The RTOS Chameleon for Linux

Jan Kiszka
Dipl.-Ing.
Leibniz Universität Hannover
Real-Time Systems Group

2007-01-24
Which RT-Linux technology?
- Co-scheduling?
- ...or native real-time Linux?

Which kernel
- Always latest 2.6?
- Or also older revisions?
- ...or even keep 2.4?

How to port from $RTOS to Linux?
- Migrate to POSIX API?
- ...or emulate the legacy API?

How to create and maintain RT-optimised drivers?
Presentation Outline

- Xenomai Approach
  - Provided APIs
  - Real-Time Driver Model
  - Architectures
  - New RT-Technologies
- Related Open Source Projects
- Application Example
- Summary & Prospects
The Xenomai Approach

- Generic RT-core ("nucleus")
- RTOS APIs provided via "skins"
- Includes hard-RT Linux technology ("I-pipe")
  - Kernel-independent
  - Light-weight
  - **But:** Highly integrated in Linux environment
- Portability framework for older kernels (2.4 and 2.6)
- Generic RT-driver model across all skins

**Our goal:**

*Real-Time APIs for any Linux*

(OK, almost any)
What Skin Do You Prefer?

- POSIX
- Native (clean RTAI-like API)
- VxWorks
- VRTX
- pSOS+
- μITRON
- RTAI
- RTDM

Avionics
ARINC 653?
Automotive
OSEK?
AUTOSAR?
Drive Hardware in Real-Time

RTDM – The Real-Time Driver Model

- Lean driver development framework
- POSIX I/O Model
- "Set-top box" for Linux
  - RTDM: time-critical services
  - Linux: non-RT setup/cleanup
- Device profiles ensure application portability
- Xenomai-independent design
  -> RTAI integrates RTDM too
- Example: Integrated RT-CAN stack
  -> Socket-based API for any CAN controller
## Permanent Work in Progress

### Supported architectures

<table>
<thead>
<tr>
<th>Architecture</th>
<th>Kernel</th>
<th>2.4</th>
<th>2.6</th>
</tr>
</thead>
<tbody>
<tr>
<td>i386</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>PPC32</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>PPC64</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>ARM</td>
<td>✔</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IA64</td>
<td>✔</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Blackfin</td>
<td>✔</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Simulator</td>
<td><strong>User-Space Application</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x86-64</td>
<td><strong>WIP</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
- Linux incrementally gains RT support ("PREEMPT_RT", Ingo Molnar et al.)
- Xenomai is going to adopt this feature
Xenomai Featuring...

- RTnet
- RT-FireWire
- USB4RT, USB20RT
- COMEDI over RTDM
- OROCOS
- RACK
- CanFestival
- Xeno--
- LTTng
- kgdb
- ...

...
Application Example: Real-Time Robotics

- Modular autonomous service robots
- Research and industrial scenarios
- Real-time 3D ladar sensor
- Low-end x86 IPCs
- RACK, RT-CAN, RTnet, fast UARTs
- Integrates standard hardware/software with strict RT
**Summery & Prospects**

- **Xenomai**: RTOS construction kit for Linux
- Portability as major goal
  - Between architectures
  - Between RT-technologies
  - Between kernel versions
- Home for RT-drivers / stacks
- What is the future about?
  - PREEMPT_RT support, more RTOS skins & drivers, ...
  - **One stop for RT**: kernel, drivers, libs, community
- And when?
  - *Counter question*: What do you need first?
  - *Any contribution/support can accelerate development!*
Thank You!

www.xenomai.org

Jan Kiszka