A journey documenting the Sanco 8003 computer

—

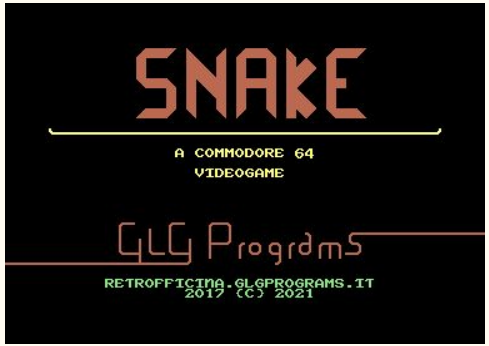
FOSDEM 2024-02-04

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/

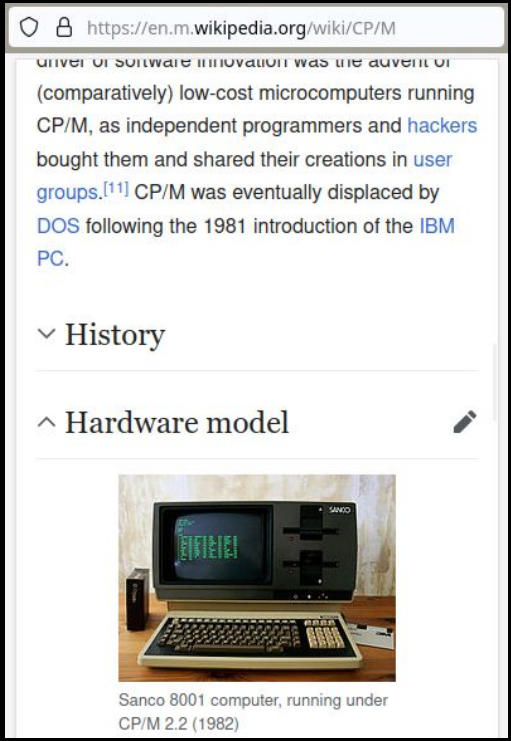
A computer murder was prevented that day…

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

+

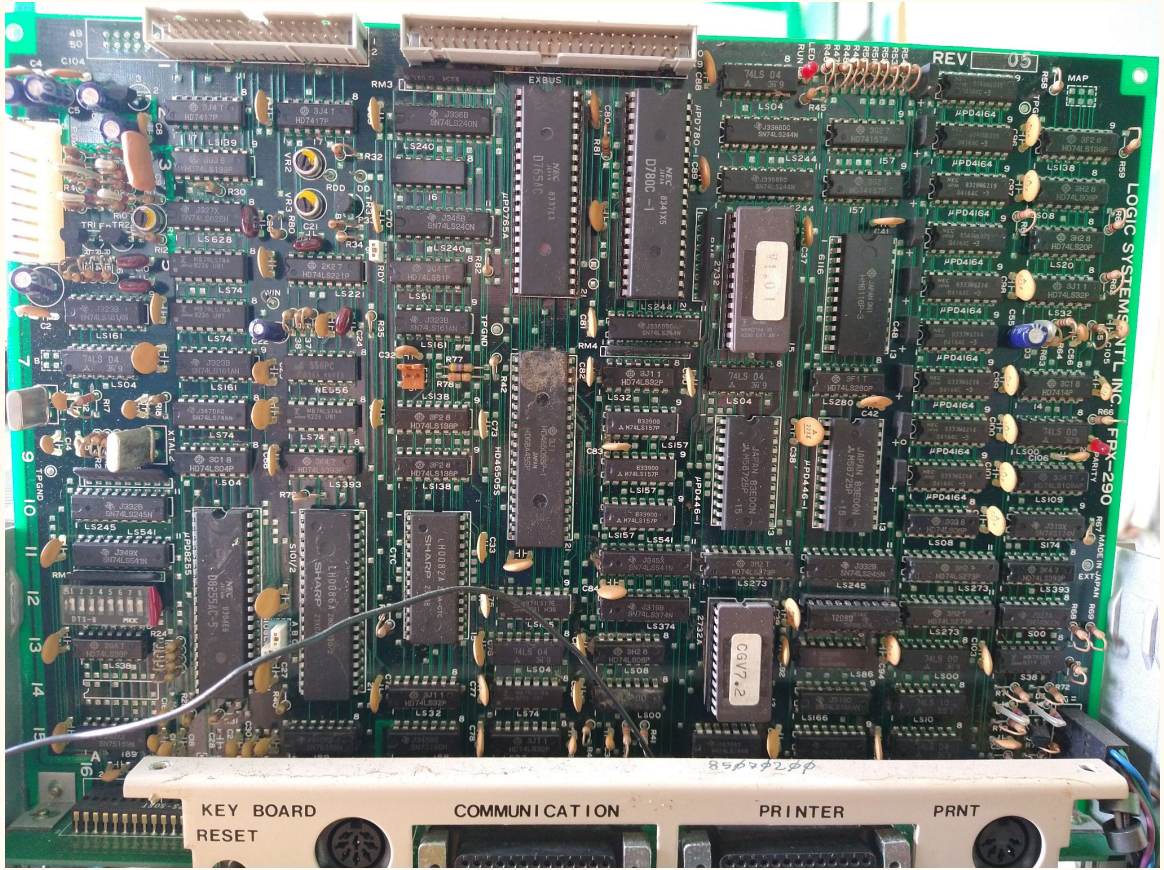
Some standard
Z80 peripherals

+

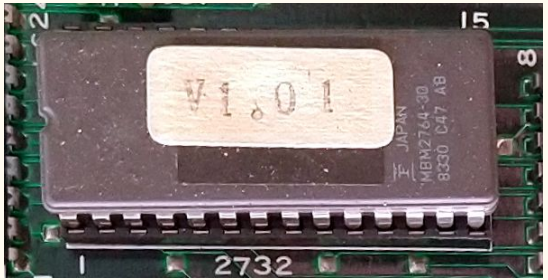
RAMs and **ROMs**

+

Tons of **74LS** logic gates!

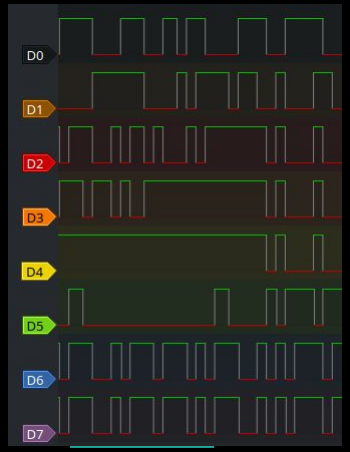


System ROM



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

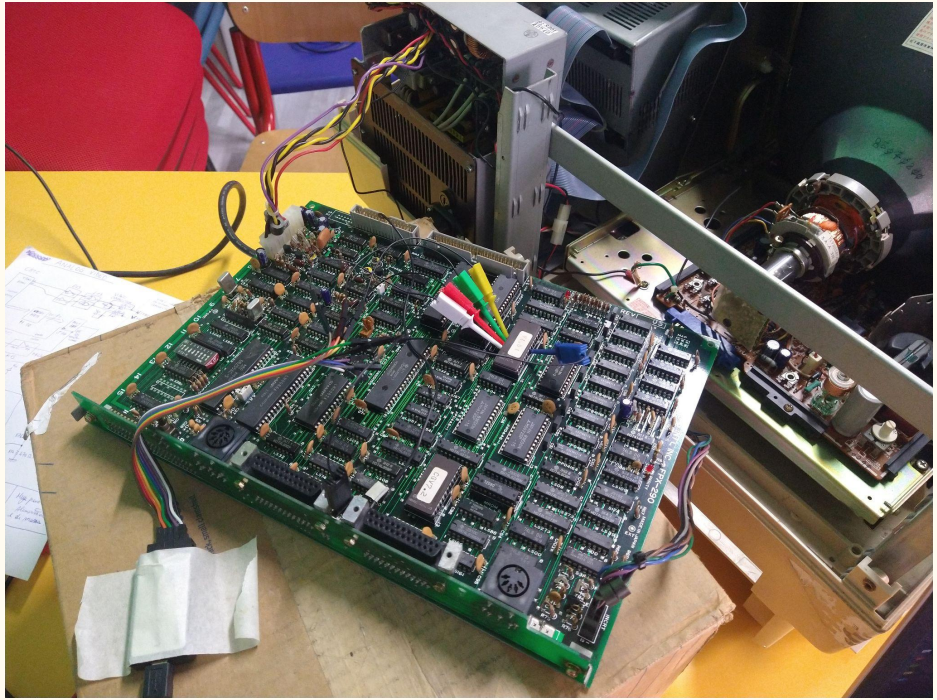
```
jp      $c030                           ;[c000]

ld      a,$89                           ;[c030]
out     ($83),a                         ;[c032]

di                                      ;[c034]

ld      a,$10                           ;[c035]
out     ($81),a                         ;[c037]
```



0000	c032
0001	0002
0002	c033
c030	8983
0001	c034
c031	....

https://z88dk.org/
https://sigrok.org/wiki/PulseView
https://github.com/GLGPrograms/ceda-rom-disassembly

Bus inspection setup



.text and .data not found

```
    ld      hl,$c134                        ;[c108]

    ; loop around $c134 table, writing to $b1 (channel A)
label_c10b:
    ld      a,(hl)                          ;[c10b] load index of internal SIO register
    inc     hl                              ;[c10c] fetch next data
    cp      $ff                             ;[c10d] check table end
    jr      z,label_c119                    ;[c10f] if table end, configure channel B
    out     ($b1),a                         ;[c111] index internal SIO register
    ld      a,(hl)                          ;[c113] load desired internal SIO register value
    out     ($b1),a                         ;[c114] write desidered internal SIO register value
    inc     hl                              ;[c116] fetch next data
    jr      label_c10b                      ;[c117] loop

    ; Configuration table for SIO ChA?
sio_chA_cfg_base:
    BYTE $00                                ;[c134] register 0
    BYTE $10                                ;[c135] reset peripheral interrupts
    BYTE $00                                ;[c136]
    BYTE $10                                ;[c137]
    BYTE $04                                ;[c138] register 4
    BYTE $44                                ;[c139] 1 stop bit, /16 clock divider
    BYTE $01                                ;[c13a] register 1
    BYTE $00                                ;[c13b] disable all interrupts
    BYTE $03                                ;[c13c] register 3
    BYTE $c1                                ;[c13d] rx 8 bit word, synch character load inhibit
    BYTE $05                                ;[c13e] register 5
    BYTE $ea                                ;[c13f] enable tx, RTS, DTR, tx 8 bit word
    BYTE $ff                                ;[c140] end of table
```

Add Github CI to automatically check for disassembly errors.

giomba authored and giuliof committed 10 months ago ✓ 1 / 1

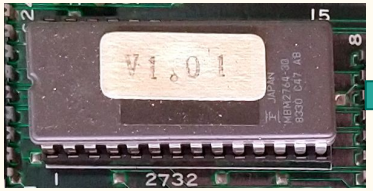
Characters generation

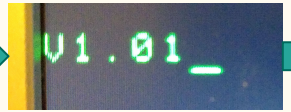
Standard 2732 (4 KiB)
8 pixel x 16 rows

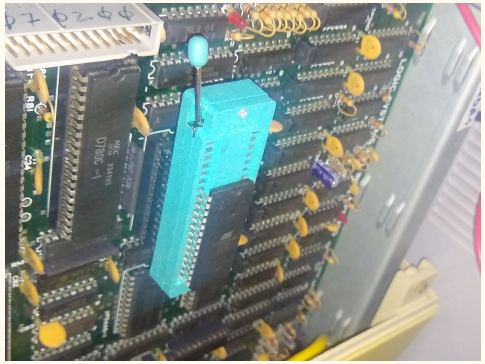
256 bytes ROM

old 28L22
narrow PDIP
dump with custom hardware
glue logic?

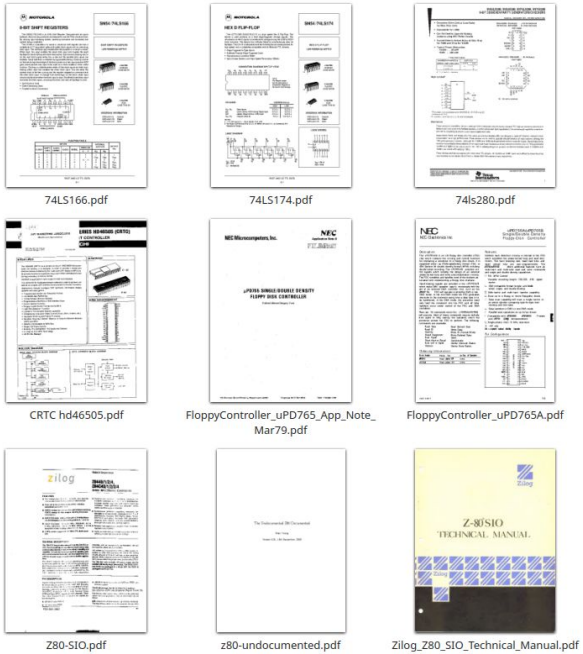
First “hacking” attempt



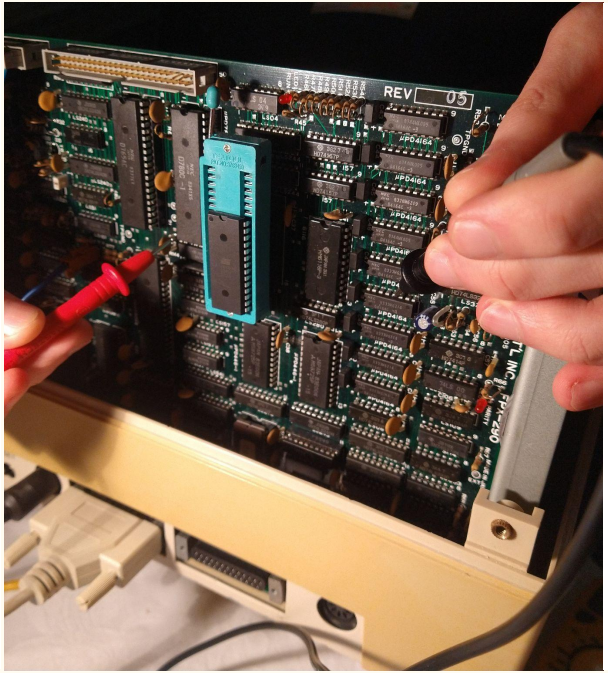




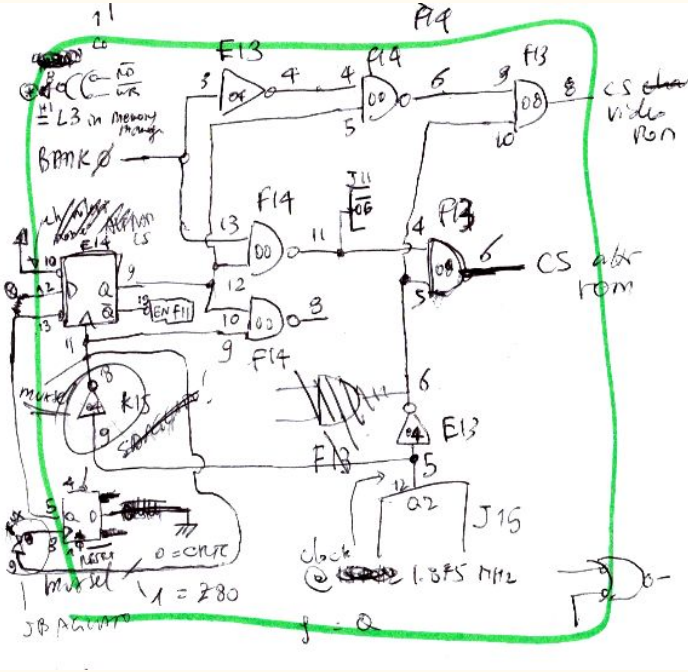
From the board to the schematics



Lots of manuals and datasheets

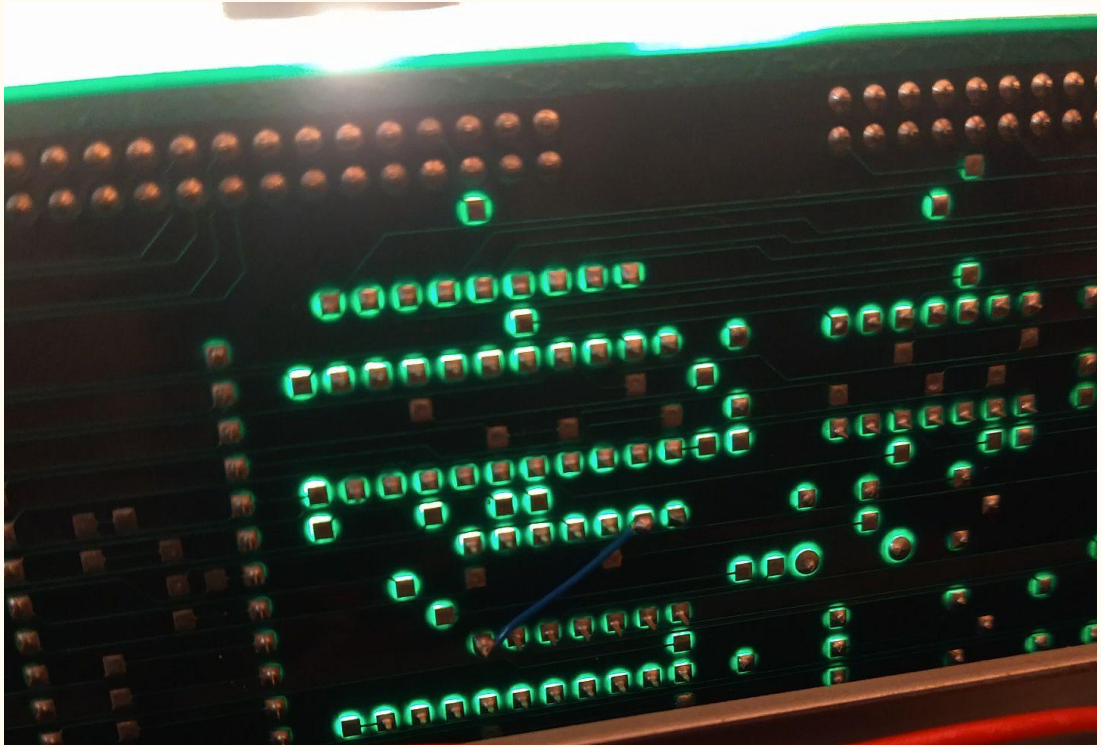


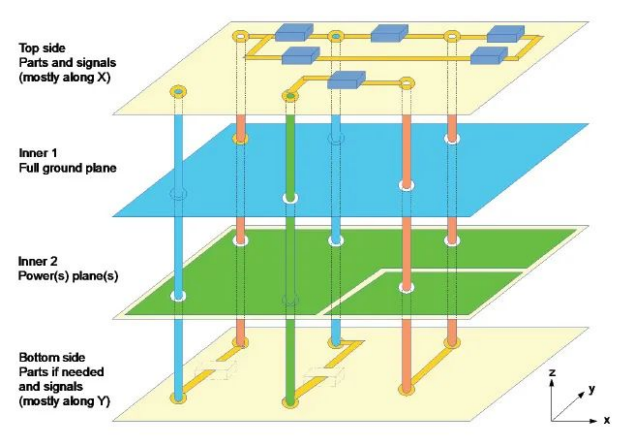
Checking connections



Drafts and connection ideas

X-raying a multi layer board



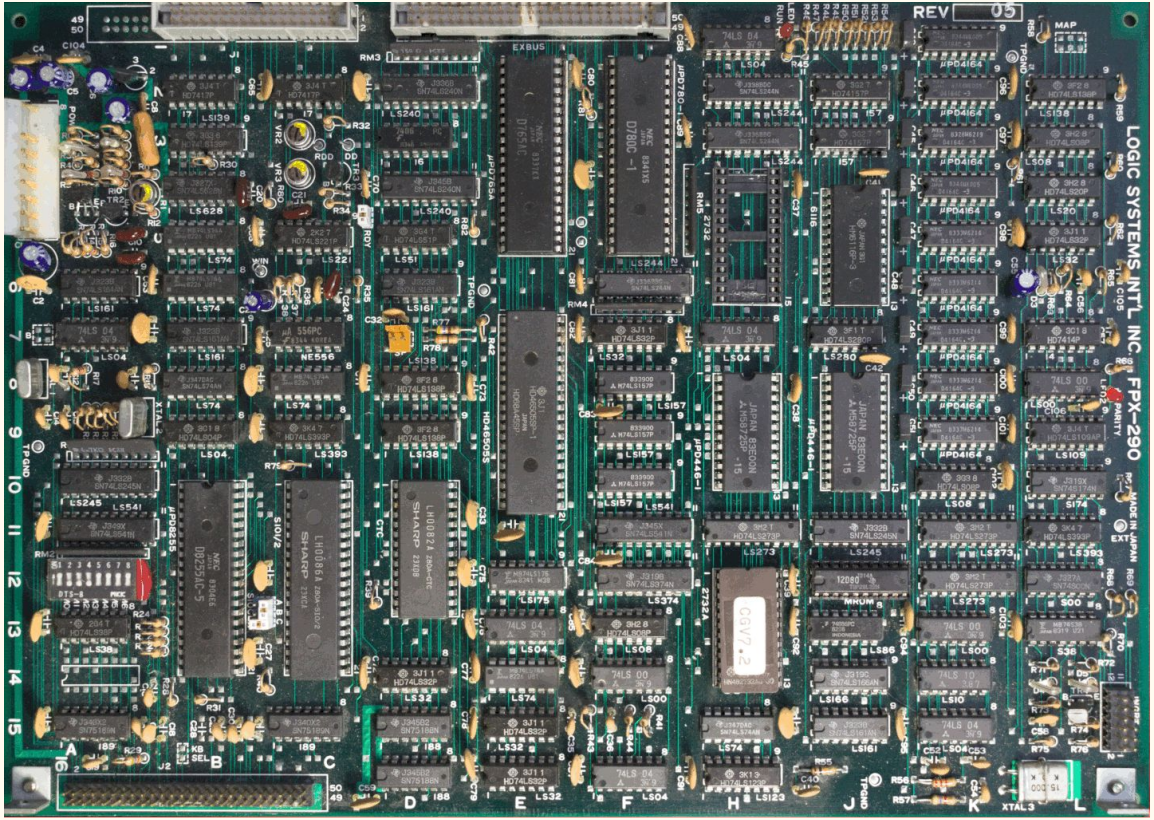


- Four layer board
- Inner layers: maybe supply rails
- Tracks are all on the outer layers...
- ... but some of them hide under ICs!

Follow that wire!

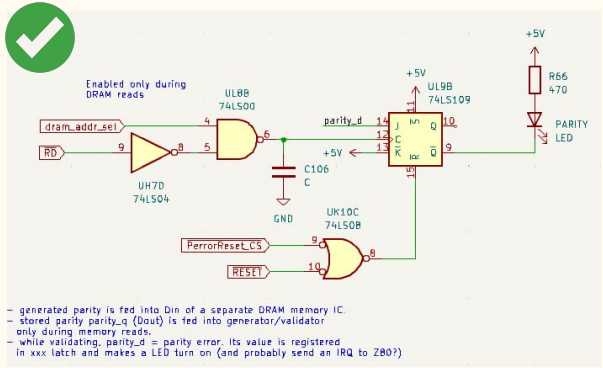
How do we know if a path was already hit?

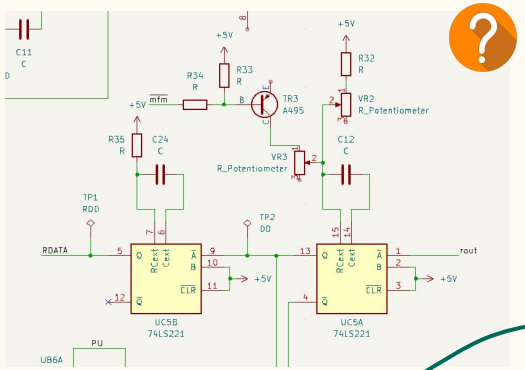
Just take notes on an actual photo using GIMP :)



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!)





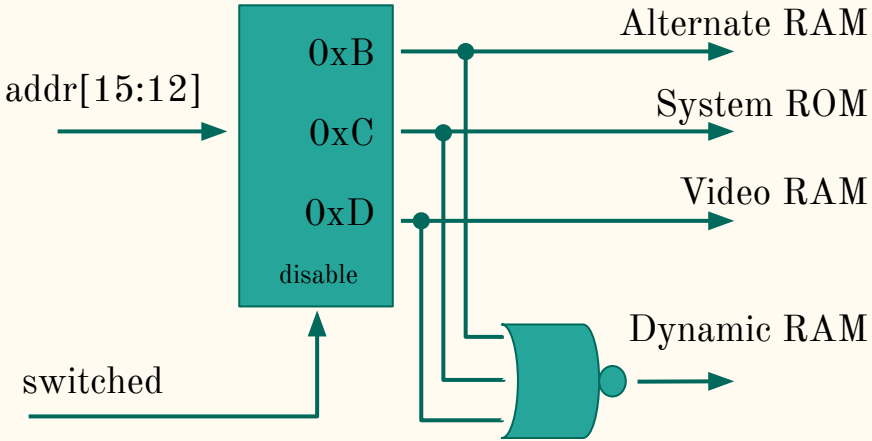
We'll say something about these four in a second

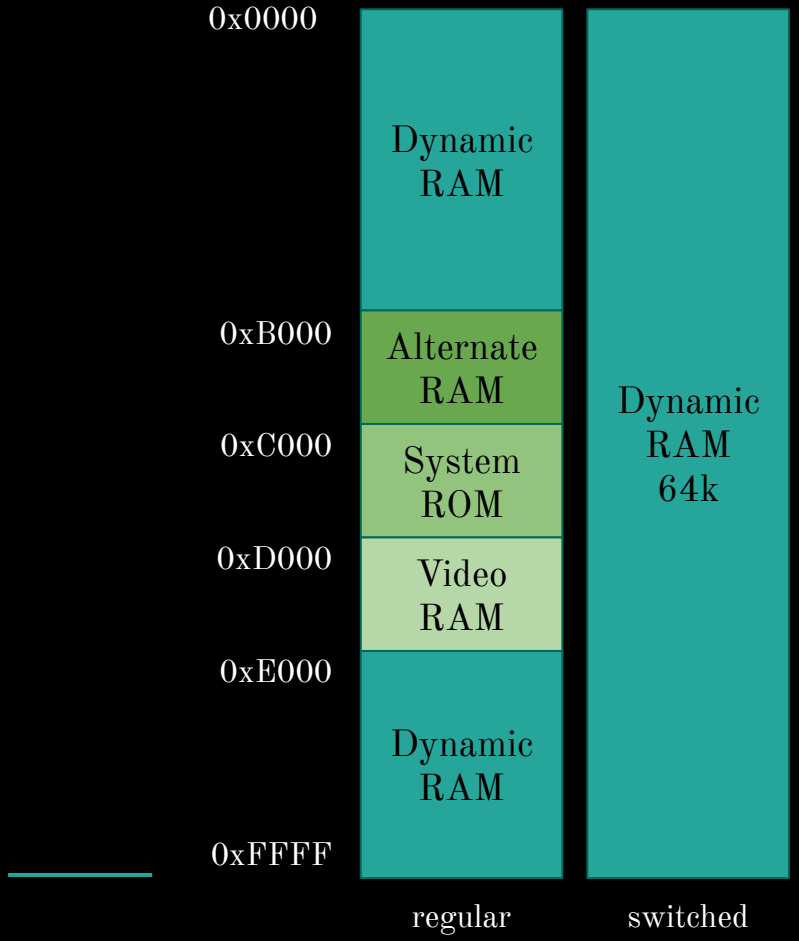
Z80	DRAM and parity	SIO
CTC	Small IO	Video Mem
Ext bus	Mem Mgmt	Reset logic
PIO 8255	CRTC 46505	FDC uPD765

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

Memory map

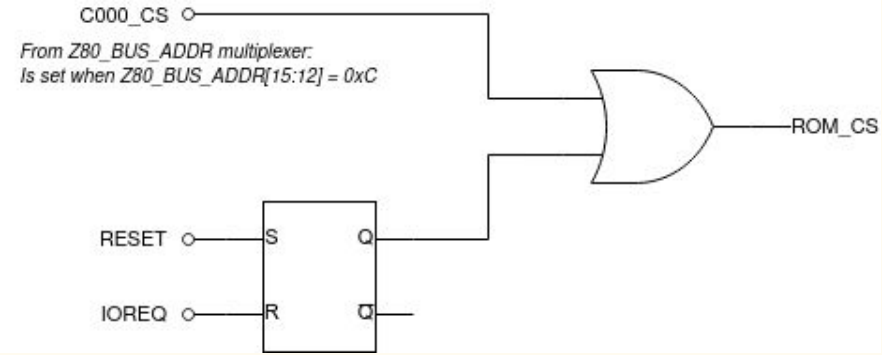
- **64k** DRAM fully addressable with **bank switching** system
- Dedicated **4k video SRAM**
- Further **4k SRAM**

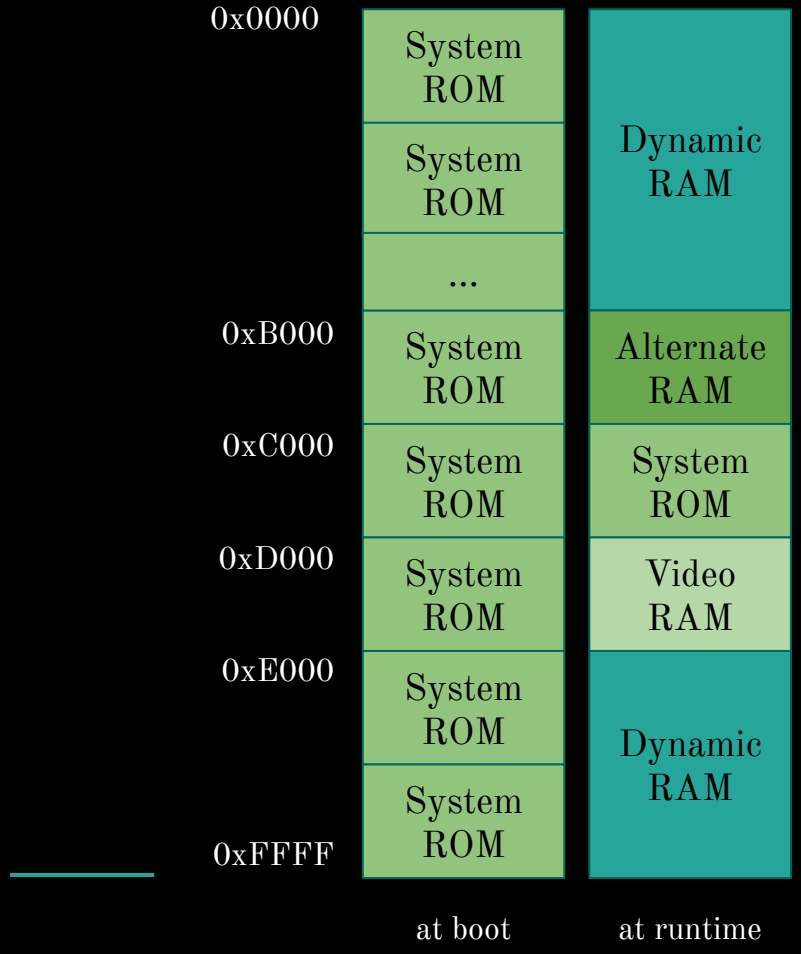




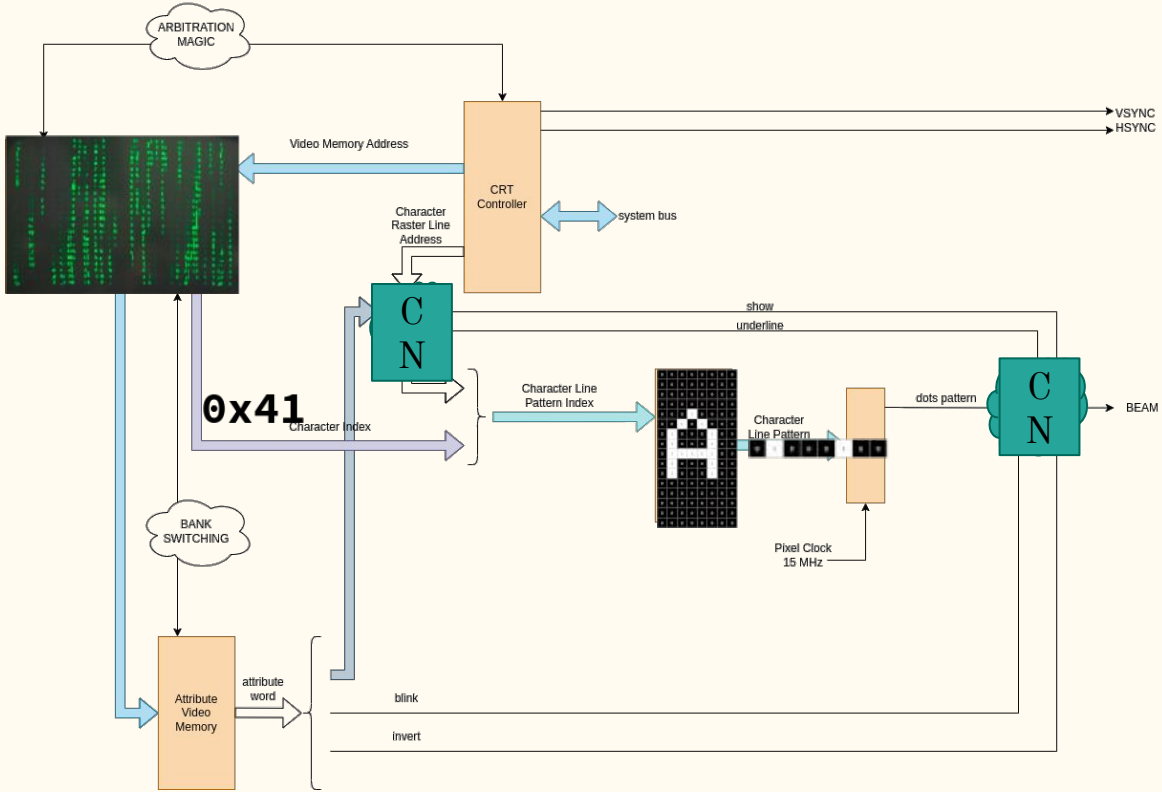
Boot circuit

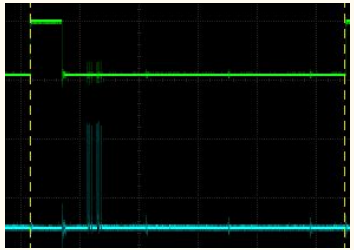
- Z80 boots from **0x0000**
- ROM isn't mapped there...
- A **latch** forces ROM to be mapped everywhere, just at boot!



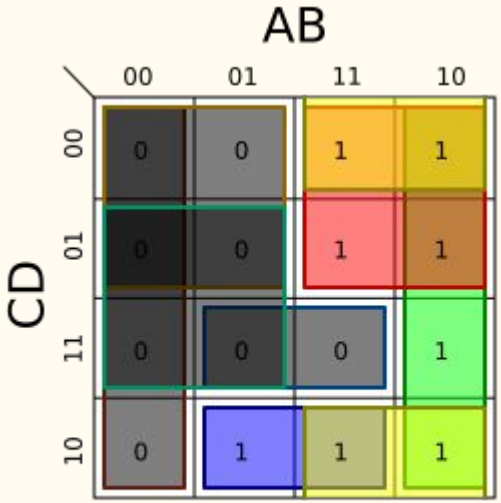


Video Generation





Combinatorial network



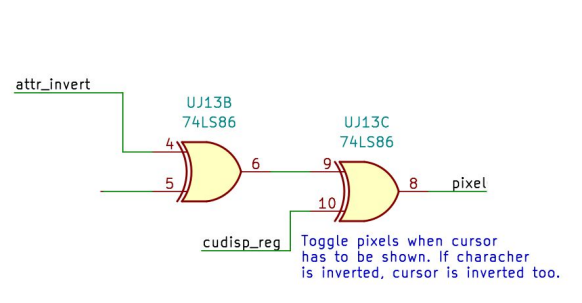
	A7	A6	A5	A4	A3	A2	A1	A0	D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
1	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	1
2	0	0	0	0	0	0	1	0	0	0	1	0	0	0	1	0
3	0	0	0	0	0	0	1	1	0	0	1	0	0	0	1	1
4	0	0	0	0	0	1	0	0	0	0	1	0	0	1	0	0
5	0	0	0	0	0	1	0	1	0	0	1	0	0	1	0	1
6	0	0	0	0	0	1	1	0	0	0	1	0	0	1	1	0
7	0	0	0	0	0	1	1	1	0	0	1	0	0	1	1	1
8	0	0	0	0	1	0	0	0	0	0	1	0	1	0	0	0
9	0	0	0	0	1	0	0	1	0	0	1	0	1	0	0	1
A	0	0	0	0	1	0	1	0	0	0	1	0	1	0	1	0
B	0	0	0	0	1	0	1	1	0	0	1	0	1	0	1	1
C	0	0	0	0	1	1	0	0	0	0	1	0	1	1	0	0
D	0	0	0	0	1	1	0	1	0	0	1	0	1	1	0	1
E	0	0	0	0	1	1	1	0	0	0	1	0	1	1	1	0
F	0	0	0	0	1	1	1	1	0	0	1	0	1	1	1	1
10	0	0	0	1	0	0	0	0	0	0	1	1	0	0	0	0
11	0	0	0	1	0	0	0	1	0	0	1	1	0	0	0	1
12	0	0	0	1	0	0	1	0	0	0	1	1	0	0	1	0
13	0	0	0	1	0	0	1	1	0	0	1	1	0	0	1	1
14	0	0	0	1	0	1	0	0	0	0	1	1	0	1	0	0
15	0	0	0	1	0	1	0	1	0	0	1	1	0	1	0	1
16	0	0	0	1	0	1	1	0	0	0	1	1	0	1	1	0
17	0	0	0	1	0	1	1	1	0	0	1	1	0	1	1	1
18	0	0	0	1	1	0	0	0	0	0	1	1	1	0	0	0
19	0	0	0	1	1	0	0	1	0	0	1	1	1	0	0	1
1A	0	0	0	1	1	0	1	0	0	0	1	1	1	0	1	0
1B	0	0	0	1	1	0	1	1	0	0	1	1	1	0	1	1
1C	0	0	0	1	1	1	0	0	0	0	1	1	1	1	0	0
1D	0	0	0	1	1	1	0	1	0	0	1	1	1	1	0	1
1E	0	0	0	1	1	1	1	0	0	0	1	1	1	1	1	0
1F	0	0	0	1	1	1	1	1	0	0	1	1	1	1	1	1

this is it now, feel old yet?

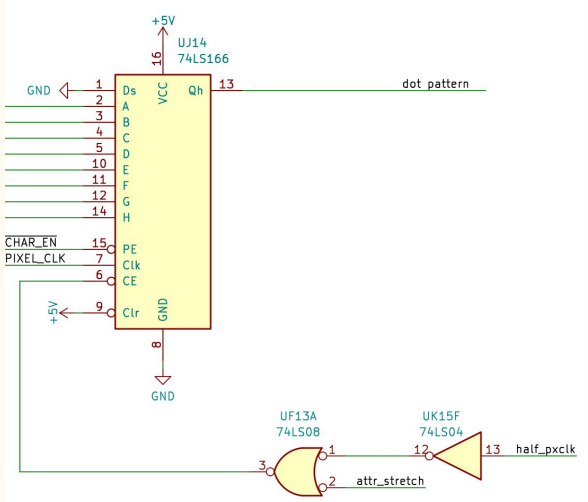
256 bytes ROM

We could have surprised you with special effects...

invert



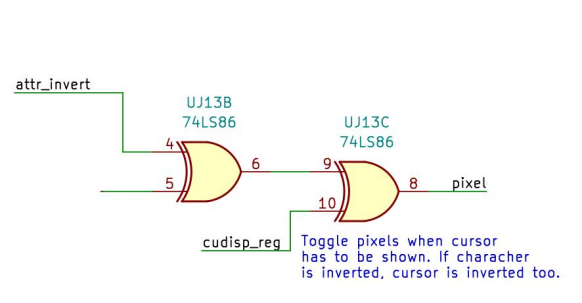
horizontal stretch



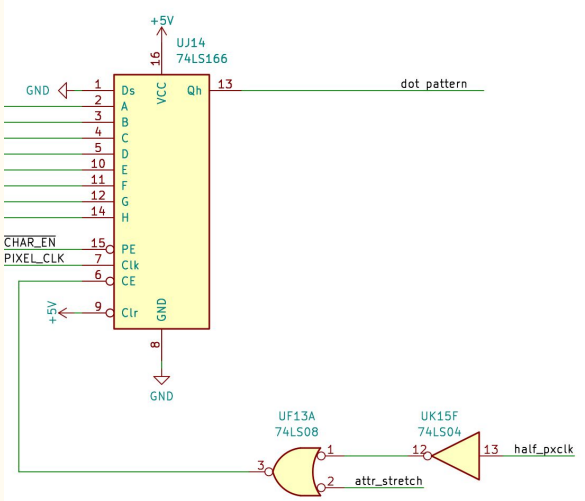
vertical stretch

We could have surprised you with special effects...

invert



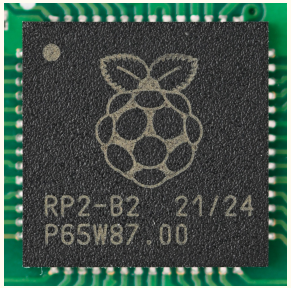
horizontal stretch



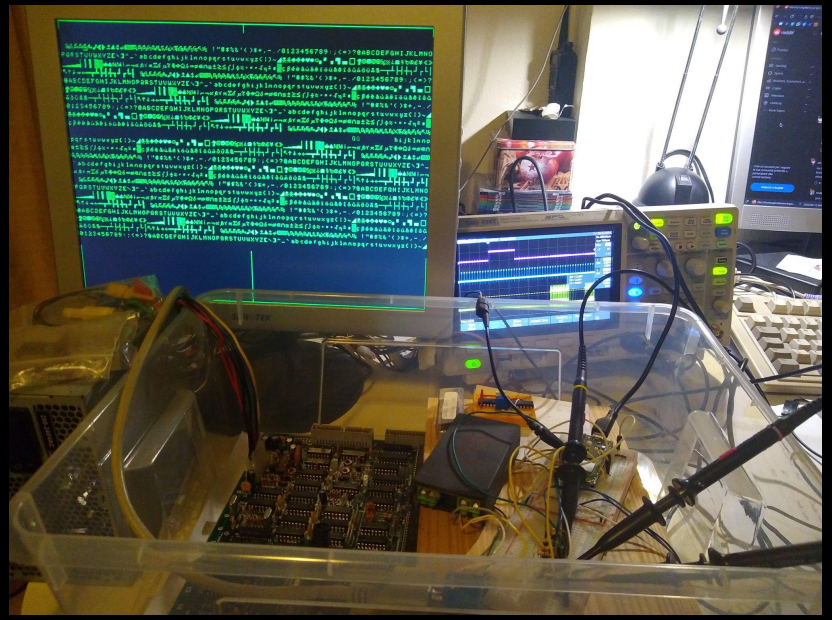
vertical stretch

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)*

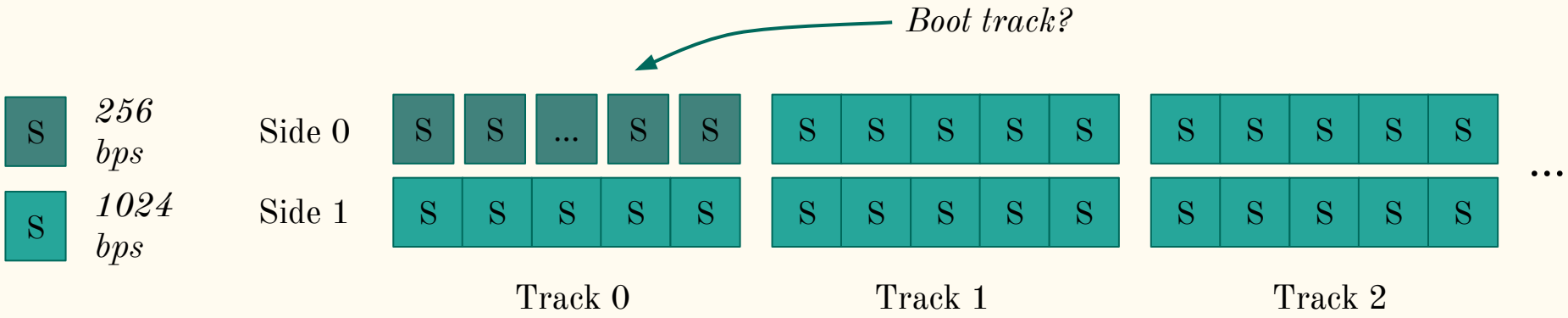
```
   ; Boot trampoline executed when BOOT key is pressed
bios_bootkey:
  ld     de,$0000                  ; track = 0; sector = 0
  ld     bc,$4000                  ; cmd = read ($40); drive = 0
  ld     hl,$0080                  ; load in $0080
  ld     a,$01                     ;
  call   fdc_rwfs                  ; 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
  ; if disk has been correctly copied into RAM, execute it
bios_bootdisk:
  ld     a,$06
  out    ($b2),a
  out    ($da),a                   ; sound speaker beep
  jp     $0080                     ; run loader
```

https://archive.org/details/sanco-8003-cpm-2.2fr.dsqd
https://github.com/GLGPrograms/ceda-cpm

Sanco CP/M floppy format

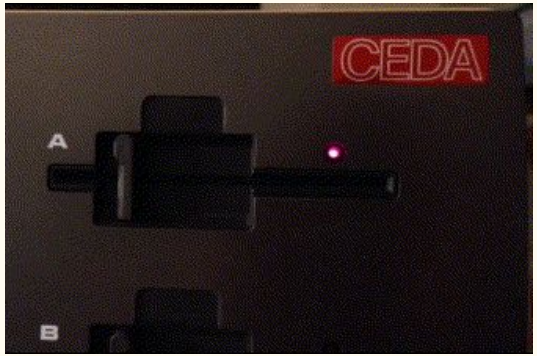
- ROM loads the “loader” from the boot track
- the “loader” loads the CP/M BIOS, BDOS and the CCP in RAM
- control is given to the CP/M, which finishes loading

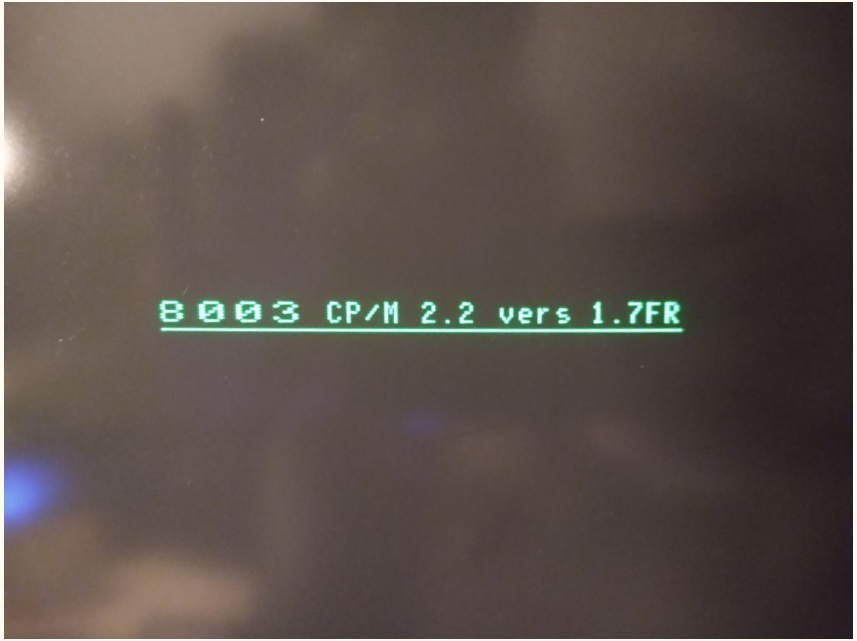
tracks	80
double side	2
sectors per track	5
bytes per sector	1024
	800k



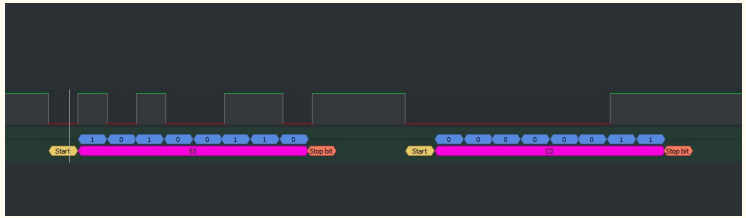
Floppy-o-burner do it yourself

- 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



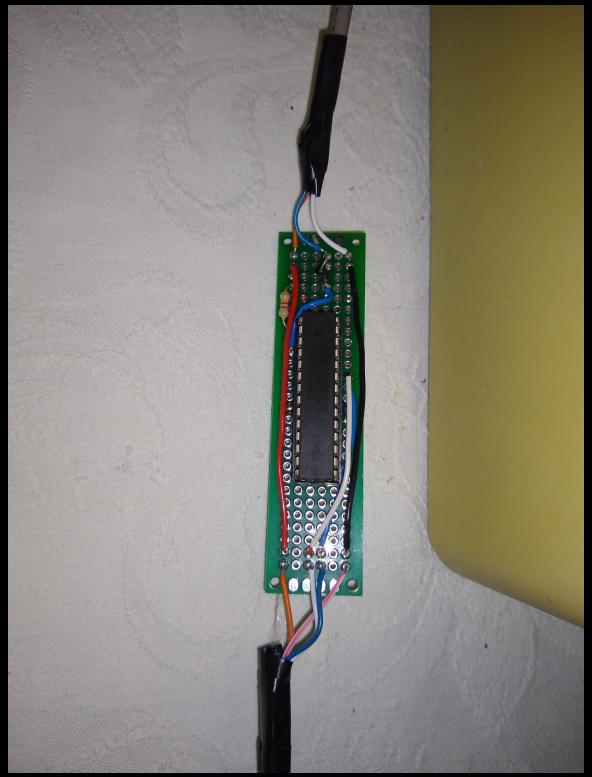


Keyboard

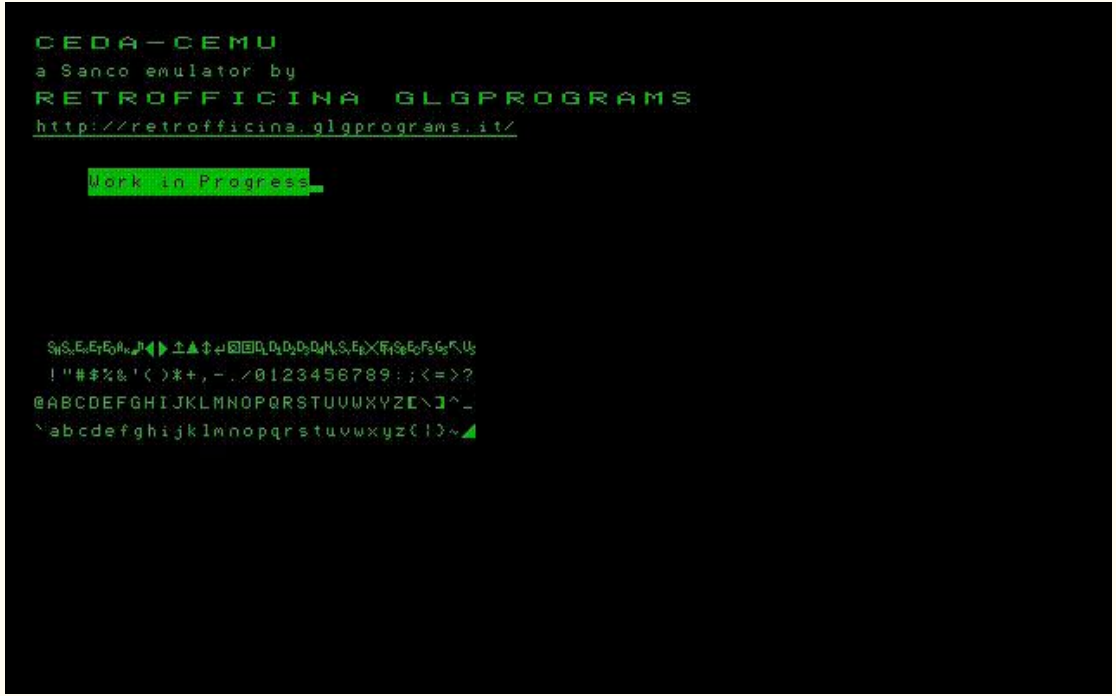


1200 8n1

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



Emulator



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

Q&A