ULIX 2013

Hans-Georg Eßer

University of Erlangen-Nuremberg
Chair for IT Security Infrastructures (i1)

Dagstuhl 12.–15.08.2013

ULIX

ULIX . . .

- is a (partially implemented) simple *Unix-like OS*
- shall improve *Operating System courses*
- by being implemented with *Literate Programming*
- and serving as an OS textbook

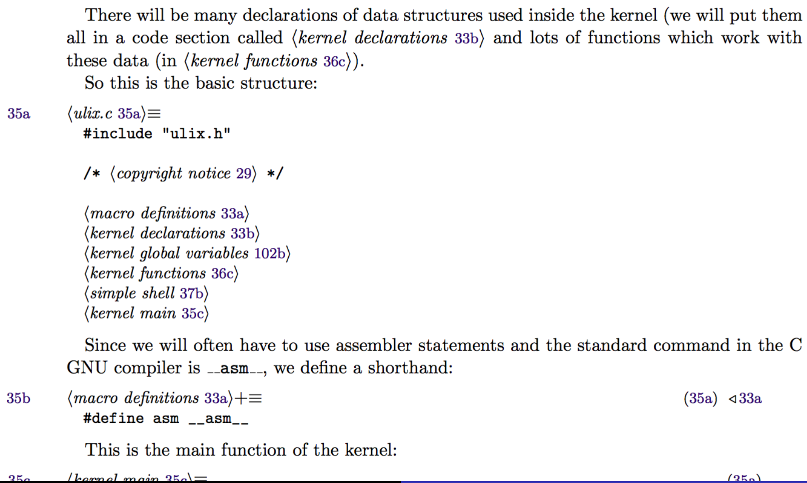
(ULIX = **L**iterate **U**nix)

ULIX Research Question:
"Is it helpful to teach students OS principles by presenting OS source code as a literate program?"

Literate Programming (Knuth)

Different approach to documentation

- *not*: Code + Documentation (à la JavaDoc)
- *but*: Documentation (book), with integrated code



There will be many declarations of data structures used inside the kernel (we will put them all in a code section called ⟨*kernel declarations* 33b⟩ and lots of functions which work with these data (in ⟨*kernel functions* 36c⟩).

So this is the basic structure:

35a ⟨*ulix.c* 35a⟩≡

```
#include "ulix.h"

/* ⟨copyright notice 29⟩ */

⟨macro definitions 33a⟩
⟨kernel declarations 33b⟩
⟨kernel global variables 102b⟩
⟨kernel functions 36c⟩
⟨simple shell 37b⟩
⟨kernel main 35c⟩
```

Since we will often have to use assembler statements and the standard command in the C GNU compiler is `__asm__`, we define a shorthand:

35b ⟨*macro definitions* 33a⟩+≡ (35a) ◁33a

```
#define asm __asm__
```

This is the main function of the kernel:

35c ⟨*kernel main* 35c⟩≡ (35a)

Design, Implementation, and Evaluation of the ULIX-i386 Teaching Operating System

My thesis will cover . . .

Part A: ∼100 pages

- overview of how ULIX was designed and implemented [x]
- description of an OS course design (using ULIX)
- evaluation of that course

Part B: ∼500 pages

- a classical textbook:
  "The Design and Implementation of ULIX" (ULIX book) [y]

Note: $y \neq x$ . . . ; current page count: 460

ULIX: Research & Teaching (1/2)

- 2008–05/2009: Felix' initial implementation of ULIX, for OS course in Mannheim
- 08/2008–06/2009: various work on the ULIX hardware
- 07/2011: Started porting ULIX to i386 architecture ($\rightarrow$ ULIX-i386)
- 10/2013–01/2014: ULIX will be used in a course called "Operating System Development with Literate Programming" (TH Nürnberg)

ULIX: Research & Teaching (2/2)

Bachelor theses supervised by me

- Implementation of a RAM Disk for the ULIX Operating System (Liviu Beraru, TH Nürnberg); 02/2013
- Implementation of an ELF Program Loader for the ULIX Operating System (Frank Kohlmann, TH Nürnberg); 02/2013
- Implementation of a Scheduler for the Educational OS ULIX (Markus Felsner, FOM München); 08/2013

not started yet (students are experimenting with the current code):

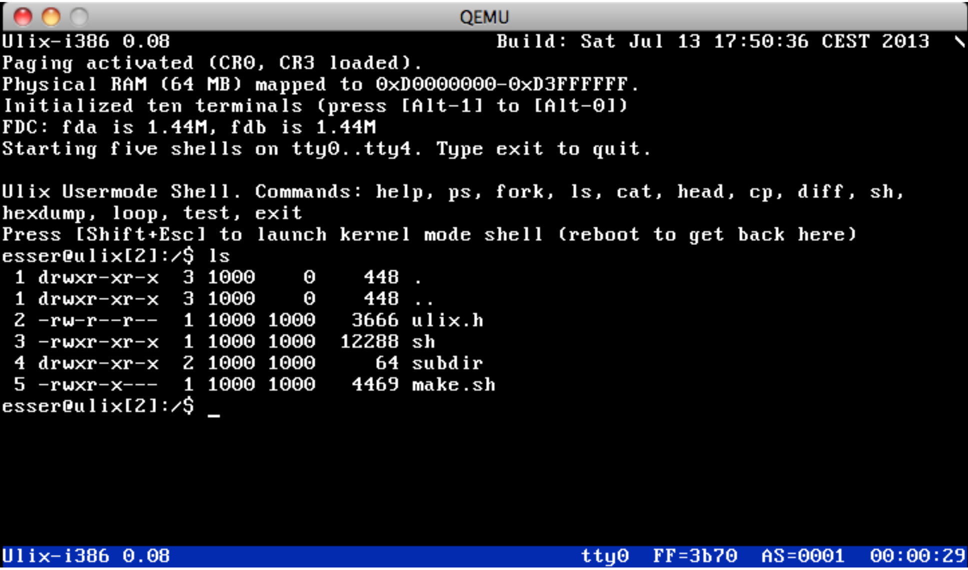
- Page Fault Handler (Florian K., FOM München)
- IP-Stack via SLIP (Cliff D., FOM München)
- N. N., VFS-related (David E., FAU Erlangen)

Students use(d) Literate Programming

What ULIX can do so far

- protected mode, virtual memory (*no paging out to disk*)
- drivers: keyboard, video (text), floppy disk, serial port, timer
- kernel mode and user mode (processes; *no threads*)
- flexible system call interface:
  - on-the-fly addition of syscalls (`insert_syscall`)
  - standard syscalls (processes: `fork`, `waitpid`, `exit`, `kill`, `signal`; files: `open`, `read`, `write`, `lseek`, `close`, `getpid`, `getcwd`, `chdir`, `link`, `unlink`)
  - non-std. syscalls: `clrscr`, `set_xy`, `get_xy`, `setterm`
- create user mode programs in C, using a standard library
- round-robin scheduler
- virtual terminals with separate shell processes
- Minix filesystem support, buffer cache
- synchronization (kernel mutexes)

ULIX-i386, version 0.08



```
Ulix-i386 0.08                         Build: Sat Jul 13 17:50:36 CEST 2013  \
Paging activated (CR0, CR3 loaded).
Physical RAM (64 MB) mapped to 0xD0000000-0xD3FFFFFF.
Initialized ten terminals (press [Alt-1] to [Alt-0])
FDC: fda is 1.44M, fdb is 1.44M
Starting five shells on tty0..tty4. Type exit to quit.

Ulix Usermode Shell. Commands: help, ps, fork, ls, cat, head, cp, diff, sh,
hexdump, loop, test, exit
Press [Shift+Esc] to launch kernel mode shell (reboot to get back here)
esser@ulix[2]:/$ ls
 1 drwxr-xr-x  3 1000    0    448 .
 1 drwxr-xr-x  3 1000    0    448 ..
 2 -rw-r--r--  1 1000 1000   3666 ulix.h
 3 -rwxr-xr-x  1 1000 1000  12288 sh
 4 drwxr-xr-x  2 1000 1000     64 subdir
 5 -rwxr-x---  1 1000 1000   4469 make.sh
esser@ulix[2]:/$ _




Ulix-i386 0.08                                 tty0  FF=3b70  AS=0001  00:00:29
```

Work in Progress

Currently working on . . .

- signal interface (syscalls `signal`, `kill`, and TCB fields `sighandlers[32]`, `sig_pending`, `sig_blocked` are there; but scheduler ignores pending signals)
- virtual filesystem
- memory management inside a process (`brk`)
- . . . and thinking about what to do in my OS Implementation course in the winter term

ULIX: To do

- Memory Management: Page Replacement ($\rightarrow$ BSc thesis)
- Proper `exec`:
  - integrate ELF loader from BSc thesis
  - switch to using one Minix floppy with kernel and binaries on it (currently all tools are functions of the shell process)
  - improve cross-compile process for ULIX programs
- properly document stuff that was recycled from other OS projects / tutorials, e. g.
  - FDD code
  - assembler code for booting
- integrate RAM disk code (BSc thesis) $\rightarrow$ `initrd`

09/2012–08/2013 (1/2)

Last Year

- Implementation of ULIX components
  - user mode shell process
  - ten virtual terminals (consoles)
  - working context switch (round-robin scheduling)
  - kernel level synchronization (mutexes)
  - floppy support, Minix filesystem
- LiPPGen
  - Development of the **Li**terate-**P**rogramming-based **P**resentation **Gen**erator (LiPPGen)
  - Publication: “LiPPGen: A presentation generator for literate-programming-based teaching”, TUGboat, Volume 34 (2013), No. 2, p. 190–195

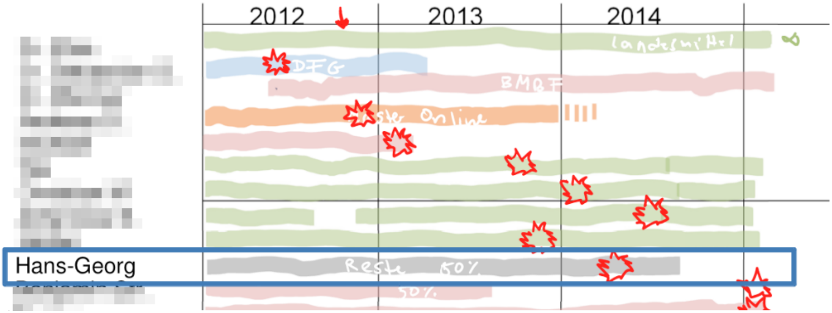
09/2012–08/2013 (2/2)

Last Year (continued)

- Teaching / Organization @ i1
  - WS 2012/13 and SS 2013: Seminar IT-Sicherheit
  - SS 2013: Systemprogrammierung, TH Nürnberg student project: implementation of a `pthread`-compatible thread library
  - Supervision of a BSc thesis at FAU (student has not started yet)
  - Univis

ULIX and my thesis – planned schedule

- 10/2013–01/2014: ULIX course at TH Nürnberg
- 02/2014: interpretation of course evaluation results
- 03/2014: ULIX 1.0 (the literate program) complete
- 04/2014: Thesis finished



from last year's slides

Demonstrations

- ULIX 0.08 live demo
- LiPPGen demo

Open Questions (1/2)

Paging to a floppy – 360 pages on a disk isn't much

- There's no IDE code for accessing HDs (and there won't be)
- Using floppies with 1.44 MB and page sizes of 4 KB
  $\Longrightarrow \quad < 360$ pages per disk
- might modify FDD driver to support 2.88 MB disks;
  not much better
- cheating: use extra RAM to page stuff out (causes no I/O)
- advanced cheating: as above, but keep some paging info on the floppy (causes I/O)
- bring back "serial disk" (access via serial port and external daemon process; unlimited space)?

None of this is pretty.

Open Questions (2/2)

Pre-/Post-testing for my ULIX course

- Students took an OS course in the summer term 2012. Contents:
  - Filesystems (FAT, NTFS, Unix System V)
  - Process Management (processes, threads, scheduling)
  - IPC (semaphores, pipelines)
  - Memory Management (paging, segmentation)
  - I/O (block vs. character devices—*no* treatment of interrupts)
  - Deadlocks
- Want pre- and post-testing to see how understanding improves through ULIX course
- What to test?
  - Goal: improve understanding of *theory*
  - Simply some theory question in areas which will be covered in ULIX course?
  - Cannot test implementation knowledge