Operating Systems Labs Agenda 2017/2018

http://d3s.mff.cuni.cz/osy

Charles University
Faculty of Mathematics and Physics

Department of Distributed and Dependable Systems

Vojtěch Horký
horky@d3s.mff.cuni.cz
General Information

- **Course web site**
  - http://d3s.mff.cuni.cz/osy

- **Course mailing list**
  - osy@d3s.mff.cuni.cz
  - https://d3s.mff.cuni.cz/mailman/listinfo/osy

- **Labs**
  - According to the schedule on the course web site
    - Odd/even weeks: *Friday 9:00 in SU2*
    - Next week: both groups together
    - Vojtěch Horký, horky@d3s.mff.cuni.cz
Course Motivation: What Are the Gains?

- Insights into the internals of an operating system kernel
  - *It is not a magical black box, but just a fairly regular program*
- Practical experiences with system-level programming
  - *Hands-on experience is much better than theoretical knowledge*
- Ability to extend existing code and follow given specification
  - *While still doing important choices*
Course Motivation: What Are the Losses?

- Some of your time will be required
  - *There is no free lunch (no pain, no gain)*
  - Prior understanding of our educational kernel
  - Implementing at least 4 out of 5 semestral tasks
    - During the labs or at home
    - Possibly passing a credits test
    - Alternatively implementing a selective semestral assignment
  - **Notice:** Developers tend to underestimate the complexity of implementation tasks up to one order of magnitude
Grading

• **Labs credits**
  - Credits for implementing semestral tasks and possibly passing the credits test
    - At most 5 tasks
  - Alternatively credits for implementing the selective semestral assignment
    - At most 30 points grading the quality of the implementation

• **Written exam**
Overall grading

- Let $P \in [0, 1]$ be the written exam score
- Let $U \in [0, 1]$ be the score from the labs
  - Derived from number of passed tasks (min 80%)
  - Or the quality of selective assignment
- The product of $U$ and $P$ determines the final grade
  - $U \times P \geq 0.7 \rightarrow 1$
  - $U \times P \geq 0.55 \rightarrow 2$
  - $U \times P \geq 0.4 \rightarrow 3$
  - Otherwise $\rightarrow 4$
### Grading (3)

<table>
<thead>
<tr>
<th>*</th>
<th>0.00</th>
<th>0.20</th>
<th>0.40</th>
<th>0.60</th>
<th>0.80</th>
<th>1.00</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>0.05</td>
<td>0.00</td>
<td>0.01</td>
<td>0.02</td>
<td>0.03</td>
<td>0.04</td>
<td>0.05</td>
</tr>
<tr>
<td>0.10</td>
<td>0.00</td>
<td>0.02</td>
<td>0.04</td>
<td>0.06</td>
<td>0.08</td>
<td>0.10</td>
</tr>
<tr>
<td>0.15</td>
<td>0.00</td>
<td>0.03</td>
<td>0.06</td>
<td>0.09</td>
<td>0.12</td>
<td>0.15</td>
</tr>
<tr>
<td>0.20</td>
<td>0.00</td>
<td>0.04</td>
<td>0.08</td>
<td>0.12</td>
<td>0.16</td>
<td>0.20</td>
</tr>
<tr>
<td>0.25</td>
<td>0.00</td>
<td>0.05</td>
<td>0.10</td>
<td>0.15</td>
<td>0.20</td>
<td>0.25</td>
</tr>
<tr>
<td>0.30</td>
<td>0.00</td>
<td>0.06</td>
<td>0.12</td>
<td>0.18</td>
<td>0.24</td>
<td>0.30</td>
</tr>
<tr>
<td>0.35</td>
<td>0.00</td>
<td>0.07</td>
<td>0.14</td>
<td>0.21</td>
<td>0.28</td>
<td>0.35</td>
</tr>
<tr>
<td>0.40</td>
<td>0.00</td>
<td>0.08</td>
<td>0.16</td>
<td>0.24</td>
<td>0.32</td>
<td>0.40</td>
</tr>
<tr>
<td>0.45</td>
<td>0.00</td>
<td>0.09</td>
<td>0.18</td>
<td>0.27</td>
<td>0.36</td>
<td>0.45</td>
</tr>
<tr>
<td>0.50</td>
<td>0.00</td>
<td>0.10</td>
<td>0.20</td>
<td>0.30</td>
<td>0.40</td>
<td>0.50</td>
</tr>
<tr>
<td>0.55</td>
<td>0.00</td>
<td>0.11</td>
<td>0.22</td>
<td>0.33</td>
<td>0.44</td>
<td>0.55</td>
</tr>
<tr>
<td>0.60</td>
<td>0.00</td>
<td>0.12</td>
<td>0.24</td>
<td>0.36</td>
<td>0.48</td>
<td>0.60</td>
</tr>
<tr>
<td>0.65</td>
<td>0.00</td>
<td>0.13</td>
<td>0.26</td>
<td>0.39</td>
<td>0.52</td>
<td>0.65</td>
</tr>
<tr>
<td>0.70</td>
<td>0.00</td>
<td>0.14</td>
<td>0.28</td>
<td>0.42</td>
<td>0.56</td>
<td>0.70</td>
</tr>
<tr>
<td>0.75</td>
<td>0.00</td>
<td>0.15</td>
<td>0.30</td>
<td>0.45</td>
<td>0.60</td>
<td>0.75</td>
</tr>
<tr>
<td>0.80</td>
<td>0.00</td>
<td>0.16</td>
<td>0.32</td>
<td>0.48</td>
<td>0.64</td>
<td>0.80</td>
</tr>
<tr>
<td>0.85</td>
<td>0.00</td>
<td>0.17</td>
<td>0.34</td>
<td>0.51</td>
<td>0.68</td>
<td>0.85</td>
</tr>
<tr>
<td>0.90</td>
<td>0.00</td>
<td>0.18</td>
<td>0.36</td>
<td>0.54</td>
<td>0.72</td>
<td>0.90</td>
</tr>
<tr>
<td>0.95</td>
<td>0.00</td>
<td>0.19</td>
<td>0.38</td>
<td>0.57</td>
<td>0.76</td>
<td>0.95</td>
</tr>
<tr>
<td>1.00</td>
<td>0.00</td>
<td>0.20</td>
<td>0.40</td>
<td>0.60</td>
<td>0.80</td>
<td>1.00</td>
</tr>
</tbody>
</table>
Labs Credits

- Two alternative ways of getting the credits
  - **Semestral tasks**
    - Five topics in total
    - Skipping a certain number of topics can be compensated by passing a credits test
  - **Selective semestral assignment**
    - For single students or a team of multiple students
    - Individual schedule, requirements and credits criteria
Semestral Tasks

- Five distinct topics
  - Memory management
  - Virtual memory
  - Synchronization
  - System calls
  - Device drivers

- Each semestral task is dedicated to one topic
Semestral Tasks (2)

- **During the labs**
  - The topic will be announced using the mailing list in advance
    - Including a recommended reading list
  - The detailed task specification will be handed out during the labs
  - The task has to be implemented during the labs where it was handed out (90 minutes should be sufficient)
    - If successful, the task is marked as *fulfilled*
  - The teacher is there to answer questions and give you basic guidance

- **As a home assignment**
  - The topic and a detailed task specification will be announced using the mailing list
  - The task has to be implemented during the following 14 days (the task is more complex than the task for the labs)
    - If successful, the task is marked as *conditionally fulfilled*
  - For questions and basic guidance use the mailing list
Semestral Tasks (3)

- **Grading of each task**
  - **Fulfilling the necessary requirements**
    - Fulfilling the required properties of the implementation
    - Fulfilling the required interface
    - Passing the unit tests provided
    - ...
  - **Other criteria**
    - Quality of the implementation (programming techniques, data structures)
    - Efficiency
    - Coding style quality (code readability, structure, consistency)
    - Quality of comments (if required)
Semestral Tasks (4)

- **Getting the credits**
  - At least 4 *fulfilled* topics → *credits*
  - At least 4 *fulfilled* and *conditionally fulfilled* topics → possibility to take the credits test

- **Credits test**
  - Similar form as the semestral tasks
    - Slightly more complex, usually for 2 hours
  - Similar requirements and criteria
    - If passed → *credits*
  - During the exam period
    - Up to 2 tries
Selective Semestral Assignments

- **Individual (bespoke) assignments**
  - A non-trivial topic related to real operating systems
    - For somewhat experienced developers
  - The topic, schedule, requirements and credits criteria need to be agreed on with the teacher
  - For single students or a team of students (up to 4)
    - This affects the complexity, deadlines, etc.
  - Possibility to consult with external companies
    - Oracle, Red Hat, SUSE, Microsoft, Avast, ...
Selective Semestral Assignments (2)

- **Deadlines**
  - Depending on the agreement with the teacher, but usually:
    - **Beta/Milestone:** End of the winter semester
    - **Final version:** End of the summer semester

- **Grading**
  - Quality of the implementation (programming techniques, data structures)
  - Fulfilling the required interface
  - Efficiency
  - Coding style quality (code readability, structure, consistency)
  - Quality of comments and of the developer's documentation
  - \( U \geq 0.5 \rightarrow \text{credits} \)
Possible Topics

- HelenOS
  - [http://www.helenos.org](http://www.helenos.org)
  - Research microkernel multiserver operating system in development at the Faculty of Mathematics and Physics
    - Porting to a new hardware architecture (i.e. ARMv8, MIPS64, Cell, Motorola 68xxx)
    - Adding support for an actual hardware (i.e. Ben NanoNote, Lemote Fuloong)
    - Real-time scheduling
    - Copy-on-write mechanism
    - ...

Possible Topics (2)

- GNU/Linux, Windows, macOS, *BSD, MINIX 3, seL4, Genode, Haiku, RTEMS, ...
  - Any non-trivial system-level topic (needs to be agreed on with the teacher)
  - The usual areas
    - Device drivers
    - Subsystem enhancements
    - Ports to a new architecture
    - Porting code from a foreign operating system
    - Performance improvements
    - Benchmarking
  - See Google Summer of Code ideas lists
Grading in a Nutshell

- **5 semestral tasks**
  - at least 4 fulfilled and conditionally fulfilled

- **selective semestral assignment**
  - at least 4 fulfilled
  - $U \geq 0.5$

- **credits test**
  - passed

- **overall grading**
  - $U \in [0, 1]$
  - $P \in [0, 1]$

- **written exam**
  - $U \times P$

- **credits**
  - $U \in [0, 1]$

- **U x P**
  - $\geq 0.7$
  - $\geq 0.55$
  - $\geq 0.4$
  - $< 0.4$

- **passed**
  - $U \in [0, 1]$
Side-note: Cheating, etc.

- Only your original code will be graded

  - This course requires individual/independent work
    
    - You want to learn how to code an operating system, right? Well, there is no other way to learn it than actually writing the code yourself.
    
    - Learning the tricks to pass off foreign code as original code is not the purpose of this course
      
      - It is surprisingly easy to detect such frauds (with and even without tools for that purpose)
      
      - Frankly, it is embarrassing for both parties
    

  - **Practical rule:** Whenever you use someone else's code, declare it clearly as such (never mislead anyone about the source of any code, even unintentionally)
    
    - If declared properly, we won't punish you for using foreign code (but it might obviously affect the grading in certain cases)
Labs Environment
MIPS R4000
Sources of Information for the Labs

- **Documentation**
  - Specification of the MIPS R4000 processor and other materials
  - On the course web site

- **Guidance**
  - Questions on the mailing list
    - *The more elaborate the question, the more elaborate the answer*
  - Individual consultations in person (ask your teacher)
    - *The better preparation, the better effect*
  - Mass consultations
    - Just ask on the mailing list
Tools to Master

- **Cross-compiler toolchain for MIPS**
  - `toolchain.mips.sh` and precompiled binaries from the course web site

- **MIPS R4000 simulator**
  - MSIM
  - Sources and precompiled binaries from the course web site

- **Simple educational operating system for MIPS R4000**
  - Kalisto
  - Sources from the course web site
Simple educational operating system for MIPS R4000

- Uses the cross-compiler toolchain to compile and MSIM to run
  - Expects a standard location of the cross-compiler toolchain
    - /usr/local/cross/mips32
  - The location can be changed via the CROSS_PREFIX environment variable
Power-on Self-test
Initial Orientation in the Labs

1. Subscribe to the mailing list.
2. You shall receive the credentials to log in to a GNU/Linux machine uniform.ms.mff.cuni.cz after this lab
   - If you are unfamiliar with GNU/Linux and basic development tools (i.e. make), contact us ASAP
3. Log in to uniform.ms.mff.cuni.cz
4. Download Kalisto from the course web site and extract it (wget & tar)
5. Compile Kalisto using make
6. Run Kalisto using msim
   - Send me (horky@d3s.mff.cuni.cz) the output of the simulation of the default run.
7. Run the unit tests using tests.sh (from the Kalisto source tree)
Q&A