Hardware-assisted software tracing

Adrien Vergé
Hardware tracing

ARM: Coresight

Freescale: Performance Monitor

Use for LTTng
Hardware-assisted tracing

Software tracing have troublesome side-effects:

- interrupts
- system events
  - cache flushing, etc.
- may add some latency

Hardware tracing:

- observes the system: almost zero overhead
- provides dedicated resources (buffers...)
Hardware-assisted tracing

Use of hardware components
▶ on chip

Advantages:
▶ dedicated circuit for tracing
▶ very detailed info
  ▶ executed instructions sequence
▶ customizable events
Studied platforms

- ARM (Coresight)
- Freescale (Performance Monitor)
ARM Coresight

What is it?

- collection of hardware components
- open architecture
- goal: trace and debug an entire SoC
  - CPUs, DSPs, buses, software (!)

Features:

- dedicated buffers
- off-chip connection (JTAG)
- direct software tracing
Coresight: ETM

Embedded Trace Macrocell

- monitors the core’s internal buses
  - no burden on performance
- hardware triggers
  - start tracing only when needed
- hardware filters
  - output only what needed
- data compression
  - full executed instructions sequence: 1 bit / cycle
Coresight: STM

System Trace Macrocell

- dedicated hardware to store software events
- timestamped
  - correlated with other events
- multisource trace in a single stream
  - optimized bandwidth
Coresight: trace buffers

Embedded Trace Buffer / Trace Memory Controller

- dedicated buffers
- multiplex data from different sources
- on-chip and off-chip connection (JTAG)
Studied platforms

Beagleboard
- OMAP3530
- ARM Cortex-A8

Pandaboard
- OMAP4430
- Dual-core ARM Cortex-A9
Freescale: Performance Monitor

Freescale processors
- PowerPC architecture

Performance Monitor
- collection of counters
- hundreds of events
  - cache misses, type of instructions, branches...

Nexus module
- program trace
- data trace
- taken interrupts...
Studied platform

Freescale QorIQ P3041

- 4 e500mc cores
- 8 Performance Monitor counters / core
- Nexus module
- external debug interface
What use for LTTng?

Research:

- What improvements can be brought to LTTng?
  - push tracing functions onto hardware?
  - enhance context info?

- Common characteristics within different hardware tracing solutions?

- What is hardware-specific?
What use for LTTng?

Planned work:

1. Activate and configure hardware tracing modules on studied processors
   - see what we can do

2. Gather common characteristics

3. Port LTTng to support them
   - enable user customization of traces
   - create efficient algorithms to avoid overhead

4. Compare performance before and after
Questions?