Open-source Multicore debugging tools for RISC-V BoF

RISC-V Summit, Santa Clara
4 December 2018

Graham Markall
(Compiler Engineer)
Compiler Tool Chain Development

Hardware Modeling

Open Source Tool Support

Machine Learning Optimization

Optimization for Energy Efficiency

Superoptimization

Compilation for Security

GCC

OpenOCD
Motivation

Multicore RISC-V system design choices and memory layouts

Support in open-source tools (GDB, GDBservers, OpenOCD)
Plan

Spend a few minutes:

• outlining where I think the state of things is in open-source tools
• Discussing changes that need to be made to the tools

Discussion:

• Understand what software tooling people are working with, what problems they’re having
• What are people working on
Single-inferior, multiple debugger

**Pros / Cons:**

- **Pro:** Simple, few if any debugger modifications needed
- **Con:** Not scalable
Single inferior, multiple threads

Pros / Cons:

- **Pro:** Simple, suits multiple cores sharing same memory
- **Con:** Cores must run the same executable
Multiple inferior, single address space

Pros / Cons:

- **Pro:** Cores can load the same, or different executables
- **Pro/Con:** Suits multiple cores sharing same memory
- **Con:** Need GDB changes at present
Multiple inferior, multiple address spaces

Pros / Cons:
- **Pro**: Cores can load the same, or different executables
- **Neutral**: Suits multiple cores with private memory
- **Con**: Need GDB changes at present
Multi-inferior, single/multi-address space

Pros / Cons:

• **Pros:** Shared / private memories, same/different executables

• **Con:** Need GDB changes at present
Tools status: GDB

• Multiple inferiors need GDB changes
• Duplicated debug info for multiple programs
  – May be a size problem
• Multiple programs linked into single binary can duplicate code at several addresses
  – GDB changes needed here
GDB Patches – need upstreaming

- Multiple inferiors can exist at the point of connection, but upstream GDB does not expect this - it only expects fork/join multi-inferior behaviour. In general, some customisation is required to deal with target-specific setup.

- Support for shared address spaces needed extending in GDB, in particular those parts concerned with bare-metal support.

- Hosted IO - this is designed for embedded single-core targets, and was written using a single inferior model. We extended hosted IO to support multiple inferiors.
Tools Status: OpenOCD (work with SiFive)

- Background: functional OpenOCD port
- Aim – increase quality of debug
- Metric: using GDB testsuite with OpenOCD and dual-core SiFive E31/E51 Coreplex and single-core Freedom E310
- Multicore approaches:
  - “Multi-GDB” mode: Single-inferior, Multiple-debugger
  - “RTOS” mode: Single-debugger, Multiple-thread
- Fixes all in riscv/riscv-openocd
<table>
<thead>
<tr>
<th></th>
<th>Freedom E310 Arty</th>
</tr>
</thead>
<tbody>
<tr>
<td># of expected passes</td>
<td>(+19572) 34078</td>
</tr>
<tr>
<td># of unexpected failures</td>
<td>(-3136) 154</td>
</tr>
<tr>
<td># of unexpected successes</td>
<td>(+1) 1</td>
</tr>
<tr>
<td># of expected failures</td>
<td>(+3) 24</td>
</tr>
<tr>
<td># of known failures</td>
<td>(+10) 23</td>
</tr>
<tr>
<td># of unresolved testcases</td>
<td>(-307) 12</td>
</tr>
<tr>
<td># of untested testcases</td>
<td>(+3) 259</td>
</tr>
<tr>
<td># of unsupported tests</td>
<td>(+19) 320</td>
</tr>
</tbody>
</table>
OpenOCD + GDB Test suite results (2)

<table>
<thead>
<tr>
<th></th>
<th>Coreplex E31 Arty</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>multi-GDB</td>
<td>RTOS</td>
<td>RTOS</td>
</tr>
<tr>
<td># of expected passes</td>
<td>(+3529) 38506</td>
<td>(+5144) 38579</td>
<td></td>
</tr>
<tr>
<td># of unexpected failures</td>
<td>(-669) 359</td>
<td>(-1066) 370</td>
<td></td>
</tr>
<tr>
<td># of unexpected successes</td>
<td>(=) 1</td>
<td>(=) 1</td>
<td></td>
</tr>
<tr>
<td># of expected failures</td>
<td>(-3) 27</td>
<td>(=) 27</td>
<td></td>
</tr>
<tr>
<td># of known failures</td>
<td>(=) 21</td>
<td>(=) 24</td>
<td></td>
</tr>
<tr>
<td># of unresolved testcases</td>
<td>(-4) 16</td>
<td>(-13) 16</td>
<td></td>
</tr>
<tr>
<td># of untested testcases</td>
<td>(+24) 284</td>
<td>(+24) 284</td>
<td></td>
</tr>
<tr>
<td># of unsupported tests</td>
<td>(+12) 324</td>
<td>(+12) 324</td>
<td></td>
</tr>
</tbody>
</table>
# OpenOCD + GDB Test suite results (3)

<table>
<thead>
<tr>
<th></th>
<th>Coreplex E51 Arty</th>
<th>multi-GDB</th>
<th>RTOS</th>
</tr>
</thead>
<tbody>
<tr>
<td># of expected passes</td>
<td></td>
<td>34898</td>
<td>34890</td>
</tr>
<tr>
<td># of unexpected failures</td>
<td></td>
<td>300</td>
<td>312</td>
</tr>
<tr>
<td># of unexpected successes</td>
<td></td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td># of expected failures</td>
<td></td>
<td>19</td>
<td>19</td>
</tr>
<tr>
<td># of known failures</td>
<td></td>
<td>22</td>
<td>22</td>
</tr>
<tr>
<td># of unresolved testcases</td>
<td></td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td># of untested testcases</td>
<td></td>
<td>359</td>
<td>359</td>
</tr>
<tr>
<td># of unsupported tests</td>
<td></td>
<td>313</td>
<td>313</td>
</tr>
</tbody>
</table>
Questions:

• What models discussed fit in with people’s needs?
• Are other models required?
• Any active work on GDB, GDBServer(s), OpenOCD to collaborate on?
• What about ISA heterogeneity? (see Gajinder Panesar of UltraSoC talk at 2:50pm!)
Broader Questions

• Who is working on a multicore RISC-V system?
• What are your plans for debug?
• What about interactions between debugger and IDEs etc?
  − Issues with it?
  − Mapping of models (e.g. multiple inferior, etc) onto IDEs?
Thank you!

https://www.embecosm.com/

Graham Markall
(Compiler Engineer)