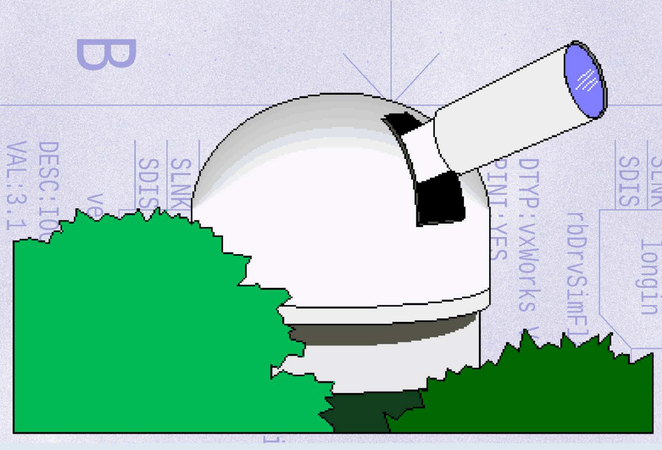
Canada France Hawai'i Telescope



The TCS Challenge:

Much of the CFH Telescope Control System is difficult to maintain and secure:

- Hardware for computing (runs pointing model and control) is obsolete
- Software is also obsolete or running very old versions

Obsolescence Management

Rewrite? Port? or *Emulate* (1990s Telescope Control System)

Sidik Isani, Chris Usher, Conrad Holmberg

The Options

A **REWRITE** of the entire TCS with modern tools, languages, and operating systems would clearly deliver the best final outcome. However, given that we know we would still end up with basically the same telescope performance (efficiency in overheads, potential pointing accuracy) and the fact that we lack expertise in EPICS and would need a lot of time to pull this off, we considered other solutions.

An effort to **PORT** tens of thousands of lines of code from VxWorks to Linux (and from SPARC architecture to AMD64) began a couple of years ago. This has effectively been abandoned for the same reasons that a rewrite is not very practical. Both of these options would also take up considerable time for on-sky testing, and it's difficult to see how we'd recoup the time.

If we **EMULATE** the existing hardware, we don't immediately solve all of our problems. We have to consider security problems with continuing to run old versions of software and the VxWorks operating system. However, it would eliminate the most serious risk, or SPARC32 hardware failure (CPU boards, backplane, power supplies) and running out of readily available spares.

Past →



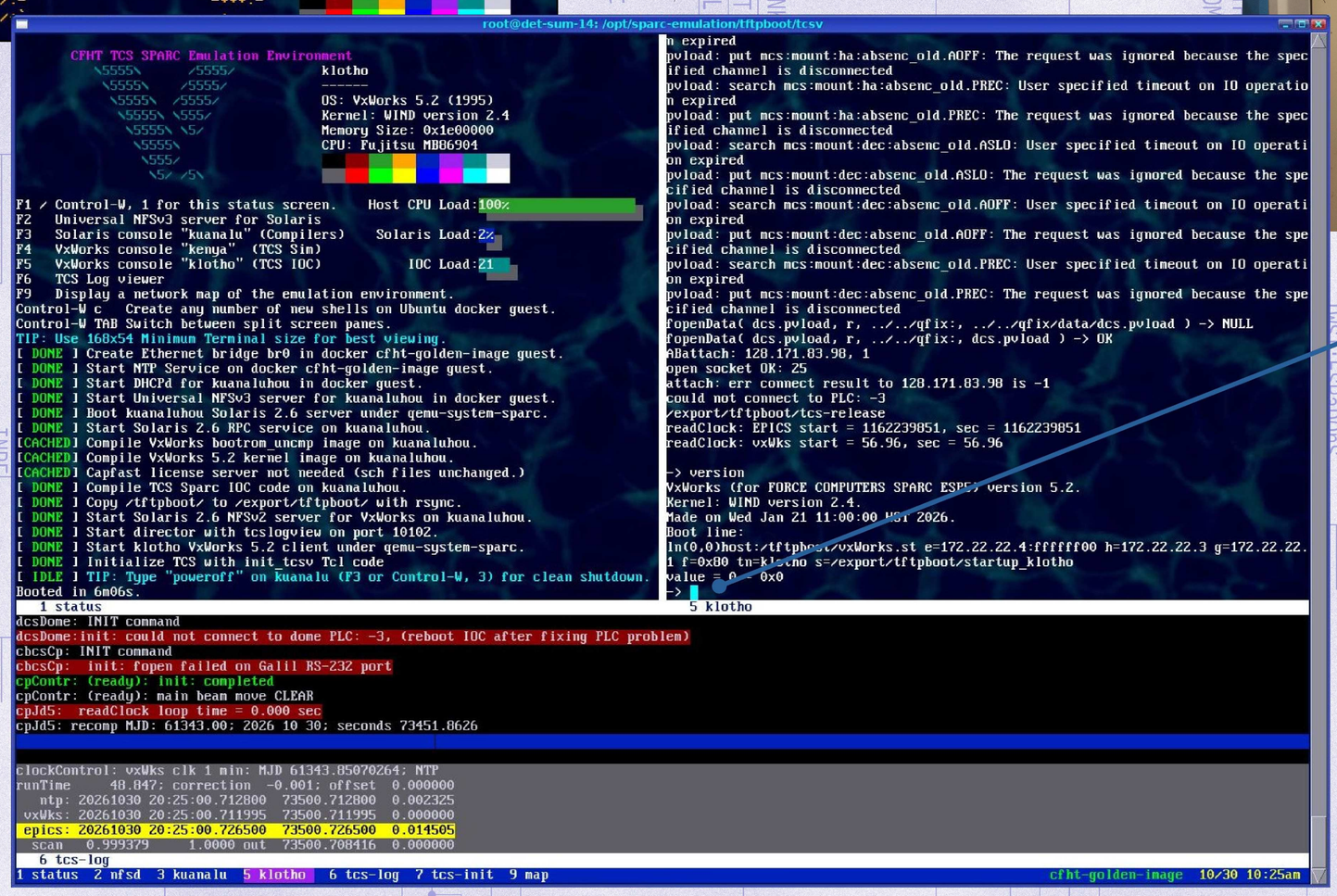
Present →

GNU screen debug TUI

Neofetch-style window dressing showing the operator that we are running two different SPARC-based operating systems both from the 1990s (VxWorks 5, & Sun Solaris 2.6).



Consoles of each emulated machine appear here. User can switch between Solaris login prompt, embedded NFS servers, CFHT "golden image" Ubuntu Linux including EPICS utilities, network Diagrams, and VxWorks serial console prompt.



Help guide (Function keys)

Linux init-style red/green startup status. A Dockerfile builds the emulator itself, including patched version of QEMU. The rest of the startup, including compilation of SPARC kernel and EPICS application code runs in pipeline each time system boots. Caching means this is normally under 3m.

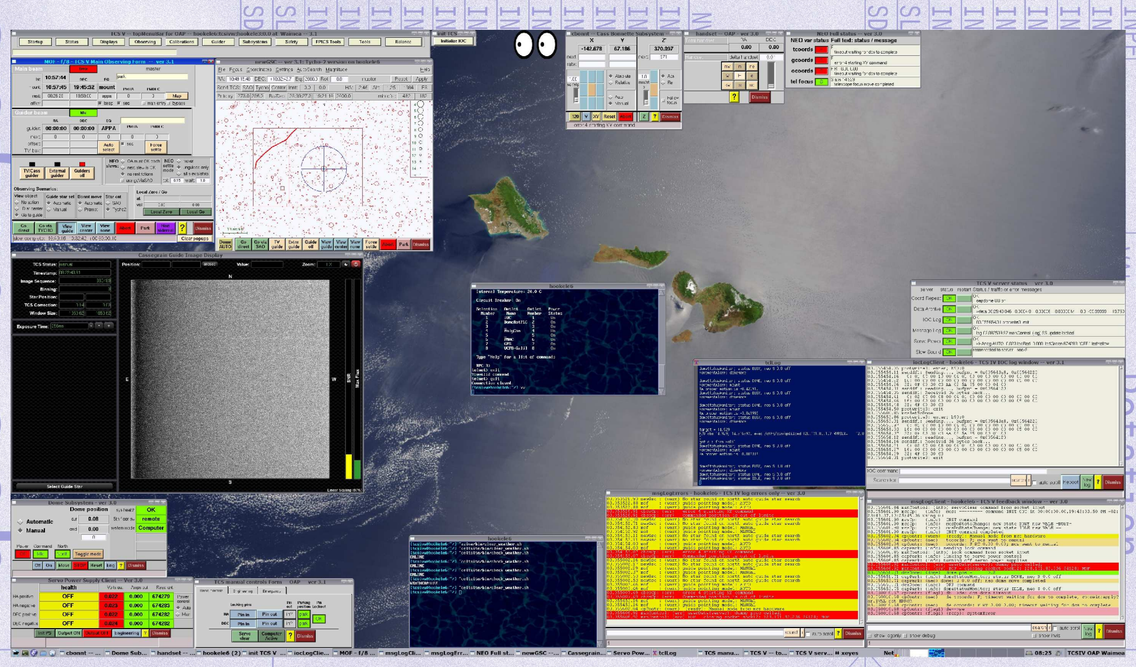
Logging and feedback

Everything which needs a SPARC CPU is now running on virtual hardware inside this environment, viewable by connecting to a running GNU screen session. This includes VxWorks 5 as it was on the FORCE CPU board, shown above. The FORCE board was virtually identical to a SPARCStation 5, which QEMU can emulate quite well. Before we ran our TCS this way, only sparc-linux and Sun Solaris / SunOS had been running successfully under QEMU. We were able to make modifications to QEMU to allow a VxWorks 5 kernel and our application software to run smoothly.

This is a VxWorks 5 console prompt, running under emulation on a normal AMD64 PC.

← Future

VNC accessible X11 Tcl/Tk GUI



Once emulation of the SPARC code is proven to be stable, the graphical user interface of the 30+ year old TCS will be reconsidered.

CFHT's graphical TCS control forms, guide viewers, and star catalog interfaces use Tcl/Tk but also communicate directly with EPICS through channel access. Unfortunately, these interfaces are not just a light layer atop a CLI. They contain much of the logic necessary to operate the telescope. At least these would not need to be **ported** as they are already running on Linux. But we have the opportunity to **rewrite** them, make them more modern, and perhaps be compatible with Wayland on modern Linux desktops, or even make them Web browser accessible. Or we could take the same approach and freeze working versions of Tcl/Tk and an X-server inside the same **virtual environment** that emulates the SPARCs.