Hacking GeckOS

Glenn Holmer
World of Commodore
2019-12-07

Speaker Bio

- ✔ a.k.a. "Cenbe"
- ✔ retired Java programmer/Linux sysadmin
- ✔ collector of programming languages and operating systems for the Commodore 64:

  `https://www.lyonlabs.org/commodore/`

Happy 50th, Unix!

wait... “Unix” on a 6502?

ARE YOU CRAZY?

wait... “Unix” on a 6502?

Let’s compare a Commodore 64 to the machines that early versions of Unix ran on:

	PDP-7	PDP-11	Commodore 64
memory:	16K	24K	64K

Processor speeds were comparable to that of a ‘64, but the architecture was very different.

Of course, there are the registers...

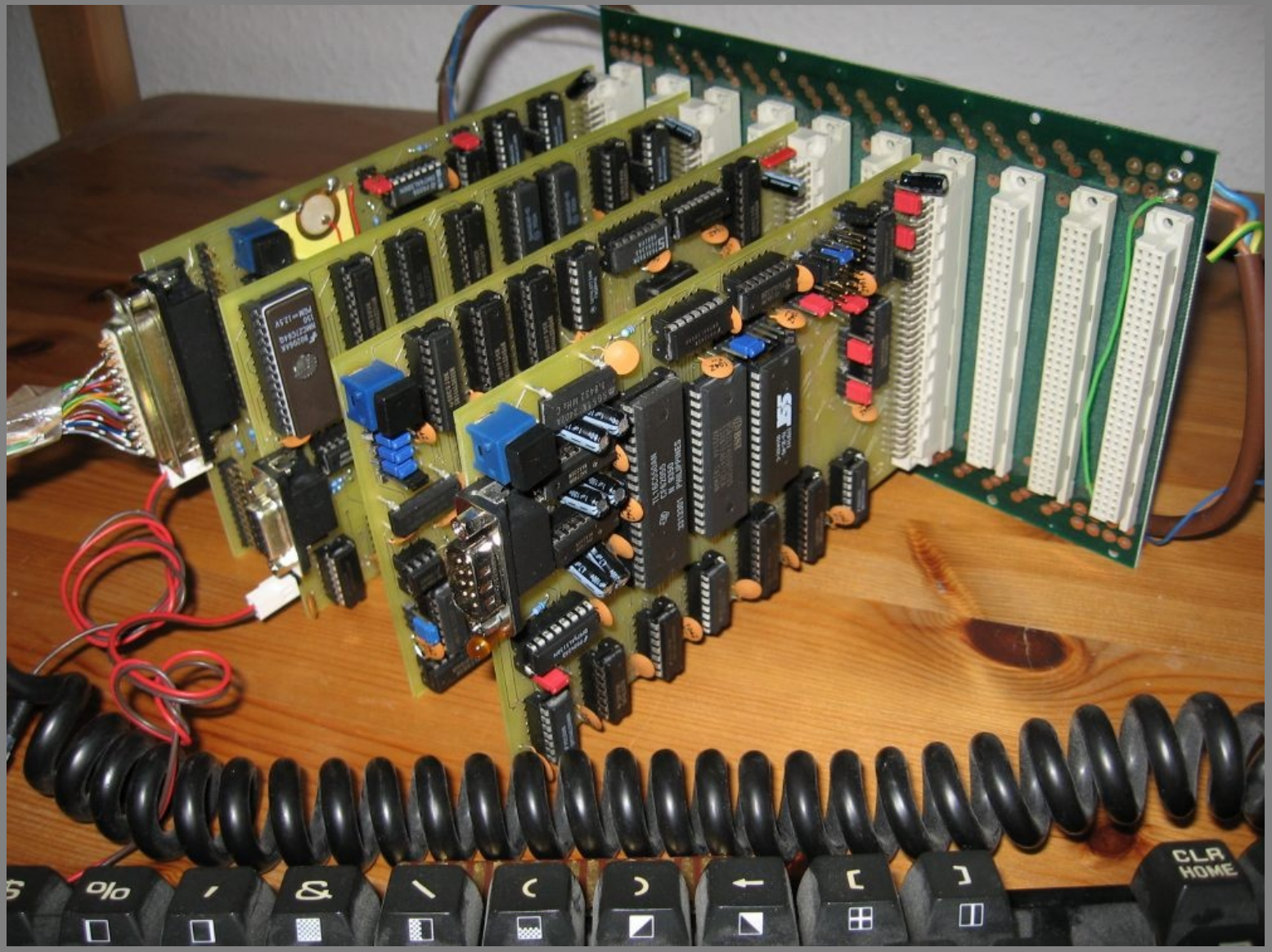
GeckOS history

- ✔ Written by André Fachat, originally for a 6502 computer with an MMU that he built in 1989.
- ✔ He later ported it to run on the PET, 8296, and C64.
- ✔ Development is *very* active!
- ✔ Source is GPL, on GitHub (fachat/GeckOS-V2).

André Fachat

CS/A65

http://www.6502.org/users/andre/csa/index.html



GeckOS features

GeckOS is a Unix-like multitasking operating system for the 6502 CPU. It supports:

- ✔ task priorities, multi-threading
- ✔ virtual consoles
- ✔ signals, semaphores
- ✔ backgrounding and redirection
- ✔ piping
- ✔ environmental variables
- ✔ relocatable file format

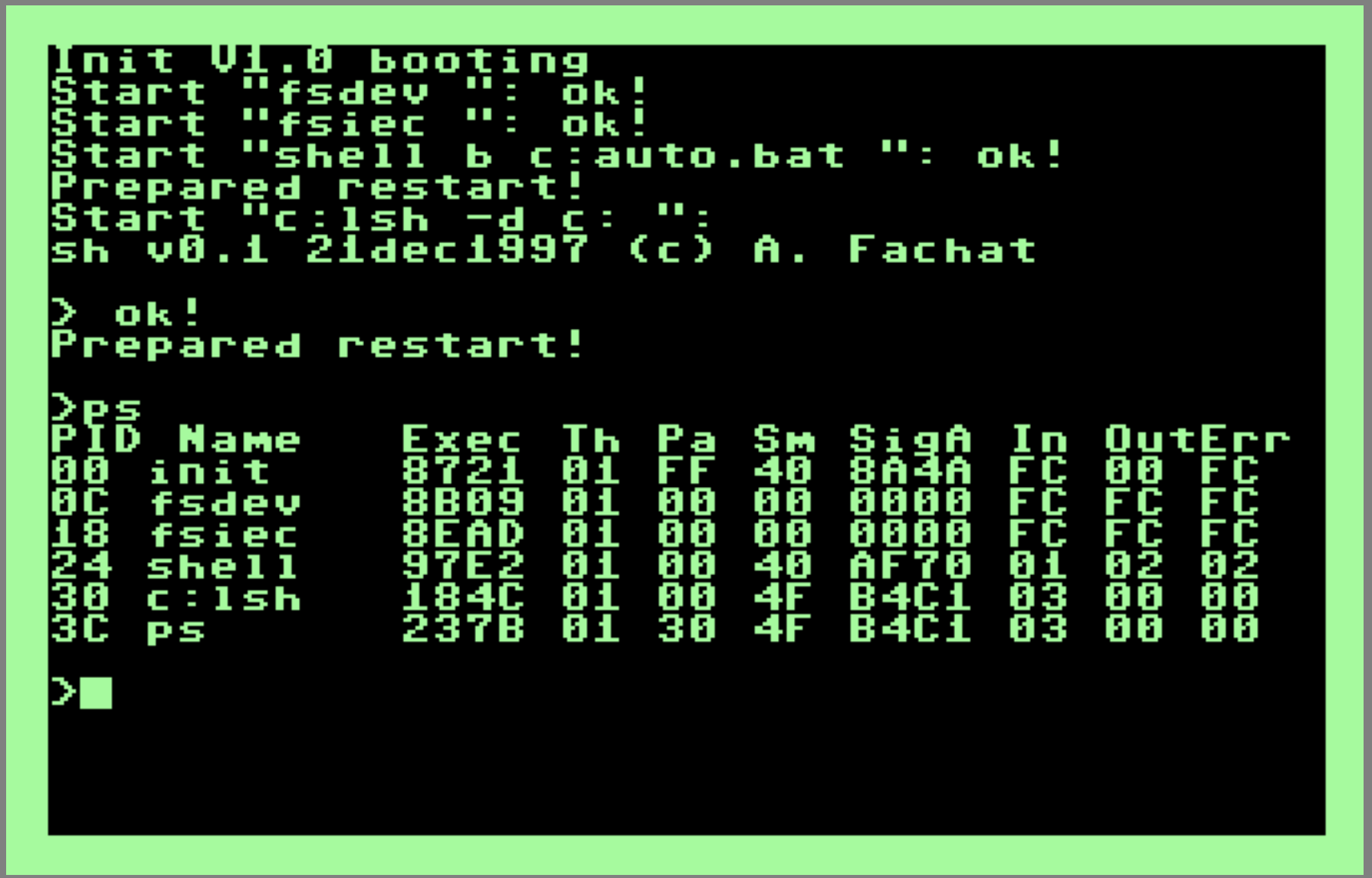
DEMO

- ✔ shell (both), monitor
- ✔ forking (one program loads and runs another)
- ✔ backgrounding a program ("the Schema demo")
- ✔ signals (sending messages between programs)
- ✔ semaphores (blocking on available resource)
- ✔ new `ps` and `kill` commands

This is a very recent build of GeckOS.

original info command (old shell)

new ps command (lsh shell)



`GETINFO` and the task table

- `info` (the old shell's "`ps`") calls the kernel `GETINFO` API, which reads the task table and returns information about all processes.
- It uses the program communication buffer (`PCBUF`, a.k.a. `SYSBUF`) to build a table (since programs should not have direct access to the task table).
- The task table did not originally have entries for either process name or exec address, although the `GETINFO` table has one for name.

adding process names (stdlib programs)

- For lib6502 programs, the name can be found in the LIBSAVE structure which is populated when a program is started. This structure is pointed to from the task table.

  PROBLEM: the kernel shouldn't assume that programs are written using the standard library, and shouldn't access the LIBSAVE structure, as it is lib6502-specific.

adding process names (kernel programs)

- For init and the device drivers, it's possible to get the name by walking the ROM image headers in the same order that kernel startup does.

  PROBLEM: this breaks if kernel initialization changes... it's also a filthy kludge!

Let the kernel do it!

SOLUTION: the kernel `FORK` routine takes process name and exec address as parameters; it should just save them in the task table.

PROBLEM: lib6502 programs pass the program name with a stream number in the first byte.

SOLUTION: change lib6502 to pass the stream number as a parameter to `FORK` (kernel passes this byte back in .A when the process starts).

adding exec address (kernel programs)

- The program headers in the kernel image contain the exec address, so it's an easy matter for the kernel to put it in the task table when starting one of these programs.

adding exec address (stdlib programs)

- The exec address is passed to FORK, and could be stored just before it passes control to the program.

  PROBLEM: lib6502 programs set a start address of lib6502's `libfork` routine (which loads and relocates the program).

  SOLUTION: provide a `SETINFO` API that would allow lib6502 to update the task table after `FORK` has been called.

A New Golden Age for GeckOS

Now that we can debug more easily, anything is possible:

- ✔ The Grand Unification of the Shells
- ✔ better support for CMD HD, *µ*IEC, 1541 Ultimate (partitions, subdirectories, disk images...)
- ✔ 1541 Ultimate networking
- ✔ native speeder in the filesystem?
- ✔ *your project here*

resources

- GeckOS (source, tools, docs, disk images):
  `http://www.6502.org/users/andre/osa/index.html`
- GeckOS source on GitHub:
  `fachat/GeckOS-V2`
- online HTML documentation:
  `https://www.lyonlabs.org/commodore/onrequest/GeckOS-docs/index.html`
- Cenbe's Commentary on GeckOS:
  `https://www.lyonlabs.org/commodore/onrequest/geckos-analysis.html`

QUESTIONS