z26

An Atari 2600 Emulator

v2.13

Table of Contents
v2.13

z26 — An Atari 2600 Emulator (2.13) — May 23, 2004

Home Page: *http://www.whimsey.com/z26* — Contact: z26@whimsey.com

Copyright/License

What's new in version 2.13?

- Converted MASM assembly code to NASM syntax, preparatory to the...
- ...Linux port! z26 can now be compiled for Windows, Linux, or just about any other x86 environment that supports a UNIX-like build environment with GCC. The Linux support is solid; preliminary testing has been done on FreeBSD and BeOS. There are a few Linux-specific command-line options, see below. A Linux static binary is available for those who don't wish to compile z26 from source.
- Improved documentation. We now have a PDF manual and a man page for Linux.
- Added ability to selectively disable graphical objects. This feature is mostly intended for developers.
- We now lock the audio device before filling the sound queue. This should prevent the long startup times that were plaguing some Windows users. It also should allow z26 to run correctly on multi-processor machines.
- Also we made the INTIM timer random at the emulator start again. Berzerk should start with a different screen each time you play it now.
- Other than the audio device locking and the random INTIM timer, version 2.13 for Windows should behave identically to version 2.12. If this isn't the case, **please** let us know.
- Thanks to Brian Watson, Eckhard Stolberg, and James Wilkinson for help with this version!

Introduction

This version of z26 is designed to run natively on 32-bit x86 Windows platforms. It provides excellent sound quality, and correct frame synchronization on Windows-XP platforms.

System Requirements

Windows

Operating System — z26 will run on 32-bit x86 Windows platforms starting with Windows 95, but best results are obtained with Windows 98 or later. For the NT based operating systems, XP or later is recommended. There are sometimes problems with Win2K.

Direct-X — For best results, get the latest and greatest. In particular you may want to be able to run DXDIAG.EXE to adjust the frame rate. This is not supported in older versions of Direct-X.

SDL — You will need a copy of SDL.DLL either in the directory that you place z26, or in the Windows directory. This .DLL is included in the .zip file along with the z26 executable.

Processor — If you have a good quality video card of fairly recent vintage, a Pentium-166 with MMX or a K6-166 is sufficient to run everything we've tried at full speed in a full-screen video mode. It takes more horsepower to run games in a windowed video mode.

Video Card — Some old video cards, like the S3 Trio or the S3 Virge are not as good at page-flipping as more recent video cards. A faster processor can help to make up for the deficiency, but a video card that was designed with Direct-X in mind is desireable.

Sound Card — Even an old Soundblaster 16 works great. The majority of sound cards that have Direct-X support work fine. Occasional problems have been reported with some less popular sound cards like the Lynx One.

Linux

Distribution — z26 is designed to be distribution-neutral. The static binary release doesn't depend on any particular libraries being installed on your system. It has been tested on at least: Slackware 8.0 and 8.1, Mandrake 9.2, Red Hat 9.0, and Knoppix 3.4. You should also be able to build from source on any distribution, provided you have GCC (tested with versions 2.95.3, 3.2, and 3.3.1), NASM (at least version 0.98.33), and SDL (tested with versions 1.2.3 and 1.2.7).

Linux kernel — A 2.6 kernel is recommended, but not required. If you want to use the -T4 (/dev/rtc) accurate timing option, you will need either 2.6 or else 2.4.19 or newer (actually, 2.4.18 and earlier would work, but you'd have to run z26 as root, which is a *Bad Idea*).

Video Driver — Most people will use X11 (XFree86). z26 has also been tested with the Linux fbcon driver. Any display device supported by SDL should be usable, though some may be faster than others. For X11, the XVideo extension is highly recommended. If you experience poor performance in full-screen modes, you will definitely want to enable XVideo or DGA 2.0, if your video card supports them.

SDL — The static binary is linked with SDL 1.2.7, and does not (can not) use any SDL shared libraries installed on your system. If you are building from source, you will need the SDL headers and sdl-config program (most distributions provide this in a package called sdl-dev or sdl-devel).

Processor — If your video card is well-supported under X (or whatever display device SDL is using), you shouldn't need a very fast processor. The slowest machine we've tested is a Celeron/366, which is capable of running z26 at full speed in windowed and fullscreen modes with either -T3 or -T4 timing. Depending on your configuration, full-screen modes may be faster or slower than windowed modes (this is different from the Windows version).

Video Card — See 'Video Driver'. Basically, anything made in the past 5 years should be usable to at least some degree.

Sound Card — Anything supported by the Linux kernel should be OK, however there are issues with certain OSS drivers. If you get no sound, or very quiet sound, you should try ALSA instead (comes with 2.6 kernels, available for 2.4 kernels in most modern distributions). If all else fails, try the -S option: it should make the sound louder, but may cause distortion (or may not, depends on your driver).

Other OS

z26 should build on any OS that runs on the x86 platform and has SDL available.

Under FreeBSD (tested on version 4.9), you can use devel/sdl11, devel/sdl_ldbad, and devel/nasm from the Ports collection, if you want to build a native FreeBSD binary. You could also try to use FreeBSD's Linux binary emulation to run the static Linux binary (I haven't tried this. If you do, let me know if it works or not).

The requirements for other x86 operating systems should be similar to the Linux requirements. Basically, SDL handles the sound and video issues, so if other SDL apps work on your OS, z26 should, too. If the framerate is too fast/slow, try the -T3 timing option.

Since large portions of z26 are written in x86 assembly, you won‘t be able to compile it for non-x86 platforms. Unfortunately, this means no MacOS/OSX (or Sparc/Solaris, or SGI IRIX, or RS/6000...) port for now. We will be rewriting the assembly code in C over the next few releases, so at some point z26 will become truly cross-platform.

Installation

Windows

Unzip the z26 executable into the directory that contains the ROM images that you want to run. If you are uncomfortable using the command line prompt, you should create a shortcut to z26.exe on your desktop. This way you can drag and drop ROM images to the z26 shortcut to run them.
You may also want to run DXDIAG.EXE to force Direct-X to run games at 60fps and set up monitor synchronization by using the -r command line switch. This way subtle graphics effects used by many Atari games will be emulated properly if you‘re running in a full-screen video mode.

Linux

```
***NOTE: Linux binaries are not yet provided
```

Binary installation - Extract the .tar.gz archive and run the **install.sh** script. This will copy the executable and man page to */usr/local/bin* and */usr/local/man/man1*, respectively.
Building from source - To build and install the dynamic executable and man page (most people will want this, and root access is generally required):

```
make linux
make install
```

To build a dynamically-linked executable, but not install it in */usr/local/bin*, use the command:

```
make linux
```

To build a statically-linked executable:

```
make linux-static
```

To build the text, manpage, and PDF versions of the manual:

```
make docs
```

If you‘re using a desktop environment such as KDE or Gnome, you might want to copy the **z26_icon.png** to your icons directory and set it as the icon for the z26 executable.

Startup

If you have created a shortcut to z26 on your desktop, just drag and drop a ROM image from the Windows explorer *or equivalent, on Linux* onto the shortcut. The game will then run. Otherwise you can run z26 from the command line prompt. Type

```
z26 filename
```

where filename is the name of a "standard" Atari 2600 .bin file. For example:

```
z26 demonatk.bin
```

will run Demon Attack. If you don‘t include the .bin extension, z26 will automatically append one for you. So you can also type:

```
z26 demonatk
```

to run Demon Attack.

You may also include a variety of command line switches to modify the behavior of z26. These are described later.

Running z26

Once you are running a game, the keyboard controls things as follows. You may find more information about these things later in the document.

Console Controls

```
F1 -- Reset
F2 -- Select
F3 -- B/W
F4 -- Color
F5 -- P0 easy
F6 -- P0 hard
F7 -- P1 easy
F8 -- P1 hard
```

Paddles

```
  Ctrl -- fire paddle 0   Use left and right arrows to move
RShift -- fire paddle 1   Use up and down arrows to move
     n -- fire paddle 2   Use 's' and 'f' to move
     v -- fire paddle 3   Use 'e' and 'd' to move
```

Player 0 Joystick

```
  Ctrl -- fire  Use the arrow keys to move.
     / -- trigger (booster grip)
RShift -- booster (booster grip)

The mouse may also be used to control the Player 0 joystick.
```

Player 1 Joystick

```
n -- fire
s -- left
e -- up
f -- right
d -- down
b -- trigger (booster grip)
v -- booster (booster grip)
```

Player 0 Driving Controller

```
    Ctrl -- accelerate
right arrow -- turn clockwise
 left arrow -- turn counter-clockwise
```

Player 1 Driving Controller

```
n -- accelerate
f -- turn clockwise
s -- turn counter-clockwise
```

Keyboard Controllers

```
    Left Port                    Right Port

7 -- 1   8 -- 2   9 -- 3     1 -- 1   2 -- 2   3 -- 3
```

```
u -- 4   i -- 5   o -- 6      q -- 4   w -- 5   e -- 6
j -- 7   k -- 8   l -- 9      a -- 7   s -- 8   d -- 9
m -- *   , -- 0   . -- #      z -- *   x -- 0   c -- #
```

Compumate Keyboard

```
Use PC keyboard columns 1 - 0
  Ctrl -- FUNC
LShift -- SHIFT
```

Mindlink Controller

```
Use the mouse to move horizontally.

button -- start the game
   Tab -- switch between player 0 and player 1
```

Mouse Handling in Paddle emulation

```
   Tab -- switch between  horiz. normal, vert. normal
          and  horiz. reversed, and vert reversed
```

Kid Vid Controller

```
Use keys  1  2  3  to start the corresponding tape.
F1 stops and rewinds the tape.
```

Trak-Ball Controllers

```
The PC-mouse works just like the Atari Trak-Ball in Trak-Ball mode.
```

Pausing a Game

```
Backspace -- pause
    Enter -- resume
```

Video Modes

```
 Alt-0 -- Mode 0
 Alt-1 -- Mode 1
 Alt-2 -- Mode 2
 Alt-3 -- Mode 3
 Alt-4 -- Mode 4
 Alt-5 -- Mode 5
 Alt-6 -- Mode 6
 Alt-7 -- Mode 7
 Alt-8 -- Mode 8

Alt-Enter -- switch between full-screen and windowed video modes
```

Scrolling the Screen

```
  PgUp -- scroll the screen up
  PgDn -- scroll the screen down
  Home -- return to the default screen position
```

Screen Capture

```
     = -- capture screen to PCX file
```

Trace Mode

```
   F11 -- resume tracing
   F12 -- pause tracing
```

Palette Change

```
- -- switch between NTSC -> PAL -> SECAM palettes
```

Configuring z26

z26 has many options that are set with command line switches. You can use a text editor like Notepad to create a file called z26.cli in the directory that contains your ROM images to set these options. Otherwise if you type z26 with some command line switches and don‘t include a filename, the command line switches are recorded in z26.cli automatically as emulator defaults. These defaults can still be overridden with command line switches that you include on command lines that include a filename.
Note that spaces are **not allowed** between an option and its argument. Example: to force the framerate to 60, use **-r60**, not **-r 60**.

Command Line Switch Summary

More detailed information about the meaning of the switches may be found later in the document.

```
-)CC -- select controller on left port
        CC = JS -- joystick
        CC = PC -- paddle controller
        CC = KP -- keypad
        CC = DC -- driving controller
        CC = LG -- lightgun
        CC = CM -- CompuMate keyboard
        CC = KV -- Kid Vid tape player
        CC = ML -- Mindlink controller
        CC = ST -- ST mouse / CX-80 Trak-Ball
        CC = TB -- CX-22 Trak-Ball
        CC = AM -- Amiga mouse
        CC = NC -- no controller connnected

-(CC -- select controller on right port (CC: see above)

  -! -- run interlaced games

  -0 -- start with player 0 hard

  -1 -- start with player 1 hard

  -4 -- allow all 4 directions on the joystick to be pressed
        simultaneously

  -b -- start in black and white

  -B -- enable screen captures to .bmp files.  Works with
        phosphorescent games.

 -cN -- color palette (N=0 -- NTSC  N=1 -- PAL  N=2 -- SECAM)

 -dN -- DSP processing level (N=1 -- low  N=2 -- high)

 -eN -- enable narrow video modes (N=1 or N=2)

 -fN -- enable phosphorescent effect (N=0 through N=100,
        77=default) The parameter is the frame mixing
        coefficient (0 through 100) which specifies how much
        the bright pixel is favored over the dim pixel when
```

```
          mixing the value of two frames. The default is 77. If
          you specify a value greater than 100 or less than zero,
          the effect is disabled for games that are automatically
          recognized as needing this effect, like Yars Revenge.
          If N<0 or N>100 the effect is disabled for games that
          are automatically recognized as needing this effect like
          Yars Revenge.

   -gN -- override game type
          N=1  -- Commavid extra RAM
          N=2  -- 8K Superchip
          N=3  -- 8K Parker Brothers
          N=4  -- 8K Tigervision
          N=5  -- 8K Decathlon & Robot Tank
          N=6  -- 16K Superchip
          N=7  -- 16K M-Network
          N=8  -- 32K Superchip
          N=9  -- 8K Atari -- banks swapped
          N=10 -- Spectravideo Compumate
          N=11 -- 32K - 512K Extended Tigervision
          N=12 -- 8K UA Ltd.
          N=13 -- 64K Homestar Runner / Paul Slocum

   -hN -- set the maximum number of scanlines to render (height)

   -iC -- [changed] inactivate PC-controller
          (C = K -- keyboard, M -- mouse, J -- joystick,
           S -- Stelladaptor)

   -jN -- [removed]

   -kN -- [removed]

   -pN -- paddle sensitivity (N=1 to 15 -- keyboard only)

   -PN -- enable lower sesitivity for PC joysticks / Stelladaptor
          in paddle emulation, and delay the paddle read-bit
          flipping by N scanlines

   -mN -- paddle to emulate with mouse (N=0 to 3)

  -m1XY -- emulate two paddles with mouse (X and Y = 0 to 3)

   -MN -- enable mouse capture in a window -- allows mouse full
          control of games in windowed video modes just like full
          screen modes  (N=0 -- off, N=1 -- on)  default = on

    -G -- Grab all mouse and keyboard events. Use this if your window
          manager's hotkeys conflict with z26 hotkeys. This option
          is primarily intended for Linux/UNIX systems, but may be
          useful on Windows as well.

   -lN -- [changed] adjust lightgun by N cycles
```

```
 -aN -- adjust lightgun emulation by N scanlines

  -n -- show scanline count and FPS on game display

  -o -- simulate PAL color loss

  -q -- quiet

 -sN -- specifies the size of the sound queue
       (default = 4608   --   max = 65536)

  -r -- Windows: run at monitor speed
        Linux:   This option doesn't work, and is disabled.

 -rN -- run at N frames per second. Give a large number
        (e.g. -r10000) to run the emulator as fast as possible.
        This option does work on Linux.

  -R -- Windows: synchronize with sound instead of display
        Linux:   This option doesn't yet work correctly,
                 and is disabled.

  -t -- write code trace to z26.log. Warning: These log files
        get ENORMOUSLY large. Try not to run out of disk space.

  -tt -- Same as -t, but start emulator with tracing disabled.
         Press F11 to enable trace while emulator is running,
         or F12 to disable.

 -uN -- start scanning game at line N

 -vN -- start game in video mode N full screen

-v1N -- start game in video mode N in a window

  -w -- [changed] swap Atari controller ports for all
        controller types

  -x -- print checksum of cartridge

 -yN -- [removed]

  -z -- disable the fast copy routines
```

Linux (and others) options

```
  -S -- OSS Sound hack.

 -TN -- Select timing mode for frame sync
        N=1 -- Use SDL for timing. Runs too slow on
               Linux 2.4 kernels, too fast on Linux 2.6.
        N=2 -- Use UNIX select() call for timing. More
               accurate than SDL, but may oscillate between
               too slow and too fast on slower systems.
        N=3 -- Use 'CPU Hog' timing. z26 will busy-wait in
```

```
            a loop polling the system clock until it's
            time to start the next frame. This is very
            accurate, but not very friendly to other
            processes on your system.
      N=4 -- Use /dev/rtc timing (Linux only). This is the
            most accurate timing mode available, but
            requires you to configure your RTC driver to
            allow z26 to set the correct clock frequency.
            As root, run the command:

            echo "8192" > /proc/sys/dev/rtc/max-user-freq

            You should add this command to your rc.local
            script (or equivalent).
```

Atari Controller Support

Joystick

PC or USB joysticks and gamepads are supported to the extent that Direct-X and SDL supports them for playing joystick games. The mouse and PC keyboard may also be used. Some games like raiders.bin use the "wrong" joystick. To reverse the joysticks use the -w command line switch.

Trak-Ball

Atari sold a Trak-Ball that could be used to play joystick games. z26 supports the use of a mouse or PC trakball to do the same thing for joystick games. If you have a mouse or PC trackball, support is automatic. Optical trackballs like the Logitech Marble Mouse are great for this. Also z26 now emulates the various types of Atari Trak-Balls in their native modes. This is done with the PC mouse as well.

Paddles

z26 supports the use of the mouse or PC trackball to play paddle games. If you want to change which paddle is emulated with the mouse use the -mN command line switch. You can change which direction the mouse must be moved in to get paddle motion by pressing the <Tab> key on the keyboard while you‘re playing a game. You can emulate two paddles with a mouse with one paddle on each axis. Use the -m1xy command line switch where x and y are the paddle numbers to emulate on that axis. You can use the keyboard to play paddle games. If z26 doesn‘t automatically support a game as a paddle game you must specify the -)PC or -(PC command line switches to get paddle support on the left or right Atari port.

Light Gun

z26 supports light gun games (Sentinel, Shooting Gallery) with the mouse.

Keyboard Controllers

z26 enables the keyboard/touch pad/keypad for recognized games that use them.

Driving Controllers

z26 suppports the game that uses the driving controller (Indy 500) from the keyboard.

Booster Grip

z26 supports the booster grip game (Omega Race) from the keyboard and the PC joystick with 3 or more buttons.

Kid Vid Tapes

z26 supports the Kid Vid games (Smurfs Save the Day and Berenstain Bears) automatically. Press 1, 2, or 3 to select the corresponding tape. F1 rewinds the tape. In addition to the rom images, you need the wave files with the music in the same directory as z26. The files are named:

Berenstain Bears:

```
KVB1.WAV,  KVB2.WAV,  KVB3.WAV
```

Smurfs Save the Day:
`KVS1.WAV, KVS2.WAV, KVS3.WAV`

Both:
`KVSHARED.WAV`

You can also play the games without the music, but it‘s much less fun.

Stelladaptor support

z26 assumes that a PC joystick with only two buttons and two axis is the Stelladaptor device, that lets you connect Atari joysticks, driving controllers and paddles to your PC‘s USB port. If you want to use a different 2-button/2-axis joystick with z26 and experience problems with the emulation of some Atari controller, you can disable the Stelladaptor support with the -iS command line switch. Please note that the Stelladaptor is seen as a joystick too, so disabling joysticks with the -iJ switch will also disable the Stelladaptor.

Inactivating PC controller support

Some Atari controllers are simulated with more than one PC controller. If you are experiencing problems with this, you can disable one or more PC controllers for these Atari controller simulations. For example you might want to disable the PC mouse when you use the keyboard to play an Atari-joystick game, so that accidently pushing the mouse doesn‘t cause any unwanted movement. Also you might want to disable the PC joystick for paddle games, because the PC joystick always overrides all other PC controllers in the paddle simulation when one is plugged into the PC. Please look at the following table to find out which Atari controllers are simulated with which PC controller.

Atari/PC		keyboard	mouse	joystick	Stelladaptor
joystick	JS	x	x	x	x
paddle	PC	x	x	x	x
keypad	KP	x	-	-	-
driving	DC	x	x	x	x
lightgun	LG	-	x	-	-
Compumate	CM	x	-	-	-
Kid Vid	KV	x	-	-	-
Mindlink	ML	-	x	-	-
ST mouse	ST	-	x	-	-
Trak-Ball	TB	-	x	-	-
Amiga mouse	AM	-	x	-	-

Please note that inactivating the PC controller only works for Atari controllers that are simulated by more than one PC controller. Also note that the Stelladaptor is just a special case of the PC joystick. So inactivating joystick support will also disable Stelladaptor support.

Video Modes

Note that by adding 10 to the video mode number z26 will run in a windowed mode. Otherwise z26 runs full-screen. On some systems, windowed modes are very slow. So for maximum performance you should run full-screen. Adding 20 to the video mode number causes the game to run full-screen with the number of bits per pixel as your current desktop. This allows the phosphorescent effect to work in full-screen video modes.

Mode 0 — 400x300 (default)
Mode 1 — 320x240

Mode 2 — 320x200
Mode 3 — 800x600 scanline/interlaced
Mode 4 — 640x480 scanline/interlaced
Mode 5 — 640x400 scanline/interlaced
Mode 6 — 800x600 double scanline
Mode 7 — 640x480 double scanline
Mode 8 — 640x400 double scanline

Modes 3-5 simulate the TV scanline effect when the -! command line switch is not specified. If the -! command line switch is specified, they run games interlaced by updating every other scanline every other frame. They also play the interlaced demos like cube_imp.bin correctly.

Modes 6-8 appear similar to modes 0-2 except that they are larger.

Switching Video Modes on the Fly

You may change video modes while playing a game by pressing one of the number keys on the keyboard (not the keypad). Pressing <Alt-0> through <Alt-8> is supported. Pressing <Alt-Enter> will switch between windowed and full-screen modes.

Adjusting Vertical Position

You can scroll games up and down on the display with the <PgUp> and <PgDn> keys. You can return to the default display position by pressing the <Home> key. You can force the game to start displaying from a particular scan line from the command line with the -uN command line option. You can set the maximum number of scanlines to render with the -hN command line switch.

Note that the <PgUp> and <PgDn> scrolling doesn‘t work for traffic.bin or pharhcrs.bin.

Forcing Black and White mode

You can force a game to start in black and white mode with the -b command line switch. Note that this doesn‘t mean that the game will actually be in black and white, this is up to the game itself to support.

Colors

z26 supports three different sets of colors. The most common games shipped in North America use the so called NTSC colors. Games shipped in other parts of the world use the so called PAL colors. z26 typically detects the correct colors to use based on which scan line a game starts displaying at. If the display starts at scan line 54 or higher, z26 uses PAL colors, otherwise NTSC colors are used. There are some exceptions, but I‘m not an expert on which games are PAL and which ones are NTSC so you can override the colors chosen by z26 with the -cN command line switch. -c0 forces NTSC, and -c1 forces PAL. Feel free to let me know if you find a game that z26 picks the wrong colors for.

z26 also supports the bug in PAL consoles that causes the display to display a monochrome image if a game has the wrong number of scanlines. This feature is enabled with the -o command line switch. Some of the 32-in-1 games which were poorly done PAL hacks show this effect.

SECAM colors can be selected with -c2. In the time before the crash Atari created most of it‘s PAL games in such a way that the B/W switch would enable SECAM compatible colors. So nostalgic gamers from France can play PAL ROMs with -b and -c2 command line options enabled.

Adjusting Game Speed

By default, games run at a speed that is determined by how many scanlines the game generates per frame. This provides correct play speed for NTSC and PAL games automatically. If you want to synchronize games with the monitor so that games like Yars Revenge play with a minimum of visual

artifacts use the -r command line switch. You may want to run DXDIAG.EXE to force Direct-X to run video modes at 60fps which is the normal speed for NTSC games. Note that monitor synchronization only works in full screen modes. Windowed modes fall back to the normal synchronization method.

You may also specify a parameter to the -r command line switch to choose a non-standard play speed. This works both in windowed and full screen modes.

Measuring Emulator Performance

The -n command line switch causes the number of frames per second to be displayed during game play. If you choose the -r command line switch with a large number the emulator will run at full speed.

Digital Signal Processing

z26 supports digital signal processing (DSP) on the sound. This is to reduce distortion on some clone sound cards, but also causes the sound to more closely approximate the sound of an old TV set for those users that prefer a more "classic" sound. To turn on DSP use the -dN command line switch. If n=1 the setting is low, if n=2 the setting is high.

Fixing certain Linux sound problems

With certain sound hardware and drivers, the sound is missing or very quiet on the Linux platform.

This can usually be corrected by using the ALSA sound drivers instead of the deprecated (but still very common) OSS drivers. If that doesn‘t help, the -S option might.

This option increases the sound volume, but may cause distortion. Use it as a last resort. In future versions of z26, this option will change or go away entirely.

Offbeat Cartridge Formats (Bank Switching)

z26 currently supports the following schemes:

1. Standard 8K bank switching.
2. Standard 16K bank switching.
3. Atari Super-Chip.
4. CBS Ram Plus.
5. Parker Brothers 8K.
6. TigerVision 8K.
7. Activision FE 8K.
8. Atari 32K.
9. M-Network 16K.
10. Pitfall II.
11. Starpath.
12. Megaboy.
13. Compumate.
14. Extended TigerVision (to 512K).
15. UA Ltd. 8K.
16. ‘EF’ 64K (used by Paul Slocum‘s Homestar Runner RPG.

Support is automatic. z26 knows how to recognize cartridges. To override the default selection you may use the -gN command line switch.

For most Starpath games, the .bin file must be a multiple of 8448 bytes big. To run multiload games you must concatenate the .bin files. A command like this will do the trick:

DOS:
```
copy /b load1.bin+load2.bin+load3.bin+load4.bin mload.bin
```

UNIX:
```
cat load1.bin load2.bin load3.bin load4.bin > mload.bin
```

Then run mload.bin in the emulator. z26 supports a maximum of 61 loads.

z26 also supports the 6144 byte Starpath format.

The Extended Tigervision (-g11) bankswitch scheme allows ROMs up to 512K in size keeping the last bank fixed. All ROM images that are bigger than 64K and not a multiple of 8448 default to this scheme.

The original Tigervision (-g4) bankswitch scheme also supports up to 512K games but with $1800-$1FFF as the fixed bank. This allows compatibility with the Cuttle Cart for games up to 64K in size.

PCX Screen Capture

Pressing the = key during game play causes a .pcx file with a screen image to be placed in your directory. Filenames are z26p0000.pcx, z26p0001.pcx etc. The filename restarts at z26p0000.pcx causing old versions to be replaced each time you restart the emulator.

Features for Game Developers

z26 supports some features to help game developers with their projects. Trace mode, and the ability to count scanlines.

Also, z26 allows game developers to disable individual graphics objects. Remember, this is a feature for game developers, not for dirty cheaters :)

Trace Mode

If you start z26 from the command line with the -t command line switch, it automatically builds a text file called z26.log. This file is a log of all instructions executed and shows some other interesting things like the (frame number, scanline number, cpu cycle, tia clock), object positions (P0, P1, M0, M1, BL), cpu flags, registers (A, X, Y, SP), the program counter, the current instruction in hex, and a disassembled listing of the instruction.

Emulator performance slows down quite a bit while writing out this file so you can turn off writing with F12 and turn it back on with F11 to make it easier to get to the part of the game that you‘re interested in.

Counting Scan Lines

Choosing the -n command line switch causes the number of scan lines that a game is using to be displayed. This is useful for testing PAL games so you can avoid the PAL color loss bug. Of course you can also enable the color loss by using the -o command line switch described above under Colors.

Disabling Graphics

While the emulator is running, use the following keys:

```
Alt+key  Effect
   Z     Enable/disable player 0
   X     Enable/disable player 1
```

```
C     Enable/disable missile 0
V     Enable/disable missile 0
B     Enable/disable ball
N     Enable/disable playfield
/     Enable all of the above (returns to default)
```

There is currently no way to disable individual playfield registers (PF0, PF1, PF2). Alt-N affects all 3 registers.

This feature can be useful for testing a game you‘re writing, or for analyzing a game someone else wrote. While a register is disabled, it‘s invisible and unable to collide with other objects. This means you can use it to cheat, too... but you wouldn‘t do that, would you?

Troubleshooting

Problem:

z26 runs too slowly

Solutions:

```
- Try a different video mode
```

Warranty

BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Tools

The Windows version of z26 is built with MinGW/Msys, with GCC and NASM.

The Linux static binary version of z26 is built with Slackware 8.1.

The documentation that you‘re reading now was generated from POD source, using either pod2pdf (slightly modified), pod2text, or pod2man, depending which format you‘re looking at. All of these are written in Perl.

Fonts were designed with CSEDIT by Matt Pritchard.

Credits

Eckhard Stolberg has made contributions to z26 that are almost too numerous to count. He has been involved since the very early days and the high quality emulation that z26 provides would certainly not have been possible without his participation.

```
      Paul Robson -- Wrote original A26.
        Ron Fries -- Original TIA sound emulation.
 Ettore Perazzoli -- Helped with 6502 undocumented instructions.
    Andreas Boose -- Also helped with 6502 undocumented
                     instructions.
 Wolfgang Lorentz -- 6502 diagnostics.
      Bob Colbert -- Helped with object wrapping.
     Piero Cavina -- Helped with multiple missiles.
      Erik Mooney -- Helped with HMOVE blanks.
     Kevin Horton -- Helped with bankswitching and
                     Quadrun voice.
        Dan Boris -- Atari 2600 schematics.
   Matt Pritchard -- Font design tools.
       Matt Conte -- Helped with Quadrun voice.
      John Dullea -- Fast elegant graphics and helped with
                     Pitfall II.
    Bradford Mott -- Helped with object positioning
                     (weird HMOVE).
    Chris Wilkson -- Helped with Pitfall II.
      Lee Krueger -- Helped with Kid Vid support, rare carts,
                     and documentation.
  Thomas Jentzsch -- Helped with trace mode, Kid Vid support
                     and some fast video routines.
  Henning Mueller -- Helped with CompuMate.
  Christian Bogey -- Helped with SECAM colors.
    Oliver Achten -- Helped with PAL colors.
     Andrew Davie -- Provided Extended Tigervision bankswitch
                     demos.
      Paul Slocum -- Provided EF bankswitch demo.
        Billy Eno -- Provided interleaved display demo.
     Adam Wozniac -- Helped with TIA sound polynomials.
    Matt Matthews -- Linux consulting.
     Brian Watson -- Helped with Linux port, ASM-code and
                     documentation conversion.
  James Wilkinson -- Helped with BeOS port.
```

Thanks to everyone else who has helped with comments, suggestions, bug reports, information, supplies and testing:
Junky, Kevin White, The Boatwrights, Miguel Guzman Centeno, Peter, Jay C. Heil, Zoop, Rob Anderson, Zophar, Peter Vogels, Stephan, Xahji, Ricardo Martinez Garza, Stephano Camarri, Peter Betz, Piero Cavina (Oystron! Helped find multiple missile bug.), Nick S. Bensema, Erik Mooney, Glenn Saunders, Abraham Velazquez, Christian Schaefers (Z-Type), Freirias, Brian Deuel, Chris Platt, Israel Geron T., jose roberto rodrigues, Dan Meyer, Martin Schaefer, Ummagumma, Digitoxin, Michael Walden Jr. (The most elaborate suggestion list!), Gilamonster, Gerald Gorman, Francisco Athens, Lex Nesta, Ben, Gerald Gray, Jose Pedro, Tadd Underhill, Ejber Ozkan, Lord Mhath, Larry Scott, Brad Komgenick, Michael J. Mika, Ettore Perazzoli (VICE), Stephan Eder, Andreas Boose (VICE), David Gray, jimnav, Dr. Simone Zanella, Brad Thomas, Jeff Cockayne, Sam Miller, EmrldSword, justin martin, Justin Scott, Jason Berk, Luis Graterol, Ricardo Soto, Brian Smith, Sam Hard, Keith Merizalde, Nate Marigoni, Kurt Woloch, GreenImp, M. C. Silvius, Matthew Conte (Nofrendo), Jason Barisoff, Rick Vasquez, Marco Turconi, Kalik, Christopher Warren, Curt Vendel, Roberto Sidney Teixeira,

raverpup, iCeFiRe, Chism, Sascha Reuter, Craig Tildesley, Michael Prentler, Cody B, Thom Rechak, Cam, Jason, Keith Weisshar, Robin Gravel, Thom Bone, Mrfriend, Edwin Helsloot, C J Biro, Michael Monson, Justin Rodriguez, Francesco Azzurri, Greg Bendokus, Bruce Clarke, John Dullea (PCAE), Tim Boston, Chris Ainsley, Ricardo Henrique Tabone, and Seth Kintigh.

And thanks to Len Shikowitz for *complaining* well beyond the call of duty. Definitely "the most obstinate, yet sincere, tester." :)

At some point I stopped keeping the list of everyone who have sent me suggestions up to date. I apologize for that. But thanks to the folks who have continued to send in suggestions, even though they didn‘t make the list.

Thanks!!!