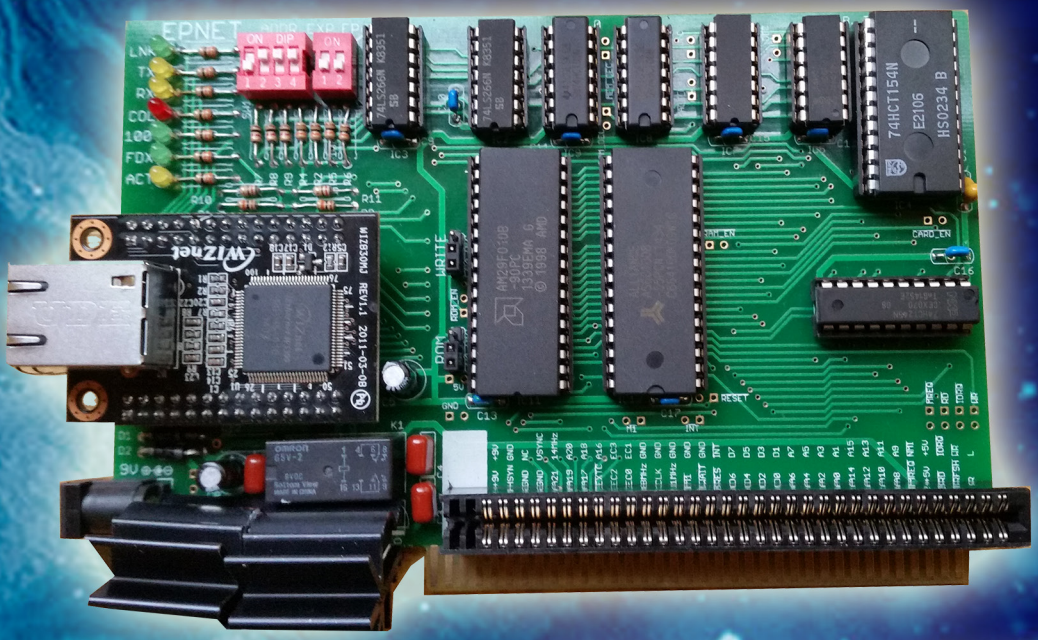
ENTERPRESS

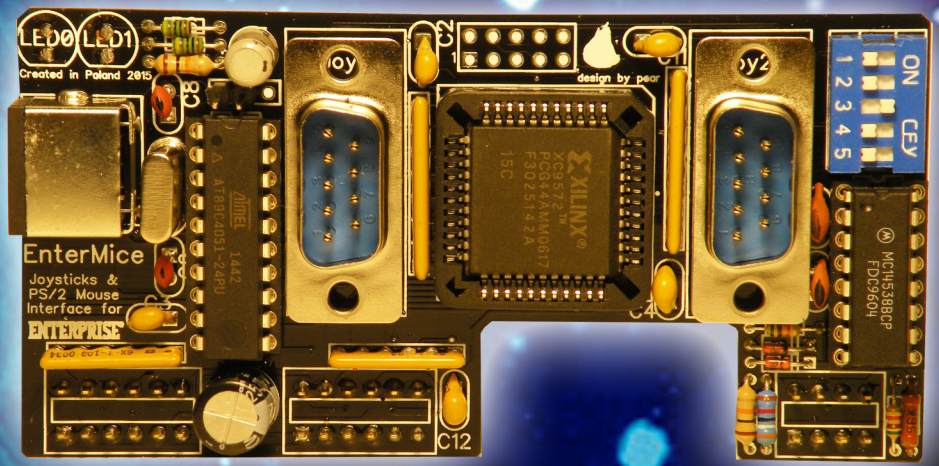
Magazine for Enterprise users

November 2016



EPNET CARD

(by Bruce Tanner)



EnterMice

(by Maciej Gruszecki)

10 Mhz on the Enterprise

(by Zozosoft)

Infinite zest

Author: Matusa István (Tutus)

No, it's not a delusion, Enterpress Magazine is here again!

The continuation of the history of the magazine is interesting. Zozo called me in May 2015 that there is going to be a jubilee club meeting. And it's not just any kind of club meeting: the Enterprise had its 30th birthday! For this occasion, foreign quests were invited: **Bruce Tanner,** one of the software developers at Enterprise and **Jörn Mika** from Germany, who is the creator of the SymbOS operating system. To tell the truth, the last time I was at the Enterprise Forever Forum before the jubilee meeting was in 2008 which means several years of hiatus. The turn up of Zozo and the birthday event made me quite excited. I figured we quickly had to make (Zozo and I had like a week) a jubilee Enterpress magazine for the meeting. Although we were in a hurry we succeeded and the printed magazine was present at the club meeting, even with a multicolour cover. After this the zest didn't diminish, I decided to digitize older issues of the Enterpress Magazine into electronic format. It was a mouthful, so far the first two issues have been published this way. I've been constantly working on digitizing the remaining issues as well. The magazine needed a website too, which came into existence at http://enterpress.news.hu. Naturally, I believed that new Enterpress Magazines will be published, periodically, in a limited number of printed copies. And in an electronic format as well which has the advantages of embedding links and videos into the articles and there is no impediment to it having more pages than the printed version as well as being able to search in the content. When more, older magazines will be online they could be organized into an archive and you could search in the whole archive as well (cross-search). But let's talk about all the changes that occurred during the years I was absent. Truth be told, Zozo trying to make me up for all the missed years made my head spinning... Fortunately I can look back on our Facebook chats :).

- **István Varga's (IstvanV)** fantastic PC emulator, video and music converter programmes,
- the just wen on-going EnterMice project, conceived by **Pear** from Poland, due to the persuasion of the Spanish guy gflorez,
- **Prodatron's** SymbOS graphical operating system, developed on CPC, runs on Enterprise engine too
- **Saint,** in the colours of the UK, designed a 1 MB memory expansion
- and the biggest hit was **SzörG's** SD card which substitutes the use of cassette machine, floppy and Winchester. This small card can be inserted into the palce of the left cartridge
- **Persa Noel's (geco)** CPC, Spectrum and TVC emulators on Enterprise
- the continuous works of **Zozo** on the EXDOS and EXOS systems, and he is the only one who supplies the home and foreign service network. By the way, if someone can arrange him at least 36 hours a day deserves a special award! :)

But let's go on with the current developments:

- **Bruce Tanner's** EPNET card which allows us to connect to the internet and with the proper programmes the following services will be available: FTP, Chat or even text-based browser,
- **Zozosoft's** FAT16 version of EXDOS, internal RAM/Flash/Turbo card
- **Gábor Lénárt's (lgb)** XEP128 EP emulator for PC, Linux and Mac OSX
- **Pear's** 3 cards, FlexiBridge, IDE-Compact Flash, RAM-Flash-Clock.
- In the meantime, the **Enterprise Dev-Compo #**1 programming contest was held for which many great programmes were created.
- Since then, **Povi's** 2048 game for Enterprise was born as well as the super demo of rookie Swd who bought his computer not long ago.

Our plans included the revival of the Budapest Enterprise Club. This seemed to be a more difficult task to do. Our age group is around 40-50, and, obviously and understandably, the family, our jobs and other problems can butt in. Nonetheless, from November 2016 on we hope to meet every month again and invite foreign guests again for our next year club meeting. Detailed information about the Club will be up on http://www.entrpriseklub.hu which will be launched on 1st January 2017 along with a club membership system, including 3 different subscriptions.

If the small community of **Enterprise Forever** could maintain this lively pace, the owners of Enterprise computers will share great joy!

If you'd like to support the issue of the ENTERPRESS Magazine, you can do it here:

http://enterpress.news.hu/donate/

EPNET

Author: Bruce Tanner (BruceTanner)

Background

30 years ago I worked for **Intelligent Software Ltd.** in England, the design company used by Enterprise Computers, where I wrote IS-BASIC, IS-FORTH and half of EXDOS and IS-DOS. Following the well-documented collapse of IS and Enterprise Computers, **Robert Madge,** myself and three colleagues from IS (one of them **Martin Lea,** who wrote EXOS and the other half of EXDOS and IS-DOS) set up **Madge Networks,** and I spent the next decade working on various networking products involving everything from low-level assembler software to Windows applications.

We barely gave the Enterprise a second thought – as far as we knew, it was dead, and we were working full-speed on new and exciting products in emerging technologies: networking and PCs.

25 years later I happened to see an Enterprise Computer for sale on eBay, and it cost more than it did 25 years earlier! It did not take me long to acquire one myself and to discover the Enterprise Forever forum. Most astonishing to me was that 25 years of Enterprise history had passed in Hungary and I was completely unaware!

The very first thing I had an urge to do with my "new" Enterprise was to download some software for it. In fact I suddenly realised that without a tape drive, which I did not have, there was no way of getting any software on to it at all.

Given my background (first Enterprise, then networks) I seemed the natural person to design a network card. Co-incidentally I was also converting my garage at home into an electronics workshop, something I had wanted to do for many years. So that was looking for a first project...

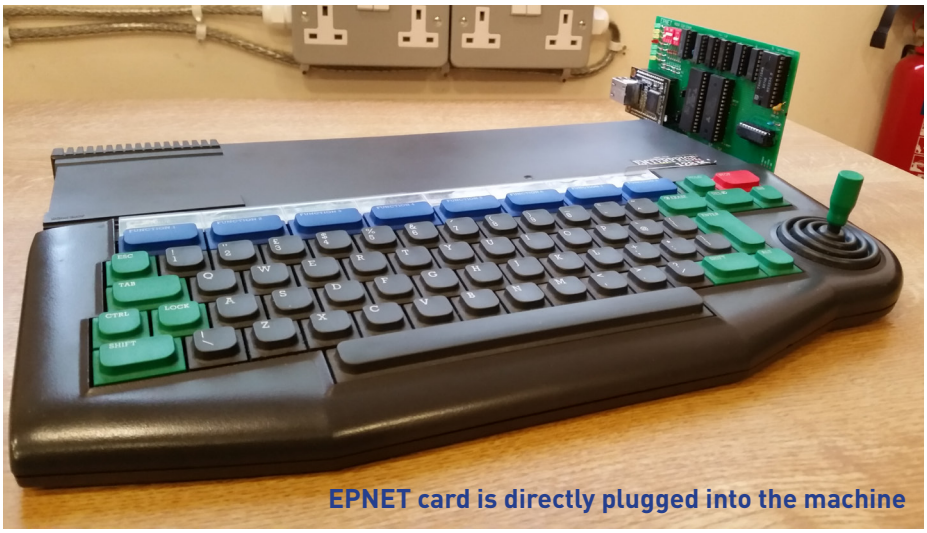
Design

Every hardware design involves a lot of trade-offs and compromises. The first and most important for me was that I did not "bite off more than I could chew" so the project stood a good chance of working and actually getting finished! I am not a hardware designer by profession, although I have spent all my professional life working closely with hardware (writing software drivers etc) and being involved with many of the issues surrounding putting a new hardware design into production. As a teenager I designed and etched my own PCBs but nothing like the complexity that was going to be needed here. I had seen CAD programs in use but not actually used one myself, so there was quite a steep learning curve ahead just for that.

Another self-imposed design constraint comes from my feeling a bit uneasy about throwing too much modern hi-tech at the Z80. Processors these days typically run at 1 or 2 GHz instead of the Z80's 4MHz – nearly 1000 times faster. In 1984 64k was a lot of memory; now even the phone in my pocket has 16Gb – over 100,000 times more!

A third and related constraint comes from the fact that this was going to be a hobby-level project. I wanted it to be reasonably easily solderable, not full of tiny surface-mount components that could only be soldered with temperature-controller hot air ovens and other specialised equipment.

So I decided to base the design very loosely on the EXDOS card: it contains address decoding logic and ROM, and drives a chip that does the complicated job of accessing the disk drive. My card would contain similar address decoding logic, ROM (and RAM) and a chip or module that does the complicated job of accessing the network



EPNET card is directly plugged into the machine

A company called WIZNet make modules that take care of the lower levels of interfacing to the network. The modules contain the physical network socket and associated electronics, and a chip that contains a processor, RAM, and software in ROM that handles the lower-level protocols such as

ARP, ICMP, IP and TCP. Furthermore the modules can be directly attached to the Z80 address and data buses and can be memory mapped or I/O mapped.

They seemed ideal! I had the choice of an older module that contains their w5100 8-bit chip with 32k RAM, or a newer version that contains their w5300 16-bit chip with 128k RAM. I was afraid the earlier module might be about to go out of production (it has been around for nearly 10years), and as the later one was more readily available in the UK and Hungary as well as being a bit cheaper, it was an easy choice. There was one downside: they did not support wi-fi. This was one of those design compromises that I decided I could live with and hope I don't come to regret!

The design would also need some RAM, some ROM (FLASH) for program storage, some LEDs to indicate the network link state and miscellaneous glue logic to connect it all together.

One tricky design goal was that it should be useable on a fully expanded system as well as an unmodified, unexpanded Enterprise 64, and I settled on having a vertical PCB with edge connector fingers on the bottom for use in an expansion bus but also with an edge connector socket just above. Thus EPNET could be used either plugged directly into the side of the Enterprise (unusually, standing upright) or plugged into an expansion bus.

On the software side, my principal goal was to be able to load and save programs over the network. Of course it would be nice if other things could be implemented, eg. read email or even display a text-only web page.

Current State

I designed a PCB using Eagle CAD and had 5 made in China at Seeed Studio for prototypes. At the moment I have built one and have written about 12k of software (all assembler). I plan to build the other four and send them to various people wider testing and software development (eg. SymbOS).

The software has three different EXOS devices using different protocols: FTP:, HTTP: and TCP:

The FTP: device allows files and programs to be read and written to an FTP server, and a series of :FTP commands provide full file control eg. :FTP DIR prints a directory listing, :FTP DEL deletes a file, :FTP MD makes a directory, :FTP CD changes the current directory etc. It is easy to run a (free) FTP server on a desktop PC under Windows or Linux and this is probably the easiest way to load and save programs over a LAN.

The HTTP: device allows reading of files from a HTTP (web) server and is probably the best way to make a series of Enterprise programs available to other people. Again it is easy to obtain a free web server to run under Windows or Linux. Unlike FTP, the HTTP protocol does not allow other file operations such as deleting a file or displaying a directory.

The TCP: device cannot by itself be used at a file level, but if you wanted to, for example, write an email reader you would need to open a channel to "TCP:<host:port>" and read a write bytes as required.

EPNET also contains several :commands. Briefly these are:

:NET DIAG

For use when there is a problem, NET DIAG initialises EPNET and the w5300 step by step with diagnostic printouts as it goes. It also does some extra checks and tests, such as a w5300 memory test.

:NET STATUS

Prints out the unique MAC address and IP address etc. that has been assigned to EPNET. The MAC address is fixed and built into EPNET; the IP address and associated parameters (subnet mask and default gateway IP address) are obtained at start up using the DHCP protocol

:PING <host> or :NET PING <host>

Used to test connectivity with another computer on the network, sends an "echo request" to <host> and waits for a reply, and repeats five times. :PING may clash with a well-known extension also called :PING that makes a "bing" sound in BASIC, so might not be available.

:FTP LOGIN <host>

Logs in to a remote FTP server so that that EXOS FTP: device and :FTP commands can be used. For example, LOAD "FTP:myfile.bas" in BASIC loads a program and :FTP DIR lists the current directory of the server you've logged on.

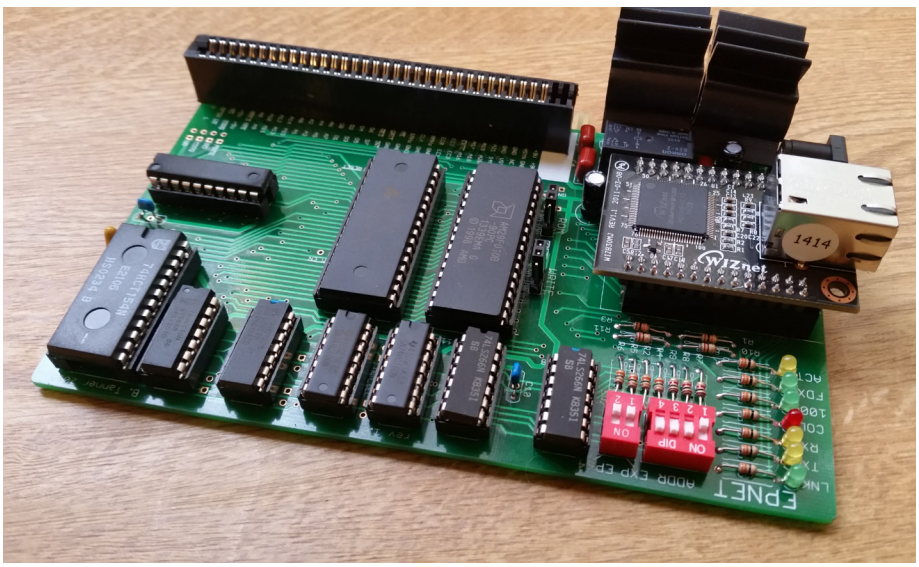
:FTP LOGOUT

Log out from a previously login server.

:FTP STATUS

Displays the status of the FTP server that has been logged in.

:FTP DIR



EPNET card is plugged into the EP bus bridge

Displays the current directory of the FTP server logged in.

:FTP DEL <file>

Deletes a file from the FTP server logged in.

:FTP MD <dir> and :FTP RD <dir>

Creates and deletes a sub-directory.

: FTP CD or :FTP CD <dir>

Displays or changes the current directory on the FTP server currently logged in.

As you can see all the FTP commands have been designed to match as far as possible the equivalent EXDOS commands.

:NET TRACE

Enters "trace mode", in which protocol diagnostic information is displayed. For example, FTP and HTTP server responses are displayed as they are received to help diagnose interoperability problems. "raw" mode can be turned on to additionally display the bytes at the start of every network packet sent and received, but results in a lot of output!

Issues

There is currently an issue with noise on the Z80 signals that I need to get to the bottom of before taking the hardware further.

There are a couple of bugs in the W5300 that I have needed to work around but hopefully that is the last of them!

EXOS creates a few issues with limited filename length and limited characters allowed in those filenames. For example the common way of specifying an IP address and port number host:port would be 192.168.0.1:80 but EXOS does not allow the : before the 80. I will provide some way around this.

The software is 80% done for a first version – it certainly meets my initial goal of being able to LOAD and SAVE programs over the network but there are a number of loose ends to tidy up.

Plans

My aim is to get to a stable hardware design and get a larger batch of PCBs made. I will then build these up and offer them for sale for the cost of PCB+parts.

I am only working on EPNET in "hobby hours" so I'm a bit worried about my ability to meet initial demand! Seeed Studio in China also offer a construction service so I may look into this as a possibility, but it would of course push the price up.

so anyone can make their own and/or modify it.

Future Possibilities

I will have to correct a few minor issues on the current PCB. Ideally the second version of a PCB would just be bug fixes from the first version but I have been asked several times if it could have more general expansion RAM on board. In fact a 512k chip is no bigger and only very slightly more expensive these days than the current 128k chip, but the memory decoding circuitry would have to change. I have also learnt that it would be very easy to add a Compact Flash socket for mass storage. The possibility of a single card that provides more memory, mass file storage and a network connection to a basic Enterprise 64 is difficult to resist...

Currently all the memory decoding is done in discreet 74HCT logic to meet my goals of EPNET being equivalent to the EXDOS card and being "hobby friendly". However it could be a lot smaller and more flexible if it used a modern programmable CPLD-type device...

Longer term it would be nice to be able to put EXDOS on the card and integrate the network with it. I envisage being able to map a network drive to an FTP server so all the EXDOS commands work seamlessly with local files or network files, eg COPY and DIR. But that would require some big changes to EXDOS...

Contact

The Enterprise Forever forum at www.enterpriseforever.com has a topic for EPNET under the English "Hardware" section. I have the user name BruceTanner (on the second page under "Members" at the time of writing) and I'm happy to message. Unfortunately I do not speak Hungarian but google translate is usually useable!

EnterMice

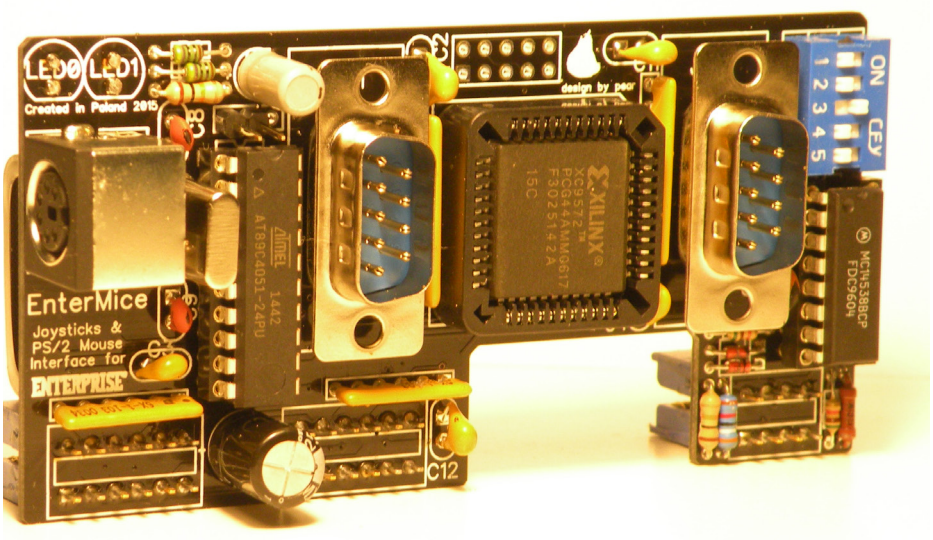
Author: Maciej Gruszecki (pear)

Introduction

This complex interface has been designed for the Enterprise 64/128 series computer.
The achievement is the culmination of six months of work of a multidisciplinary Polish developer known on relevant forums as Pear (at the instigation of the Spanish colleague GFlorez, who with Zozo, from Hungary, where the first testers of the EnterMice).
The main goal of the adapter is to allow the use of cheap and widely available PS/2 mice on the Enterprise computer. In addition, it enables direct connection of two joysticks in Atari standard without additional adapters.
EnterMice is based on solutions used in the BoxSoft mouse interface, which supports the Neos mouse (MSX protocol).
One of the EnterMice operating modes is fully backward compatible with the old BoxSoft mouse interface. An other operating mode converts the mouse movement to joystick pulsations.

A little story about Enterprise and mice

Enhancing the Enterprise 128 with mouse control is not a new and eccentric idea of a geekish retro-computing group, it was already planned by the original EP designers. The problem was that the task was relegated to Aztec Software Ltd, a modest English company that came up with a disappointing job; for two main reasons: first, the product was ugly [1], nothing near the rounded shape of the Enterprise. Secondly, it was not an adequate mouse as it relied on emulated joystick movement. Also at that time Enterprise Computing Ltd. was disappearing.
But the user base in the United Kingdom remained very active in the years following. Little companies were founded by Enterprise enthusiasts. New products were offered: memory expansions, customized cables, games, utilities, Spectrum emulators, an IDE controller, Eprom, motherboard expansions, serial modems and so on.
The most prolific company was Boxsoft, founded by developer Tim Box. Boxsoft offered most of the products listed above. Bundled with a paint program called Paintbox, they marketed an adapter plus a Neos mouse with a driver included. The Neos, internally a MSX mouse, had already been successfully adapted to other 8-bit computers like the Commodore 64.
At that time only a few 8-bit computers had a working mouse with some form of GUI. It was very difficult to sell something new of unknown use and not completely integrated on the computer. Seen from today's perspective, the advantage of navigating operating systems and applications with a mouse is obvious, its convenience being realized instantly when the mouse stops working...
After all, only about a hundred of the Paintbox bundles were sold.
The Paintbox program arrived to Hungary separately. Hungarians bought Enterprise computers knowing that they came from the UK., but not much more. When all the Enterprise 128s were sold, the user base was abandoned as in the country of origin. Once again, new companies surfaced offering new products. Then a company named „A studio" began to manufacture and sell the Boxsoft items, camouflaging them as their own products. One of the programs was a Paintbox version without the mouse option. It still needed the driver, but a crippled version. At the east side of the Iron Curtain nobody ever heard of the Neos mouse.
Later, in Hungary, a serial card expansion [6] for the Enterprise was realized, thanks to the genius of a team of Hungarian developers: Gyula Mészáros for the hardware, and László Haluska (HSOFT) for the software part. HSOFT took the crippled „A studio" mouse driver and modified it to suit the serial interface with a Mouse Systems mouse and later they supplied a built-in driver for the EPDOS 2.x operating system.
This is ancient history, but in recent years with WWW, Internet and globalization, some of the old artefacts Boxsoft+Neos emerged and all pieces of the puzzle began to fit. At last, the old promises (seen on Enterprise Computers Ltd. advertisments) of a real proprietary Enterprise mouse interface turned a reality.



The new EnterMice interface presented here is based on the work of the Boxsoft team, but adapted to recent times, supporting new options. (It is a miracle what Tim Box achieved with only a few logical gates and diodes back then[7]). As for the software, the driver was very well written by D. Rabson, Andrew Fitter and Andrew Richards. With minor modifications, the same driver will be used with the EnterMice adapter.

The principal problem interfacing the Enterprise is that connectors and signals are far from being a joystick-mouse standard. Little edge connectors that can be accidentally plugged upside down, commons pulled to +5v for the buttons when normally zero volt is used, lack of protection of the return of the signal when two or more buttons are pressed, no way to send information or state to the controller, and probably some more issues.

Now what Boxsoft pulled off with common, cheap components, has been redesigned and improved with two programmable chips and other modern components. One of the chips (a Xilinx) is in charge of the pinout and signal conversion to the Atari standard game port connectors. The original Boxsoft interface defined/standardized the first controller port only, that was used to connect the Neos mouse or a standard joystick (but not both at the same time). In the present project two joystick ports have been provided and they don't interfere with the mouse port.

The Neos/MSX mouse is not widely found today so, a PS/2 to MSX conversion adapter has been implemented. It is driven by simple and inexpensive 12 MHz Atmel AT89C4051, able to adapt itself to the frequency of the Z80 processor installed on the Enterprise. With this chip and the proper coding it has inside, the EnterMice can work on a 4 MHz EP or on a supercharged 10 MHz one the same way.

Also, the mouse reading protocol has been expanded following Prodatron and NYYRIKKI's new extended MSX protocol so it can send information about five buttons on a wheel mouse, if present.

List of mouse compatible programs up to date

- Paintbox by the mouse driver
- The Enterprise Graphical Interface by the mouse driver
- SymbOS internally, on EnterMice mode
- EDC Windows, internally in EnterMice mode and Hsoft driver mode.
- SPEmu128 from geco, the great Spectrum emulator for Enterprise, now is capable of emulate Kempston-mouse adapted games from EnterMice mouse movement. The list of compatible games is growing constantly.

Games list

- Pasziansz (Solitaire)
- SWAP
- Chess Master 2000 Amstrad conversion by Geco

XEP128 and EP128emu Emulators

Even if you don't have an EnterMice adapter, you can test the feel of managing an emulated Enterprise with a mouse.
Now LGB's XEP128 great emulator is able to mimic it accurately from the PC mouse movement. You can even combine it with SPEmu128 to emulate a Spectrum within the emulated Enterprise. The Kempston mouse will obey your PC mouse movements...
Is an explanation on how to organize the XEP128 emulator to correctly work on a Windows PC.
And here is the explanation for OSX.

Thanks to the works of LGB on his XEP128 emulator, IstvanV has also updated EP128emu for EnterMice emulation. There is a Windows installer for easy installation.

Features

The interface can operate in several modes. To configure it, a five sectional dip-switch is provided.
The configuration settings can be changed during operation of the interface. You do not have to restart your computer.
Below is the description of all possible settings.

DS1	DS2	Work mode
ON	ON	EnterMice native mouse mode
OFF	ON	BoxSoft compatible mouse mode. The two main buttons are swapped. Main now is the right one.
OFF	OFF	EnterMice joystick mode (Movement made to the mouse is redirected to the input of Joystick 1, and that port is disabled)
ON	OFF	not used

DS3	LED status
OFF	LED status disabled
ON	LED status enabled

Joystick 2 always works the same way, regardless of the interface operating mode being selected.

DS4	Sensitivity
OFF	Normál
ON	High

DS5	Diagonal correction
OFF	Disable
ON	Enable

Examples

```
100 PROGRAM „ms_test.bas"
110 NUMERIC X,V,MAX_Y",X_COUNT,LASTX,LASTY
120 GRAPHICS
130 CALL MOUSE_SETUP
130 OPEN £1:"mouse:" ! Open a channel to the Mouse
140 DO
15D CALL MOUSE_POS
160 LOOK 1:FIRE ! Check the fire button
170 IF FIRE THEN
130 PRINT £1:"o"; ! Turn the pointer off
190 PLOT LASTX,LASTY,X,Y
200 PRINT 1:"O"; ! Turn the pointer on
210 ELSE
l10 PLOT X,Y,
230 END IF
240 LET LASTX=X:LET LASTY=Y
250 LOOP
250 END
270 DEF MOUSE_SETUP
280 LET CHAR_Y=20 ! No. of characters screen high
290 LET X_COUNT=2 ! Pixel calc no
300 SET 180,101 ! Video channel to put
                mouse pointer on
310 SET 181,30 ! Position of co-ordinates on status line
320 SET 182,1 ! Show co-ordinates
330 SET 183,255" ! Colour of pointer
310 LET MAX_Y=CHAR_Y*36-2 ! Convert char_y
                          to co-ordinates
350 END DEF
```

THE ENTERPRISE FOREVER PRESENTS:

You have a program?
Have an Enterprise program!

ENTERPRISE DevCompo #1

Programming competition

LIFE BEGINS AT 30

Competition begins: 16 November, 2015.
Deadline and start of voting: 16 May, 2016, 23:59:59
End of voting: 22 May, 2016, 23:59:59
Winners announced on 23 May, 2016

Categories:
- Enterprise program
- Conversion
- Other

Winners of each category receive:
1st place: min. 150€
2nd and 3rd places are also rewarded
Winners of each category can only be rewarded upon presenting a 100% complete program.

You can ask the community of the Enterprise forum for help.

Conditions of participating:
A short video about the program (even uploaded to YouTube)
You can share the link on the Enterprise forum - http://www.enterpriseforever.com
- screenshots of the menu
- the program while operating
- the program itself

These can be all uploaded to the appropriate topic of the Enterprise forum. If you are having troubles with signing up, you can also send these to this e-mail address persa75@freemail.hu and we will upload them for you, but in this case please send us some information about the creators.
In order to enter the competition you need an at least 80% complete program.
It can only be called by a program which has not been published Enterprise.

If you submit a conversion it must also contain an Enterprise-related element.
You can also use ideas from other programs, but you have to note this in you own program.
One programmer/team can submit multiple programs.

Your program must be able run on any Enterprise computer with any kind of hardware of software extensions.
Further details about the conditions and the competition can be found below or on the Enterprise forum.

Countdown started... 42... 30...

Write a program for the 30 years old ENTERPRISE!

Az Enterprise DevCompo#1 végeredménye:
Here is the final result of Enterprise DevCompo#1:

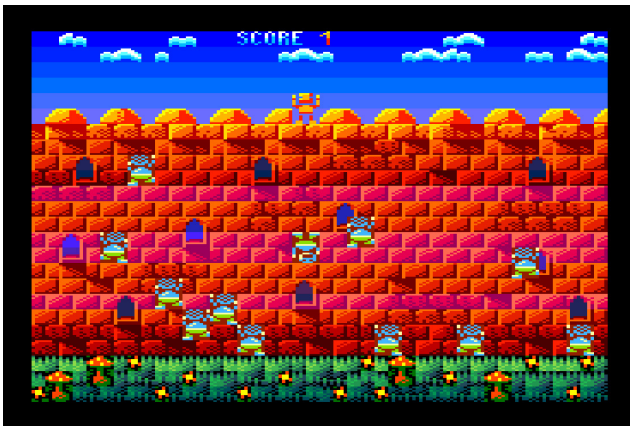
```
Basic:
Wall Defender játék:     7,789473684
Bomber 3in1:             7,1
FlapFlap játék:          6,263157895
Jumprise játék:          6,210526316
Snaki:                   6,105263158

conversion:
LIRUS:                   8,526315789
Filler:                  8,368421053
Pacman:                  7,75
Chase HQ:                7,631578947
ZenLoops:                7,473684211

Enterprise program:
Bricky Prise:                          8,894736842
Team Hat Trick aka Six Men One Puck:   8,631578947
Flora:                                 6,526315789
```

Mindenkinek köszönjük szépen a részvételt, külön köszönet gOblinishnek, aki nem EP-sként csatlakozott, és 3 nagyon jó programot köszönhetünk neki.
Thanks to everybody for joining, special thanks to gOblinish who met the Enterprise the first time and made 3 very good programs for us.

ENTERPRISE PROGRAM
Bricky Prise - Persa Noel (Geco)

KONVERZIÓ
Lirus - Németh Zoltán (Zozosoft)

BASIC
Wall Defender - Baráth Endre (Endi)

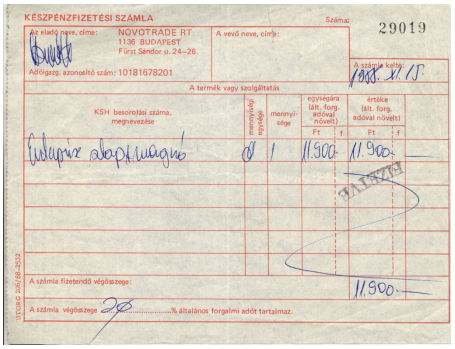


CONGRATULATIONS TO THE WINNERS!

How I became Enterprise programmer?

Author: Povázsay Zoltán (Povi)

We bought our Enterprise computer in November 1988. Here is the receipt :)



KÉSZPÉNZFIZETÉSI SZÁMLA

Száma: 29019

Az eladó neve, címe: NOVOTRADE RT. 1136 BUDAPEST Fürst Sándor u. 24–26.

A vevő neve, címe:

Adóigazg. azonosító szám: 10181678201

A szállítás kelte: 1988. XI. 15.

A termék vagy szolgáltatás

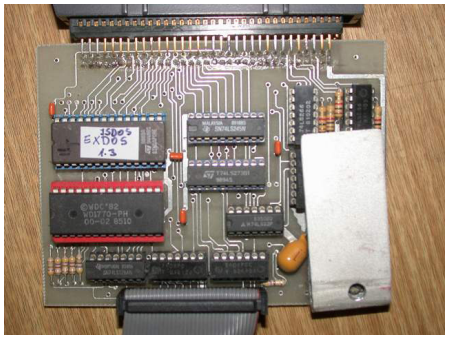
KSH besorolási száma, megnevezése | mennyiségi egység | mennyiség | egységára (ált. forg. adóval növelt) Ft f | értéke (ált. forg. adóval növelt) Ft f

Enterprise alapfelszerelés | db | 1 | 11.900 | 11.900

A számla fizetendő végösszege: 11.900

A számla végösszege 20 ..... % általános forgalmi adót tartalmaz.

In 2001, or 2002, I bought an EXDOS-card from Misi Bihari for 7000 HUF (~ 23 USD/ 18 GBP). This is a picture of it.



At that time I was a university student at ELTE, majoring in geography. I borrowed Kálmán Sztrókay's Z80 Assembly book for HT1080Z from the library of the department of meteorology, I learned the Z80 assembly from it. Also, there was a Tetris for Spectrum (Spectris) which had the 8th place on the 2002 MiniGame Compo in the 1kB category, and I had the documented asm source code of this.

I had the idea to convert it to EP according to the article published in SPV while learning the Z80 assembly. So I did, and a few years later with the help of Zozo it was compatible for EXOS as well :)

Encouraged by this success, a year later I entered the MiniGame Compo: I made a 50-level Sokoban in 4kB in 2004 and a 1kB Nibbles with 9 levels in 2005.

The Sokoban runs on 16-colour graphic screen, I drew the "icons" in Paintbrush, and the Nibbles runs on character mode screen.

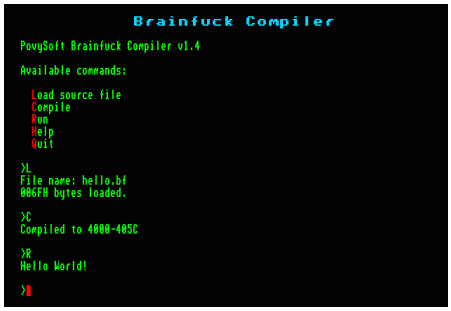
There is no music in any of the games, later I put the music of EDC's Nibbles into mine and the game extended to 3 KB :)

I started my most earnest game in 2005, at Christmas, and finished in 2006, and it's called Atomix. This uses the 16-colour graphic screen as well. Its pixilation matches that of the version of C64 except for the home screen (adapted from the PC version). Unfortunately, neither does this have music.

The level data are taken out of the PC version, where they were stored in a separate file so it was easy to get them. I got the graphics from the C64 emulator which runs on PC, and then I wrote a Pascal program to convert the elements making up the levels into EP format that I saved into paint. So I had to play through the game on the C64 emulator to have every graphic element on each levels (I'm not familiar with the C64 graphics, that's why I didn't get it from the file).

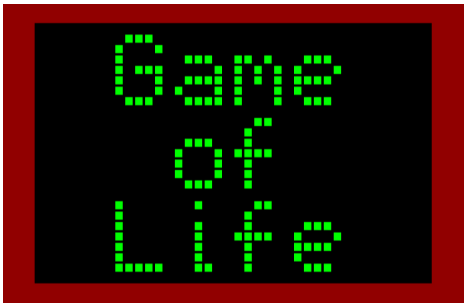
"User" programs:

Altair BASIC – you can read about this logout in details: http://logout.hu/cikk/negyven_eves_az_altair_basic_a_microsoft_elso_term/bevezetes.html



Brainfuck translator http://povi.uw.hu/brainfuck_compiler_z80.html

Lifegame / Életjáték:



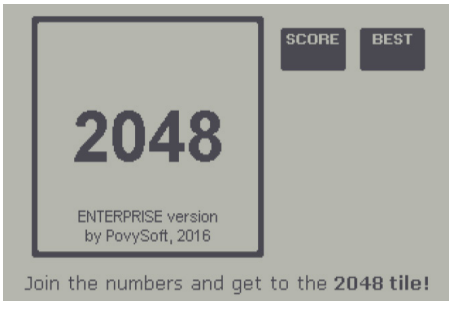
http://ep.lgb.hu/jsep/demo/?disk=http%3A%2F%2Fpovi.uw.hu%2Fgame_of_life%2Fgame.com&diskhack=load&autostart=yes&mem=128

it runs on character mode screen, there are some pre-built-in starter formulas (F2-F7, F8: exit, F1: random formula

SCELBAL

SCELBI was an Intel 8008 based computer, for which a BASIC was extended, and this was the SCELBAL. The assembly source code of the program (fully documented) was published in a 50$ book so this was the first open source program :)

Since the Z80 is very similar to Intel 8008 (every intel 8008 command has a Z80 pair) it was really easy to write an emu for it. The emulated 8008 registers are stored in the comma registers of the Z80.



The 2048 PC and mobile game of the success I thought I would write the Enterprise in 2048 by game machine as well. The game is made in machine code.

10 Mhz on the Enterprise

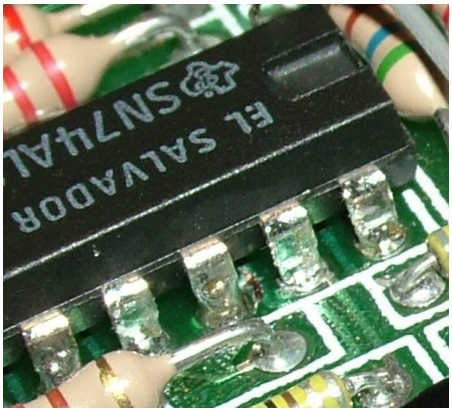
Author: Németh Zoltán (Zozosoft)

The hardware design of the Enterprise is originally ready for 6MHz model. The most important evidence about it is the option bit in the Dave (at port BFh), set the Dave sound frequency divider for the 4 or 6 Mhz system.
Very strange - why was 6Mhz machine not released?

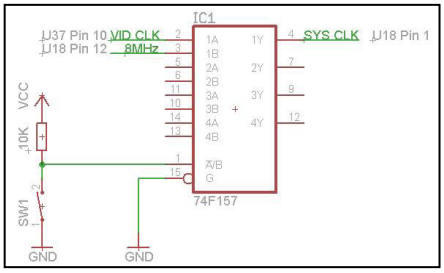
Originally only 3 modifications needed:
-Z80B instead of Z80A
-12 MHz crystal instead of 8 MHz
-modify one bit in the EXOS code, for setting right frq for Dave.

Unfortunately the Enterprise company made later a big mistake with the EXDOS: the WD disk controller does not have an own clock generator, it uses the system clock from the machine instead. So with 6 MHz model the EXDOS cannot work...
I think this is because of cost reduction... but make some additional work when want to make a Turbo Enterprise.

See the schematic.
The 8 MHz System Clock come from U18 Pin 12, need to cut this output line. Unfortunately it is hard to access, the best way is removing U18 and restoring it after cutting.

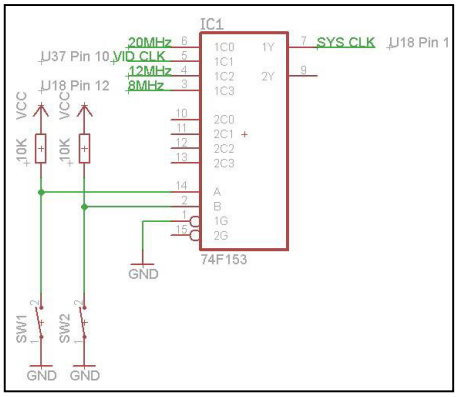


Next: cut the System Clock from the Expansion Connector then connect the A23 pin to original 8 MHz clock at the U18 pin12.
Then a new clock source is needed, 5V TTL compatible oscillator is the easiest solution, can be piggyback on U8, GND and VCC pins connected to the IC GND and VCC pins.
The clock output goes to the U18 pin 1 which are the input for the new System Clock.



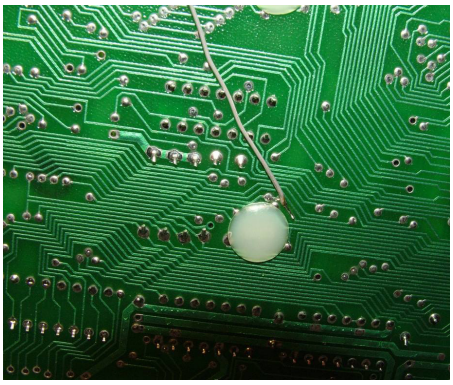
Replacing CPU to Z80B is optional, most of Z80A CPUs work on 6 MHz. Try it and if it doesn't work then replace it. (I have never replaced it for 6 MHz!).

The 7.12 MHz turbo is a cheaper trick because no new clock generator is needed, using the 14M Video Clock!
All same as above, just instead of the



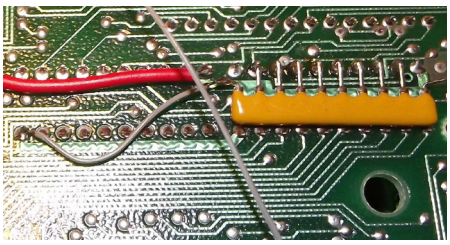
oscillator output, connect the U37 Pin 10 (Video Clock) to U18 Pin 1 as System Clock. Many Z80A work at 7.12 MHz but it is more possible that it needs to be replaced.

On some motherboards some problem can be found with the video memory on turbo speed, then short circuit R12. On higher speed (10 MHz :-)) also needs removing C7.
In 128K machines there also might be a problem with the internal 64K expansion, then short circuit R151.



It will be good up to 7.12 MHz CPU, but for 10 MHz new super fast SRAM expansion should be bought from Saint :)

It is a good idea to add Turbo Switch because not all the games are better at faster :-)
Very ugly solution just add directly a switch to the clock lines: one input from U18 pin 12 (8 MHz) another from the new clock generator or the Video Clock, and output go U18 pin 1. It is worked to me for many years in many machines.



The next lesson will be: how to build electronic switch with 74LS157 IC :-)

Final words about the replacement of Z80: buy least 10 MHz model (Z84C0010), but don't order it from China! Previously I wrote my experiences, the Chinese sellers sold relabeled old and slow Z80 CPUs.

How I became Enterprise programmer? #2

Author: Persa Noel (geco)

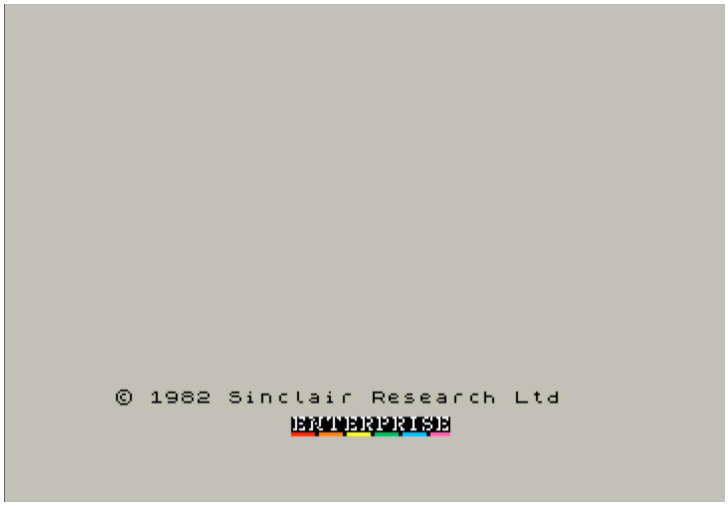
I asked our forum member, active in software developing, nick named Geco, about his relationship to Enterprise and his emulators, namely AMSTRAD CPC (2006), TVC (2008), VIC-20 (2009), ZX81 (2010), ZxSpectrum 48K (2014), ZxSpectrum 128 (2015).

My parents bought an Enterprise 128 for the Christmas of 1987 for family uses but I became the user of it. Later, as it turned out, my grandmother had a surprise for me as well, she bought a Commodore 64. So, 9 months after I got the Enterprise, the C64 was in my hands too. I played a lot more on the C64 but the Enterprise was in extensive use too and although I had more and better games for the C64, the Enterprise was closer to my heart. I still can't explain why, maybe because its external appearance or because you could program in the Basic, achieving quite a few things. Even back then, I perceived there were much more opportunities in the EP than I'd seen (mostly through the speccy transcription).

After the Basic I wanted to learn how to program in machine language but in my town I didn't have any acquaintances with Enterprise (nor Z80) who could have helped me. And the available literature wound up in Spectrum világ magazine. Even though I had the technical manual of EXOS, I couldn't understand it back then.

My first unsuccessful try was about the Mooncresta conversion, detailed in Spectrum Világ, and I tried it without any previous experience. What is more, one article of the series in the Spectrum világ was missing. After this, I didn't try for a while, I picked up again around 2002, with more experience, downloading all of the Enterpress magazines.

At the end of 1990s I met something called Armstrad CPC online – a marvel I had never met before. By chance, I discovered that the CPC and EP share almost same bit Airwolf code, so I was intrigued.



I grabbed the game called Alien Attack since it was a program using ROM calls, I started creating my own routines and switch the CPC ROM routines for mines. I got stuck at the sounds, and then I got to the Exolon by-pass, and I was lucky with that one – looking back on it, it is the easiest part of transcription. My initial gladness over the success in conversion was crushed by Zozo who pointed out that the program is not compatible with EXOS. Of course, just like 90% of transcribers at that time, I spent quite some time converting, although being easy.

I didn't even know what EXOS compatible meant, **Zozo** shared his loader, which helped me with amending the program. I'm still thankful that he drew attention to this and helped me.

In the beginning, my motivation was to provide more, better programs next to the tons of Speccy transcriptions that use the capability of EP better. And, as time went by, I tried adding some extras into the transcriptions: making them better than the originals for which the EPIMG-CONV and EPCOMPRESS/DTF utilities by István were great help.

While converting the games I realised that there are several Speccy software emulators for EP. Encouraged by this, I started developing the CPC emulator. And, carrying-on, during the Speccy conversion I had the idea to write a more effective

software Speccy emulator which converts the code after loading. This was my initial idea. I didn't think anything specific at that time, when I took the ROM source of the 48K Spectrum and made it work on EP. All this could do was like the old emulators. After this came the modification of the Speccy loading. On one hand, I wanted to have optional Speccy/ EP EXOS loading, and after loading the running shouldn't go back to the usual ROM routine but to the code examiner routine which searches for the predefined bite sequences and changes them for emulated versions – the initial conception was the 0FEH and 0FFH port reader and writer instructions.

Of course, not all of them, since the before or after in/out instructions are checked by the program in order to not change data which value matches the in/out instructions, so this is why the keyboard doesn't work in some programs since not implemented instruction combinations can be found in the program. With this, the Speccy program ran and this is where I had to face errors. Most of the Speccy games run in IM2 and due to my mischief the M1 programs do not work. So, how could I bridge this problem?

The main cause of the problem was that on EP the interrupt handler should be deleted in the interrupt routine or else we end up in an infinite interrupt cycle. If this didn't happen, the M2 wouldn't have caused a problem. How should this be solved? My first idea was to change the M2 instructions but this was a failure.
Instead of it the LD I,A instructions were changed and by this, the necessary EP interrupt holder deletion could be inset before the original interrupt. But some problems occurred again: there was a program which got into IM2 interrupt on EP sooner than the I register would have been set or the jump table would have been ready. This was followed by the next interesting Speccy solution: with the I register many programs set the jump table into the ROM or to a 0FFFH address since only the last two elements of the jump table are used on speccy. Although giving me another food for thought I was able to overcome this problem too so new programs were added to the programs running on the emulator.

After that, it was not the usual programs loading with ROM calling that gave me a headache. After some searching I was able to find a solution that resolves most of the errors made by such loadings. At the end of this whole tedious journey, a Spectrum emulator was created on which 40-60% of Speccy softwares run.

About the TVC emulator: an upgraded version is under preparation and it is going to be the same code changing version as the Speccy emulator.

The VIC-20 emulator for SAM Coupé was written by Simon Owen, this is what I converted while also making some modifications. Like this, in speed it was closer to the SAM Coupé version which speed is 6MHz. In the original SAM version there was no possibility for file selection. The program to run always had to be translated with the emulator, so we're in a more comfortable position than the SAM users.

Lacika's comparative test:
1 FOR I=1 TO 255
2 PRINT I
3 NEXT I

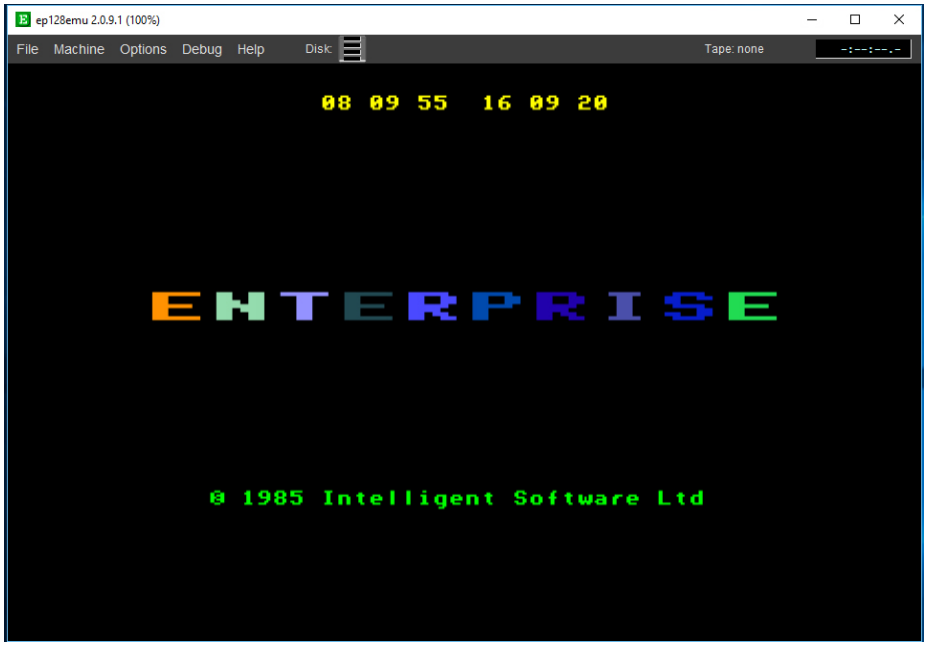
VIC-20	0:07
Enterprise (6MHz)	0:46
SAM Coupé	1:09
Enterprise (4MHz)	1:12
Spectrum	1:43

The ZX-81 emu is a conversion as well and it was published in 1997 by YRS (I don't know who exactly they are) for Spectrum. I disassembled the code, made a translatable code out of it and then made it 'EP-like'. It was modified as well, for example the screen on EP is a hardware program using ROM calls, that's why it's faster than the Spectrum, and the speccy-pushing, described here, has been removed as well which resulted in speed increase. (http://www.ep128.hu/Ep_Util/ZX81_Emulator.htm)

This, like the VIC20 emu, belongs to the emulator category since the Z80 instruction set is emulated as well – it may sound strange but this is the truth. The emulator uses the DE register as a PC, checks what instruction is on the given address, jumps on the instructed code, changes from the '-s register set to plain, performs the operation (the plain register set is the register set used by the zx81) and then changes back to the '-s register set, and checks the next instruction.

Ep128emu – Enterprise emulator by IstvanV

Author: Kiss László - ep128.hu



About

EP128emu is a free, portable emulator of the Enterprise 64/128, ZX Spectrum 48/128, and Amstrad CPC 464/664/6128 computers, written by **Istvan Varga,** using Z80 emulation code from **Kevin Thacker's** ENTER emulator.
ep128emu implements all the basic functionality of an Enterprise 128, such as the Z80 CPU (all documented and many undocumented opcodes are supported), NICK chip, DAVE (there is very good sound emulation in recent releases), keyboard, disk, and tape. For more details, read about the current features of the emulator.

EP128emu 2.0 is a major new version that has a GUI, a built-in debugger, better tape emulation, a tape editor utility, and many other features. This version is written in C++, and uses the FLTK and PortAudio libraries for GUI, video, and real time audio output. It can also make use of hardware accelerated OpenGL for faster and higher quality video display.

Note: pre-release versions of EP128emu 2.0 also included support for emulating the Commodore Plus/4; this functionality has been removed, and is now available here as a separate project.

General

Graphical user interface using the FLTK library.
Software (FLTK based) or OpenGL video, with resizable emulator window, fullscreen mode, brightness, contrast, gamma, hue, and color saturation control; additional features in OpenGL mode only:

- single or double buffered (with synchronization to vertical refresh) mode
- linear texture filtering
- resampling video output to the monitor refresh rate
- some display effects: motion blur, scanline shading, and (if OpenGL 2.0 shaders are available) PAL TV emulation

Real time audio output uses the PortAudio library (v18 or v19), with support for many native audio APIs (MME/DirectSound/WDM-KS/WASAPI on Windows, OSS/ALSA/JACK on Linux, and CoreAudio on MacOS X); high quality sample rate conversion with low aliasing; volume control, two first order highpass filters with configurable cutoff frequency, and an optional parametric equalizer can be applied to the audio signal.
Recording audio output to a WAV format sound file.
Recording video and sound output to an AVI format video file, with 768x576 RLE8 or 384x288 uncompressed YV12 video at 24 to 60 frames per second, and 48000 Hz stereo 16-bit PCM audio.
Saving screenshots as 768x576 8-bit PNG format files.
Saving and loading snapshots of the state of the emulated machine.
Demo recording (snapshot combined with stream of keyboard events which can be played back with accurate timing).
GUI tape editor utility for copying Enterprise files from/to ep128emu tape images.
GUI debugger with support for breakpoints/watchpoints, viewing the current state of CPU and I/O registers and memory paging, displaying memory dump and searching for a pattern of bytes, and disassembler with support for all documented and undocumented Z80 opcodes.
A simple monitor is also included, with commands like assemble, disassemble (also to a file), trace, memory and I/O port dump and modify, printing and changing CPU registers, memory compare, copy, fill, search, load and save, and more.
For most operations, addresses can be 16 bit CPU (affected by current paging) or 22 bit physical (all ROM and RAM data can be accessed, regardless of memory paging)

addresses. Watchpoints can also be set on I/O ports and physical addresses.
The debugger supports scripting in the Lua language, to allow for advanced uses like breakpoints with custom defined, complex set of conditions.
Configurable keyboard map for the emulated machine; it is also possible to use external game controller devices like joysticks and gamepads.

Enterprise emulation

Instruction based emulation of the Z80 CPU, supports all documented and undocumented opcodes, and memory wait states (including synchronization with the NICK chip when accessing video memory).
RAM size can be set in 16 kilobyte steps in the range 64 to 3712.
ROM can be loaded from external image files to segments 00h to 07h, 10h to 13h, 20h to 23h, 30h to 33h, and 40h to 43h.
Using external configuration files, it is also possible to define any memory configuration without the above limitations.
NICK chip emulation, supporting all documented and undocumented video modes.
DAVE emulation, including timers, interrupts, external ports for tape and keyboard/joystick, memory paging, and sound output (all effects are supported, and the polynomial counters generate the same pseudo-random „noise" pattern as on the real machine).
Tape emulation with playback, recording, and setting tape position; markers can be created for quick positioning to specific tape locations (useful for tapes with multiple files); uses custom file format which is PCM audio data with 1 to 8 bits of sample resolution and variable sample rate, and header including the table of markers. There is also limited (read only) support for EPTE format tape files, as well as read-write (although without markers) support for sound files like WAV, AIFF, etc.
WD177x (floppy drive controller) emulation for EXDOS.
IDE hard disk emulation; supports up to 4 2 GB disks, image files can be in raw or VHD format.
Optional extension ROM (epfileio.rom) that implements a FILE: device for direct access to files on the host system in a single user selectable directory.
Spectrum emulator card emulation.
Real time clock (at ports 7Eh, 7Fh).
External 4-channel 8-bit DAC at ports F0h to F3h.
Mouse emulation, an EnterMice device with up to 5 buttons and mouse wheel is emulated in native mode on column K of the keyboard matrix.
SD card (SDEXT) emulation, this is an experimental feature based on code from LGB's Xep128 emulator.

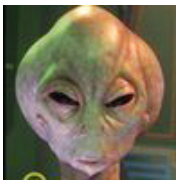
Spectrum emulation

Instruction based emulation of the Z80 CPU, supports all documented and undocumented opcodes, and cycle accurate synchronization with the ULA chip when accessing video memory and I/O ports.
RAM size can be 16, 48, or 128 kilobytes.
ROM can be loaded from external image files.
ULA, AY-3-8912, keyboard, and Kempston joystick emulation.
Tape emulation with playback, recording, and setting tape position; markers can be created for quick positioning to specific tape locations (useful for tapes with multiple files); uses custom file format which is PCM audio data with 1 to 8 bits of sample resolution and variable sample rate, and header including the table of markers; there is also read-write (although without markers) support for sound files like WAV, AIFF, etc.
Spectrum tape files in .tap format can be used as tape images for read-only hardware level tape emulation, or loaded directly to memory if the „Enable virtual file I/O" option is checked in the machine configuration.
It is also possible to load .tzx format tape images, although the support for these is not complete yet.
Loading SNA and Z80 format snapshot files created by other emulators is supported; however, snapshot saving and demo recording are only possible in the native ep128emu format.

Amstrad CPC emulation

Instruction based emulation of the Z80 CPU, supports all documented and undocumented opcodes, and cycle accurate synchronization with the CRTC and gate array when accessing video memory and I/O ports.
RAM size can be 64, 128, 192, 320, or 576 kilobytes.
ROM can be loaded from external image files to the lower ROM and expansion ROMs 0 to 7.
6845 CRTC (currently type 0 only), gate array, AY-3-8912, 8255 PPI (modes 1 and 2 are not supported yet), keyboard, and joystick emulation.
Tape emulation with playback, recording, and setting tape position; markers can be created for quick positioning to specific tape locations (useful for tapes with multiple files); uses custom file format which is PCM audio data with 1 to 8 bits of sample resolution and variable sample rate, and header including the table of markers; there is also read-write (although without markers) support for sound files like WAV, AIFF, etc.
CPC tape files in .cdt (same as Spectrum .tzx) format can be used as tape images for read-only hardware level tape emulation, although the support for this format is not complete yet.
uPD765 (floppy drive controller) emulation; it supports standard and extended .DSK files, and also access to real disks (not very useful in practice, since only PC format disks can be used this way), as well as limited emulation of the timing of disk rotation, stepping, and data transfers. The FORMAT TRACK (0x0D) command is not implemented yet, and some copy protected games and .DSK files with unusual track formats may not work correctly.
Loading SNA snapshot files (v1 or v2; v3 is loaded, but the extra information about the internal state of the hardware is currently ignored) created by other emulators is supported; however, snapshot saving and demo recording are only possible in the native ep128emu format.

Me, the Enterprise and my adventures with writing emulators



Author: Lénárt Gábor (lgb)

Preface

This article was written long ago so some updates can be read in the last part. My first intention was to rewrite the whole article (which failed as I had not enough time). This may not be a problem as the events can be followed in progress.

Googling around the net in "retro computing topics" I happened to bump into the Enterprise in about 2000. As I never heard about this machine before (however later it came to my mind that one of my primary school classmates had had one and I had seen it for a minute) I started to read about it. Its smart hardware and software solutions seemed amazing at once, at least compared to most of the other 8-bit computers. Of course I wanted to try it too using my usual method, so searching for an emulator, as there was no real machine here. I had to detect that there were some problems here however I found the emulator called "Enter" and perhaps EP32 as well (I don't remember any more), neither of them worked with my preferred operating system Linux. I never had Windows; the Wine (a Windows emulator-like thing) did not really run either of them. For some days I had been working with their source code but I realised that a better solution would be if I tried to write an emulator myself (learning more at the same time). I took a liking for writing an emulator soon however I wasn't able to achieve a usable one (after the Enterprise logo had turned up it always freezed). But everything was my own work in this, including the (half ready made) Z80 emulation – I began to deal with Z80 at this time; I only had some thoughts about it before. The whole thing ran only under Linux and I never published it. By the way, its name would have been Xepem "X EnterPrise EMulator".
As can be seen from the above mentioned, I hang out a little bit from the Enterprise community as I didn't have an Enterprise (only a C64) long time ago, unlike most of the others who are interested in this topic.

A lot of things happened then, I found the Hungarian Enterprise community, the forum, and I got a real Enterprise computer. (I am ashamed of myself because I haven't really used it except for some short attempts – I am going to use it more soon!) But the most important thing in our story is that the EP128Emu was also born (it is also possible that it's me who found it a bit late) which ran natively with Linux too. I would like to make clear: in my opinion the EP128Emu is the best and most useful Enterprise emulator. I would prefer to write a sensational review about my "attempts" here, I wouldn't affirm that my works are as serious and usable (far not!) as the EP128Emu. As an emulator for general purposes I recommend the EP128Emu to everyone. My "first" unpublished emulator (Xepem) was not needed any more as the EP128Emu appeared. This also worked which was not a bit of advantage. :)

In 2013 it happened that I was browsing the Enterprise forum and bumped into the "Web emu" topic. Here is the idea of an emulator running in a web browser however the original question referred to a plug-in or an extension. I thought I should see about the JavaScript possibilities as this technology was getting more and more popular and it's at that time that I found the JavaScript based ZX Spectrum emulator called JSspeccy too. This thing interested me too because I wanted to know about JavaScript more than it can be known from the some-line-long "web page decorating" solutions. I looked at the source code of the JSspeccy and realised that I may not be able to write something like this by myself for the first time (since then I would be able to do). Thus I decided to try to convert it step by step with the purpose that only the Z80 emulation should remain at the end, the rest would be my work in its whole and, of course, it would emulate an Enterprise instead of a Spectrum. The first, somewhat usable solution was created surprisingly fast though the cursor was even green here, so there were some lacks of Nick emulation and further faults, "small" problems. :) The emulator got the name JSep – hey, at last something whose name I don't put an X in. Thus JSep was also a learning process for me. Unfortunately this can be seen: with my present knowledge of JavaScript I begin to cry when I see the methods I used there. In fact this is the main obstacle for further development: simply I don't feel like dealing again with this primitive code. So, a whole rewriting would be needed – I could do this, however, it would not have much sense, see the last part of the article. Currently the emulator does not know a lot of things which would be nice, I mention only some of them: there is no sound, Z80/Nick/VRAM synchronization is inexact, you cannot save with the emulator, Nick/video emulation is inexact (interlace and so on), the emulator cannot be used on an own page or embedded. But there are at least two things that the JSep can and the EP128Emu can't: it emulates the Boxsoft mouse interface (EnterMice is also planned) and Am9511 APU (today it would be called FPU – however APU emulation has never been seriously tested as there is no suitable software). Certain limitations come from the web technology: lots of people have asked for "setting menus" and "attaching floppy disk images". The problem is that they forget it's a web emulator, its purpose is the integration into web environment, and nothing can be attached from a local computer. (Follow-up completion: Now it would be possible though it is not comfortable and not practical from some points of view) Everything has to be forwarded to the emulator through the web with URL (for downloading) or URL parameter (config settings and so on) format. The JSep can be useful perhaps to quickly show a program, mostly for those who do not necessarily want to install an emulator

XEP 128 Enterprise emulator for Windows, Linux and Mac OSX platform

to their computer. In this case a weblink is acceptable. Briefly: the aim of a web emulator is to run in a web environment which is not too surprising. This creates in most cases somehow different solutions that we are addicted to, with the native applications (e.g. EP128Emu). For example, I could imagine an Enterprise software collection on a webpage where the chosen program could be tried in the web with a click at once before downloading.

It surprisingly turned out however that the JSep solution can be useful in other cases too (mostly will be useful, as a thorough rewriting would be needed for acceptable results): I mean the smart phones, tablets, portable devices. At that time I tried out my work with my own smart phone: the result was depressing, it was similar to a slideshow. However with my up-to-date smart phone with an up-to-date browser the result has been surprisingly good! This is firstly due to the hardware development of the devices in question; however the efficiency of JavaScript is improving in the last years and this makes the browser manufacturers almost compete each other.
In the future the tendency can make a uniform solution possible in a heterogenous portable-mobile environment (Android? iOS? Windows phone?) – mostly with the present smart-phone specific API solutions that can even detect the device movement. I note that JSep is a "hand-written" JavaScript code, not an asm.js solution containing a C compiler as well (e.g. emscripten) which is able to "compile" the C code into the subsets of JavaScript. These solutions are said to be one and a half times or twice slower in certain cases than the native code (I cannot prove this, I haven't tried it). Of course, the web techniques can develop a lot, particularly the JSep. :)

The other success of the JSep was the deterrence. :) Namely I found the multi-tasking operating system called SymbOS created for the Z80 which didn't run on the Enterprise at that time. Having a crazy idea I loaded the CPC version in a CPC emulator which I converted on the source code level: in a certain moment saving the memory content and Z80 state into an own format. Then I slightly modified my Enterprise emulator so that it be able to load this, then I spent some hours to find out how the CPC memory/CPU state snapshot can be continued to run on a quite different computer, the Enterprise. :) The modified JSep emulated CPC too this way somehow! My plans would have been to patch the saved CPC memory content step by step until it runs in the emulator without any CPC hack, then only a loader would have been needed. So this was about the port of SymbOS for the Enterprise, based on memory snapshots. I know the idea is rather strange. So strange that Prodatron (the author of SymbOS) answered me and said it is so stupidity that it is already great. :) More exactly being frightened by this he himself ported the SymbOS for the Enterprise accurately. The truth is that there had already been plans for the Enterprise port but (as far as I know) nothing more happened. So, if the JSep had no other sense it was useful in connection with the appearance of SymbOS for the Enterprise. :) It is possible that Prodatron would tell this story differently. :) Dear readers, excuse me but let me be happy with a little success. The sense of achievement is important to motivate not to stop doing the work. :)
On a nostalgic day (already in 2015) my first Enterprise emulator project came into my mind, written in C, running only on Linux (that is Xepem). As I intended to know better the SDL2 too (SDL is a cross-platform media layer available for e.g. Linux, Windows, Android) I decided to try to "imitate" my old project based on the SDL2, using not an own Z80 emulation but that of z80ex project (which I have previously used in some small projects, e.g. CP/M emulation layer in UNIX environment and similar craziness). This time I could do it better than earlier which may be due to the fact that the Z80 emulation was surely full and suitable. :) The new emulator was named Xep128. This name had UNIX origin (names of GUI software running under X11 used to begin with an X traditionally). My first goal was learning, similarly to the JSep, this time the SDL2. To tell the truth I didn't intend to get until the point I got with the JSep.
The Xep128 also ran only under Linux but I was thinking: this is now SDL (which is a cross-platform) - how complicated it would be to create a Windows program. (This is now simplification, apart from SDL2 it contained Linux specific things too which I had to solve first.) I repeat: I never had a Windows and I never programmed anything under Windows! So the solution was that I tried the Mingw cross-compiler which is able to create a Windows .exe file from the C code. You can imagine how surprised I was when the exe was created with half an hour of work which ran under Wine. Reaching to this point (first it was only some curiosity from me if it could work under Windows) I decided to keep the dual-platform nature (since then it has not only been dual), the Xep128 should work with both Linux (or other similar UNIX) and Windows operating systems.
I tried to improve my habits of development (this time with my hobby as I have always done this with my work): I put everything into version-tracking systems (git) and I made it available for everybody using Github. Furthermore my other aim was to specialise in something with Xep128, not being only a "general" emulator: as JSep is for the web, Xep128 is to emulate the "more extreme" hardware parts. Why? As Xep128 has (updated: had) at least as much deficiency as (or even more than!) JSep, I only had to find out something what it should be good at. :) Yes, the Xep128 is (was) able to do less things than the JSep, e.g. it has (had) only VINT/1Hz interrupt. But the Boxsoft mouse (since then EnterMice mouse as well) does work. This and more other things were easy to do as I only had to rewrite my JSep JavaScript code into C. But, for example the SD card support is new which can be found neither in the JSep nor in the EP128Emu (since then EP128Emu has had this). Now only loading is possible (since then saving too), but it shows that the native implementation has its advantage compared to the JSep; this would be rather difficult to solve with the web, especially with a large disk image. The SymbOS works with Xep128, with a mouse, from SD card too (it works with JSep too, but, of course not from SD).
To show what else is considered to be "extreme hardware" a good example is the Z180 emulation in the Xep128. Not many people have an Enterprise with a Z180, as I know, only one has. :) Though it seemed an interesing problem to solve this. (It is not ready yet, but is already suitable to detect, for example, the Z80 opcodes unknown for Z180). Emulating the ZX Spectrum hardware emulator card :) also seems to be a crazy idea but it might not be an original idea (EP128Emu also has this as I know). However, without any doubts it would be the most subtantial if I went on with the half-ready made work, the EPNET emulation. If you haven't heard of this, EPNET is Bruce Tanner's project (who created the original IS-BASIC and a part of EXOS), briefly: "Connect the Enterprise to our Ethernet LAN and to the Internet." A webemulator wouldn't be able to do this as it cannot use the network with different native protocol than http. The Xep128 being a native application can be able to do this. In my opinion the EPNET will be cool hardware as "Enterprise connected to the net" is something new. Its emulation may be thus a quite important task. In the new SymbOS version there is network support (however not in the Enterprise version) and Prodatron would certainly put network support into the Enterprise version with pleasure. Using the EPNET emulation of the Xep128 this could be tried without real hardware (could be, if I could go ahead with this).

One of the biggest disadvantages of the Xep128 at the moment is that it is far not user friendly (except the user who loves command lines like me). In the case of JSep it can be accepted because of its characteristics (web), that it cannot be freely configured, cannot load local files as ROM or disk images, etc. But one most essential characteristic of the Xep128 is exactly that it is a native application. What makes the problem more complicated is that the SDL (SDL2 or more) mainly gives only a "window" into which you can "draw" freely. If a control panel is needed it is your task to draw them pixel by pixel (printing a text goes in the same way). The other possibility is to use the SDL as well as the own GUI elements of the operating system that runs the emulator. This is problematic because the almost entirely common Linux-Windows code base would be given up (which mostly meets the operating system through the SDL, so it cannot see any

difference, this is done by the SDL), and separate Windows and Linux specific implementations would be needed to create this. (Since then e.g. the FILE: implementation has used a file selector window in both Windows and Linux.) I wouldn't like to introduce other dependence. However there are platform independent SDL GUI solutions, they don't look good and their size is not really small either ...

What I find interesting is the "great unification theory": Some "app store"-like thing for the Enterprise, through the net, as it has to be. In real hardware there is the EPNET for this, it works. In the case of JSep through HTTP/AJAX too. Finally in the case of the Xep128 we are at the first case with the EPNET emulation.

By the way it is very interesting to see that the emulators are connected through the source code. The Z80 emulation of the JSep is still from the JSspecy which comes from the FUSE (FUSE is originally a Spectrum emulator written for UNIX: Free Unix Spectrum Emulator). The Z80 emulation of the Xep128 comes from the Z80 emulator project called z80ex (however I significantly modified it) which surprisingly comes from FUSE as well. I found out these only some time later, in any case it is interesting. All these above mentioned project software licence (GNU/GPL) make these source code borrowings also possible (meeting some requirements, e.g. the source code of your own work has to be attached). In connection with the Xep128 I am satisfied with the z80ex but I plan to rewrite it in the JSep, I want to create Z80 emulation from the original FUSE code base as the JSspeccy did.

It is an interesting fact that the "quite fast" JavaScript seems to be developed into the direction of asm.js and similar technics which are not favourable for the "hand written" JavaScript code like the JSep. Moreover the mentioned emscripten (from C into asm.js) compiler "emulates" exactly SDL API. So, with some modifications, the Xep128 could theoretically be able to "generate" web emulator in Windows and Linux native usage, using emscripten (since then it has now been possible). It is also true that SDL exists for Android too so it is also possible to compile the Xep128 into an Android native application, expending some work. Who knows; in the long run it would worth developing only one emulator and not both a web and a native one? For this however, my knowledge must still grow. Exactly this is one of the edifications of my article (at least, in my opinion): if one wants to learn something why not to do this through a project in connection with one's hobby but the knowledge can be utilized in more other ways? The other interesting consequence: Lots of people asked me why I didn't use Windows as anybody else; I wouldn't have had any problems with the emulator. Yes, it is possible (of course one's operating system selection doesn't depend on only this) but then I wouldn't have wanted to write one so I wouldn't have known the machine so much, and the machine code, etc. The Enterprise might have been only a quick adventure for me like "I have seen this machine as well" (I could mention lots of examples for this like most people, I think). This way I took a liking for it in the meantime, I can say. I had similar experiences with other topics too, and not only in connection with the "old computers". It sometimes worths to have a go for heavier things too.

Finally I would like to thank everyone on the forum especially two people: Zozo who provided me with lots of information about the Enterpise (or sometimes with requests in connection with the emulator) and István (the author of the EP128Emu) who showed me directions more time in the maze of emulator writing. And everyone who tried out my work and said opinion (especially our forum member Gflorez who enthusiastically tried out everything, gave suggestions with "mousing", and even designed an icon for the emulator). Special thanks to the reader who could read my short novel here and now.

But this is not the end yet ...

My sets of letters of this article (which would have ended above) is not up-to-date, much water has run in the Danube since then. The most interesting news is that István has begun to deal with the EP128Emu again, which is great news for all the Enterprise users and all who are interested. Myself, indeed, as the author of the Xep128 have to admit: the Xep128 cannot substitute the EP128Emu at all. Since then a lot of missing things have been built into the EP128Emu (compared to the Xep128), e.g. the emulation of the mouse and SD card. In this case I also see that the Xep128 might not be directly useful for many people, it was not senseless as the EP128Emu "borrowed" the SD emulation capability from the Xep128. This way my work is more useful as if this (in my opinion useful) capability were only in the, by less people known Xep128.

The Xep128 has developed a lot before the above mentioned happening as well as after the writing of the original article. Only in broad terms: a FILE: device as file manager (which has arrived at the EP128emu too), native Linux (GTK-3) and Windows file selector window (at least some GUI), all the Dave interrupts have been implemented, SD card writing is available, there is unprofessional sound, and so on. We can say that the Xep128 is able to do as many things as (or more than) the JSep was able to however it is far not something like the EP128Emu.

Actually the JSep is a "dead" project in the sense that I don't think I will go on developing it. After all, it is also difficult to improve one emulator, you shouldn't want to write and continuously develop two emulators especially if they are not "interchangeable" (as one was written in C, the other in JavaScript) respectively the emulation of the same computer is in the air, in addition. At the same time, fresh news is that the Xep128 can be compiled into the HTML5/JavaScript (the "web") platform to what I have written about above, and there is a working demo version too in this regard (which is also suitable "instead of the JSep"). Among other things it is good because it is enough to develop an only emulator and it is available for numerous platforms inculding Linux/UNIX (including Raspberry Pi), OSX, Windows or even the "web". What is missing so much are perhaps some mobile platforms, for me personally the Android (it would look great on a tablet ... however the web version might also give acceptable results on such a device nowadays). Let me mention a misbelief: the JavaScript has nothing to do with the Java! They only had similar names possibly because of marketing reasons. The JavaScript is - mainly today - "the programming language of the web" and is a native element of the browser while Java needs a separate plug-in and the JRE, moreover as programming languages they are most different.

The Xemu project is also among my activities. Here my aim is to create an "emulation framework" within which I can emulate different computers so I don't need a separate emulator project for everything, beginning from the bases. The next ones are now available: Commodore LCD (a rather unknown computer that had never been released to which it is me who wrote the first working emulator, still in JavaScript), Commodore 65 (yes, 65, not 64), the Mega-65 (nowadays it is the main part of the Xemu project; this would be a very long topic of 10 pages and is not related the Z80 and Enterprise so I dispense with this) and some more known things too (Primo, Commodore VIC 20). Of course to be continued both as for the number of emulated computers and as for the features. E.g. TVC and Jupiter Ace are in progress – They will be ready in the next millenium as I know myself. As the Xep128 has similar bases my long range plan is to continue it under the Xemu project, soon or late. This won't mean a negative change for the Xep128 users (if there are any...) but it is about the question of maintenance again. The Xemu will also continue the multi-platform feature (including the "web" as a platform) but the "multi-purpose" aspect is also added where purpose means a computer as a purpose. But, of course, now the Enterprise is the main thing for us.

http://xep128.lgb.hu/
http://jsep.lgb.hu/
http://xep128.lgb.hu/web-demo/
https://github.com/lgblgblgb/xemu/wiki
http://c65.lgb.hu/web-xemu/xc65.html

- LGB

Using EPIMGCONV

An example for using it:

epimgconv foo.jpg foo.com

Parameters

Video mode can be chosen by -mode parameter, using the next values:

* **0: 2 colours,** normal vertical resolution (not surely black and white but an optimized palette is generated for each lines)
* **1: 4 colours,** normal vertical resolution
* **2: 16 colours;** a picture dithered to 256 colours is being converted into a picture of 16 colours without dither (default setting), normal vertical resolution
* **3: 16 colours;** it generates the same palette as mode 2 but the original picture is being dithered to 16 colours, normal vertical resolution
* **4: 16 colours;** it tries to mix generating palette and dithering but it's very slow, normal vertical resolution
* **5: 256 colours,** normal vertical resolution
* **6: attribute mode,** normal vertical resolution
* **10: 2 colours** (black and white), interlace (not surely black and white)
* **11: 4 colours,** interlace
* **12: 16 colours;** a picture dithered to 256 colours is being converted into a picture of 16 colours without dither, interlace
* **13: 16 colours;** it generates the same palette as mode 2 but the original picture is being dithered to 16 colours, interlace
* **14: 16 colours;** it tries to mix generating palette and dithering but it's very slow, interlace
* **15: 256 colours,** interlace
* **16: attribute mode,** interlace

So **-mode 0-6** uses the normal vertical resolution and **-mode 10-16** uses the interlace.

--help prints brief usage information, and the list of all available command line options.

-raw 1 parameter enables to save in raw format (not usable without a separate viewer program), instead of the default .com application format; this way the size also can be larger (if the output format is application, size can be at maximum 46 characters * 288 lines only). -outfmt n also enables to save in raw format, it can have the following parameters:

-outfmt 0, -outfmt 1: same as -raw 0 and -raw 1.

-outfmt n (11<=n<=19): Here n means the compression level of raw file. epcompress utility enables to convert between compressed and uncompressed raw format. On the Enterprise, the uncompress extension included with the epcompress package can be used to decompress and load compressed images.

-outfmt 2 saves the converted picture into Agsys .crf format which can be used only in mode 1 (4 colours/lines, not interlace); furthermore Agsys cannot load files of large size (-size 36 216 still works but -size 40 240 doesn't). Though Agsys supports palette that changes every lines, the picture may be easier to edit by setting a fix palette with -palres 0 or the parameters -color0..-color3.

-outfmt 3 saves the picture into ZozoTools format (which uses VL and VS commands). It doesn't support interlace and attribute modes, and fix file size can be at the most 16K. This format sets automatically -palres 0 and EXOS-compatible picture-size (at the most 42x27 characters, height divisible by 9) like -outfmt 4.

-outfmt 4: PaintBox format; it doesn't support interlace modes either, the file cannot store FIXBIAS, so it has to be set either according to the value given by the converter

or you have to use -bias 23 parameter when converting. This format sets automatically -palres 0 and EXOS-compatible picture-size (at the most 42x27 characters, height divisible by 9) as well as -outfmt 3.

-outfmt 5: Zaxial format, with similar limitations as in the case of PaintBox. This format does store the bias value, however. Attribute mode is not supported by Zaxial, but pictures converted to this mode and saved in Zaxial format can be loaded with the original VLOAD command.

-size W H enables to set picture size. W = Width in characters, H = Height in lines (interlace: field lines). Being one of the given values 0 or negative, it'll be automatically calculated from the other value and from the size of the picture. E.g. -size 46 0 means that width is always 46 characters and height is automatically chosen by right of the size of the picture. This way however height could be also too big, so giving a negative value the maximal size can be set. E.g. -size 46 -300 effects the same as -size 46 0, but if the height were bigger than 300 lines, it'll be 300 lines and width will be also reduced.

As default, Epimgconv converts images into the largest possible size, but by using -scale X Y and -offset X Y parameters the image can be enlarged and offset.

-scalemode 1 resizes the picture not in the way that it have the largest size without the cut parts but without the smallest empty (border-coloured) areas. So if the proportion of width and height given by the -size parameter is not fit, there won't be any border-coloured areas but a part of the picture will be cut sideways or underneath and above.

-color0 n ... -color7 m parameters can set fix palette colours (actually setting of avoiding using certain colours is not available). As -mode -0 and -mode 10 parameters don't result a black-and-white picture, -color0 0 -color1 255 parameters are needed to be given to reach black-and-white pictures. The colours can also be given in #RGB format, for example #000 is black and #773 is white.

-bias parameter sets fixbias - maybe better quality is available than by automatically selected colours. The colours can be given in #RGB format as well, e.g. #331 means bias=31.

If the .com file is too large (>47.75K), it'll be automatically compressed; so EXOS (being lucky) can load it though it starts much slowlier. Using -nocompress 1 parameter, automatic compression of „large" programs can be turned off.

Parameter **-nointerp** 1 turns off interpolation while resizing a picture (e.g. converting from 32x32 to 512x512 2 colour interlace format without interpolation - this way all the pixels became a square of size 16x16.)

-palres 0 disables palette changing each line (default: -palres 1), so the palette will be the same in the whole picture. It generally gets quality worse so it might seem not to be very useful, but this way the converted file can get somewhat smaller and it might be easier to use in some other programs.

IstvanV

Suggestions, tips

I used the following methods and combined them:

- attribute mode results the best quality and then the 4 colour mode;
- I picked pictures containing not too complex structures;
- I cut the better part from the picture but in the way that its composition remain pretty;
- I looked for pictures where it's useful that there are lots of colours in the horizontally separated parts;
- I picked pictures of strong colours and of many, diverse colours.

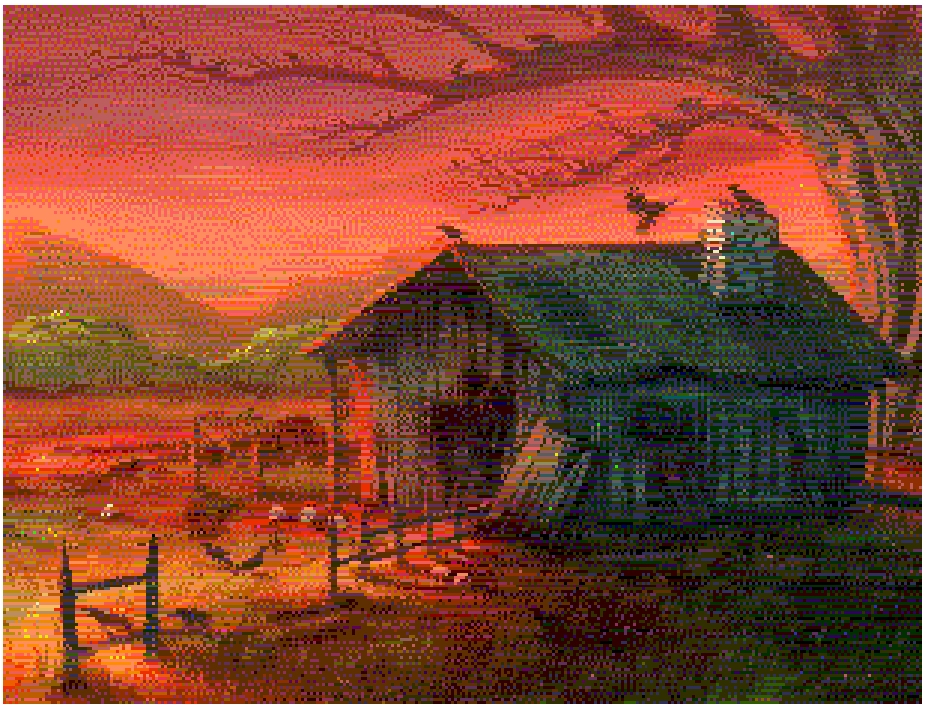
The better conversions were where these factors turned up the best way.

Endi

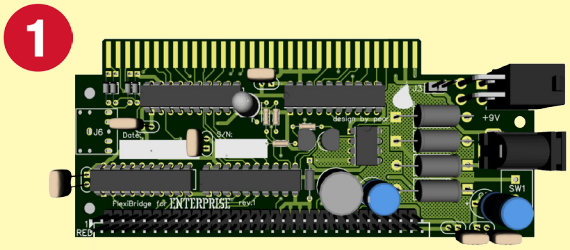
This 16-color mode attribute, Palette line by line, and interlaces and photo-like images on it has become the most:

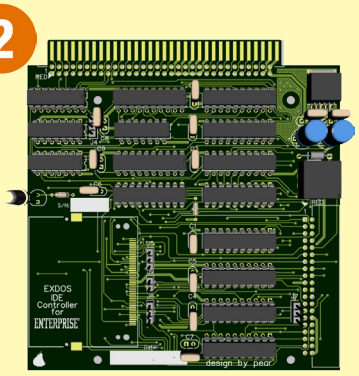
epimgconv_sse2 jpgname epicname -size 42 280 -quality 9 -mode 16

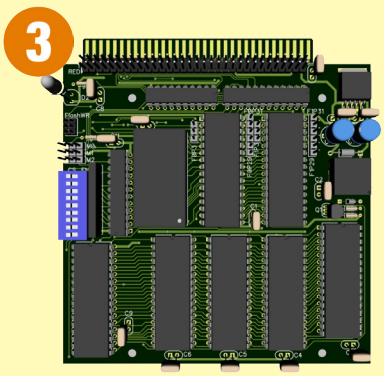
Zozosoft



3 new hardver from Pear (Maciej Gruszecki)





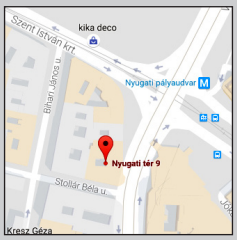


A new revised version of the FlexiBridge and IDE-CF module I sent into production.
Almost finished a module RAM-Flash-Clock.
I'm a little caught a cold and I do not give advice to prepare files for production in this year.
I need some rest.

Short description of RFC module:
- RTC consistent with the project M.Gy. Enterprise Clock & Calendar (DS12887),
- sockets for two FlashROM and five SRAM (512KB each),
- both FlashROM can be replace by SRAM (configuration jumpers)
- possible to program 8 modes memory mapping (3 jumper M0, M1, M2) if I manage to fit, so much logic in the PLD (GAL 22V10)
- each chip can be independently off, including the RTC (dip-switch).

ENTERPRISE KLUB

Every last Saturday of the month
Location:
Hungary
Budapest (V. ker.)
Nyugati tér 9.
Skála terem
2 pm. – 7 pm.



Information: www.enterpriseklub.hu

If you want to become a writer at ENTERPRESS, send us an article or game review or writing of any kind relating to the Enterprise computer and we will get back to you as soon as possible!

E-mail address:

info@enterpress.news.hu

ENTERPRISE FOREVER

https://enterpriseforever.com

ENTERPRESS Magazine - November 2016

Editor in chief: Matusa István
Deputy editor in chief: Németh Zoltán (Zozosoft)
The Team: geco, Povi, Kiss László, SzörG, szipucsu, lgb
English translate: Kovács Bogi, Hajdó Máté, szipucsu
Design, printing works: Matusa István
Web: http://enterpress.news.hu
E-mail: info@enterpress.news.hu
The magazine will now periodically appear in electronic form.
Limited number of copies will appear printed form.
ENTERPRESS e-magazines:
http://enterpress.news.hu/index.php/en/enterpress-news