RetroMagazine World

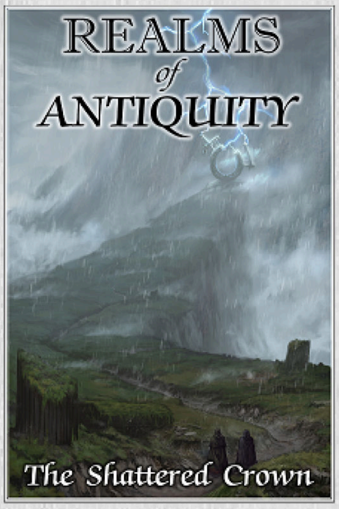
future days are back

Issue 6 Year 2 - March 2021 - https://www.retromagazine.net - Free Publishing

SHARP X68000

The arcade home computer from the Land of the Rising Sun

feeling that sense of freedom...



ANBERNIC RG351P

A portable mini-console to emulate consoles, arcades and 8-16 bit computers

Game reviews!

SUPER MARIO RPG (SNES)
ADVENTURE BIT (PC WIN)
TINY QUEST (C64)
SOUL FORCE (C64)
PIER SOLAR AND THE GREAT ARCHITECTS (Megadrive)

- RetroMath: Finding solutions by discretizing and iterating
- C128: redefining characters for 40 columns display
- May the FORTH be with us - part two
- Connecting the Sinclair ZX80 to an LCD TV
- Installing VICE 3.5 on RaspberryOS compiling the source code

INTERVIEW: Commodore Engineering - Welcome back to the scene!

Kick Off VS Sensible Soccer

Memories of a past present

The dim light of the lamp illuminates the room as shivering shadows of the evening pass swiftly over the walls. Life can be strange sometimes: one joke leads to another and here I am writing my first editorial for this fantastic magazine. I followed it for a long time because in my opinion it represents, in the video game scene, the painting on canvas of the current artistic current known as “Retrogaming”; therefore I let you imagine that sense of amazement mixed with joy that pervaded me at the time when I was asked if I had enjoyed joining the team.

I see myself sitting on the old desk at home in those magical eighties while, lost on some eight-bit screen, I imagined what the future of gaming machines would be and more. Precisely in this issue we find an exhaustive paper about the beautiful SHARP X68000 system, which for me stood at home computers as the PC Engine stood at gaming consoles: it was a dream. In those glittering eighties the portable games appeared with the beautiful but unmanageable Atari Lynx and the less colourful but much more effective Nintendo Game Boy.

It seemed like a dream with those little gaming machines in your hands and today you can choose from a myriad of models, including the amazing ANBERNIC RG351P of which you will find an accurate paper on this issue. It is incredible to see where technological progress has taken us today, but we must never forget where we came from, because history is a teacher of life. It demands respect and the paper about the presentation of Sinclair QL in Italy in 1984 is one of our own ways of showing respect. Moreover, today we have plenty of technological goodies, systems such as the new Raspberry PI 400 are obtaining enormous success and, as you will find in this issue, it can be an excellent idea to make it a "stand alone” machine: minimum cost and maximum gain to have a targeted and perfect emulation. Becoming young again through this magazine is the easiest thing in the world: we find the good Basic, listings, very interesting papers such as that on characters redefined for C128 with 40 columns. True, I may be overwhelmed by passion and emotion, but how could it be otherwise? Here we go to touch the strings of the most visceral emotions, those that take the stomach and bring back memories of crazy challenges elbow-to-elbow that often led almost to a fight, with the article on the summa of football simulators, those two giants of Kick Off and Sensible World of Soccer. The crime is complete with the sumptuous paper about Commodore Engineering and their CP-64 project, something that makes you faint.

The weak light of the lamp is still there, and it still illuminates my room and provides suggestions, images, visions. Progress, science and technology have made giant steps as a bluish screen prints fiercely on the monitor, just like in the old days. Next Gen systems have been created and with them ways of playing never seen before. The arcades have disappeared but not their games and this is thanks to technology. Preserving, sharing and transmitting: these words have always been my mantra, since when computer stores making “backup copies” of games was not perceived as a crime. Preserving, sharing and transmitting. Because history cannot and must not be forgotten.

Mic "The Biker" Novarina

SOMMARIO

People involved in preparing this issue of RetroMagazine World (in no particular order):

- Alberto Apostolo
- Gianluca Girelli
- Michele Ugolini
- Carlo N. Del Mar Pirazzini
- Daniele Brahimi
- Mic "The Biker" Novarina
- Francesco Fiorentini
- Ermanno Betori
- Edoardo Ullo
- Hakim Rezki
- Massimo Sanna
- Giuseppe Fedele
- Roberto “The Bard" Del Mar Pirazzini
- Cover by Flavio Soldani
- Proof-reading by RMW editorial staff

A bit of rarity

(rummaging here and there)

Connecting the Sinclair ZX80 to an LCD TV

by Alberto Apostolo

In 2015, one of my father's colleagues gave me a Sinclair ZX80 with 8K ROM Upgrade which had been in a basement for a long time. I turn it on from time to time to check if it is still work. Everything OK, except the disturbed video signal coming out of the TV modulator (probably due to some oxidized contact).
I was always afraid to open the computer because I didn't want to damage it, until one day I took courage and asked a professional electronic technician (my brother) to make a change to fix the problem.
Very carefully, we opened the computer without breaking the very delicate plastic rivets that connect the two parts of the casing. Then a normal electrical cable was welded upstream of the TV modulator to bring out the Composite Video signal. The technique is not new and was previously used on ZX Spectrum and ZX81.

Fig. 1 shows where to weld the two wires conducting the cable (GND=black). The welded cable passes through the hole where the TV output is already present, sacrificing the use of the original coaxial cable (**Fig.1**). In addition to a green phosphor CRT monitor (**Fig. 2**), you can connect your computer to a modern LCD TV with a Composite to SCART Video adapter combined with a SCART to MINI-SCART gearbox (**Fig.3**), bringing the Sinclair ZX80 into the 21st century (**Fig.4**). If the image quality does not satisfy us, it is suggested to change the contrast, brightness, etc. parameters of the LCD TV (restoring the previous values when we finished using the computer).

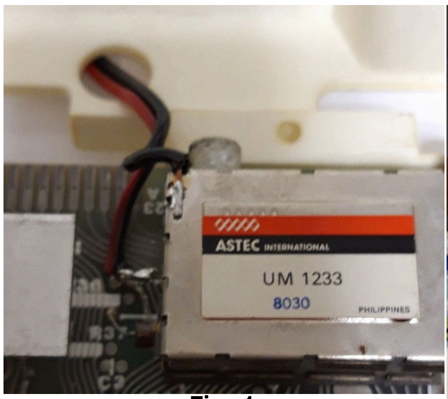


Fig. 1

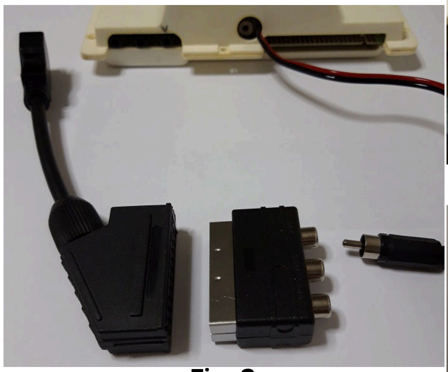


Fig. 3

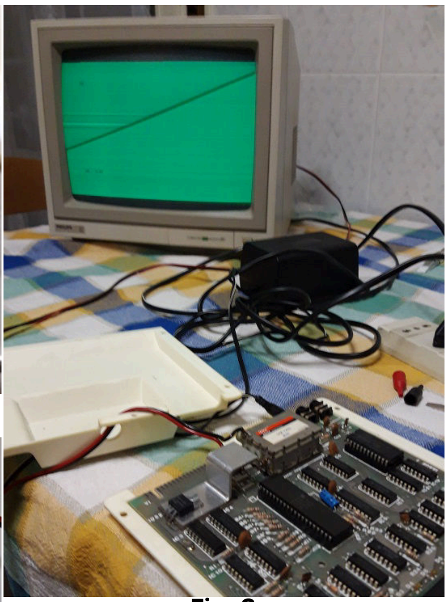


Fig. 2

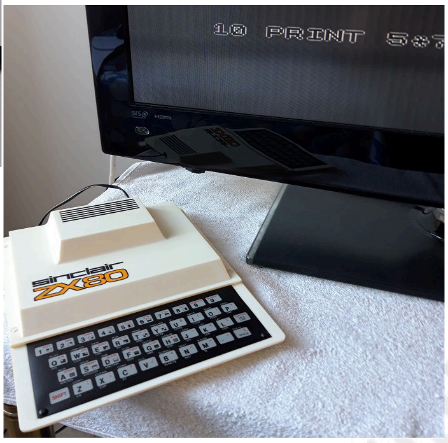


Fig. 4

WARNING
The modification described here assumes that TV and monitor are in excellent condition. The risk (not frequent, fortunately!) is to have return signals that can damage your computer. Alternatively, a protection circuit such as that proposed by the Italian magazine Sinclair Computer [Fer85] can be built.

```
Sinclair ZX80 New ROM
CPU Zilog Z80 at 3.25 MHz
ROM 8 KB (BASIC Sinclair)
RAM 1 KB (up to 16KB)
Screen: 22 rows x 32 columns
Video chip: nothing
Sound chip: nothing
Media storage: tape
```

Bibliography

[Fer85] F. Ferrario, "Intorno al monitor", Sinclair Computer #16, Sep.1985, p.45-46. https://archive.org/details/Sinclair-Computer-16

SHARP X68000

The arcade home computer from the Land of the Rising Sun

by Carlo N. Del Mar Pirazzini with the help of Takahiro Yoshioka from Japan

The Sharp X68000, often referred to as the X68k, was a home computer developed by Sharp Corporation and marketed only in Japan between 1987 and 1993.

The first model was introduced in 1987, fitting a Motorola 68000 processor like Amiga and Atari ST, but at 10 Mhz and with 1 mega Ram. It had no hard drive and some models mounted 5 inch ¼ drives (only very few).

Hardware expansions and numerous gadgets were planned. In 1993 the model was released with a Motorola 60030 at 25 Mhz, with 4 Mega Ram and an internal hard drive ranging from 80 to 120 MB. The ram of this model could be expanded to 12 mb although most applications and games did not require more than 2 Mb.

It came out on the market at a decidedly crazy cost for the period. As much as 369,000 Yen, about 3000 dollars in 1987 which at the current exchange rate would be about 6900 dollars. Despite this, it had a discreet commercial success in Japan and many publishers realized incredible porting. Due to strange commercial choices, it was never officially marketed outside the country.

As we said, the CPU was running at 10 mhz (in the first model) against Amiga's 7.16 and Atari ST. 's 8
In the original model it supported a palette of 65536 colors, against the 4096 of Amiga OCS and a maximum resolution of 1024x1024, even higher than the 1280x512 of the AGA chipset introduced on the Amiga 1200/4000. This “abnormal” resolution required a specific monitor, which had no common outputs (no vga, no rgb or anything else), but only that for Sharp.

Still in graphics, the 512k of VRAM text, 512k of Graphic VRAM and 32k of VRAM sprite made the Sharp X68k the ideal machine for arcade game porting such as Strider, Final Fight, Street Fighter 2, Ghouls n Ghost or R-Type. Tonight Capcom pushed hard for the development of his games on this machine.

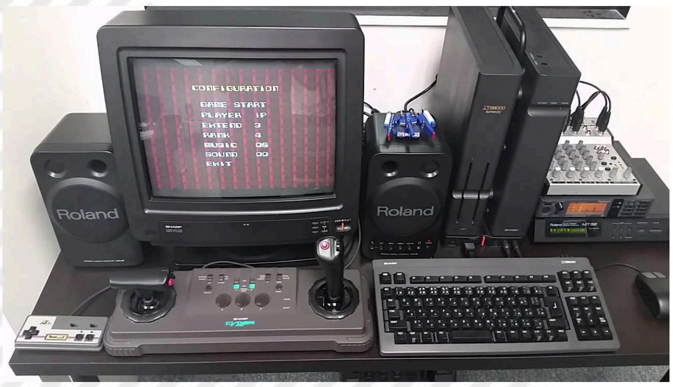


Fig. 1 - Awesome Sharp X68000 setup

Technical specifications

X68000 models: Motorola 68000 / 10MHz
XVI models: Motorola 68000 / 16MHz
X68030 models: Motorola 68030 / 25MHz

- ROM: 1 MiB (128KB BIOS,768KB Character Generator)
- RAM: 1-4 MiB (Expandable up to 12 MB)
- VRAM: 512 KiB graphics + 512 KiB text + 32KiB sprite VRAM
- SRAM: 16 KiB static RAM

- Screen Resolution:
256 x 240
256 x 256
512 x 240
512 x 256
512 x 512
640 x 480
768 x 512
1024 x 1024

- Maximum number of colors on screen: 65536
- Number of sprites: 128 sprites, 32 sprites per scanline
- Sprite size: 8 x 8 or 16 x 16
- Sprite colors: selectable from 16 different palettes, each consisting of 16 colors
- Graphics hardware: Scrolling hardware, priority control, super-impose

- Sound chips:
Yamaha YM2151 (eight-channel stereo FM)
OKI MSM6258 (one 4-bit mono PCM channel)

- Expansion: 2 card slots (4 on Pro models)
- I/O Ports:
2 MSX-compatible joystick ports
Audio IN / OUT
Stereo/3D glasses port
TV/monitor Control
RGB/NTSC Video Image I/O
Expansion (2 slots)
External FDD (up to 2)
SASI/SCSI (depending on model)
RS232 Serial Port
Parallel port
Headphone and microphone ports

- Floppy Drive:
2 5.25 floppy drives with soft ejector, 1.2MB each
2 3.5 floppy drives, 1.44MB each (compact models)

- Hard Disk: 20-80MB SASI/SCSI (depends on model)
- Operating System: Human68k (MS-like DOS developed by Hudson), SX-Windows GUI, NetBSD, OS-9
- Input Current: AC 100v, 50/60Hz

The Operating System is the Human68k, developed by Sharp itself and Hudson Soft (those of Bomberman, ndN). A similar MS-DOS system with commands similar to those of the DOS.
Other operating systems have been released, including NetBSD for 68000 processors and OS 9.

It also has a graphical interface similar to the Amiga Workbench, which allowed you to use the system "easily" and install applications and games in a more friendly way, just like the Amiga GUI.

Particular mention should be made of THE SASI bus, the forerunner of SCSI, which equipped the first versions of the system, and the audio compartment, entrusted to the Yahama YM2151 synthesizer.

Best X68k Games by Retromagazine World Staff

Final Fight – Capcom
Capcom's historic sliding hitter on Sharp is... perfect! Perfect in development, graphics, gameplay, sound. In everything.
The best conversion ever for a home system.

Bubble Bobble - Taito
Another perfect conversion. There's all the paintings, all the secrets and it's super fast. To be seen and tried.

Parodius – Konami
The funny horizontal sliding shoot-em up that we have already enjoyed on Super Nintendo, but without slow downs in this version.

Ghouls n Ghost – Capcom
The perfect conversion. There are no other words.

Street Fighter 2 - Capcom
Again, the machine has been used perfectly. The best home computer edition ever despite the terrifying US-GOLD conversion for Amiga.

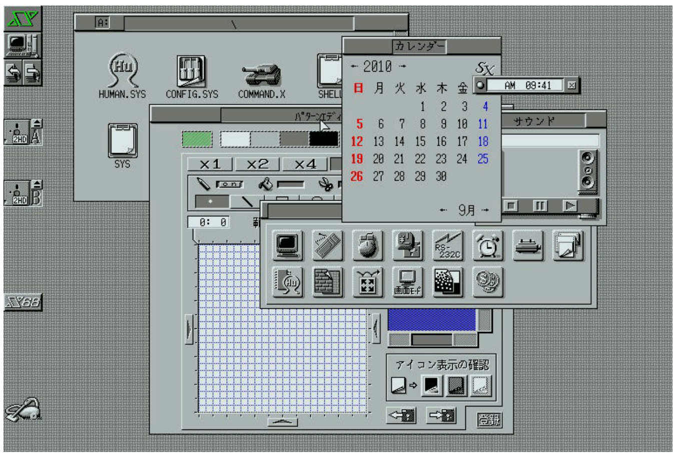


Fig. 2 - The Human68k Operative System

Where can you find it?
Finding a Sharp real hardware has a very high cost (look on Ebay), but fortunately you can also test the machine on MiSTER (FPGA), where the core is well developed, or emulate it on PC by software emulation at these addresses:

http://retropc.net/x68000/software/sharp/index.htm (in Japanese, use the translator)

https://web.archive.org/web/20171008111639/http://nfggames.com/x68000/OperatingSystems/SX-Window/

Final Fight - Capcom

Ghouls n Ghost - Konami

Street Fighter 2 – Capcom

ANBERNIC RG351P

A portable mini console to emulate consoles, arcades and 8-16 bit computers

by Francesco Fiorentini

I've been following modern portable consoles for emulating consoles and home computers with interest for some time. Last year I was very tempted by the purchase of the Anbernic RG350M, but for some reasons I was never completely convinced... Towards the end of the year (2020) the new version of this mini console was presented, the RG351... And this time I couldn't resist!

After viewing several reviews on YouTube channels, all quite enthusiastic, I made my decision and bought the console. The RG351P can be purchased from different resellers and at slightly different prices; I personally preferred not to risk waiting too long or that the console was blocked at customs (with consequent request for additional payment) and I chose to buy it through a reseller based in Europe (in Spain).

After about fifteen days of waiting, the console was delivered to me and I was able to put my hands on this desired object.

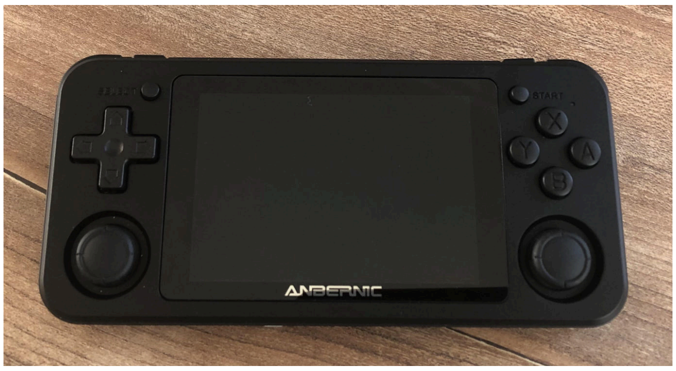
Packaging
The Anbernic RG351P will arrive in its cardboard box that vaguely resembles that of an iPhone. The box contains, in addition to the console itself, a USB C cable, an OTG USB A (female) - USB C (male) converter and the 802.11n WI-FI antenna. Some vendors also equip the console with a MicroSD card containing firmware, standard emulators (EmuELEC) and some games for some of them.
I have also chosen to buy a small bundle case; to protect the console during any trips.

Design and materials
The first impression is very positive. The RG351P is ergonomic and all buttons are adequately accessible. If I have to make a small note, maybe the keys L1, L2, R1 and R2 are a little uncomfortable to reach given the small size of the console. It probably also depends on the size of your hands, so I don't feel like reporting it as a big flaw. Additionally, for the games I want to play, they are not necessary. :-)

Although it is entirely made of plastic, its heaviness helps to give it an impression of solidity. You won't feel like you're holding a super-expensive Chinese console, but neither will a much more expensive PSP. Let's say it's a right balance, which combined with a certain sophistication in the positioning of the directional cross, the buttons and the two joypads, gives the RG351P a decidedly appealing appearance. Special mention for the two joypads, they are really well made and are very precise.

The Micro SD housing is flawed. It's located at the bottom of the console, but it's not really user friendly. To extract the Micro SD you have to press it to trigger the spring, but the space to carry out the pressure is undeniably limited and you will have to resort to a tool that helps you in the operation.

I personally use the corner of the MicroSD to SD plastic adapter and I have to admit that it is functional for the purpose.



The screen
The RG351P is equipped with a 3.5-inch screen, very bright and perfectly inserted into the structure. Someone might argue that its size is a little small, and perhaps it is true, but given the small size of the console it could not be bigger without completely upsetting the project. During my tests, however, the quality of the monitor has always proved up to the situation and on very few occasions I felt the need for a larger screen (Switchblade on Amiga). Beautiful colors and acceptable response time, I never noticed lag.

The sound

Again, the very small size of the Anbernic RG351 forced the designers to compromise. The small speakers that equip the console are located at the bottom of the console and although they work quite well, do not seem up to our beloved masterpieces. The speech is greatly improved if we plug headphones into the headphone socket. In this case, the sound is undoubtedly at another level. Try listening to Turrican's music or the main theme of IK+ on Amiga. Two true masterpieces faithfully reproduced by the Anbernic.

Emulators

As I said at the beginning, the Anbernic RG 351 P will arrive with a standard firmware containing EmuELEC, a LInux distro based on CoreELEC and Lakka and with Batocera tidbits. This distribution is more than enough to run a disproportionate amount of emulators for consoles and home computers and, for a user who wants a product ready to use and without many 'bangs', it is the ideal solution. In the SD bundled with the console you will find a partition accessible from windows where you can insert the Roms you want. Ready-to-go solution for the first-time users, so they can access the retro-emulation without too many problems.

But we want something more, don't we? Well, you know there's a distro created specifically for RG351. It is called 351ELEC and can be reached here: https://github.com/351ELEC/351ELEC.

This distribution is THE distribution for your Anbernic RG 351 P and will allow you to get the most out of the console since it can only be used with RG351P/M and is not compatible with other consoles.

The list of emulated machines is indeed impressive:

3DO
AMIGA
AMIGA CD32
AMSTRAD CPC
ARCADE
ATARI 800
ATARI 2600
ATARI 5200
ATARI 7800
ATARI LYNX
ATARI ST
ATOMISWAVE
BANDAI WONDERSWAN
BANDAI WONDERSWAN COLOR
COLECOVISION
COMMODORE 16 (PLUS/4)
COMMODORE 64
COMMODORE 128
COMMODORE VIC-20
EASYRPG
GAME & WATCH
INTELLIVISION
MAME
MICROSOFT MSX
MICROSOFT MSX2
NEC PC 9800
DAPHNE
NEC PC ENGINE
NEC PC-FX
NEC SGFX
NEC TG16
NEC TG16CD
NINTENDO FAMICOM
NINTENDO FAMICOM DISK SYSTEM
NINTENDO GAME BOY
NINTENDO GAME BOY ADVANCE
NINTENDO GAME BOY COLOR
NINTENDO ENTERTAINMENT SYSTEM
SUPER NINTENDO ENTERTAINMENT SYSTEM
NINTENDO SUPER FAMICOM
NINTENDO SUPER NINTENDO MSU1
NINTENDO VIRTUALBOY
ODYSSEY 2
OPENBOR
PHILIPS VIDEOPAC
PC (DOS x86)
PICO-8
PSP MINIS
SEGA GAMEGEAR
SEGA GENESIS
SEGA NAOMI
SEGA MASTER SYSTEM
SEGA MEGADRIVE
SEGA MEGADRIVE-JAPAN
SEGA 32X
SEGA CD
SEGA SG-1000
SEGA SC-3000
SNK NEO-GEO
SNK NEO-GEO CD
SNK NEO-GEO POCKET
SNK NEO-GEO POCKET COLOR
SONY PLAYSTATION
SCUMMVM
TIC-80
UZEBOX
VECTREX
SHARP X68000
ZX81

And these are just the perfectly implemented ones, to which you have to add NINTENDO N64, NINTENDO DS, DREAMCAST SAW, SATURN SAW, SONY PSP that are supported, but in some games they may show emulation problems.
And that's not all, to this plethora of possibilities we have to add ports, collections (CPS1, CPS2, CPS3...) and DOS games emulated through DOSBox Pure.
Well, I think there's enough material to meet all the most pressing needs and players.

For an exhaustive and updated list and related instructions (core and rom path), please refer to the official page:
https://github.com/351ELEC/351ELEC/wiki/Supported-Emulators-and-Ports

In conclusion
As you may have noticed, during the review I purposely provided only some technical features of the console, in favour of much more personal considerations. The reason for this choice is quite simple. There are hundreds of technical reviews but I wanted to do something much more personal.

In this final nail I want to maintain the same style and therefore I will tell you that the RG351P is definitely exciting to me, awakening in me the desire to play retro-games despite the little time available. With this portable console it is easy to play a game (an Amiga ones for example) in a satisfactory way and at any time, even in bed before sleep.

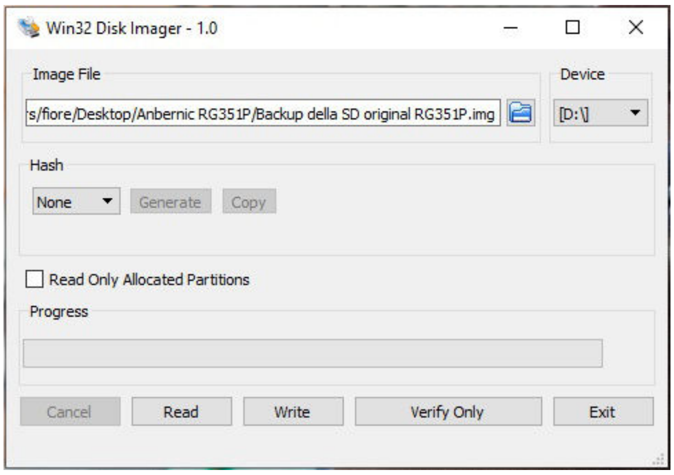
Another consideration that I want to make is related to the satisfaction deriving from ‘hacking’ our technological objects. Frequently I enjoy the most to download updated firmware, install it, configure it and make sure it works perfectly, rather than just pick-up a game and play it. This passion is widely supported and satisfied by RG351P. For a geek, retro-gamer like me, it's the best.

Personally, I can only recommend buying the Anbernic RG351P. Whatever your passion, you will find bread for your teeth.

Bonus track - Install 351ELEC and configure Amiberry emulator via Windows
I couldn't finish this mini review without providing instructions on how to emulate one of the most iconic machines in retrogaming, the Amiga. Among other things, Amiga's emulation support was one of the levers that made me decide to buy the 351P.

Before flashing a new microSD, I still recommend that you backup the one you received bundled with your console. Just in case...

- Then download Win32 Disk Imager:
http://sourceforge.net/projects/win32diskimager/
- Insert the microSD into the PC
- Name the image you want to create
- Select the Device to backup (any of those associated with the microSD)
- Press READ and wait for the operation to finish


Win32 Disk Imager - 1.0
Image File: s/fiore/Desktop/Anbernic RG351P/Backup della SD original RG351P.img
Device: [D:\]
Hash: None
Read Only Allocated Partitions
Progress

Once we have made the backup and therefore ensured that we have a lifesafer in the event of a "disaster", we proceed to install the 351ELEC firmware.

- Navigate to:
https://github.com/351ELEC/351ELEC/releases
- Download the latest release. I have installed release 1.0.8 but we are already at 1.0.10
- Flash the image on a microSD using Win32 Disk Imager or BalenaEtcher
- Wait for the image to finish writing to the microSD
- Extract the microSD and insert it into the console
- Start the console and wait for the software to self-configure and generate the Roms directories

- Extract the microSD again and insert it into the PC to copy the Roma to the appropriate directories

- NB: If you have connected the RG351P to the network you can copy the files directly via WI-FI

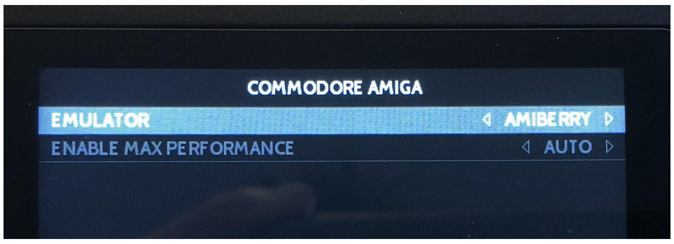
So far the procedure is common for any emulator. To emulate the Friend, however, we need to take a few additional steps.

- AMIGA games should be copied to amiga directory
- Supported formats are: .zip .uae .adf .dms .fdi .ipf .adz .lha
- I suggest creating a_kick subfolder where you copy the kickstart 1.3 rev. 34.5 (personally tested and therefore guaranteed to work) to emulate the Amiga 500

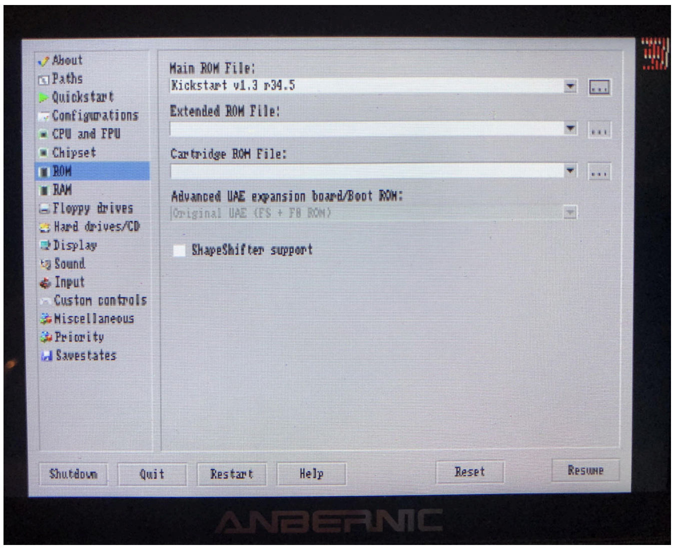
PS: do not ask us to provide kickstart as it is still protected by copyright

At this point we can insert the microSD back into the Anbernic and switch to setting up the Amiga emulator (sorry for the poor quality of the photos).

- Place yourself in the Friend's folder.
- Press SELECT to enter THE OPTIONS MENU
- Then select ADVANCED SYSTEM OPTIONS
- And on the next screen select AMIBERRY as Amiga EMULATOR


COMMODORE AMIGA
EMULATOR ◁ AMIBERRY ▷
ENABLE MAX PERFORMANCE ◁ AUTO ▷

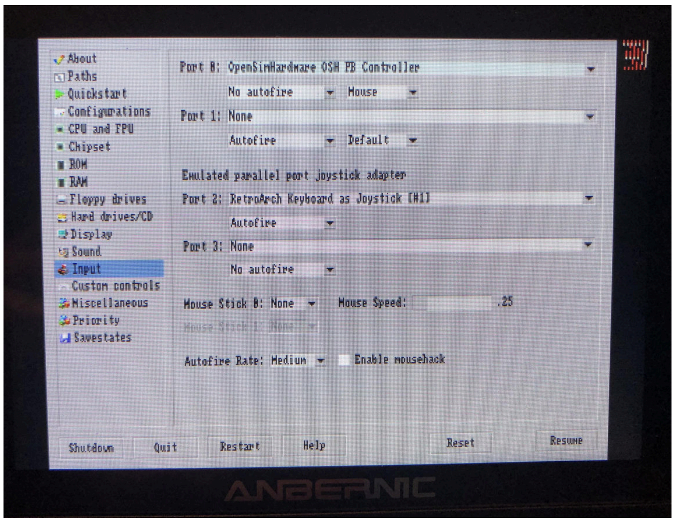
- Then launch the game; in my case Switchblade
- When the load stops at the white screen, press the SELECT + X buttons to enter the Amiberry menu
- Move TO ROMA and Main ROM File select your Kickstart 1.3 r34.5



Main ROM File:
Kickstart v1.3 r34.5
Extended ROM File:
Cartridge ROM File:
Advanced UAE expansion board/Boot ROM:
Original UAE (FS + F0 ROM)
ShapeShifter support

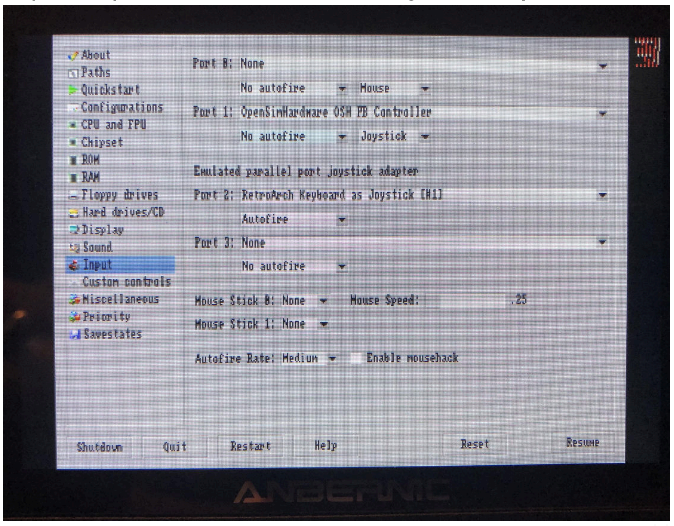
- Activate THE MOUSE using the Input menu

- Port 0: OpenSimHardware OSh PB Controller and choose Mouse



Port 0: OpenSimHardware OSH PB Controller
No autofire Mouse
Port 1: None
Autofire Default
Emulated parallel port joystick adapter
Port 2: RetroArch Keyboard as Joystick [#1]
Autofire
Port 3: None
No autofire
Mouse Stick 0: None Mouse Speed: .25
Mouse Stick 1: None
Autofire Rate: Medium Enable mousehack

- Then click Reset and wait for the game to load (it is usually very fast)
- The mouse will be used to bypass any intros that can only be navigated by mouse
- Once you have passed the intro and finished loading the game
- Press SELECT + X again to enter Amiberry menu
- And from Input select the Joystick
- Port 1: OpenSimHardware OSh PB Controller and choose Joystick (I recommend deselecting Autofire)



Port 0: None
No autofire Mouse
Port 1: OpenSimHardware OSH PB Controller
No autofire Joystick
Emulated parallel port joystick adapter
Port 2: RetroArch Keyboard as Joystick [#1]
Autofire
Port 3: None
No autofire
Mouse Stick 0: None Mouse Speed: .25
Mouse Stick 1: None
Autofire Rate: Medium Enable mousehack

- Press Resume and... Have fun!

To exit emulation press SELECT and START at the same time.

Structuring old BASIC dialects with FOR-NEXT loops

by Alberto Apostolo

In RMW 27 ITA (RMW 05 ENG) it was shown how to write programs without "GO TO" with the Sinclair Basic wired into the Sinclair ZX Spectrum ROM.
This article will show that you can write structured programs even when you have a very simplified old BASIC dialect, with the following characteristics:
1) existence of STEP clause in FOR-NEXT structure,
2) possibility to write only IF commands in the format IF [condition] THEN [single-statement],
3) inability to modify the control variables within a FOR-NEXT loop,
4) inability to use conditional expressions in arithmetic calculation,
5) impossibility to use the calculated GOSUB or the ON-GOSUB structure,
6) inability to perform POKE with system variables.

There will be three structures implemented: until, WHILE , IF-THEN-ELSE.

For the examples, the GW-BASIC will be used applying the conditions listed above.

The iterative structures UNTIL and WHILE

Fig. 1 shows the conversion of the UNTIL and the WHILE structures into FOR-NEXT loops. The STEP 0 clause allows infinite loops while placing the IF statement...THEN NEXT... interrupts an infinite loop.

Emulating the WHILE structure is more complicated. An external FOR-NEXT "protection" loop must be added that bypasses the instruction block when the condition is false. Without "protection", the BASIC interpreter could "go crazy" looking for a "NEXT F" command to close the internal loop, since it is in the THEN branch of an IF command.

Of course, in case of nested structures, it is mandatory to use control variables with different names in FOR-NEXT loops.

The IF-THEN-ELSE structure

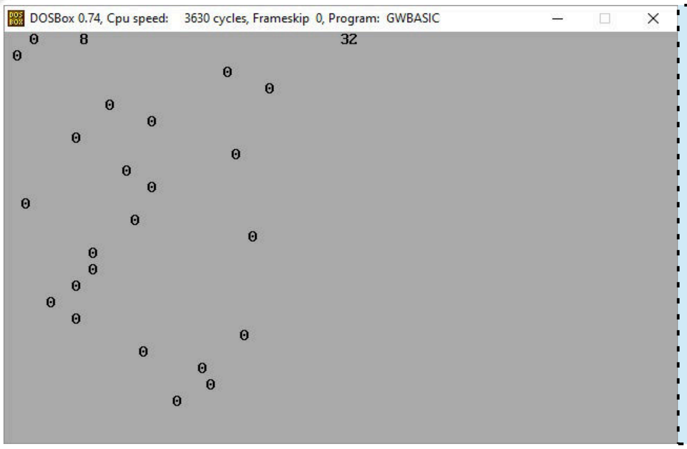
Two consecutive FOR-NEXT loops (Fig.2)

```
                  UNTIL condition
                         [statements]

 FOR F = 0 TO 1 STEP 0

     statements

 IF NOT(condition) THEN NEXT F



                   WHILE condition
                         [statements]

 LET A = 0

 IF condition THEN LET A = 1

 FOR G = 1-A TO 0
 FOR F = 1-A TO 0 STEP 0

     statements

 IF condition THEN NEXT F
 NEXT G
```

Fig. 1

```
REM IF
         LET A = 0

         IF cond THEN LET A = 1

REM THEN
         FOR F = 1-A TO 0

            statement_1_1
            ...
            statement_1_n

         NEXT F

REM ELSE
         FOR F = A TO 0

            statement_2_1
            ...
            statement_2_m

         NEXT F

REM END-IF
```

Fig. 2


DOSBox 0.74, Cpu speed: 3630 cycles, Frameskip 0, Program: GWBASIC
0 8 32

```
1000 CLS:RANDOMIZE TIMER:A=9:S=0
1010 FOR F=0 TO 1 STEP 0
1020 LOCATE 1,A:PRINT 8
1030 FOR I=1 TO 968:NEXT
1040 X=ASC(INKEY$+" ")
1050 A=A+SGN(X-49)^2-SGN(X-51)^2
1060 A=(31+A-ABS(31-A))/2
1070 A=(1+A+ABS(1-A))/2
1080 LOCATE 24,INT(RND*31)+1:PRINT 0
1090 S=S+1:LOCATE 1,40:PRINT S
1100 IF SCREEN(1,A+1)=32 THEN NEXT
```

Fig.3

"synchronized" with a working variable are used to implement an IF-THEN-ELSE structure.

Here, too, the same recommendations as discussed in the previous paragraph apply to nested FOR-NEXT structures and to the control variable specified in the NEXT command.

Write a program in a single row of BASIC

In the program in Fig.3, a car symbolized with the number "8" must avoid obstacles along the path by moving left with the key "1" and right with the key "3". The score is displayed at the top right.

On line 1040, the space character added to INKEY$ prevents an error in GW-BASIC on the ASC function when no keys are pressed and INKEY$ returns a null string.
In line 1050 the movement is managed with the keys "1" and "3" (no movement for any other key pressed). As an alternative to the ABS function (SGN(z)), you can use SGN(z)^2 because SGN(z) only takes values of -1.0,+1 (also saving characters written in the program).
Lines 1060 and 1070 respectively contain controls that emulate the MIN(A,31) and MAX (A,1) functions to delimit the movement of the car.

Writing a non-trivial program contained in a single line is more difficult and it will only be possible to separate with ":" the instructions that are on the same line.

It is suggested to use a "UNTIL NOT (condition)" structure as shown in Fig.4 (compare with Fig.1).

The program in Fig.5 is a slightly different version of the program in Fig. 3, written to be contained in a single line (255 characters is the maximum limit reserved by the GW-BASIC for a line of code).

```
2 [initialization]:
  FOR F = 0 TO 1 STEP 0:
     [statements]:
  IF condition THEN NEXT F
```

Fig.4

```
2 CLS:RANDOMIZE TIMER:A=9:S=0:FOR F=0 TO 1 STEP 0:L
OCATE 1,A:PRINT 8:FOR I=1 TO 968:NEXT:X=ASC(INKEY$+
" "):A=A+SGN(X-49)^2-SGN(X-51)^2:A=(31+A-ABS(31-A))
/2:A=(1+A+ABS(1-A))/2:S=S+1:LOCATE 24,INT(RND*31)+1
:PRINT 0;TAB(40);S:IF SCREEN(1,A+1)=32 THEN NEXT
```

Fig.5

```
2 CLS:RANDOMIZE TIMER:S=0:T=50:B$=SPACE$(59)
+"3":FOR F=0 TO 1 STEP 0:B$=RIGHT$(B$,59)+LE
FT$(B$,1):LOCATE 1,1:PRINT B$:FOR K=1 TO RND
*1500:NEXT:X=1-SGN(ASC(INKEY$+B$)-48)^2:T=T-
X:S=S+X*VAL(MID$(B$,30,1)):LOCATE 15,1:PRINT
S;T;TAB(29);S:IF T>0 THEN NEXT
```

Fig.6

The paragraph ends with another virtuosity in a single line for GW-BASIC. Fig. 6 shows a kind of "shooting range" in which pressing the "0" key hits a target that moves at random speed (3 points for each target hit).

Performance check

Emulating structured programming commands with FOR-NEXT loops complicates program writing and slows performance.

Two programs written in GW-BASIC that print 1000 prime numbers from 1 (fig. 7) were compared.

Program 1 uses the "GO TO" command while in program 2 two iterative structures nested using FOR-NEXT loops have been implemented.

Both programs ran on a laptop with DOSBox and execution times were 86 sec. and 96 sec. respectively with a slowdown of about 12%.

On other computers and with other BASIC dialects (e.g. BASIC Sinclair) the measured slowdown may have a different value.

It is up to each of us to accept this slowdown.

```
1000 REM PROG.1 : PRIME NUMBERS
1010 CLS
1020 PRINT TIME$
1030 N = 1000
1040 PRINT 1,2,3,
1050 I = 3
1060 FOR J = 1 TO N-3
1070 I = I + 2
1080 K = 1
1090 K = K + 2
1100 Q = INT(I/K)
1110 R = I - Q*K
1120 IF (K^2 < I) AND (R > 0) THEN GO TO 1090
1130 IF R = 0 THEN GO TO 1070
1140 PRINT I,
1150 NEXT J
1160 PRINT
1170 PRINT TIME$
1180 END



1000 REM PROG.2 : PRIME NUMBERS
1010 CLS
1020 PRINT TIME$
1030 N = 1000
1040 PRINT 1,2,3,
1050 I = 3
1060 FOR J = 1 TO N-3
1070 FOR F=0 TO 1 STEP 0
1080 I = I + 2
1090 K = 1
1100 FOR G=0 TO 1 STEP 0
1110 K = K + 2
1120 Q = INT(I/K)
1130 R = I - Q*K
1140 IF (K^2 < I) AND (R > 0) THEN NEXT G
1150 IF R = 0 THEN NEXT F
1160 PRINT I,
1170 NEXT J
1180 PRINT
1190 PRINT TIME$
1200 END
```

Fig.7

Bibliography

[Bas85] AA.VV. "GW-BASIC User's Manual", 1985.

[Got84] B.S. Gottfried, "Programmare in BASIC", Ed. 2, ETAS Libri, 1984, (orig. "Programming with BASIC", McGraw-Hill).

[Lam82] J.P. Lamoitier, "50 Esercizi in BASIC", Ed. 1, Gruppo Editoriale Jackson, 1982.

Interesting Links

[Koe07] J. Koelman, "ONELINERS" (ZX Spectrum one-line programs collection), retrieved in 2021/01/08 from https://spectrumcomputing.co.uk/entry/17912/ZX-Spectrum/ONELINERS

C128: redefining characters for 40 columns display

by Gianluca Girelli

As repeatedly reiterated on the pages of RetroMagazine World starting from the first issue, the possibility of going beyond the limits of the system to obtain particular effects was one of the most sought after and appreciated features by programmers of the 1980s.

This possibility is of fundamental importance even today and therefore, after explaining how to redefine the characters on the C64 and Amstrad CPC, we decided to explore the C128 in order to highlight the differences compared with other computers while consolidating the knowledge of the systems.

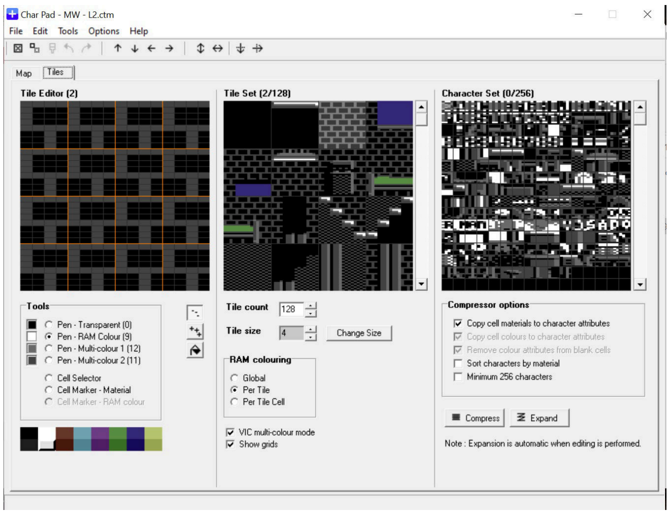


Fig. 1 - Characters redefinition in progress...

As reported by Francesco Fiorentini on Italian Issue 1, characters redefinition was not only a way to add "fonts" to computers that only had one font by default, but also a way to create graphic elements that were particularly easy to handle, especially on machines that did not have primitive graphics. For this article I chose to deepen my knowledge of the subject highlighting the implementation differences between C64 and C128 (in 40 column mode) using as a basis the "Future Character Set" originally designed for the Amstrad CPC (see, in this regard, Francesco's article on Italian Issue 24). The management of the 80-column video will be covered in a forthcoming article.

Unlike the Amstrad, which has a command dedicated to this purpose and allows a quick redefinition of even individual elements directly from THE BASIC (SYMBOL <character number>,<list of: <row>), both the C64 and the C128 perform the required task by copying the description of the characters taken from the ROM into RAM and, after redefining the set of interest, forcing the system to search for the characters at the new address in RAM.

Without going into too much detail (more information can be found in the aforementioned Issue 1), for C64 it all translates into:

- disable interrupts to prevent I/O (input/output) processes from interfering with the copy procedure;
- make the char ROM visible to the system;
- copy the characters from the ROM to the selected RAM bank;
- indicate to the system that the operations will start from the RAM;
- re enable the interrupts;
- point the system to the new address in RAM.

On the C64 the VIC-II chip expects no more than 2KB of data for the character set: this set, that can contain a maximum of 256 elements (256*8=2048), can be placed at various (contiguous) memory locations depending on our needs. The C128, on the other hand, can handle up to 4KB and therefore, unlike the C64, it is possible to redefine both sets at the same time. They are Uppercase/Graphics and Uppercase/Lowercase, each one consisting of 256 symbols as shown in the following diagram:

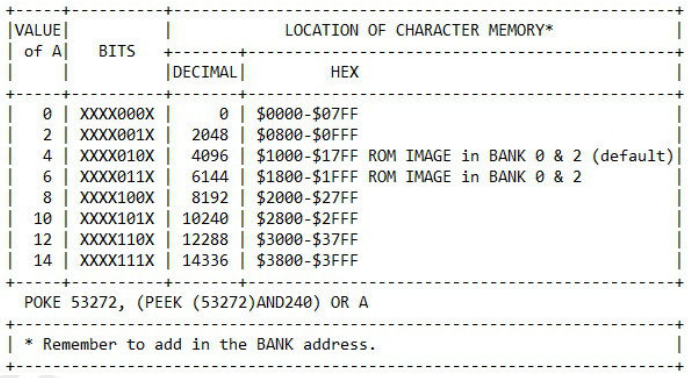
```
+-----+-------------------+---------+-------------------------------+
|     |      ADDRESS      | VIC-II  |                               |
|BLOCK+-------+-----------+  IMAGE  |           CONTENTS            |
|     |DECIMAL|    HEX    |         |                               |
+-----+-------+-----------+---------+-------------------------------+
|  0  | 53248 | D000-D1FF | 1000-11FF | Upper case characters       |
|     | 53760 | D200-D3FF | 1200-13FF | Graphics characters         |
|     | 54272 | D400-D5FF | 1400-15FF | Reversed upper case characters |
|     | 54784 | D600-D7FF | 1600-17FF | Reversed graphics characters |
|     |       |           |         |                               |
|  1  | 55296 | D800-D9FF | 1800-19FF | Lower case characters       |
|     | 55808 | DA00-DBFF | 1A00-1BFF | Upper case & graphics characters|
|     | 56320 | DC00-DDFF | 1C00-1DFF | Reversed lower case characters |
|     | 56832 | DE00-DFFF | 1E00-1FFF | Reversed upper case &       |
|     |       |           |         | graphics characters           |
+-----+-------+-----------+---------+-------------------------------+
```

By comparing the above addresses with the code published on Issue 1 (in particular line 110), readers will have noticed that the locations occupied by the characters ROM are the same as those occupied by the VIC-II chip control registers.

This is possible because they do not occupy the same positions at the same time: when the VIC-II chip has to access the character data the ROM comes into play, which in fact becomes an image inside the 16K memory bank that the VIC-II chip is "observing"; otherwise, the area is occupied by the I/O control registers.

Since the tutorial is designed to highlight these differences, I have chosen to redefine the C64 set with uppercase and lowercase characters, while for the C128 I have kept separated the two sets that can be accessed through the combination of SHIFT+C= keys (SHIFT + Commodore key). In both cases, the characters were implemented only in their basic form and not in the reverse field. The SHIFT+C= combination obviously also works on the "biscuit", but in that case only for the memory banks at the adjacent addresses $1000-$17FF and $1800-$1FFF (see diagram).

We have seen that for Commodore 64 the location chosen to store the redefined characters starts from the decimal address 14336 ($3800 hexadecimal, selected via the POKE command 53272, (PEEK (53272)AND240) OR 14). It is just one of the memory banks that you can use: there are many others, and the choice of which one to use depends on where the code of our program will later reside. Let’s now proceed to examine the problem from the point of view of the C128, remembering that the characters ROM is at the same address of the C64 ones.



VALUE of A	BITS	LOCATION OF CHARACTER MEMORY*	
		DECIMAL	HEX
0	XXXX000X	0	$0000-$07FF
2	XXXX001X	2048	$0800-$0FFF
4	XXXX010X	4096	$1000-$17FF ROM IMAGE in BANK 0 & 2 (default)
6	XXXX011X	6144	$1800-$1FFF ROM IMAGE in BANK 0 & 2
8	XXXX100X	8192	$2000-$27FF
10	XXXX101X	10240	$2800-$2FFF
12	XXXX110X	12288	$3000-$37FF
14	XXXX111X	14336	$3800-$3FFF

```
POKE 53272, (PEEK (53272)AND240) OR A
```

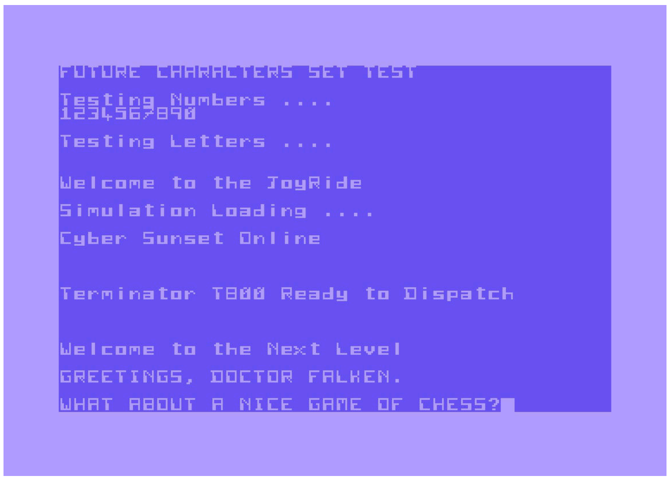
* Remember to add in the BANK address.

When Commodore decided to produce a worthy successor to C64, the engineers were faced with the problem of how to expand the available memory (both RAM and ROM). It was still an 8-bit machine which, by its very nature, cannot "see" more than 64KB at the same time. It was therefore decided to take to the extreme the concept of the division into banks, which on the C128 are 16, accessible with the "BANK" BASIC7.0 command. Without going into details about what each individual bank does (please refer to the user manual), just know that for our purposes it is sufficient:
- access desk 14 (where characters the ROM reside) and copy the chosen character;
- access bank 0 (RAM) and write in the character code;
- repeat these two operations for all the characters we need;
- at the end of the procedure, restore normal operations by selecting bank 15 (default bank).

The operation of switching from one bank to another is managed by the system and therefore, unlike the C64, there is no need to disable the interrupts.

Even in the case of C128, the choice of where to insert our redefined character map depends on what we want to do with them later. In our example, assuming we do not need to use the graphics screen, we chose to place the characters in the locations normally used by the bitmap screen (decimal addresses 8192 - 12287). As it is probably known, unlike the C64's BASIC2.0, the 7.0 is much more advanced and has many commands to handle graphic primitives. One of the first commands to use is "GRAPHIC", which can select the hi-res screen (320x200, 2 colors, both normal and shared with text), the multicolor screen (160x200, 4 colors, both normal and shared with text) or the 80-column screen (text only). A peculiarity of the "GRAPHIC" command is that, once invoked, it "moves" the text for BASIC program 8KB upwards, beyond the area occupied by the graphic screen. This means that it is sufficient to invoke any command (eg: "GRAPHIC 2,1" that allocates and "cleans" the hi-res screen) to automatically have 8KB in which to put our redefined characters. Unless you call a "GRAPHIC CLR" command, which de-allocates the dot-matrix screen by moving the BASIC text down again, simply returning to the text screen ("GRAPHIC 0") does not alter our redefined characters in any way.

After copying the characters into RAM we will have to



```
FUTURE CHARACTERS SET TEST
Testing Numbers ....
1234567890
Testing Letters ....

Welcome to the JoyRide
Simulation Loading ....
Cyber Sunset Online


Terminator T800 Ready to Dispatch


Welcome to the Next Level
GREETINGS, DOCTOR FALKEN.
WHAT ABOUT A NICE GAME OF CHESS?
```

Fig. 2 - Future Set on Commodore 64

instruct the system to take them from the new address by manipulating the following bytes:
byte 217 ($00D9):
- 0 indicates that the characters ROM is active;
- 4 indicates that the I/O block is active (and therefore characters should be searched in RAM);
byte 2604 ($0A2C):
- bit 7-4: number that multiplied by 1024 gives the start address of the video map;
- bits 3-0: number that multiplied by 1024 gives the address of the description of the characters in a block of 16K, starting from address 0.

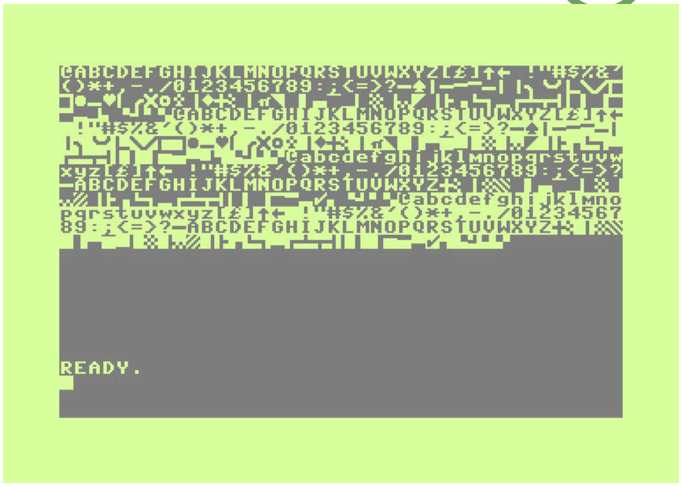
Therefore, forcing byte 217 to contain 4 the system takes the character description from RAM
to the address indicated by bits 3-0 of byte 2604.

With the C128, our BASIC code will therefore take the following form:

```
5 GRAPHIC 2,1: REM MOVES BASIC TEXT UP,
FREEING THE VIDEO MEM
9 REM LOGIC BANK 14 FROM 53248 TO 57343
11 REM GOES  TO LOGIC BANK 0 FROM 8192 TO 12287
13 FOR K=53248 TO 57343
15 BANK 14:Y=PEEK(K): REM COPIES FROM CHAR ROM
17 BANK 0:POKE K-45056,Y: REM WRITES TO RAM
18 NEXT K: REM 53248-45056=8192
19 BANK 15: REM RESET DEFAULT
21 GRAPHIC 0:REM BACK TO TEXT MODE
25 REM WITHOUT CANCELLING THE GRAPHIC
27 REM AREA WHERE WE STORE CHAR DESCRIPTION
29 REM PUTS 8 INSTEAD OF  4 IN 2604 SINCE
31 REM 8*1024=8192 STARTS OF VIDEO RAM
33 POKE 2604, PEEK(2604) AND 240 OR 8
35 REM PUTTING 4 INTO LOCATION 2017 WE TELLS
THE SYSTEM
37 REM NOT TO USE THE CHAR ROM
39 POKE 217,4
....
.... CHARACTER DEFINITION CODE FOLLOWS
....
```

In case we want to return to the default situation, we will need to add the following instructions at the end of the program:

```
55 POKE 217.0:REM RESETS CHAR ROM
```



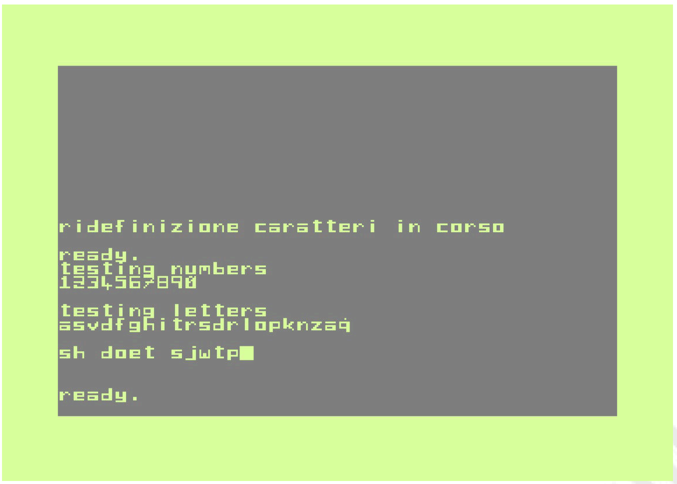
```
READY.
```

Fig. 3 - Commodore 128 - Full Charset

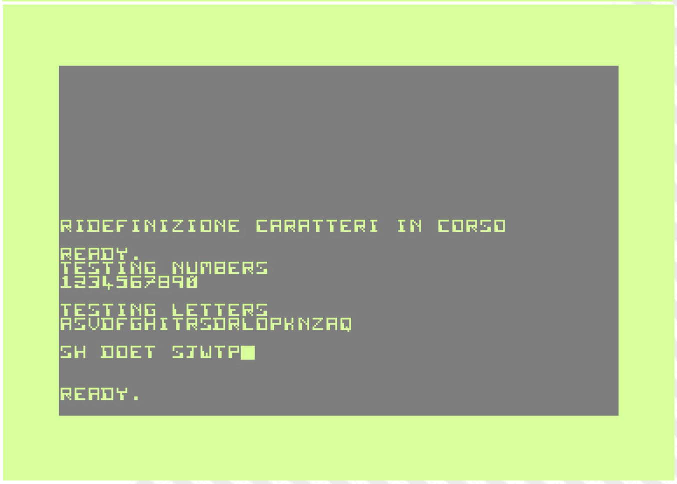
```
56 REM AND RESET THE POINTER TOWARDS  ROM
57 POKE 2604, PEEK(2604) AND 240 OR 4
```

Finally, to speed up copying and redefinition operations, you can use the "FAST" command which, by deactivating the 40-column screen, doubles the clock speed to 2MHz. The "SLOW" command will return our system to normal operationS.

At the end of this cycle of articles we then consolidated the baseICs for completely redefining character sets on three different platforms: C64, C128 (40 columns) and



```
ridefinizione caratteri in corso

ready.
testing numbers
1234567890

testing letters
asvdfghitrsdrlopknzaq

sh doet sjwtp

ready.
```



```
RIDEFINIZIONE CARATTERI IN CORSO

READY.
TESTING NUMBERS
1234567890

TESTING LETTERS
ASVDFGHITRSDRLOPKNZAQ

SH DOET SJWTP

READY.
```

Fig. 3 - Commodore 128 - Lower and Upper Case chars

CPC Amstrad. Although for complex projects it is much more convenient and faster to use modern tools (such as "CBM prg Studio" or "CharPad"), it is my opinion that you are a better programmer when you understand how the system works.

All you have to do is to launch the program you find on the following pages and start experimenting.
As always, the code has been deliberately left unoptimized for easy comprehension and readability.

For a wider practical application you can refer to the game "Alien Attack!" that Francesco has implemented both for the Amstrad CPC (RMW Num. 25) and for C64 (RMW No. 26). A later article will cover character redefinition for the 80-column screen.
Have fun!

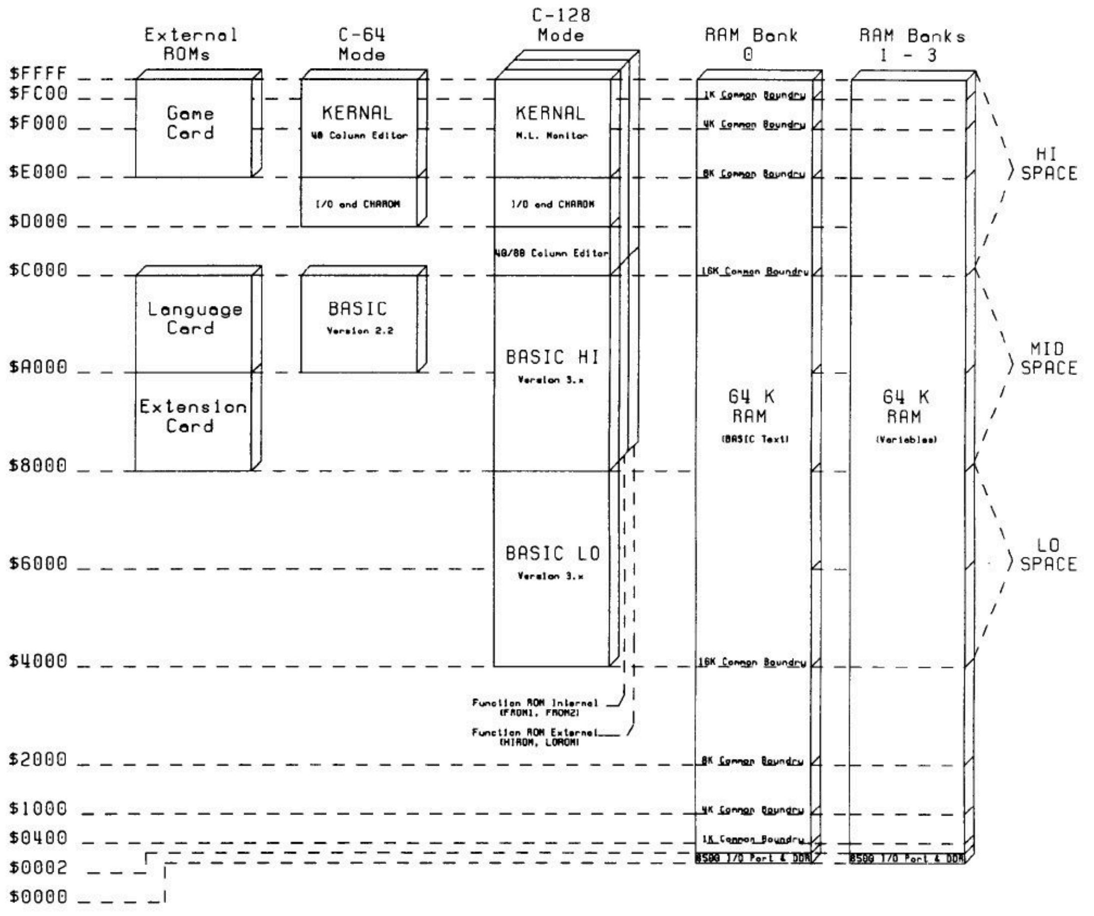
```
10 PRINT CHR$(147)
11 PRINT "FUTURE SET ON COMMODORE 64/128"
12 PRINT "ORIGINAL CODE BY PETE WHITE"
13 PRINT "POPULAR COMPUTING WEEKLY 713 AUG
1983"
14 PRINT ""
15 PRINT "TYPED AND CORRECTED BY"
16 PRINT "FRANCESCO FIORENTINI ON JUNE 2020"
17 PRINT "C64/128 VERSION BY"
18 PRINT "GIANLUCA GIRELLI, SEPT 2020 - JAN
21"
19 PRINT "RETROMAGAZINE WORLD - JANUARY
2021"
20 PRINT
"***********************************"
25 PRINT ""
30 PRINT "CHOOSE YOUR PLATFORM"
40 PRINT "1. C64"
50 PRINT "2. C128"
60 INPUT A$
70 IF A$="1" THEN GOTO 105
80 IF A$="2" THEN GOTO 410
90 PRINT CHR$(147)
100 GOTO 30
105 REM * C64 CHARACTERS REDEFINITIO CODE
106 REM ----------------------------------
110 RESTORE
125 PRINT "DISABLES IRQ TO AVOID
CONFLICTION"
130 POKE 56334, PEEK(56334) AND 254
135 PRINT "ENABLES ACCESS TO CHAR ROM"
140 POKE 1, PEEK(1) AND 251
150 PRINT "COPIES CHARS TO SELECTED RAM
LOCATION"
160 FOR I = 0 TO 255*8
170 POKE 14336+I, PEEK(53248+I): REM CHAR
ROM STARTS AT $D000 (53248)
180 NEXT I
185 PRINT "RESTORES NORMAL CHAR ROM
OPERATIONS"
186 POKE 1, PEEK(1) OR 4
195 PRINT "RE-ENABLES IRQ"
200 POKE 56334, PEEK (56334) OR 1
205 PRINT "MOVES CHAR MEM POINTER TO $3800
(DEC. 14336)"
210 POKE 53272, (PEEK (53272)AND240) OR
14:REM PRESERVES SCREEN MEM
215 REM REDEFINES UPPER CASE LETTERS
220 FOR C=65 TO 90
225 FOR I=0 TO 7
230 READ A
235 POKE 14336+I+C*8, A
240 NEXT I
250 NEXT C
260 REM REDEFINES LOWER CASE LETTERS
270 FOR C=1 TO 26
280 FOR I=0 TO 7
290 READ A
300 POKE 14336+I+C*8, A
310 NEXT I
320 NEXT C
330 REM REDEFINES NUMBERS
340 FOR C=48 TO 57
350 FOR I=0 TO 7
360 READ A
370 POKE 14336+I+C*8, A
380 NEXT I
390 NEXT C
400 END
410 REM * C128 CHARACTERS REDEFINITION CODE
411 REM ----------------------------------
420 GRAPHIC 2,1
430 REM BANCO LOGICO 14 DA 53248 57343
440 REM VA IN BANCO LOGICO 0 DA 8192 A
12287
450 FOR K=53248 TO 57343
460 BANK 14:Y=PEEK(K)
470 BANK 0:POKE K-45056,Y:NEXT K
480 BANK 15: REM RIPRISTINA DEFAULT
490 GRAPHIC 0:REM TORNA IN MODO TESTO
500 REM PONE 8 AL POSTO DI 4 IN 2604
510 REM 8*1024=8192 INIZIO DESCRIZIONE
CARATTERI
520 POKE 2604, PEEK(2604) AND 240 OR 8
530 REM PONENDO 4 IN 217 AVVISA CHE NON
ACCEDE
540 REM ALLA ROM DEI CARATTERI
550 POKE 217,4
555 PRINT"RIDEFINIZIONE CARATTERI IN CORSO"
560 REM REDEFINES UPPER CASE LETTERS
570 FOR C=1 TO 26
580 FOR I=0 TO 7
590 READ A
600 POKE 8192+I+C*8, A: REM SET 1
610 POKE 8192+I+(C+320)*8, A: REM SET 2
620 NEXT I
630 NEXT C
640 REM REDEFINES LOWER CASE LETTERS
650 FOR C=257 TO 282: REM SET 2
660 FOR I=0 TO 7
670 READ A
680 POKE 8192+I+C*8, A
690 NEXT I
700 NEXT C
710 REM REDEFINES NUMBERS
720 FOR C=48 TO 57
730 FOR I=0 TO 7
740 READ A
750 POKE 8192+I+C*8, A: REM SET 1
760 POKE 8192+I+(C+256)*8, A: REM SET 2
780 NEXT I
790 NEXT C
800 END
1030 REM UPPER CASE CHARS
1040 DATA 126,66,66,126,98,98,98,0
1050 DATA 126,66,66,126,98,98,126,0
1060 DATA 126,64,64,96,96,96,126,0
1070 DATA 254,66,66,98,98,98,254,0
1080 DATA 126,64,64, 120,96,96,126,0
1090 DATA 126,64,64,120,96,96,96,0
1100 DATA 126,64,64,102,98,98,126,0
```

```
1110 DATA 66,66,66,126,98,98,98,0
1120 DATA 60,16,16,24,24,24,60,0
1130 DATA 126,8,8,24,24,24,120,0
1140 DATA 68,68,68, 120,100,100,100,0
1150 DATA 64,64,64,96,96,96, 126,0
1160 DATA 126,74,74,98,98,98,98,0
1170 DATA 98,82,74,102,98,98,98,0
1180 DATA 126,66,66,98,98,98,126,0
1190 DATA 126,66,66,126,96,96,96,0
1200 DATA 126,66,66,98,98,106,126,4
1210 DATA 126,66,66,126,106,100,98,0
1220 DATA 126,64,64,126,6,6,126,0
1230 DATA 126,16,16,24,24,24,24,0
1240 DATA 66,66,66,98,98,98,126,0
1250 DATA 66,66,66,66,66,36,24,0
1260 DATA 66,66,66,98,106,106,126,0
1270 DATA 102,102,36,24,36,102,102,0
1280 DATA 66,66,126,16,24,24,24,0
1290 DATA 126,4,8,16,32,64,126,0
1295 REM LOWER CASE CHARS
1300 DATA 0,0,126,6,126,70,126,0
1310 DATA 96,96,96,126,98,98,126,0
1320 DATA 0,0,126,96,96,96,126,0
1330 DATA 6,6,6,126,70,70,126,0
1340 DATA 0,0,126,98,126,96,126,0
1350 DATA 60,48,48,120,48,48,48,0
1360 DATA 0,0,126,70,70,126,6,126
1370 DATA 96,96,96,126,98,98,98,0
1380 DATA 24,0,24,24,24,24,24,0
1390 DATA 6,0,6,6,6,6,6,126
1400 DATA 96,96,102,108,120,108, 102,0
1410 DATA 24,24,24,24,24,24,24,0
1420 DATA 0,0,126,90,90,66,66,0
1430 DATA 0,0,108,114,98,98,98,0
1440 DATA 0,0,126,102,102,102,126,0
1450 DATA 0,0,126,98,98,126,96,96
1460 DATA 8,0,126,70,70,126,6,6
1470 DATA 0,0,108,114,96,96,96,0
1480 DATA 0,0,126,96,126,6,126,0
```

```
1490 DATA 24,62,24,24,24,24,30,0
1500 DATA 0,0,102,102,102,102,126,0
1510 DATA 0,0,102,102,102,60,24,0
1520 DATA 0,0,66,66,90,90,126,0
1530 DATA 0,0,198,104,16,104,198,0
1540 DATA 0,0,102,102,102,126,6,126
1550 DATA 0,0,126,12,24,48,126,0
1555 REM NUMBERS
1560 DATA 26,102,110,118,102,102,126,0
1570 DATA 24,56,24,24,24,24,126,0
1580 DATA 126,2,2,126,96,96,126,0
1590 DATA 126,2,2,30,6,6,126,0
1600 DATA 96,96,96,96,104,126,8,8
1610 DATA 126,64,126,6,6,6,126,0
1620 DATA 126,64,64,126,98,98,126,0
1630 DATA 126,2,4,62,16,32,64,0
1640 DATA 126,66,66,126,66,66,126,0
1650 DATA 126,66,66,126,6,6,6,0
```

BIBLIOGRAPHY

- C64 Programmer's Reference Guide - Copyright (C) 1982 by Commodore Business Machines, Inc.
- Compute's_Mapping the 64 and 64C - Sheldon Leemon, Compute! Publications Inc 1984,1987
- C128-128D System Manual(IT) - Commodore Italiana SpA, 1985
- C128 Service Manual - CBM, Inc 1987
- C128 Oltre il Manuale - Rita Bonelli, Edizioni Jackson 1986
- Mapping the C128 - Ottis R. Cowper, Compute! Publications Inc. 1986
- Future Set on Amstrad CPC - Pete White, Popular Computing Weekly 713 Aug 1983
- Amstrad Colour Personal Computer (CPC) 464 User Manual - Copyright 1984 AMSOFT



C128 Memory Map

Installing VICE 3.5 on Raspberry OS compiling the source code

by Massimo Sanna

Tested on **Raspberry PI 400**.

VICE is pre-installed on retropie and on other distributions dedicated to emulation. But if you want only VICE on your clean Raspberry OS, it's not packaged yet and you have to do it by hand. Everything is done through console commands.

Let's start updating the system

```
sudo apt update
sudo apt full-upgrade
```

if you have updated the kernel you also need a

```
sudo reboot
```

at the end

```
sudo apt autoremove
```

to free up space from unnecessary packages

install the necessary to compile DEPUTIES

```
sudo apt install autoconf automatic build-essential byacc \
dos2unix flex libavcodec-dev libavformat-dev libgtk2.0-cil-dev \
libgtkglext1-dev libmp3lame-dev libmpg123-dev libpcap-dev \
libpulse-dev libreadline-dev libswscale-dev libvte-dev libxaw7-dev \
yasm libgtk3.0-cil-dev xa65 libsdl2-dev libsdl2-image-dev \
libgtk-3-dev libglew-dev
```

end-of-line backslashes are used to break a single command line across multiple lines. You can make a single line by deleting them. If you leave them, just tap enter immediately after the backslash and continue to the next line.

create a folder for compilation

da /home/pi

```
mkdir src
cd src
```

download the package with the sources of 3.5 (copy-paste the url below, in the browser address bar)

https://sourceforge.net/projects/vice-emu/files/releases/vice-3.5.tar.gz/download

save the file in the newly created dir 'src' (full path **/home/pi/src**)

Alternatively we can download the package with the sources in the local direcory always from the command line using curl:

```
curl -L --output vice-3.5.tar.gz URL
```

by clearly replacing “URL” with the url written above.

decompress

```
tar xzf vice-3.5.tar.gz
```

enter the directory created by decompression

```
cd vice-3.5
```

and give the following commands:

```
./autogen.sh
```

autogen configures the sources for compilation on the current environment

```
./configure --disable-pdf-docs --disable-rs232 --disable-ipv6 \
--without-png --with-sdlsound --enable-x64 --enable-desktop-files
```

./configure configures the package (features to include or exclude from compilation)

a brief explanation of the individual options

--disable-pdf-docs does not generate PDF docs to save time and disk space

--disable-rs232 disables the ability to map the emulated computer serial to a physical RS232 serial

--disable-ipv6 speaks for itself

--without-png this inhibits the ability to take screenshots (png is the default supported uico format)

--with-sdlsound enables the sdl driver for sound, if someone prefers. I prefer bracelet

--enable-x64 with this we also create the oldest x64

emulator faithful (by default create only the fastest x64sc) --enable-desktop-files adds the emulators menu with icons in the GUI menu (rasp menu at the top left)

if you would like to enable screenshots and audio/video capture:
you can remove
--without-png
(.png screenshots are enabled by default)

and add
--enable-lame --with-mpg123 --enable-static-ffmpeg

the configure run for a while, does its checks, verifies that it has all the tools and libraries required for compilation. In the end it should end without errors.
In case of errors you have to understand what blocks it by reading the handout, and take care of it. It could be, for example, a missing *-dev library. In these cases, simply install it with the command
sudo apt install missing filename-dev

in the prerequisites I should have already put everything I need, but I may have forgotten some libraries that I had already installed before. Once any faults have been repaired, restart the command./configure

After running the./configure without errors, we then start the actual compilation
make -j4

the -j4 serves to get all 4 CPU cores used to speed up compilation. Even make should end without errors.
Let's give the command
sudo make install

this command copies the compile results to **/usr/local/bin** the compilation creates all the emulators available IN VICE, accessible from command line or menu (Fig. 1).
Emulators run in windows in the graphical environment. For full-screen mode, ALT-d can be used from the booted emulator.

For everything else there is the Settings menu in each emulator. Options apply hot when possible. Otherwise they require the emulator to be reset. And you can save several and re-write them.
For all usage details, please refer TO THE DEPUTY's official

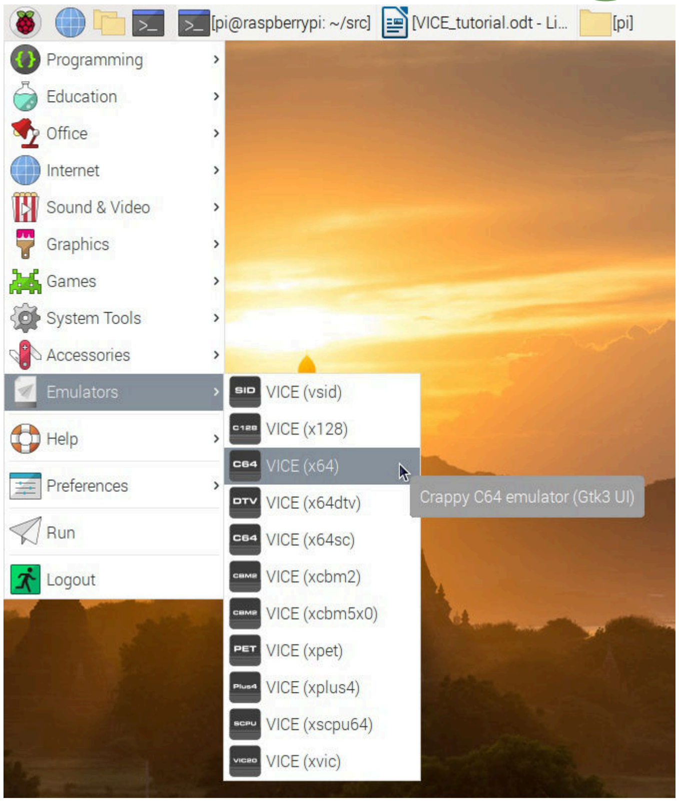
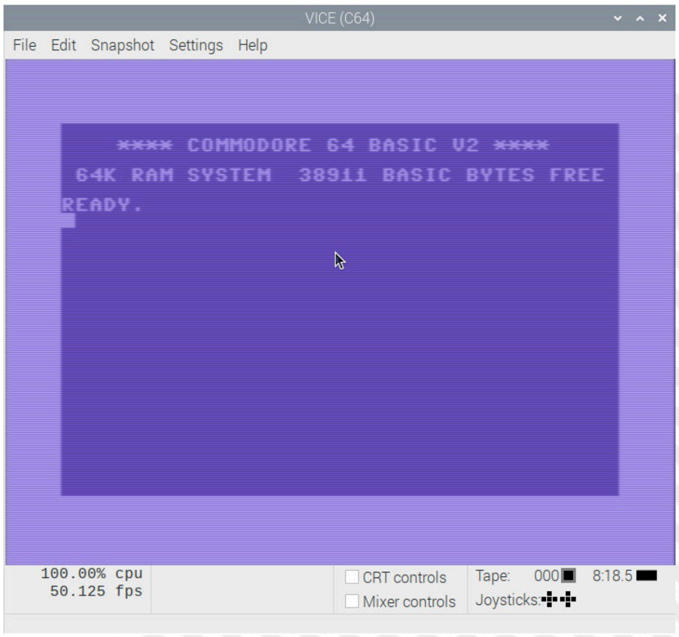


Fig. 1 - Emulators available in VICE

documentation on:
https://vice-emu.sourceforge.io/vice_toc.html

If you want to try other build configurations, just give first:
sudo make uninstall
make clean

then the new command./configure with the new parameters and finally always
make -j4
sudo make install



```
    **** COMMODORE 64 BASIC V2 ****
 64K RAM SYSTEM  38911 BASIC BYTES FREE
READY.
```

```
100.00% cpu
 50.125 fps
```

Fig. 2 - VICE (C64)

Below is the summary of my configuration with other parameters (more or less useful...) that can be passed to the **./configure**, are also indicated.

```
configure summary:
Platform canonical: armv7l-unknown-linux-gnueabihf

Architecture      : Unix
GUI              : NATIVE GTK3
Multithreaded     : yes

SCREEN/UI
---------
Hardware scaling support      : yes

SOUND
-----
FastSID support              : no  (--with/without-fastsid)
ReSID support               : yes (--with/without-resid)
New 8580 filter support      : yes (--enable/disable-new8580filter)
PortAudio sound input support: no  (--enable/disable-portaudio)
OSS sound support            : no  (--with/without-oss)
ALSA sound support           : yes (--with/without-alsa)
Pulseaudio sound support     : yes (--with/without-pulse)
SDL sound support            : yes (--with/without-sdlsound)
NetBSD/Solaris sound support  : no
MIDI support                 : no  (--enable/disable-midi)
Catweasel MK3 support        : no  (--enable/disable-catweasel)
HardSID support              : no  (--enable/disable-hardsid)
ParSID support               : no  (--enable/disable-parsid)
SSI2001 support              : no  (--enable/disable-ssi2001)
direct I/O access support     : no
lpt port access              : no
PCI utils support            : no
MP3 encoding support         : no  (--enable/disable-lame)
MP3 decoding support         : no  (--with/without-mpg123)
FLAC en/de-coding support     : no  (--with/without-flac)
Vorbis en/de-coding support   : no  (--with/without-vorbis)

SCREENSHOTS
-----------
GIF encoding support : no  (--with/without-gif)
JPEG encoding support: no  (--with/without-jpeg)
PNG encoding support : no  (--with/without-png)

VIDEO RECORDING
---------------
FFMPEG support              : no  (--enable-shared-ffmpeg/--enable-static-ffmpeg/--
enable-external-ffmpeg)
FFMPEG swscale support        : no
FFMPEG swresample support     : no
FFMPEG avresample support     : no
included shared FFMPEG support: no  (--enable-shared-ffmpeg)
included static FFMPEG support: no  (--enable-static-ffmpeg)
```

```
INPUT
-----
Mouse support               : yes
Lightpen support             : yes
Linux style joystick support: yes
BSD style joystick support  : no
Digital joystick support    : no
USB joystick support        : no

MODEM/NETWORK
-------------
RS232 device support             : no  (--enable/disable-rs232)
Network support                  : yes
RS232 network support            : yes
IPv6 network support             : no  (--enable/disable-ipv6)
Network capture/injection support: no  (--enable/disable-ethernet)

DRIVE
-----
Real device (OpenCBM support)    : yes (--enable/disable-realdevice)
X64 image support                : no  (--enable/disable--x64-image)

LIBS
----
Dynamic linking support: yes
Zlib support           : yes (--with/without-zlib)
Libieee1284 support    : no

DOCUMENTATION
-------------
Generate Info/text: yes
Generate PDF      : no (install texinfo)
Install HTML      : no

MISC
----
65xx CPU history support   : no  (--enable/disable-cpuhistory)
Debug support              : no  (--enable/disable-debug)
Threading debug support    : no  (--enable/disable-debug-threads
Embedded data files support: no  (--enable/disable-embedded)
Build old x64 emulator     : no (--enable/--disable-x64)
Gtk3 sandbox mode          : no  (--enable/disable-sandbox-mode)
Install XDG .desktop files : yes
icotool for Windows found  : no
```

How to simulate "PRINT AT" on the C64 in BASIC V2

by Attilio Capuozzo Founder of RetroProgramming Italia – RP Italia

A quick way to move the Cursor to a specific screen location identified by Row and Column is to use a specific KERNAL Routine, the C64 Operating System (OS), called PLOT.

KERNAL Routines are made in Machine Language (ML).

To securely call the aforementioned System Routine, the call should be made by jumping to an entry point of the so-called JUMP TABLE which in turn contains a JMP at the actual address of the recalled Routine; each entry point of THE JUMP TABLE then occupies 3 Bytes between the jump Opcode and the absolute 16-bit address (=2 Bytes=1 Word) of the recalled Routine.

The last 8K ROMs of the C64 Memory, from 57344/$E000 to 65535/$FFFF, host KERNAL.

KERNAL JUMP TABLE occupies addresses ranging from 65409/$FF81 to 65525/$FFF5.

The JUMP TABLE served to make calls to the System Routines independent of any updates to the Operating System (and therefore from any changes to the Effective Addresses of the KERNAL Routines).

The PLOT routine is allocated in THE JUMP TABLE at 65520/$FFF0.

The Parameters to be passed before the call through SYS, are theRow number (0 - 24) in the X Index Registry and the Cursor Column number (between 0 and 39) in the Y Index Registry. In addition, the Routine provides, before the call, the reset of the Flag C (Carry Flag) of the P Status Registry (the Status Register of 6510).

The 3 memory locations used by BASIC to save the contents of the aforementioned 3 Registers are 781 (Index Register X), 782 (Index Register Y) and 783 (Status Register P) respectively.

For example, to print a string in Row 2, Column 10, we will write the following instructions:

**POKE 781,2:POKE 782,10:POKE 783,0:SYS 65520:
PRINT"RetroProgramming Italia – RP Italia"**

Another way to position the Cursor uses location 214 (Physical Line Number of the Cursor), followed by a "blank" PRINT (i.e. without Parameters) necessary for the actual update of the Screen Editor and therefore for positioning the Cursor on the indicated Row, as well as location 211 which contains the number of the Cursor Column within the Logical Line of the Screen Editor (corresponding to 2 Physical Lines of 40 Columns each for a total of 80 Columns):

**POKE 214,2:PRINT:POKE 211,10:PRINT
"RetroProgramming Italia – RP Italia"**

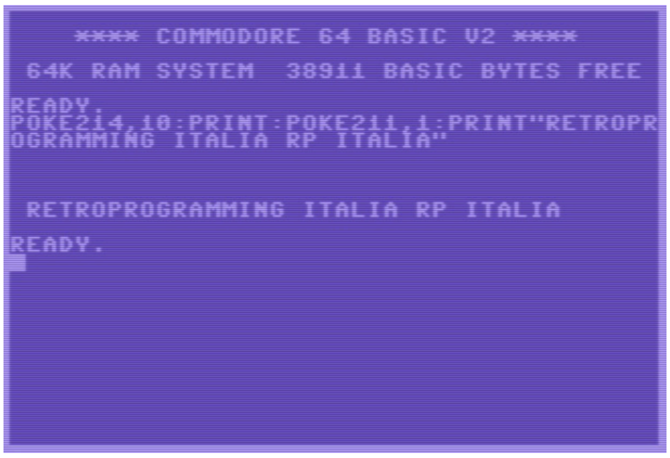
However, this technique has the disadvantage of NEVER being able to position the Cursor in Line 0 (the First Line of the Screen) due to PRINT without Parameters.

To overcome this disadvantage, you can still resort to the PLOT Routine through a "direct" call to the System Routine without going through the relative entry point of JUMP TABLE (a programming style, in fact, not recommended by the C64 PROGRAMMER'S REFERENCE GUIDE due to the loss of compatibility with other 8 Commodore bits and/or any updated/modified versions of the KERNAL):

**POKE 214,2:POKE 211,10:SYS 58640:PRINT
"RetroProgramming Italia – RP Italia"**

Although SYS 58640 corresponds to a jump to an "unsecured" entry point, in fact there are in principle no particular problems whatsoever since it is RetroComputing - and RetroProgramming - and therefore a Firmware (as we would say today) now for decades obviously no longer updated and supported.

Goodbye to the next paper...



```
    **** COMMODORE 64 BASIC V2 ****

 64K RAM SYSTEM  38911 BASIC BYTES FREE

READY.
POKE214,10:PRINT:POKE211,1:PRINT"RETROPR
OGRAMMING ITALIA RP ITALIA"


 RETROPROGRAMMING ITALIA RP ITALIA

READY.
```

Fig. 1 - Example of PRINT AT using PLOT

A formula to quickly calculate Screen Memory and Character Memory (and not only...)

by Attilio Capuozzo Founder of RetroProgramming Italia – RP Italia

Welcome back :-). In the tutorial "HOW TO CREATE CUSTOM CHARACTERS ON C64" published in 4 parts on RetroProgramming Italia - RP Italia, we analyzed in detail which bits need to be modified within the Register 53272 of the VIC-II graphics chip, to properly relocate the Character Memory RAM and/or the Screen Memory.

We have seen the Characters customizing process could also involve the Register 56576 of the Complex Interface Adapter CIA #2, if we wanted to change the current 16K module. Even the address 648 could still be implicated; it communicates to KERNAL (the C64 Operating System) where to go PRINTing the characters when we relocate the Screen Memory normally mapped in addresses from 1024 to 2023.

Not to mention, finally, the possible creation of a SAFE AREA (as explained in the aforementioned Tutorial) that involves the modification of the TOM Pointer (Top Of Memory) to the Top of BASIC RAM, i.e. the pair of 55/56 addresses in the small endian format Low Byte/ High Byte.

For this reason, it is very useful for Programmers, being able to apply universal formulas for calculating the values to POKE in the aforementioned memory addresses.

Without going any further into theoretical dissertations, let's see immediately a typical header of a BASIC V2 Program where we want to load an entire C64 Standard Characters Set and then customize them:

```
10 tb = (New Address for Top of BASIC
RAM)
20 poke 56, (tb/256): clr
30 rom =53248 (Starting Address Character
Generator ROM)
40 ba = (Memory Bank Number from 16K: 0
to 3)
50cm = (Starting Address Character Memory
RAM)
60 sm = (Starting Address Screen Memory)
70 poke 56334, peek(56334) and 254
80 poke 1, peek(1) and 251
90 for by=0 to 2047
100 poke cm+by, peek(rom+by)
110 next
120 poke 1, peek(1) or 4
130 poke 56334, peek(56334) or 1
140 poke 56576, (23-ba)
150 poke 648, (sm/256)
160 poke 53272, (sm/64) + ((cm/1024) +1)
- (ba*272)
170 print chr$(147)chr$(8)
```

The header does not need many comments as most of the topics have been analyzed in the 4 parts of the Tutorial mentioned in the introduction and to which we refer you to read.

Here we would like to point out that if we DO NOT want to change the current default 16K bank, i.e. the number 0 starting from address 0 to address 16.383, we will have to skip the program lines 40 and 140.
Similarly, if we DO NOT change the Screen Memory Starting Address, we WILL NOT type lines 60 and 150.

We always keep in mind that the Character Memory RAM must start at an address divisible by 2048 and occupies 2K of memory while the Screen Memory must have a Starting Address divisible by 1024 and occupies 1K of memory.

In addition, both Character Memory RAM and Screen Memory must reside in the same 16K Memory Bank.
We remind you each Set contains 256 characters of 8 bytes each for a total of 2,048 bytes or 2K (FOR-NEXT loop in Lines 90-110).

Finally, it is interesting to note, in the last line 170, we have entered a Control Character with ASCII/PETSCII Code 8 corresponding to disable the Combined Press Keys COMMODORE and SHIFT that allow switching from one Character Set to another on a C64 keyboard (i.e. from Set 1 Uppercase/Graph to Set 2 Lowercase/ Uppercase and vice versa).
In this way we prevent users changing our copied and customized CharSet.

In our example we used the default set 1 (Uppercase/ Graph); if we wanted to use the Standard Set 2 Lowercase/Uppercase, we could have added row 15 with a Print Chr$(14) corresponding, in fact, to the Control Character for switching to Set 2 (while the Control Character 142 switches to Set 1 Uppercase/ Graph).

That's all folks!

The group **RetroProgramming Italia - RP Italia** can be found on FaceBook:
https://www.facebook.com/groups/retroprogramming/

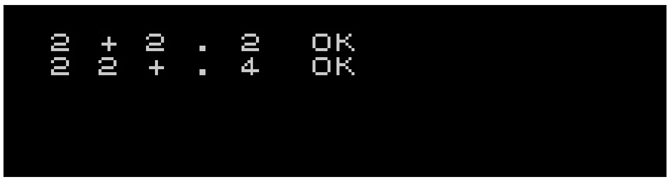
May the FORTH be with us - part two

by Francesco Fiorentini

Welcome back to our Forth learning adventure. In the last issue I realized that I had omitted the explanation of a fundamental command of this language. Perhaps the most attentive readers have already realised that I am referring to the command . (period). This command is for printing on screen; basically the dot replaces the PRINT command (abbreviated to ?) in BASIC.
WARNING: the command "." works only within a definition of a word or in a mathematical operation; you cannot use it at the command line as you would with the basic print command.

After exploring the first of the characteristics of this language, its Vocabulary and the possibility of adding words to it, we immediately move on to a second peculiarity of the Forth.

One of the easiest things to do on a computer is the sum. So let's try writing:
`. 2 + 2`
What's going on here? Our computer replies 2. It's certainly not the result we would have expected... Is Forth not able to calculate a simple sum? Of course it is. Try writing:
`2 2 + .` ***(pay attention to blank spaces)***
Our computer correctly replies 4.



```
2 + 2 . 2   OK
2 2 + . 4   OK
```

Let's try to understand the reason for this unusual syntax. The Forth language uses Reverse Polish Notation (RPN), take a look at the box on this same page for a correct definition of RPN.

RPN and Stack
Forth language makes extensive use of the stack to store numbers and operations. Imagine the stack as a stack of cards.
Let's take the first card, a 2, and put it on top of our stack. Now let's take another card, again a 2 and put it on top of the previous one. We have 2 cards containing the numbers 2 in the stack (our stack).

Reverse Polish Notation (RPN)

Reverse Polish Notation (RPN) is a mathematical notation used in mathematical formulas. Invented by the Australian philosopher and computer scientist Charles L. Hamblin, was so called in analogy with Polish Notation invented by Łukasiewicz. With RPN it's possible to carry out every kind of operation, with the advantage of not using parenthesis and the rules of precedence (first division, prima la divisione, then sum etc.).

The syntax of several pocket calculators is still RPN (users have to type the formulas in this format). Initially used to simplify machine hardware it has become a standard syntax also used by the user. In Reverse Polish Notation, also called Postfix Notation in contrast to the normal Infix Notation, first insert operands and then operators : a RPN example is 3 2 + equivalent to the classic 3+2, or 10 2 / which makes 5. When using the RPN you realize that you have a stack on which you gradually accumulate operands.

source Wikipedia

The Forth + command tells the computer to take the two cards at the top of the stack, sum them up, and write the result to a new card (back in the stack).
Command "." instructs the computer to pick up the paper at the top of the stack and print it on screen. Then it cleans the stack.
Easy? Maybe not, but it's the computer calculating system and with a little practice, it'll get familiar.

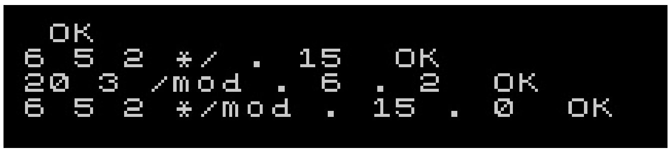
Let's look at the basic Forth operations vocabulary:

```
+ (n1, n2 + -> n1 + n2)
- (n1, n2 - -> n1 - n2)
* (n1, n2 * -> n1 * n2)
/(n1, n2 / -> n1 / n2)
MOD (n1, n2 MOD -> rest of n1/n2)
```

Nothing unusual so far, but as we may be beginning to sense, the Forth always holds surprises for us. There are special operators who add some very interesting commands to this language.
Let's see some of them together:

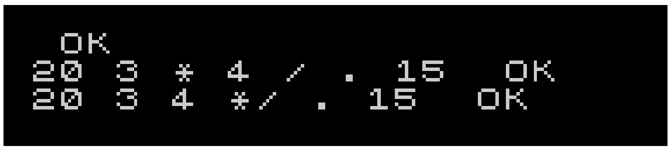
```
*/(n1, n2, n3 */- > (n1*n2)/n3)
/MOD (n1, n2 /MOD -> rest, quotient of n1/n2)
*/MOD (n1, n2,n3 */MOD -> rest, quotient (n1*n2)/n3)
```

Let's see them on the computer:



```
 OK
6 5 2 */ . 15  OK
20 3 /mod . 6 . 2  OK
6 5 2 */mod . 15 . 0  OK
```

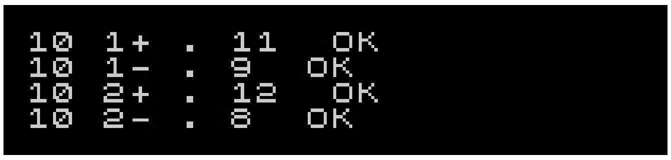
Note that the operand */ is equivalent to a multiplication followed by a division and that writing:



```
 OK
20 3 * 4 / . 15  OK
20 3 4 */ . 15  OK
```

it's exactly the same thing, but you save a character.
Note that these commands also exist:

```
1+ (n 1+ -> n + 1)
1- (n 1- -> n - 1)
2+ (n 2+ -> n + 2)
2- (n 2- -> n - 2)
```



```
10 1+ . 11  OK
10 1- . 9 OK
10 2+ . 12  OK
10 2- . 8  OK
```

Probably not really super useful, but there are and it is good to know that they exist. :-)

The words and the stack

Now that we have learned the first rudiments of this language, let's try to build some new words with the instructions just seen.

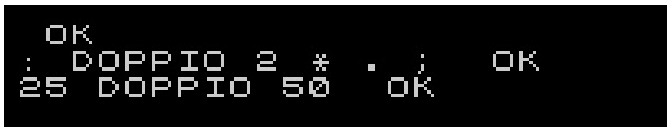
Let's try to create this simple word:

```
: DOUBLE
2 * .
;
```

You should receive an OK from your ACE Jupiter.
Now type:

```
25 DOUBLE
```

and you will get 50 as a result.



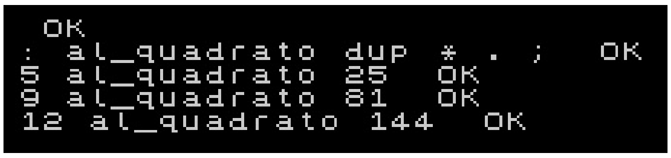
```
 OK
: DOPPIO 2 * . ;  OK
25 DOPPIO 50  OK
```

Nothing particularly exciting, but definitely interesting to understand the true potential of Forth.
For example, it immediately occurs to me that we could easily create a series of words to teach children the basic arithmetic operations.

There are also a number of commands to manage the stack, the most used are **DUP**, **DROP** and **SWAP**:
- DUP duplicates the element at the top of the stack
- DROP deletes the element at the top of the stack
- SWAP swaps the two elements at the top of the stack

Here's a practical example.

```
: squared
DUP * .
;
```



```
 OK
: al_quadrato dup * . ;  OK
5 al_quadrato 25  OK
9 al_quadrato 81  OK
12 al_quadrato 144  OK
```

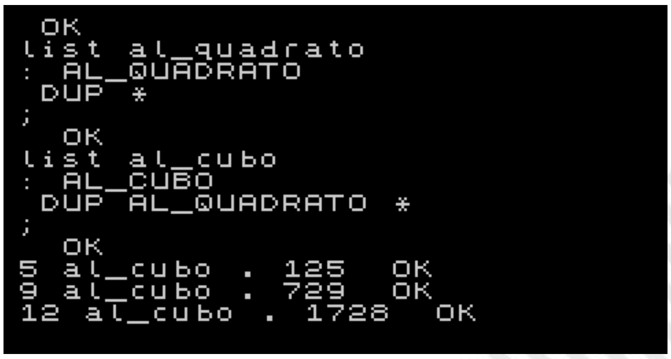
Obviously, as we learned in the last issue, we can also combine the words to build more complex ones.
Let's try to build a word to elevate a number to the cube. Begin by forgetting the previous_square to rewrite it by deleting screen printing:

```
: squared
DUP *
;
```

and then we move on to define cubed word:

```
: cubed
DUP squared *
;
```

and then we run the whole thing remembering to put the "." after the word to print the result on screen:



```
 OK
list al_quadrato
: AL_QUADRATO
 DUP *
;
  OK
list al_cubo
: AL_CUBO
 DUP AL_QUADRATO *
;
  OK
5 al_cubo . 125  OK
9 al_cubo . 729  OK
12 al_cubo . 1728  OK
```

Here we are at the end of this second episode dedicated to the ACE Jupiter Forth. Please note that you can test all the examples presented using the SpudACE emulator. See you in the next issue!

ATTENTION: Screenshots were taken in italian
- **DOPPIO** means **DOUBLE**
- **AL_QUADRATO** means **SQUARED**
- **AL_CUBO** means **CUBED**

RetroMath: Finding solutions by discretizing and iterating

by Giuseppe Fedele

In many mathematical problems, it is not possible to identify exact algorithms of resolution and it is therefore necessary to settle for approximate numerical solutions. These solutions are based on techniques of 'discretization of continuous domains' or 'iterative methods'.

Numerical integration

Discretization methods are widely used in integration problems where it is necessary to calculate the integral of a function between two extremes:

$$I_f[a,b] = \int_a^b f(x)dx$$

Consider the function in Fig. 1 and suppose to divide the integration interval into *N* subintervals. Each subinterval is of size

$$d = \frac{b-a}{N}$$

The idea of the rectangular integration is to approximate the integral, i.e. the area under the curve f(x), by the sum of the areas of N rectangles all having the same base *d* and height

$$f(x_k), \qquad k = 0, \dots, N-1$$

where

$$x_k = a + kd$$

Therefore, the integral becomes:

$$I_f[a,b] = d\sum_{k=0}^{N-1} f(x_k)$$

An improvement of the algorithm consists in considering the height of the rectangle equal to the value that the function assumes in the middle of each subinterval (Fig. 2).

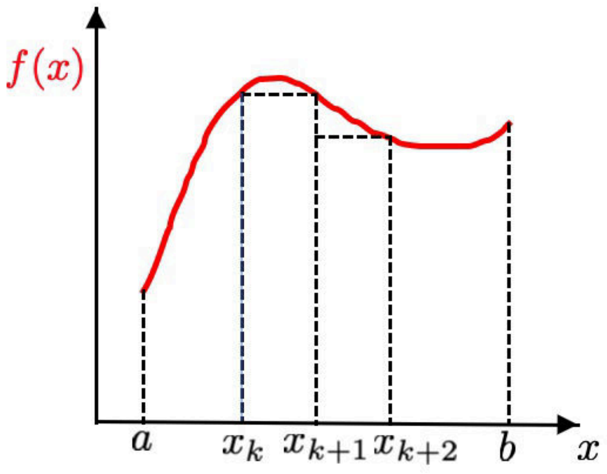


Figure 1. Rectangle Method

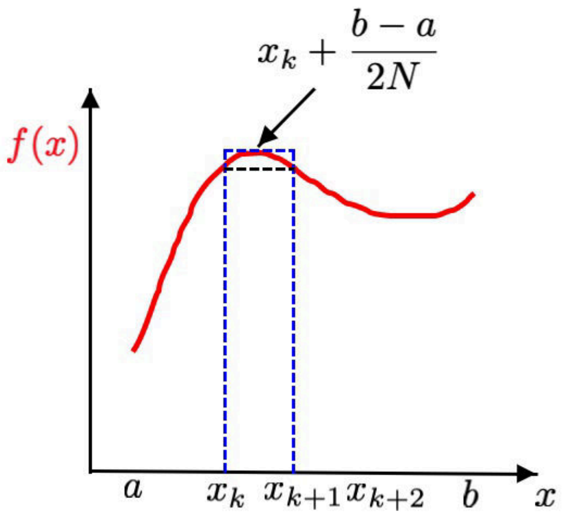


Figure 2. Change the rectangle method.

Calculation of the k-th root

Iterative methods, on the other hand, are used in the numerical calculation of some functions (now considered elementary). Any calculator has a function that calculates the square root. However, it is an approximate value whose accuracy, i.e., the number of decimal digits provided by the algorithm, depends on the number of performed iterations. The Babylonians were the first to deal with algorithms for calculating the square root.

Given a value a, its square root $\sqrt{a}$ is calculated as the limit of a succession

$$\sqrt{a} = \lim_{n\to\infty} x_n$$

where

$$x_{n+1} = \frac{1}{2}\left(x_n + \frac{a}{x_n}\right)$$

It can be shown that the algorithm converges to the desired value with exponential velocity. Another algorithm for calculating the square root of a number can be inherited from the theory of continuous-time differential equations. If we solve the equation

$$\frac{dx(t)}{dt} = -x(t)^2 + a$$

with the initial condition $x(0)$, then we will find the correct solution $\sqrt{a}$. In fact, the equation has the following equilibrium points

$$\bar{x}_1 = -\sqrt{a}, \quad \bar{x}_2 = \sqrt{a}$$

with $\bar{x}_1$ locally unstable. This means that by choosing as the initial condition $x(0) \neq -\sqrt{a}$, the solution of the differential equation will converge on $\bar{x}_2$. To implement an iterative algorithm then the equation can be discretized by approximating the derivative as

$$\frac{dx(t)}{dt} \approx \frac{x_{n+1} - x_n}{T_s}$$

with a sufficiently short sampling time T_s, thus obtaining

$$x_{n+1} = aT_s + (1 - T_s x_n)x_n$$

Iterative methods are also the basis for calculating the zeros of a generic function, $f(x)$, i.e., the values x for which

$$f(x) = 0$$

An important thing to do before solving an equation by any method is to get an idea of the trend of the function $f(x)$, to determine the number of solutions and separate each solution, i.e., to find, for each solution, an interval that contains no other solutions. The method we wish to examine requires the function to be continuous in the interval $[a, b]$ and to assume values of opposite sign at the extremes, i.e.,

$$f(a)f(b) < 0$$

Let us also assume that there is only one zero $\bar{x}$ of the function in the considered interval. The simplest method to find the value of $\bar{x}$ is the **bisection method** that proceeds by dividing, at each step, the interval into two sub-intervals determining which interval contains the solution. If we denote with a_i and b_i the extremes of the current interval, initially $a_0 = a$ and $b_0 = b$, the method computes the mean value of the interval and the value of the function at that point:

$$x_{i+1} = \frac{a_i + b_i}{2}, \qquad f(x_{i+1}), \qquad i = 0,1,\ldots$$

and proceeds by determining the sub-interval of interest according to the following rules:

if $f(a_i)f(x_{i+1}) < 0$, then $a_{i+1} = a_i$ and $b_{i+1} = x_{i+1}$;
if $f(a_i)f(x_{i+1}) > 0$, then $a_{i+1} = x_{i+1}$ and $b_{i+1} = b_i$;
if $f(x_{i+1}) = 0$, then $\bar{x} = x_{i+1}$.

The algorithm ends when $b_i - a_i < \epsilon$ where $\epsilon > 0$ is a fixed tolerance. It can be shown that the number of steps required to achieve the set accuracy is

$$n > \log_2\left(\frac{b - a}{\epsilon}\right)$$

A more efficient method for calculating the zero of a function is the **tangent method** which applies to the case where the function $f(x)$ admits derivative in the interval $[a, b]$. Starting from a x_0 (Fig. 3) the tangent line to the curve at the point $(x_0, f(x_0))$ of equation

$$y = f(x_0) + f'(x_0)(x - x_0)$$

is considered.
The point x_1 where the line intersects the x axis is obtained by setting $y = 0$ from which we obtain

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}$$

The method continues by substituting x_1 in place of x_0 and by iterating the logic. The sequence of points generated then satisfies the formula

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}, \quad n = 0,1, \dots$$

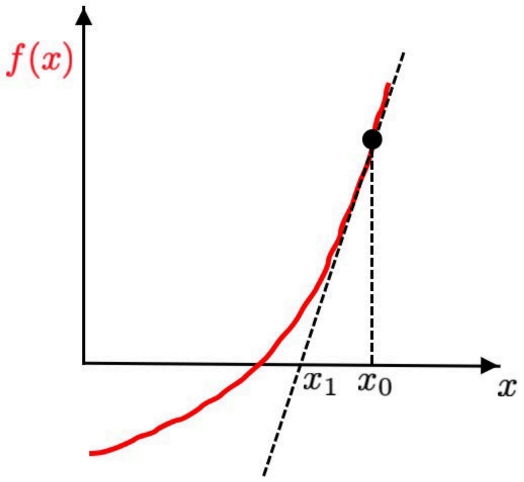


Figure 3. Bribery method.

Suppose now we want to calculate the k-root of a number *q*. We could then consider the function

$$f(x) = x^k - q$$

which has *for k* integer and for $x > 0$ only one real solution $\sqrt[k]{q}$. Applying the tangent method we obtain the iterative relation

$$x_{n+1} = x_n - \frac{x_n^k - q}{k x_n^{k-1}}$$

which provides after a sufficient number of steps the value of $\sqrt[k]{q}$.

Calculation of trigonometric functions

Iterative methods can also be used to compute trigonometric functions. Let's look at the case of the function $\sin(x)$, leaving the extension to the other functions to those who are interested.
The idea is to consider the Taylor series expansion with the initial point $x_0 = 0$ of the function $\sin(x)$:

$$\sin(x) = \sum_{i=0}^{\infty} T_i$$

where

$$T_i = (-1)^i \frac{x^{2i+1}}{(2i+1)!}$$

It can easily be obtained that

$$T_i = -T_{i-1} \frac{x^2}{2i(2i+1)}$$

from which the terms of the summation can be computed iteratively. The calculation of the function is then reduced to the sum of the ters $T_i, i = 0,1, \dots, M$, i.e. stopping the series expansion after *M* terms with *M* such that

$$\frac{x^{2M+1}}{(2M+1)!} \leq \epsilon$$

where, as usual, ϵ is the desired degree of accuracy.

In the following are reported the algorithms of integration with the rectangle method, square root calculation and bisection method implemented in Pascal Language using the online compiler https://www.onlinegdb.com/online_pascal_compiler

```
{
Integrazione con Metodo dei rettangoli
2021, Giuseppe Fedele
}
program Integrazione;
var
   N : Integer;
   a,b : Real;

function f(x:Real):Real;
begin
   f:=sin(x)
end;

function metodoRettangoli(a,b:Real; N:Integer):Real;
var
   INT,d,xk:Real;
   k:Integer;
begin
   d:=(b-a)/N;
   INT:= 0.0;
   for k:=0 TO N-1 do begin
      xk:=a+k*d+d/2;
      INT:=INT+d*f(xk);
   end;
   metodoRettangoli:=INT;
end;


begin
  writeln ('Integrazione con il Metodo dei
metodoRettangoli');
  write('a = '); readln(a);
  write('b = '); readln(b);
  write('N = '); readln(N);
  writeln(metodoRettangoli(a,b,N));
end.

--------------------------------------------
{
Radice quadrata
2021, Giuseppe Fedele
}
program RadiceQuadrata;
var
   a : Real;

function metodoBabilonese(a : Real):Real;
var
   xp,xs,acc: Real;
begin
   acc:=1e-6;
   xs:=0.5;
   repeat
      xp:=xs;
      xs:=(xp+a/xp)/2;
      {writeln(xs);}
   until abs(a-xs*xs)<=acc;
   metodoBabilonese:=xs;
end;

function metodoOde(a: Real):Real;
var
   xp,xs,Ts,acc: Real;
begin
   Ts:=1e-6;
   acc:=1e-6;
   xs:=0.5;
   repeat
      xp:=xs;
      xs:=a*Ts+(1-Ts*xp)*xp;
      {writeln(xs);}
   until abs(a-xs*xs)<=acc;
   metodoOde:=xs;
end;
```

```
begin
  write ('a = '); readln(a);
  writeln('Metodo Babilonese');
  writeln('-----------------');
  writeln(metodoBabilonese(a));
  writeln('Metodo ODE       ');
  writeln('-----------------');
  writeln(metodoOde(a));
end.

-------------------------------------------
{
Metodo di bisezione
2021, Giuseppe Fedele
}

program Bisezione;
var
   a,b,xs,acc : Real;

function f(x : Real):Real;
begin
   f:=1-x
end;

begin
  a:=0;
  b:=1;
  acc:=1e-6;

  while (b-a)>acc do
  begin
    xs:=(a+b)/2;
    if f(a)*f(xs)<0 then
       b:=xs
    else
       a:=xs;
  end;
  writeln('zero : ',xs);
end.
```

BIBLIOGRAPHY

[1] A. Quarteroni, R. Sacco, F. Saleri, P. Gervasio
Matematica Numerica
Springer, 2008

[2] R. Bevilacqua, D. Bini, M. Capovani, O. Menchi
Metodi Numerici
Zanichelli, 1992.

INTRODUCTION TO AREXX – part 5

by Gianluca Girelli

GAME CODING WITH AREXX - PART 2 of 2

Let's close this overview of the ARexx language by publishing the second and last part of the tutorial on how to implement a text adventure. Please note that to better understand the topics that will here be covered please also refer to the previous parts published on RetroMagazineWorld ENG, as well as the article "Introduction to Game Coding" published on issue 17 (Italian version only).

By the end of this tutorial we will have learned all of the basics for building a game-engine for text-adventures: switching from one genre to the other will be "just" a matter of changing the setting of the story (eg: western, rather than cyberpunk) and the interactions on objects that drive the story progress.

1. EVENTS HANDLING

In past issues we have learned how to physically build the game world (for simplicity we have decided to represent it as a two-dimensional matrix), how to describe its locations (also store into a second matrix, complementary to the first), how to navigate in this world (management of variables of movement) and how to prepare to interact with it (implementation of a simple syntactic parser). What we now need is to figure out how to actually handle events (i.e. make them happen) in order to progress in the story-line. Let's consider, for example, this situation: we are inside a pawn shop and in one window is exposed valuable hardware, essential for "living" in the cyberpunk world we are building.

The classic mechanics behind a text-adventure is: read (or reread with an appropriate command) the description of the enviroment; examine the place in detail to understand if there are objects to be collected or interacted with; take (use) said objects; if taken, examine them for any hidden information. Translated into code:

```
/*-------------------------*/
/*     LOOK                */
/*-------------------------*/
Look:
  call Scenario(Pos)
  say
return
```

N.B. The implementation of the Scenario() routine (which prints out the description of the current location) has been published on the previous issues.

As we learned, the penultimate line of the description of each location (contained in the aforementioned matrix) may contain important information, in this case:

Situation.6.34='In one of the windows you see your "Ono-Sendai" cyberspace DECK, model UXB 7000.'

Since we spot the object, let's take it:

```
/*-------------------------*/
/*     TAKE                */
/*-------------------------*/
Take:
.....
.....
if Pos=34 & name='DECK' then do
        og_num.1.1=1
        say 'Since you had the receipt with
you, Shin gives you the DECK back, but not'
        say 'before having "withdrawn" 50
credits from your pockets!!'
        og_num.5.1=0
        og_num.6.1=0
        Situation.6.34=''
        end
if Pos~=34 & name='DECK' & og_num.1.1=0 then
say 'A receipt found in your pocket says
your DECK is at Shin''s.'
.....
.....
return
```

Please note that the variables used are:
og_name.1=deck
og_name.5=credits

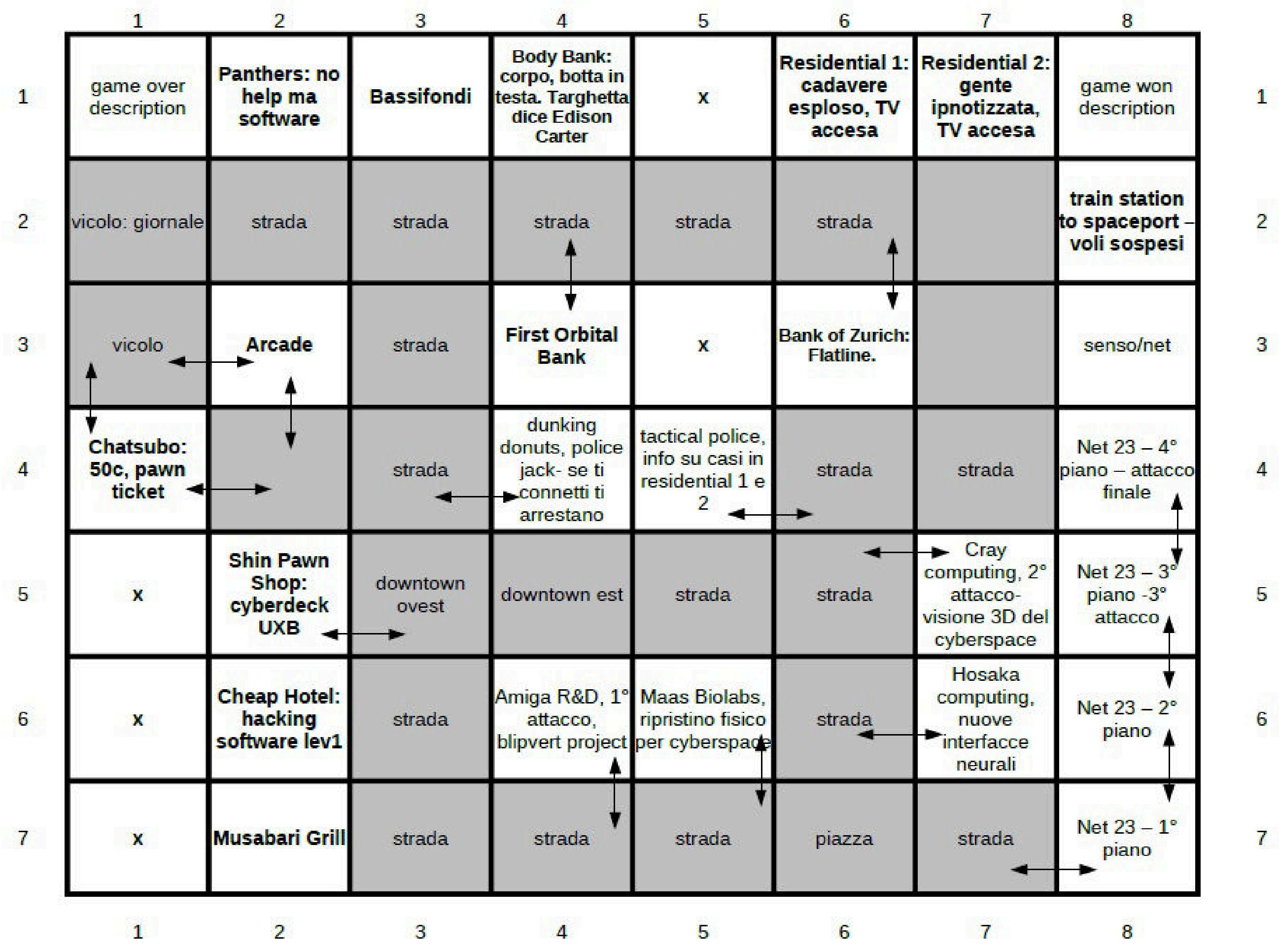

game over description
Panthers: no help ma software
Bassifondi
Body Bank: corpo, botta in testa. Targhetta dice Edison Carter
x
Residential 1: cadavere esploso, TV accesa
Residential 2: gente ipnotizzata, TV accesa
game won description
vicolo: giornale
strada
train station to spaceport – voli sospesi
vicolo
Arcade
First Orbital Bank
Bank of Zurich: Flatline.
senso/net
Chatsubo: 50c, pawn ticket
dunking donuts, police jack- se ti connetti ti arrestano
tactical police, info su casi in residential 1 e 2
Net 23 – 4° piano – attacco finale
Shin Pawn Shop: cyberdeck UXB
downtown ovest
downtown est
Cray computing, 2° attacco-visione 3D del cyberspace
Net 23 – 3° piano -3° attacco
Cheap Hotel: hacking software lev1
Amiga R&D, 1° attacco, blipvert project
Maas Biolabs, ripristino fisico per cyberspace
Hosaka computing, nuove interfacce neurali
Net 23 – 2° piano
Musabari Grill
piazza
Net 23 – 1° piano

Fig. 1 - Map of the game

og_name.6=receipt

Furthermore, thanks to the power and flexibility of ARexx in managing variables, we have implemented a construct that, if og_name.1='deck' then: og_num.1.1=1 means "deck taken", og_num.1.2=1 means "deck examined" while og_num.1.3=1 means "deck used".

Without going into too much detail, our plot begins with the protagonist having only two objects: one receipt from a Pawn Shop and some loose change. The script predicts that if the "take deck" command is used inside location 34 (Shin's Pawnshop, please check out the map in the figure) the protagonist is given back his "cyberspace deck" in exchange for the receipt and obviously some money. Any other input that uses the word "deck" outside the location 34 will inform the player that the object is somewhere else, as long as the protagonist still has the receipt in his pocket, otherwise the object has obviously already been taken. Finally, it's important to note how the Take() routine also takes care of resetting line 6 within the description of the current location (Situation.6.34 = '') thus when returning to Shin, or using the "Look" command, the deck inside the shelf will no longer be mentioned.

Having received the object, it is therefore time to examine it closely:

```
/*-------------------------*/
/*     EXAMINE            */
/*-------------------------*/
Examine:
select
....
....
   when name='DECK' & og_num.1.1 then
      do
        say 'It''s your old cyberspace deck,
Model Ono-Sendai UXB 7000. You can use it
to connect to a JACK.'
        og_num.1.2=1
        if regeneration=0 then say 'You're
not ready yet.'
```

```
     end
....
....
otherwise say 'You don''t have this object
with you.'; say; return
end
return
```

We now have the deck (og_num.1.1=1, initialized in the "take" routine) and we have examined it (og_num.1.2=1, initialized in the "examine" routine). Depending on the state of other variables, there are things that we can already do or not. Later in the story, once the hardware is retrieved, the player will also need to retrieve adequate software which, however, even when combined with the deck, will not allow the protagonist to enter the cyberspace until he heals his neural damage (ie: until the "regeneration" variable is initialized to 1).

Now suppose that, after visiting some locations, we found a disk containing the software we need. We are obviously in need of a routine that allows us to use it:

```
/*-------------------------*/
/*      USE               */
/*-------------------------*/
Use:
....
....
if name='DISK' & og_num.8.1 & og_num.1.1 then do
                                             say
                                             say
'You put the DISK in your DECK and loaded
the software.'
                                    og_num.8.3=1
                                          end
else do
     if (name='DISK' & og_num.8.1=0) then
say 'You don''t have the DISK with you.'
     if (name='DISK' & og_num.8.1 & og_num.
1.1=0) then say 'You don''t have a DECK where
to load it.'
    end

if name='DECK' & og_num.1.1 then say 'You
can''t use the DECK as a stand-alone device.
You have to find a JACK.'
if name='DECK' & og_num.1.1=0 then say 'You
don''t have this object with you.'

if name='JACK' then call Combat(Pos)
....
....
return
```

As you can guess, if the disk is in our inventory (og_num.8.1=1) it will be loaded into the deck (og_num.8.3=1).
If an element is missing (the deck or the disk itself) the procedure will handle the relevant exception. Using the right object in the right place is the key to solve text-adventures so the routine "Use" also takes care of invoking (if the conditions allow it) the "combat" routine. It may not always be necessary and it can take on various forms. In our case it deals with simulating a hacking attack to the system simply by printing the appropriate messages on the screen.
Using the "hack_1_disp" and "first_attack" variables helps us to keep track of the situation to monitor the progression of events.
Other variables with similar syntax and functions will be initialized as the player discovers new objects that will allow him to upgrade his/her equipment and progress the story until its conclusion.

```
/*-------------------------*/
/*   COMBAT (hacking)      */
/*-------------------------*/
Combat:
select
  when Pos=44 & hack_1_disp then do          /
* Amiga R&D */
                say clear
             say 'Firts Level Hacking initiated'
                call Sleep()
                say
                say 'Accessing CyberSpace.
Breaching in progress ....'
                 say 'Data found on: BLIPVERT
PROJECT'
                 say 'The experiments carried
out by this Research and Development Division
confirm '
                say 'the individual's positive
response to subliminal cues. '
               say 'Hiding a photo of a product
inside a movie, the subject will feel a strong'
```

```
               say 'impulse to get it. We
have the technology to compress the message '
               say 'all along the carrier
signal of the TV to send a continuous subliminal
message..'
               say
'============================================
==='
               say 'Hacking complete.
Disconnecting from CyberSpace...'
               first_attack=1
         end
....
....
 otherwise say 'You''re not ready yet.'
end
return
```

2. TRACKING THE PROGRESSION OF STORY-PLOTS

Our work is almost done: the only thing still missing is a way to keep track of the progress made by the player so that he/she can advance through the story-line as the assigned tasks are completed.
Since this software was developed basically for educational purposes, the procedure we need will be very simple. Its level of complexity, however, can be freely increased according to the most different needs. Let's see an example:

```
/*--------------------------*/
/*    GAME PROGRESS        */
/*--------------------------*/
GameProgress:
 if regeneration & og_num.1.1 & og_num.8.3
then do

hack_1_disp=1
                                          say
'=============================='
                                          say
'You are now ready to enter Cyberspace. First-
level hacking is now available.'
                                          say
'=============================='
                                       end
 if first_attack & og_num.3.3 then do
                                 hack_2_disp=1
                                          say
'=============================='
                                          say 'Now
you are ready to enter 3D Cyberspace. Second-
level hacking is now available.'
                                          say
'=============================='
                                        end
....
....
return
```

GameProgress() works in conjunction with other game routines and aggregates the results they produce to enable new options, thus allowing you to advance in the story. We saw in the previous paragraph that the player, after having retrieved adequate hardware (deck), software (disk) and be healed from neural damage (regeneration), had reached the minimum requirements to progress in the game. This was achieved thanks to the first IF-THEN construct of this routine which, after having verified the existence of said requirements, initialized the "hack_1_disp" variable to "1". If you go back and take another look to the code of the Combat() routine it will be very clear why the "first_attack" variable was inserted at the end of the first attack: it is one of the prerequisites for the second level. The other one is to find and use a new object (in our case a new neural interface, og_num.3.3), which will enable us to the next hack.

GameProgress() could do many other things, such as updating the score system (just in case there is one, as in the old Infocom adventures) or assign "achievements" and "trophies" (as for the XBox and PlayStation). As the complexity and length of our adventure increase, new constructs will be needed inside our code . The reason is due to the fact that the original content of all the variables (and in this case the integral content of the matrices describing the game world) must always be preserved.
Otherwise, after the first save, those information will be irretrievably lost (since they will be overwritten) and there would be no way to play a second match. One solution to the problem is to expand the matrix of descriptions in a way transparent to the user by adding additional descriptions in locations not reachable by the player. In this way the software will always be able to load the original world if requested thus preserving the original game data over time; GameProgress() will contain dedicated instructions that will modify the game world only during runtime. In

our case, for example, the routine could contain a statement like:

if og_name.1 = 1 then Situation.6.34 = 'The shelf is now empty. ".

Since GameProgress() is called on every loop we are sure it will faithfully reflect our progress without altering the original construct.

3. INITIALIZATION AND MAIN LOOP

After having broken down the game engine into all its parts and having built them one by one, I report below the entire initialization sequence and the main loop of the game, entirely taken from the adventure source code:

```
/*==================== GAME INIT
=======================*/
do until (answer='NO' | answer='N')

  call Levels()          /* sets up level
descriptions */
  call Variables()       /* initializes game
variables */
  call Titles()          /* initial splash-
screen routine */
  call Instructions()    /* displays how to
play */
  call Scenario(Pos)     /* current player
position and location description  */
  options prompt ">"

/* main routine start */

do until (phrase='QUIT')
  say
  request=random(1,4,time("E")*100)  /*
random number between 1 and 4 included */
  say prompt.request
  pull phrase
  call Parser(phrase)
  select
        when verb='GO' then call Go(name, Pos)
        when verb='LOOK' then call Look()
        when verb='TAKE' then call Take(name)
      when verb='EXAMINE' then call
Examine(name)
        when verb='USE' then call Use(name)
        when verb='LIST' then call List()
        when verb='VOC' then call Vocabulary()
        when verb='DEBUG' then call Debug()
        when verb='VAR' then call Var(name)
        when verb='QUIT' then nop
        otherwise say 'I don''t know the
meaning of ' || verb
  end
  call GameProgress()
  if ~treasure_found then
number_of_moves=number_of_moves+1
end

  say 'another match?'
  pull answer
end
/* main routine end */

say clear
say 'thank you for playing'
say
say 'you made ' number_of_moves 'moves'; say
do j=1 to 7
  total=total+score.j
end
say 'your''score is:' total; say
if treasure_found~=1 then say '....but you
didn''t uncover the truth behind the BlipVert
Project!'
call Sleep()
call Credits()
exit
/*========================== GAME END
=======================*/
```

As you can see, having broken down the problem into its elementary parts allows us to write a main loop extremely easy to read and maintain: after the initialization phase you enter the game loop, inside which the parser will direct actions by calling the appropriate routines according to user input. At each cycle GameProgress() manages the player's progress. Upon exiting the game, the user will be notified of his/her score and the credits will be displayed.

4. ADDITIONAL ELEMENTS FOR THE GAME CODE

As mentioned earlier this text-adventure game-engine was developed years ago for educational reasons and that's why it lacks some elements that would be mandatory if you'd want to pack a product with (at least) a semi-professional appearance. The most important thing is definitely the implementation of the loading and saving routines: it is in fact unreasonable to expect the user to reach the end of the adventure quickly and without the need for intermediate stops. Another important thing is the localization of the software: the parser, as well as the files containing the locations descriptions, should therefore be developed together with a set of supporting variables that allows to avoid the use of "hard-coded" strings and therefore, as an example, the "examine" routine should be written like this:

```
/*--------------------------*/
/*     EXAMINE            */
/*--------------------------*/
Examine:
select
....
....
   when name=var_deck & og_num.1.1 then
      do
        say string1
        og_num.1.2=1
        if regeneration=0 then say string2
      end
....
....
otherwise say string3; say; return
end
return
```

Where "var_deck" contains the name of the object, while "string1", "string2" and "string3" contain the descriptions of the actions. These strings would be stored in the specific localization file for the chosen language, available from the main pre-game menu.

Finally, we may want to expand the game with side -uests, a better score system or a wider range of words in our vocabulary which, combined to a more refined parser, could provide the user with a more immersive experience.

5. CONCLUSIONS

The journey was a long one but we finally reached our destination. I hope you enjoyed reading these article as much as I enjoyed designing the game and writing about it. The game code is too long to be attached to this article, but the entire source code is available on request from our official website. To play it, simply import it into your favorite emulator or, even better, onto a real hardware if you are a lucky owner. Any version of AmigaOS, starting from version 2, will do.

BIBLIOGRAPHY

- Mike Cowlishaw "The REXX Language: A Practical Approach to Programming" (1985) Prentice Hall. ISBN 0-13-780651-5.
- Chris Zamara, Nick Sullivan "Using Arexx on the Amiga" (1991) Abacus Software Inc. ISBN 1-55755-114-6.
- AmigaOS 2.0 System Manual
- http://it.wikipedia.org/wiki/Game_engine
- http://it.wikipedia.org/wiki/Porting

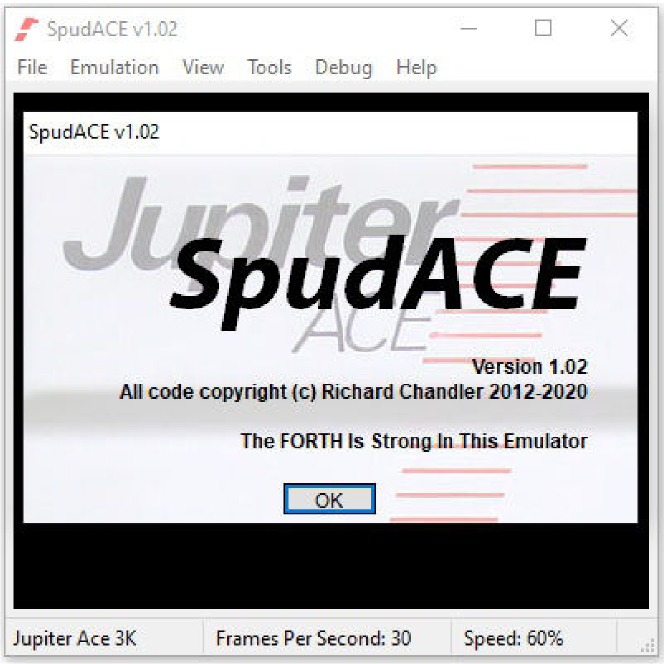
Using the examples:

Remember that to use the examples you need to save the script in text mode in the format "script_name.rexx".
To run the script just type from shell
"> rx script_name.rexx"
or, more simply,
"> rx script_name".

Emulators: **SpudACE** versione 1.02

by Francesco Fiorentini

We had been talking about it in the editorial staff for a while and in fact I had also mentioned it in the first issue of RetroMagazine in 2017: a section dedicated to emulators. There are always many topics to discuss and for one reason or another we have always delayed talking about emulators. Until today! Welcome to a new section dedicated to the world of emulation.


SpudACE v1.02

File Emulation View Tools Debug Help

SpudACE v1.02

Version 1.02

All code copyright (c) Richard Chandler 2012-2020

The FORTH Is Strong In This Emulator

OK

Jupiter Ace 3K | Frames Per Second: 30 | Speed: 60%

SpudACE - version 1.02

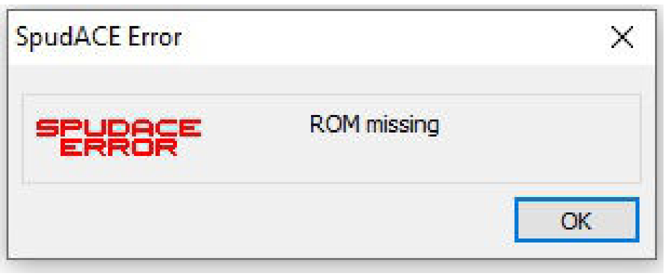
SpudACE is an ACE Jupiter emulator created by Richard Chandler and written entirely in C++.

You're probably wondering why I chose this emulator to open this section. Easy to say; I had to start somewhere and considering that I am carrying on a series of articles dedicated to the Forth of Jupiter ACE, talking about its emulator seemed like a good idea to me. In addition, this computer is little known in Italy, so I suppose the SpudACE emulator is too.

The emulator can be downloaded from: http://www.jupiter-ace.co.uk/downloads/SpudACE[V102].zip
Version 1.02 is from June 2020.

Installation

Installing SpudACE is very easy. Simply unpack the zip file and copy all the contents to a folder. Caution: The emulator looks for the ace.rom file in a subfolder called roms. If it doesn't find it you will receive this error:


SpudACE Error

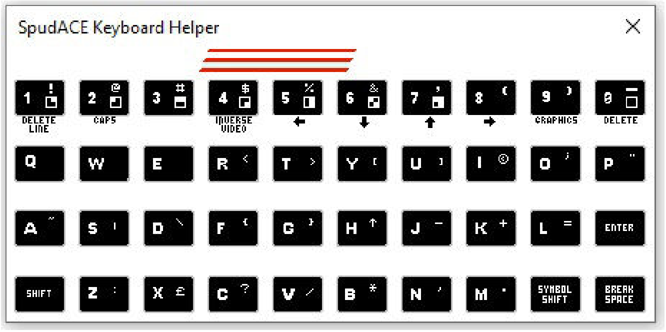
SPUDACE ERROR

ROM missing

OK

Key features of the emulator:
- 99.45% accurate Z80 emulation
- accurate sound emulation
- drag and drop support
- supports WAV, TZX and TAP cassette formats
- built-in debugger
- saving on ACE, TAP or WAV formats
- supports uploading text files

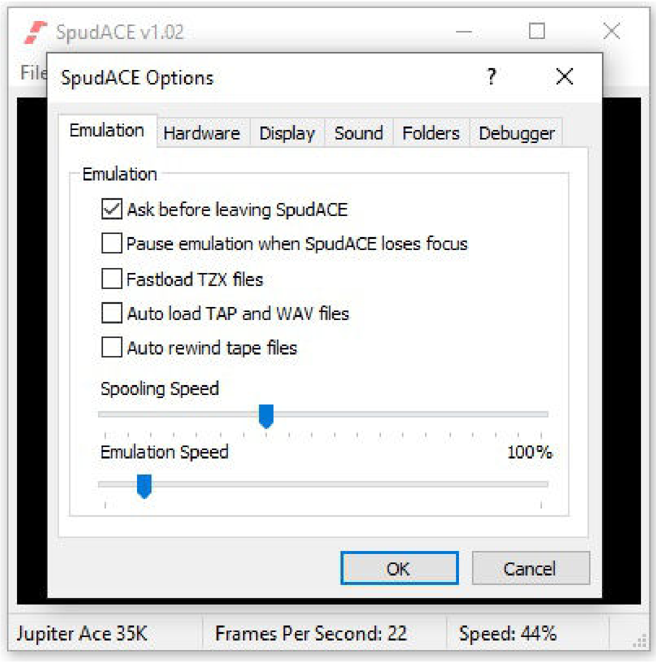
To start the emulator simply double-click on the **SpudACE.exe** file. The ACE Jupiter will welcome you with an entirely black screen and a laconic little cursor at the bottom, waiting for your commands.
From here you can insert the code presented in the two articles dedicated to the Forth. You will notice, however, that writing directly into the emulator is not the best experience. Some keys may not match the original keyboard map of the Jupiter ACE and try to guess them is not exactly congenial. Fortunately, SpudACE helps us. Using the Tools menu you can view a keyboard map of the ACE - > **Keyboard Display** or directly use a virtual keyboard -> **Keyboard Helper**.



If for short listings this solution is acceptable, it is decidedly unacceptable for listings of a certain size. Again, the emulator runs to our rescue, giving us the ability to edit listings in TXT format using a much more advanced editor and then import them into the ACE Jupiter. Using the **File -> Load Spool File** menu you can select a txt file and the content will be written directly to the ACE. Well, I did say written because you will see your list write character by character on your computer. Nice to be seen, but for very long listings we can easily go for a coffee...

Emulator Options

Tools - > Options is the true heart of the SpudACE emulator. From here it is possible to configure its graphic and sound rendering, as well as the hardware that the emulator must emulate.

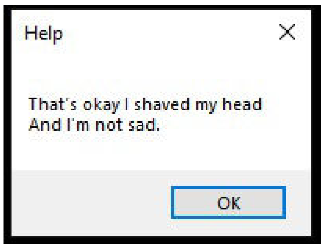


[✓] Ask before leaving SpudACE
☐ Pause emulation when SpudACE loses focus
☐ Fastload TZX files
☐ Auto load TAP and WAV files
☐ Auto rewind tape files
Spooling Speed
Emulation Speed 100%

Jupiter Ace 35K | Frames Per Second: 22 | Speed: 44%

Spudace is in fact able to emulate the following machines:
- Jupiter ACE 3K
- Jupiter ACE 19K
- Jupiter ACE 35K
- Jupiter ACE 51K

While the graphics can be rendered by:
- GDI
- DirectDraw
- Direct3D



That's okay I shaved my head
And I'm not sad.

Special mention for the **Help**. According to the author, it is absolutely not supported, however I recommend you take a look at it. Whenever you invoke it, you will be rewarded with a nice message.

Jupiter ACE

Jupiter ACE is a home computer created in 1982 by Jupiter Cantab, founded by Steve Vickers and Richard Altwasser. The two, former Sinclair engineers, decided to embark on their own, jealous of the money earned by Sir Clive. The peculiarity of this computer, in addition to a strong resemblance to the Spectrum, is the programming language included with the machine.
Unlike all home computers of the 80s, generally equipped with BASIC, the Jupiter ACE used Forth language. Although it was faster and better performing than Basic, perhaps due to a longer learning curve, it did not have the same success as the antagonist. Jupiter Cantab in fact ceased production of ACE in the autumn of 1983 while in November of the same year it was put into liquidation.

Technical specifications
CPU: 3.25 MHz Z80A
Language and O/S: Ace Forth
ROM: 2x4kB EPROMs
  containing THE FORTH compiler and editor
RAM: 3 kB expandable to 51kB
Mass Memory:
  1500 baud cassette
Video display:
  Monochrome 32 x 24 graphics
  High resolution 64 x 48
  128 redefinable characters
Keyboard:
  40 rubber membrane keys
Sound:
  Single Channel Buzzer
UHF TV Connector (set to channel 36)
Input for 2x cassette: Ear & Mic
Power supply (9v)
Dimensions: 215 x 190 x 30mm
Weight: 246 grams

Commodore Engineering: Welcome back to the scene!

by Carlo N. Del Mar Pirazzini

Hot Topic!!!

Commodore.
A brand that resonates so powerful for many of us that it awakens incredible memories.

Memories related to our beloved 8 and 16 bit home computers. We all know the story of the great Jack Tramiel, of SID, of Commodore 64, of computers for the masses and not for the classes and, unfortunately, we also know the decline and destruction of the brand.

The Commodore brand has been used for good and bad since 1994 by several companies or so-called companies.

In recent days we have contacted the newborn Commodore Engineering, an Italian company that is returning to the market with "old loves" and new products.
Let's hear what they have to tell us in this small but interesting interview.

Welcome back to the scene. Could you explain to us who you are and where you are leaving from and above all in which direction you are going?

First of all, thank you, Commodore Engineering comes from afar, from a group of Italian entrepreneurs who have been working in the field of computer engineering and institutional communication for 23 years.

We have developed our company by building software for banks and insurance, our core business is based on taylor made development of business applications and remote and frontal training portals. Our CEO Luigi Simonetti, a great enthusiast and collector Commodore, a few years ago decided to re-found this fantastic company and save it from being forgotten, one morning, during the typical meeting at the beginning of the week, he introduced himself to everyone saying: I WANT TO RE-FOUND LA COMMODORE, and from there everything was born.

I must say that not everyone had immediately given weight to this project, but over time it is entering the DNA of every collaborator, it is a dream that is becoming reality, certainly with many sacrifices, but it is worth it. We firmly know who we are and where we are going, that is, towards a new Commodore capable of satisfying more types of customers, from companies with the development of dedicated business systems and portals, public administration with the development of Intranets and ERPs, SMEs with the development of Apps and sites and, finally, individuals with the development of Video Games and Hardware peripherals such as Consoles, Desktop and laptop computers, tablets and phones. We want to provide a Commodore with an economic and social value, it must be an opportunity for all young Italian minds who work and study in the field of Electronics and Computer Science. In the new Commodore there is room for everyone, the important thing is that those who come to Commodore must be or become a true Commodorian respecting the sound principles of entrepreneurship and economic development typical of our country.

Tell us a little about yourselves. How is Commodore Engineering and its staff developed?

The new Commodore is not a vertical/Top-Down company but certainly a horizontal/bottom-up company, everyone here can have their great opportunity, everyone here is an indispensable piece.

At the moment our staff is composed of the General Manager Luigi Simonetti who is the focus of the company, the thinking mind and the commercial and public output of the brand, Stefano Cianfanelli as director of development, Giovanni Celauro as commercial and agency manager, Floriana Fascetti as head of relations with customers, Fabrizio Francucci as administrative director and Luigi Sestili as head of institutional image. Approximately 45 employees make up the operating staff.

What kind of plans do you have on site? Which in the short term and which in the long term?

At the moment I have to say that there are several construction sites in place, some already in the public domain, others still top secret :-)!

We have just introduced the new management system for synchronous and asynchronous distance learning called GEL, which will be used during 2021 by large companies such as TIM, ENEL, HELVETIA, HDI and many others to form a park of 160,000 learners/agents/executives/employees/brokers. During 2021 we will go out with 3 new video games and if everything assists us with a new portable console, the CP-64, which makes the verse in the style of the legendary Commodore 64.

For the future we certainly have as our goal, a Laptop, to finalize the production of the Tablet which is now at 70% and finally a phone.

In the past there have been several "COMMODORES", what kind of relationship do you have with these realities?

This is a really hot topic, our legal department headed by the Maroscia firm has found thousands of companies worldwide called Commodore, but fortunately only 2 are

really close to the brand. At the moment these are non-operational companies, indeed, there are no companies at all, we are the only ones with all the cards in order, at the national level we are recognized by Italian M.I.S.E. (Ministero dello Sviluppo Economico, Ministry of Economic Development, e.n.), and at the international level by all of Control Authorities, we have as our final project to bring all these independents under a single Corporate because in Commodore, as already mentioned, there is room for all.

Many of our readers are proud owners of C64 and Amiga and more generally of 8-bit and 16-bit home computers. Is there anything moving on the horizon to satisfy this whole fan base?

Obviously our aspiration is to continue the fantastic wave of success achieved by the "old" Commodore, but times change and it is certainly no longer possible to present 8/16 bit systems on the market, we also have to deal with the budget, so it is necessary to present peripherals capable of opposing current competitors.
Surely our peripherals start from that design, start from that user experience proposed by the C64 and the Amiga platform.

We can tell you that our general manager has already put in place a nice project, the CP-64 (we talk about it at the bottom of this paper, e.n.), and there are 2 others in the design phase. We hope that the Commodore Community will start to trust us and support us as much as possible, even during the design phases, because the final product is all of us, not just Commodore Engineering.

Thanks again for your attention, if you want to add something, you have carte blanche.

Thanks again for your attention, the only thing worthy of note is to remind everyone that if you have interesting projects, we are ready to listen and give you space.

Come on Commodore!

Oh yes… Come on Commodore… And most of all, Hello again and welcome.

We look forward to a bright future for this brand that is printed with fire in our hearts.

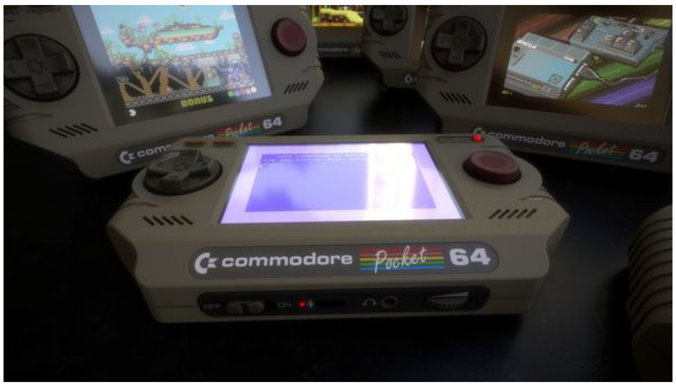
CP-64 Commodore Engineering Pocket 64

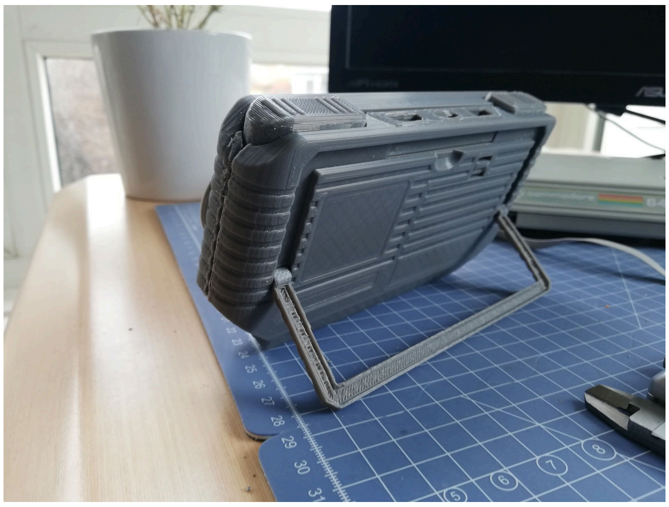
Here is the project of the new Commodore Engineering for all lovers of the "breadbin".

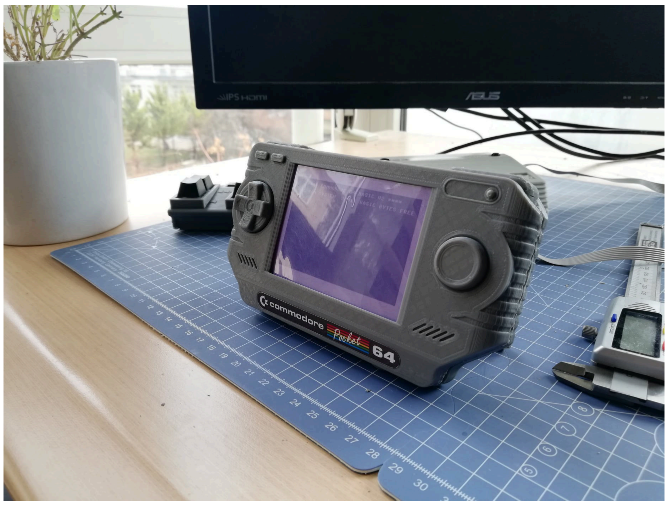
It is a small portable console under development.

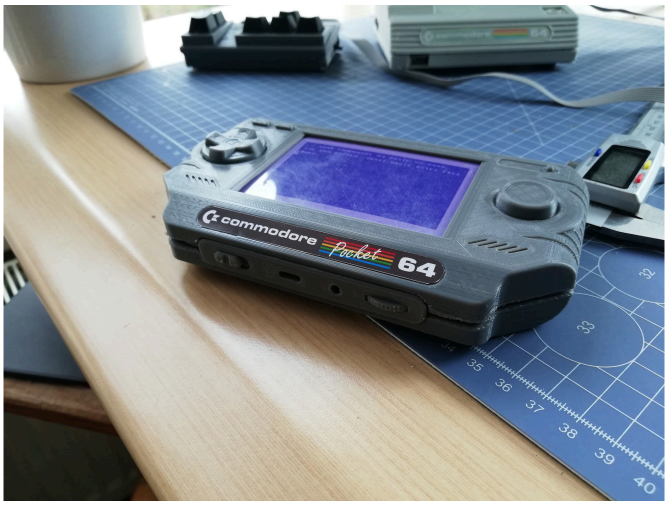
For now the technical specifications (whether it will be emulation or implementation) have not yet been released, but on the Commodore Engineering facebook page you can follow the developments.

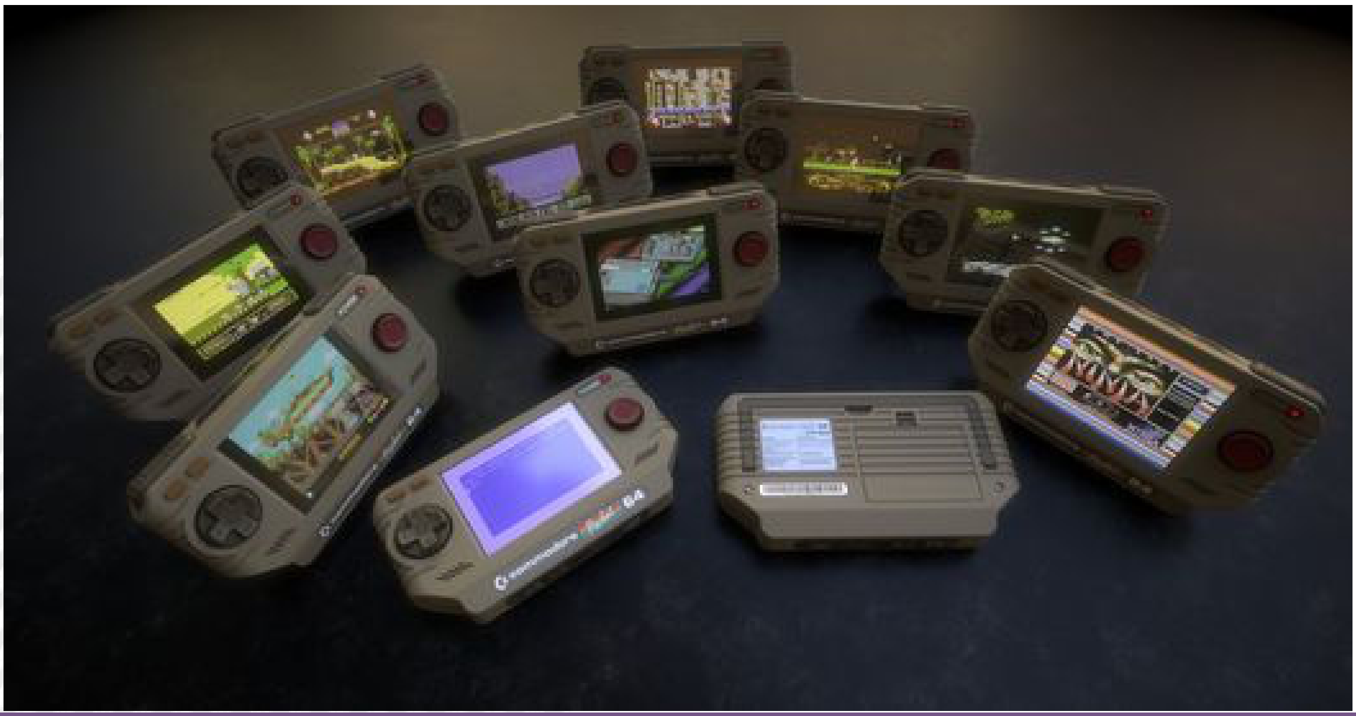
In the meantime you can enjoy the preview photos of the product that give hope.











20 Febbraio 1984: Sinclair QL presentation in Italy

by Alberto Apostolo

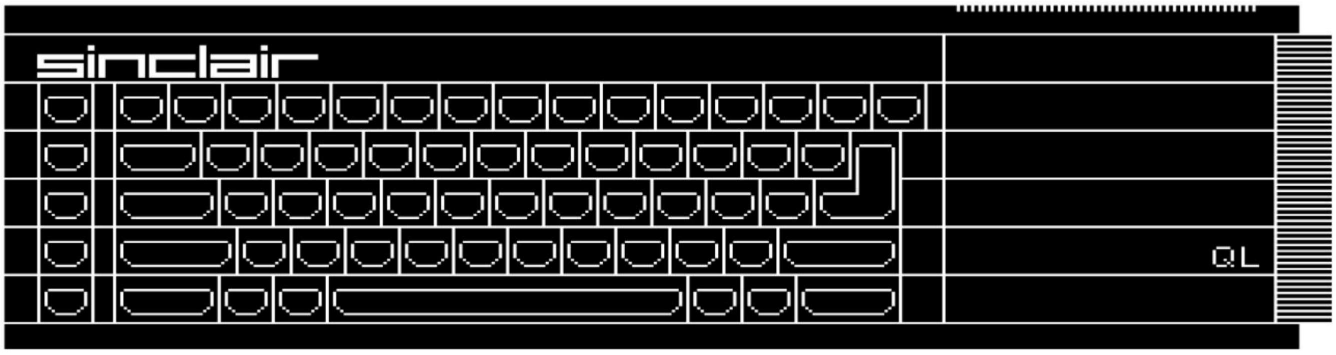


Fig. 1: an artwork by E.M.Giordano [EMG91].

After the articles published in RM13 (ITA) and RMW00 (ENG) it is a pleasure to return talking about Sinclair QL.

In a press release issued on 15 November 1983, it could be read that Sinclair held 64% of the personal computers market in Italy. In addition, the Italian city of Milan (together in Paris, Munich, Stockholm, Madrid) was indicated as the venue for events and press conferences to promote Sinclair products [SQ20].

Not surprisingly, on February 20, 1984, in Milan, in a crowded room of the Hotel Michelangelo (Fig.2), there was the press conference presentation in Italy of the Sinclair QL [Fio84]. Over 120 journalists could record the speeches of the speakers present at the table (Fig.3): Charles Cotton (at the time Overseas Business Manager of Sinclair), Jacopo Castelfranchi (1922-2017, president of the Italian GBC, entrepreneur, publisher, sports club manager), Eng. Claudio Fiorentini (at the time Marketing Director of Rebit, IT division of the GBC). Unlike the presentation press conference for the ZX Spectrum held a year and a half earlier, there was no applause but much silence about the high expectations raised by the new computer.

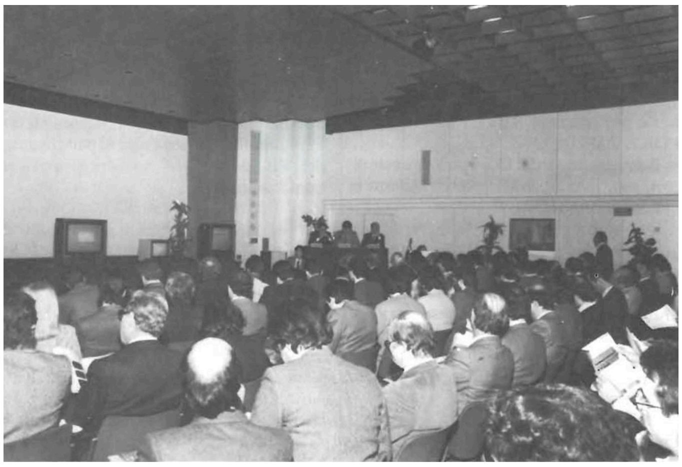


Fig. 2: the press conference [Fio84].

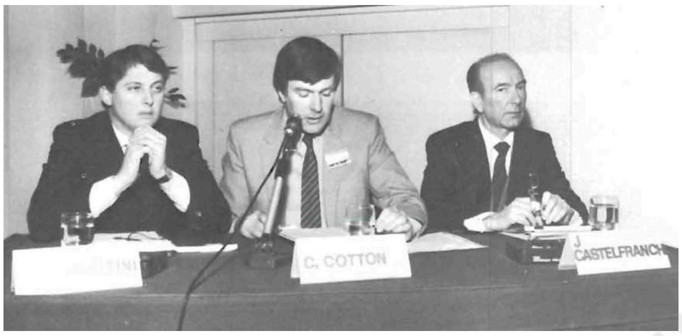


Fig.3:(from left) Fiorentini,Cotton,Castelfranchi [Fio84].

Sinclair wanted to demonstrate that it could also compete in the Professional Computing market and the QL model had to be the forerunner of machines built for this sector. During the specification of the technical characteristics, it was forecast that the personal computer market would develop during 1984. Software-houses would put groups of specialists to work, decreeing the end of the era when an enthusiast produced the software himself.

To promote the dissemination of QL and its accessories, the creation of a QLUB was also announced, an association of users that would inform subscribers by sending a newsletter. However, given the high cost of an office computer, the 8-bit home computer market would continue to exist and IT magazines and publishing houses would also have to meet the needs of more "serious" and professional users, changing the style of their publications.

Sinclair QL in pills (Wikipedia)

Computer Class: Personal computer
Manufacturer: Sinclair Research (United Kingdom)
Start of sale: 12 January 1984
End of sale: 1993
Release price: £399 (in the UK)
CPU: MC68008 at 7.5 MHz
FPU: not present
MMU: not present
Other processors: 1 Intel 8049, 1 ZX8301, 1 ZX8302
ROM: 32 or 48 KiB
Standard RAM: 128 KiB expandable up to 640KiB
Standard OS: Sinclair QDOS
Other sw: SuperBASIC, Quill, Archive, Abacus, Easel

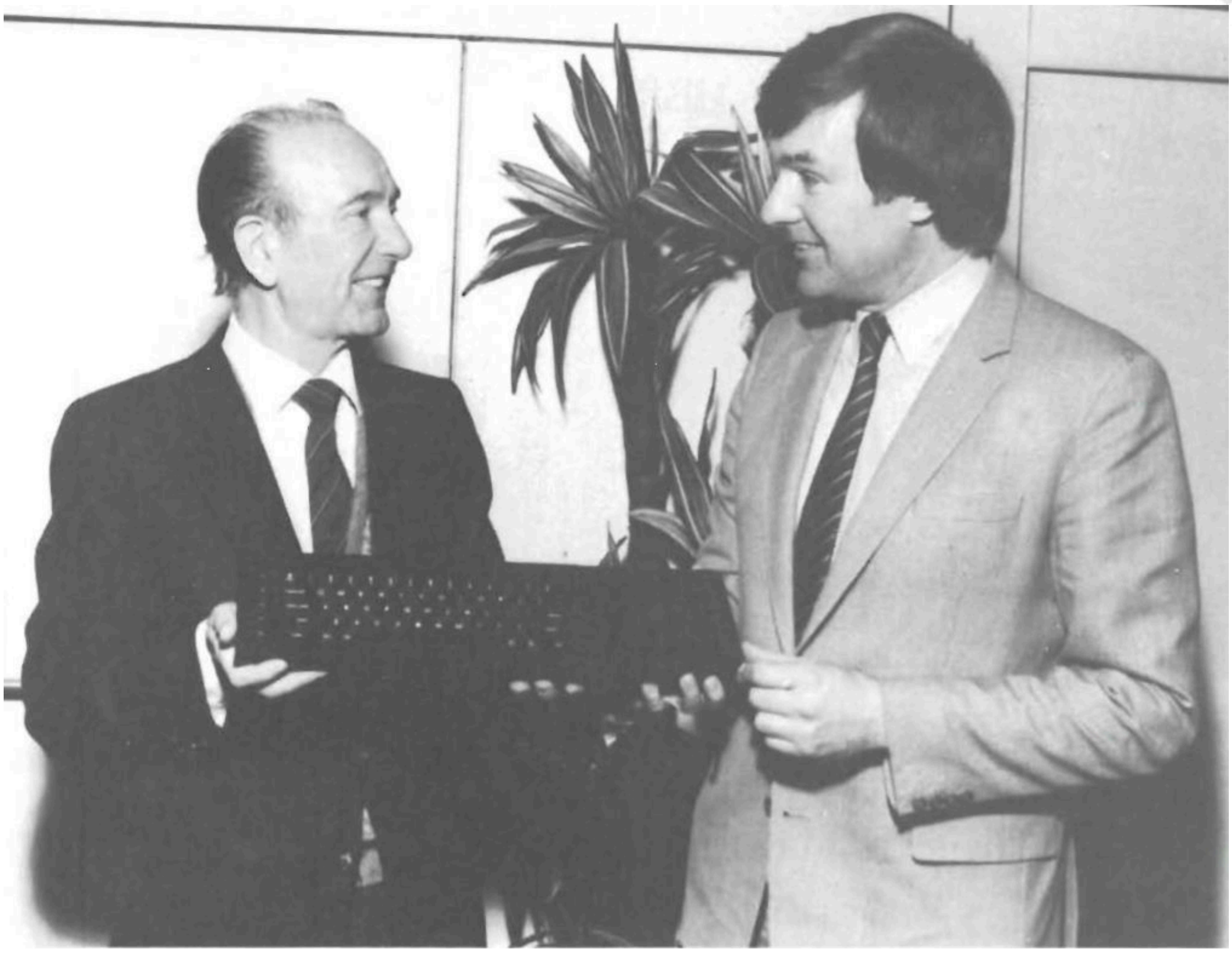


Fig. 4: a QL proudly shown by Jacopo Castelfranchi and Charles Cotton [Fio84].

QL with Italian keyboard
Version 1.13 of the QDOS corresponding to the MGx ROM (x=E Spain, F France, G Germany, I Italy, S Sweden) and EFP (Greece) (Wikipedia source) was planned for the European market.

In [Gra85] there is a review of the "Italian" version of the QL with QZERTY keyboard (Fig.5), QDOS with commands in Italian, manuals of application software translated into Italian.

Owners of a "normal" QL could purchase (at a favourable price) a transformation kit consisting of two new ROM (ver$ MGI with QDOS 1I13, i.e. 1.13 Italian), the QZERTY keyboard, PSION programs version 2.23, the manual in Italian, automatic enrollment in the QLUB.

An Italian clone of QL
A quote must be reserved about S.P.E.M. company in Turin. Under Guido Masoero direction, in 1985 System 2 was produced, a QL mounted in a metal case, equipped with floppy controllers, two 3.5-inch drives, 512 KB RAM (Fig.6). The keyboard (similar to a PC) was connected via a flat cable. The assembly was cleverly designed, so as to leave the QL expansion slot free (unlike so many interfaces on sale at the time).

A complete article on System 2 was published in no. 56 of the Italian online journal Jurassic News [CB15]. S.P.E.M.'s

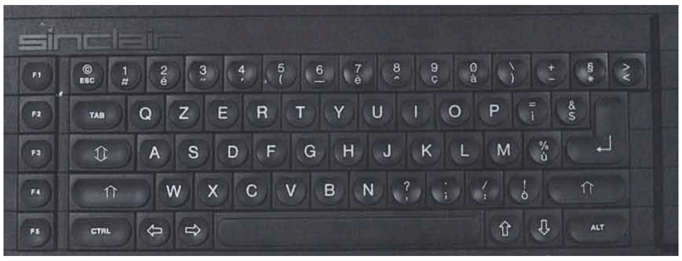


Fig. 5: the QZERTY keyboard [Gra85].

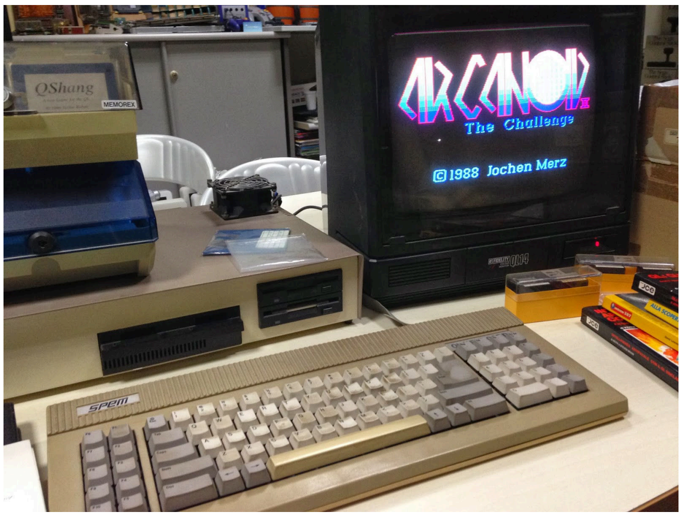


Fig. 6: SPEM System 2 (da Google).

Advertising advertisements were published in several Italian magazines of Informatica in the 1980s (e.g. 1985-86-87).

It was not easy to find more information about S.P.E.M. but in the end I had some success. On the Spanish fanzine QLave [QL86] I found the logo and the meaning of the acronym S.P.E.M. (Studio Produzioni Elettroniche Masoero, i.e. Masoero Electronic Production Studio, Fig. 7). Later I discovered that it was founded on 1 July 1982 and then ceased operations on 31 December 1993.

Guido Masoero, born in Turin on March 20, 1942, after practicing the profession of designer from 1970 to 2001 (LinkedIn source), has become an appreciated wood craftsman (Jurassic News n.59). Until the end he maintained a family tradition (his grandfather Antonio had founded a woodworking tool factory, completely destroyed during World War II).

Seriously ill, Guido Masoero passed away around the middle of February 2019, leaving his wife Silvana and depriving all those who esteemed him of his valuable advice (Google source).

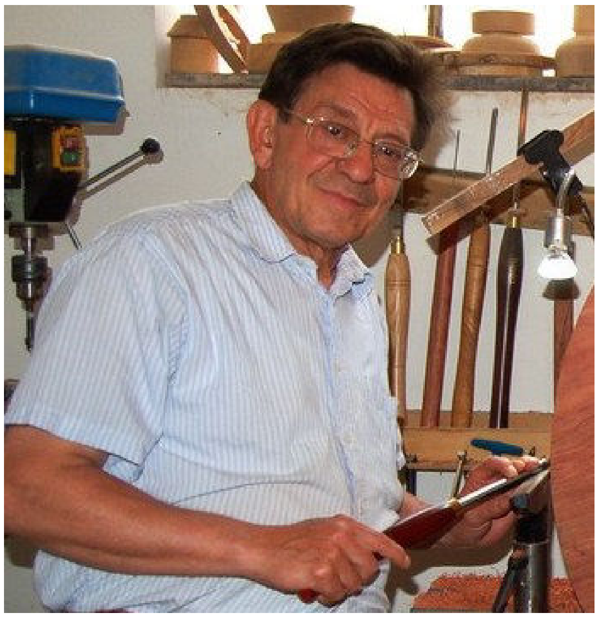
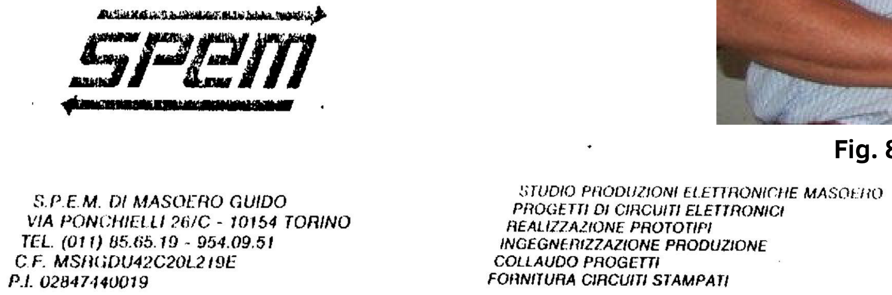


Fig. 8: Guido Masoero [Mas20].


S.P.E.M. DI MASOERO GUIDO
VIA PONCHIELLI 26/C - 10154 TORINO
TEL. (011) 85.65.19 - 954.09.51
C.F. MSRGDU42C20L219E
P.I. 02847440019

STUDIO PRODUZIONI ELETTRONICHE MASOERO
PROGETTI DI CIRCUITI ELETTRONICI
REALIZZAZIONE PROTOTIPI
INGEGNERIZZAZIONE PRODUZIONE
COLLAUDO PROGETTI
FORNITURA CIRCUITI STAMPATI

Fig. 7: SPEM's logo [QL86].

Bibliography
[CB15] D.Cavicchio, D.Baldi, "Un Sinclair QL italiano", Jurassic News n.56, Nov. 2015, pagg.20-24,
https://www.nightfallcrew.com/wp-content/uploads/downloads/2015/11/jurassicnews_issue_56-2015.pdf
[EMG91] E.M.Giordano, Sinclair News n.2, Jul-Aug-Sep 1991, pag.10,
https://www.emagsoftware.it/spectrum/Sinclair%20News%202.pdf
https://www.emagsoftware.it/spectrum/
[Fio84] C.Fiorentini, "Il salto di qualità nel professionale", Sperimentare con l'Elettr. e il Comp., May 1984, Pagg.24-25,
http://www.introni.it/pdf/Sperimentare%201984_05.pdf
[Gra85] S.Grandi, "QL", Sperimentare con l'Elettronica e il Computer, Nov. 1985, Pagg.60-65,
http://www.introni.it/pdf/Sperimentare%201985_11.pdf
[Mas20] "Tornitura di Guido" (2020/08/27) retrieved from https://tornituradiguido.wordpress.com/
[QL86] Fanzine QLave, Oct. 1986, https://sinclairql.speccy.org/ceiuql/qlave/qlave_v2_n3_octubre_1986.pdf
[SQ20] AA.VV., (2020/08/25) retrieved from, "Bill's Papers - The Sinclair Press Release", pag. 239,
http://www.sinclairql.net/downloads/Bills_Papers_The_Sinclair_Press_Releases-SCN07-0793-0300dpi-col-SSLk-ReadIRIS-SQPP.pdf

Links to interesting readings

Italian readings
G.Marano, "Programmi per Sinclair QL", https://archive.org/details/programmipersinclairql
A.Nelson, "Alla scoperta del QL", https://archive.org/details/allascopertadelqlilcomputersinclair
Italian magazines (tra cui Sperimentare con l'Elettronica e il Computer) http://www.introni.it/riviste_selezione.html
"Il nuovo Sinclair QL",pagg.57-68, http://www.introni.it/pdf/Sperimentare%201984_02.pdf
"QL USER", pagg.19-25, http://www.introni.it/pdf/Sperimentare%201984_06.pdf
"QL USER" (interview with David Karlin), pagg.36-37, http://www.introni.it/pdf/Sperimentare%201984_07_08.pdf
"Signore & Signori Sir Clive Sinclair" by Eveline Moore, pagg. 64-67, http://www.introni.it/pdf/Sperimentare%201984_09.pdf
"QL - Plus 4 , sfida all'ultimo bit", pagg.28-32, http://www.introni.it/pdf/Sperimentare%201985_07_08.pdf

English readings
Dilwyn Jones Sinclair QL Pages, www.dilwyn.me.uk
Various documents, https://sinclairql.speccy.org/archivo/docs/docs.htm
Jan Jones, "QL SuperBASIC THE DEFINITIVE HANDBOOK", https://sinclairql.speccy.org/archivo/docs/books/qlsbtdh.pdf

Japan part 16: Nintendo G&W Collection

by Michele Ugolini

Talking about Game & Watch is never trivial.
Talking about collecting, often means taking part in a Texas Hold'em game.

Finally, talking about Nintendo, always requires the use of a good toolbox, full of appropriate words.
Let's try mixing these three sentences, talking about two objects of worship. Expensive, rare, limited, Nintendo.

Big "N", in its history, has rarely produced objects without deep market research and above all has hardly ever produced jewelry that could leave the video game set unchanged after such events.
The collecting market is full of secrets and virtues, especially when it comes to high-demand items such as Nintendo collections and game collections. Did Nintendo produce a few games that contained themed collections? I prefer not to answer that question. It's more intriguing to find out how forward-looking Nintendo is in easily squandering the word "collection" on store shelves. So, it's time to talk about two rather uncomfortable objects in Nintendo's video game terminology.

They may not be uncomfortable for the company, but they certainly are for us collectors.
Nintendo G&W Collection volume 1 and volume 2.
The first volume is a limited edition collection of only three Game & Watch games for Nintendo DS. Yeah, right, just three games.
Oil Panic, Donkey Kong, Green House.

Game & Watch Collection for Nintendo DS was initially exclusive to the Nintendo Club, available only in Japan, in 2006. Subsequently, Nintendo brought the Nintendo Club to America, Europe and Australia as of December 15, 2008 and Game & Watch Collection was also made available in these new areas.

A sequel to this game, Game & Watch Collection 2, was originally released on December 15, 2008 in Japan, again as part of the Nintendo Club program. It was made available in North America on March 31, 2010 and in Australia in September 2010.

This collection features the games Parachute and Octopus, as well as a new game: Parachute × Octopus, a combination of the two.
Yeah, just three games. On the contrary, on reflection: only two games. This is Nintendo: game playing power as well as visceral calculation in marketing.
Do we want to talk about the dizzying prices of these two collections? Is it possible to find them at reasonable prices?
Are online auctions beneficial?
Do thrift stores own these items?
I've asked all these questions myself. Sometimes a lucky hand... as in Texas Hold'em: I own them both, at "reasonable" prices.

Oil Panic
The game is set in a gas station. Be careful though: there is a tube that leaks and drips oil. A station employee must collect these drops in a bucket and discharge them into his boss's oil drum, so he can take them to the cars waiting next to the petrol

Figura 1

pumps. It must also be fast, as the dripping oil is right above the flammable sources that will cause a fire if the droplets come into contact with them. The bucket can contain only three drops of oil, represented by three bucket filling lines. For every drop of oil the employee takes, he receives a point. Earn one point for discharging one drop of oil into the oil drum, two points for discharging two drops and five points for discharging three drops. As the game progresses, the oil will drip faster.
If you lose a drop, if the bucket overflows, or if the employee unloads the bucket on a customer, rather than properly to their boss, you lose one of four lives. This game uses two notifications: one for missing or overflowing oil and the other for the error in pouring oil on customers. If the player gets three errors from one of these categories, they receive a Game Over.

Donkey Kong
Donkey Kong for Game & Watch is a double-screen version of the classic arcade game. It was released as part of the Multi Screen series on June 3, 1982. His gameplay is similar to that of the arcade version: Donkey Kong has captured Pauline and Mario must save her. The game was later transferred to Game & Watch Gallery 2 and Game & Watch Gallery 4, both of which include an updated "Modern" version of the game. Donkey Kong's latest porting was released in the Game & Watch Collection for the Nintendo DS dual-screen, allowing an exact replica of the original gameplay.

Green house
Green House for Game & Watch is always double-screen, released as part of the Multi Screen series on December 6, 1982. It is present in the Museum of the Game & Watch Gallery 2. It was later re-released for Game & Watch Gallery 3, with a classic version and an updated "Modern" version.
In gameplay we have to defend our plants from spiders and worms.

Parachute
Parachute is a Game & Watch game released as part of the Wide Screen series on June 19, 1981.
It was the first game in the Wide Screen series. The goal of the game is to catch parachutists before they land in shark-infested waters. It later appeared in Game & Watch Gallery 2, Game & Watch Gallery 4 and Game & Watch Collection 2. In Game & Watch Collection 2 there is also the extension of the game that interfaces this very title with Octopus.

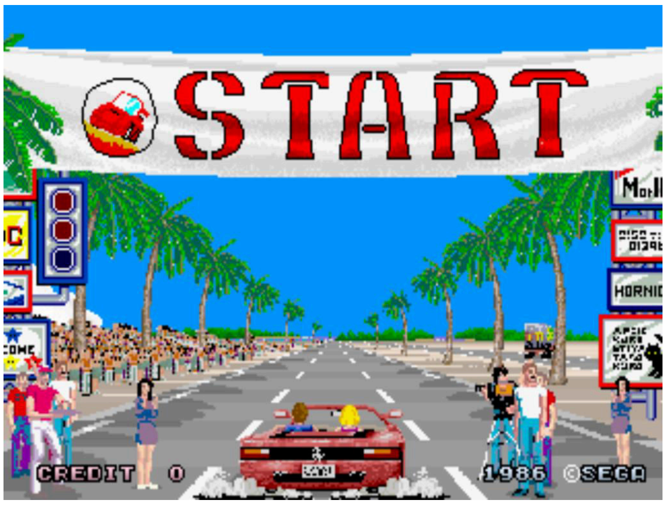
Octopus
Octopus, known as Mysteries of the Sea and Mysteries of the Deep in the UK, is a title of Game & Watch released as part of the Wide Screen series on 16 July 1981. In this game you must get as much gold as possible. However, you have to be careful of the Octopus near the treasure because touching the tentacles, you lose a life. Losing all three lives means Game Over. Octopus also appears as a mini-game in Game & Watch Gallery, Game & Watch Gallery 4 and here in Game & Watch Collection 2. In addition, in Game & Watch Collection 2, as already mentioned, the game interfaces with Parachute.

Well, dear readers, that's all for this issue of RMW, I hope you'll be playing these beautiful Nintendo titles. I await you in the next issue: there will be interesting innovations inherent in this particular world endowed with a stylized immortal soul.
See you soon!

Figura 2

Out Run, feeling that sense of freedom...

by Mic the Biker Novarina



There was a time when entering the arcade was a gesture like any other but seasoned with an incredible sacredness. He crossed the threshold and any connection with the outside world changed. You could run by bike launching newspapers as well as making endless brawls with a friend at your side but always in a virtual way.

Then there was that kind of game that stays on you, that every day, even decades later, you relive. Out Run for me is one of these. Released in September 1986, the game in question could be found either with a deluxe car style cab or with a simple one with steering wheel and gears to play standing: in all there were 4 cabin combinations, two seated and two standing.

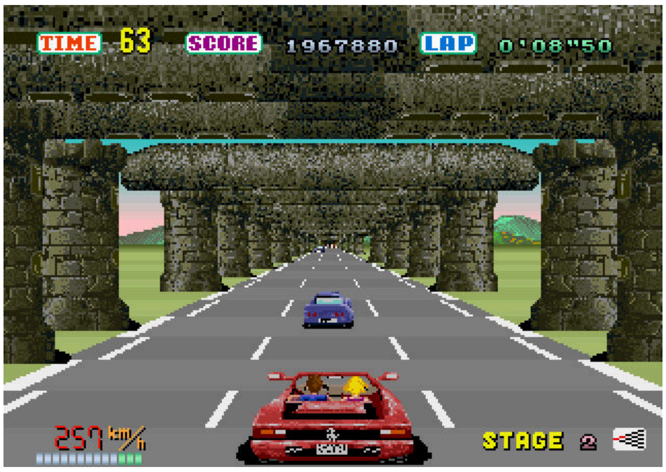
For the graphic, the same engine as Space Harrier was used, the famous super scaler, revised and enhanced ad hoc: in fact, the graphic rendering left the world speechless. The insert coin screen sees the funny cartoon of a sports car locked in an oval while images of the game run on the screen. You were already captured by this but the moment you insert a coin is the first reason why this game stays inside you: car radio, hand on the knob and possibility to choose your favorite radio station.

The songs Passing Breeze, Splash Wave and Magical Sound Shower keep us company and over time have become a small soundtrack of those years. Choosing the song comes one of the most iconic screenshots in the history of video games: Ferrari Testarossa spider, us driving (yes, because “we” were the protagonist) and a blonde sitting next to you. A road, palm trees, friends beside us to start and we leave. Beach, sun, sea and speed, curves at maximum, snail car and truck to overcome.

Crashing is easier than it seems and can lead to either a simple slowdown of the vehicle, perhaps a spun, or in extreme cases toppling over with even passengers thrown in the air and a great waste of time. Time to get to the checkpoint, it's worth a burning game over.

The dynamics of play here were revolutionary: after endless curves and slopes, the road is divided into two: right or left? Depending on the direction taken, it obviously changes the graphic appearance but also the difficulty. We are faced with 15 possible scenarios that form a pyramid with the start level as the tip, which is always the same. Hence each checkpoint represents a game that lends itself to being continued in many ways. You can always go right or left to run more external levels but also choose to jump into the center, increasing the combinations that lead us to run the 5 paths that separate us from the finish line: each final will be different depending on the road taken.

In terms of graphics, the development was a bit troubled in the beginning. Sega gave as an input to be inspired by American landscapes but the supreme programmer, Mr. Yu Suzuki, was not of the same opinion. He found the States too large and therefore too dense with empty spaces to draw graphic inspiration. This is what Europe proposed, given that it had recently turned it around. Sega accepted the proposal and inspirations were taken from locations such as the Monegasque coast, the Swiss Alps, the French Riviera, some views of the central European plains and skylines on the horizon of places such as Frankfurt, Rome and Milan.



In Suzuki's mind everything had to participate in creating something unique and fun: goal achieved!

Out Run was for many years “THE” arcade driving game. For many it is still “THE” reference, although it has been more than 30 years. Because Out Run, if you played it in the' 80s, you’ve got that feeling..

I was talking to a dear friend who is also passionate about Gaming, turn and turn for the umpteenth time we happened to talk about Out Run. It was a dream: you could choose the music, a Ferrari, which way to turn in the checkpoint. The usual praises we'll never finish.

In front of the umpteenth beer, a moment of silence. Then the question: "Why always Out Run?"... Silence... "Well, because it's made history... Because it's beautiful... ". But one answer out of all was the most beautiful: "For everything reminds you. Those palm trees, the sea beside, the road running. For the emotions it still gives you today. It's freedom"... Silence.

That sentence was the reason I'm here writing now. Because it's all true. Emotions, feelings, freedom. Out Run has transformed over the years: it has gone from being a game masterpiece to the symbol of an era. Just like Madonna's look, Sony's walkman and audiocassettes, Out Run became part of the common imagination when you think of the 1980s.

A small curiosity: do you know that there is only one model of Testarossa Spider? It was requested by Gianni Agnelli himself. The car in question is number 62897, painted Nurburgring silver.

The request dates back to 1986. The natural question, for this incredible coincidence, can be “Was the egg or the hen born first?”
We will never have an answer but we old style gamers like to think that even after seeing Out Run and who knows, maybe having played a game, he fell in love with that sense of freedom that pervaded you and had a Testarossa modified.

There's a lot of romance in all this, a lot of intimacy, like all the good things that stay on you. Now I'm going, I have the radio tuned on Splash Wave: I have to go for a drive along the promenade!

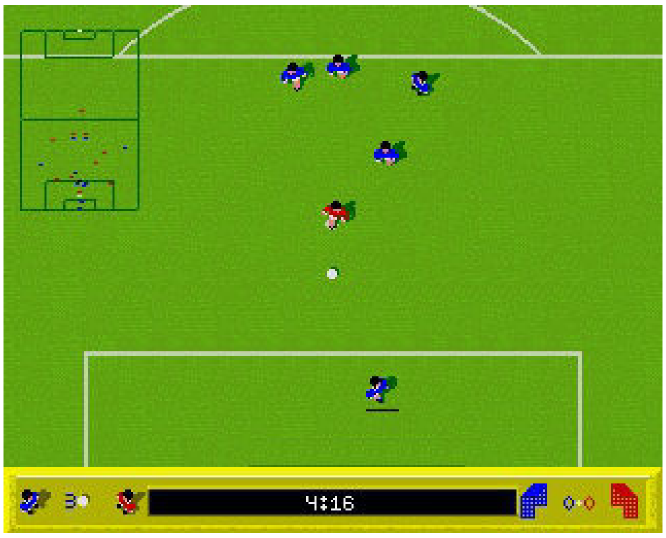
Kick Off VS Sensible Soccer
A duel by goals and fun

by Edoardo Ullo

Football on computers and consoles today lives on the dualism between FIFA and PES, with very little space for additional uncomfortable third parties. A challenge that has lasted annually since the early 2000s. But it's not the only duel we've had. In the early 1990s, in fact, we witnessed a head-to-head between two series that revolutionised football games by carrying them towards the modernity and complexity we know today.

Let's talk, of course, about the dualism between Anco's Kick Off series and Renegade/Sensible Software's Sensible Soccer series. If we were to make an impartial judgement, it would be difficult to say which is the best. And also public and critical often and willingly have divided partaking for one or the other. The game is the same but interpreted differently by two leading actors of absolute importance. A little more serious approach to Kick Off, while more immediate is definitely Sensible Soccer. Our recent survey on the Facebook page rewarded Kick Off who was preferred over the rival. This event was an opportunity to take us back in time and you were talking about that "remote" football challenge that inflamed and excited especially the players who saw the quality of gameplay improve, and not a little, of what is certainly one of the most loved sports in the world. For many the most beautiful game in the world: football.

Until 1989, in fact, we had experienced the splendour of Match Day, and in the games room of Exciting Soccer (the first football arcade, in memory of man, with off-game) and Mexico 86 (or Kick and Run, Taito, which allowed a very good gameplay). Very interesting arcade titles. In 1988 Microprose Soccer and Emlyn Hughes International Soccer had their great moment of glory thanks to their more advanced gameplay. But the world hadn't met Dino Dini yet. It was 1989 and Anco – a small British software house - published Kick Off. A real revolution. The reason is soon said: realistic physics and proportions between the size of the football field and the players.



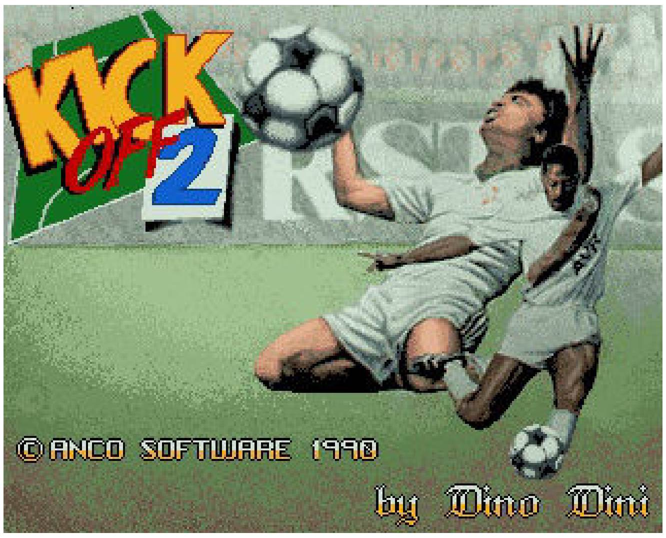
Kick Off's advanced physics meant that for the first time in a football game, the ball wasn't literally glued to his feet. This gave a certain realism and allowed games never seen before. But this also meant a lot of dedication and training. The most talented and the most willing were able to build fast and spectacular actions. And how enjoyable it was to discard the goalkeeper and place the ball on the net, a touch of class that before was substantially impossible to do in other titles of the time or of that recent past.

We have also mentioned previously the proportions between field and players. Thelatter were small compared to seemingly enormous terrain. The bird 's-eye view and multi scrolling, however, allowed an unparalleled fluidity of action. And after a first impact, we all realized that in fact this view by Anco's product was true: before then the fields were either too long or too narrow. Or vice versa. A marked defect, just to make you understand, in our opinion, in the splendid Emlyn Hughes International Soccer.

Failures, admonitions and expulsions did the rest, giving a more simulative and “serious” tone to the title.
The game wasn't flawless. And in addition to still being a bit edgy especially at the beginning, there was a lack of off-game, the fulcrum rule of football. But it was only

the first Kick Off, the kick-off of a series that was just beginning...

In 1990, the year of the "Magic Nights" (at least until the semi-finals of that World Cup of Italy 90 for us, ed.), Anco published Kick Off 2. Slightly better graphic appearance but substantially similar to the first but with a lot more "code": ergo a more refined and improved gameplay.



The second chapter of the series introduced Aftertouch, which is the shooting effect. This feature was already present in Microprose Soccer (many will remember the fun "banana shot") but clearly with different results. In Kick Off 2, the effect shot allowed interesting trajectories, very dangerous shots for goalkeepers but also played in other parts of the field. But there was more: you could import the tactics from Player Manager, a managerial title always based on Anco, however, the first Kick Off. In Player Manager we played the role of coach and you could place the players on the field in complete freedom. It was then possible to import the tactics and use them in Kick Off 2 with amazing results and never achieved until then.

In addition, the new game allowed to play championships, tournaments, cups, friendly on four different types of surface: normal, muddy, rainy and synthetic. The difference between the four fields was not merely visual but the gameplay was affected. On a muddy field the ball slowed his run and it was harder to control him. So the game was broken. On the rainy ground the ground became slimy with insidious bounces. On the synthetic side, however, the field was very hard and even here the bounces were irregular. There was also the unknown wind that further changed the cards on the table.

But there was still a big flaw: there was no play. In addition, two typos became famous in Italy: the famous "Calcio d 'angalo" and "Cartellino gaillo". However, the offside was added with the data disks and in particular thanks to The Final Whistle which also added other playing fields and the possibility of hitting the ball upside down and expanded the sound effects a little, until then a little anonymous albeit good. And there was also recovery time.

The Kick Off series is at its peak and millions of video players around the world are eagerly awaiting Kick Off 3 news. This will come in 1994 but in another form. We'll get there soon. Let's go back to after Kick Off 2 because this represents the pinnacle of success. There is, however, crack: Anco separates from Dino Dini.

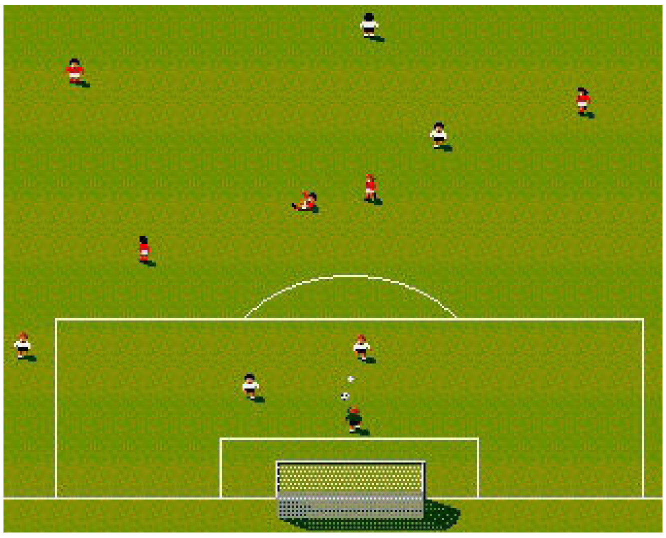
Meanwhile, in 1992 came the competition response that materialized with and in Sensible Soccer, developed by Sensible Software and published by Renegade. The football background of Sensible Software is interesting: four years earlier he had created Microprose Soccer which, in a good way, can be considered the "father" of Sensible Soccer and which on C64 did really well delivering to history one of the most beautiful sports games of the 8-bit computer of the Commodore house.

The great qualities of this Sensible Soccer were immediately visible to everyone: neat graphics (more "round" and pleasant than the direct competitor, ed), visual from above but at "three-quarters", speed and remarkable variety of gameplay, ball control more affordable than the "King Kick Off" despite the ball not remaining attached to the players' feet.

Obviously this title also made use of the effect shots but the construction of the choral actions was more possible since by pressing lightly you passed underground to the desired player. It was a real success that opened the duel with Kick Off.

However, it was the first title in a series that would become planetary within a few years. A few months later Sensible

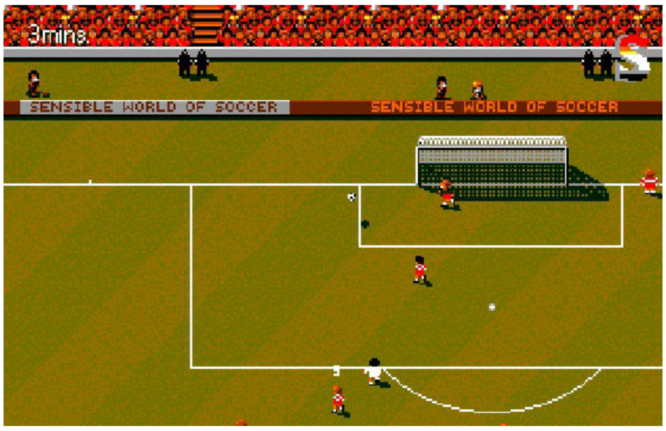
Soccer 1.2 was released, adding yellow and red tags to the Amiga version. Beauty comes in 1994.



But let's take a small step back to 1993: Goal is released, Dino Dini's Kick Off 3 but not Anco's. Dini, after leaving the parent company, she settled down with Virgin Games who published Goal. Title with enormous potential. Visual from above with a choice of camera between the traditional vertical and horizontal, zoom, bigger players, stunts, but as much fun as it was, it lacked a certain I don't know what. Let's go back to 1994, at the end of an year that we (as italian) will remember for a World Cup decided on penalty kicks in a dramatic final between Italy and Brazil won by the green gold for the mistakes of Baresi, Massaro and the "divine tail" Baggio (the amazing save in the first series of penalties of Pagliuca was not enough ahinoi); Sensible World of Soccer has arrived in stores.

A colossal game that in just two records managed to condense a football universe. The improved gameplay combined with a frightening amount of data for the time and a Career mode that brought football games into the modern era. Now almost all football titles have a Career mode. You could choose from over 1,400 clubs present and 131 national clubs with over 140 competitions and a database of over 26,000 athletes.

In addition, there were two powerful editors: the tactics editor and the competitions editor. Thelatter made it possible to design mixed tournaments of all kinds.
There was a football market that was already structured: players could be bought by offering not only the sum of money desired but also a technical counterpart or a direct exchange of players.
The introductory soundtrack titled "Goalscoringsuperstarhero" is legendary. A catchy, funny song that many still hum.



The gameplay was still filed with the 96/97 edition which added the ground passes to the effect that could also be transformed into shots. Gameplay freedom was very good and the game was really fun giving you the opportunity to have fun in multiple ways by finding multiple routes on the net. Career did the rest: 20 seasons where you could take your club to the forefront of the world and coach your favourite national team.

What made the audience fall in love with Sensible World of Soccer? In our opinion, immediacy and fun together with a very good depth. The contents, then, are definitely at the top and in this aspect, this series wins.

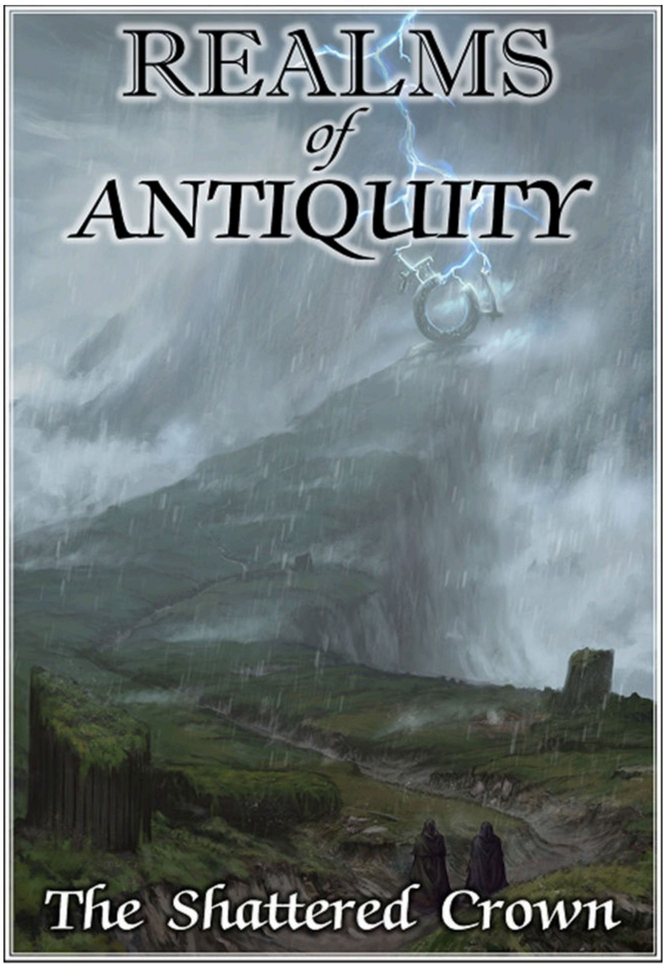
Kick Off, less immediate than the competitor, is however more simulative and offers – if well trained – an even wider gameplay.

We believe that the two series are complementary. It is no coincidence that those who try to develop indie football games look at Kick Off and Sensible Soccer as symbols to look at to find the right compromise at the gameplay level.

Realms of Antiquity

Hot Topic!!!

by Ermanno Betori



Today we present a new game for TI99/4A, "RoA" which stands for Realms of Antiquity.

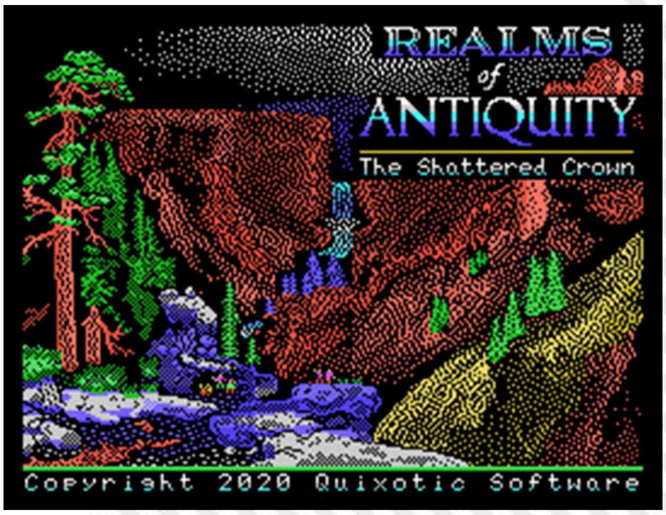
What is ROA? It is the ultimate RPG for the TI99. Few of these types of games for TI99 computer have been produced. We can mention, as a game on cartridge, the famous Tunnels oF Doom that in the late 70s, early 80s, was a reference as a type of interaction for RPGs created on the computer. We also have RPGs created for the TI99 with the Extended Basic module plus memory expansion (32k) and among the best titles we have, Legends, Old dark Caves, Living Tomb... But there was no real good RPG with a deep storyboard like Ultima, Pool of Radiance, Magic Candle , Wizardry, etc.

These famous RPG game series were born in the first three years of the 80s almost all them for the Apple II computer, but they had poor graphics, low iteration between characters and cumbersome game management. Although later, these games were moved to the other home computers of the time, they did not have a great following. We will have to wait some more years, from 1985 to 1990, to get games with captivating graphics and a script where the player plays as the hero. But all these RPGs were created, except for rare exceptions, for 16-bit machines such as Amiga, Atari ST, Macintosh, PC Ibm... and after their commercial success in some cases there was transposition on 8-bit computers such as Commodore 64, MSX, etc...

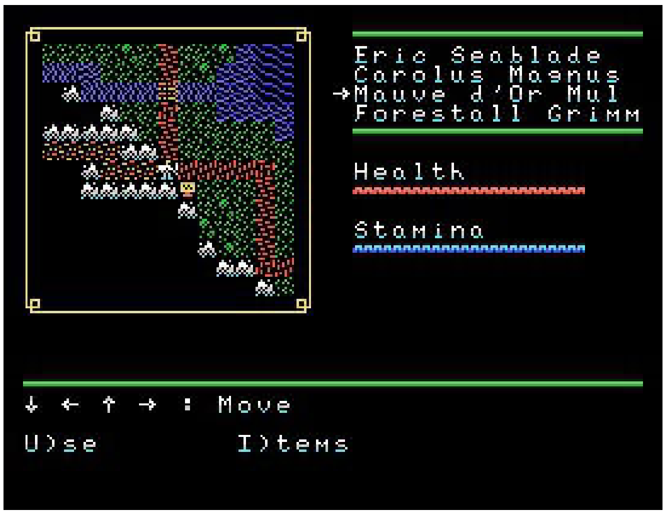
The titles listed below marked a watershed in the role-playing world:
- Last IV: Quest of the Avatar
- Last V: Warriors of Destiny
- The Magic Candle
- Wizardry 6 Bane of the Cosmic Forge
- Pool of Radiance

Therefore, the RPG enthusiast almost never found in the home computers sold between the years 1977-1985 a video game conversion of this type of games such as those derived from a series of books/games called Dungeons and Dragons already on sale in 1974.
Today, however, thanks to the commitment of an expert programmer, as well as RPG enthusiast, Mr. Adam Haase, we have the ROA game that fills this gap in the gaming world of TI99 users.

ROA is a video game that collects many ideas developed in games like Ultima 4 and 5, Tunnels of Doom, Legends, Realms of Arkania and Avernum; in fact we have an excellent storyboard and a high quality technical game management.
As technical features, the game takes advantage of many graphic and descriptive elements used in very high complex



RPGs and as already mentioned, found on much more powerful computers. In fact, we find the group with a maximum of four characters that can be created at the start of the game or enlisted during the game.

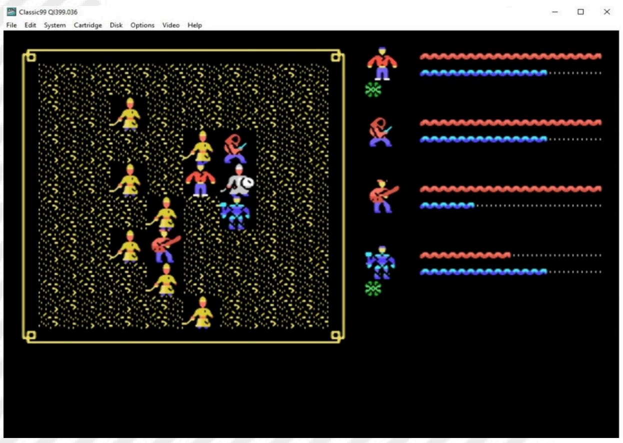


We have the main attributes such as Health, Resistance, Agility, Energy and character classes (nine of which are classic Tunnels of Doom warrior, wizard and thief). There are schools of magic, a feature developed in modern RPGs, a myriad of spells, the experience gained both in missions and in killing monsters which is not always present in old RPGs.

Also exist in the game some graphic elements that influence the gameplay such as the scrolling, enveloping, edged and inclined maps, the block of vision (such as when you are in a forest) or the elevation (where only part of the map is shown) and of course the light, necessary to illuminate the inside of the dungeons.

Multi-colour graphics were obtained on the TI99 thanks to the bitmap mode that allows 2 colors for each row of 8 pixels in one font, so you have both characters and enemies very well drawn and especially multicolored!

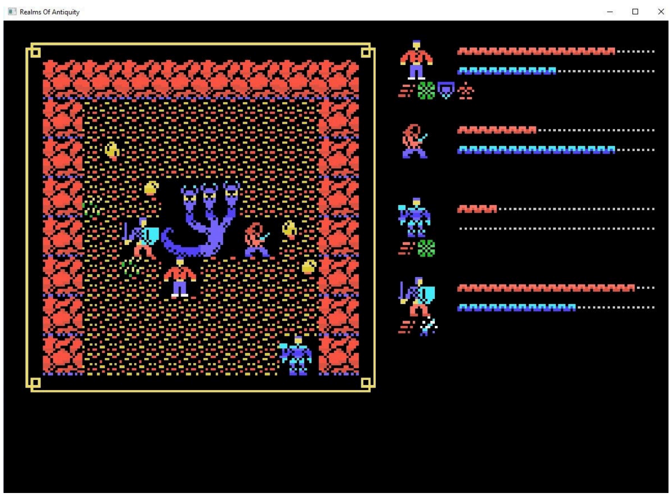
The game allows a very modern equipment management,



in fact the younger RPG players take for granted and obvious the fact that the characters can hold an incredible variety of melee weapons such as swords, bats, axes and distance weapons such as bows, rifles, slings that need ammunition without which they are unusable. Another bonus inherited from the more modern games is the idea of how the secondary hand is equipped with consequent variation in attack and defense parameters. That is, in addition to the classic way usually intended to hold a shield, there is the possibility of using objects that offer magical protection and that the thieving class can use both hands to hold weapons.

In the game, each character has a backpack that holds 10 objects and the tactical combat is very similar to that present in the Ultima 4/5 games.
In fact, it is a rotating fight, battle maps are randomly generated and you can see the effects of launch weapons, magic wands, spells by both heroes and enemies.

Above we see some classic combat screens with a wide variety of creatures summoned...

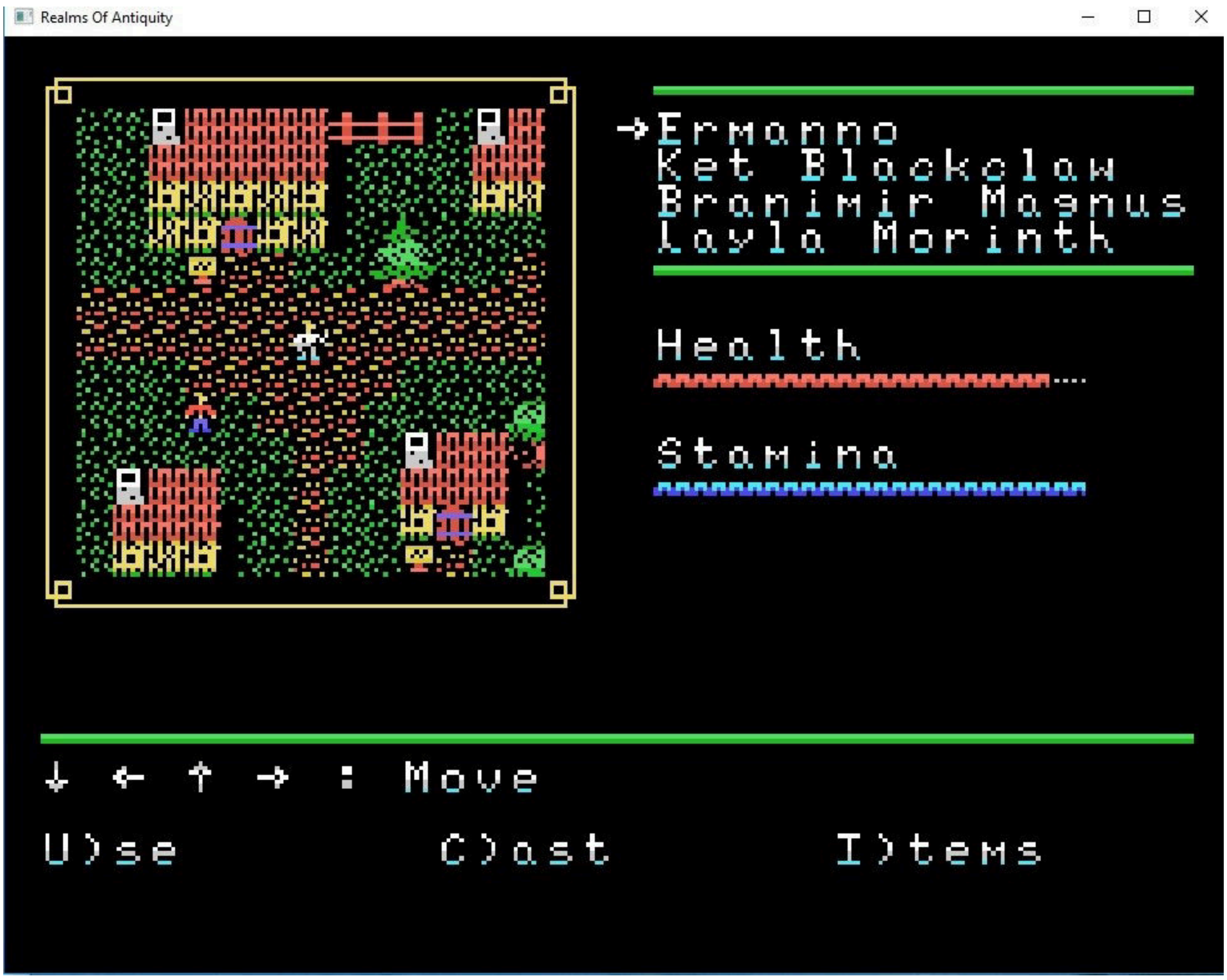


Over 225 different creatures and/or enemies. Also during battle you can find different types of monsters.

Some enemies are double-sized "bosses" like dragons, golems etc.
It is important to note that combat is also influenced by active magic, which can often turn a tough battle into a simple skirmish.

Another important factor is the presence of many below this that are usually activated based on the questions or answers you have with the various characters present in the various villages or cities scattered on the map.

Definitively ROA is a game created thanks to the sum of



the experience gained as a RPG video player over many years, in fact it has conceptually absorbed in itself many gaming techniques that start from the basic ones present already in the early 1980s up to almost those created in 2000.

Mr. Adam designed the game in its current form and programmed it on the TI99, overcoming many technical difficulties, which took a long time: almost 15 years!
But the result is exceptional, as I often forget that I am playing with the TI99 as a computer and not an IBM PC or other much more powerful machines!

ROA because of its complexity does work on the TI99 but needs an expanded machine in its entirety, with the only hardware compromise that needed to use a new memory card called SAMS that brings the computer's memory to 1Mbyte.

In fact, the Hardware requests to run ROA are:
- TI-99/4a Home Computer (Console)
- or Disk System TYPES with at least two disk drives that read 360k
- SAMS card, 1024k RAM or better
- Monitor or TV (LOL is preferred in colour)
- Cartridges such as Editor/Assembler, Extended BASIC, TI-Writer or the modern FinalGrom99.

With this article I think I have provided a good description of this RPG to you.

The evaluation? Mr. Adam created a new unit of measurement to set a standard in the games for the TI99, that's all I can say!

I strongly recommend buying it not only for TI99 users but also for all RPG game enthusiasts.

NEW GAME!!!

SOUL FORCE

Year: 2020
Editor: Protovision
Developer: Sarah Jane Avory
Genre: Horizontal shoot'em-up
Platform: Commodore 64

Masterpiece [comp. Head and work]. - 1. a. The best in a series of works by an artist, a writer... - Italian Dictionary "TRECCANI"

Gentlemen, we are looking at the best shoot em up of the new era for Commodore 64 ever made.

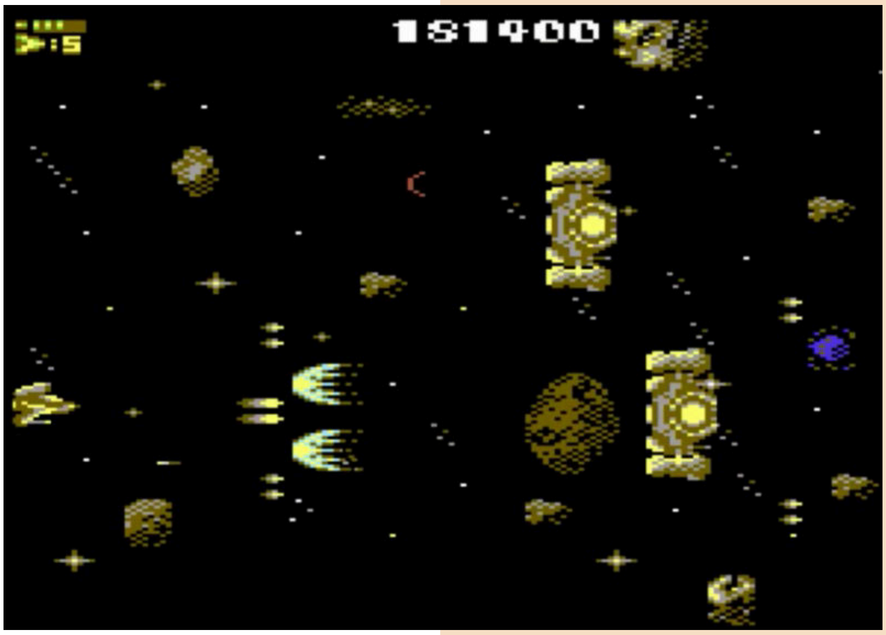
Soul Force takes first place for technical quality, playability and development and I think it will stay there for a very long time.
Sarah Jane Avory is a woman to marry!

This latest game closes a circle made up of wonderful shooters-all that in recent years have shown us the skills of those who know how to "use" with criteria a machine with almost 40 years of history.

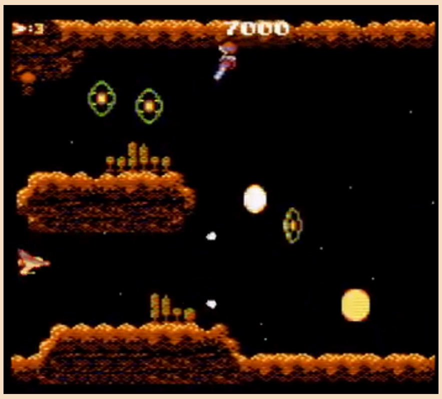
Soul Force is a classic of its kind. Spaceships, horizontal scrolling, parallax backgrounds, a bunch of sprites on the screen, power ups and lots and lots of frantic action.

Classical Story: A mysterious bio-mechanical fleet is intent on invading Soultron's peaceful star system, so it can steal all its resources. However, the star system anticipated such a move, building an advanced combat ship prototype to help defend the system.
When aliens invade, it's up to you to launch, pilot the new Soul Force fighter and face the enemy, hoping to discover their secret and destroy them once and for all.

We start the game with three ships and a basic weapon. Points are awarded for each enemy unit knocked down with a bonus of 500 points awarded when destroying a complete

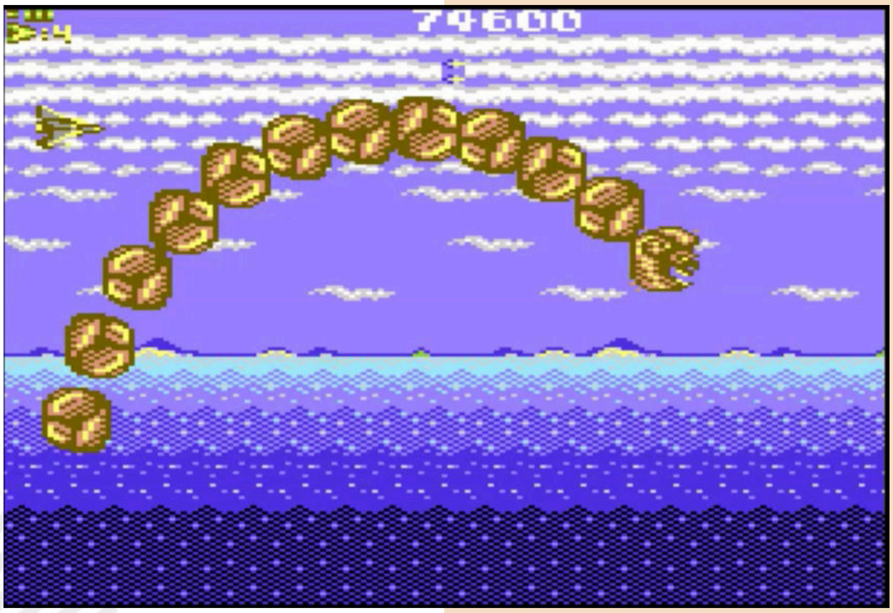


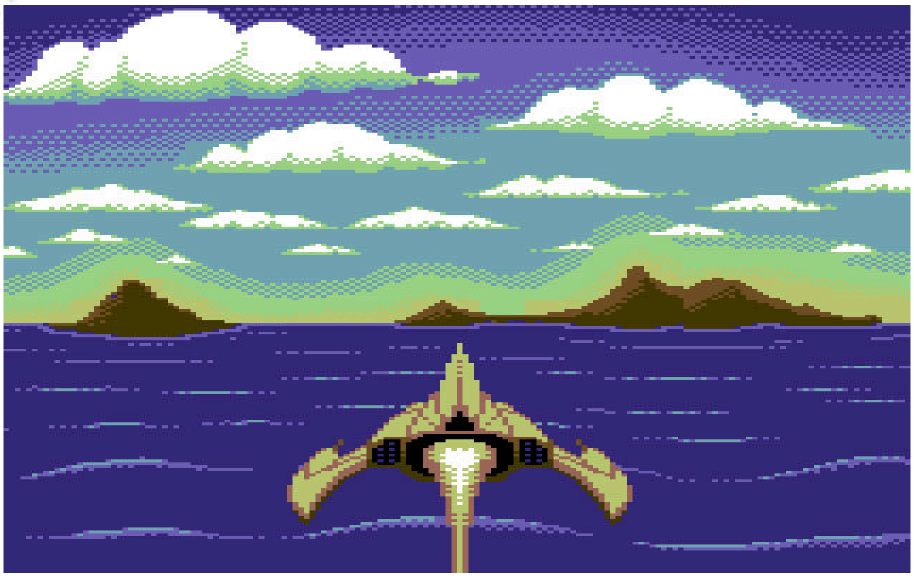
enemy wave. You earn one more ship for every 50,000 points scored.



Our weapon can be upgraded or replaced by shooting at a refueling ship which results in an upgrade that becomes available for collection.

The power of the currently equipped weapon is displayed in the upper-left corner of the screen above the lives





counter and each weapon has its own strengths and weaknesses; this adds some strategic element as you need to decide which weapon would be best suited to face the next waves.

Other pickups include smart bombs, an invulnerability shield and an energy shield. When you collect the shield, it remains active until the energy gauge is exhausted by the bullets or enemy ships.

The energy shield can be restored after destroying a number of enemy ships. A perfect animated introduction will launch us into the game and it will be immediately blasting and epicness The overall action within Soul Force is truly incredible.

Although there are so many things on the screen (and I assure you there are times when the screen will be really full, ndN), the game behaves smoothly by easily managing the movement of ships, bullets, landscapes and multilayer parallax scrolling.

The variation in environmental themes and music at each level makes it impossible to get tired or bored of the game as the will to see what the next level looks like is strong.

This is a great looking and sounding game with big bosses to tackle at all levels.

In conclusion, Soul Force is an example of the incredible and resounding current scene of Commodore 64.

Go to the Protovision website and buy it. Worth it.

by **Carlo N. Del Mar Pirazzini**

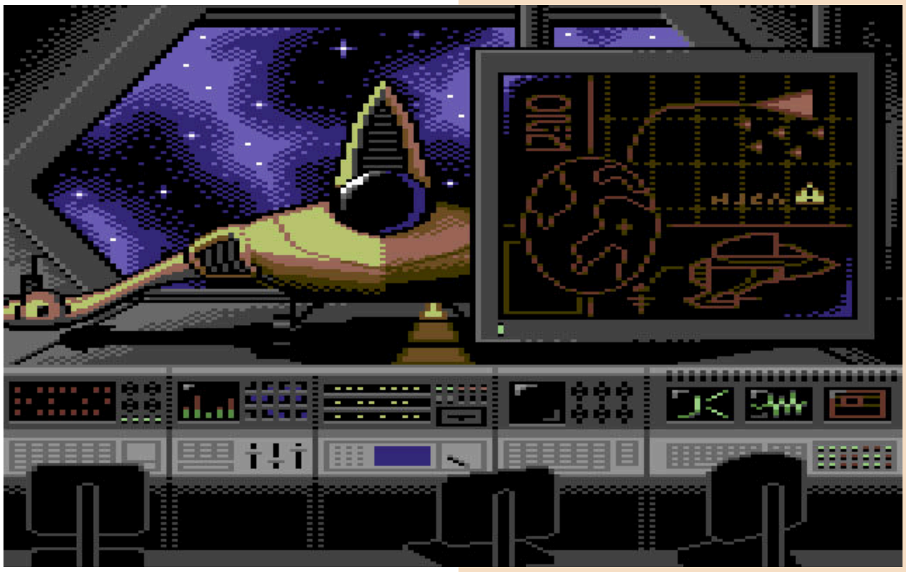
OUR FINAL SCORE

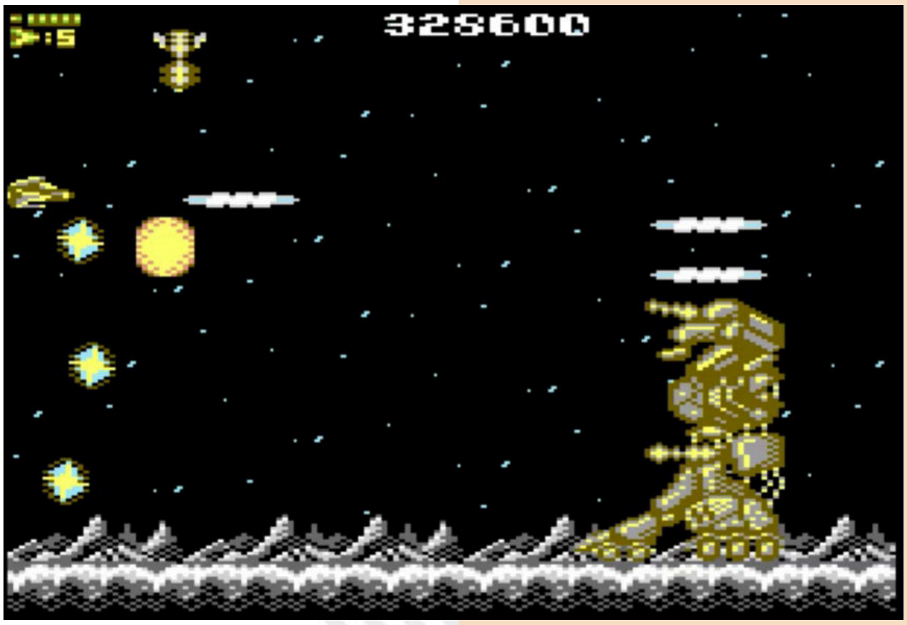
» Gameplay 95%

Everything is perfect! Power ups, additional weapons, different "firing" systems, fluidity of play in the "hottest" moments of the game.
One of the best products ever on Commodore 64.

» Longevity 95%

Increasing degree of difficulty and never destructive.
Requires attention but is never impossible to play. A modern product in a machine from 40 years ago. You will play it again and again...





STARCRAFT

BATTLE.NET EDITION

Publication date: 1998
Editor: Blizzard
Platform: PC Windows/Mac
Genre: Real Time Strategy

How do video games age?
That's a particular question.

There are immortal titles that we will play forever and there are titles that are beautiful in our minds, but then playing them again we realize that perhaps few wrinkles made them less special than before.

A video game ages well not because of beautiful graphics and great sound, but thanks to its gameplay and the rate of playability over time.

Starcraft has definitely aged well. It belongs to that series of titles that changed the way we play together, like Doom, the Warcraft series, the Lucas titles, Mario... in short, the absolute gotha of video games.

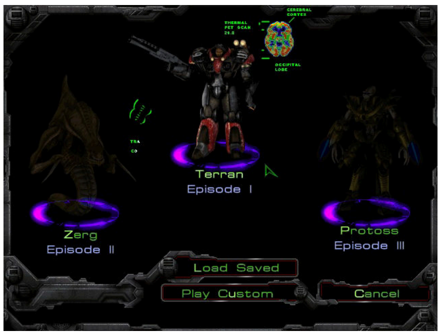
The version that I am going to review is the one that appeared on the market in 1998; it has been given for free on the Blizzard site along with its expansion Broad War through the Battle.net platform.

With 13 million copies sold, it is one of the best-selling video games for PC. The Mac version was released in March 1999 and was also converted to Nintendo 64 (indeed a good conversion) and released in 2000.

We know all about this title. Terraforming, invasion of insectoids, the superior mind of alien-nerds (this is how a dear friend of mine calls the Protoss), betrayals, building construction, upgrading and so much, so much, so much war.

It changed the much-loved concept of Warcraft 2 and brought it into space between colonies, marine spaces and aliens, expanding its development and gameplay.

Potentially endless. Not only the three campaigns to be played in a single, but thanks to the possibility of playing online (at the time we played on LAN over TCP-IP, who remembers? NdN) and the beautiful text editor becomes really endless.



Thousands of additional maps and game modes can be found on the internet.
What are you waiting for? Download and play Starcraft again in this free version.

Maybe we could arrange a nice series of battles with the RetroMagazine staff...
Do you agree?

by **Carlo N. Del Mar Pirazzini**

OUR FINAL SCORE

» Gameplay 90%
The control system is still very functional. Even after more than twenty years.

» Longevity 99%
Do I really have to tell you? The three campaigns, the expansion, the online mode, the editor... Eternal.

GHOST CHASER

Year: 1984
Editor: Artworx Software Company
Developer: Frank Cohen
Genre: Platform
Platform: Commodore 64

Here we are again at the beginning of a new year which I hope will be rich in pages, entertainment, enthusiasm and much more after a very hard 2020. 2020 Which was tough for everyone, but not for our magazine since we never stopped. We had time to write papers, reviews and interviews directly from our homes.

We also had the possibility to play games, again and again, until their completion. Who was quarantined and who was not; simply by respecting the national rules of this emergency that we are still facing.

Very recently I discovered Ghost Chaser, a fun small computer game. A little too short and difficult game, but in the right way.

The story is about a little man entering yet another ghost-infested house. From the outside it would look like a house like many others, but inside it is huge and full of rooms and basements to explore.

Each room is platform-style in which you must be careful not to fall from a certain height, avoid the ghosts that haunt you and when possible throw an object in your possession when the big ghost tries to meet us to scare us.

Our goal is to get to the basement where there is a prison and lock up the ghost that haunts us and the whole house. A task that would seem easy at first glance but once halfway, it proves difficult due to the millimetric precision in the jumps, the multitude of enemies and the amount of keys to be collected to unlock the rooms. All in a perfect maze style (and who wouldn't get lost in a Mansion, besides haunted?).

Gameplay and graphics are, as we all know, those from cheap 80s software on tapes as well as the duration of the game. It certainly cannot be attributed a low grade especially if you like ghost stories and exploration.

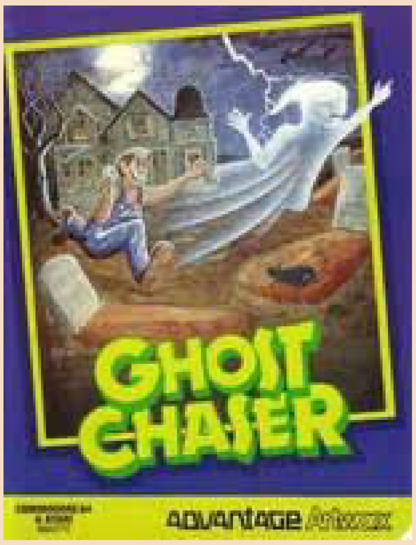
In addition, for those who can't get to the end, before the second part of the game, in the dungeon, you can enter a code that will allow you to resume from that point; making your task easier. A rarity for the games of the time.

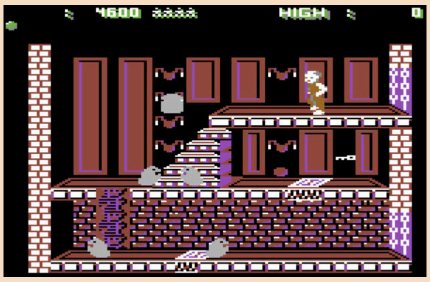
Once you're done, you might want to play it again. In short, it will not be one of those games that once finished will be destined to be touched only by the rag of dust.

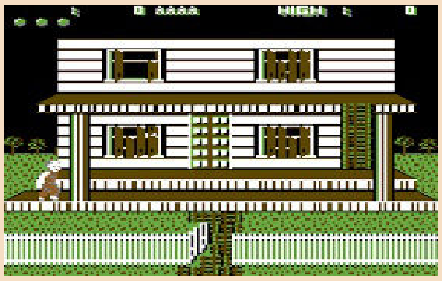
As for the sound it doesn't have much to offer but they could make it a little more atmospheric with what little music capable of creating suspense and the howling of the ghost... Instead, we must settle for the noise of our protagonist's steps and jumps. But that's okay.

Overall, a decent game.

by **Daniele Brahimi**







OUR FINAL SCORE

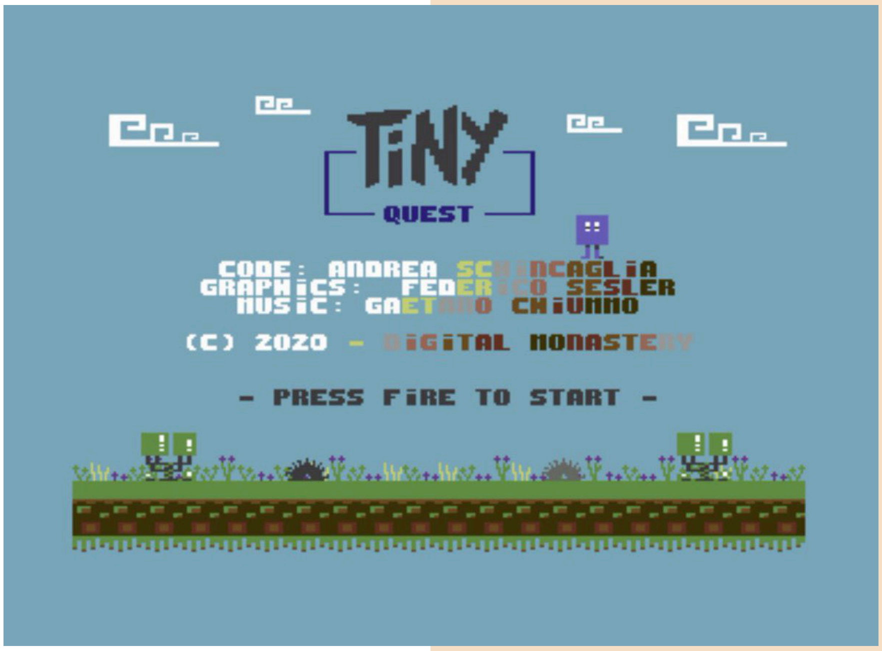
» Gameplay 70%
Not difficult and challenging just enough.

» Longevity 60%
Short... But I am convinced that you will re-play it.

NEW GAME!!!

TINY QUEST

Year: 2020
Editor: Bitmap Soft
Developer: Real Bites of Digital Monastery
Genre: Platform
Platform: Commodore 64



What a game, guys!! What a game! Simple, fast, beautiful in an absurd way!

The protagonist? A Cube, indeed Mr. Cube.

The scope? Simple, crossing his world of polygons in search of his beloved cube.

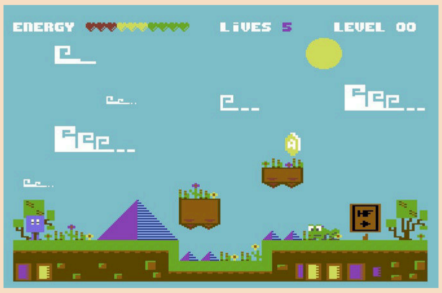
How to play? A simple joystick and a lot of skill.

Tiny Quest is the easiest and sometimes trivial thing in the video game world. Gentlemen, it's a platform, so what can you add more? It's all in "Jump, Pick, Run." But it's not trivial at all.

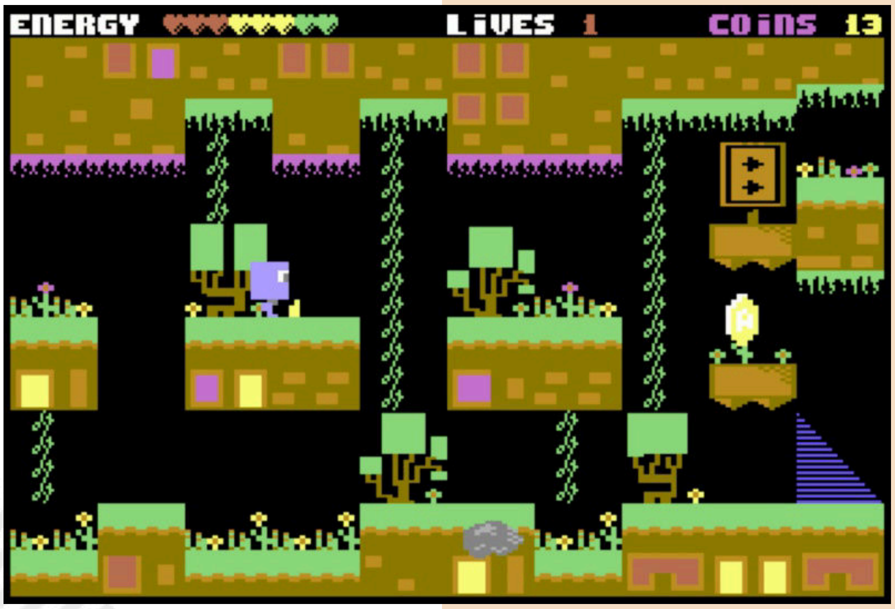
Developed by the Italian team of Real Bytes of Digital Monastery, it is a platform that requires lightning reflexes, advanced planning skills and perfect timing. Only in this way we will be able to accompany Mr. Cube through a world of 60 screens full of dangers, deadly obstacles and bewildering creatures.

Already while loading the game you realize how well the work behind its development is done. A simple but damned effective "cubic" graphic style mixed with a perfect balance of the Commodore 64 palette that gives the whole a truly "cool" style. In-game music is pleasant and adapts to the fear of travel.

Returning to the gameplay, the premise for each level is simple: we will take Mr. Cube on the right side of the playing area and we will have to take him to the exit on the opposite side. Even though our hero can run fast and is quite experienced in jumping, he doesn't have the ultimate stamina and, as such, needs to get to the next screen before his energy is completely depleted (and I assure you it runs out really quickly).



In general, this means we have about 8-10 seconds to complete a screen, otherwise we will end up losing one of your five lives.
Coming into contact with traps and



enemy creatures or falling into pits will also cause a loss of human lives, indeed not... cubic lives.

To ensure a safe passage home for both himself and his sweetie, Mr. Cube needs to raise enough money during his search. Each screen will contain a coin to collect and it is only then that the output signal is activated and we can safely escape.

This frenzy is Tiny Quest's real stroke of genius. The game will keep us glued to the screen. Jumping between platforms, barely avoiding instant death and getting to the exit point is immensely fun and compelling. Every time we lose a life, the action is restored and starts again almost immediately. This game will force us to think fast and react as quickly as possible. And to hell with real life, work problems, covid and all that... we'll stick to the joystick until we get through it all.

To avoid heart failure or the possibility of orthopedic damage to our fingers, each set of 15 screens will be provided with a password that will allow us to start over from the pre-set point. However, it is good to remember that these passwords only last for the current game session and are reset when you reload the game (how cruel, ndN)

Tiny Quest does not offer anything too thorough, nor a plot at the Final Fantasy, nor ultra violence or absurd tutorials, but what is there is very funny and the challenge within the game is sufficiently balanced to make it somehow compelling. A jewel!

Tiny Quest (Disk and Cassette Edition) can be ordered from Bitmap Soft at this address:
https://www.bitmapsoft.com

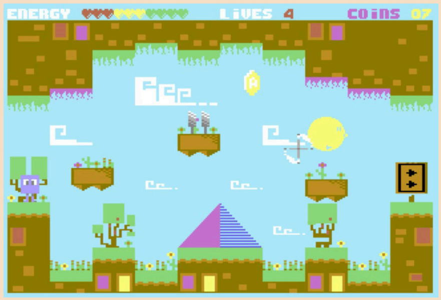
Don't miss it.

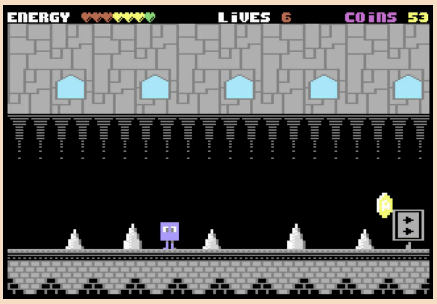
by **Carlo N. Del Mar Pirazzini**

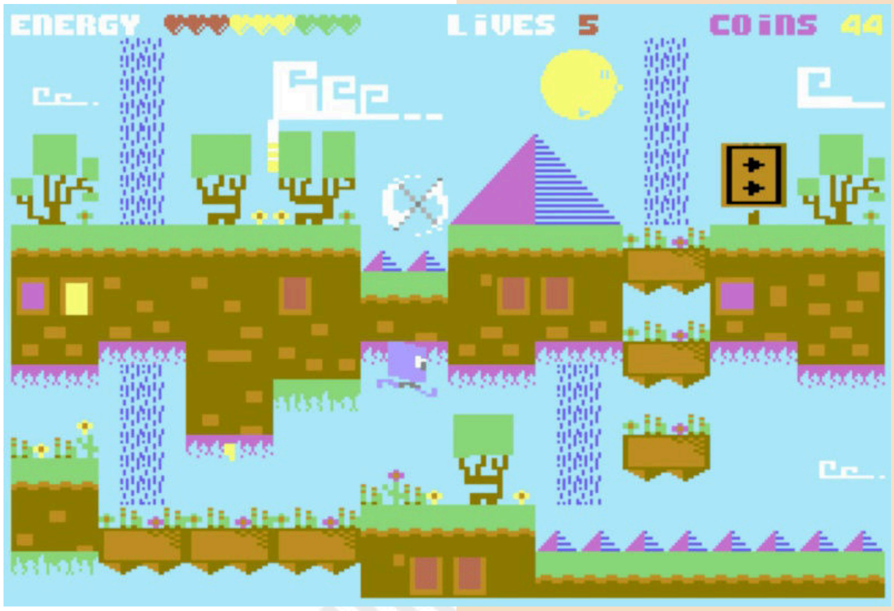
OUR FINAL SCORE

» Gameplay 90%
Simple, straightforward, tutorial-free. Old school.
Lets you play with ease as you did with the old titles of yesteryear.

» Longevity 90%
Pay attention!!! Difficult!
That gentle difficulty that tests our reflexes and brains. You won't finish it easily.







PIER SOLAR AND THE GREAT ARCHITECTS

Year: 2010
Editor: WaterMelo
Developer: WaterMelon
Genre: JRPG
Platform: Multiple
Reviewed version: Sega Megadrive

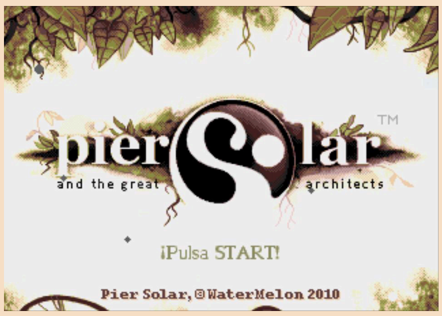
Pier Solar and the Great Architects is a homebrew role-playing game developed and published by WaterMelon for Sega Megadrive. The game was released in December 2010.
The story centers around three best friends: Hoston, Alina, and Edessot. Hoston's father gets sick and the three friends venture out to find a rare magic herb to cure him.
This story later unfolds in a much broader plot surrounding the game itself.

Characters
Hoston is a young botanist, educated by his father. He's a carefree, relaxed young man. When his father gets seriously ill, Hoston realizes he is the only person who can save him and heads to Reja's forbidden caves where he hopes to find the herbs that will save his father's life.

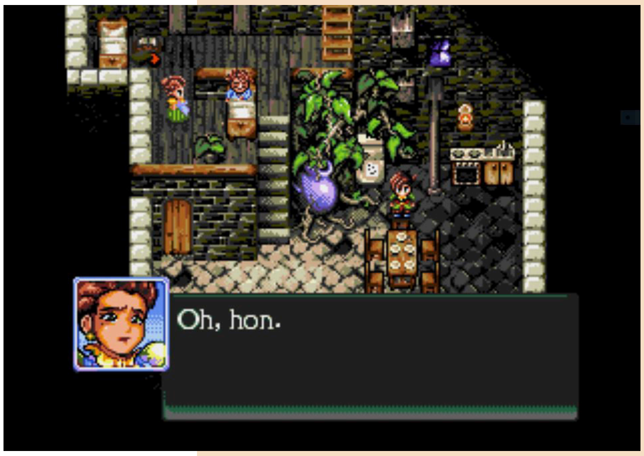
Alina was adopted at a young age. She grew up with Hoston and Edessot as her only friends and is desperately trying to convince her father to approve. Even if she is superficially strong and responsible, she reveals herself alone in the end, because she feels that Hoston and Edessot are the only people who really take care of her. She is protective of them and in many ways considers them her real family, although neither Hoston nor Edessot understands it.

Edessot is a mechanical genius and mature beyond his years. He is part of a wealthy and broad-minded family and has been free to explore the world to his liking. Usually happier alone and tinkering with machines, he still enjoys spending time with Hoston and Alina, having a particularly strong bond with the latter.

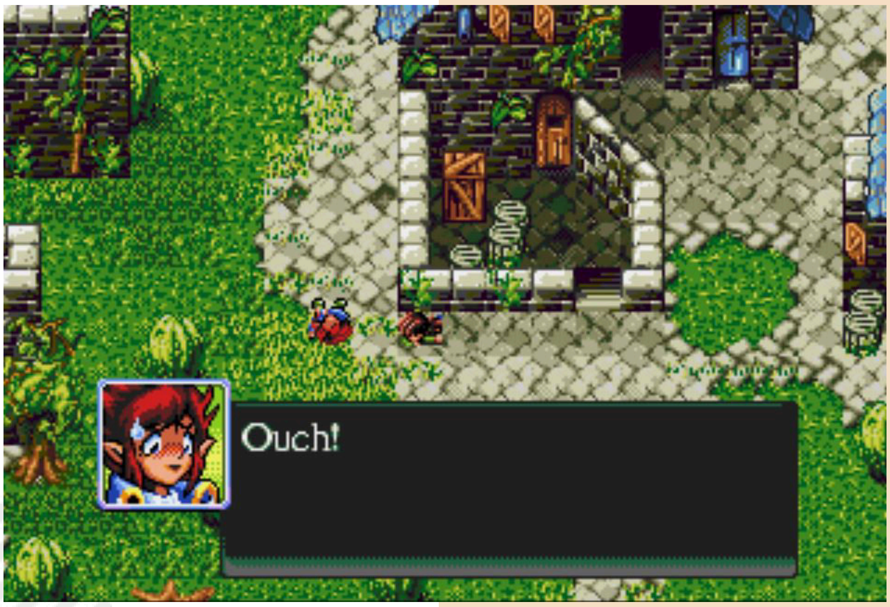
Game development began in June 2004 as a small project within a web community dedicated to developing homebrew products for Sega consoles. Initially it had to be a simple role-playing game based on community members and the target platform had to be the only Mega CD.
As time progressed, the development abandoned the initial and simple idea in favor of a fantasy product on a larger scale.



The team abandoned the idea of the Mega CD and started thinking about a cartridge version for Megadrive and the use of a product on a 64 megabit memory system, in all respects the biggest game for the Sega system.


Oh, hon.

The game's goals have become increasingly ambitious, with the development that has gone from Mega-CD, for which CD-ROMs would have been inexpensive to produce, to Sega Megadrive, a system that uses storage media based on more


Ouch!

expensive cartridges. To allow the transition without having to reduce the amount of game content, it was decided to use a cartridge with 64 megabits of memory, making it technically the "largest" game cartridge for the system, finding a way to use the hardware top sound of the Mega-CD at the same time (if the device was present).

The game was released in December 2010, two years after the initial release date. Three different versions of the game have been released: Classic, Posterity and Reprint. The Classic and Posterity editions each have three different language packs, while the reprint includes the three most common European languages: English, French and German. The Japanese language pack originally included the Japanese and English languages, but the Japanese language was abandoned and French and Spanish included due to the lack of volunteers to correct the drafts of the Japanese translation.

Even before the official release date of the game, the game was already sold out via pre-orders. Due to the enormous demand, WaterMelon decided to produce a second edition, also limited, with a so-called "Reprint Edition", sold out in 12 days. WaterMelon announced the production of additional copies on Thursday, September 15, 2011. The third and final expected reprint of the Mega Drive/Genesis cartridge was due on March 25, 2014. It still had to happen, but Watermelon had assured the fans that they would be ready by the end of April / early May. However, this release date has not been respected either. This reprinted edition finally started shipping to customers at the end of February 2015.

Now, after all this preamble on the history of the game, let's go analyze it.

How can you define this Jrpg?

Decent... nothing more, nothing less.

Technically flawless, indeed almost to the limit of perfection, but lacks something.

The gameplay is too frustrating, with its design in some places labyrinthine and often accompanied by too many long animations and boring and repetitive combat phases. More could be done. A product with an incredible look but little gameplay and this, for a gdr, is terrifying.

The soundtrack is incredible. Perhaps the most careful and impactful part, which manages to give atmosphere and which will make you better digest the (numerous) frustrating moments present.

If only you had dared more in the real game phase we would have the best role-playing game for this console. But the podium on Megadrive is full of titles of greater impact and greater care in gameplay.

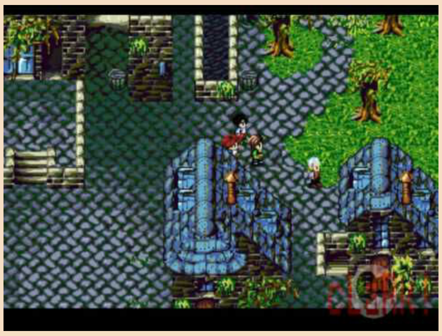
And that's it, Bardo's word.

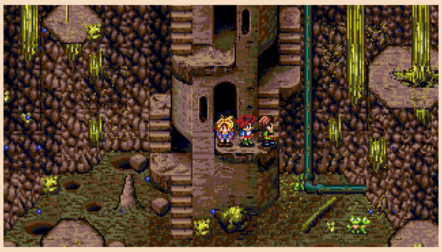
by **Roberto "il Bardo" Pirazzini**

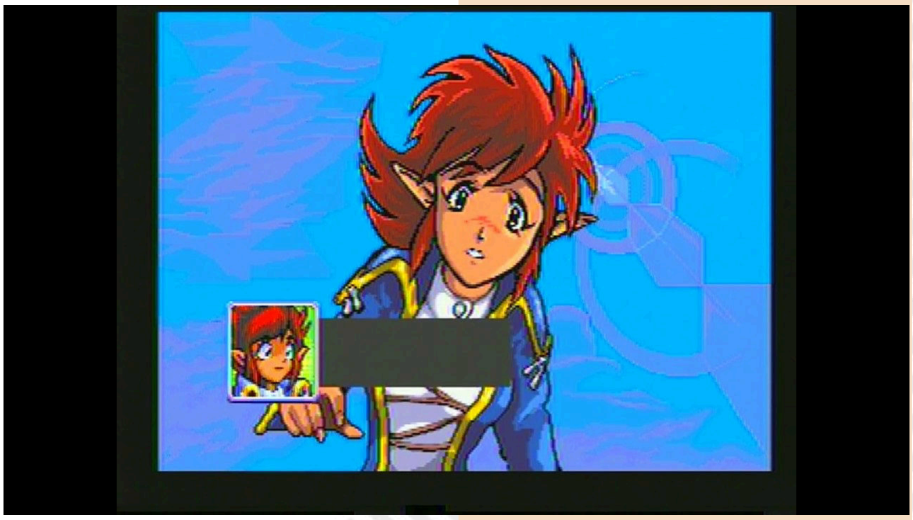
OUR FINAL SCORE

» Gameplay 55%
Game system with cumbersome menus, frustrating in some parts of the exploration and very with repetitive fighting. Too bad.

» Longevity 70%
If you survive the frustration, the game will put you in front of numerous quests and a quite nice adventure.

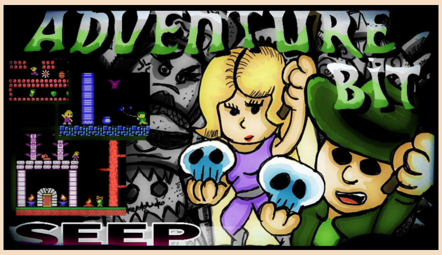


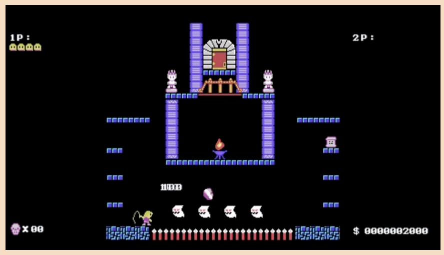


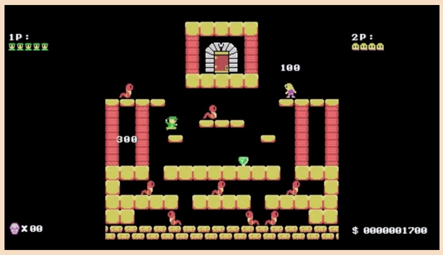


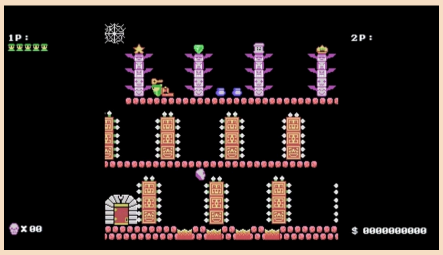
ADVENTURE BIT

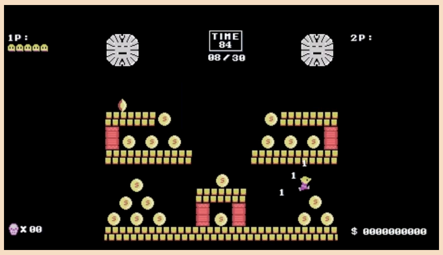
Year: 2020
Editor: Seep
Genre: Adventure
Platform: PC











Hello friends of Retromagazine World, today thanks to Adventure Bit our journey through time takes on a new dimension, different but equally fascinating. The good DeLorean is parked by the mechanic and stays there after the myriad of raids he accompanied us. Doc Brown tells me he's got a lot of fixes to make on the car, and believe it or not, these scientists always know more tricks than a cartful of monkeys.

The other day something incredible happened to me: I was quietly working on the computer, detaching myself between 3D polygons and design studios when suddenly the laptop started to light up with its own light. Sparks, electric shocks, and smoke poured out of the slits as the screen suddenly turned black. Instinctively I leap backwards as a rainbow of colors begins on the screen, the ones that flooded our screens during our childhood video game uploads. Writings appear on the blue screen, they read: SEEP, 21111 bytes free, disk, okay. In the lower part of the same color, car, goto, list, run. What the hell is going on here? Is my laptop owned by any Malaware? No guys, it was just hit by Retrogaming Mania too! The SEEP name is a guarantee in this case: reading it at the beginning of the screen relaxes me a bit, I understand that I am only dealing with the new magic of the Giansoldati brothers. It is very reassuring for a retrogamer to know that there are still people like them in the world, feeding the “old-fashioned" video game market with brand-new titles. Perhaps not everyone knows this small Software House, so a little review I think may be useful: SEEP was born in 2015 in Turin, and for me this is twice a source of pride. Brothers Sergio and Enrico are two Turin brothers passionate about retrogaming with the habit of developing new video games with an old-school flavour. The story tells us that they started as modders, and then gave vent to their passion by starting to program and auto produce video games. And what games, gentlemen, we are faced with real masterpieces full of the flavour of the past.

Authentic gems like the commodorian KATANA SOUL and the damned "SNK Oriented" THUNDERFLASH, which we will have a chance to talk about in the future, give the real size of this team of developers. All this work is dictated by the passion that aims at a path of constant growth in game design to offer increasingly attractive products. The two games listed above are a hymn to retrogaming, they pay homage and bring to the forefront of the genres now placed in the trunk of memories. Types of games that were a must in the 80s that then experienced a sharp decline in interest over time, from about the middle of the 90s, where games have increasingly aimed to imitate the real world, with its few strengths and many defects. These titles are a tribute to the video games that marked our childhood as video players, the golden age, in which playing was projecting into a parallel world, absolutely different and detached from the real one. Adventure bit does not minimally change the focus of the two titles mentioned above: it wants to rediscover the beginnings of the video game industry even for those who have not experienced it first-hand. Personally, after a few minutes of playing, I came up with a handful of old masterpieces that absolutely deserve to be played again and passed down to those who may not know them, so that they can get passionate in turn, perhaps just starting with this new game. I read some time ago, around the web, a phrase by Sergio that struck me a lot: "Sometimes the titles of the past are unfairly called games. In fact, behind it is a lot of study on what concerns game and level design. " Never were words truer.

The SEEP guys usually spoil me by offering me the chance to test their games and for me it's always an emotion since these are very well made Old School titles, stuff that make me fly straight back to the Eighties. I think it is necessary to start talking about the game by giving Sergio a voice, who kindly accepted my proposal to send me some curiosities about it. "Adventure Bit is inspired by the great classics for MSX, bringing back colors and resolution of the original hardware, all adapted to be played safely on Windows PCs." In fact it was a blow to the heart to see the palette of the computer mentioned above, a machine of all respect that would have deserved much greater success. The graphic is really beautiful, super faithfully reflecting what the platforms of the past were: colors and animations are a real time machine boys! The Giansoldati brothers go on to say that "The game is inspired in particular by Montezuma's Revenge, Rick Dangerous and Pitfall". Absolutely, they are the first three titles that come to mind when playing it, but in my opinion, when you play in two, there is much more. What are you talking about? Ah yes, I didn't tell you: you can play in two together, bringing the level of fun to other dimensions. And player two is a tribute to an initially controversial character who wrote the story of the eight-bit platforms, the brat GIANA SISTERS who made us have so much fun on C64. Not only does the character recall its aesthetics, but some screens have the same color choices, even the money you have to take seems to recall the gems on the famous Rainbow Arts platform.

Sergio confides a secret to me saying that "Initially the game was designed not to have music, then we changed our mind and in fact so the adventure is more appealing!". And it's a good thing they decided to put the music on it since they give the game the frame that transforms a beautiful painting into a masterpiece. The SEEP guys tell us that "The main theme is the work of Andrea Baroni, a professional composer who has already collaborated with us in Thunderflash and Katana Soul". We can only take off our hats and pay homage to such skill!

Sergio also illustrates the modus operandi that led to the birth of the game in question. "The development time of the game was short but spread throughout 2020. It was born almost in the "recreation" phase, we wanted to create something just fun and funny. In my personal opinion, this is the game that I enjoyed the most ". The gems are not lacking: "For the occasion, as good old people we are, we've seen all the Indiana Jones movies again. And there we decided to add the crystal skulls to be recovered as an extra. " The deepening continues, the info continuously follows: "Adventure Bit may seem like a small game, but we have experienced a totally different game design approach different from the past. In other words, the difficulty is growing, with a really low curve at the beginning, and then getting harder and harder. The result is in our opinion (and by many feedback we received) satisfactory".

The team confides in us that "There are two Easter Eggs hidden. The first can be revealed without problems: in the totems in play and on the cover, you can see the two faces of the programmers: one with glasses (Enrico), the other angry (Sergio). The other Easter Egg is about one of our games published in the past, to see it you simply have to end the adventure in all modes!". Does the title of the game also hide something from us? Why this name? Adventure is seeing that the platform is themed, but the word bit?
Here's the answer: "That final BIT in the title is an acronym that we also used with Abduction Bit 2016. The idea is to create a small series of minimal games with this style. So in the future we do not exclude a third chapter. " And we will be here, eager, ready to play it to the death!

by **Michele Novarina**

Download:
https://store.steampowered.com/app/1460050/Adventure_Bit/

BIKER REFLECTIONS

I'm obsolete, not old. And as a good old-timer, I'm more and more happy to be able to play something old-school, even if it's new programming. The hours of the day to relax and play are less and less and wandering around in some open world does not pay the time invested in doing so. A healthy, good and direct match to the glories of the past or titles like this Adventure Bit leaves behind that sense of satisfaction and carefree that we felt in that golden age. Whiplash through mysterious ruins and temples of ancient times. The enemies are not lacking, the action is pressing but never suffocating. There are traps worthy of George Lucas films but with the right amount of practice in movements they can be overcome. The difficulty is perfectly balanced, you always have the impression of being able to proceed even in difficulties and this never leads to having that terrible feeling of ineptitude. There are many levels and the two player mode gem together on screen brings longevity and fun to the top. A game that obsoletes like me will certainly like, but that will also amaze the new levers that perhaps need to make a journey in pixel art, detaching themselves between platforms and jumps to be made precisely. Honestly at the price of 1.59 euros the game deserves great. Even the pirated tapes in the back of a store weren't that cheap. The old three thousand lire, I also dare you to look for something less in the newsstand! A small figure for a great game that has the advantage of sliding into an era where everything tasted different, where games did not want to seem like a replica of reality but a dimension parallel to it. And it was so beautiful all this, it was carving out a corner of our own where we could put the world aside and live our adventures pixel after pixel.

SUPER MARIO RPG LEGEND OF THE SEVEN STARS

Year: 1996
Editor: Nintendo
DEveloper: Square Soft
Genre: RPG/Platform
Platform: Super Nes

I was young (early 2000s) when I first played a Mario game (the game was Mario 64 on Nintendo 64) and have since fallen in love with the series: Super Mario Bros. 3, Super Mario Galaxy and many others all related to the original series.

As the only spin-off I always followed the Mario Kart series. I liked to run through the crazy circuits hitting the opponents with the sound of shells. Some time later I was struck by the idea of a role-playing game that I knew existed and that came from the Super Nintendo.

Thanks to Nith I was able to get my hands on this title a short time ago and decided to play it.

Super Mario Rpg: Legend of the Seven Stars begins with Mario's typical quest: rescuing Princess Peach from Bowser. Everything as planned. Mario just saves her from the situation, but a new villain appears who throws Mario, the princess and Bowser in different places, scattering fragments of stars throughout the mushroom kingdom.

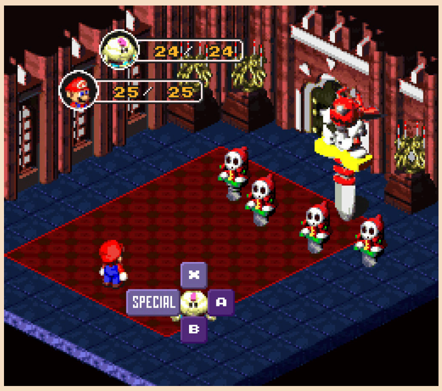
Released in 1996 on Super Nintendo, the game uses 3D rendered backgrounds very similar to those seen in the Donkey Kong Country series and let's say it does very well by showing the console's capabilities and the synergy of two companies, Nintendo and Square soft, which worked together to create a great product. 2D sections are not great, but they still do well.

What stands out about graphics in Mario RPG is the simplicity that captures you and the smoothness with which the graphics engine moves everything. Playing with Mario in 1996 with Mario's 3D rendered characters was incredible and even in 2021 it is a great sight to see.

It is a rotating role-playing game with numerous random encounters during the game, along the lines of Final Fantasy (made by Square too). During combat the control system is quite easy to learn and it is easily managed by using the coloured Snes buttons. Each button has its own function (attack, parade, magic, inventory...). Some action commands must be applied during battle to cause more damage or even to take less damage. Everything is immediately explained but then there are changes to what was expected from this genre of game.

The magic system works easily and during exploration we will find shops where we can buy potions, cures, mushrooms and other useful gadgets to use during the quest. This is on the role-playing side that Square Soft helped to develop, doing a great job.

Nintendo, on the other hand, had its



SMITHY: NOOOOOOOOOOOOO...!!!

Only For Nintendo
Mario Returns in a Wild New Role-Playing Adventure!
SUPER MARIO RPG
LEGEND OF THE SEVEN STARS
SUPER NINTENDO ENTERTAINMENT SYSTEM
INSTRUCTION BOOKLET

fair share of Mario platforms ready to be distributed along the many levels.

The game view is isometric and this also renews the genre. In short a winning mix between Final Fantasy and Mario.

The sound sector is really well done with Yoko Shimomura (Composer of Kingdom Hearts) providing such magical, dark and decidedly compelling themes. This great woman succeeds perfectly when it comes to making the player feel the momentum.

The Fight Against Smithy theme that is played during the final match is a theme that always makes me headbang or clap my feet because of how eavesdropping it is. I can name others as the theme of the battle or both themes of the battle of the boss. Music in Super Mario RPG is a joy to listen to.

A game that unites two worlds? Yes, it is. Can you do that? Yeah, definitely. And it is to be considered one of the best examples of mixing ever.

Unique, I wish it had been longer.

The challenges are not so difficult and this greatly penalizes its final longevity.

But in addition to this, this is a must for fans of both genders.

by **Hakim Rezki**

OUR FINAL SCORE

» Gameplay 90%
In perfect Nintendo style, nothing is left to chance. Simple to learn and effective during the game. Few commands and all of them in the button panel of the wonderful joypad of the Snes.

» Longevity 60%
Barely sufficient. There is fun but it lasts little, too little and is not particularly difficult to finish... Only sore point.

Only For Nintendo
Mario Returns in a Wild New Role-Playing Adventure!
SUPER MARIO RPG
LEGEND OF THE SEVEN STARS
SUPER NINTENDO ENTERTAINMENT SYSTEM

NEW GAME!!!

HERMES
RUN A LA CARTE

Year: 2020
Developer: Retroguru
Genre: Jump'n Run
Platform: Various
Review: Amiga

Hermes, lives in France, the land of love also known for its exquisite cuisine.
One day he gets hungry for grilled chicken. Certainly a man like Hermes wouldn't go to the supermarket for frozen chicken. No, no, no! He's about to slaughter his farm-raised free-range chicken, but this chicken has a mind of its own. This is where our story begins...

Violent, irreverent, dirty, sometimes vulgar. Hermes is jump 'n run with a lot of bad humor. If you are an avid vegetarian, you may not appreciate the game or you may have a healthy, fat laugh.

Purpose: chase the chicken, collect the donuts, avoid the opponents, drink the potion/cola but in all this you must be careful not to starve (that's why we will eat the donuts) or surplus food that will cause a terrifying diarrhea!

When we reach the limit, Hermes himself will inform us not to continue eating and to stop at the bathroom that we will find scattered throughout the levels.
I mean, a crazy game! Cute, worthy of the Retro Trash column on our facebook page.

The game was created by the Retroguru group, a collective of coders, graphics and musicians working on different platforms. A group mostly French. They are specialized in fast and hectic games that have the advantage of taking away little time and let us forget the bad days with a laugh.

Very well planned and realized. The retro style graphics make it eye-catching and the music is a real treat for the ears.

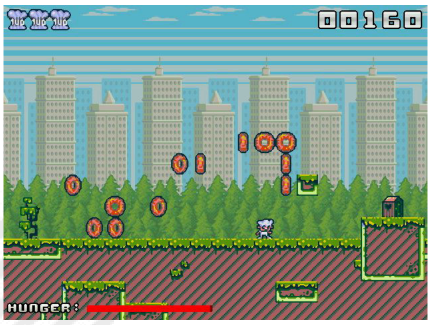
Playable with both the keyboard and a pad, it does not require you to use your brain but a fair dose of manual skills to avoid the numerous traps along the levels.
The game extends in 32 levels with a fair amount of difficulty that increases its overall longevity.

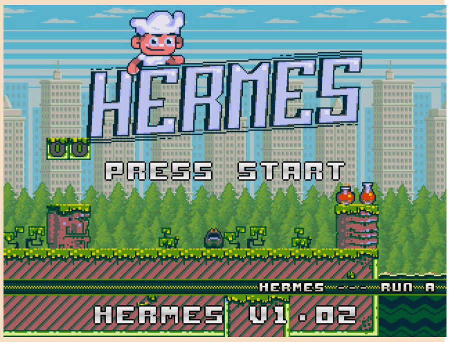
It is a versatile product released for an incredible number of platforms: Amiga, Android, AROS, PocketGo, Caanoo, Haiku, Linux, MorphOS, Mac OS X, NetBSD, Nintendo 3ds, Nintendo GameCube, Nintendo Wii, Pandora, Playstation 2, Playstation Mini, PSP Portable, PS Vita, Raspberry Pi, Symbian OS and Windows. I mean, you find it everywhere.

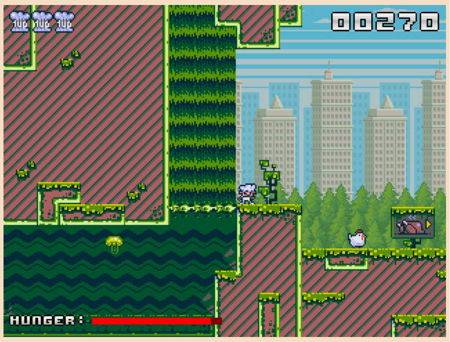
The Amiga version we reviewed requires 1 Mb of Ram and runs on both Aga and ECS chipsets, so all you need is a plain AMIGA with expansion (where I tested the game, ndN). Now all you have to do is chase the chicken, grab it and make an unforgettable grilled chicken.

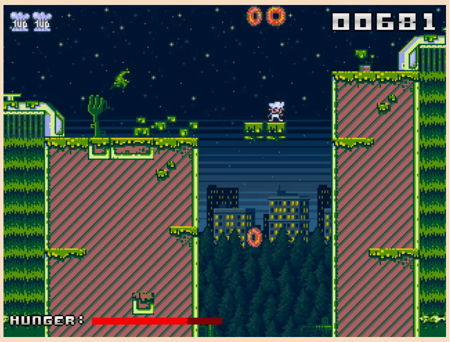
by **Carlo N. Del Mar Pirazzini**

P.S. - I forgot to mention... it's FREE!









OUR FINAL SCORE

» Gameplay 75%
Simple and immediate. It doesn't require a master to play it nor thousand of hours to learn.

» Longevity 65%
The game is difficult but not impossible. Once you learn the patterns you will succeed in the task. Every now and then you will play it again because it is fun.

Death is only the beginning

January 2021. Start of the fifth year of RetroMagazine World.

I never had a chance to ask our Chief Editor Francesco, but I'm pretty sure that when he started this adventure at the end of 2017 as a "one man army," he would never expect the team to grow day after day to become what it is today. Five years is a great result, especially if you think that it is more or less the duration of the 8-bit universe: exploded globally in early 1981 with the VIC-20, already in 1985, with the presentation of Amiga (16 bits) at the Lincoln Center in New York, it was becoming history.

As we know, the C64, which many of us had as our first machine, was then produced until 1994 (well beyond the successor C128 who saw its official end in 1989), but there is no doubt that already in 1986 the world was evolving towards something new.

And yet, 40 years later, we are here.

I had the privilege of meeting Dave Haynie personally, one of the designers of the C128 who then passed to Amiga who, discovering that my C128 is still working perfectly in my studio, told me something like, "We never consciously designed them to last this long, but we did it with passion."

This is perhaps the keystone and the secret of alchemy that brings us regularly among our readers month after month: it will also be nostalgia, it will also be the attempt to go in search of lost time or even an unhealthy Peter Pan Syndrome, but what we do we do with passion and love. Today as yesterday, we are united by the desire to study, experiment, discover, go beyond our limits and learn from each other.

For this reason, the so-called 'console war' is contrary to our nature and any contribution is welcome. We have never snorted any machines and we are looking, as already written in other closures, for willing people who share our passion and help us fill our gaps (for example, about Atari ST).

It is the beginning of the fifth year and, as it was written on Dave's shirt that day when I met him related to Amiga's "death" and Commodore's bankruptcy: "death is only the beginning". Retrogaming and retrocomputing are here to stay. And we're with them!

Gianluca Girelli

RetroMagazine World
Year 2 - Issue 6 - MARCH 2021

Chief Editor
Francesco Fiorentini
Managing Editor
David La Monaca
Editing Manager
Marco Pistorio
Web Manager
Giorgio Balestrieri