Towards a formal x86 ISA specification

Alastair Reid / @alastair_d_reid

Imperial College, 6th September 2022

About me

https://alastairreid.github.io/papers/

@ Arm Research: 15 years

- Creating formal Arm ISA specification
- Formal verification of CPUs

@ Google Research: 2 years

- Software verification (Rust)

@ Intel Labs: 9 months

- Creating formal x86 ISA specification

Most projects I work on take 3-5 years to produce results

Our industry needs better ISA specs

For humans

- Clear, trustworthy, authoritative human readable specifications

For tools

- To verify hardware https://alastairreid.github.io/papers/CAV_16/
- To verify (critical) software
- To check that (critical) software is secure?
- To verify compiler backends
- To build binary analysis tools (e.g., malware analysis)

Existing formal ISA specifications

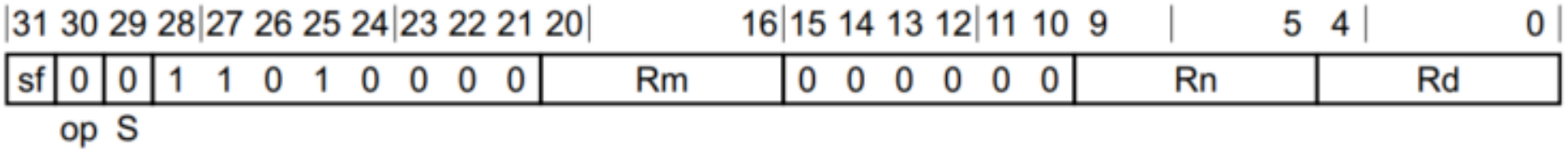
Arm formal ISA spec

- Arm's official specification
- Can boot Linux
- Publicly available in machine readable form
- Passes Arm's architectural conformance test suite
- Used to formally verify parts of Arm processors
- Used in Arm's documentation

RISC-V formal ISA spec

- RISC-V's official formal spec
- Can boot Linux
- Publicly available in machine readable form
- Thoroughly tested

Arm specification (ASL language)




```
integer d = UInt(Rd);
integer n = UInt(Rn);
integer m = UInt(Rm);
integer datasize = if sf == '1' then 64 else 32;

bits(datasize) result;
bits(datasize) operand1 = X[n];
bits(datasize) operand2 = X[m];

(result, -) = AddWithCarry(operand1, operand2, PSTATE.C);

X[d] = result;
```

RISC-V specification (SAIL language)

```
mapping clause encdec = RTYPE(rs2, rs1, rd, RISCV_ADD)  <-> 0b0000000 @ rs2 @ rs1 @ 0b000 @ rd @ 0b0110011

function clause execute (RTYPE(rs2, rs1, rd, op)) = {
  let rs1_val = X(rs1);
  let rs2_val = X(rs2);
  let result : xlenbits = match op {
    RISCV_ADD  => rs1_val + rs2_val,
    RISCV_SLT  => EXTZ(bool_to_bits(rs1_val <_s rs2_val)),
    ...
  };
  X(rd) = result;
  RETIRE_SUCCESS
}
```

IA specification (WIP)

Current

Operation

DEST := DEST + SRC + CF;

Flags Affected

The OF, SF, ZF, AF, CF, and PF flags are set according to the result.

Proposed

```
let (result, CF, OF, AF) = AddWithCarry(src1, src2, FLAGS.CF);

FLAGS.OF = OF;
FLAGS.SF = result[operand_size-1];
FLAGS.ZF = if IsZero(result) then '1' else '0';
FLAGS.AF = AF;
FLAGS.PF = if ParityEven(result[0 +: 8]) then '1' else '0';
FLAGS.CF = CF;
```

Creating a formal ISA spec

What makes it easy?

What makes it hard?

Where is the research?

Helping you use the specification

How to write an ISA spec

Choose a language

- Goal: industry standard for ISA specification
- Arm's Architecture Specification Language

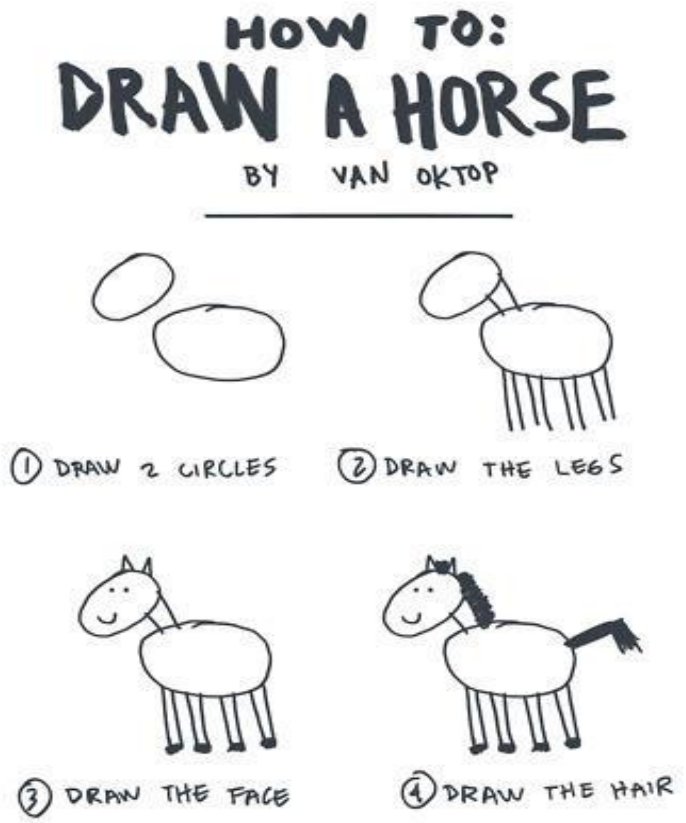
Write a specification

- Based on existing documentation, simulators, etc.

Write lots of tools

- Generate documentation from spec
- Support development, migration, checking, …

https://oktop.tumblr.com/post/15352780846



What makes it easy?

Done it before: experience + examples

Lots of testing collateral inside company

Recognized need and benefits

- Multiple internal attempts
- Successful external attempts (other architectures)

Motivating a company

https://alastairreid.github.io/mrs-at-scale/

Formally verify hardware

Validate simulators, firmware, microcode, …

Support development of new architecture extensions

Move faster

Improve coherence across the company

What makes it hard?

Lots of testing collateral inside company

- Need for extremely high quality
- Need efficient bug triage tools

Must not accidentally change the architecture

- But need to fix gaps, bugs, accidental ambiguities

Underspecification

- Instructions allow a range of behaviors

Where are the research challenges? Part 1

Security

- Can we detect security issues in new architecture extensions?
- Can we say anything useful about side channels?
  - e.g., Formally specify things that can be observed and what can / cannot influence them?
- Can we say anything useful about speculation, etc.?
- Can we verify security of hardware against the spec?

Many, many security-related questions

Q: Is my software secure?

Q: Is my architecture secure? (See https://www.lightbluetouchpaper.org/2022/07/22/formal-cheri/)

Q: What security properties does my architecture satisfy?

Q: Can we verify security of hardware wrt the spec?

Q: Are those properties useful to software developers?

Q: Does change X to the architecture break existing security properties?

Q: What to do about side channels?

Q: What to say about transient execution?

Q: What about the rest of the chip?

Where are the research challenges? Part 2

Enabling novel tools

https://alastairreid.github.io/uses-for-isa-specs/

- Compilers: verify peephole optimizations?
- Synthesis: generate peephole optimizations?
- Analysis: malware analysis of binaries
- Security: detect security vulnerabilities in binaries

How best to develop new uses?

Open source our specification

Open source our tools?

Collaborate with academia, etc.?

Blog posts?

Example code?

Tutorials?

Please talk to me about how I can help your research

Summary

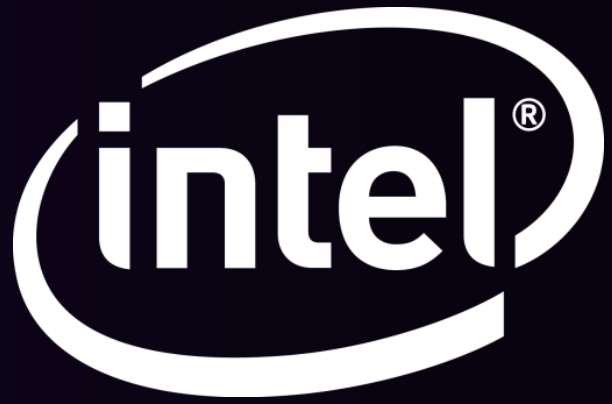
Starting to build an official formal spec of the Intel Architecture

Industry standard around how we write ISA specs?

Still early days (which makes this a great time to start talking to us about your needs!)

Please talk to me about how I can help your research

(Obvious omissions from this talk: formally verifying x86 processors; weak memory model)


intel®