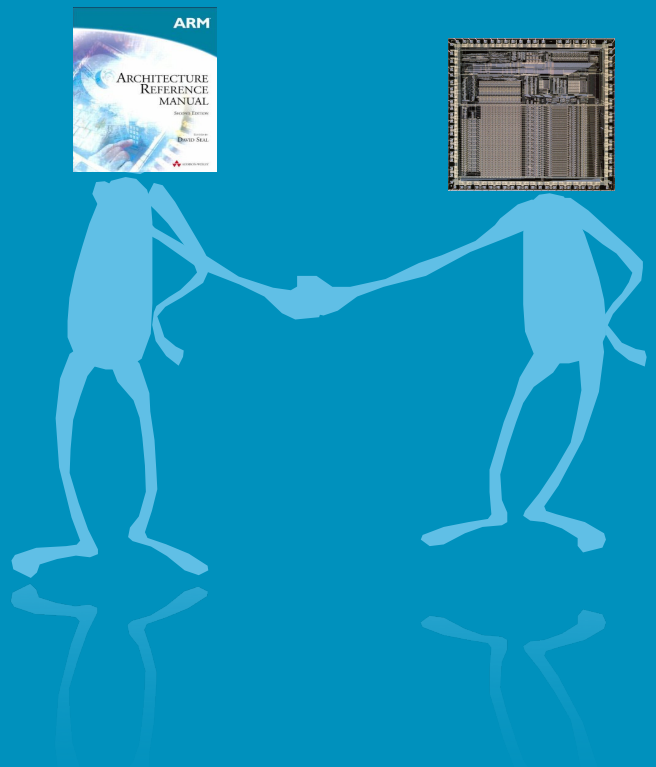
arm



How Can You Trust Formally Verified Software?

Alastair Reid

Arm Research
@alastair_d_reid

Software

US aviation authority: Boeing 787 bug could cause 'loss of control'

More trouble for Dreamliner as Federal Aviation Administration warns glitch in control unit causes generators to shut down if left powered on for 248 days

The Boeing 787 has four generator-control units that, if powered on at the same, could fail simultaneously and cause a complete electrical shutdown. Photograph: Elaine Thompson/AP

https://www.theguardian.com/business/2015/may/01/us-aviation-authority-boeing-787-dreamliner-bug-could-cause-loss-of-control

arm

Buffer over-read vulnerabilities

CVE-2014-0160 Detail

Modified

This vulnerability has been modified since it was last analyzed by the NVD. It is awaiting reanalysis which may result in f

Current Description

The (1) TLS and (2) DTLS implementations in OpenSSL 1.0.1 before 1.0.1g do not properly handle Heartbeat Extension packets, which allows remote attackers to obtain sensitive information from process memory via crafted packets that trigger a buffer over-read, as demonstrated by reading private keys, related to d1_both.c and t1_lib.c, aka the Heartbleed bug.

Source: MITRE **Last Modified:** 04/07/2014 + View Analysis Description

Buffer over-read vulnerabilities

CVE-2

Modified

This vulnerabilit

Current

The (1) TLS and

packets, which allows remote attacker

trigger a buffer over-read, as demonstrated by reading private

bug.

Source: MITRE **Last Modified:** 04/07/2014 View Analysis Description

CVE-2008-3803 A "logic error" in Cisco IOS 12.0 through 12.4, when a Multiprotocol Label Switching (MPLS) VPN with extended communities is configured, sometimes causes a corrupted route target (RT) to be used, which allows remote attackers to read traffic from other VPNs in opportunistic circumstances.

CVE-2008-0059 Race condition in NSXML in Foundation for Apple Mac OS X 10.4.11 allows context-dependent attackers to execute arbitrary code via a crafted XML file, related to "error handling logic."

CVE-2007-4613 SSL libraries in BEA WebLogic Server 6.1 Gold through SP7, 7.0 Gold through SP5 might allow remote attackers to obtain plaintext from an SSL stream via a man-in-the-middle ata and measures the elapsed time before an error response, a different vulnerability than CVE-200

CVE-2007-2989 The libike library in Sun Solaris 9 before 20070528 pointer, which allows remote attackers to cause a denial of service (in.iked daemon a source port different from 500. NOTE: this issue might overlap CVE-2006-2298.

CVE-2007-2444 Logic error in the SID/N Samba 3.0.23d through 3.0.25pre2 allows local users to gain temporary privileges and e unspecified vectors that cause the daemon to transition to the root user.

CVE-2007-0414 BEA WebLogic Se 7.0 through 7.0 SP6, 8.1 through 8.1 SP5, and 9.0 allows remote attackers to cause a denial of service (server requests that cause muxer threads to block when processing error pages.

CVE-2006-4394 A logic error in Log ndow in Apple Mac OS X 10.4 through 10.4.7, allows network accounts without GUIds to bypass service access controls and log into the system using loginwindow via unknown vectors.

CVE-2006-0381 A logic error in the IP fragment cache functionality in pf in FreeBSD 5.3, 5.4, and 6.0, and OpenBSD, when a 'scrub fragment crop' or 'scrub fragment drop-ovl' rule is being used, allows remote attackers to cause a denial of service (crash) via crafted packets that cause a packet fragment to be inserted twice.

CVE-2006-0380 A logic error in FreeBSD kernel 5.4-STABLE and 6.0 causes the kernel to calculate an incorrect buffer length, which causes more data to be copied to userland than intended, which could allow local users to read portions of kernel memory.

CVE-2005-0198 A logic error in the CRAM-MD5 code for the University of Washington IMAP (UW-IMAP) server, when Challenge-Response Authentication Mechanism with MD5 (CRAM-MD5) is enabled, does not properly enforce all the required conditions for successful authentication, which allows remote attackers to authenticate as arbitrary users.

Logic error vulnerabilities

Buffer over-read vulnerabilities

CVE-2

Modified

This vulnerability

Current

The (1) TLS and

packets, which allows remo

trigger a buffer over-read, a

bug.

Source: MITRE **Last Mo**

CVE-2008-3803 A "logic error" in Cisco IOS 12.0 through 12.4, when a Multiprotocol Label Switching (MPLS) VPN with extended communities is configured, sometimes causes a corrupted route target (RT) to be used, which allows remote attackers to read traffic from other VPNs in opportunistic circumstances.

CVE-2008-0059 Race condition in NSXML in Foundation for Apple Mac OS X 10.4.11 allows context-dependent attackers to execute arbitrary code via a crafted XML file, related to "error handling logic."

CVE-2007-4613 SSL libraries in BEA WebLogic Server 6.1 Gold through SP7, 7.0 Gold through SP

CVE-2007-2989 The libike library in Sun Solaris 9 before 20070528 attackers to obtain plaintext from an SSL stream via a man-in-the-mid before an error response, a different vulnerability than CVE-200 cause a denial of service (in.iked daemon might overlap CVE-2006-2298.

CVE-2007-2444 Logic error in the SID/ privileges and e

CVE-2007-0414 BEA WebLogic Se of service (serv

CVE-2006-4394 A lo

CVE-2006-

CVE-2006-03

CVE-2005-019

CVE-2017-9989 util/outputtxt.c in libming 0.4.8 mishandles memory allocation. A crafted input will lead to a remote denial of service (NULL pointer dereference) attack.

CVE-2017-9988 The readEncUInt30 function in util/read.c in libming 0.4.8 mishandles memory allocation. A crafted input will lead to a remote denial of service (NULL pointer dereference) attack against parser.c.

CVE-2017-9986 The intr function in sc denial of service (ove message queue head

CVE-2017-9985 The snd_msndmidi_ allows local users to by changing the value of a message fetch" vulnerability.

CVE-2017-9984 The snd_msnd_interrupt function in sound/isa/msnd/msnd_pinnacle.c in the Linux kernel through 4.11.7 allows local users to cause a denial of service (over-boundary access) or possibly have unspecified other impact by changing the value of a message queue head pointer between two kernel reads of that value, aka a "double fetch" vulnerability.

CVE-2017-9843 SAP NetWeaver AS ABAP 7.40 allows remote authenticated users with certain privileges to cause a denial of service (process crash) via vectors involving disp+work.exe, aka SAP Security Note 2406841.

CVE-2017-9680 In all Qualcomm products with Android releases from CAF using the Linux kernel, if a pointer argument coming from userspace is invalid, a driver may use an uninitialized structure to log an error message.

CVE-2017-9631 A Null Pointer Dereference issue was discovered in Schneider Electric Wonderware ArchestrA Logger, versions 2017.426.2307.1 and prior. The null pointer dereference vulnerability could allow an attacker to crash the logger causing a denial of service for logging and log-viewing (applications that use the Wonderware to run when the Wonderware ArchestrA Logger service is unavailable).

Null pointer dereference

Buffer ov

CVE-

Modifi

This vulnera

Current

The (1) TLS and

packets, which allows remc

trigger a buffer over-read, a

bug.

Source: MITRE **Last Mo**

CVE-2017-9953

CVE-2017-9935

CVE-2017-9789

CVE-2017-9685

CVE-2017-9684

CVE-2017-9682

CVE-2017-9602

CVE-2006-

CVE-2006-03

CVE-2005-019

CVE-2017-9988

CVE-2017-9986

CVE-2017-9985

CVE-2017-9984

CVE-2017-9843

CVE-2017-9680

CVE-2017-9631

There is an invalid free in Image::printIFDStructure that leads to a Segmentation fault in Exiv2 0.26. A crafted input will lead to a remote denial of service attack.

In LibTIFF 4.0.8, there is a heap-based buffer overflow in the t2p_write_pdf function in tools/tiff2pdf.c. This heap overflow could lead to different damages. For example, a crafted TIFF document can lead to an out-of-bounds read in TIFFCleanup, an invalid free in TIFFClose or t2p_free, memory corruption in t2p_readwrite_pdf_image, or a double free in t2p_free. Given these possibilities, it probably could cause arbitrary code execution.

When under stress, closing many connections, the HTTP/2 handling code in Apache httpd 2.4.26 would sometimes access memory after it has be... ulting in potentially erratic behaviour.

In all Qualcomm products... AF using the Linux kernel, a race condition in a WLAN driver can lead to a ...

In all Qualcomm pr... releases from CAF using the Linux kernel, a race condition in a USB driver can lead to a ... ree condition.

In all Qualcomm products with Android releases from CAF using the Linux kernel, a race condition in two KGSL driver functions can lead to a Use After Free condition.

KBVault Mysql Free Knowledge Base application package 0.16a comes with a FileExplorer/Explorer.aspx? id=/Uploads file-management component. An unauthenticated user can access the file upload and deletion functionality. Through this functionality, a user can upload an ASPX script to Uploads/Documents/ to run any arbitrary code.

lead to ...

The intr function... denial of service (ove... message queue head...

The snd_msndmidi_... allows local users to ... by changing the value of a message ... fetch" vulnerability.

The snd_msnd_interrupt function in sound/isa/msnd/msnd_pinn... local users to cause a denial of service (over-boundary access) or possi... changing the value of a message queue head pointer between two kernel rea... fetch" vulnerability.

SAP NetWeaver AS ABAP 7.40 allows remote authenticated users with certain privileges to ca... service (process crash) via vectors involving disp+work.exe, aka SAP Security Note 2406841.

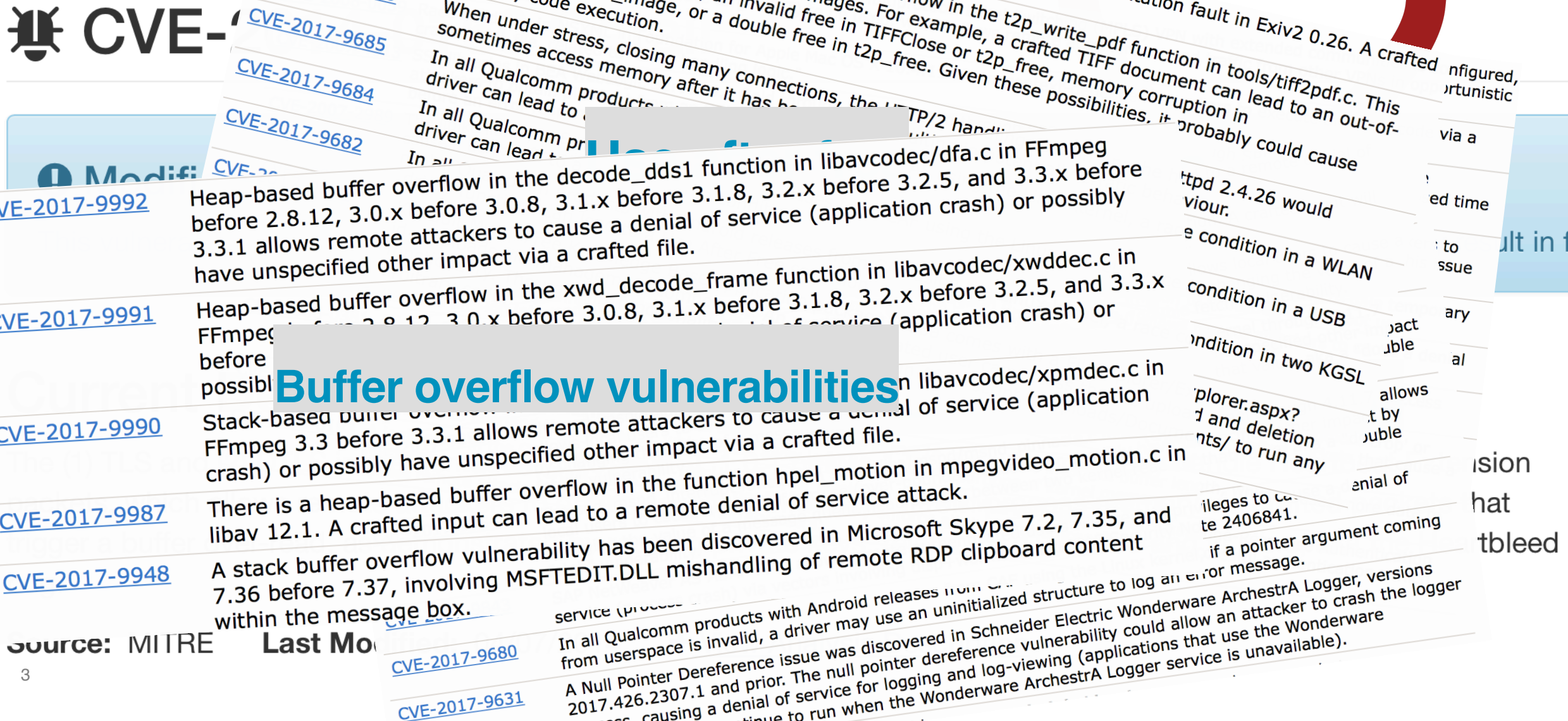
In all Qualcomm products with Android releases from CAF using the Linux kernel, if a pointer argument coming from userspace is invalid, a driver may use an uninitialized structure to log an error message.

A Null Pointer Dereference issue was discovered in Schneider Electric Wonderware ArchestrA Logger, versions 2017.426.2307.1 and prior. The null pointer dereference vulnerability could allow an attacker to crash the logger ... causing a denial of service for logging and log-viewing (applications that use the Wonderware ... to run when the Wonderware ArchestrA Logger service is unavailable).

Use after free

Null pointer

Buffer overflow vulnerabilities



CVE-2017-9992	Heap-based buffer overflow in the decode_dds1 function in libavcodec/dfa.c in FFmpeg before 2.8.12, 3.0.x before 3.0.8, 3.1.x before 3.1.8, 3.2.x before 3.2.5, and 3.3.x before 3.3.1 allows remote attackers to cause a denial of service (application crash) or possibly have unspecified other impact via a crafted file.
CVE-2017-9991	Heap-based buffer overflow in the xwd_decode_frame function in libavcodec/xwddec.c in FFmpeg before 2.8.12, 3.0.x before 3.0.8, 3.1.x before 3.1.8, 3.2.x before 3.2.5, and 3.3.x before 3.3.1 allows remote attackers to cause a denial of service (application crash) or possibly have unspecified other impact via a crafted file.
CVE-2017-9990	Stack-based buffer overflow in libavcodec/xpmdec.c in FFmpeg 3.3 before 3.3.1 allows remote attackers to cause a denial of service (application crash) or possibly have unspecified other impact via a crafted file.
CVE-2017-9987	There is a heap-based buffer overflow in the function hpel_motion in mpegvideo_motion.c in libav 12.1. A crafted input can lead to a remote denial of service attack.
CVE-2017-9948	A stack buffer overflow vulnerability has been discovered in Microsoft Skype 7.2, 7.35, and 7.36 before 7.37, involving MSFTEDIT.DLL mishandling of remote RDP clipboard content within the message box.
CVE-2017-9680	In all Qualcomm products with Android releases from CA... service (process... from userspace is invalid, a driver may use an uninitialized structure to log an error message.
CVE-2017-9631	A Null Pointer Dereference issue was discovered in Schneider Electric Wonderware ArchestrA Logger, versions 2017.426.2307.1 and prior. The null pointer dereference vulnerability could allow an attacker to crash the logger service, causing a denial of service for logging and log-viewing (applications that use the Wonderware ... continue to run when the Wonderware ArchestrA Logger service is unavailable).

Formal verification

Of libraries and apps

Of compilers

COMPCERT

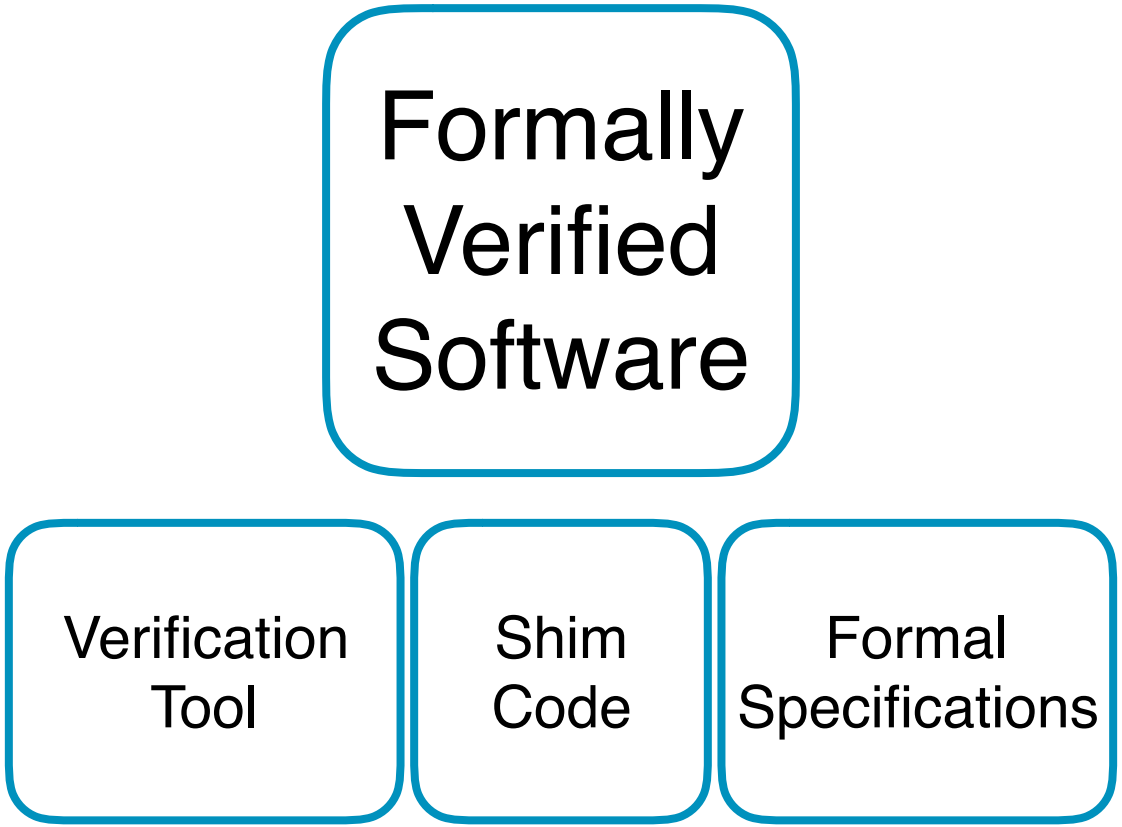
Of operating systems

arm

Fonseca et al., An Empirical Study on the Correctness of Formally Verified Distributed Systems, Eurosys ‘17

Formally Verified Software

Fonseca et al., An Empirical Study on the Correctness of Formally Verified Distributed Systems, Eurosys ‘17

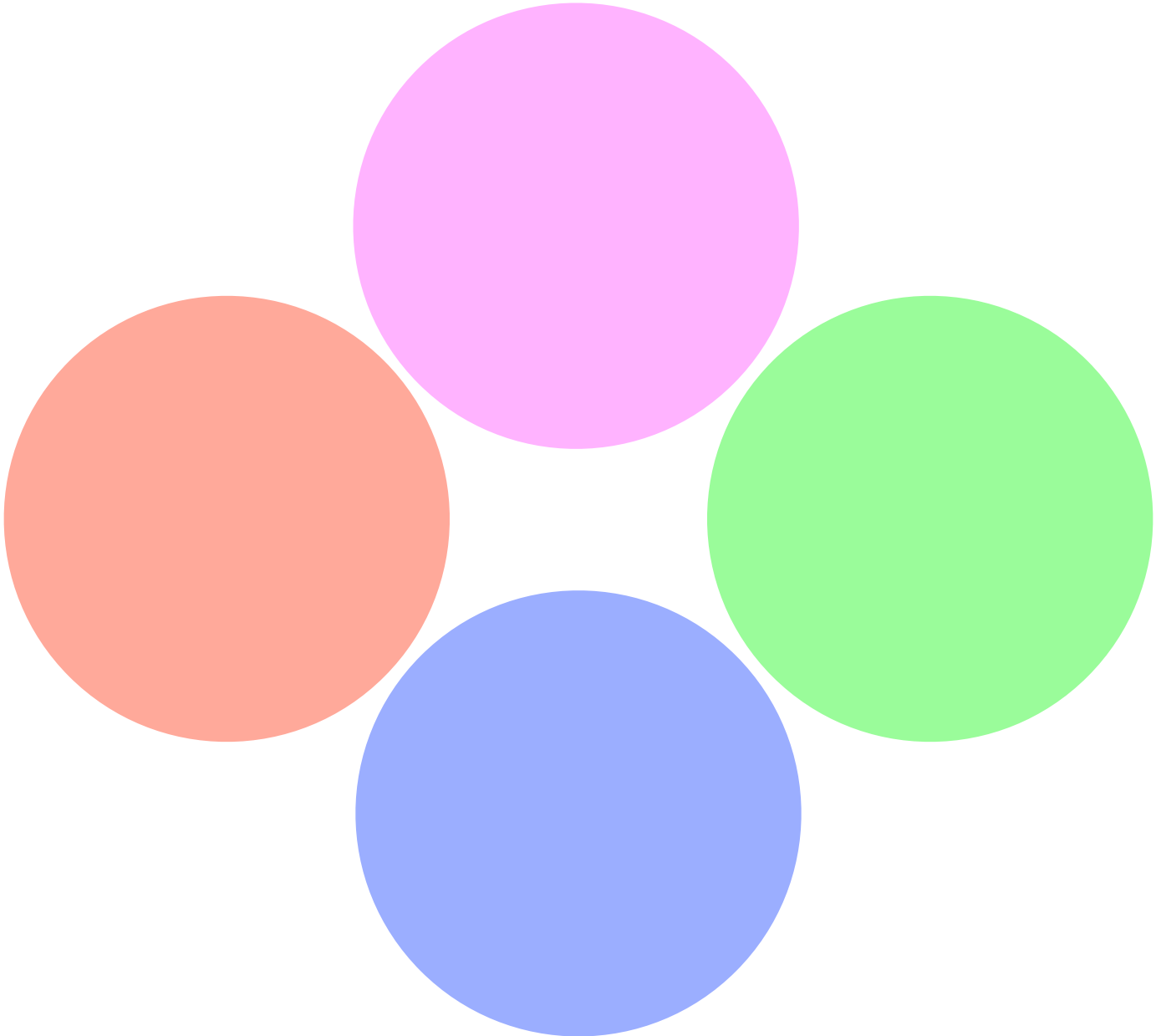


arm

Takeaway #1: 3 key questions to ask

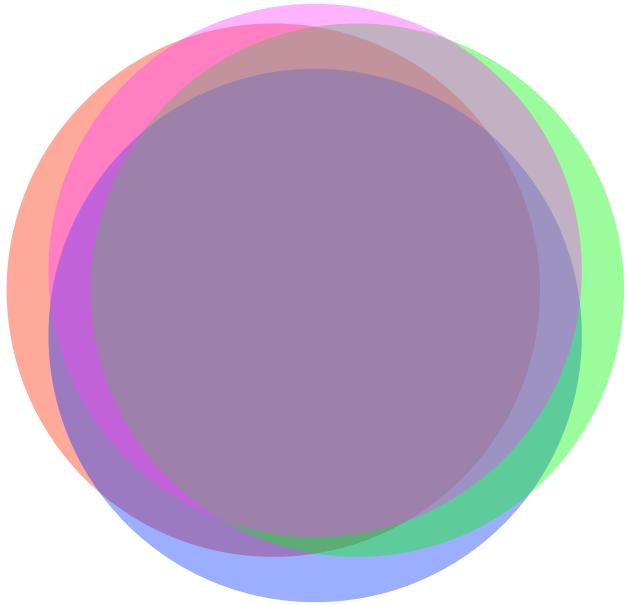
1. What specifications does your proof rely on?
2. Why do you trust those specifications?
3. Does anybody else use these specifications?

Takeaway #2: Specifications must have multiple uses



arm

Takeaway #2: Specifications must have multiple uses



How can you trust formally verified software?

How can you trust formally verified software?

- Specifications are part of the TCB
- 3 key questions
- Specifications must have multiple users

How can you trust formal specifications?

- Testing specifications
- Verifying processors
- Verifying specifications

How can you trust formally verified software?

arm

Creating trustworthy specifications

arm

The state of most processor specifications

Large (1000s of pages)

Broad (10+ years of implementations, multiple manufacturers)

Complex (exceptions, weak memory, …)

Informal (mostly English prose)

We are all just learning how to (retrospectively) formalize specifications

Arm Processor Specifications

A-class (phones, tablets, servers, …)

6,000 pages
40,000 line formal specification

Instructions (32/64-bit)
Exceptions / Interrupts
Memory protection
Page tables
Multiple privilege levels
System control registers
Debug / trace

M-class (microcontrollers, IoT)

1,200 pages
15,000 line formal specification

Instructions (32-bit)
Exceptions / Interrupts
Memory protection
~~Page tables~~
Multiple privilege levels
System control registers
Debug / trace

arm

English prose

R_{JRJC}

Exit from lockup is by any of the following:

- A Cold reset.
- A Warm reset.
- Entry to Debug state.
- Preemption by a higher priority exception.

R_{VGNW}

Entry to lockup from an exception causes:

- Any Fault Status Registers associated with the exception to be updated.
- No update to the exception state, pending or active.
- The PC to be set to `0xEFFFFFFE`.
- EPSR.IT to be become UNKNOWN.

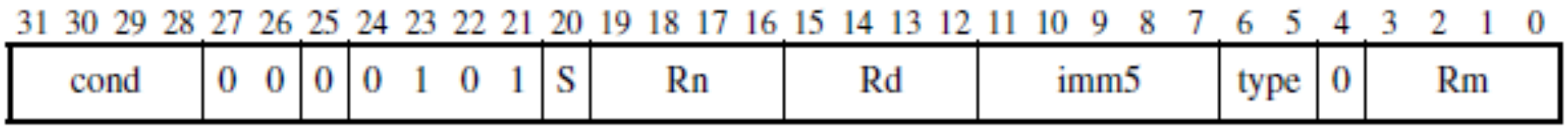
In addition, HFSR.FORCED is not set to 1.

arm

Pseudocode

Encoding A1 ARMv4*, ARMv5T*, ARMv6*, ARMv7

```
ADC{S}<c> <Rd>,<Rn>,<Rm>{,<shift>}
```



31 30 29 28	27 26	25	24 23 22 21	20	19 18 17 16	15 14 13 12	11 10 9 8 7	6 5	4	3 2 1 0
cond	0 0	0	0 1 0 1	S	Rn	Rd	imm5	type	0	Rm

```
if Rd == '1111' && S == '1' then SEE SUBS PC, LR and related instructions;
d = UInt(Rd);  n = UInt(Rn);  m = UInt(Rm);  setflags = (S == '1');
(shift_t, shift_n) = DecodeImmShift(type, imm5);

if ConditionPassed() then
    EncodingSpecificOperations();
    shifted = Shift(R[m], shift_t, shift_n, APSR.C);
    (result, carry, overflow) = AddWithCarry(R[n], shifted, APSR.C);
    if d == 15 then          // Can only occur for ARM encoding
        ALUWritePC(result);  // setflags is always FALSE here
    else
        R[d] = result;
        if setflags then
            APSR.N = result<31>;
            APSR.Z = IsZeroBit(result);
            APSR.C = carry;
            APSR.V = overflow;
```


arm

Arm Architecture Specification Language (ASL)

Indentation-based syntax

Imperative

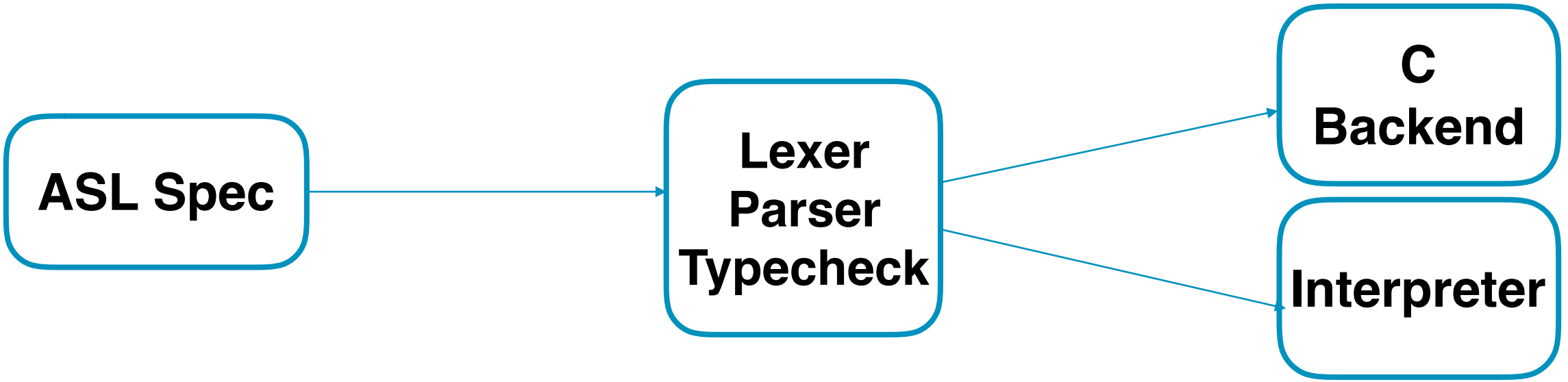
First-order

Strongly typed (type inference, polymorphism, dependent types)

- Bit-vectors
- Unbounded integers
- Infinite precision reals
- Arrays, Records, Enumerations

Exceptions

arm



arm

Architectural Conformance Suite

Processor architectural compliance sign-off

Large

- v8-A 11,000 test programs, > 2 billion instructions
- v8-M 3,500 test programs, > 250 million instructions

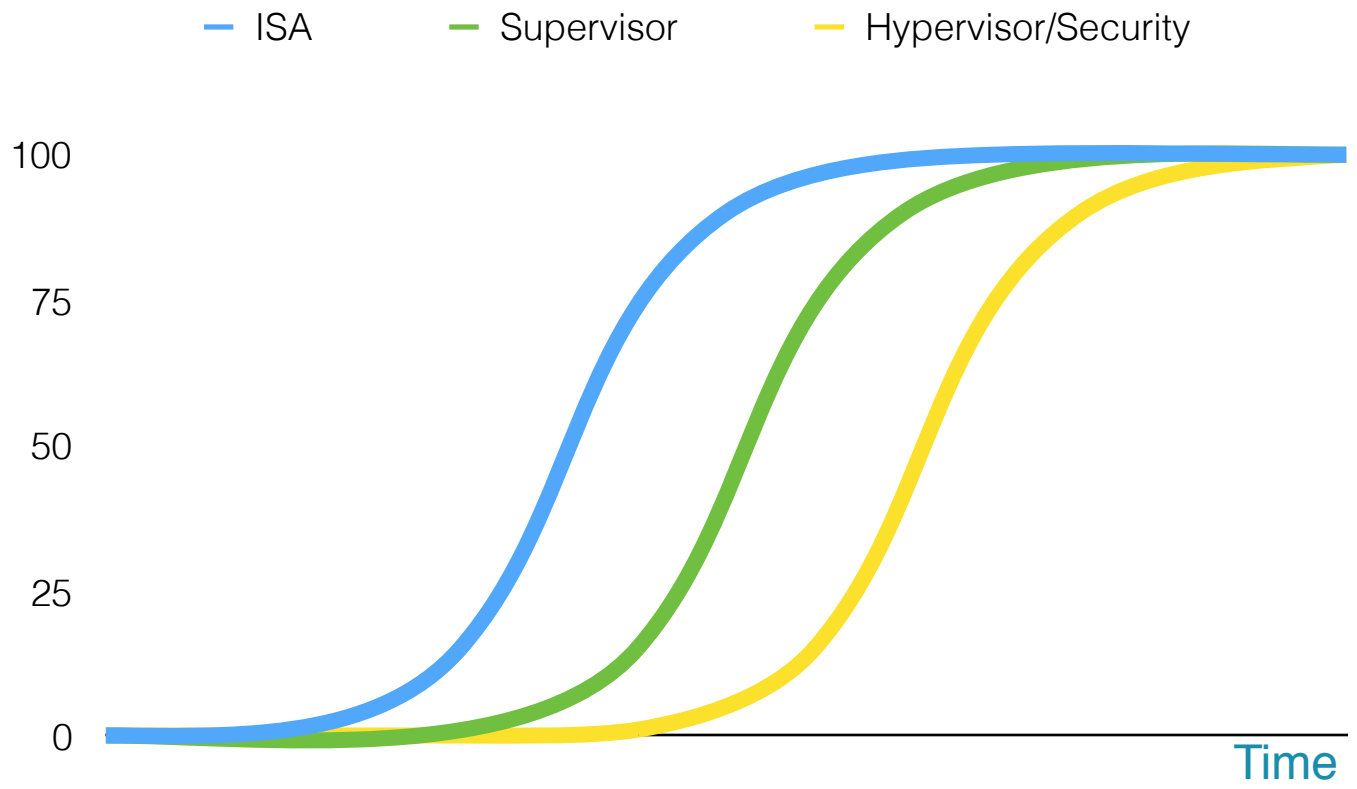
Thorough

- Tests dark corners of specification

arm

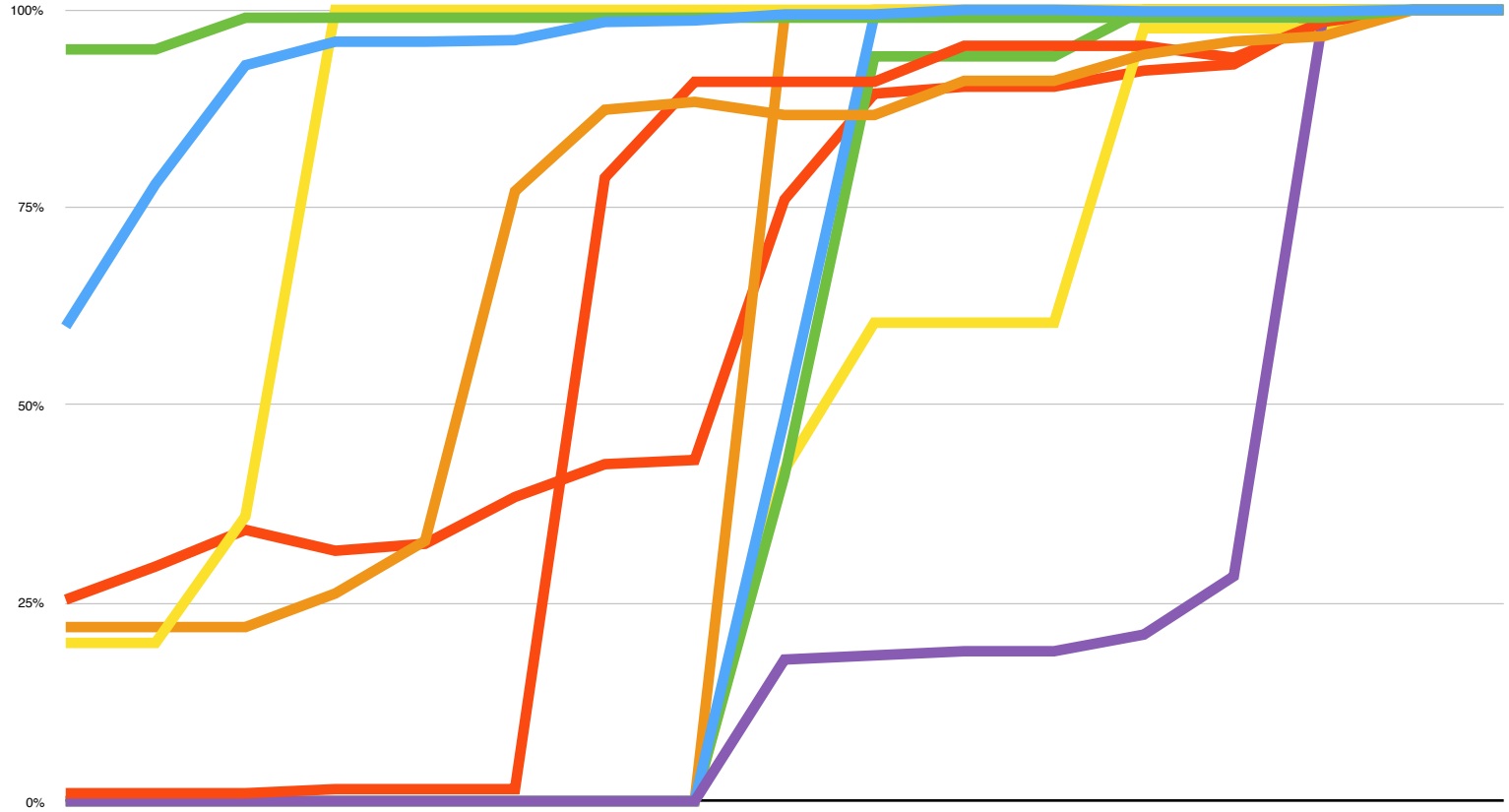
Testing Pass Rate

(Artists Impression)



arm

v8-M



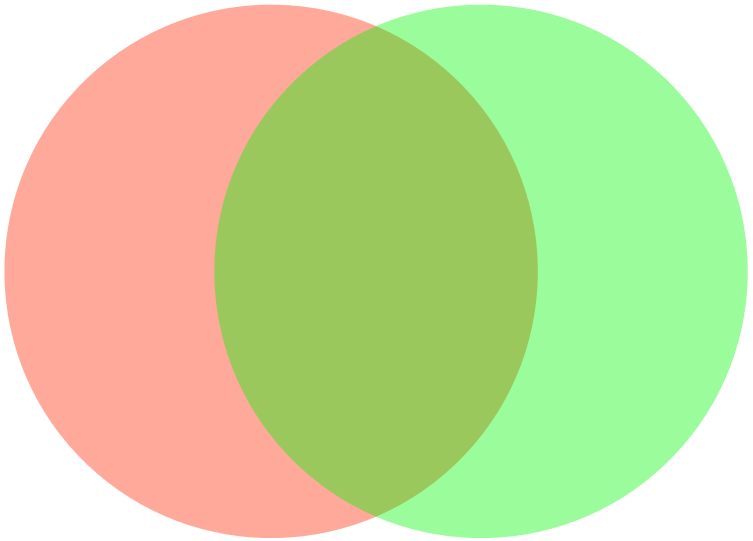
arm

Measuring architecture coverage of tests

Untested: op1*op2 == -3.0, FPCR.RND=-Inf


```
bits(N) FPRSqrtStepFused(bits(N) op1, bits(N) op2)
    assert N IN {32, 64};
    bits(N) result;
    op1 = FPNeg(op1);    // per FMSUB/FMLS
    (type1,sign1,value1) = FPUnpack(op1, FPCR);
    (type2,sign2,value2) = FPUnpack(op2, FPCR);
    (done,result) = FPProcessNaNs(type1, type2, op1, op2, FPCR);
    if !done then
        inf1 = (type1 == FPType_Infinity);
        inf2 = (type2 == FPType_Infinity);
        zero1 = (type1 == FPType_Zero);
        zero2 = (type2 == FPType_Zero);
        if (inf1 && zero2) || (zero1 && inf2) then
            result = FPOnePointFive('0');
        elsif inf1 || inf2 then
            result = FPInfinity(sign1 EOR sign2, N);
        else
            // Fully fused multiply-add and halve
            result_value = (3.0 + (value1 * value2)) / 2.0;
            if result_value == 0.0 then
                // Sign of exact zero result depends on rounding mode
                sign = if FPCRRounding() == FPRounding_NEGINF then '1' else '0';
                result = FPZero(sign, N);
            else
                result = FPRound(result_value, FPCRRounding());
    return result;
```


arm



arm

“End to End Verification of ARM processors with ISA Formal,” CAV 2016

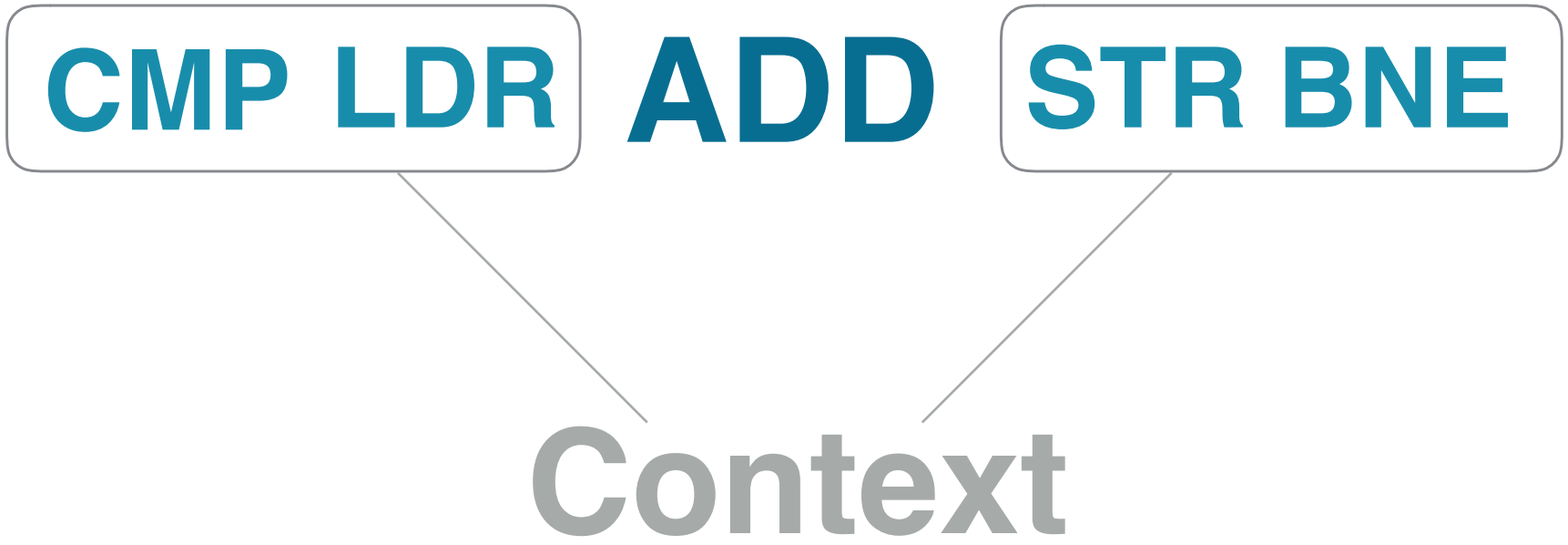
Formal verification of processors

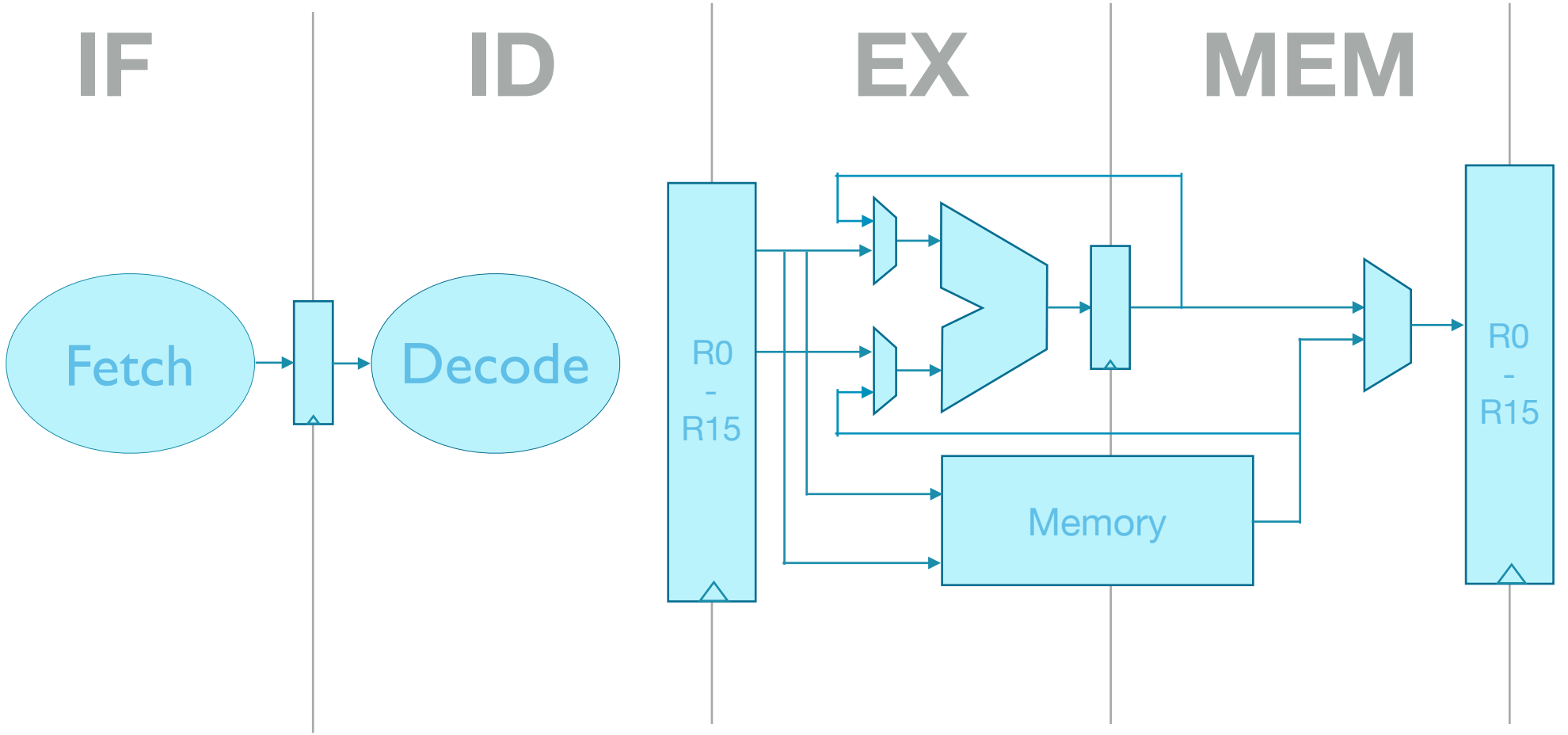
arm

Checking an instruction

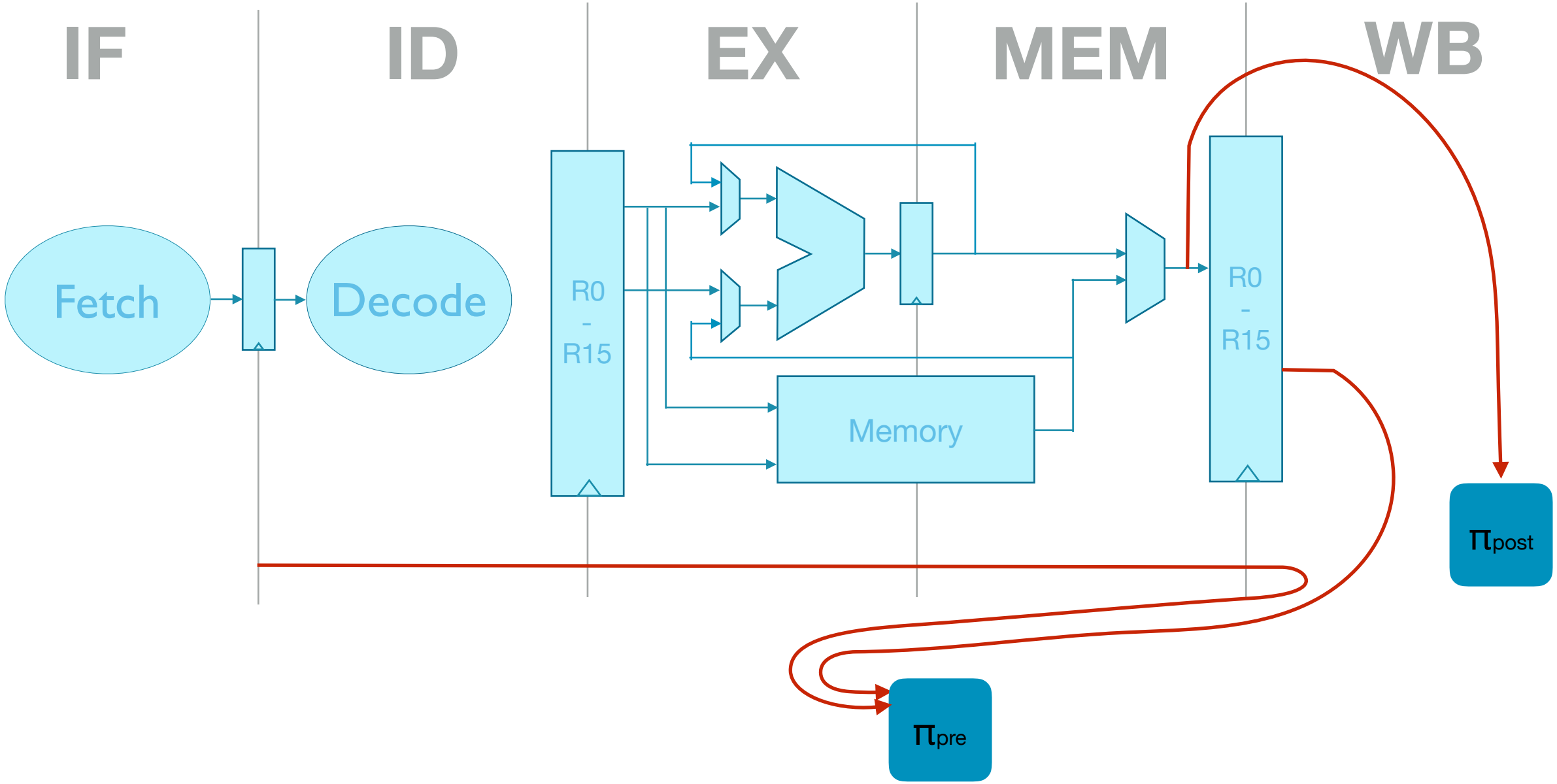
ADD

Checking an instruction

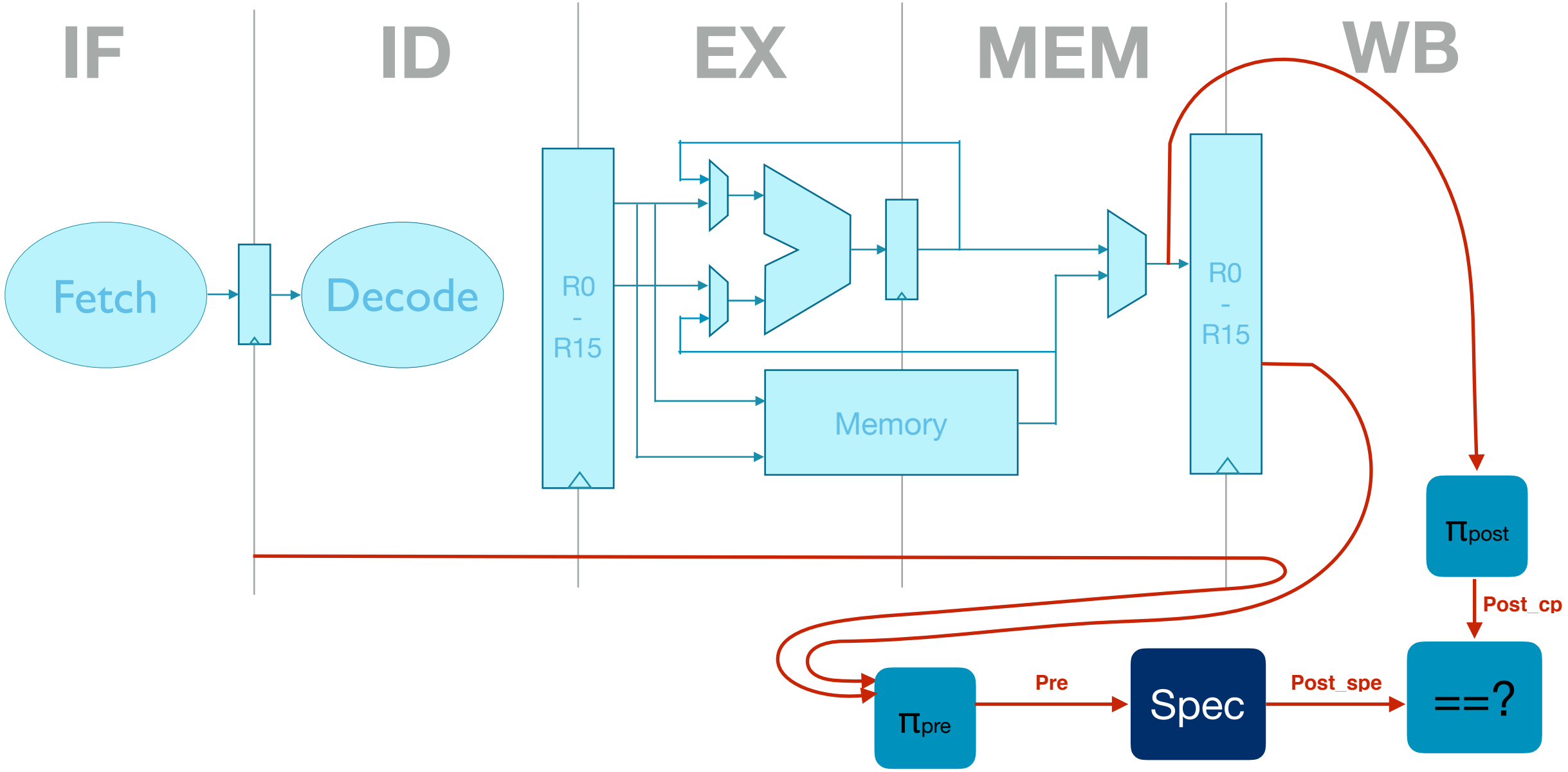




arm



arm



arm

Specialize
Monomorphize
Constant Propagation
Width Analysis
Exception Handling
…

Arm CPUs verified with ISA-Formal

A-class	R-class	M-class
Cortex-A53	Cortex-R52	Cortex-M4
Cortex-A32	Next generation	Cortex-M7
Cortex-A35		Cortex-M33
Cortex-A55		Next generation
Next generation		

Cambridge Projects

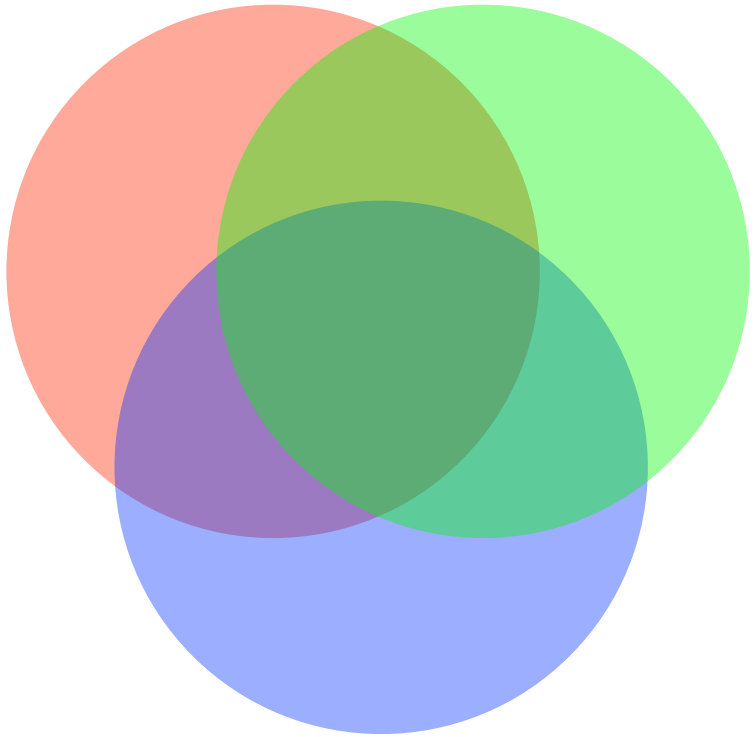
Rolling out globally to other design centres

Sophia, France - Cortex-A75 (partial)

Austin, USA - TBA

Chandler, USA - TBA

arm



Formal validation of specifications

arm

Suppose…

Last year: audited all accesses to privileged registers

- Specification: Added missing privilege checks
- Testsuite: Added new tests to test every privilege check
- Formal testbench: Verify every check

This year: add new instruction but accidentally omit privilege check

How many tests in the test suite will fail on new specification?

Executable Specification

Defines what *is* allowed

Animation → Check spec matches expectation

Testable → Compare spec against implementation

Executable Specification

Defines what *is* allowed

- Animation → Check spec matches expectation
- Testable → Compare spec against implementation

Does *not* define what is *not* allowed

- e.g., Impossible states, impossible actions/transitions, security properties
- No redundancy
- Problem when extending specification

Creating a specification of disallowed behaviour

Where to get a list of disallowed behaviour?

How to formalise this list?

How to formally validate specification against spec of disallowed behaviour?

Rule JRJC

Exit from lockup is by any of the following:

- A Cold reset.
- A Warm reset.
- Entry to Debug state.
- Preemption by a higher priority processor exception.

Rule R

State Change X by any of the following:

- Event A
- Event B
- State Change C
- Event D

Rule R

State Change X by any of the following:

- Event A
- Event B
- State Change C
- Event D

And cannot happen any other way

Rule R

State Change X by any of the following:

- Event A
- Event B
- State Change C
- Event D

And cannot happen any other way

Rule R: $X \rightarrow A \vee B \vee C \vee D$

arm

State Change X	Exit from lockup	Fell(LockedUp)
Event A	A Cold reset	Called(TakeColdReset)
Event B	A Warm reset	Called(TakeReset)
State Change C	Entry to Debug state	Rose(Halted)
Event D	Preemption by a higher priority processor exception	Called(ExceptionEntry)

arm

Rule JRJC

Exit from lockup is by any of the following:

- A Cold reset.
- A Warm reset.
- Entry to Debug state.
- Preemption by a higher priority processor exception.

Rule JRJC

Exit from lockup is by any of the following:

- A Cold reset.
- A Warm reset.
- Entry to Debug state.
- Preemption by a higher priority processor exception.

$$\begin{aligned} \text{Fell(LockedUp)} \rightarrow\ & \text{Called(TakeColdReset)} \\ & \vee\ \text{Called(TakeReset)} \\ & \vee\ \text{Rose(Halted)} \\ & \vee\ \text{Called(ExceptionEntry)} \end{aligned}$$

Rule JRJC

Exit from lockup is by any of the following:

- A Cold reset.
- A Warm reset.
- Entry to Debug state.
- Preemption by a higher priority processor exception.

$$\text{Fell(LockedUp)} \rightarrow \text{Called(TakeColdReset)} \vee \text{Called(TakeReset)} \vee \text{Rose(Halted)} \vee \text{Called(ExceptionEntry)}$$

```
__Called_TakeColdReset      = FALSE;    assert((__Past_LockedUp > LockedUp)
__Called_TakeReset          = FALSE;             ==>
__Called_TakeExceptionEntry = FALSE;              (  __Called_TakeColdReset
__Past_LockedUp = LockedUp;                       || __Called_TakeReset
__Past_Halted   = Halted;                         || __Past_Halted < Halted
                                                  || __Called_ExceptionEntry));
```

Rule VGNW

Entry to lockup from an exception causes

- Any Fault Status Registers associated with the exception to be updated.
- No update to the exception state, pending or active.
- The PC to be set to 0xEFFFFFFE.
- EPSR.IT to become UNKNOWN.

In addition, HFSR.FORCED is not set to 1.

Rule VGNW

Entry to lockup from an exception causes

- Any Fault Status Registers associated with the exception to be updated.

Out of date
- No update to the exception state, pending or active.

Misleading
- The PC to be set to 0xEFFFFFFE.

Untestable
- EPSR.IT to become UNKNOWN.

Ambiguous

In addition, HFSR.FORCED is not set to 1.

arm

Arm Specification Language → SMT

- Arithmetic operations
- Boolean operations
- Bit Vectors
- Arrays
- Functions
- Local Variables
- Statements
  - Assignments
  - If-statements
  - Loops
  - Exceptions

- Arithmetic operations
- Boolean operations
- Bit Vectors
- Arrays
- ~~Functions~~
- ~~Local Variables~~
- ~~Statements~~
  - ~~Assignments~~
  - ~~If-statements~~
  - ~~Loops~~
  - ~~Exceptions~~

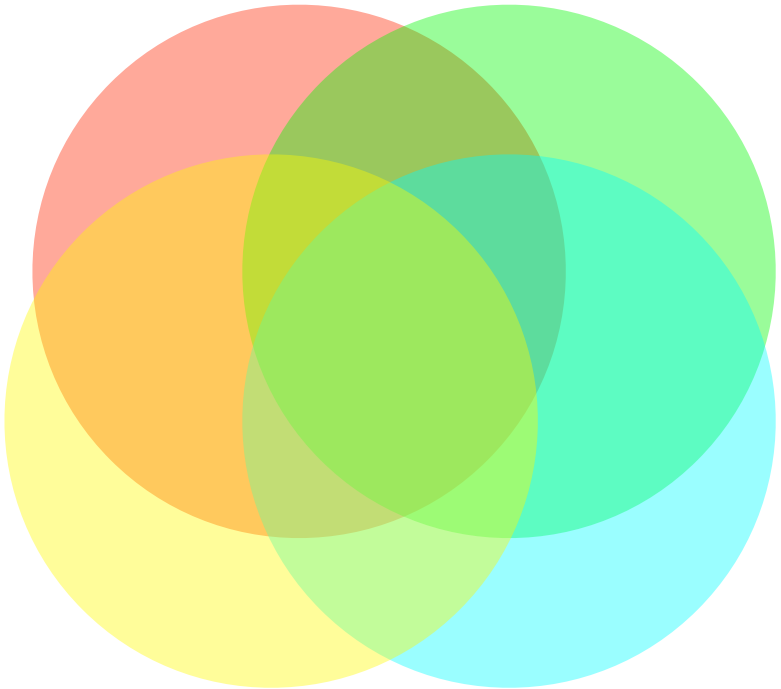
arm

Formally Validating Specifications

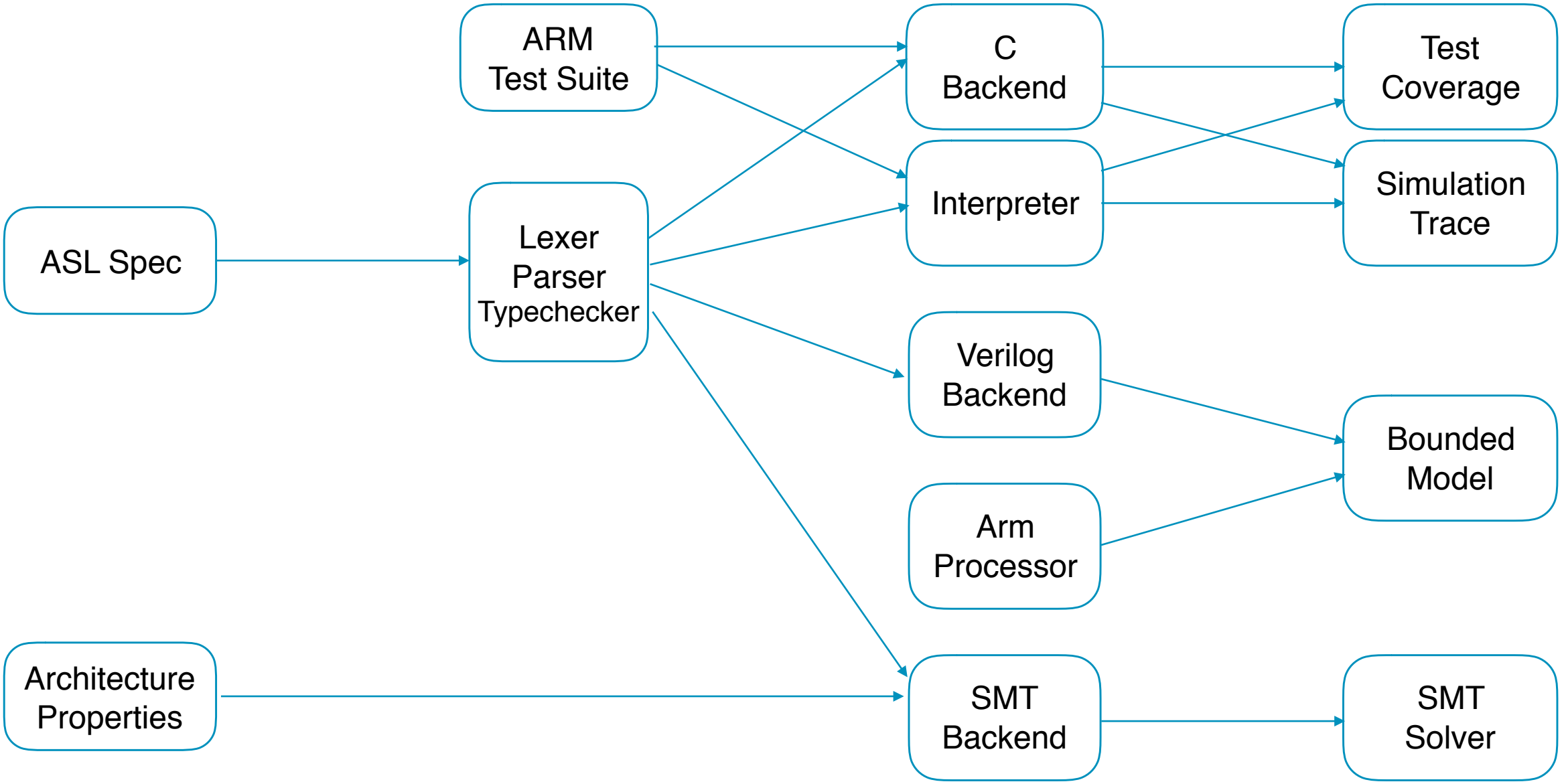
arm

Formally Validating Specifications

arm



arm



arm

Public release of machine readable Arm specification

Enable formal verification of software and tools

Releases

- April 2017: v8.2
- July 2017: v8.3

Working with Cambridge University REMS group to convert to SAIL

- Backends for HOL, OCaml, Memory model, (hopefully Coq too)

Tools: https://github.com/alastairreid/mra_tools

Talk to me about how I can help you use it

Potential uses of processor specifications

Verifying compilers

Verifying OS page table / interrupt / boot code

Verifying processor pipelines

Verification and discovery of peephole optimizations

Automatic generation of binary translators

Automatic generation of test cases

Decompilation of binaries

Abstract interpretation of binaries

etc.

How can you trust formally verified software?

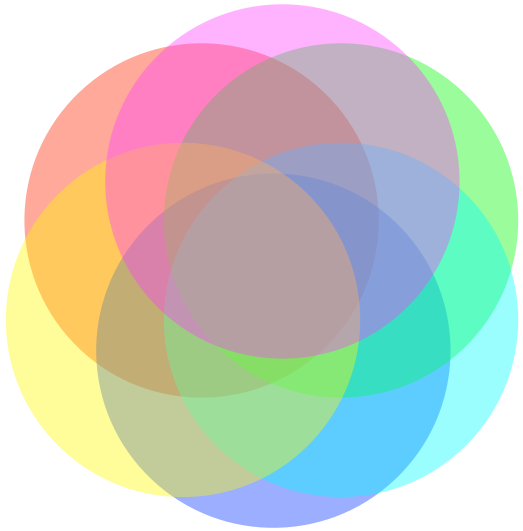
arm

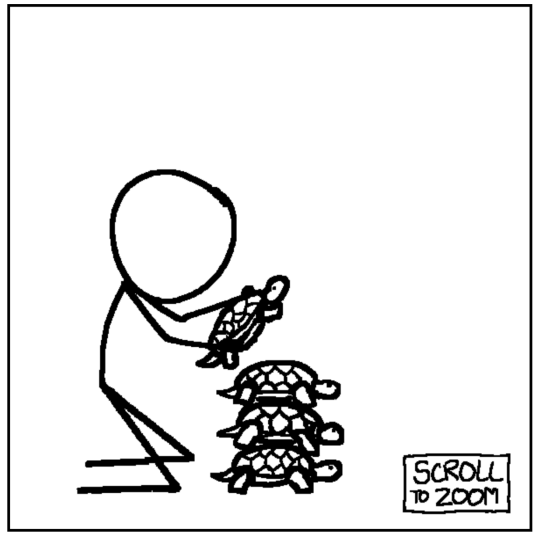
How can you trust formal specifications?

Test the specifications you depend on

Ensure specifications have multiple uses

Create meta-specifications





https://xkcd.com/1416/

Hiring in Security and Correctness group — contact me

arm

Thank You!
Danke!
Merci!
谢谢!
ありがとう!
Gracias!
Kiitos!

@alastair_d_reid

arm

“Trustworthy Specifications of the ARM v8-A and v8-M architecture,” FMCAD 2016
“End to End Verification of ARM processors with ISA Formal,” CAV 2016
“Who guards the guards? Formal Validation of ARM v8-M Specifications,” OOPSLA 2017