A journey documenting the Sanco 8003 computer

—

FOSDEM 2024-02-04
Giovan Battista - giomba

- MS Computer Engineering @ University of Pisa
- Embedded software developer
- Retrocomputer Enthusiast

Find me on...

- www.giomba.it
Giulio - giuliof

- MS Electronic Engineering @ University of Pisa
- Firmware engineer in machine vision field
- Too many hobbies to fit in three-line bio

Find me on...

Github: giuliof
Mastodon: @giuliof@mastodon.uno
Twitter (?): @GiulioFie
Personal: me.giuliof.it/en/
A computer murder was prevented that day...

However... we owe a special thank you to “Associazione Porte Aperte” for this gift!
What is this?

- Computer from the 80s
- Runs CP/M
- Possible italian rebrand
- Very little information from the internet

http://www.vintads.it/file.php?cod=802
Motherboard

Z80

+ Some standard Z80 peripherals
+ RAMs and ROMs
+ Tons of 74LS logic gates!
System ROM

Standard 2732 (4KiB)
Z80 instructions
Mapped at... 0xC000?
“BIOS”

https://z88dk.org/
https://sigrok.org/wiki/PulseView
https://github.com/GLGPrograms/ceda-rom-disassembly
Bus inspection setup
.text and .data not found
Characters generation

Standard 2732 (4 KiB)
8 pixel x 16 rows
256 bytes ROM

old 28L22
narrow PDIP
dump with custom hardware
glue logic?
First “hacking” attempt
From the board to the schematics

Lots of manuals and datasheets  Checking connections  Drafts and connection ideas
X-raying a multi layer board

- Four layer board
- Inner layers: maybe supply rails
- Tracks are all on the outer layers...
- ... but some of them hide under ICs!
Follow that wire!

How do we know if a path was already hit?

Just take notes on an actual photo using GIMP :)
What’s inside a Sanco?

- Schematic is now on KiCad
- About 90% of the board is mapped and documented
- FDC circuitry is pretty much at early stage (help required!)

We’ll say something about these four in a second

https://github.com/GLGPrograms/ceda-schematics
Memory map

- 64k DRAM fully addressable with bank switching system
- Dedicated 4k video SRAM
- Further 4k SRAM
Boot circuit

- Z80 boots from **0x0000**
- ROM isn’t mapped there...
- A **latch** forces ROM to be mapped everywhere, just at boot!

```
0x0000 System ROM
      System ROM
      ...
0xB000 System ROM
0xC000 System ROM
0xD000 System ROM
0xE000 System ROM
0xFFFF System ROM

0x0000  Dynamic RAM
0xB000  Alternate RAM
0xC000  System ROM
0xD000  Video RAM
0xE000  Dynamic RAM
0xFFFF  at boot
        at runtime
```

From Z80 BUS_ADDR multiplexer:
Is set when Z80_BUS_ADDR[15:12] = 0xC
Video Generation
Combinatorial network

\[ f(A, B, C, D) = E(6, 8, 9, 10, 11, 12, 13, 14) \]

\[ F = A\bar{C} + A\bar{B} + \overline{B}C + A\bar{D} + D \]

\[ F = (A + B)(A + C)(\overline{B} + C + D')(A + D') \]

this is it now, feel old yet?

256 bytes ROM
We could have surprised you with special effects...

- invert
- horizontal stretch
- vertical stretch
We could have surprised you with special effects...

- invert
- horizontal stretch
- vertical stretch
ceda2vga

Sanco 8003
a desktop computer

https://git.giomba.it/giomba/ceda2vga
Boot from floppy

- We don’t have any floppy for Sanco...
- We know a little about how a floppy is supposed to work...
- ... but we have the boot code
- ... and a CP/M disk image (thanks to bayo7 and pconseil)

```asm
: Boot trampoline executed when BOOT key is pressed
bios_bootkey:
  ld    de,$0000 ; track = 0; sector = 0
  ld    bc,$4000 ; cmd = read ($40); drive = 0
  ld    hl,$0080 ; load in $0080
  ld    a,$01 ;
call   fdc_rwfs ; invoke bios routine
  cp    $ff ; check for error...
  jr    nz, bios_bootdisk ; if ok, go on
  out   ($da),a ; ... else, beep and retry
  jr    bios_bootkey
  out   $(b2),a ; sound speaker beep
  jp    0080 ; run loader
```

https://github.com/GLGPrograms/ceda-cpm
Sanco CP/M floppy format

- ROM loads the “loader” from the boot track
- the “loader” loads the CP/M BIOS, BDOS and the CCP in RAM
- control is given to the CP/M, which finishes loading

<table>
<thead>
<tr>
<th>Tracks</th>
<th>80</th>
</tr>
</thead>
<tbody>
<tr>
<td>Double side</td>
<td>2</td>
</tr>
<tr>
<td>Sectors per track</td>
<td>5</td>
</tr>
<tr>
<td>Bytes per sector</td>
<td>1024</td>
</tr>
<tr>
<td>Total capacity</td>
<td>800k</td>
</tr>
</tbody>
</table>

**Boot track?**

```
Side 0
S  S  ...  S  S
S  S  S  S  S  S
Track 0
```

```
Side 1
S  S  S  S  S  S
S  S  S  S  S  S
Track 1
```

```
S  S  S  S  S  S
Track 2
```

- **Side 0**: 0-255 bytes (256 bps)
- **Side 1**: 256-511 bytes (1024 bps)

The “loader” loads the CP/M BIOS, BDOS and the CCP in RAM.
Floppy-o-burner do it yourself

- Custom Z80 assembly code, small serial parser with read, write and format commands
- Python script that feeds the whole disk image, track per track, through serial
Keyboard

1200 8n1

https://github.com/GLGPrograms/ceda-ps2-keyboard/
Emulator

CEDA-CEMU
a Sanso emulator by RETROFFICINA GLGPROGRAMS
http://retrofficina.glprograms.it/

Working:
- Z80 (lib)
- Mode2 IRQ
- CRTC
- SIO/2 peripherals
- keyboard
- integrated monitor, breakpoints

Work in progress:
- PIO
- Floppy

Future:
- More tests
- More tests
- Did I already say more tests?

https://github.com/GLGPrograms/ceda-cemu