LKST for SH

Lineo Solutions, Inc
Presentation Overview

- LKST (Linux Kernel State Tracer) Porting to SH4
  - Renesas RTS7751R2D (CELF reference platform)
  - LKST kernel configuration
- Key Point of the Porting
  - Output comparison of x86 (Reference Model) vs. SH
- Examples for the LKST Output
- Demonstration
  - Visualization technique of LKST output
Our Motivation & Objective

- LKST as Debugging Tool
  - Powerful and efficient
  - Event tracing function is useful for trouble analysis
- Porting of Major Tracers (LKST, LTT, …)
  - LTT: Version 0.9.5a supports x86, PPC and SH architecture
  - LKST : supports x86
- Contribution to Linux Improvements in Numerical Quantification Aspect
  - Performance Evaluation (Plans, exams and analyses with Visualization)
  - Supporting Performance Evaluation (Porting and/to integrated environments)
System Environments

- Hardware
  - RTS7751R2D(SH4)

- Software
  - Linux 2.6.8.1
  - LKST 2.2.1
  - GCC 3.2.3

- Cooperative Development for This System
  - System Concept and Hardware Support
  - Technical Advisory for LKST Technologies
  - System Construction
LKST Kernel (1/2)

- [Setup LKST Kernel Configuration Environments](http://sourceforge.jp/projects/lkst/)
  - lkst-2.2.1.tar.gz
  - lkstpatchset-2.2.1-for-2.6.8.1-2.tar.gz

- `http://www.kernel.org`
LKST Kernel (2/2)

- Expand linux-2.6.8.1.tar.gz
- Expand lkst-2.2.1.tar.gz
- Expand lkstpatchset-2.2.1-for-2.6.8.1-2.tar.gz
- mv lkst-2.2.1/patches lkst-2.2.1/patches-2.6.9
- mv patches-2.6.8.1 lkst-2.2.1/patches
- make patch KPRESRC= <Kernel Expand Directory>
### Port to SH CPU (1)

- 基于Linux内核的移植
- 基于SH CPU的移植

<table>
<thead>
<tr>
<th>File Name</th>
<th>Comment</th>
<th>Event Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>arch/sh/kernel/irq.c</code></td>
<td>Adds hook-points to <code>do_IRQ()</code></td>
<td><code>INT_HARDWARE_ENTRY</code></td>
</tr>
<tr>
<td><code>arch/sh/kernel/process.c</code></td>
<td>Adds hook-points to <code>kernel_thread()</code></td>
<td><code>PROCESS_LTHREAD GEN</code></td>
</tr>
<tr>
<td><code>arch/sh/kernel/time.c</code></td>
<td>Adds cpu_khz variables initialization using <code>time_init()</code></td>
<td></td>
</tr>
<tr>
<td><code>arch/sh/mm/fault.c</code></td>
<td>Adds hook-points to <code>do_page_fault()</code></td>
<td><code>LOOPS_PGFAULT</code></td>
</tr>
<tr>
<td><code>arch/sh/boot/compresse d/misc.c</code></td>
<td>Adds <code>#define __DISABLE_LKST_HOOK__</code></td>
<td></td>
</tr>
</tbody>
</table>
## Port to SH CPU (2)

<table>
<thead>
<tr>
<th>File Name</th>
<th>Comment</th>
<th>Event Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>include/asm-sh/hook.h</td>
<td>SH Porting corresponding _IF_HOOK_ENABLED in i386</td>
<td></td>
</tr>
<tr>
<td>include/asm-sh/hook_private.h</td>
<td>SH Porting Corresponding to is_asm_hook() in i386</td>
<td></td>
</tr>
<tr>
<td>include/asm-sh/lkst.h</td>
<td>Changes Defined Value for LKST_BUFFER_SIZE_MAX (1MByte for Default)</td>
<td></td>
</tr>
</tbody>
</table>
| include/asm-sh/lkst_etype.h | Comment Out for SYSCALL_SYSENTER and SYSCALL_SYSEXIT  
SH Porting Corresponding to Atomic_read_and_add() in i386  
SH Porting Corresponding to local_atomic_read_and_add() in i386 |            |
## Port to SH CPU (3)

<table>
<thead>
<tr>
<th>File Name</th>
<th>Comment</th>
<th>Event Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>include/asm-sh/</td>
<td>SH Porting Corresponding to</td>
<td></td>
</tr>
<tr>
<td>lkst_private.h</td>
<td>lkst_evhandlerprim_mc() in i386</td>
<td></td>
</tr>
<tr>
<td>include/asm-sh/timex.h</td>
<td>Adds extern Declaration for cpu_khz</td>
<td>SYSCALL_ENTRY_HEADER</td>
</tr>
<tr>
<td></td>
<td>Adds hook-points to syscall_call</td>
<td></td>
</tr>
<tr>
<td>arch/sh/kernel/entry.S</td>
<td>Adds hook-points to syscall_exit</td>
<td>SYSCALL_EXIT_HEADER</td>
</tr>
<tr>
<td></td>
<td>Adds DEBUG_KERNEL</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Adds source &quot;drivers/lkst/Kconfig&quot;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Adds config depends on config HOOK DEBUG_KERNEL</td>
<td></td>
</tr>
<tr>
<td>arch/sh/Kconfig</td>
<td>Adds config ASM_HOOK</td>
<td></td>
</tr>
<tr>
<td>include/asm-sh/hook*.h</td>
<td>config ASM_HOOK</td>
<td></td>
</tr>
</tbody>
</table>
Port to SH CPU (4)

• Output Example for LKST

```bash
>>>/root/lkstutils/lkst stat
press return key:

Current status:
version of LKST: 2.2.1
number of cpus: 1
number of masksets: 3
number of event-handlers: 3
current maskset_id: 2
current writing buffer_id (cpu: 000): 0

>>>/root/lkstutils/lkst stop
press return key:
Stop LKST event tracing.

>>>/root/lkstutils/lkst start
press return key:
```
Port to SH CPU (5)

• Output Example of LKST

```c
event_type=interrupt_hardware_entry
  cpu=00, pid=00000410
  time=Sat Jan 01 00:00:01.060551991 2000
  arg1=0x0000010 00000000 : IRQ number irq
  arg2=0x0000001 00000000 : interrupt status status
  arg3=0x8f8f3e68 00000000 : pointer to register stack

event_type=process_add_waitq
  cpu=00, pid=00000410
  time=Sat Jan 01 00:00:01.060551275 2000
  arg1=0x8f8f3c40 00000000 : pointer to wait_queue_head
  arg2=0x8fe6a460 00000000 : pointer to added process

event_type=context_switch
  cpu=00, pid=00000000
  time=Sat Jan 01 00:00:01.060551258 2000
  arg1=0x8c21ba9c 00000000 : pointer to task_struct prev
  arg2=0x8fe6a460 00000000 : pointer to task_struct next
  arg3=0x00000000 00000000 : process state
  arg4=0x00000000 00000000 : process count
```
Port to SH CPU (6)

- Porting was Smooth & Quick
  - Become Available … about 1 month
  - Evaluation, Comparison with x86, Visualizing Tool … another 1 month
Demonstration

• Development of Visualization Tool for Output Results
  – From huge & complex output log text
  To BIOS-like Display
  – Implemented on Target Board
  • Output Log can be checked right there on the target board.
Visualization Tool

- Block Diagram

On RTS7751R2D(SH4)
Visual Demo

- Video demo is available
Summary

• Through LKST porting, we found;
  • High & smooth portability
  • Possible future approach: Usage as new visualizing tool

• Scopes in the future
  • Port other major/useful tracers
  • Stacking analysis
    • We Contribute to improve quality of Linux in its performance from the numerical quantification viewpoint.

• Our challenge
  • Hook points to the system calls and exception processing
Thank You!

ATTENTION
Corresponding Demonstration
Jan. 25  17:00-18:30