# The Flux Operating System Toolkit (OSKIT)

Mike Hibler
and
The Flux Folks

# OSKIT: Overview

- A set of carefully modularized libraries with extensive documentation

- Provides low-level "grunge" needed in every O.S. (e.g., device drivers)

- Can be used either directly as libraries or cannibalized for "spare parts"

# OSKIT: Goals

- Make it easier for O.S. developers to
  - Create a new O.S.
  - Port an existing O.S.
  - Enhance an O.S. to support more devices, filesystem formats, or executable formats.
- Free the O.S. developer to concentrate on “interesting” issues

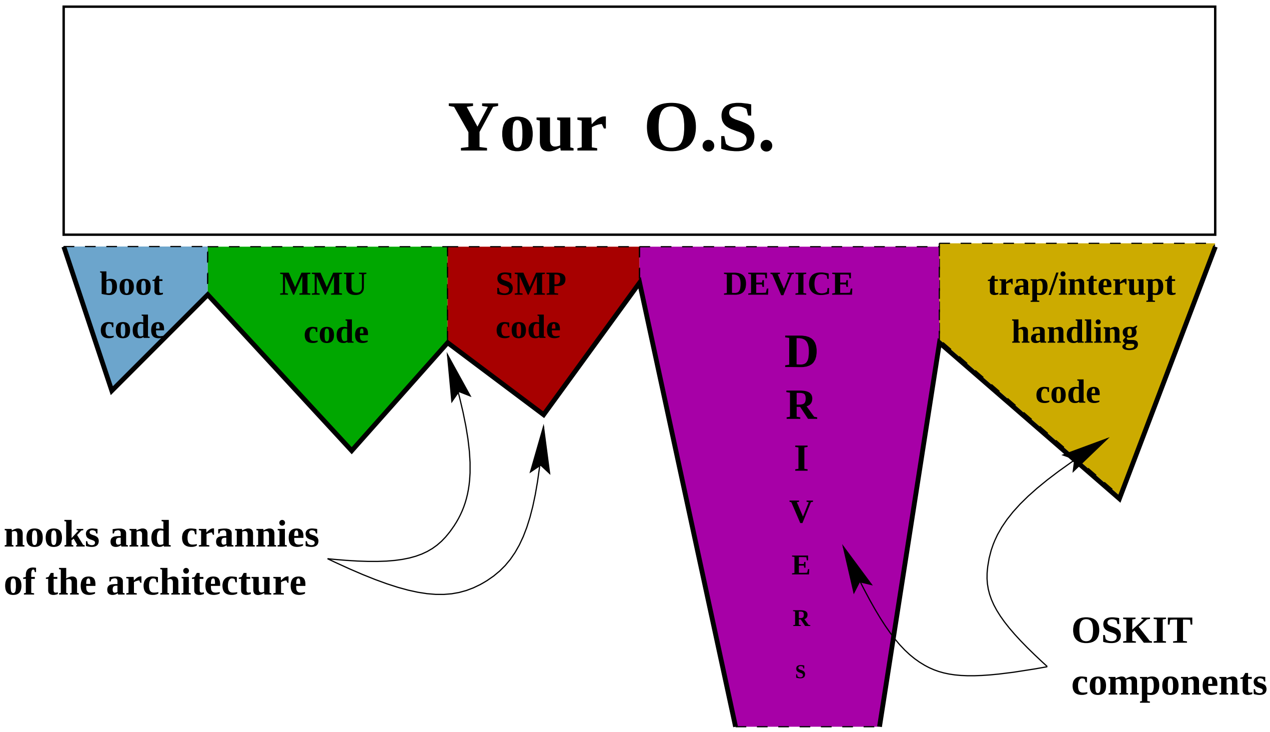
# OSKIT: Components

- `liblmm`: flexible memory management
- `libamm`: address map management
- `libexec`: executable loader for various object formats (supports ELF, a.out)
- `libdiskpart`: support for reading disk partition maps (for DOS, BSD, Linux)
- `libfsread`: read-only filesystem interpretation (for FFS, ext2fs, MINUX)
- `libmc`: minimal C library with few external dependencies (includes string, memory, stdio functions)

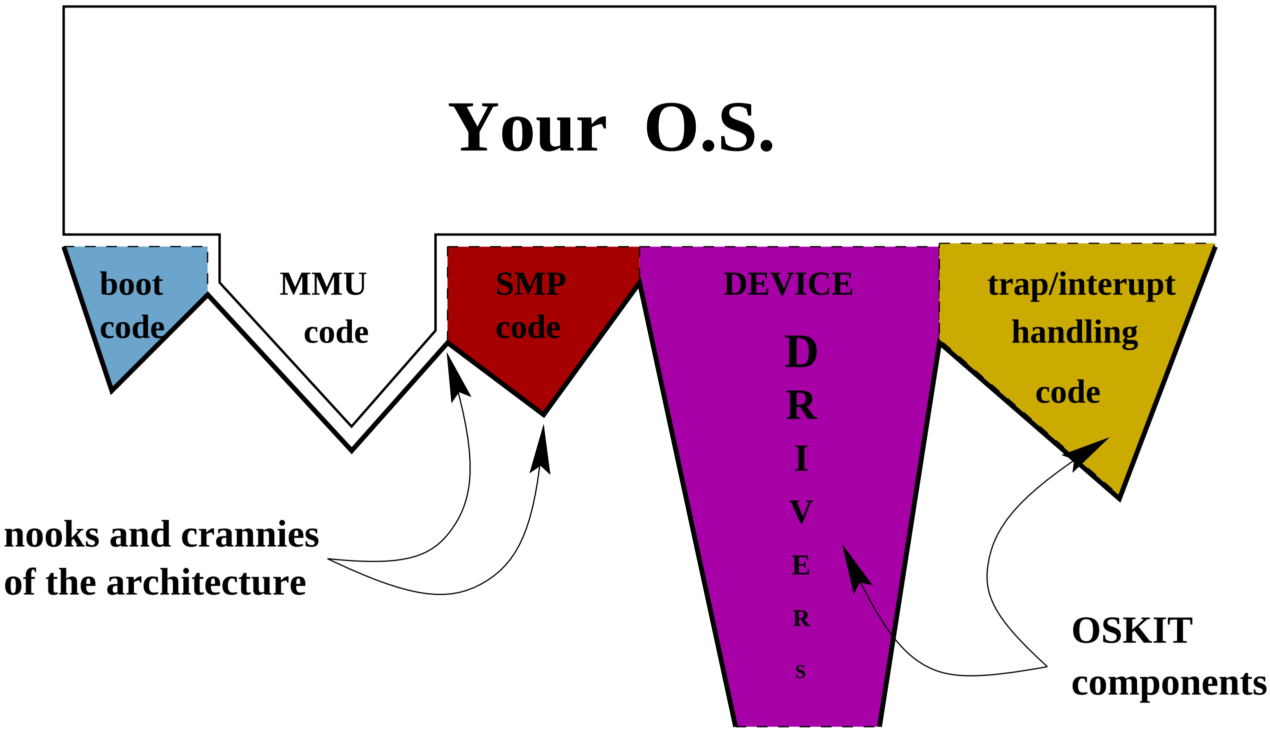
# OSKIT: Components (cont.)

- `libkern`: kernel support
  - Multiboot bootstrap support
  - Page table setup
  - trap/interrupt handling
  - processor mode switching
  - remote debugging support
- `libsmp`: symmetric multiprocessor support
- `libfdev*`: device driver framework (allows use of *unchanged* Linux, FreeBSD device drivers)
- *Extensive* documentation (300+ pages)

# OSKIT: Example 1



# OSKIT: Example 2



# OSKIT: Device driver architecture

- Uses existing widely-available device drivers, unchanged

- Defines standardized interfaces, and implements them via "glue":

  – Around drivers, exported to OS
    E.g. `fdev_blk_read`

  – Around OS routines, exported to drivers
    E.g. `fdev_mem_alloc(..., memtype_flags)`, `fdev_intr_disable`

- Provides default implementations of base OS routines. Mature OS expected to override them.

- Architecture allows user-mode device drivers

# OSKIT: Status

- Support for x86 architecture
- Support for Linux device drivers
- Used extensively in Fluke:
  - 75% of linked kernel comes from OSKIT (with Linux driver set, optimized, no debug, 260K total)
  - 20% of linked kernel comes from OSKIT (no drivers, debug enabled, 126K total)
- Used by Olin Shivers at MIT to boot ML on bare hardware as the O.S.
- 55 external people on mailing list
- Three releases to date

# OSKIT: Future

- More device driver sets, including binary (e.g., *BSD, Novell ODI?)
- More architectures (e.g., PA-RISC, PPC, Alpha, ARM?)
- Incorporate feedback from other O.S. researchers (SPIN, Exokernel, Scout, Cache kernel, VINO, OSF)
- Library'ize some existing parts
- Finish documentation
- Continue to add more useful components

# Unsolicited Testimonial

"I'm booting ML directly on the hardware *as* the OS. Your code meant that I could *instantly* get up and going by linking the SML/NJ runtime against OSKit and booting. I have a lot of research directions to chase with the theme of the advanced-language OS, and much further down the line we'll hopefully rip out a lot of what we've used from OSKit. But OSKit gives me an instant platform to start with.

I could have tried to tease apart Linux, and used it, but it would have been a long, drawn-out mess. OSKit was perfect for my needs, especially since it has a lot of support for porting user code (like SML/NJ) to run on a raw machine (e.g., the stdlib and I/O libs). The remote gdb support was also key, and your documentation was very valuable."

–Olin Shivers, MIT LCS, October 7, 1996