER colour may (or may not) be a very different from an INK colour in the same attribute. Every colour from 32 colours (16 for INK and 16 for PAPER) in **Indexed Colour Table** is defined in TrueColour palette, per 3 bytes, in **B-G-R** or **R-G-B** format.

KeyLayer Mode

The 3rd bit of the paging port `h7ffd`, which displays the main or shadow video-page on the ZX Spectrum 128k, determines which screen is a transparent one, "on top" and carries the KeyColour. The second screen, all pixels and colour information in attributes, are positioned below and being displayed through a KeyColour then.

KeyLayer in ZX and HRC modes

One colour from a group of 15 basic ZX colours can be marked as a transparent **Key colour** (KC, see the register **g_zxi_004** - Layer mode - additional parameters). Both screens use the ZX Spectrum colour palette with 8 brighted and non-brighted colors. Black colours in positions 0 and 8 are not distinguished by bright, so if KC = 0, the color no. 8 also becomes a transparent color (and vice versa).

The flash-attribute is not much usable and is not supported in the KeyLayer mode.

KeyLayer in HRXC mode

One colour from 32 unique colours can be marked as a transparent **Key colour** (KC, see the register **g_zxi_004** - Layer mode - additional parameters). The KeyColour is determined by a colour index, not by a colour content. Every screen uses its own colour palette. 16 colours for INK and next 16 colours for PAPER. Because HRXC colours do not have bright and flash attributes, a PAPER colour may (or may not) be a different from an INK colour in the same attribute.

Changing a color palette per line

The HRXC palette is always updated at the beginning of the frame, so changing it during the display process has no effect. The KeyColour can be changed only.

On the contrary, in the HRC mode you can't change the palette but you can change a KeyColour per (micro)line.

HGFX

The HGFX provides

- planar (bitplane layer) memory organisation
- 1 byte for one pixel
- 2, 4, 16, 32, 64, 128 or 256 colours of TrueColour palette
- lowres (256*192), hires (512*192 and 512*384) and ultra-hires (720*546) resolutions
- direct actress to video buffers: chunky (linear) graphics memory
- special display mode FILL (“wired”)
- special display mode HAM (“photorealistic” with ten thousands colours)

The HGFX is controlled with the g_zxi ports, **g_zxi_01C** to **g_zxi_02F** and HGFX memory registers.

1. Turn on a HGFX resolution via g_zxi_001 - Extra graphics modes.
2. Set the g_zxi ports, mainly
   a. HGFX general settings - **g_zxi_020**
   b. HGFX video RAM - **g_zxi_021**
   c. HGFX registers RAM - **g_zxi_022**
   d. HGFX Indexed colour table, **g_zxi_023**
3. Set all values of HGFX memory registers (it is highly suggested to write values of all 16 bytes!).
4. Fill the HGFX colour table.

See the tutorial on https://wiki.ilnx.cz/doku.php/hgfx:start

HGFX memory registers

The HGFX registers area is set via the **g_zxi_022** (high byte only).

Note: AdvancedGraphicsFeature > FILL, HAM were added in the HGFX version 2.0
SelPaperIndex, FillPlannarMask were added in the HGFX version 2.0
Planar Mode > ATTR were added in the HGFX version2.1

h00 - Video parameters

bit0 = 0 - display video buffer 0; 1 - display video buffer 1
bit1 = destination video buffer for writing for all PlanarModes
bit2 = source video buffer for reading in COPY mode and CHUNKY access
bit3 = 0 - none ; 1 - reduce operation only to PlannarWorkingWindow

h01 - Planar Mode

0 - INK
1 - COPY from secondary to primary Offset
2 - ATTR

h02 - Advanced commands

0 - NOP
1 - COPY_ALL - source to destination buffer - PlanarMask respected
2 - ERASE_ALL - destination buffer - PlanarMask respected

h03 - IndexColour

Set index colour for graphics operation

h04 - PlanarMask

Set bitplanes mask for graphics operation. 8 bits mask for index colour

Example:

```
PlanarMode:     INK
CurrentColour: 0 0 1 1 0 0 1 1  - Current Pixel Colour
PlanarMask:    0 0 0 0 1 1 1 1  - PlanarMask for IndexColour
IndexColour:   1 1 1 1 1 1 1 0  - IndexColour
-----------------------------
NEW Colour:    0 0 1 1 1 1 1 0    no change  changed
BYTE :         0 0 0 0 1 0 1 1  - byte written to videoRAM
PIXEL:         x x x x 1 x 1 1  - x: nochange; 1: set NEW Colour
```

h05 to h0C - Offsets

`h05 to h06` - OffsetX primary - write
`h07 to h08` - OffsetY primary - write
`h09 to 0Ah` - OffsetX secondary - read
`h0Bh to h0C` - OffsetY secondary - read

All **Offsets** take **two bytes** and start on position 0,0 like ZX, but are signed.
Due to this feature it is possible to write outside in a hidden position and show only part of the data.
e.g. position -1,-1 (OffsetX=65535,OffsetY=65535) show data shifted left and up by one pixel).

h0D - AdvancedGraphicsFeature

HGFX can interpret graphics memory data in two special ways.

HAM8 is a feature well know from the Amiga graphics system. With a R/G/B shift from the last operated colour (of the pixel on the left) you can achieve thousands colours on the screen.
FILL plots automatically all pixels with colour value 0 as the last non-zero pixels (from the left). It helps prepare large colour areas and change their colours very quickly.

```
0 - NOP - disable
1 - FILL
2 - HAM
```

h0E - SelPaperIndex

h0F - FillPlannarMask

h10 to h17 - PlannarWorkingWindow

`h10 to h11` - LeftUp corner X
`h12 to h13` - LeftUp corner Y
`h14 to h15` - Width X
`h16 to h17` - Height Y

HGFX Colour table layout

See RGB and BRG tables in the TrueColour palette section.

HGFX in chunky format

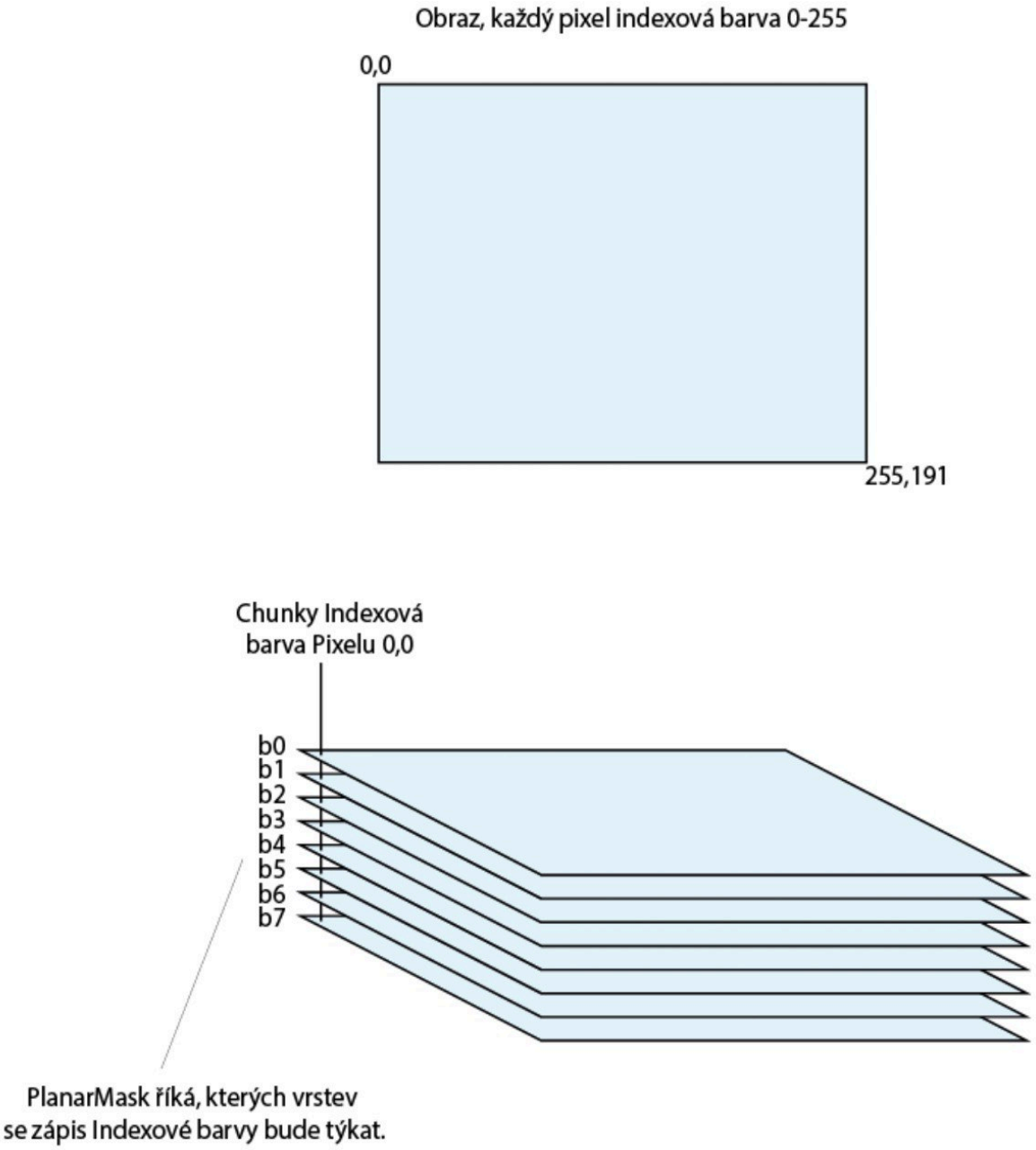
HGFX memory registers basically provide planar-oriented access and manipulations with graphics data. To handle a raw (graphics) memory of both HGFX videobuffers, use fast 8bit ultimate extra memory ports (see Memory parameters (port h53).

Access to this “chunky” memory, when activated, reflects:

- OffsetY
- PlanarMask
- bit1 and bit2 of the HGFX register no. 0 (video parameters)

How does planar graphics work?

See a picture which shows how bitplanes are masked for graphics operation. Colour information is a byte with value (0-256 colours) and can be applied to all planar layers. This number is combined with an eight-bits PlanarMask. PlanarMask indicates, in what planar layer will be the colour effective.



FastDMA

FastDMA allows to transfer the same large groups of bytes in one run. It is necessary to set how many bytes will be transferred in the batch (batch length) and how many will be skipped, i.e. added after the batch transfer (skip length). Such transfers are repeated until the total number of bytes is reached (block length).

See also `g_zxi_040` and `g_zxi_041`.

FastDMA registers

All FastDMA registers area is set via the g_zxi_040 (high byte only).

h00 - FastDMA - Parameters (byte)

```
bit0 = source address 0-keep/1-change value after transfer
bit1 = destination addr. 0-keep/1-change value after transfer
bit2 = transfer type: 0-LDIR/1-LDDR
bit3 = floating source skip: 0-disable/1-enable
bit4 = floating destination skip: 0-disable/1-enable
bit5 = fill 0-disable/1-enable; filling by lower byte of reg 3
bit6-7 = reserved, must be 0 (eg,. select memory/port)
```

Write to this register causes immediate transfer!

h01 - FastDMA - Block length of transferred data (word)

```
h0000 - default (means 65536bytes)
```

h03 - FastDMA - Source - address (word) / fill mode filling byte

```
h0000 - default
```

h05 - FastDMA - Source - batch length (word)

```
h0000 - default (means 65536bytes)
```

h07 - FastDMA - Source - skip length (word)

```
h0000 - default
```

h09 - FastDMA - Destination - address (word)

```
h0000 - default
```

h0B - FastDMA - Destination - batch length (word)

```
h0000 - default (means 65536bytes)
```

h0D - FastDMA - Destination - skip length (word)

```
h0000 - default
```

h10 - FastDMA - Parameters#2 (byte)

bit0 = 0-disable/1-enable 24bits transfer
bit1 = source 0-Z80 Memory/1-URAM (24bits transfer)
bit2 = destination 0-Z80 Memory/1-URAM (24bits transfer)
bit3-7 = reserved, must be 0

h11 - FastDMA Source - 24bits address (3-bytes)

h14 - FastDMA Destination - 24bits address (3-bytes)

h17 - FastDMA Block length of transferred data - 24bits address (3-bytes)

SoundDMA

Main characteristics:

- 6 audio channels #A - #F
- each channel can play 8bits mono or stereo sound
- stereo VOLUME and PAN have 256 levels
- channel #A can play 16bits mono sound
- normal 16bits or extended 24bits period for fine-tunning
- unsigned and signed sample support
- play and loop mode
- wavetable synth ready (variable loop lenght)
- streaming ready (from SD)
- highest data-transfers priority, above the FastDMA and the classic Z80-DMA
- amiga frequencies compatibility

SoundDMA registers:

All SoundDMA registers are set via the g_zxi_049 (high byte only).

h00 - SoundDMA Channel #A - Parameters (byte)

```
bit0 = 0-stop(pause)/1-play
bit1 = 0-play from start/1-continue
bit2 = 0-play only one/1-loop
bit3 = 0-unsigned/1-signed sample format
bit4 = 0-normal /1-extended period
bit5 = not used, must be 0
bit6 = 0-8bits /1-16bits:channel #A mono only, max volume
bit7 = 0-mono/1-stereo
Writing this reg starts transfer
```

h01 - SoundDMA Channel #A - Volume Left (byte)

`bit7-0 = Volume Left` *max recommended value is 64. it's related to 0dB*

h02 - SoundDMA Channel #A - Volume Right (byte)

`bit7-0 = Volume Right` *max recommended value is 64. it's related to 0dB*

h03 - SoundDMA Channel #A - Start address (3-bytes)

works with protected URAM only

h06 - SoundDMA Channel #A - Loop address (3-bytes)

h09 - SoundDMA Channel #A - Length (3-bytes)

h0C - SoundDMA Channel #A - Period (2 or 3-bytes)

h1x to h5x - SoundDMA Channels #B to #F

channels B to F follow… e.g: **`h20 - SoundDMA Channel #C`** – Parameters (byte)

Sample rates

normal period:

$$\text{SampleRate} = \frac{\sim 7.1\text{MHz}}{\text{Period(16bits)}}$$

extended period:

$$\text{SampleRate} = \frac{\sim 113\text{MHz}}{\text{Period(24bits)}}$$

There are _NO_ standard sample rate for the samples used in modules. But most often the samples are done at the rate called C-3. Sometimes drums are sampled at A-3 (around 28 kHz), and some sounds are at ~8 kHz or anything else to save space.

The sample rate on each of the channels can be selected by a period value, which tells the hardware how many ~3.5 MHz clocks to count down before playing the next sample. If you have a 16 kHz sample you simply play it at a note that gives you a 16 kHz sample rate. If you play it one octave lower, you get a 8 kHz sample rate (and a double period value).

So, the most normal rate is (C-3, period 214):

$$\frac{\sim 7.1\text{MHz}}{214 * 2} = 16588.78\ \text{Hz}$$

SD cards interfaces

Z-Controller

simplified implementation of the original schematic, reading `h77` returns `b11111100`

```
119    h77    Z-controller config       bit 1 = active
 87    h57    Z-controller data
```

divSD

```
231    hE7    divSD device select
235    hEB    divSD data
```

The DivSD ports are both write- and readable.
For a memory part of the divSD, see the divRAM.

related eZXi registers

- e_zxi_001 select divSD machine
- e_zxi_02d keep divSD over other memory mode (address trapping works in URAM area)

Serial and Wi-Fi communication

ZXi ports used

g_zxi_030 - UART speed - read, UART_FIFO buffer erase - write
e_zxi_01A - Joystick/DE9 connector
e_zxi_039 - Expansion pack B - UART(WiFi) and second PS2
u_zxi_0C6 & u_zxi_0C7 - UARTDATA & UARTSTAT

The eLeMeNt ZX has one serial port which can be configured for either the **internal UART** pins or the **external pins of the DE9** joystick connector. The internal UART primarily serves as a link to the Wi-Fi ESP module. Three internal UART/Wi-Fi connection standards are available: ZX128-AY, ZX-Uno and buffered MB03+. In addition to the joystick connection, the external DE9 can also provide serial signals as follows:

DE9 pins	RS-232 signals	**ZX128-AY mode**	**MB03+ mode**
1	CTS - input	CTS - **I/O A6**	**-**
2	RxD - input	RxD - **I/O A7**	**RxD**
3	TxD - output	TxD - **I/O A3**	**TxD**
4	RTS - output	RTS - **I/O A2**	**-**
5	**+5V**		
8	**GND**		

Pins 6, 7 and 9 are not used for serial communication. The joystick connector uses digital (TTL) voltages instead of analogue voltages and requires a TTL-RS232 converter.

RX and TX ports

The ports are **h143B** = 5179 for RX and **h133B** = 4923 for TX.
Size of the read (RX) FIFO buffer is 2048 bytes. Data are lost when the buffer is full. Reading from the TX port shows the status of the FIFO buffer and the TX line. This can be used to control the flow of data.

`bit 0` = RX status: '0' if the buffer is **empty**, '1' if there is a data to be collected
`bit 1` = TX status: '1' when the TX is still transmitting the last byte or '0' when the TX is idle
`bit 2` = RX status: '1' if the buffer is **full** (all 2048B), '0' when the buffer has space to hold
`bit 3` = RX status: '0' if the buffer is **almost empty**, '1' when the buffer includes 256B or more
`bit 4` = RX status: '1' if the buffer is **almost full** (1024B or more), '0' when thas space to hold

Writing to the RX port configures the UART speed (the default speed upon reset is 115200):

```
0 =  115200        8 =   230400        16 =   3456000
1 =   57600        9 =   460800        17 =   4608000
2 =   38400       10 =   576000        18 =   5760000
3 =   31250       11 =   921600        19 =   6912000
4 =   19200       12 =  1152000        20 =   8064000
5 =    9600       13 =  1500000        21 =   9216000
6 =    4800       14 =  2000000        22 =  10368000
7 =    2400       15 =  2304000        23 =  11520000
```

Note: The Wi-Fi specification was compatible with the ZX Spectrum Next (they had changed specifications of the port h133B).

Wi-Fi ESP module

To set up a Wi-Fi ESP module, you can use the Wi-Fi Setup and Terminal applications.
The ESP modules in the MB03+ and the eLeMeNt ZX are different, but the software is 100% compatible.

See also ***ESP8266 AT instruction set*** and ***FPT AT tutorial***
in the eLeMeNt ZX/MB downloads (section Other hardware docs): https://sites.google.com/view/elementzx/docs

Real Time Clock

Batteryless RTC72421

```
3-3843     h0003-h0F03
```

Address (Hex)	A3	A2	A1	A0	Register name	Data				Count (BCD)	Remarks
						D3	D2	D1	D0		
0	0	0	0	0	S1	s8	s4	s2	s1	0 to 9	1-second digit register
1	0	0	0	1	S10	*	s40	s20	s10	0 to 5	10-seconds digit register
2	0	0	1	0	MI1	mi8	mi4	mi2	mi1	0 to 9	1-minute digit register
3	0	0	1	1	MI10	*	mi40	mi20	mi10	0 to 5	10-minute digit register
4	0	1	0	0	H1	h8	h4	h2	h1	0 to 9	1-hour digit register
5	0	1	0	1	H10	*	PM/AM	h20	h10	0 to1 or 2	10-hours digit register
6	0	1	1	0	D1	d8	d4	d2	d1	0 to 9	1-day digit register
7	0	1	1	1	D10	*	*	d20	d10	0 to 3	10-days digit register
8	1	0	0	0	MO1	mo8	mo4	mo2	mo1	0 to 9	1-month digit register
9	1	0	0	1	MO10	*	*	*	mo10	0 to 1	10-months digit register
A	1	0	1	0	Y1	y8	y4	y2	y1	0 to 9	1-year digit register
B	1	0	1	1	Y10	y80	y40	y20	y10	0 to 9	10-years digit register
C	1	1	0	0	W	*	w4	w2	w1	0 to 6	Day-of-the-week register
D	1	1	0	1	CD	30s ADJ	IRQ FLAG	BUSY	HOLD		Control register D
E	1	1	1	0	CE	t1	t0	ITRPT/ STND	MASK		Control register E
F	1	1	1	1	CF	TEST	24/12	STOP	RESET		Control register F

Annex I - Reserved ports

reserved for future use, these ports comes from MB02+ where are not used, but locked by non full decoder

035 **h23** b00100011 - *reserved*
067 **h43** b01000011 - *reserved*
099 **h63** b01100011 - *reserved*
115 **h73** b11100011 - Ultimate memory *reserved*
075 **h4B** b01001011 - FLASH FPGA Core *reserved*
107 **h6B** b01101011 - *reserved (currently used by Datagear, Next)*
055 **h37** b00110111 - *reserved (Next-Kempston Joystick 2, Joystick I/O)*
087 **h57** b01010111 - *reserved (Next-Sprite Attribute Upload)*
119 **h77** b01110111 - *reserved (currently used by ZController)*

Annex II - Source codes examples

HGFX PAL546 viewer

```
;720*546 pixels, 4 colours
;1 byte = 4 pixels
;PICTURE.546 = 98292 bytes
;incl. 12 bytes palette
;indexed,  B-G-R (TrueColour)
;
; gfx mode ZXi_001
; hgfx options ZXi_020
; set hgfx memory registers
; -default at end of zx-screen
; only two memregs used
; -PlanarMask, OffSetY (dest.)
; misc HGFX settings via
; Ultimate RAM ZXi_010
; -linear screen memory
; -no zx-screen transparency
; -border uses indexed colour
; set UltimateRAM mode via $53
; -both 8K URAM Areas writable
; -chunky format(no planar!)
; -Zone0 = 0-16383, this is
; -videobuffer instead of ROM
; load per parts to upper RAM,
; one part is 78 lines * 180 B
; then copy it to videobuffer
; do not calculate line adr
; it is driven by offset Y
; leave dest. adr at 0  :-)
; always from begin. of line
; finally load 4* B-G-R values
; to fill palette indexes
; TrueColour pal. table is by
; default at zx-attributes
; to return to BASIC
; set ZXROM back (URAM off)
; switch off ZXi graphics
;
; for used esxDOS services and
; SD-FAT API please check docs

gzxi    equ $7C3B
ula     equ $FE
uram    equ $53

  ; HGFX Planar Mask
pmask   equ $5B04
  ; HGFX destination OffSetY
offd_y equ $5B07

  di
  ld e, 6      ;HGFX PAL546
  call hgfxpal ; switch on

  ld a, $20    ;HGFX settings
  ld bc, gzxi  ;generalZXi port
  out (c), a   ;gZXi_020
  ld a, 7      ;zx transp. off,
    ;border index., linear mem.
  inc b        ;ld bc,$7D3B
  out (c), a   ;generalZXi data
  ld a, 3      ;now border with
  out (ula), a ; indexed colour

  ld a, 255    ;PlanarMask
  ld (pmask), a ;all bits used

  ld hl, 0     ;OffSetY
  ld (offd_y), hl ;from line 0

  call rom_off ;Ultimate RAM on
  ld a, 131    ;chunky linear +
  out (uram), a ; write enable

  call rom_on  ;ROM for esxDOS

  xor a        ;get drive
  rst 8 : db $89 ;esxDOS call
  ld (defdrive), a ;keep drive

  ld hl, fname ;set filename
  push hl : pop ix
  ld a, (defdrive) ;open file
  ld b, 1  ;file mode (esx API)
  rst 8 : db $9A ;esxDOS call

  jp c, error  ;file is open?
  ld (fhandle), a ;keep fileID

;DATA LOADING per 7+1 parts
  call loader  ;batch of 14040B
  call loader : call loader
  call loader : call loader
  call loader : call loader

  ld a, (fhandle) ;last part
  ld bc, 12    ;4 colours BGR
  ld hl, $5C00 ;HGFX pal table
  call loaddata ;read bytes

  ld a,(fhandle) ;close file
  rst 8 : db $9B ;esxDOS call

  call waitkey
  xor a        ;black border
  jr no_error

  ld a, 2  ;red border warning
no_error
  out (ula), a

  ld e, 0      ;HGFX PAL546 off
  call hgfxpal
  ei : ret     ;RETURN TO BASIC

hgfxpal        ;input: e=0 or 6
  ld a, 1      ;gZXi_001 port
  ld bc, gzxi  ; gfx modes reg
  out (c), a
  ld a, e
  inc b        ;ld bc,$7D3B
  out (c), a   ;gZXi_001 data
  ret

loader         ;load a part
  ld a, (fhandle)
  ld bc, 14040 ;78 * 180
  ld hl, 35000 ;tmp storage
  call loaddata ;read bytes

  call rom_off ;URAM for copy
  ld b, 78     ;78 lines batch
  ld hl, 35000 ;source address

nextline       ;one line copy
  push bc
    ;destination is 1st pixel
  ld de, 0     ;=start of line
  ld bc, 180   ;all 720 pixels
  ldir

;MOVE TO NEXT LINE
  push hl
  ld hl, (offd_y) ;read memreg
  inc hl       ;OffSetY
  ld (offd_y), hl ;increase
  pop hl
  pop bc       ;counter (78)
  djnz nextline  ;last line?

  call rom_on  ;ROM on for next
  ret          ; loading

;READ BYTES from opened file
loaddata
  push hl : pop ix
  rst 8 : db $9D ;esxDOS call
  ret

waitkey        ;press any key
  or a
  in a, (ula)
  cpl : and 31
  jr z, waitkey
  ret

rom_on
  ld a, $10
  ld bc, gzxi
  out (c), a
  xor a        ;Ult. RAM off
  jr rom
rom_off
  ld a, $10
  ld bc, gzxi
  out (c), a
  ld a, 2      ;Ult. RAM on
rom            ;(proteted mode)
  inc b        ;ld bc,$7D3B
  out (c), a
  ret

fname  db "PIC.DAT", 0
fhandle  db 0
defdrive  db 0
```