

# The auxiliary device under CP/M emulation

[Back](#)

## The auxiliary device under CP/M emulation:

The auxiliary device is implemented only for historical reasons. When I had implemented enough of the Z80 simulation in 1988 to run some CP/M software, I had no UNIX tools to access disk images. I needed some way to get the CP/M files and programs onto the disk images. Because I still was familiar with paper tape punchers and readers and how to use them with CP/M's pip command, the idea was born to emulate this with named pipes under UNIX.

All output to the auxiliary device is written into the named pipe **~/z80pack-x.y/cpmsim/auxout**. The CP/M emulation forks the program **~/z80pack-x.y/cpmsim/receive**, which reads everything from the named pipe and writes the data into file **~/z80pack-x.y/cpmsim/auxiliary.cpm**.

Input from the auxiliary device is read from the named pipe **~/z80pack-x.y/cpmsim/auxin**. So before starting the CP/M emulation and using programs to read from the device, some sender process needs to be started first. This is what the program **~/z80pack-x.y/cpmsim/send** was used for, it writes data from the file given as argument into the named pipe.

One problem with the auxiliary device input and output was that pip used the CP/M EOF (CTRL-Z) to stop reading from the device, so only text files and no binaries could be exchanged. Binary files first had to be converted to Intel hex so that they could be transferred with pip, and under CP/M converted back to binaries with the load command.

And that is how I got my first CP/M files onto the empty disk images:

```
bin2hex -o 256 ddt.com ddt.hex
send ddt.hex &
cpmsim
pip ddt.hex=rdr:
load ddt
```

Of course there is a catch22 situation here, how did I get the pip program onto the very first disk image? That is what the cpmsim -x option was used for, to load the emulators memory from a file. Because CP/M's save command is builtin it could be used to save the memory image to disk then.

Nowadays all this isn't necessary anymore, there are tools available to read and write CP/M disk images on UNIX systems. Whatever, if the auxiliary device is needed for some reason it is available under CP/M 1, CP/M 2 and CP/M 3, MP/M doesn't support the device.

| CP/M version | Sending           | Receiving         |
|--------------|-------------------|-------------------|
| CP/M 1 & 2   | pip pun:=file.txt | pip file.txt=rdr: |
| CP/M 3       | pip aux:=file.txt | pip file.txt=aux: |

[Back](#)