The lowRISC project

Alex Bradbury

lowRISC C.I.C.

3rd April 2017

lowRISC

- We are producing an open source Linux capable System-on-a-Chip (SoC)
  - 64-bit multicore
  - Aim to be the
- *“Linux of the Hardware world”*

Motivation

- An open-source Raspberry Pi style board?
- Encourage community effort by providing regular tape-outs
- Timeliness

The lowRISC platform

- Extends the Berkeley Rocket chip generator
- Opportunity to improve the status quo?
- Not just delivering the hardware, we aim to share as much information as possible
- Encourage development of other open-source hardware (incl. PHYs and analog IP)

The lowRISC platform

- **Minion cores**
  - Simple cores are free, tape-outs are expensive
  - Aim to use minion cores liberally to reduce complexity, distribute processing and create a more flexible SoC (single ISA)
- **General-purpose tagged memory support**
  - Support for fine-grain memory protection, debugging, profiling, synchronization, etc.
  - Can also be extended to support provenance tracking, capabilities, etc.

Minion cores

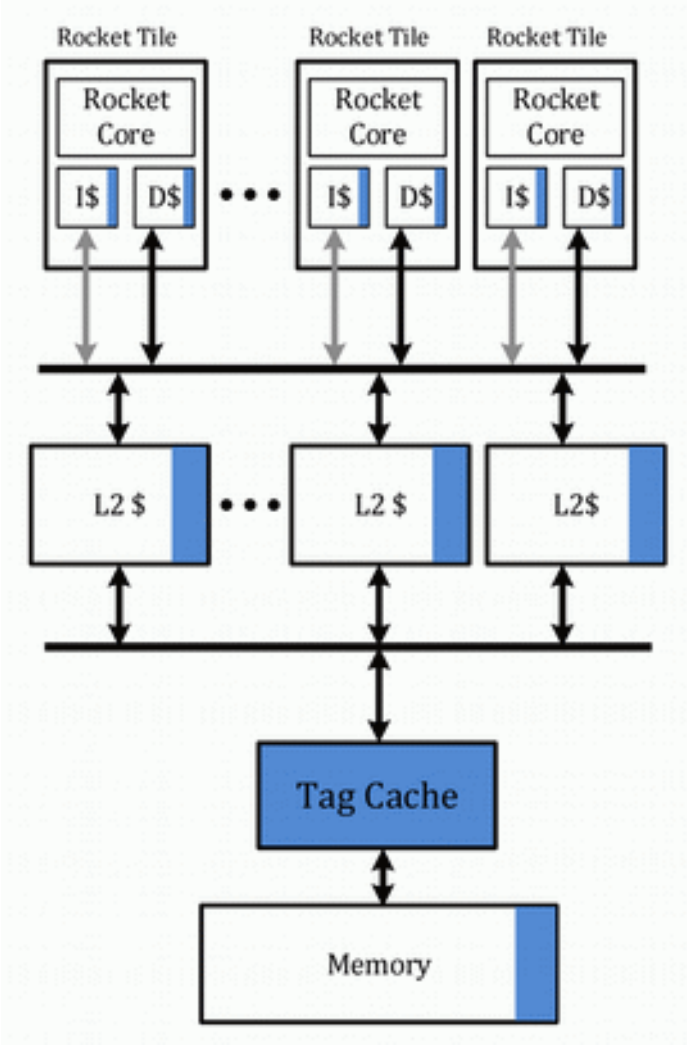
Minion = simple core + custom instruction-set extensions + HW “shim” +

- low-latency communication links
- Based on PULPino RISC-V core from
- ETH Zurich (www.pulp-platform.org)
- Initially used for programmable I/O
  - Preprocessing, monitoring, tagging of I/O data
  - Secure isolated execution, thread offload, rapid/flexible implementation of IP blocks,
- use multiple as a parallel processing resource,

…

Tagged memory

- Associate tags (metadata) with each physical memory location
- Tags are stored in on-chip caches and a tag cache is inserted between the LLC and main memory



Tagged memory security use cases

- Protection of code pointers
- Limited control-flow integrity
- Infinite hardware memory watchpoints
  - E.g. canaries on stack
- Poisoning (simple IFT)
  - Ensure sensitive data does not leak

Tagged memory - example

- Tag rules are currently specified using a single control register (CSR) <64-bits
  - Specifies bit masks that indicate when exceptions are raised and when tags are propagated for different classes of instructions
- E.g. code pointer protection
  1. Mark valid code pointers
  2. Prevent them being overwritten with arbitrary data

Tagged memory - example

- Choose a single tag bit (CPTR) for this use
- Load: ensure the CPTR tag bit is propagated
- Trigger exception if during indirect branch rs1 doesn’t have CPTR tag set
- Store
  - Trigger exception if attempting to write to memory location holding a code pointer
  - Ensure CPTR tag bit is stored (i.e. create CP)

Tagged memory - example

```
e.g. CPTR = 4'b0001
Load_propagation_mask = CPTR
Load_exception_mask = 0
INDIR_BRANCH_exception_mask = CPTR
INDIR_BRANCH_rs_mask = CPTR
Store_exception_mask = CPTR
Store_propagation_mask = CPTR
ALU_propagation_mask = 0
ALU_exception_mask = 0
```

Extending tagged memory

- Could provide larger rule table or cache rules [see PUMP work, DARPA CRASH project]
- Could exploit unused bits of virtual address to extend scheme (e.g. to provide a capability or provenance ID)
- Tag rules could trigger operations on a minion core

RISC-V LLVM Compiler

- LLVM is a prerequisite for many industrial and academic projects. We plan to use it for a number of tagged memory use-cases
- We are producing a well documented “reference” backend for RISC-V
- Alex Bradbury is leading this effort
  - >90% of GCC torture test suite is passing (RV32I). Basic support by end Q2
  - Full support, incl. ISA variants, est. Q4’17.

Status

- See lowrisc.org for current releases + tutorials
- **Current release (v0.3)**
  - Extends Rocket chip
    - Based on the Rocket 64-bit in-order core
  - untethers Rocket chip (from ARM host)
  - Verilog top-level (rather than Chisel)
  - Trace debug support (Open SoC Debug)
  - Supports low-cost Nexys4 DDR FPGA board

Status

- **New release** (April 2017)
  - Tagged memory implementation
    - Optimized hierarchical tag cache
    - Minimises additional off-chip traffic due to tags
    - Tag rule checking in Rocket core pipeline
    - GCC stack smashing protection example
  - Integration of Pulpino-based minion core

(with SD-card demo)

Future releases

- Full tagged memory support
  - with Linux and LLVM support, Q3’17
- More minion-based IP and integration support (Q4’17)
- Customisable and extensible SoC platform with system-level simulation support (2018)
- Test-chips 2018, Community board 2019

Funding

- Traditional research funding sources
- Donations and sponsorship
  - private individuals, Google
- lowRISC C.I.C
  - Not-for-profit organisation
  - Undertakes RISC-V consultancy work
  - Profits are used to support the project and community

5-year vision

- Industry
  - Easier to create derivative designs, faster time-to-market
  - Significantly lower the barriers to producing an SoC design
  - Cores are modified/adapted and used freely to quickly produce flexible SoCs
  - Shared technology investment permits focus on differentiation and innovation
    - Shared costs and reduced risk

5-year vision

- Academia
  - Undertakes more detailed evaluations and design comparisons (reproduces results!)
  - Ideas transferred more quickly to industry
  - More of the best students interested in hardware

How can we help?

- What are your priorities and areas of particular interest?
- Potential for collaborations?
- Might you have IP to contribute?

Thank you

asb@lowrisc.org
www.lowrisc.org