TECHNISCHE UNIVERSITÄT DRESDEN

Faculty of Computer Science Institute for System Architecture, Operating Systems Group

Alexander Warg

Complex Lab — Operating System 2007 Winter Term

Introduction

Requirements

Basic Operating Systems Know-How
- Virtual Memory (Paging/Page Tables)
- Memory Management
- File Systems
- Device Drivers
- Processes and Threads
- Operating System Structures (What is a microkernel?)

Experience with UNIX/Linux
- Editor (for Programming)
- Using a command shell
- Development Tools (GNU Make, Binutils, GCC)

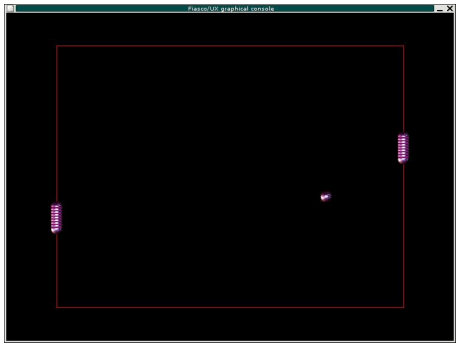
Experience with C/C++ Programming Language
- Writing C Programs
- Using Pointers (Pointer Arithmetics)
- What is the Stack?

Administration

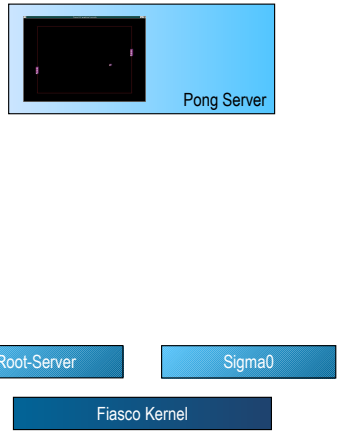
- Consultation bi-weekly (Wed. 2.50PM, INF E09)
- Website:
  http://os.inf.tu-dresden.de/Studium/Praktikum/
- Mailinglist (Announcements, Discussions):
  http://os.inf.tu-dresden.de/mailman/listinfo/kpr2007/
- Groups of 3-4 students (Individual tutor for each group)
- Presentation at the end of the part
- System requirements
  - Linux (Debian at best, Open Suse)
  - ~100 MB disk space (FRZ quota will be increased)
- Deadline for Final Assignment: 30.04.2008
  - Results containing your source codes and documentation have to be sent to the Tutor

Assignment

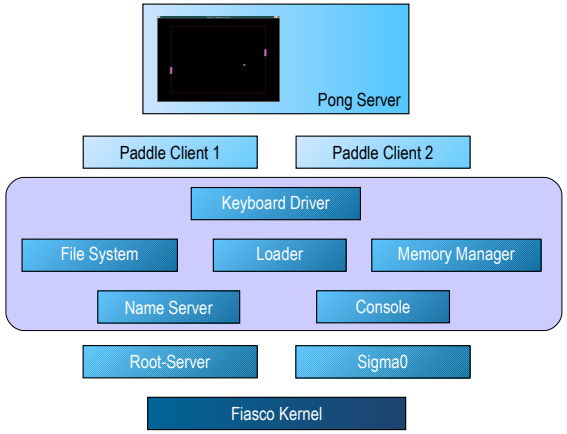
Multi-User Pong game



Course Starting Point

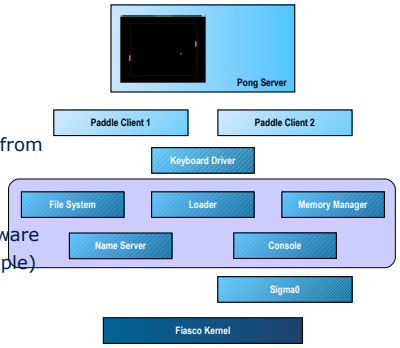


Course Goal



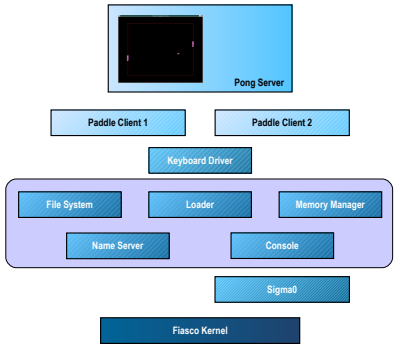
Course Goal

- Paddle Client
  - Uses pong server's paddle interface
    to set paddle position
  - Requests keyboard events from keyboard driver
- Keyboard Driver
  - Attaches to keyboard hardware
  - Provides interface to (multiple) paddle clients
- Loader
  - Starts paddle clients
  - Interprets ELF binaries



Course Goal

- File System
  - Stores paddle-client binaries /
    high-score lists / …
- Memory Manager
  - Memory allocation
- Console
  - Synchronizes text outputs
- Name Server
  - Naming service
  - ⇨ Translation server name ⇨ thread id



- Each group has to build the whole system
  - The work should be distributed among the group members
  - You should define a group leader who coordinates the work
  - You need regular group meetings to fit your work together
- "Milestones" for each consultation

⇨ Step-by-step construction of the system

- Short presentation by one group at the beginning of each consultation

In general:

- kpr2007/doc/html/index.html
  - Links to PDF documentation
  - Links to generated documentation for the Framework

Start with:

- kpr2007/doc/pdf/lnx86-21.pdf
- kpr2007/doc/pdf/bid-tut.pdf
- kpr2007/doc/fiasco_jdb.pdf
- kpr2007/doc/pong.pdf

Components of the Framework

- Fiasco Mikrokernel (configured to run on Linux)
- The Sigma0 memory manager
- Basic C/C++ Environment
- Example Root-Server
- Pong-Server and Pong-Example Client
- GNU make based build system
  - For the Fiasco-UX kernel
  - For the user-level applications (BID)

BID is a collection of Makefiles and makefile templates

- Support easy building of
  - Applications
  - Libraries
- Supports different target environments
  - We use 'normal' mode for our applications
- Simple Makefiles for Applications and Libraries

Very simple package makefiles:

```
# directories we need to know
PKGDIR ?= ../..
L4DIR ?=  $(PKGDIR)/../..

# source files
SRC_C =   main.c                # C sources
SRC_CC =  extension.cc          # C++ sources

# build target
TARGET = my_prog
# additional libraries
LIBS =
# binary link address
DEFAULT_RELOC = 0x01000000

# include BID prog rule (compile an application)
include $(L4DIR)/mk/prog.mk
```


BID templates for new packages:

```
kpr2007/src/kpr> mkdir my_pkg
kpr2007/src/kpr> cd my_pkg
kpr2007/src/kpr/my_pkg> ../../mk/tmpl/inst
```

Installs package directory structure and makefile templates

Package include files:

kpr2007/src/kpr/my_pkg/include/my_header.h

will be installed to:

kpr2007/build/l4/include/l4/my_pkg/my_header.h

can be used by other packages with:

```
#include <l4/my_pkg/my_header.h>
```


Fiasco port to Linux User-Mode:

http://os.inf.tu-dresden.de/fiasco/ux

Startup script:

```
FILES="local_rm aw-example1 aw-example-ipc aw-example-ipc2 \
       pong-server pong-client"
ROOTARGS="rom/pong-server rom/pong-client rom/aw-example-ipc2"

FIASCODIR=./fiasco
BINDIR=./l4/bin/x86_586/l4v2
FIASCO=${FIASCODIR}/fiasco

for f in ${FILES}; do
  ARGS="${ARGS} -l ${BINDIR}/$f"
done

${FIASCO}  -I ${FIASCODIR}/irq0 -S ${BINDIR}/sigma0 \
           -F ${FIASCODIR}/ux_con \
           -R ${BINDIR}/root-server" ${ROOTARGS}" \
           ${ARGS} -G800x600@16
```

Interfaces are based on IPC messages

- C++ stream syntax to generate/read messages
- Server Framework to handle server objects
- Client Stub code to provide a client side API

Examples in kpr2007/src/kpr/examples/

./hello simple Hello World Application, no IPC
./ipc simple client/server IPC (single server object)
./ipc2 client/server IPC (multiple objects)

Server Interface:

dispatch Function of server object

- Sender ID (./ipc) or Object ID (./ipc2)
- Opcode (determines the method to call)
- Ipc_iostream (input and output message)

Client Stub:

Example: src/kpr/base/

- ./include/name.h The client side interface
- ./lib/src/name.cc The client stub code

- Make familiar with source archive
- Try out Fiasco-UX
- Build client-server version of "hello":

```
class Hello {
    void show(char const * str);
};
```

- Server has to implement `Hello_server::dispatch(...)` function (print the string <str>)
- Client has to call the server (`Hello::show(...)` function)
- Try other interfaces
- Deadline for this Assignment is November 6th

Topics:

- Discussion about the solutions for the first practical assignment
- Theoretical Questions (see Requirements) in next Consultation (Prerequisite for attending the course)
- Introduction of the first building blocks:
  Name & Console Server

Literature

Operating System Basics:

- Modern Operating Systems – Andrew S. Tanenbaum (OS Concepts, OS Structures, Virtual Memory)

L4 Specific Documentation:

- http://os.inf.tu-dresden.de/L4/bib.html
- L4 V.2 Specification (http://os.inf.tu-dresden.de/L4/lnx86-21.ps.gz)
- BID, Fiasco, Environment (kpr2007/doc/...)
- Mailing Lists (kpr2007, l4-hackers)
- Doxygen comments in source code