Trustworthy Specifications of ARM® System Level Architecture

Alastair Reid
alastair.reid@arm.com
@alastair_d_reid
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

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|---|
| 1  | 1  | 0  | 1  | 0  | 1  | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |

Rd | Rm

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;

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

route_to_el3 = HaveEL(EL3) && SCR_EL3.EA == '1' && IsExternalAbort(fault);
route_to_el2 = (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_el3 then
    AArch64.TakeException(EL3, exception, preferred_exception_return, vect_offset);
el elseif PSTATE.EL == EL2 || route_to_el2 then
    AArch64.TakeException(EL2, exception, preferred_exception_return, vect_offset);
el else
    AArch64.TakeException(EL1, exception, preferred_exception_return, vect_offset);
## ARM Spec (lines of code)

<table>
<thead>
<tr>
<th>Category</th>
<th>v8-A</th>
<th>v8-M</th>
</tr>
</thead>
<tbody>
<tr>
<td>Instructions</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Int/FP/SIMD</td>
<td>26,000</td>
<td>6,000</td>
</tr>
<tr>
<td>Exceptions</td>
<td>4,000</td>
<td>3,000</td>
</tr>
<tr>
<td>Memory</td>
<td>3,000</td>
<td>1,000</td>
</tr>
<tr>
<td>Debug</td>
<td>3,000</td>
<td>1,000</td>
</tr>
<tr>
<td>Misc</td>
<td>5,500</td>
<td>2,000</td>
</tr>
<tr>
<td>(Test support)</td>
<td>1,500</td>
<td>2,000</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>43,000</strong></td>
<td><strong>15,000</strong></td>
</tr>
</tbody>
</table>
# System Register Spec

<table>
<thead>
<tr>
<th></th>
<th>v8-A</th>
<th>v8-M</th>
</tr>
</thead>
<tbody>
<tr>
<td>Registers</td>
<td>586</td>
<td>186</td>
</tr>
<tr>
<td>Fields</td>
<td>3951</td>
<td>622</td>
</tr>
<tr>
<td>Constant</td>
<td>985</td>
<td>177</td>
</tr>
<tr>
<td>Reserved</td>
<td>940</td>
<td>208</td>
</tr>
<tr>
<td>Impl. Defined</td>
<td>70</td>
<td>10</td>
</tr>
<tr>
<td>Passive</td>
<td>1888</td>
<td>165</td>
</tr>
<tr>
<td>Active</td>
<td>68</td>
<td>62</td>
</tr>
<tr>
<td>Operations</td>
<td>112</td>
<td>10</td>
</tr>
</tbody>
</table>
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?
ARM Spec

Test Stimulus

Oracle

=?=
Tests:
- Directed
- Random
- ...

Generators:
- Memory Aborts
- Interrupts
Tests:
- Directed
- Random
- ...

Generators:
- Interrupts
- Memory Aborts

ARM Spec

Oracle

Self-checking Trace compare
Bus monitors
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)

Time

ISA  Supervisor  Hypervisor/Security
Pass / Fail

Implementation Defined

ARM Spec

ELF Test

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

Counterexample

ARM Spec

ARM CPU

Model Checker

Counterexample
Information Leak

- ARM Spec
- Stimulus
- ASL Interpreter
- Information Flow
Creating a Virtuous Cycle

- Random Instruction Sequences
- ARM Conformance TestSuite
- Information Flow Analysis
- Software Verification
- Processor Verification
- Boot OS
- AFL Fuzzer
- Testcase Generation
- ARM Spec

The Architecture for the Digital World®
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