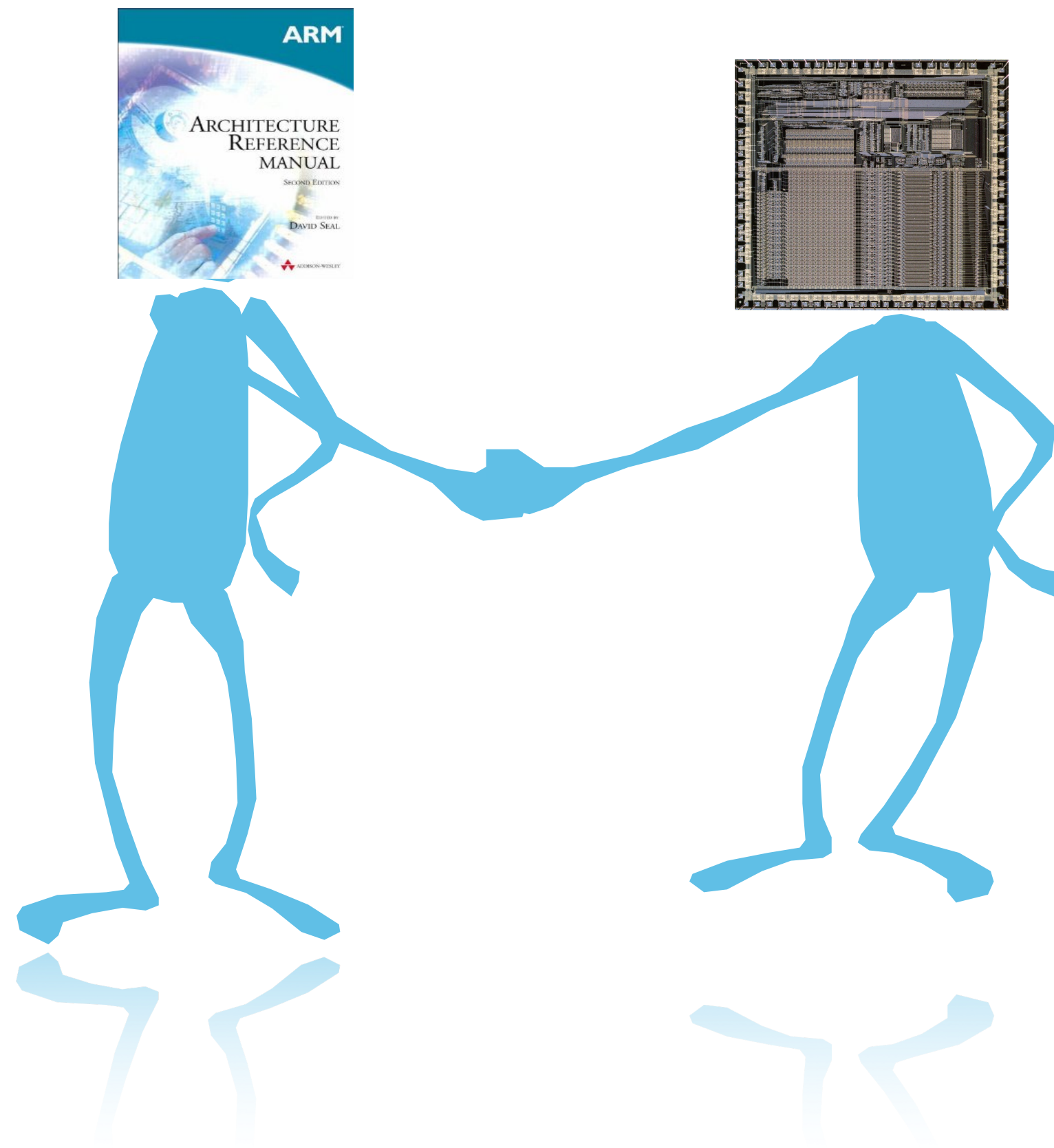


Trustworthy Specifications of ARM[®] System Level Architecture



Alastair Reid
alastair.reid@arm.com
[@alastair_d_reid](#)

Application

Library

OS

Compiler

Processor

Qualities of a Specification

Applicability

Scope

Trustworthiness

Applicability

v1 (1985)

...

v6 (1997)

v7 (2005)

v8.0 (2013)

v8.1 (2015)

v8.2 (2016)

A-class (phones/tablets/servers)

R-class (real-time, lock-step support)

M-class (microcontroller)

Scope

Compiler targeted instructions?

User-level instructions?

User+Supervisor?

User+Supervisor+Hypervisor+Secure Monitor?

ISA Specification - ASL

Encoding T3 ARMv7-M

MOV{S}<c>.W <Rd>, <Rm>

Opcode



15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0															
1	1	1	0	1	0	1	0	0	1	0	S	1	1	1	1	(0)	0	0	0												Rd															Rm

Decode



```
d = UInt(Rd); m = UInt(Rm); setflags = (S == '1');
if setflags && (d IN {13,15} || m IN {13,15}) then UNPREDICTABLE;
if !setflags && (d == 15 || m == 15 || (d == 13 && m == 13)) then UNPREDICTABLE;
```

Execute



```
if ConditionPassed() then
  EncodingSpecificOperations();
  result = R[m];
  if d == 15 then
    ALUWritePC(result); // setflags is always FALSE here
  else
    R[d] = result;
    if setflags then
      APSR.N = result<31>;
      APSR.Z = IsZeroBit(result);
      // APSR.C unchanged
      // APSR.V unchanged
```

System Architecture Specification

```
AArch64.DataAbort(bits(64) vaddress, FaultRecord fault)

    route_to_e13 = HaveEL(EL3) && SCR_EL3.EA == '1' && IsExternalAbort(fault);
    route_to_e12 = (HaveEL(EL2) && !IsSecure() && PSTATE.EL IN {EL0,EL1} &&
                    (HCR_EL2.TGE == '1' || IsSecondStage(fault)));

    bits(64) preferred_exception_return = ThisInstrAddr();
    vect_offset = 0x0;

    exception = AArch64.AbortSyndrome(Exception_DataAbort, fault, vaddress);

    if PSTATE.EL == EL3 || route_to_e13 then
        AArch64.TakeException(EL3, exception, preferred_exception_return, vect_offset);
    elsif PSTATE.EL == EL2 || route_to_e12 then
        AArch64.TakeException(EL2, exception, preferred_exception_return, vect_offset);
    else
        AArch64.TakeException(EL1, exception, preferred_exception_return, vect_offset);
```

ARM Spec (lines of code)

	v8-A	v8-M
Instructions Int/FP/SIMD	26,000	6,000
Exceptions	4,000	3,000
Memory	3,000	1,000
Debug	3,000	1,000
Misc	5,500	2,000
(Test support)	1,500	2,000
Total	43,000	15,000

System Register Spec

	v8-A	v8-M
Registers	586	186
Fields	3951	622
Constant	985	177
Reserved	940	208
Impl. Defined	70	10
Passive	1888	165
Active	68	62
Operations	112	10

Trustworthiness

Trustworthiness

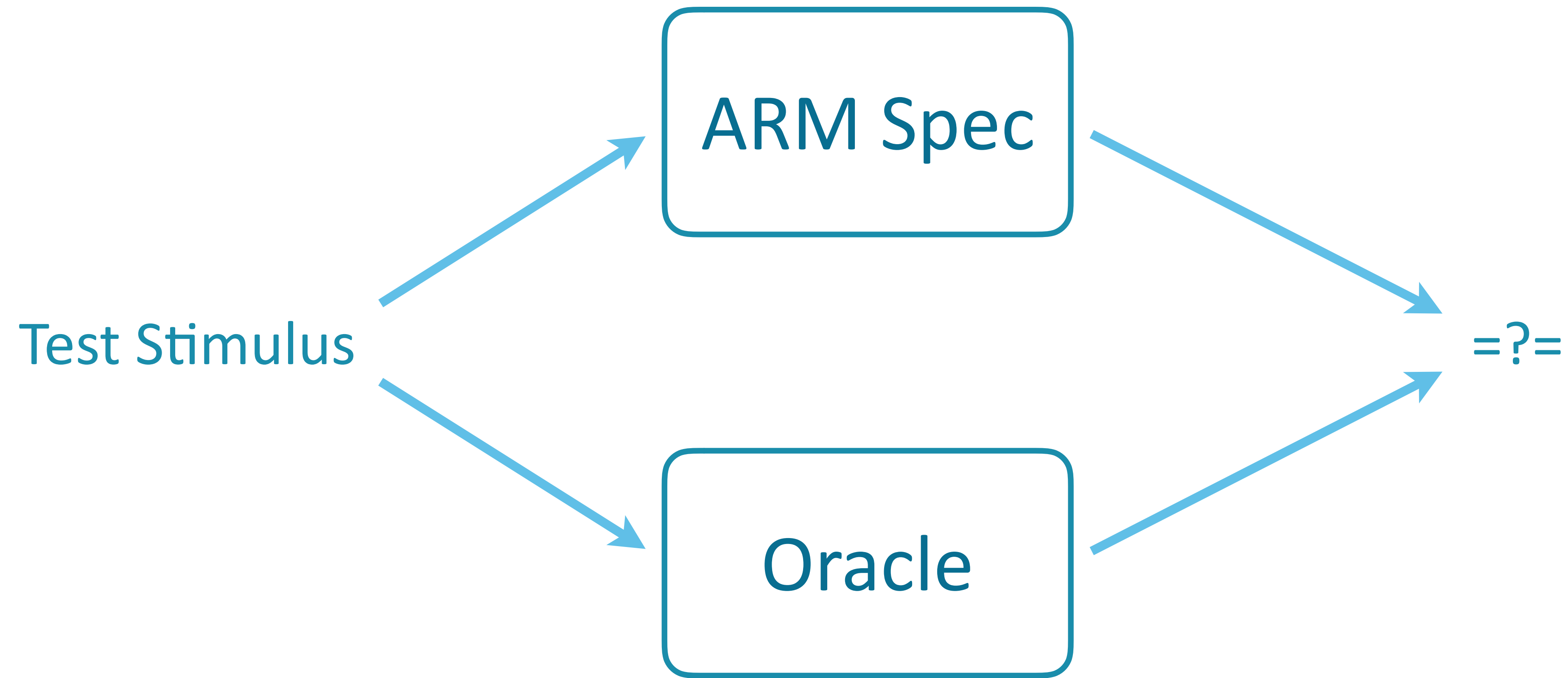
ARM's specification is correct *by definition*

Trustworthiness

~~ARM's specification is correct *by definition*~~

Trustworthiness

Does the specification match the behaviour
of all ARM processors?

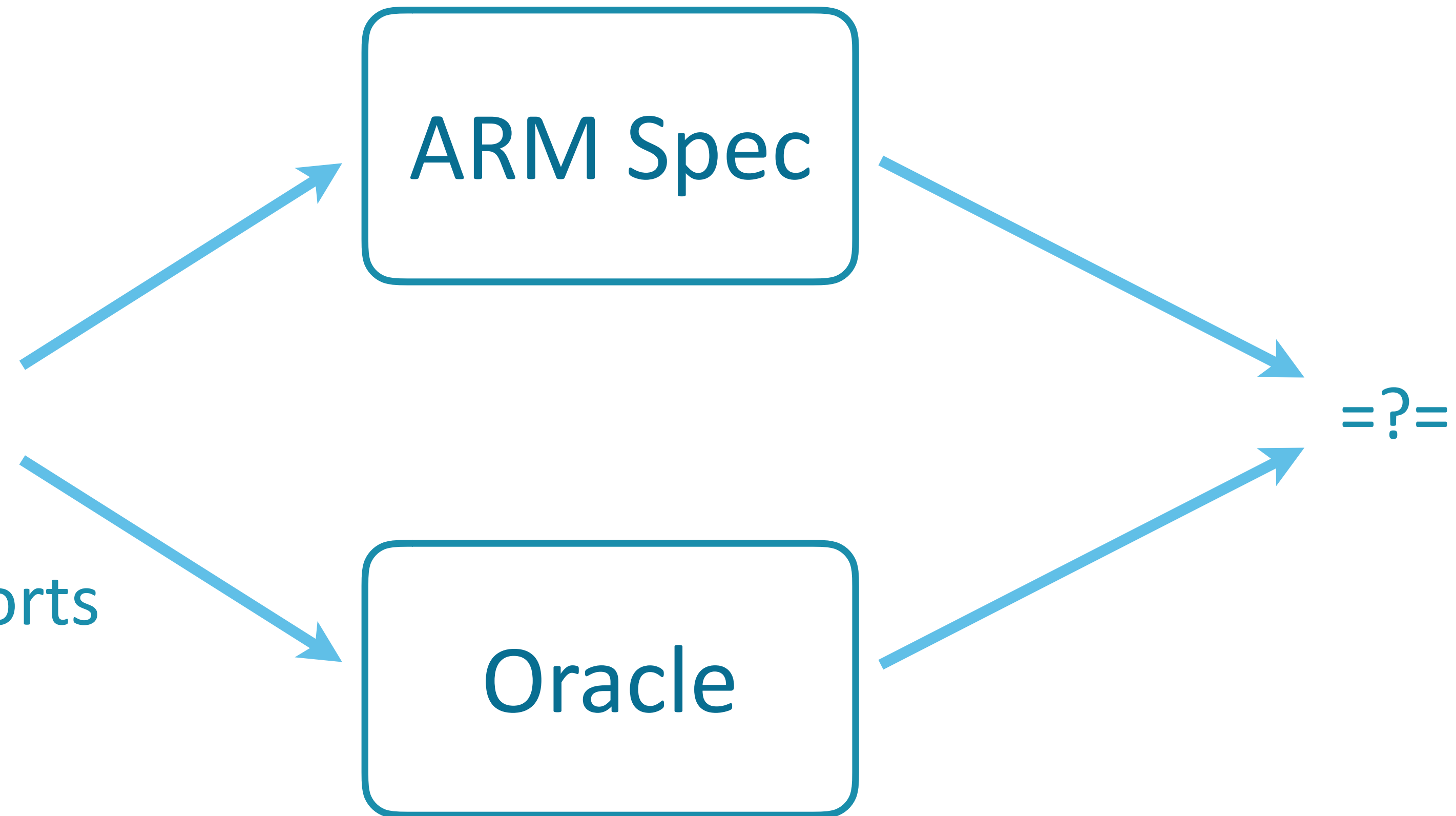


Tests:

- Directed
- Random
- ...

Generators:

- Memory Aborts
- Interrupts

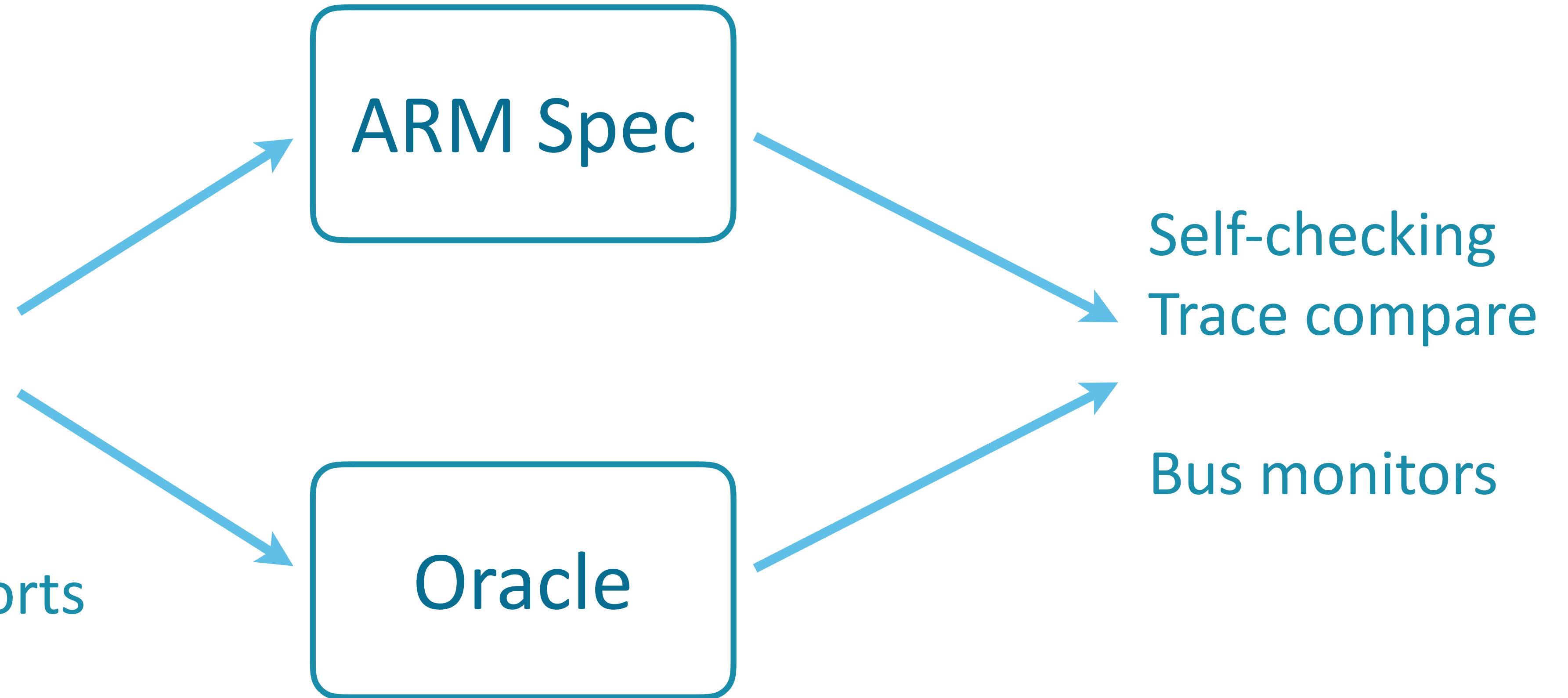


Tests:

- Directed
- Random
- ...

Generators:

- Interrupts
- Memory Aborts



Architecture Conformance Suite

Processor architectural compliance sign-off

Large

v8-A 11,000 test programs, > 2 billion instructions

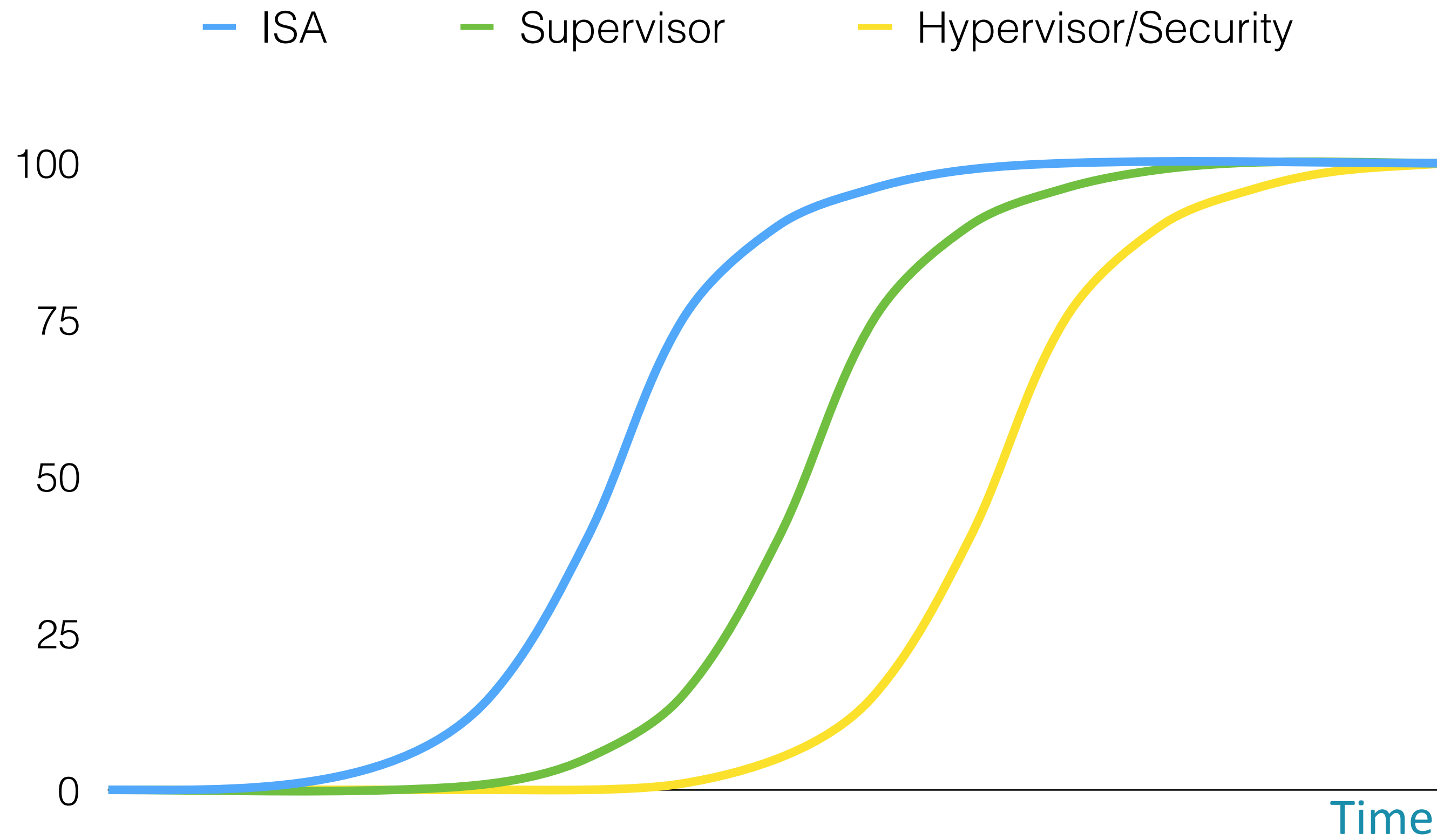
v8-M 3,500 test programs, > 250 million instructions

Thorough

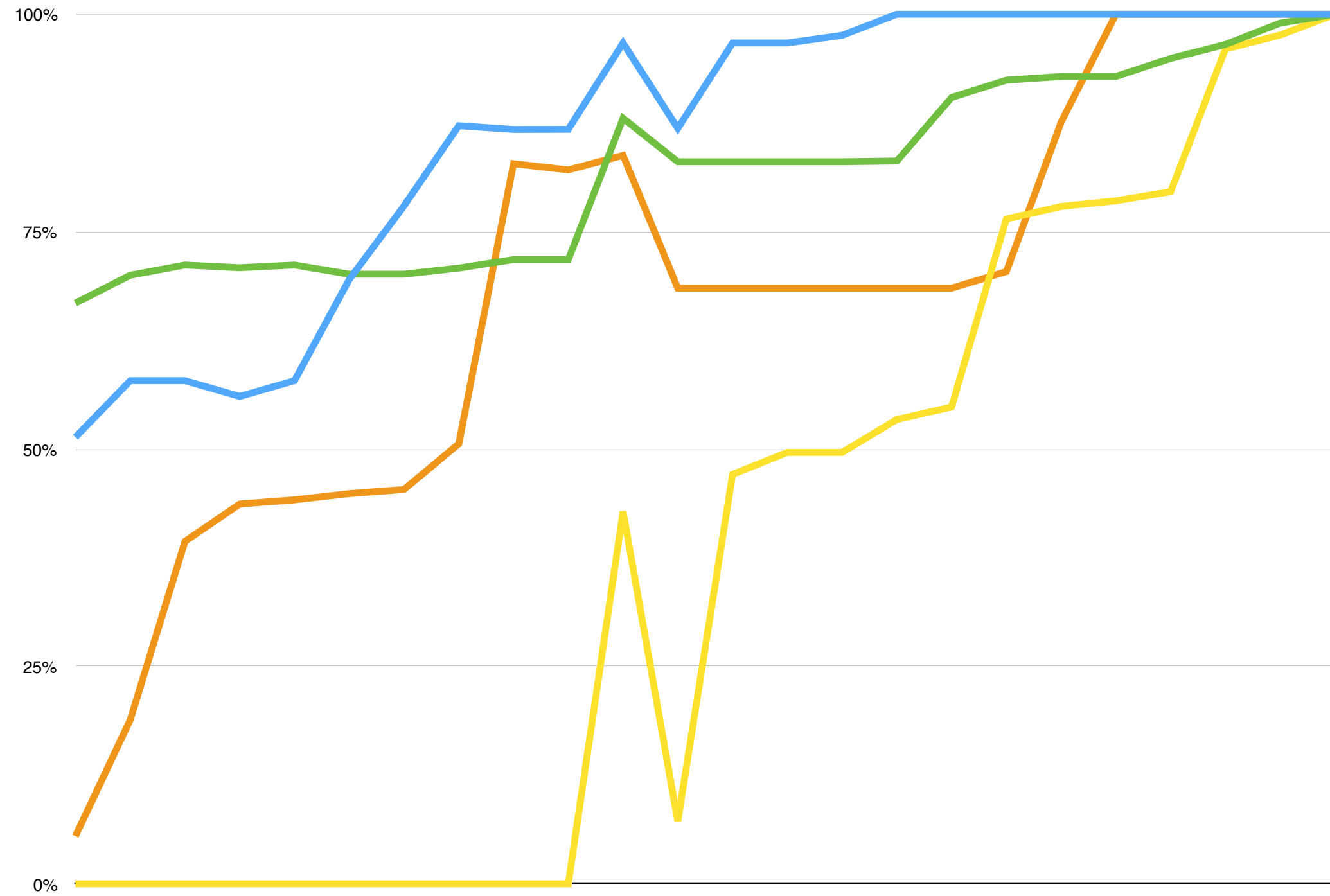
Tests dark corners of specification

Testing Pass Rate

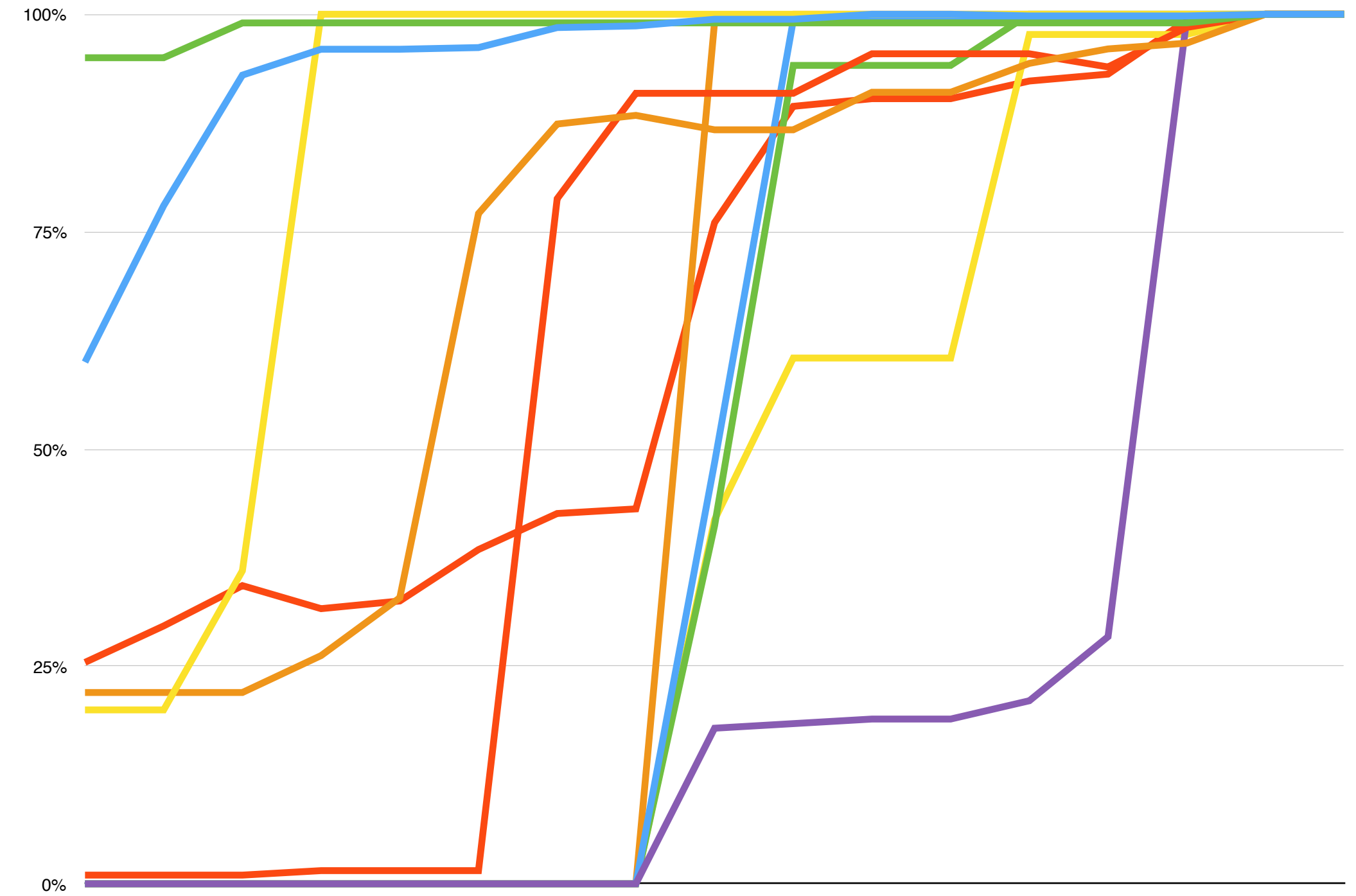
(Artists Impression)

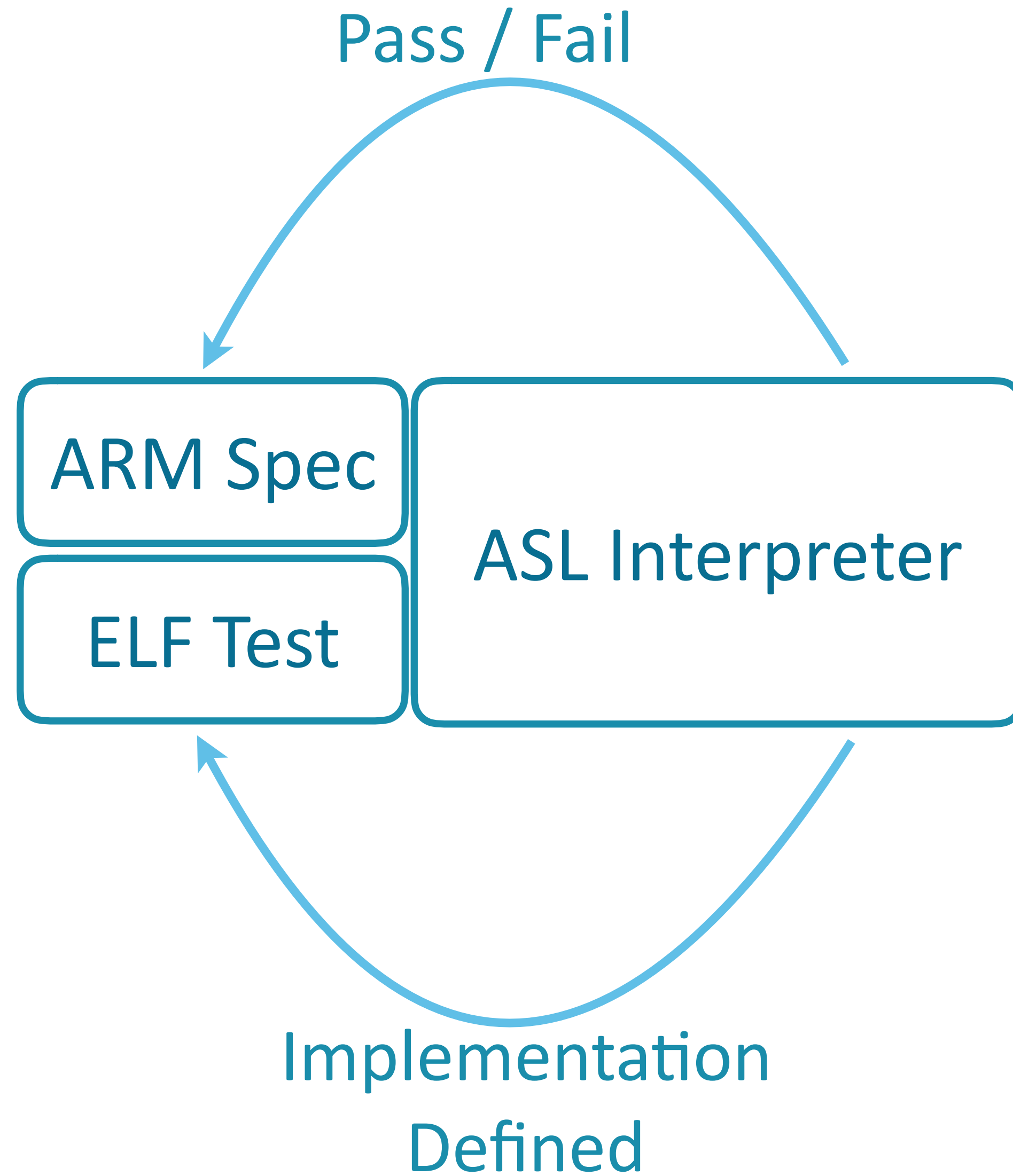


v8-A

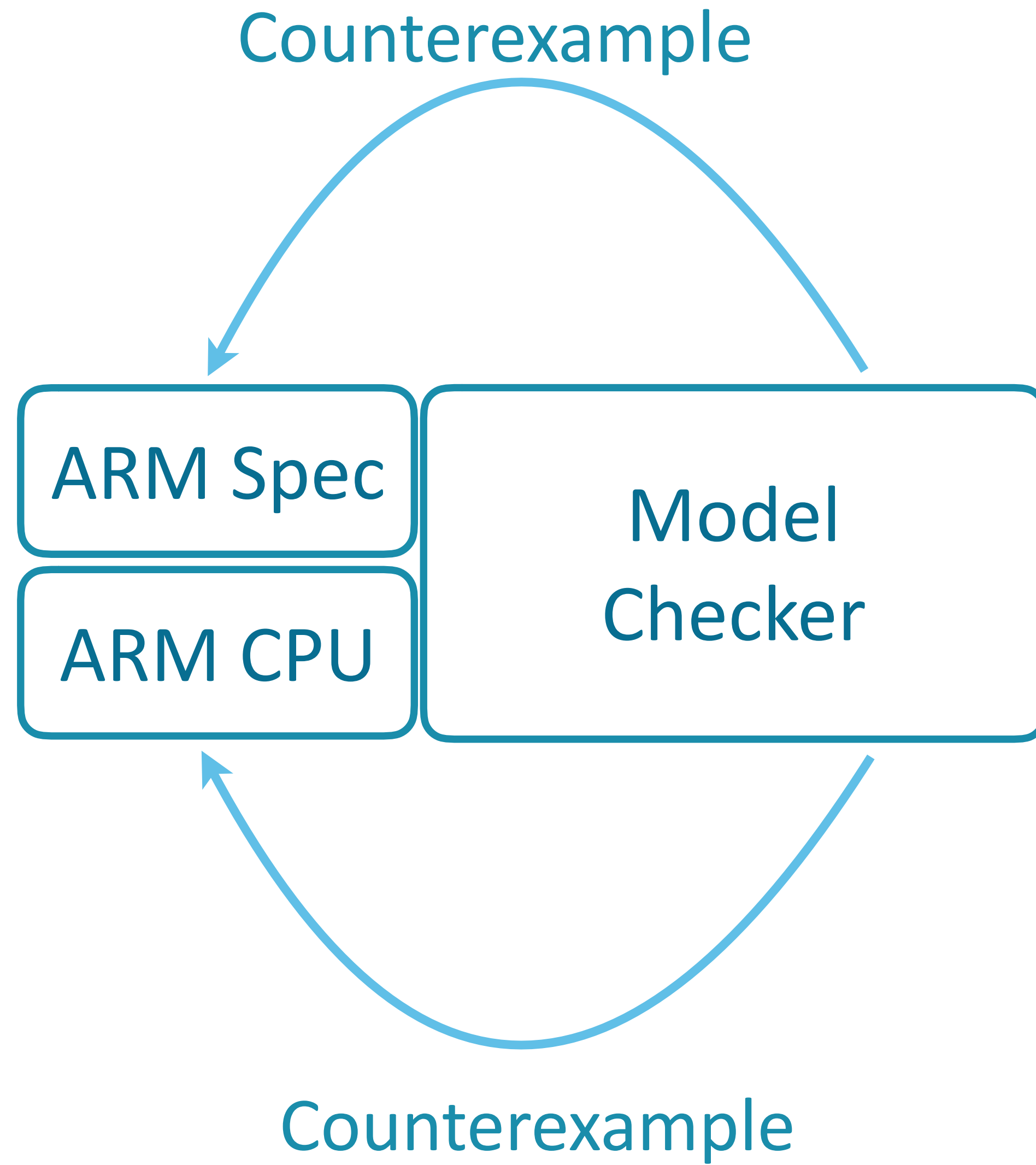


v8-M

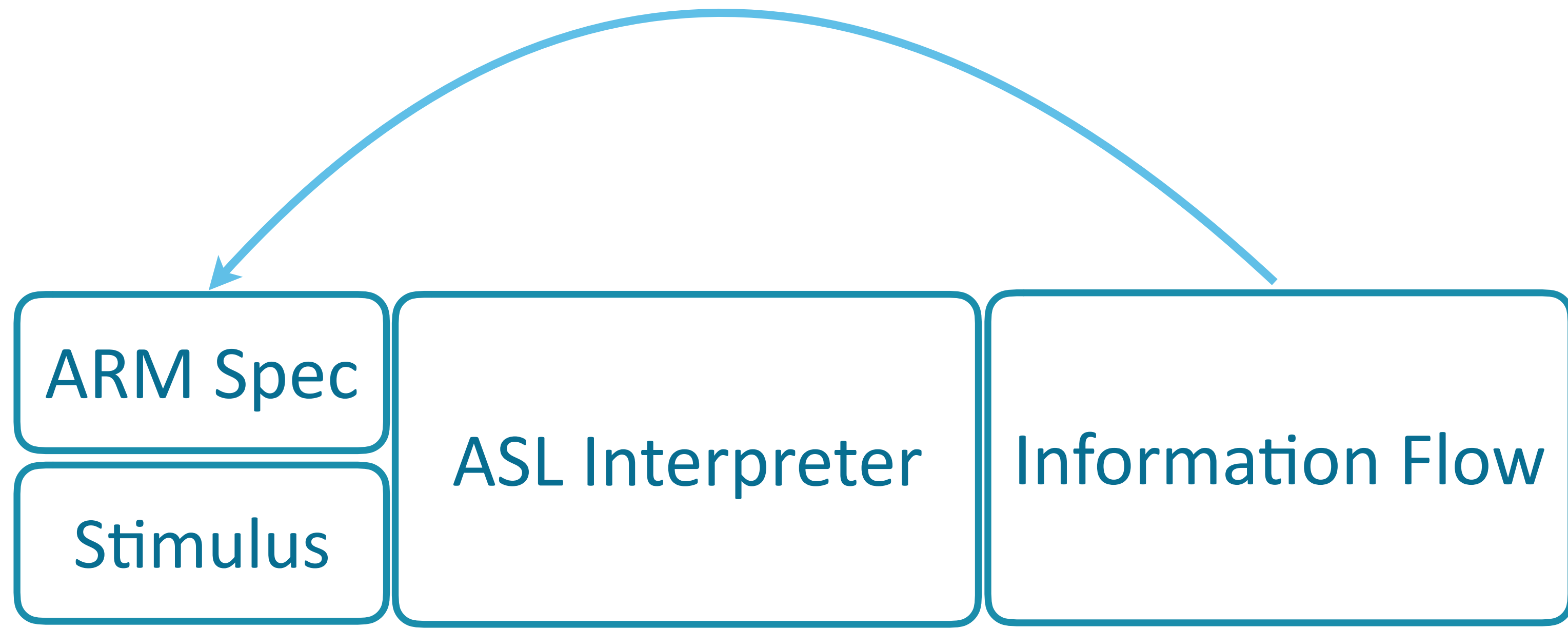




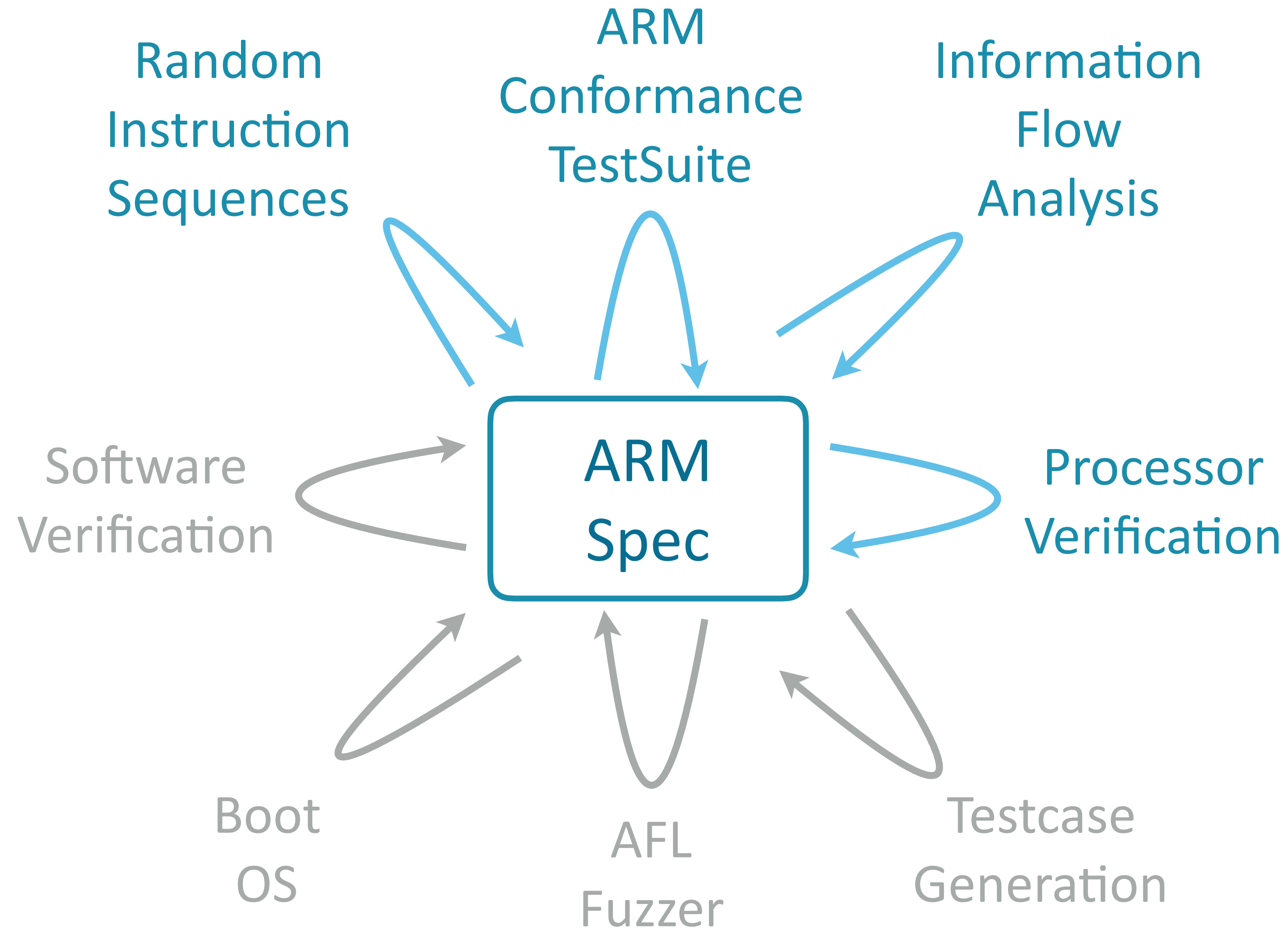
End to End Verification of ARM Processors with ISA-Formal, CAV 2016



Information Leak



Creating a Virtuous Cycle



Preparing public release of ARM v8-A specification

- Enable formal verification of software and tools
- Public release planned for 2016 Q4
- Liberal license
- Cambridge University REMS group currently translating to SAIL

Talk to me about how I can help you use it

CPU Specifications

Basis of a lot of formal verification

Too large to be “obviously correct”

Reusable specs enable “virtuous cycle”

- Increases Scope requirements

- Share testing / maintenance effort

- More likely to be correct

Preparing public release of machine readable ARM Specification

End

Alastair Reid

ARMResearch

alastair.reid@arm.com

@alastair_d_reid

