ARM® Glossary

ARM AEG 0014G (ID012615)

ARM Glossary

Release Information

The following changes have been made to this book.

Change history

Date	Issue	Confidentiality	Change
26 March 2010	A	Non-Confidential	New document.
27 September 2010	B	Non-Confidential	Second revision. Updated with new terms for the ARM architectures.
27 January 2011	C	Non-Confidential	Third revision. Updated with new terms for DS-5.
24 August 2011	D	Non-Confidential	Fourth revision. Updated with enhancements.
17 May 2012	E	Non-Confidential	Fifth revision. Updated with enhancements.
06 June 2014	F	Non-Confidential	Sixth revision. Updated with enhancements and new terms. Internal release only.
26 January 2015	G	Non-Confidential	Seventh revision. Updated with enhancements and new terms.

Proprietary Notice

Confidentiality Status

This document is Non-Confidential. The right to use, copy and disclose this document may be subject to license restrictions in accordance with the terms of the agreement entered into by ARM and the party that ARM delivered this document to.

Product Status

The information in this document is final, that is for a developed product.

Web Address

`http://www.arm.com`

Contents
ARM Glossary

Chapter 1
Using the ARM Glossary

This glossary is a collection of ARM-specific terminology for product documentation.

If you have comments on content, send an e-mail to `errata@arm.com`. Give:

- The title.
- The number, ARM AEG 0014G.
- The glossary item to which your comments apply.
- A concise explanation of your comments.

ARM also welcomes general suggestions for additions and improvements.

Glossary

This glossary describes some of the technical terms used in ARM documentation.

AAPCS

See Procedure Call Standard for the ARM Architecture (AAPCS).

AArch32 state

The ARM 32-bit Execution state that uses 32-bit general purpose registers, and a 32-bit program counter (PC), stack pointer (SP), and link register (LR). AArch32 Execution state provides a choice of two instruction sets, A32 and T32.

In implementations of versions of the ARM architecture before ARMv8, and in the ARM R and M architecture profiles, execution is always in AArch32 state.

See also AArch64 state, A32 instruction, T32 instruction.

AArch64 state

The ARM 64-bit Execution state that uses 64-bit general purpose registers, and a 64-bit program counter (PC), stack pointer (SP), and exception link registers (ELR). AArch64 Execution state provides a single instruction set, A64.

AArch64 state is supported only in the ARMv8-A architecture profile.

See also AArch32 state, A64 instruction.

A32 instruction

An instruction executed by a core that is in AArch32 Execution state and A32 Instruction set state. A32 is a fixed-width instruction set that uses 32-bit instruction encodings. Previously, this instruction set was called the ARM instruction set.

See also AArch32 state, A32 state, T32 instruction.

A32 state

When a core is in the AArch32 Execution state, if it is in the A32 Instruction set state then it executes A32 instructions.

See also AArch32 state, A32 instruction, T32 state.

A64 instruction
The instruction set used by an ARMv8-A core that is in AArch64 Execution state. A64 is a fixed-width instruction set that uses 32-bit instruction encodings.

See also AArch64 state, A32 instruction, T32 instruction.

ABI
See Application Binary Interface for the ARM Architecture (ABI).

Abort
An abort occurs when an illegal memory access causes an exception. An abort can be generated by the hardware that manages memory accesses, or by the external memory system. The hardware that generates the abort might be a *Memory Management Unit* (MMU) or a *Memory Protection Unit* (MPU).

See also External Abort, Data Abort, Prefetch Abort.

Abort model
Describes the changes to the core state when a Data abort exception occurs. Different abort models behave differently with regard to load and store instructions that specify base register writeback.

ACE interface
An AMBA AXI4 interface that includes full support for the ACE protocol. An ACE interface adds:

- Signals to some of the AXI4 channels.
- Channels to the AXI4 interface.

See also ACE protocol, ACE-Lite interface.

ACE-Lite interface
An AMBA AXI4 interface that includes support for a subset of the full ACE protocol. An ACE-Lite interface adds signals to some of the AXI4 channels, but does not add any channels to the interface.

See also ACE interface, ACE protocol.

ACE protocol
The AXI Coherency Extensions protocol, that adds signals to an AMBA AXI4 interface, to support managing the coherency of a distributed memory system.

See also ACE interface, ACE-Lite interface.

Adaptive clocking
A technique where the debug interface hardware sends out a clock signal and then waits for the returned clock before generating the next clock pulse. This technique enables the run control unit in the debug hardware to adapt to differing signal drive capabilities and differing cable lengths.

Addressing mode
An addressing mode specifies a method of generating the memory address that a load or store instruction uses.

The addressing modes mechanism can generate values for data processing instructions to use as operands.

ADI
See ARM Debug Interface (ADI).

Advanced SIMD
A feature of the ARM architecture that provides *Single Instruction Multiple Data* (SIMD) operations on a dedicated bank of registers. If an implementation also supports scalar floating-point instructions, the floating-point and Advanced SIMD instructions use a common register bank. ARM NEON technology provides the Advanced SIMD instructions, and therefore these are sometimes called the NEON instructions.

See also NEON technology, Single Instruction, Multiple Data (SIMD).

AEL
See ARM Embedded Linux (AEL).

AHB
An AMBA bus protocol supporting pipelined operation, with the address and data phases occurring during different clock periods. This means the address phase of a transfer can occur during the data phase of the previous transfer. AHB provides a subset of the functionality of the AMBA AXI protocol.

See also AMBA, AHB-Lite.

AHB Access Port (AHB-AP)
An optional component of the DAP that provides an AHB interface to a SoC.

CoreSight supports access to a system bus infrastructure using the AHB Access Port in the *Debug Access Port* (DAP). The AHB-AP provides an AHB master port for direct access to system memory. Other bus protocols can use AHB bridges to map transactions. For example, you can use AHB to AXI bridges to provide AHB access to an AXI bus matrix.

See also Debug Access Port (DAP).

AHB Trace Macrocell (HTM)
A trace source that makes bus information visible. This information cannot be inferred from the processor using just a trace macrocell. HTM trace can provide:

- An understanding of multi-layer bus utilization.
- Software debug. For example, visibility of access to memory areas and data accesses.
- Bus event detection for trace trigger or filters, and for bus profiling.

See also AHB.

AHB-AP
See AHB Access Port (AHB-AP).

AHB-Lite
A subset of the full AMBA AHB protocol specification. It provides all of the basic functions required by the majority of AMBA AHB slave and master designs, particularly when used with a multi-layer AMBA interconnect.

Aligned
A data item stored at an address that is exactly divisible by the number of bytes that defines its data size. Aligned doublewords, words, and halfwords have addresses that are divisible by eight, four, and two respectively. An aligned access is one where the address of the access is aligned to the size of each element of the access.

AMBA
The AMBA family of protocol specifications is the ARM open standard for on-chip buses. AMBA provides solutions for the interconnection and management of the functional blocks that make up a *System-on-Chip* (SoC). Applications include the development of embedded systems with one or more processors or signal processors and multiple peripherals.

APB
An AMBA bus protocol for ancillary or general-purpose peripherals such as timers, interrupt controllers, UARTs, and I/O ports. Using APB to connect to the main system bus through a system-to-peripheral bus bridge can help reduce system power consumption.

APB Access Port (APB-AP)
An optional component of the *Debug Access Point* (DAP) that provides an APB interface to a SoC, usually to its main functional buses.

APB-AP
See APB Access Port (APB-AP).

Application Binary Interface for the ARM Architecture (ABI)
A collection of specifications, some open and some specific to the ARM architecture, that regulate the inter-operation of binary code in a range of execution environments for ARM cores. The base standard specifies those aspects of code generation that must conform to a standard that supports inter-operation. It is aimed at authors and vendors of C and C++ compilers, linkers, and runtime libraries.

Architecturally executed

An instruction is architecturally executed only if it would be executed in a simple sequential execution of the program. When such an instruction has been executed and retired it has been architecturally executed. Any instruction that, in a simple sequential execution of a program, is treated as a NOP because it fails its condition code check, is an architecturally executed instruction.

In a processor that performs speculative execution, an instruction is not architecturally executed if the processor discards the result of the speculative execution of that instruction.

See also Condition code check, Simple sequential execution.

ARM Compiler for DS-5

A suite of tools, together with supporting documentation and examples, that you can use to write and build applications for the ARM family of processors. ARM Compiler for DS-5 supersedes the RealView Compilation Tools.

See also armar, armasm, armcc, fromelf.

ARM Debug Interface (ADI)

The ADI connects a debugger to a device. The ADI is used to access memory-mapped components in a system, such as processors and CoreSight components. The ADI protocol defines the physical wire protocols permitted, and the logical programmers model.

ARM Embedded Linux (AEL)

A version of embedded Linux OS ported to the ARM architecture.

ARM instruction

See A32 instruction.

ARM profiler

A plug-in to the ARM Workbench Integrated Development Environment that provides non-intrusive analysis of embedded software over time, on targets running at frequencies that are typically 250MHz. Targets can be *Real-Time System Models* (RTSMs) and hardware targets.

See also ARM Workbench IDE (AWIDE), RealView Development Suite (RVDS).

ARM state

See A32 state.

ARM TrustZone® technology

The hardware and software that enable the integration of enhanced security features throughout a SoC. In ARMv6K and ARMv7, the Security Extensions implement the TrustZone hardware. In ARMv8, EL3 incorporates the TrustZone hardware.

See also TrustZone Software.

ARM Workbench IDE (AWIDE)

ARM Workbench IDE is based on the Eclipse IDE, and provides additional features to support the ARM development tools provided in RVDS.

See also RealView Development Suite (RVDS).

armar

The ARM librarian that you can use to create libraries of files, such as object files.

See also Development Studio 5 (DS-5), RealView Compilation Tools (RVCT).

armasm

The ARM assembler. This converts ARM assembly language into machine code.

See also Development Studio 5 (DS-5), RealView Compilation Tools (RVCT).

armcc

The ARM compiler for C and C++ code.

See also Development Studio 5 (DS-5), RealView Compilation Tools (RVCT).

ArtiGrid

A power routing scheme, also referred to as Over The Cell.

ATB
An AMBA bus protocol for trace data. A trace device can use an ATB to share CoreSight capture resources.

ATB bridge
A synchronous ATB bridge provides a register slice that meets timing requirements by adding a pipeline stage. It provides a unidirectional link between two synchronous ATB domains.

An asynchronous ATB bridge provides a unidirectional link between two ATB domains with asynchronous clocks. This means it connects components in different clock domains.

See also ATB, Register slice.

Atomicity
Describes actions that appear to happen as a single operation. In the ARM architecture, atomicity refers to either single-copy atomicity or multi-copy atomicity. The *ARM Architecture Reference Manual*s define these forms of atomicity.

Authentication Asynchronous Bridge
Transfers authentication signals between two asynchronous clock domains.

See also Authentication Synchronous Bridge.

Authentication Synchronous Bridge
Transfers authentication signals between two synchronous clock domains.

See also Authentication Asynchronous Bridge.

AWIDE
See ARM Workbench IDE (AWIDE).

AXI
An AMBA bus protocol that supports:

- Separate phases for address or control and data.
- Unaligned data transfers using byte lane strobes.
- Burst-based transactions with only the start address issued.
- Separate read and write data channels.
- Issuing multiple outstanding addresses.
- Out-of-order transaction completion.
- Optional addition of register stages to meet timing or repropagation requirements.

The AXI protocol includes optional signaling extensions for low-power operation.

See also AXI Coherency Extensions (ACE), Burst, Byte lane strobe.

AXI Coherency Extensions (ACE)
The *AXI Coherency Extensions* (ACE) provide additional channels and signaling to an AXI interface to support system level cache coherency.

Back-annotation
The process of applying timing characteristics from the implementation process onto a model.

Banked registers
A register that has multiple instances. A property of the state of the device determines which instance is in use. For example the Security state might determine which instance is in use.

Base Platform Application Binary Interface (BPABI)
The base standard for the interface between executable files, such as dynamic shared objects and DLLs, and the systems that execute them.

Base porting layer
A platform-dependent base driver software component that communicates with the Mali GPU. For example, the base porting layer controls the Mali GPU registers. You implement, or port, the base porting layer onto different target platforms.

Base register
A register specified by a load or store instruction that is used as the base value for the address calculation for the instruction. Depending on the instruction, an offset can be added to or subtracted from the base register value to form the address that is used for the memory access.

Base register writeback
Writing back a modified value to the base register used in an address calculation.

Base Standard Application Binary Interface (BSABI)
See Application Binary Interface for the ARM Architecture (ABI).

BCD file
See Board and Chip Definition (BCD) file.

Beat
An alternative term for an individual transfer within a burst. For example, an INCR4 burst comprises four beats.

See also Burst.

Big-endian
In the context of the ARM architecture, big-endian is defined as the memory organization in which, for example:

- A byte or halfword at a word-aligned address is the most significant byte or halfword in the word at that address.
- A byte at a halfword-aligned address is the most significant byte in the halfword at that address.

See also Little-endian.

Board and Chip Definition (BCD) file
In the context of RealView Debugger, a BCD file enables you to define the memory map and memory-mapped registers for a target development board or core. ARM provides various BCD files with RVDS for ARM development boards.

See also Board file, Debug configuration.

Board file
A Real View Debugger uses this term to refer to the top-level configuration file, normally called `rvdebug.brd`, that references one or more other configuration files. A board file contains:

- The Debug Configuration (connection-level) settings.
- References to the Debug Interface configuration file that identifies the targets on the development platform.
- References to any *Board and Chip Definition* (BCD) files assigned to a Debug Configuration.

See also Board and Chip Definition (BCD) file, Debug configuration.

Bounce
In an ARMv7 floating-point implementation that includes a VFP subarchitecture, a mechanism for handling floating point exceptions and instructions that are not supported by the hardware. A bounce generates an Undefined Instruction exception, and the exception handler can call support code to respond to the bounce.

See also Exceptional state, Trigger instruction.

Boundary scan chain
A boundary scan chain is made up of serially-connected devices that implement boundary scan technology using a standard JTAG TAP interface. Each device contains at least one TAP controller containing shift registers that form the chain, connected between the **TDI** and **TDO** signals, through which test data is shifted. A core can contain several shift registers, enabling a scan to access selected parts of the device.

BPABI
See Base Platform Application Binary Interface (BPABI).

Branch folding
A technique where, on the prediction of a branch, the target instructions are completely removed from the instruction stream presented to the execution pipeline. Branch folding can significantly improve the performance of branches, and take the CPI for branches below one.

Branch phantom
Branch target instructions speculatively executed, in parallel with the main instruction stream, as a result of branch folding.

Branch prediction
The selection of a future execution path for instruction fetch. For example, after a branch instruction, the core can choose to speculatively fetch either the instruction following the branch or the instruction at the branch target.

See also Prefetching.

Breakpoint
A debug event triggered by the execution of a particular instruction. It is specified by the address of the instruction and, optionally, by the state of the core when the instruction is executed.

See also Watchpoint.

Breakpoint unit
In the context of an ARM debugger, a unit in a Chained breakpoint that combines with other breakpoint units to create a complex hardware breakpoint.

In an M-profile core, a hardware debug component that can be part of the Flash Patch and Breakpoint unit.

See also Hardware breakpoint, Chained breakpoint.

BSABI
See Application Binary Interface for the ARM Architecture (ABI).

Burst
A group of transfers that form a single transaction. With AMBA protocols, only the first transfer of the burst includes address information, and the transfer type determines the addresses used for subsequent transfers.

See also Beat.

Byte lane strobe
A signal that determines which byte lanes are active, or valid, in a data transfer. Each bit of this signal corresponds to eight bits of the data bus.

Byte swizzling
Re-arranging the order of bytes in a word or halfword.

Byte-invariant
In a byte-invariant system, the address of each byte of memory remains unchanged when switching between little-endian and big-endian operation. When a data item larger than a byte is loaded from or stored to memory, the bytes making up that data item are arranged into the correct order depending on the endianness of the memory access.

The ARM architecture supports byte-invariant systems in ARMv6 and later versions.

See also Word-invariant.

CADI
The debug control and inspection API to a fast model.

Canonical Frame Address (CFA)
In *Debug With Arbitrary Record Format* (DWARF), this is an address on the stack specifying where the call frame of an interrupted function is located.

Captive thread
Captive threads are all the threads that can be brought under the control of RVDS. Special threads, called non-captive threads, are essential to the operation of *Running System Debug* (RSD) and so are not under debugger control.

See also RealView Development Suite (RVDS), Running System Debug (RSD).

Cast out
See Victim.

CFA
See Canonical Frame Address (CFA).

Chained breakpoint
In the context of an ARM debugger, a complex breakpoint that comprises multiple hardware breakpoint units.

See also Breakpoint unit, Conditional breakpoint, Data breakpoint, Hardware breakpoint, Software breakpoint.

Chained tracepoint
In the context of an ARM debugger, a complex tracepoint that comprises multiple tracepoint units.

See also Tracepoint unit, Tracepoint.

Characterized
Designates a cell that includes timing data.

Clean
A cache line that has not been modified while it is in the cache is said to be clean. To clean a cache is to write dirty cache entries to main memory.

See also Dirty.

Clock gating
Gating a clock signal for a macrocell or functional block with a control signal and using the modified clock that results to control the operating state of the macrocell or block.

Clocks Per Instruction (CPI)
See Cycles Per Instruction (CPI).

Cluster
The preferred term for a group of cores. Typically, the set of cores in a cluster share some functionality, for example you might refer to a cluster that shares a single level 2 cache. To distinguish it from other clusters you might describe this group of cores as a level 2 cluster.

See also Core.

CMM
In the context of an ARM debugger, a scripting language provided for compatibility with other debuggers.

If you are writing new scripts, ARM recommends that you use the *GNU Debugger* (GDB) scripting commands because these offer more functionality in the ARM Debuggers.

Coherence Order
See Coherent.

Coherent
Data accesses from a set of observers to a byte in memory are coherent if accesses to that byte by the members of the set of observers are consistent with there being a single total order of all writes to that byte in memory by all members of the set of observers. This single total order of all writes to that byte is the *coherence order* for that byte.

See also Memory coherency.

Cold reset
A cold reset has the same effect as starting the processor by turning the power on. This clears main memory and many internal settings. Some program failures can lock up the core and require a cold reset to restart the system.

This is also known as power-on or powerup reset.

See also Warm reset.

Communications channel
The hardware used for communicating between the software running on the core, and an external host, using the debug interface. When this communication is for debug purposes, it is called the *Debug Communications Channel* (DCC).

See also Debug Communications Channel (DCC).

Condensed Reference Format (CRF)
An ARM proprietary file format for specifying test vectors. Typically, ARM supplies a script to convert CRF format to *Verilog Reference Format* (VRF).

Condition code check
The process of determining whether a conditional instruction executes normally or is treated as a NOP. For an instruction that includes a condition code field, that field is compared with the condition flags to determine whether the instruction is executed normally. For a T32 instruction in an IT block, the value of the ITSTATE register determines whether the instruction is executed normally.

See also Condition code field, Condition flags, Conditional execution.

Condition code field
A 4-bit field in an A32 instruction that specifies the condition under which the instruction executes.

See also Condition code check.

Condition flags
The N, Z, C, and V bits of PSTATE or of a *Program Status Register* (PSR).

See also Conditional execution, Current Program Status Register (CPSR), PSTATE, Saved Program Status Register (SPSR).

Conditional breakpoint
In the context of an ARM debugger, a breakpoint that has one or more condition qualifiers assigned. The breakpoint is activated when all assigned conditions are met, and either stops or continues execution depending on the action qualifiers that are assigned. The condition normally references the values of program variables that are in scope at the breakpoint location.

See also Chained breakpoint, Software breakpoint.

Conditional execution
When a conditional instruction starts executing, if the condition code check returns TRUE, the instruction executes normally. Otherwise, it is treated as NOP.

See also Condition code check.

CONSTRAINED UNPREDICTABLE
Where an instruction can result in UNPREDICTABLE behavior, the ARM architecture can specify a narrow range of permitted behaviors. This range is the range of CONSTRAINED UNPREDICTABLE behavior. All implementations that are compliant with the architecture must follow the CONSTRAINED UNPREDICTABLE behavior. However, software must not rely on any CONSTRAINED UNPREDICTABLE behavior.

When CONSTRAINED UNPREDICTABLE appears in body text, it is always in SMALL CAPITALS.

See also UNPREDICTABLE.

Context switch
The saving and restoring of computational state when switching between different threads or processes. Context switch describes any situation where the context is switched by an operating system and might or might not include changes to the address space.

Context synchronization operation
A context synchronization operation is one of:

- In all versions of the ARM architecture:
  - The execution of an `ISB` instruction that does not fail its condition code check.
  - The taking of an exception.
  - The return from an exception.
- In addition, in ARMv8:
  - Exit from Debug state.
  - Executing a `DCPS` instruction.
  - Executing a `DRPS` instruction.

The architecture requires a context synchronization operation to guarantee visibility of any change to a System register.

See also ISB.

Coprocessor

A processor, or conceptual processor, that supplements the main processor to carry out additional functions. In AArch32 Execution state, the ARM architecture defines an interface to up to 16 coprocessors, CP0-CP15.

In ARMv8, AArch32 state supports only conceptual coprocessors CP10, CP11, CP14, and CP15. In previous versions of the architecture, coprocessors CP0-CP7 are available for IMPLEMENTATION DEFINED features, and coprocessors CP8-CP15 are reserved for use by ARM.

In all architecture versions for the A and R architecture profiles, in AArch32 state:

- CP15 instructions access the System registers. Some documentation describes this set of registers as the System Control Coprocessor.
- CP14 instructions access System registers for debug, trace, and execution environment features.
- The CP10 and CP11 instruction space is for floating-point and Advanced SIMD instructions if supported, including the instructions for accessing the floating-point and Advanced SIMD System registers.

Core

In ARM processor documentation, the entity that is described by an ARM *Architecture Reference Manual*. ARM *Architecture Reference Manuals* describe this entity as a PE.

See also Processing Element (PE), Processor.

Core module

In the context of an ARM Integrator development board, an add-on development board that contains an ARM processor and local memory. Core modules can run standalone, or can be stacked onto Integrator development boards.

See also Integrator.

Core register

Processing registers used in AArch32 Execution state, comprising:

- 13 general-purpose registers, R0 to R12, that software uses for all data processing when using the base instruction set instructions.
- SP, the Stack Pointer, that can also be referred to as R13.
- LR, the Link Register, that can also be referred to as R14.
- PC, the Program Counter, that can also be referred to as R15.

In some situations, software can use SP, LR, and PC for processing. The instruction descriptions include any constraints on the use of SP, LR, and PC.

See also LR, SP.

Core reset

See Warm reset.

CoreSight

ARM on-chip debug and trace components, that provide the infrastructure for monitoring, tracing, and debugging a complete system on chip.

See also CoreSight ECT, CoreSight ETB, CoreSight ETM.

CoreSight ECT

See Embedded Cross Trigger (ECT).

CoreSight ETB

See Embedded Trace Buffer (ETB).

CoreSight ETM

See Embedded Trace Macrocell (ETM).

CoreSight STM

See System Trace Macrocell (STM).

CoreSight TMC *See* Trace Memory Controller (TMC).

CPI Cycles or clocks per instruction.

See also Cycles Per Instruction (CPI).

CPSR *See* Current Program Status Register (CPSR).

CRF *See* Condensed Reference Format (CRF).

Cross-path blocking This occurs when a divergent node has congestion on one of its output nodes which blocks bus traffic to its other output nodes.

See also Quality of Service using Virtual Networks (QVN), Head-of-line blocking.

Cross Trigger Interface (CTI)
Part of an *Embedded Cross Trigger* (ECT) device. In an ECT, the CTI provides the interface between a processor or ETM and the CTM.

Cross Trigger Matrix (CTM)
Part of an *Embedded Cross Trigger* (ECT) device. In an ECT, the CTM combines the trigger requests generated by CTIs and broadcasts them to all CTIs as channel triggers.

CTI *See* Cross Trigger Interface (CTI).

CTM *See* Cross Trigger Matrix (CTM).

Current Program Status Register (CPSR)
In AArch32 state, the register that holds the current program status.

See also PSTATE, Program Status Register (PSR), Saved Program Status Register (SPSR).

Cycles Per Instruction (CPI)
A measure of the number of computer instructions that can be performed in one clock cycle. It is also called clocks per instruction. You can use this value to compare the performance of different processors that implement the same instruction set. The lower the value, the better the performance.

DA *See* Debug Agent.

DAP *See* Debug Access Port (DAP).

DAPBUS interconnect The DAPBUS interconnect connects a *Debug Port* (DP) to the *Access Ports* (APs) in a CoreSight DAP.

See also Debug Access Port (DAP).

Data Abort An indication to the core of an attempted data access that is not permitted. The Data Abort might be generated by access permission checks performed by the memory system on the core, or might be signaled by the memory system.

See also Abort, External Abort, Instruction Abort, Prefetch Abort.

Data breakpoint In the context of an ARM debugger, a hardware breakpoint that activates when an access to a specified location meets a set of specified conditions. The conditions can include a check for a specific data value being accessed at the given location.

See also Chained breakpoint, Conditional breakpoint, Software breakpoint.

Data-active write transaction
A transaction that has completed the address transfer or leading write data transfer, but has not completed all of its data transfers.

Daughterboard Configuration Controller
A microcontroller on a Versatile Express LogicTile or CoreTile daughterboard. It controls the power up and configuration of the daughterboard in conjunction with the *Motherboard Configuration Controller* (MCC) on the Motherboard Express motherboard.

See also Motherboard Configuration Controller (MCC).

DBGTAP *See* Debug Test Access Port (DBGTAP).

DCC *See* Debug Communications Channel (DCC).

Debug Access Port (DAP)
In an external debugger, a block that acts as a master on a system bus, to provide access to the debug target.

Debug Agent In the RealView Debugger, the Debug Agent provides target-side support for *Running System Debug* (RSD). The Debug Agent can be a software thread or be built into the RTOS. The Debug Agent and RealView Debugger communicate with each other using the DCC. This communication passes data between the debugger and the target using a hardware debug interface, without stopping the program or causing entry to Debug state.

See also Running System Debug (RSD), Debug Communications Channel (DCC).

Debug Communications Channel (DCC)
A channel that carries data between a debugger and debug logic in the target processor. It can do this without stopping the program flow or causing entry to Debug state, but can also be used when the target is in Debug state. The DCC is part of the debug register interface of the target.

Debug configuration In the context of an ARM debugger, a debug configuration defines a debugging environment for the development platform that is accessed through a particular debug interface. Multiple debug configurations can be created for a debug interface, each providing a separate debugging environment to different development platforms, or different debugging environments to the same development platform.

All debug configurations are stored in the main debugger board file. Each configuration might reference one or more BCD files.

See also Board file, Target.

Debug illusion The view of the software being debugged that a debugger presents to its user. The features of the debug illusion include:

- Mapping between assembler code and source code, including displaying assembler and source code simultaneously if required.
- Support for source-level stepping and breakpoints.
- Visibility of the source-level function call stack, even when called functions are generated inline.
- Display of variable values and structure field values, even when these values migrate between various locations. This includes displaying registers and the stack.

Debug interface In the context of RealView Debugger, the debug interface identifies the targets on your development platform, and provides the mechanism that a RealView Debugger uses to communicate with those targets. The debug interface corresponds directly to a piece of hardware or a software simulator.

See also Debug configuration, Target.

Debug Test Access Port (DBGTAP)
A debug control and data interface based on IEEE 1149.1 JTAG *Test Access Port* (TAP). The interface has four or five signals.

Debugger
A debugging system that includes a program, used to detect, locate, and correct software faults, together with custom hardware that supports software debugging.

Default NaN mode
In floating-point operation, a mode in which all operations that result in a NaN return the default NaN, regardless of the cause of the NaN result. This mode is compliant with the IEEE 754 standard but implies that all information contained in any input NaNs to an operation is lost.

See also NaN.

Denormalized value
See Subnormal value.

Design Simulation Model (DSM)
A functional simulation model of the device derived from the RTL but that does not reveal its internal structure. The DSM does not model any features added during synthesis such as internal scan chains.

Development platform
Contains the components, either hardware or simulated, that you use to develop your application. It can include:

- A development board, such as an Integrator or CP.
- Peripherals.
- One or more processors that implement the ARM architecture.
- CoreSight components.
- One or more *Digital Signal Processors* (DSPs).

See also CoreSight.

Development Studio 5 (DS-5)
The suite of software development tools, together with supporting documentation and examples, that you can use to write and debug applications for ARM processors. DS-5 supersedes RealView Development Suite.

See also RealView Development Suite (RVDS).

Device
In the context of an ARM debugger, a component on a target. The device contains the application that you want to debug.

See also Target.

Device Validation Suite (DVS)
A set of tests that you can use to check the functionality of a device against that defined in the Technical Reference Manual.

Direct-mapped cache
A one-way set-associative cache. Each cache set consists of a single cache line, so cache look-up selects and checks a single cache line.

Direct read
A direct read of a System register is a read performed by a System register access instruction.

For more information see the appropriate *ARM Architecture Reference Manual*.

See also Direct write, Indirect read, Indirect write.

Direct write
A direct write of a System register is a write performed by a System register access instruction.

For more information see the appropriate *ARM Architecture Reference Manual*.

See also Direct read, Indirect read, Indirect write.

Dirty
A line in a write-back cache that has been modified while it is in the cache. Typically, a cache line is marked as dirty by setting the dirty bit to 1.

See also Clean.

DNM *See* Do-Not-Modify (DNM).

Do-Not-Modify (DNM) A value that must not be altered by software. A DNM field read as an UNKNOWN value, and must only be written with the value read from the same field on the same core.

Doubleword A 64-bit data item. Doublewords are normally at least word-aligned in ARM systems.

Doubleword-aligned A data item having a memory address that is divisible by eight.

Draw mode In the context of graphics processing, a draw mode is one of the different ways to specify the primitives to draw. The primitives can be specified individually or as a connected strip or fan. They can also be either:

- Non-indexed, meaning that vertices are passed in a vertex array and processed in order.
- Indexed, meaning that vertices are passed as indices into a vertex array.

DS-5 Debugger An ARM software development tool that enables you to make use of a debug agent to examine and control the execution of software running on a debug target. It is fully integrated into Eclipse for DS-5.

See also Eclipse for DS-5.

DSM *See* Design Simulation Model (DSM).

DVS *See* Device Validation Suite (DVS).

Early-Z In a Mali GPU implementation, a Z-testing scheme that performs the actual Z-test before texturing or fragment shading when it is safe to do so. This increases the performance of the Mali GPU and reduces the required bandwidth.

Eclipse for DS-5 Eclipse for DS-5 is based around the Eclipse IDE, and provides additional features to support the ARM development tools provided in DS-5.

See also Development Studio 5 (DS-5).

ECT *See* Embedded Cross Trigger (ECT).

EGL *See* Embedded-System Graphics Library (EGL).

ELR *See* Link Register (LR).

Embedded assembler Embedded assembler is assembler code that is included in a C or C++ file, and is separate from other C or C++ functions.

Embedded Cross Trigger (ECT) A modular system that supports the interaction and synchronization of multiple triggering events with an SoC. It comprises:

- *Cross Trigger Interface* (CTI).
- *Cross Trigger Matrix* (CTM).

Embedded Trace Buffer (ETB) A logic block that extends the information capture functionality of a trace macrocell.

Embedded Trace Macrocell (ETM) A hardware macrocell that, when connected to a core, outputs trace information on a trace port. The ETM provides core-driven trace through a trace port compliant to the ATB protocol. An ETM always supports instruction trace, and might support data trace.

EmbeddedICE logic An on-chip logic block that provides TAP-based debug support for an ARM core. It is accessed through the DAP on the ARM processor.

EmbeddedICE-RT Hardware provided by an ARM core to aid debugging in real-time.

Embedded-System Graphics Library (EGL)
A standardized set of functions that communicate between graphics software, such as OpenGL ES or OpenVG drivers, and the platform-specific windowing system that displays the image.

Emulator
In the context of target connection hardware, an emulator provides an interface to the pins of a real processor. It emulates the pins to the external world, and enables you to control or manipulate signals on those pins.

Endianness
The scheme that determines the order of the successive bytes of data in a larger data structure when that structure is stored in memory.

See also Big-endian, Little-endian.

ESSL compiler
The compiler that translates shaders, written in ESSL, into binary code for the shader units in the Mali GPU. There are two versions of the ESSL compiler:

- The on-target compiler.
- The offline compiler.

ETB
See Embedded Trace Buffer (ETB).

ETM
See Embedded Trace Macrocell (ETM).

ETV
See Extended Target Visibility (ETV).

Event
In an ARM trace macrocell:

Simple An observable condition that a trace macrocell can use to control aspects of a trace.

Complex A boolean combination of simple events that a trace macrocell can use to control aspects of a trace.

Event asynchronous bridge
A fixed component that synchronizes events on a single channel from the slave domain to the master domain. In addition, the event acknowledge from the master domain is synchronized and signaled to the slave domain.

Exception
A mechanism to handle a fault, error event, or external notification. For example, exceptions handle external interrupts and undefined instructions.

Exception Link Register (ELR)
See Link Register (LR).

Exception vector
A fixed address that contains the address of the first instruction of the corresponding exception handler.

Exceptional state
In an ARMv7 implementation that includes a VFP subarchitecture, in floating-point operation, if the floating-point hardware detects an exceptional condition, the ARM floating-point implementation sets the FPEXC bit and loads a copy of the exceptional instruction to the FPINST register. When in the exceptional state, the issue of a trigger instruction to the floating-point extension causes a bounce.

See also Bounce, Trigger instruction.

Execution vehicle
A part of the debug target interface that processes requests from the client tools to the target.

See also Debug interface.

Execution view
The address of regions and sections after the image is loaded into memory and execution has started.

See also Scatter-loading, Load view.

Explicit access
A read from memory, or a write to memory, generated by a load or store instruction executed by the core. Reads and writes generated by hardware translation table accesses are not explicit accesses. For more information, see the appropriate ARM *Architecture Reference Manual*.

Extended Target Visibility (ETV)
Extended Target Visibility enables RealView Debugger to access features of the underlying target such as, for example, chip-level information provided by the hardware manufacturer or SoC designer.

Extensible Verification Component (XVC)
A model that provides system or device stimulus and monitor responses.

See also XVC Test Scenario Manager (XTSM).

External Abort
An abort generated by the external memory system.

See also Abort, Data Abort, Prefetch Abort.

Fast Context Switch Extension (FCSE)
Before ARMv8, an extension to the ARM architecture that modifies the behavior of the memory system. It enables multiple programs running on the core to use identical address ranges, while ensuring that the addresses they present to the rest of the memory system differ.

From ARMv6, ARM deprecates use of the FCSE. The FCSE is optional in ARMv7, and obsolete from the ARMv7 Multiprocessing Extensions.

Fast Models from ARM
Instruction-accurate models for early software development on ARM systems. You can use the models, together with ARM Profiler and an ARM debugger, to optimize and debug your applications early in the development cycle.

See also ARM profiler, DS-5 Debugger.

Fault
An abort generated by the memory system, for example by the *Memory Management Unit* (MMU) or *Memory Protection Unit* (MPU).

FCSE
See Fast Context Switch Extension (FCSE).

FIQ
FIQ interrupt. **nFIQ** is one of two interrupt signals on many ARM processors.

See also IRQ.

Flash Patch and Breakpoint unit (FPB)
In an ARM M-profile processor, a FPB can:

- Remap sections of ROM, typically Flash memory, to regions of RAM
- Set breakpoints on code in ROM.

You can use the FPB for debug, and to provide a code or data patch to an application that requires field updates to a product ROM.

Flat address mapping
A memory system where the physical address for every access is equal to its virtual address.

Flush-to-zero mode
In floating-point operation, a special processing mode that optimizes the performance of some floating-point algorithms by replacing the denormalized operands and intermediate results with zeros, without significantly affecting the accuracy of their final results.

Formatter
In an ETB or TPIU, an internal input block that embeds the trace source ID in the data to create a single trace stream.

FPB
See Flash Patch and Breakpoint unit (FPB).

Fragment
In the context of graphics processors, a fragment consists of all data, such as depth, stencil, texture, and color information, required to generate a pixel in the framebuffer. A pixel usually comprises several fragments.

Fragment processor
In the context of graphics processors, a programmable processor that performs rendering operations to produce a final image for display. The fragment processor receives completed vertex data from the vertex processor and then runs fragment shader programs.

The fragment processor was originally called the pixel processor.

Fragment shader
A program running on the fragment processor that calculates the color and other characteristics of each fragment.

Fragment Thread Creator
In a Mali GPU, a functional block that creates and issues threads to the tri-pipe processing unit pipeline. It is used for all fragment jobs output from the tiler.

See also Generic Thread Creator.

fromelf
The ARM image conversion utility. This accepts ELF format input files and converts them to a variety of output formats. fromelf can also generate text information about the input image, such as code and data size.

See also RealView Compilation Tools (RVCT).

Fully-associative cache
A cache that has only one cache set, that consists of the entire cache.

See also Direct-mapped cache.

General-purpose register
See Core register.

Generic Thread Creator
In a Mali GPU, a functional block that creates and issues threads to the tri-pipe processing unit pipeline. It is used for all non-fragment jobs, including vertex shading, geometry shading, and OpenCL jobs.

See also Fragment Thread Creator.

Granular Power Requester
Used by a debugger to control powerup and powerdown of specific components within a CoreSight system.

Graphics application
A custom program that executes in the Mali GPU system and displays content in a framebuffer for transfer to a display.

Graphics driver
A software library implementing OpenGL ES or OpenVG, using graphics accelerator hardware.

See also OpenGL ES driver, OpenVG driver.

Halfword
A 16-bit data item. Halfwords are normally halfword-aligned in ARM systems.

Halfword-aligned
A data item having a memory address that is divisible by 2.

Halted System Debug (HSD)
In the context of an ARM debugger, means that a target can only be debugged when it is not running. With the target stopped, RealView Debugger presents OS awareness information by reading and interpreting target memory.

See also Running System Debug (RSD).

Halting debug
In ARM A-profile and R-profile processors, in AArch32 state, one of two mutually exclusive debug modes. When Halting debug is enabled, core execution halts when a breakpoint or watchpoint is encountered. You can use the debug interface to examine and alter all core state, memory, input and output locations.

See also Monitor debug.

Hardware breakpoint
In the context of an ARM debugger, a breakpoint that is implemented using non-intrusive hardware. Hardware breakpoints are the only method of halting execution on a specific instruction when the instruction is located in *Read Only Memory* (ROM) or Flash.

See also Chained breakpoint, Data breakpoint.

Head-of-line blocking
In an interconnect, this occurs when a node prevents an important transaction from progressing because a less important transaction blocks the path to the same destination.

See also Quality of Service using Virtual Networks (QVN), Cross-path blocking.

Hierarchical Tiler (HT)
In the context of a Mali GPU, the HT sorts all the primitives in the scene into a hierarchical list structure. These lists are subsequently processed by the shader cores.

High registers
In AArch32 state, core registers R8-R12, SP, LR, and PC. Some T32 instructions cannot access these registers.

See also Core register.

High vectors
In AArch32 state, one of two possible locations for exception vectors. The high vector address range is near the top of the address space, rather than at the bottom.

Hint instruction
A hint instruction provides information that the hardware can take advantage of.

Hit-Under-Miss (HUM)
A buffer that means a memory access can hit in the cache, even though there has been a data miss in the cache.

Host
A computer that provides data and other services to another computer. In the context of an ARM debugger, a computer providing debugging services to a target being debugged.

HSD
See Halted System Debug (HSD).

HT
See Hierarchical Tiler (HT).

HTM
See AHB Trace Macrocell (HTM).

HUM
See Hit-Under-Miss (HUM).

ICE Extension Unit
A hardware extension to the EmbeddedICE logic that provides more breakpoint units.

IGN
An abbreviation for Ignore, when describing the behavior of a register or memory access.

IGNORED
Indicates that the architecture guarantees that the bit or field is not interpreted or modified by hardware.

In body text, the term IGNORED is shown in SMALL CAPITALS.

Illegal instruction
See UNDEFINED.

Immediate values
Values that are encoded directly in the instruction and used as numeric data when the instruction is executed.

IMPLEMENTATION DEFINED
Behavior that is not defined by the architecture, but must be defined and documented by individual implementations.

When IMPLEMENTATION DEFINED appears in body text, it is always in SMALL CAPITALS.

IMPLEMENTATION SPECIFIC

In the context of ARM trace macrocells, behavior that is not architecturally defined, and might not be documented by an individual implementation. Used when there are a number of implementation options available and the option chosen does not affect software compatibility.

When IMPLEMENTATION SPECIFIC is used with this meaning in body text, it is always in SMALL CAPITALS.

See also IMPLEMENTATION DEFINED.

Imprecise tracing

In an ARM trace macrocell, a filtering configuration where instruction or data tracing can start or finish earlier or later than expected. Most imprecise cases cause tracing to start or finish later than expected.

In-Circuit Emulator

A device that provides access to the signals of a circuit while that circuit is operating, and lets you moderate those signals.

Index register

A register specified in some load or store instructions. The value of this register is used as an offset to be added to or subtracted from the base register value to form the address that is sent to memory. Some instruction forms permit the index register value to be shifted before the addition or subtraction.

Indirect read

When an instruction uses a System register value to establish operating conditions, that use of the System register is an indirect read of the System register.

For more information, including additional examples of indirect reads, see the appropriate *ARM Architecture Reference Manual*.

See also Direct read, Direct write, Indirect write.

Indirect write

An indirect write of a System register occurs when the contents of a register are updated by some mechanism other than a *Direct write* to that register. For example, an indirect write to a register might occur as a side effect of executing an instruction that does not perform a direct write to the register, or because of some operation performed by an external agent.

For more information see the appropriate *ARM Architecture Reference Manual*.

See also Direct read, Direct write, Indirect read.

Input section

Contains code or initialized data or describes a fragment of memory that must be set to zero before the application starts.

Instruction Abort

An indication to the core of an attempted instruction fetch that is not permitted. The Instruction Abort might be generated by access permission checks performed by the memory system on the core, or might be signaled by the memory system. An exception is taken only if the core attempts to execute the instruction. No exception is taken if the core does not execute an instruction that it attempted to fetch or prefetch from a faulting memory location.

The AArch64 architecture definitions introduce the term *Instruction Abort*. Descriptions of AArch32 state use the term *Prefetch Abort*.

See also Abort, Data Abort, External Abort, Prefetch Abort.

Instruction breakpoint

In the context of an ARM debugger, a location in the image containing an instruction that, if executed, activates a breakpoint. The breakpoint activation can be delayed by assigning condition qualifiers, and subsequent execution of the image is determined by any actions assigned to the breakpoint.

See also Conditional breakpoint, Hardware breakpoint, Software breakpoint.

Instruction Set System Model (ISSM)
In the context of RVDS, a set of models that simulate the ARM Cortex family of processors. These models are provided with RVDS.

See also Real Time System Model (RTSM), Simulator.

Instruction Synchronization Barrier (ISB)
An operation to ensure that any instruction that comes after the ISB operation is fetched only after the ISB has completed. For more information, see the appropriate ARM *Architecture Reference Manual*.

Instrumentation trace A component for debugging real-time systems through a simple memory-mapped trace interface. It provides `printf`-style debugging.

Integrator A range of ARM hardware development platforms. Core modules are available that contain the processor and local memory.

See also Core module.

Intelligent Energy Manager
An energy manager solution consisting of both software and hardware components that function together to prolong battery life in an ARM processor based device.

Intermediate Physical Address (IPA)
In an implementation of virtualization, the address to which a Guest OS maps a virtual address.

See also Virtual Address (VA), Physical Address (PA).

Intermediate result In floating-point operation, an internal format used to store the result of a calculation before rounding. This format can have a larger exponent field and fraction field than the destination format.

Internal scan chain A series of registers connected together to form a path through a device, used during production testing to import test patterns into internal nodes of the device and export the resulting values.

See also Boundary scan chain.

Interworking In AArch32 state, a method of working that permits branches between software using the A32 and T32 instruction sets.

Invalidate Marking a cache line as being not valid. This must be done whenever the line does not contain a valid cache entry. For example, after a cache flush all lines are invalid.

IPA *See* Intermediate Physical Address (IPA).

IRQ IRQ interrupt. **nIRQ** is one of two interrupt signals on many ARM processors.

See also FIQ.

ISB *See* Instruction Synchronization Barrier (ISB).

ISSM *See* Instruction Set System Model (ISSM).

IT block In AArch32 state, a block of up to four instructions following a T32 `IT` (If-Then) instruction. Each instruction in the block is conditional. The condition for each instruction is either the same as or the inverse of the condition specified by the `IT` instruction.

Jazelle DBX *See* Jazelle Extension.

Jazelle Extension Some ARM processors before ARMv8 included the Jazelle Extension, to provide hardware execution of some Java bytecodes in AArch32 state, as part of a *Java Virtual Machine* (JVM) implementation. From ARMv8, the architecture supports only an AArch32 Trivial Jazelle implementation, meaning a JVM must be implemented entirely in software.

The Jazelle Extension technology is called Jazelle DBX.

Jazelle RCT (Runtime Compilation Target)
On an ARM processor, a modification of the T32 instruction set to make it a better target for code generated at runtime. This is also called the T32 Execution Environment (T32EE).

From ARMv8, ARM processors do not support Jazelle RCT.

See also T32EE instruction.

Jazelle state In AArch32 state, in the Jazelle Instruction set state the core executes Java bytecodes as part of a *Java Virtual Machine* (JVM).

From ARMv8, ARM processors do not support Jazelle state.

See also A32 state, T32 state.

Jazelle Technology Enabling Kit (JTEK)
A kit containing source code for integration with a Java Virtual Machine to enable Jazelle DBX on an ARM-based host platform.

Job object In the context of graphics processing, a Mali job system component that provides jobs with required content for Mali GPU execution.

Job system back-end In the context of graphics processing, a Mali job system component that shares some priority handling, but mainly requests jobs from queues and sends job requests to the Mali GPU.

Joint Test Action Group (JTAG)
An IEEE group focussed on silicon chip testing methods. Many debug and programming tools use a *Joint Test Action Group* (JTAG) interface port to communicate with processors.

See *IEEE Std 1149.1-1990 IEEE Standard Test Access Port and Boundary-Scan Architecture* specification available from the IEEE Standards Association, `http://standards.ieee.org`.

JTAG *See* Joint Test Action Group (JTAG).

JTAG Access Port (JTAG-AP)
An optional component of the DAP that provides debugger access to on-chip scan chains.

JTAG-AP *See* JTAG Access Port (JTAG-AP).

JTAG-DP *See* Debug Access Port (DAP).

JTAG interface unit In the context of ARM RealView tools, a protocol converter that converts low-level commands from RVDS debuggers into JTAG signals to the processor, for example to the EmbeddedICE logic and the ETM.

See also Development Studio 5 (DS-5), RealView ICE.

JTEK *See* Jazelle Technology Enabling Kit (JTEK).

K Virtual Machine (KVM)
A small implementation of a Java Virtual Machine. It was originally derived from the Sun Spotless Virtual Machine.

KVM *See* K Virtual Machine (KVM).

Link Register (LR) On ARM processors, LR refers to the Link Register for exception returns, and:

- In AArch32 state, the LR is register R14 in the general-purpose register file.

- In AArch64 state, there is a dedicated LR for each implemented Exception level, called the *Exception Link Register* (ELR) for that Exception level, for example, ELR_EL1.

See also Program Counter (PC), Stack Pointer (SP).

Little-endian In the context of the ARM architecture, little-endian is defined as the memory organization in which the most significant byte of a word is at a higher address than the least significant byte.

See also Big-endian.

Load view The address of regions and sections when the image has been loaded into memory, before execution has started.

See also Execution view, Scatter-loading.

Load/store architecture
A processor architecture where data processing operations only operate on register contents, not directly on memory contents. The ARM architecture is a Load/Store architecture.

Low-Power Interface (LPI)
An interface that can provide control of the clock and power states of a device.

LR *See* Link Register (LR).

Mali DDK A set of drivers, typically for AEL or Android, that enable communication with the Mali GPU. These drivers are available as normal or instrumented versions.

Mali MMU A full-featured *Memory Management Unit* (MMU) that is present on Mali GPUs.

Memory coherency A memory system is coherent if the value read by a data read or instruction fetch is always the value that was most recently written to that location. Memory coherency is difficult when the memory system includes multiple possible physical locations, such as main memory and at least one of a write buffer or one or more caches.

See the appropriate ARM *Architecture Reference Manual* for a more extensive and more rigorous definition of memory coherency.

See also Coherent.

Memory hint *See* Hint instruction.

Memory Management Unit (MMU)
Provides detailed control of the memory system. Most of the control uses translation tables that are held in memory. An MMU is the major component of an ARM *Virtual Memory System Architecture* (VMSA).

Memory Protection Unit (MPU)
A hardware unit that controls a limited number of protection regions in memory. An MPU is the major component of an ARM *Protected Memory System Architecture* (PMSA).

MMU *See* Memory Management Unit (MMU).

Model manager A software control manager that handles the event transactions between the model and simulator.

Modified Virtual Address (MVA)
The address produced by the FCSE that is sent to the rest of the memory system to be used in place of the normal virtual address.

If the FCSE is absent or disabled, the MVA and the *Virtual Address* (VA) have the same value. From ARMv6, ARM deprecates any use of the FCSE. The FCSE is optional in the unextended ARMv7 architecture, and obsolete from the introduction of the Multiprocessing Extensions.

See also Fast Context Switch Extension (FCSE).

Monitor debug

In ARM A-profile and R-profile processors, in AArch32 state, one of two mutually exclusive debug modes. When Monitor debug is enabled, a debug event generates a debug exception, that is taken as a Prefetch Abort or Data Abort exception. Breakpoints and watchpoints are examples of debug events.

See also Halting debug.

Motherboard Configuration Controller (MCC)

A microcontroller on Versatile Express motherboards. It controls the power up and configuration of the Versatile Express development system, which consists of the motherboard and fitted CoreTile and LogicTile Express daughterboards. The MCC configures the system in conjunction with the Daughterboard Configuration Controllers.

See also Daughterboard Configuration Controller.

MPCore

An integrated *Symmetric Multiprocessor System* (SMP) or *Asymmetric Multiprocessor System* (AMP) with multiple cores in a single device.

MPU

See Memory Protection Unit (MPU).

Multi-ICE

A JTAG-based tool for debugging embedded systems.

Multi-layer interconnect

An interconnect scheme similar to a cross-bar switch. Each master on the interconnect has a its own direct link to each slave, meaning that link is not shared with other masters. This means each master can process transfers in parallel with other masters. Contention in a multi-layer interconnect only occurs at a payload destination, typically a slave.

Multi-master AHB

Typically a shared, not multi-layer, AHB interconnect scheme. More than one master connects to a single AMBA AHB link. In this case, the bus is implemented with a set of full AMBA AHB master interfaces. Masters that use the AMBA AHB-Lite protocol must connect through a wrapper to supply full AMBA AHB master signals to support multi-master operation.

MVA

See Modified Virtual Address (MVA).

NaN

Not a number. In floating-point operation, NaNs are special floating-point values that can be used when neither a numeric value nor an infinity is appropriate. NaNs can be either:

- Quiet NaNs that propagate through most floating-point operations.
- Signaling NaNs that cause Invalid Operation floating-point exceptions.

For more information, see the IEEE 754 standard.

NEON technology

The ARM technology that provides SIMD processing using a dedicated SIMD and floating-point register bank. Registers in this bank can be accessed as 128-bit registers, 64-bit registers, 32-bit registers, 16-bit registers, or 8-bit registers.

See also Advanced SIMD.

Normal and Secure Worlds

In software descriptions of the operation of an ARM core, effectively the environments that contain two virtual processors that run on a single core. The Secure World processes operations that are security-critical, and non security-critical operations are performed in the Normal World.

Hardware descriptions use Secure state to describe a core that is executing in the Secure World, and Non-secure state to describe a core that is executing in the Non-secure state.

See also Secure monitor.

Normal World *See* Normal and Secure Worlds.

nSRST Abbreviation of *System Reset*. The signal that causes the target system other than the TAP controller to be reset. This signal is known as **nSYSRST** in some documentation.

See also nTRST, Joint Test Action Group (JTAG).

nTRST Abbreviation of *TAP Reset*. The signal that causes the target system TAP controller to be reset. This signal is known as **nICERST** in some documentation.

See also nSRST, Joint Test Action Group (JTAG).

Offline compiler A command line tool that translates vertex shaders and fragment shaders written in the ESSL into binary vertex shaders and binary fragment shaders that you can link and run on the Mali GPU.

Offset addressing Addressing where the memory address is formed by adding an offset to, or subtracting an offset from, a base register value.

On-target compiler A component of the Mali GPU OpenGL ES 2.0 driver that translates shader source files provided by the graphics application, into binary shader code, at runtime.

OpenGL ES driver Part of a driver stack that translates OpenGL ES API commands into data and instructions for the Mali GPU. Only the device driver controls the Mali GPU directly.

OpenVG driver Part of a driver stack that translates OpenVG API commands into data and instructions for the Mali GPU. Only the device driver controls the Mali GPU directly.

OS-awareness OS-awareness is a feature provided by RealView Debugger that enables you to:

- Debug applications running on an embedded OS development platform, such as a *Real-Time Operating System* (RTOS).
- Present thread information and scope some debugging operations to specific threads.

OTC *See* Over The Cell (OTC).

Output section A contiguous sequence of input sections that have the same RO, RW, or ZI attributes. The sections are grouped together in larger fragments called regions. The regions are grouped together into the final executable image.

See also Region.

Over The Cell (OTC) A power routing scheme, also referred to as ArtiGrid.

See also ArtiGrid.

PA *See* Physical Address (PA).

PCH *See* PreCompiled Header (PCH).

Page table In the ARM processor architecture, a page table is called a translation table, and a page table walk is called a translation table walk.

See also Translation table, Translation table walk.

Page table walk In the ARM processor architecture, a page table is called a translation table, and a page table walk is called a translation table walk.

See also Translation table, Translation table walk.

PE *See* Processing Element (PE).

Penalty The number of cycles in which no useful Execute stage pipeline activity can occur because an instruction flow is different from that assumed or predicted.

Physical Address (PA) The address that identifies a location in physical memory.

PISMO *See* Platform Independent Storage Module (PISMO).

Platform Independent Storage Module (PISMO)
Memory specification for plug-in memory modules.

PLI *See* Programming Language Interface (PLI).

POP A performance optimization package for the implementation of an ARM processor using ARM Artisan optimized logic and memory physical IP.

Power Management Module (PMM)
In the context of graphics processors, a software routine that tracks the hardware blocks that can be enabled or disabled to reduce power. The PMM can control a specialized hardware unit, or a third-party power-management device, to power up or down each processor separately.

Power-on reset *See* Cold reset.

Powerup reset *See* Cold reset.

PreCompiled Header (PCH)
A header file that is precompiled. This avoids the compiler having to compile the file each time it is included by source files.

Prefetch Abort The AArch32 name for an *Instruction Abort*.

See also Abort, Data Abort, External Abort, Instruction Abort.

Prefetching The process of fetching instructions from memory before the instructions that precede them have finished executing. Prefetching an instruction does not mean that the instruction must be executed.

Primitive In the context of graphics processors, a basic element that the Mali GPU uses, with other primitives, to generate images.

See also Vertex.

Procedure Call Standard for the ARM Architecture (AAPCS)
Defines how registers and the stack are used for subroutine calls.

Profiling In the context of RealView Trace, the accumulation of statistics during execution of a program to measure performance or to determine critical areas of code.

Processing Element (PE)
The abstract machine defined in the ARM architecture, as documented in an ARM *Architecture Reference Manual*. A PE implementation that is compliant with the ARM architecture must conform with the behaviors described in the corresponding ARM Architecture *Reference Manual*.

ARM processor documentation usually describes a PE as a Core, and this is the term used in this glossary.

See also Core.

Processor

A general term for an entity that performs processing. Often used to describe the set of *Cores* that are marketed as a single product, for example a four-core ARM Cortex-A53 MPCore processor.

Older ARM documentation uses the term *processor* to refer to a *Core* or a *Processing Element (PE)*.

See also Core, Processing Element (PE).

Program Counter (PC)

The PC holds the virtual address of the next instruction to be executed, and:

- In AArch32 state, the PC is a 32-bit register, and is register R15 in the general-purpose register file.
- In AArch64 state, the PC is a 64-bit register, that is independent of the general-purpose registers X0-X30. In AArch64 state, software cannot write directly to the PC.

See also Link Register (LR), Stack Pointer (SP).

Program Flow Trace (PFT)

The *Program Flow Trace* (PFT) architecture assumes that any trace decompressor has a copy of the program being traced, and generally outputs only enough trace for the decompressor to reconstruct the program flow. However, its trace output also enables a decompressor to reconstruct the program flow when it does not have a copy of parts of the program, for example because the program uses self-modifying code.

A *Program Flow Trace Macrocell* (PTM) implements the Program Flow Trace architecture.

Program Status Register (PSR)

Holds status and control information for a core, or for a program running on the core. When executing in AArch32 state, the *Current Program Status Register* (CPSR) is the active PSR that affects operation of the core. When executing in AArch64 state, PSTATE holds equivalent status information, but is not accessible as a single register.

The *Saved Program Status Register* (SPSR) is a copy of the current state of the cores saved by the hardware on taking an exception. At the point where a core recognizes an exception:

- If the exception is taken to AArch32 state, it copies the CPSR into the SPSR.
- If the exception is taken to AArch64 state, it saves the current PSTATE to the SPSR.

See the appropriate ARM *Architecture Reference Manual* for more information.

See also Current Program Status Register (CPSR), PSTATE, Saved Program Status Register (SPSR).

Programming Language Interface (PLI)

For Verilog simulators, an interface by which foreign code can be included in a simulation. Foreign code is code written in a different language.

Project template

A collection of configuration files for specific target development platforms. These templates enable you to create a target-specific development project in the ARM Workbench IDE.

See also ARM Workbench IDE (AWIDE), RealView Compilation Tools (RVCT).

Protection region

A memory region whose position, size, and other properties are defined by the Memory Protection Unit registers.

See also Memory Protection Unit (MPU).

Protection Unit (PU)

See Memory Protection Unit (MPU).

PSR

See Program Status Register (PSR).

PSTATE In ARMv8, an abstraction of process state. All of the instruction sets include instructions that operate directly on elements of PSTATE. In previous versions of the architecture, the CPSR provides the equivalent functionality.

See also Program Status Register (PSR).

PU *See* Protection Unit (PU).

Quadword A 128-bit data item. Quadwords are normally at least word-aligned in ARM systems.

Quadword-aligned A data item having a memory address that is divisible by 16.

Quality of Service using Virtual Networks (QVN)
A protocol to avoid head-of-line blocking and cross-path blocking between different data flows.

See also Head-of-line blocking, Cross-path blocking.

QVN *See* Quality of Service using Virtual Networks (QVN).

RAO *See* Read-As-One (RAO).

RAO/SBOP Read-As-One, Should-Be-One-or-Preserved on writes.

Hardware must implement the field as Read-as-One, and must ignore writes to the field. Software can rely on the field reading as all 1s, but must use an SBOP policy to write to the field. This description can apply to a single bit that reads as 1, or to a field that reads as all 1s.

See also Read-As-One (RAO), Should-Be-One-or-Preserved (SBOP).

RAO/WI Read-As-One, Writes Ignored.

Hardware must implement the field as Read-as-One, and must ignore writes to the field. Software can rely on the field reading as all 1s, and on writes being ignored. This description can apply to a single bit that reads as 1, or to a field that reads as all 1s.

See also Read-As-One (RAO).

RAZ *See* Read-As-Zero (RAZ).

RAZ/SBZP Read-As-Zero, Should-Be-Zero-or-Preserved on writes.

Hardware must implement the field as Read-as-Zero, and must ignore writes to the field. Software can rely on the field reading as all 0s, but must use an SBZP policy to write to the field. This description can apply to a single bit that reads as 0, or to a field that reads as all 0s.

See also Read-As-Zero (RAZ), Should-Be-Zero-or-Preserved (SBZP).

RAZ/WI Read-As-Zero, Writes Ignored.

Hardware must implement the field as Read-as-Zero, and must ignore writes to the field. Software can rely on the field reading as all 0s, and on writes being ignored. This description can apply to a single bit that reads as 0, or to a field that reads as all 0s.

See also Read-As-Zero (RAZ).

Read A memory operation that has the semantics of a load. See the *ARM Architecture Reference Manual* for more information.

Read-As-One (RAO) Hardware must implement the field as reading as all 1s. Software can rely on the field reading as all 1s. This description can apply to a single bit that reads as 1, or to a field that reads as all 1s.

Read-As-Zero (RAZ) Hardware must implement the field as reading as all 0s. Software can rely on the field reading as all 0s. This description can apply to a single bit that reads as 0, or to a field that reads as all 0s.

Read, modify, write
In a read, modify, write sequence, a value is read to a general-purpose register, the relevant fields updated in that register, and the new value written back.

Read-Only Position Independent (ROPI)
In the context of software executing on a core that implements the ARM architecture, code or read-only data that can be placed at any address.

Read Write Position Independent (RWPI)
In the context of software executing on a core that implements the ARM architecture, read-write code or data that can be placed at any address.

Real Time System Model (RTSM)
A software model of a development system, for example, the Emulation Baseboard. The model can run applications at almost full speed. This enables applications and operating systems to be written and debugged without a requirement for actual hardware.

RealMonitor
A small program that, when integrated into your target application or *Real-Time Operating System* (RTOS), enables you to observe and debug your target while parts of your application continue to run.

RealView Compilation Tools (RVCT)
A suite of tools that, together with supporting documentation and examples, enables you to write and build applications for ARM processors.

See also armcc, armasm.

RealView Debugger
An ARM debugger that enables you to examine and control the execution of software running on a debug target. RealView Debugger is supplied as part of RVDS in both Windows and Red Hat Linux versions.

RealView Debugger Trace
Part of RVDS that extends the debugging capability with the addition of real-time program and data tracing. It is available from the RealView Debugger Code window.

See also RealView Debugger Trace, RealView ICE.

RealView Development Suite (RVDS)
The suite of software development tools, together with supporting documentation and examples, that enable you to write and debug applications for ARM processors.

See also ARM Compiler for DS-5, ARM profiler, Eclipse for DS-5, Development Studio 5 (DS-5), RealView ICE, RealView Trace and RealView Trace 2.

RealView ICE
An ARM JTAG interface unit for debugging embedded processor cores that uses a DBGTAP or Serial Wire interface.

RealView Instruction Set Simulator (RVISS)
One of the ARM simulators supplied with RVDS. RVISS is a collection of programs that simulate the instruction sets and architecture of various ARM processors. This provides instruction-accurate simulation and enables A32 and T32 executable programs to be run on non-native hardware. RVISS provides modules that model:

- The ARM core.
- The memory used by the core.

There are alternative predefined models for each of these parts. However, you can create your own models if a supplied model does not meet your requirements.

See also Instruction Set System Model (ISSM), Real Time System Model (RTSM).

RealView Trace and RealView Trace 2
Work in conjunction with RealView ICE to provide real-time trace functionality for software running in System-on-Chip devices with deeply embedded ARM processors. RealView Trace 2 also supports data streaming directly to ARM Profiler, providing real-time hardware platform profiling.

See also RealView Debugger Trace, RealView ICE.

Region
In an image, a region is a contiguous sequence of one to three output sections (RO, RW, and ZI). A region typically maps onto a physical memory device, such as ROM, RAM, or peripherals.

See also Root region.

Register slice
An AMBA component that comprises a slave interface connected directly to a master interface with the same set of pins. A register slice regenerates the AMBA signal but does not have any other AMBA functionality. It adds a delay of one or more clock cycles to the signal.

In an AMBA interconnect, adding a register slice to one channel does not require the addition of a register slice to any other channel.

Remapping
Changing the address of physical memory or devices after an application has started executing. This might be done to permit RAM to replace ROM when the initialization has completed.

Replicator
In an ARM trace macrocell, enables two trace sinks to be wired together and to operate independently on the same incoming trace stream. The input trace stream is output onto two independent ATB ports.

RES0
A reserved bit or field with *Should-Be-Zero-or-Preserved (SBZP)* behavior. Used for fields in register descriptions, and for fields in architecturally-defined data structures that are held in memory, for example in translation table descriptors.

Note RES0 is not used in descriptions of instruction encodings.

Within the architecture, there are some cases where a register bit or bitfield:
- Is RES0 in some defined architectural context.
- Has different defined behavior in a different architectural context.

This means the definition of RES0 for register fields is:

If a bit is RES0 in all contexts

It is IMPLEMENTATION DEFINED whether:

1. The bit is hardwired to 0. In this case:
   - Reads of the bit always return 0.
   - Writes to the bit are ignored.

   The bit might be described as RES0, WI, to distinguish it from a bit that behaves as described in 2.
2. The bit can be written. In this case:
   - An indirect write to the register sets the bit to 0.
   - A read of the bit returns the last value successfully written to the bit.

     Note As indicated in this list, this value might be written by an indirect write to the register.

     If the bit has not been successfully written since reset, then the read of the bit returns the reset value if there is one, or otherwise returns an UNKNOWN value.

- A direct write to the bit must update a storage location associated with the bit.
- The value of the bit must have no effect on the operation of the core, other than determining the value read back from the bit.

Whether RES0 bits or fields follow behavior 1 or behavior 2 is IMPLEMENTATION DEFINED on a field-by-field basis.

If a bit is RES0 only in some contexts

When the bit is described as RES0:

- An indirect write to the register sets the bit to 0.
- A read of the bit must return the value last successfully written to the bit, regardless of the use of the register when the bit was written.

  Note As indicated in this list, this value might be written by an indirect write to the register.

  If the bit has not been successfully written since reset, then the read of the bit returns the reset value if there is one, or otherwise returns an UNKNOWN value.
- A direct write to the bit must update a storage location associated with the bit.
- While the use of the register is such that the bit is described as RES0, the value of the bit must have no effect on the operation of the core, other than determining the value read back from that bit.

For any RES0 bit, software:

- Must not rely on the bit reading as 0.
- Must use an SBZP policy to write to the bit.

The RES0 description can apply to bits or bitfields that are read-only, or are write-only:

- For a read-only bit, RES0 indicates that the bit reads as 0, but software must treat the bit as UNKNOWN.
- For a write-only bit, RES0 indicates that software must treat the bit as SBZ.

This RES0 description can apply to a single bit that should be written as its preserved value or as 0, or to a field that should be written as its preserved value or as all 0s.

In body text, the term RES0 is shown in SMALL CAPITALS.

See also Read-As-Zero (RAZ), Should-Be-Zero-or-Preserved (SBZP), UNKNOWN.

RES1

A reserved bit or field with *Should-Be-One-or-Preserved (SBOP)* behavior. Used for fields in register descriptions, and for fields in architecturally-defined data structures that are held in memory, for example in translation table descriptors.

Note RES1 is not used in descriptions of instruction encodings.

Within the architecture, there are some cases where a register bit or bitfield:

- Is RES1 in some defined architectural context.
- Has different defined behavior in a different architectural context.

This means the definition of RES1 for register fields is:

If a bit is RES1 in all contexts

It is IMPLEMENTATION DEFINED whether:

1. The bit is hardwired to 1. In this case:
   - Reads of the bit always return 1.

- Writes to the bit are ignored.

The bit might be described as RES1, WI, to distinguish it from a bit that behaves as described in 2.

2. The bit can be written. In this case:
   - An indirect write to the register sets the bit to 1.
   - A read of the bit returns the last value successfully written to the bit.

     Note As indicated in this list, this value might be written by an indirect write to the register.

     If the bit has not been successfully written since reset, then the read of the bit returns the reset value if there is one, or otherwise returns an UNKNOWN value.
   - A direct write to the bit must update a storage location associated with the bit.
   - The value of the bit must have no effect on the operation of the core, other than determining the value read back from the bit.

Whether RES1 bits or fields follow behavior 1 or behavior 2 is IMPLEMENTATION DEFINED on a field-by-field basis.

If a bit is RES1 only in some contexts

When the bit is described as RES1:

- An indirect write to the register sets the bit to 1.
- A read of the bit must return the value last successfully written to the bit, regardless of the use of the register when the bit was written.

  Note As indicated in this list, this value might be written by an indirect write to the register.

  If the bit has not been successfully written since reset, then the read of the bit returns the reset value if there is one, or otherwise returns an UNKNOWN value.
- A direct write to the bit must update a storage location associated with the bit.
- While the use of the register is such that the bit is described as RES1, the value of the bit must have no effect on the operation of the core, other than determining the value read back from that bit.

For any RES1 bit, software:

- Must not rely on the bit reading as 1.
- Must use an SBOP policy to write to the bit.

The RES1 description can apply to bits or bitfields that are read-only, or are write-only:

- For a read-only bit, RES1 indicates that the bit reads as 1, but software must treat the bit as UNKNOWN.
- For a write-only bit, RES1 indicates that software must treat the bit as SBO.

This RES1 description can apply to a single bit that should be written as its preserved value or as 1, or to a field that should be written as its preserved value or as all 1s.

In body text, the term RES1 is shown in SMALL CAPITALS.

See also Read-As-One (RAO), Should-Be-One-or-Preserved (SBOP), UNKNOWN.

Reserved

Unless otherwise stated in the architecture or product documentation:

- Reserved instruction and 32-bit system control register encodings are UNPREDICTABLE.

- Reserved 64-bit system control register encodings are undefined.
- Reserved register bit fields are UNK/SBZP.

RM An ARM abbreviation for *Round towards Minus Infinity* rounding mode, see *Rounding mode*.

RN An ARM abbreviation for *Round to Nearest* rounding mode, see *Rounding mode*.

Root region In an image, regions having the same load and execution address. A non-root region is a region that must be copied from its load address to its execution address.

See also Region.

ROPI *See* Read-Only Position Independent (ROPI).

Rounding error Is defined to be the value of the rounded result of an arithmetic operation minus the exact result of the operation.

See also Rounding mode.

Rounding mode In floating-point operation, specifies how the exact result of a floating-point operation is rounded to a value that is representable in the destination format. The IEEE 754 standard defines the required rounding modes for compliant floating-point implementations, and ARM implementations support these rounding modes. See the appropriate ARM *Architecture Reference* manual for more information about support for the different rounding modes.

Note The IEEE 754-2008 standard changes the term *Rounding mode* to *Rounding-direction attribute*. ARM documentation continues to use the term Rounding mode, as defined in the IEEE 754-1985 standard.

See also RN, RM, RP, RZ, Rounding error.

RP An ARM abbreviation for *Round towards Plus Infinity* rounding mode, see *Rounding mode*.

RSD *See* Running System Debug (RSD).

RTSM *See* Real Time System Model (RTSM).

Running System Debug (RSD)
Means that a target can be debugged while it is running. RSD gives access to the application using a *Debug Agent* (DA) that resides on the target. The Debug Agent is scheduled with other tasks in the system.

See also Halted System Debug (HSD), Debug Agent.

RVCT *See* RealView Compilation Tools (RVCT).

RVDS *See* RealView Development Suite (RVDS).

RVI *See* RealView ICE.

RVISS *See* RealView Instruction Set Simulator (RVISS).

RWPI *See* Read Write Position Independent (RWPI).

RZ An ARM abbreviation for *Round towards Zero* rounding mode, see *Rounding mode*.

Saved Program Status Register (SPSR)
A register used to save the state of the core on taking an exception. For more information about the use of the SPSR seeProgram Status Register (PSR).

See also Current Program Status Register (CPSR), PSTATE.

SBO *See* Should-Be-One (SBO).

SBOP *See* Should-Be-One-or-Preserved (SBOP).

SBZ *See* Should-Be-Zero (SBZ).

SBZP *See* Should-Be-Zero-or-Preserved (SBZP).

Scatter-loading Assigning the address and grouping of code and data sections individually rather than using single large blocks.

SCC *See* Serial Configuration Controller (SCC).

SDF *See* Standard Delay Format (SDF).

Section In the context of applications targeted at cores that implement the ARM architecture, a block of software or data for an image.

See also Input section and Output section.

In the context of an MMU, when using the short-descriptor translation table format, a 1MB region of virtual address space that a translation table descriptor assigns to a 1MB block of physical address or intermediate physical address space.

Secure monitor In software descriptions, the module that switches the ARM core between Normal World and Secure World execution environments.

Secure World *See* Normal and Secure Worlds.

Security hole A mechanism by which execution at the current level of privilege can achieve an outcome that cannot be achieved at the current or a lower level of privilege using instructions that are not UNPREDICTABLE and are not CONSTRAINED UNPREDICTABLE. The ARM architecture forbids security holes.

Semihosting A mechanism to communicate *Input/Output* (I/O) requests from application code to a host workstation running a debugger. For example, you can use semihosting to permit functions in the C library, such as `printf()` and `scanf()`, to use the screen and keyboard on the host workstation instead of having a screen and keyboard on the target system.

Serial Configuration Controller (SCC)
On CoreTile Express daughterboards, a register interface in the test chip or development chip on the daughterboard, that the Versatile Express configuration system uses to configure the chip. On LogicTile Express daughterboards, a register interface in the FPGA on the daughterboard, that the Versatile Express configuration system uses to configure the FPGA.

The configuration system loads values into the registers at powerup or reset and during runtime.

Serial Power Controller (SPC)
On Versatile Express CoreTile daughterboards, a register interface in an ARM test chip or development chip that the Versatile Express configuration system uses to configure the power controller within the chip.

The SPC registers define the power status, supply levels and clock values of the internal power domains within the chip.

Serial Wire Debug (SWD)
A debug implementation that uses a serial connection between the SoC and a debugger. This connection normally requires a bidirectional data signal and a separate clock signal, rather than the four to six signals required for a JTAG connection.

Serial Wire Debug Port (SW-DP)
The interface for Serial Wire Debug.

Serial Wire or JTAG - Debug Port (SWJ-DP)
The SWJ-DP is a combined JTAG-DP and SW-DP that you can use to connect either a Serial Wire Debug (SWD) or JTAG probe to a target.

Shared layer
In general, the Shared layer contains functions used by more than one Mali GPU driver. It contains math functions, texture processing functions, and list utilities.

Should-Be-One (SBO)
Hardware must ignore writes to the field.

Software should write the field as all 1s. If software writes a value that is not all 1s, it must expect an UNPREDICTABLE result.

This description can apply to a single bit that should be written as 1, or to a field that should be written as all 1s.

Should-Be-One-or-Preserved (SBOP)
The ARMv7 Large Physical Address Extension modified the definition of SBOP to apply to register fields that are SBOP in some but not all contexts. From the introduction of ARMv8 such register fields are described as RES1, see *RES1*. The definition of SBOP given here applies only to fields that are SBOP in all contexts.

Hardware must ignore writes to the field.

If software has read the field since the core implementing the field was last reset and initialized, it should preserve the value of the field by writing the value that it previously read from the field. Otherwise, it should write the field as all 1s.

If software writes a value to the field that is not a value previously read for the field and is not all 1s, it must expect an UNPREDICTABLE result.

This description can apply to a single bit that should be written as its preserved value or as 1, or to a field that should be written as its preserved value or as all 1s.

See also Should-Be-Zero-or-Preserved (SBZP), Should-Be-One (SBO).

Should-Be-Zero (SBZ)
Hardware must ignore writes to the field.

Software should write the field as all 0s. If software writes a value that is not all 0s, it must expect an UNPREDICTABLE result.

This description can apply to a single bit that should be written as 0, or to a field that should be written as all 0s.

Should-Be-Zero-or-Preserved (SBZP)
The ARMv7 Large Physical Address Extension modified the definition of SBZP to apply to register fields that are SBZP in some but not all contexts. From the introduction of ARMv8 such register fields are described as RES0, see *RES0*. The definition of SBZP given here applies only to field that are SBZP in all contexts.

Hardware must ignore writes to the field.

If software has read the field since the core implementing the field was last reset and initialized, it must preserve the value of the field by writing the value that it previously read from the field. Otherwise, it must write the field as all 0s.

If software writes a value to the field that is not a value previously read for the field and is not all 0s, it must expect an UNPREDICTABLE result.

This description can apply to a single bit that should be written as its preserved value or as 0, or to a field that should be written as its preserved value or as all 0s.

See also Should-Be-One-or-Preserved (SBOP), Should-Be-Zero (SBZ).

Signaling NaN
In floating-point operation, the floating-point hardware causes an Invalid Operation exception whenever any floating-point operation receives a signaling NaN as an operand. You can use signaling NaNs in debugging, to track down some uses of uninitialized variables.

Sign-Off Model (SOM) An opaque, compiled simulation model generated from a technology-specific netlist of an ARM processor, derived after gate level synthesis and timing annotation, that you can use in back-annotated gate-level simulations to prove the function and timing behavior of the device. A SOM provides accurate timing simulation of a SoC, and supports simulation using production test vectors from the *Automatic Test Pattern Generation* (ATPG) tool. It only supports back-annotation using SDF files. The SOM includes timing information but provides slower simulation than a DSM.

SIMD *See* Single Instruction, Multiple Data (SIMD).

Simple sequential execution
The behavior of an implementation that fetches, decodes, and completely executes each instruction before proceeding to the next instruction. Such an implementation performs no speculative accesses to memory, including to instruction memory. The implementation does not pipeline any phase of execution. In practice, this is the theoretical execution model that the architecture is based on, and ARM does not expect this model to correspond to a realistic implementation of the architecture.

Simple tracepoint In the context of an ARM debugger, a type of tracepoint that enables you to set trigger points, trace start and end points, or trace ranges for memory and data accesses.

See also Tracepoint.

Simulator In the context of the ARM tools, a simulator executes non-native instructions in software, simulating a core.

See also Real Time System Model (RTSM), Instruction Set System Model (ISSM).

Single Instruction, Multiple Data (SIMD)
In the ARM instruction sets, supported SIMD instructions can comprise:

- Instructions that perform parallel operations on the bytes or halfwords of the ARM core registers.
- Instructions that perform vector operations. That is, they perform parallel operations on vectors held in multiword registers.

Different versions of the ARM architecture support and recommend different instructions for vector operations. See the appropriate version of the *ARM Architecture Reference Manual* for more information.

See also Advanced SIMD.

Software breakpoint In the context of an ARM debugger, a breakpoint that is implemented by replacing an instruction in memory with one that causes the core to take an exception. Because instruction memory must be altered, software breakpoints cannot be used where instructions are stored in read-only memory.

See also Breakpoint unit, Chained breakpoint, Conditional breakpoint, Data breakpoint, Instruction breakpoint.

SOM *See* Sign-Off Model (SOM).

SP *See* Stack Pointer (SP).

SPC *See* Serial Power Controller (SPC).

SPSR *See* Saved Program Status Register (SPSR).

Stack Pointer (SP) On ARM cores, SP refers to the stack pointer for the hardware-managed stack, and:

- In AArch32 state, the SP is register R13 in the general-purpose register file.

- In AArch64 state, there is a dedicated SP for each implemented Exception level.

See also Link Register (LR), Program Counter (PC).

Standard Delay Format (SDF)
A file format that contains timing information to the level of individual bits of buses and is used in SDF back-annotation. An SDF file can be generated in a number of ways, but most commonly from a delay calculator.

STM *See* System Trace Macrocell (STM).

Strongly-Ordered Memory
In descriptions before the introduction of ARMv8, memory regions with the strongest ordering requirement. From the introduction of ARMv8, these regions are described as Device-nGnRnE, indicating that the region is:

non-Gathering Multiple memory accesses must not be merged into a single access.

non-Reordering Multiple memory accesses must not be reordered.

no Early Write Acknowledge
A hint to the memory system that only the endpoint of a write access should return an acknowledge for that write access.

For more information see the *ARMv8 Architecture Reference Manual*.

Subnormal value The IEEE 754-2008 standard term for a floating-point operand with a zero exponent and a nonzero fraction field. ARM documentation describes these operands as *denormal* or *denormalized*, as defined by the IEEE 754-1985 standard.

Note
- ARM floating-point implementations comply with the IEEE 754 requirement that denormalized operands are generated and manipulated with the same precision as normal operands.
- Plus or minus 0 have zero exponent fields, but are not denormals because there is no loss of accuracy.

Supervisor Call (SVC) An instruction that causes the core to take a Supervisor Call exception.

Used by the ARM standard C library to handle semihosting.

See the appropriate ARM *Architecture Reference Manual* for more information.

Support code In a floating-point implementation that requires a floating-point subarchitecture, system software that complements the hardware floating-point implementation. The support code can provide a library of routines that perform operations beyond the scope of the hardware, and can include a set of exception handlers to process exceptional conditions in compliance with the IEEE 754 standard.

SVC *See* Supervisor Call (SVC).

SWD *See* Serial Wire Debug (SWD).

SW-DP *See* Serial Wire Debug Port (SW-DP).

SWI *See* Supervisor Call (SVC).

SWJ-DP *See* Serial Wire or JTAG - Debug Port (SWJ-DP).

Synchronization primitive
An instruction that is used to ensure memory synchronization, for example `LDREX` or `STREX`. See the ARM *Architecture Reference Manual* for more information.

System Control Space On Cortex-M series processors, a memory-mapped region from `0xE000E000` to `0xE000EFFF` that provides system control and configuration registers, including control of the *Nested Vectored Interrupt Controller* (NVIC) and debug functions.

System Trace Macrocell (STM)
A trace source designed primarily for high-bandwidth trace of instrumentation embedded into software. This instrumentation is made up of memory-mapped writes to the STM, which carry information about the behavior of the software.

T32 instruction An instruction set that can be used by a core that is in AArch32 execution state. T32 is a variable-length instruction set that uses both 16-bit and 32-bit instruction encodings. It is the only instruction set supported by ARM M-profile processors. Previously, this instruction set was called the Thumb instruction set.

See also AArch32 state, A32 state.

T32 state When a core is executing in AArch32 state, if it is in T32 Instruction set state then it executes T32 instructions.

See also AArch32 state, T32 instruction, A32 state.

T32EE instruction One or two halfwords that specify an operation for a core in AArch32 state in T32EE Instruction set state to perform. T32EE is the T32 Execution Environment and the T32EE instruction set is based on the T32 instruction set, with some changes and additions to make it a better target for dynamically generated code, that is, code compiled on the device either shortly before or during execution. ARMv8 obsoletes the T32EE instruction set.

See also A32 instruction, T32 instruction.

T32EE state In AArch32 state, in the T32EE Instruction set state the core executes the T32EE instruction set.

See also A32 state, T32 state, T32 instruction, Jazelle state.

TAP *See* Test Access Port (TAP).

TAP Controller Logic on a device that enables access to some or all of that device for test purposes. The circuit functionality is defined in the IEEE1149.1 standard.

See also Joint Test Action Group (JTAG).

Target In the context of an ARM debugger, the part of your development platform to which you connect the debugger, and on which debugging operations can be performed. A target can be:

- A runnable target, such as a core that implements the ARM architecture. When connected to a runnable target, you can perform execution-related debugging operations on that target, such as stepping and tracing.
- A non-runnable CoreSight component. CoreSight components provide a system-wide solution to real-time debug and trace.

See also Debug interface.

Target Vehicle Target vehicles provide RVDS with a standard interface to disparate targets so that the debugger can connect easily to new target types without having to make changes to the debugger core software. A Target Vehicle can be a hardware or software interface.

See also RealView ICE, Real Time System Model (RTSM).

TCD *See* Trace Capture Device (TCD).

TCK The clock for the TAP data lines **TMS**, **TDI**, and **TDO**.

See also Test Data Input (TDI), Test Data Output (TDO).

TCM *See* Tightly Coupled Memory (TCM).

TDI *See* Test Data Input (TDI).

TDO *See* Test Data Output (TDO).

Test Access Port (TAP)

The collection of four mandatory and one optional terminals that form the input/output and control interface to a JTAG boundary-scan architecture. The mandatory terminals are **TDI**, **TDO**, **TMS**, and **TCK**. In the JTAG standard, the **nTRST** signal is optional, but this signal is mandatory in ARM processors because it is used to reset the debug logic.

See also Joint Test Action Group (JTAG), TAP Controller, TCK, Test Data Input (TDI), Test Data Output (TDO), TMS.

Test Data Input (TDI) *Test Data Input* (TDI) is the input to a TAP controller from the data source (upstream). Usually this is seen connecting the RealView ICE run control unit to the first TAP controller.

See also Joint Test Action Group (JTAG), RealView ICE, TAP Controller.

Test Data Output (TDO)

Test Data Output (TDO) is the output from a TAP controller to the downstream data sink. Usually this connects the last TAP controller to the RealView ICE run control unit.

See also Joint Test Action Group (JTAG), RealView ICE, TAP Controller.

Texture Descriptor Data structure used by the Mali GPU to describe one texture map.

Thin Links (TLX) A protocol to reduce the number of signals in an AXI point-to-point connection to simplify routing.

Thumb instruction *See* T32 instruction.

Thumb state *See* T32 state.

Thumb-2 The technology, introduced in ARMv6T2, that extends the T32 instruction set to a variable-length instructions set that includes both 16-bit and 32-bit instructions.

See also T32 instruction, T32 state.

Tightly Coupled Memory (TCM)

An area of low-latency memory that provides predictable instruction execution or data load timing, for cases where deterministic performance is required. TCMs are suited to holding:

- Critical routines such as for interrupt handling.
- Scratchpad data.
- Data types whose locality is not suited to caching.
- Critical data structures, such as interrupt stacks.

Tile buffer A memory buffer inside the Mail GPU that holds the framebuffer contents for the tile that is currently being rendered. The tile buffer can be accessed without using the memory bus.

TLB *See* Translation Lookaside Buffer (TLB).

TLB lockdown Prevents specific translation table walk results being removed from the TLB. This ensures that accesses to the associated memory areas do not cause a translation table walk.

TLX *See* Thin Links (TLX).

TMC *See* Trace Memory Controller (TMC).

TMS Test Mode Select.

TPA *See* Trace Port Analyzer (TPA).

TPIU *See* Trace Port Interface Unit (TPIU).

Trace Capture Device (TCD)
A generic term to describe Trace Port Analyzers, logic analyzers, and on-chip trace buffers.

Trace driver A remote debug interface target that controls a piece of trace hardware. That is, the trigger macrocell, trace macrocell, and trace capture tool.

Trace funnel In an ARM trace macrocell, a device that combines multiple trace sources onto a single bus.

See also AHB Trace Macrocell (HTM), CoreSight.

Trace hardware A term for a device that contains an ARM trace macrocell.

Trace Memory Controller (TMC)
Controls the capturing or buffering trace generated by trace sources within a system. The TMC receives trace from an ATB interface and can be configured to perform one of the following:

- Route the trace out over an AXI master interface, to allow trace to be captured in system memory or in other peripherals.
- Capture the trace in a circular buffer in dedicated SRAM.
- Buffer the trace in a *First In First Out* (FIFO) style, to smooth over peaks in trace bandwidth.

Trace port A port on a device, such as a processor or ASIC, used to output trace information.

Trace Port Analyzer (TPA)
A hardware device that captures trace information output on a trace port. This can be a low-cost product designed specifically for trace acquisition, or a logic analyzer.

Trace Port Interface Unit (TPIU)
Drains trace data and acts as a bridge between the on-chip trace data and the data stream captured by a TPA.

Tracepoint A tracepoint can be set on a line of source code, a line of assembly code, or a memory address. In an ARM debugger, you can set a variety of tracepoints to determine exactly what program information is traced.

See also Chained tracepoint, Tracepoint unit.

Tracepoint unit In the context of an ARM debugger, a unit within a Chained tracepoint that combines with other tracepoint units to create a complex tracepoint.

See also Chained tracepoint, Tracepoint.

Translation Lookaside Buffer (TLB)
A memory structure containing the results of translation table walks. TLBs help to reduce the average cost of memory accesses.

Translation table A table, held in memory, that contains descriptors that define the properties of regions of memory, and the mapping between a supplied input address and the corresponding output address.

Note In an ARM system without support for virtualization, the input address is a physical address and the output address is a virtual address.

See also Physical Address (PA), Virtual Address (VA).

Translation table walk A full translation table lookup. It is performed automatically by hardware.

Trap enable bits
For floating-point operation, the trap enable bits determine whether trapped or untrapped exception handling is selected. If trapped exception handling is selected, the way it is carried out is IMPLEMENTATION DEFINED.

Triangle setup unit
A component of a fragment processor. The triangle setup unit prepares primitives for rendering by calculating the data required to rasterize and shade the primitive.

Trigger
In the context of tracing, a trigger is an event that instructs the debugger to stop collecting trace and display the trace information around the trigger position, without halting the core. The exact information that is displayed depends on the position of the trigger within the buffer.

Trigger instruction
In a floating-point implementation that requires a floating-point subarchitecture, a floating-point instruction that causes a bounce when it is issued.

See also Bounce.

Trigger Interface
See Cross Trigger Interface (CTI).

TrustZone Software
A secure software framework that uses the ARM architecture Security Extensions.

UAL
See Unified Assembler Language (UAL).

UMP
See Unified Memory Provider (UMP).

Unaligned
An unaligned access is an access where the address of the access is not aligned to the size of the elements of the access.

Unconditional breakpoint
In the context of an ARM debugger, a breakpoint that does not have a conditional qualifier assigned. The breakpoint activates immediately it is hit, but subsequent image execution is determined by any actions assigned to the breakpoint.

See also Conditional breakpoint, Hardware breakpoint.

UNDEFINED
Indicates an instruction that is not architecturally defined. It generates an Undefined Instruction exception. See the *ARM Architecture Reference Manual* for more information.

Unified Assembler Language (UAL)
A common assembler language for the A32 and T32 instruction sets. See the appropriate A*RM Architecture Reference Manual* for more information.

See also A32 instruction, T32 instruction.

Unified Memory Provider (UMP)
Provides a safe way to share memory across processes, drivers and hardware components, possibly using an MMU or MPU for memory protection. The Mali GPU driver stack uses the UMP API for certain optional functionality.

UNK
An abbreviation indicating that software must treat a field as containing an UNKNOWN value.

In any implementation, the bit must read as 0, or all 0s for a bit field. Software must not rely on the field reading as zero.

See also UNKNOWN.

UNK/SBOP
Hardware must implement the field as Read-As-One, and must ignore writes to the field.

Software must not rely on the field reading as all 1s, and except for writing back to the register it must treat the value as if it is UNKNOWN. Software must use an SBOP policy to write to the field.

This description can apply to a single bit that should be written as its preserved value or as 1, or to a field that should be written as its preserved value or as all 1s.

See also Read-As-One (RAO), RES1, Should-Be-One-or-Preserved (SBOP), UNKNOWN.

UNK/SBZP Hardware must implement the field as Read-As-Zero, and must ignore writes to the field.

Software must not rely on the field reading as all 0s, and except for writing back to the register it must treat the value as if it is UNKNOWN. Software must use an SBZP policy to write to the field.

This description can apply to a single bit that should be written as its preserved value or as 0, or to a field that should be written as its preserved value or as all 0s.

See also Read-As-Zero (RAZ), RES0, Should-Be-Zero-or-Preserved (SBZP), UNKNOWN.

UNKNOWN An UNKNOWN value does not contain valid data, and can vary from moment to moment, instruction to instruction, and implementation to implementation. An UNKNOWN value must not return information that cannot be accessed at the current or a lower level of privilege using instructions that are not UNPREDICTABLE or CONSTRAINED UNPREDICTABLE and do not return UNKNOWN values.

An UNKNOWN value must not be documented or promoted as having a defined value or effect.

When UNKNOWN appears in body text, it is always in SMALL CAPITALS.

UNP *See* UNPREDICTABLE.

UNPREDICTABLE For an ARM processor, UNPREDICTABLE means the behavior cannot be relied upon. UNPREDICTABLE behavior must not perform any function that cannot be performed at the current or a lower level of privilege using instructions that are not UNPREDICTABLE.

UNPREDICTABLE behavior must not be documented or promoted as having a defined effect.

An instruction that is UNPREDICTABLE can be implemented as UNDEFINED.

In an implementation that supports Virtualization, the Non-secure execution of UNPREDICTABLE instructions at a lower level of privilege can be trapped to the hypervisor, provided that at least one instruction that is not UNPREDICTABLE can be trapped to the hypervisor if executed at that lower level of privilege.

For an ARM trace macrocell, UNPREDICTABLE means that the behavior of the macrocell cannot be relied on. Such conditions have not been validated. When applied to the programming of an event resource, only the output of that event resource is UNPREDICTABLE. UNPREDICTABLE behavior can affect the behavior of the entire system, because the trace macrocell can cause the core to enter debug state, and external outputs can be used for other purposes.

When UNPREDICTABLE appears in body text, it is always in SMALL CAPITALS.

See also CONSTRAINED UNPREDICTABLE.

VA *See* Virtual Address (VA).

VDMA Video Direct Memory Access. The VDMA transfers data in a burst efficient way to and from system memory.

Vectorization Using multiword registers to hold multiple values of the same type for SIMD processing. For example, software might use doubleword registers to hold four 16-bit unsigned integers. Vectorization also describes the process of adapting software to use SIMD processing.

Vector operations are provided by:

- The VFP instructions in ARMv6.

- The Advanced SIMD instructions from ARMv7.

Veneer A small block of code used with subroutine calls when there is a requirement to change the state of the core or to branch to an address that cannot be reached in the current state of the core.

Vertex The data set that defines the properties of one point of a primitive. For example, a point primitive, an endpoint of a line primitive, or a corner of a triangle primitive.

Vertex attributes The data provided by the application, to define a vertex.

Vertex loader A component of the vertex processor that loads vertex attributes from memory into the vertex shader unit.

Vertex processor A programmable processor that executes vertex shaders with typical transform and lightning calculations, and generates lists of primitives for a fragment processor to draw.

The vertex processor was originally known as a geometry processor.

Vertex shader A program running on a vertex processor or shader core, that calculates the position and other characteristics, such as color and texture coordinates, for each vertex.

Vertex shader unit A programmable component of the vertex processor that runs vertex shaders

VFP The original name of the extension to the ARM architecture that provided floating-point arithmetic. In ARMv7, the extension is called the Floating-point extension. From ARMv8, the architecture includes support for floating-point instruction, rather than this being an architecture extension.

Victim A cache line that is selected to be discarded to make room for a replacement cache line that is required because of a cache miss. How the victim is selected for eviction is processor-specific. A victim is also known as a cast out.

Virtual Address (VA) An address used in an instruction as a data or instruction address. The PC, LR, and SP always hold virtual addresses. For a *Protected Memory System Architecture* (PMSA) implementation, the virtual address is identical to the physical address.

Warm reset Also known as a core reset. Initializes most of the processor functionality, excluding the debug controller and debug logic. This type of reset is useful if you are using the debugging features of a processor.

See also Cold reset.

Watch In an ARM debugger, a watch is a variable or expression that you require the debugger to display at every step or breakpoint so that you can see how its value changes.

Watchpoint A debug event triggered by an access to memory, specified in terms of the address of the location in memory being accessed.

In DS-5, this is a hardware breakpoint.

See also Breakpoint.

Word A 32-bit data item. Words are normally word-aligned in ARM systems.

Word-aligned A data item having a memory address that is divisible by four.

Word-invariant In a word-invariant system, the address of each byte of memory changes when switching between little-endian and big-endian operation, in such a way that the byte with address A in one endianness has address A EOR 3 in the other endianness. As a result, each aligned word of memory always consists of the same four bytes of memory in the same order, regardless of endianness. The change of endianness occurs because of the change to the byte addresses, not because the bytes are rearranged.

The ARM architecture supports word-invariant systems in ARMv3 and later versions. When word-invariant support is selected, the behavior of load or store instructions with unaligned addresses is instruction-specific, and is in general not the expected behavior for an unaligned access. ARM strongly recommends that word-invariant systems use the endianness that produces the required byte addresses at all times, possibly apart from very early in their reset handlers before they have set up the endianness, and that this early part of the reset handler uses only word-aligned memory accesses.

See also Byte-invariant.

Write Operations that have the semantics of a store. See the *ARM Architecture Reference Manual* for more information.

Write completion The memory system indicates to the core that a write is complete at a point in the transaction where the memory system can guarantee that the write is observable by all processors in the system.

An additional recommendation for Device-nGnRnE memory (Strongly-ordered memory), is that a write to a memory-mapped peripheral is complete only when it reaches that memory-mapped peripheral and therefore can trigger any side effects caused by the memory-mapped peripheral. Write completion is not required to ensure that all side effects are globally visible, although some peripherals might define this as a required property of completed writes.

See also Strongly-Ordered Memory.

Write interleave capability
For an interface to an interconnect, the number of data-active write transactions for which the interface can transmit data. This is counted from the earliest transaction.

Write interleave depth The number of data-active write transactions for which an interface can receive data.

XTSM *See* XVC Test Scenario Manager (XTSM).

XVC *See* Extensible Verification Component (XVC).

XVC Test Scenario Manager (XTSM)
This coordinates the operation of multiple XVCs.

See also Extensible Verification Component (XVC).

Appendix A
Revisions

This appendix describes the technical changes from Issue E and Issue F of this book.

Table A-1 Differences between Issue F and Issue G

Term	Change
Architecturally executed	Added
ARM Debug Interface (ADI)	Added
Conditional execution	Updated
DAPBUS interconnect	Added
Event asynchronous bridge	Added
Simple sequential execution	Added
System Trace Macrocell (STM)	Added
Trace Memory Controller (TMC)	Added
UNDEFINED	Updated

Table A-2 Differences between Issue E and Issue F

Term	Change
A32 instruction, was *ARM instruction*	Updated
A32 state	Added
A64 instruction, was *A64*	Updated
AArch32 state, was *AArch32*	Updated
AArch64 state, was *AArch64 state*	Updated
Abort model	Updated
ACE interface	Added
ACE protocol	Added
ACE-Lite interface	Added
Advanced SIMD	Updated
APB	Updated
ARM instruction	Updated
ARM state	Updated
ARM TrustZone® technology	Updated
ATB	Updated
ATB bridge	Updated
Automatic Test Pattern Generation (ATPG)[a]	Removed
Authentication Asynchronous Bridge	Added
Authentication Synchronous Bridge	Added
Banked registers	Updated
Base register	Updated
Bounce	Updated
Cluster	Added
Coherence Order, *Coherent*	Added
Cold reset	Updated
CONSTRAINED UNPREDICTABLE	Added
Condition code check	Added
Condition flags	Updated
Conditional execution	Updated
Context synchronization operation	Updated
Context switch	Updated
Coprocessor	Updated
Core	Added

Table A-2 Differences between Issue E and Issue F (continued)

Term	Change
Core register	Updated
Cross-path blocking	Added
Current Program Status Register (CPSR)	Updated
Data Abort	Updated
Data Timing Module (DTM)[a]	Removed
Daughterboard Configuration Controller	Added
Debug Communications Channel (DCC)	Updated
Direct read	Added
Direct write	Added
Endianness	Updated
Exception Link Register (ELR)	Added
Exceptional state	Updated
Explicit access	Updated
Fast Context Switch Extension (FCSE)	Updated
Fixed-function pipeline	Removed
Fragment Thread Creator	Added
G-POP	Removed
Generic Thread Creator	Added
Granular Power Requester	Added
Graphics Processor Unit (GPU)	Removed
Half-rate clocking	Removed
Halting debug, was *Halting debug-mode*	Updated
Head-of-line blocking	Added
Hierarchical Tiler (HT)	Added
High registers	Updated
High vectors	Updated
Hit-Under-Miss (HUM)	Updated
HT	Added
IEEE 1149.1[a]	Removed
IGNORED	Added
IMPLEMENTATION SPECIFIC	Updated
Imprecise tracing	Updated
In-Circuit Emulator	Updated

Table A-2 Differences between Issue E and Issue F (continued)

Term	Change
Indirect read	Added
Indirect write	Added
Instruction Abort	Added
Instruction Synchronization Barrier (ISB)	Updated
Instrumentation trace	Updated
Interworking	Updated
ISSM	Added
IT block	Updated
Jazelle Extension, was *Jazelle architecture*	Updated
Jazelle DBX	Updated
Jazelle RCT (Runtime Compilation Target)	Updated
Jazelle state	Updated
Job system back-end	Updated
Link Register (LR)	Added
Low-Power Interface (LPI)	Added
Memory coherency	Updated
Memory Management Unit (MMU)	Updated
Memory Protection Unit (MPU)	Updated
Monitor debug, was *Monitor debug-mode*	Updated
Motherboard Configuration Controller (MCC)	Added
NEON technology, was *NEON*	Updated
Normal and Secure Worlds	Updated
OpenGL ES Shading Language (ESLL)[a]	Removed
PLI	Added
Powerup reset	Added
Prefetch Abort	Updated
Processing Element (PE)	Added
Processor	Added
PSTATE	Added
Quality of Service using Virtual Networks (QVN)	Added
QVN	Added
RAZ/WI	Updated
Read-Only Position Independent (ROPI)	Updated

Table A-2 Differences between Issue E and Issue F (continued)

Term	Change
Read Write Position Independent (RWPI)	Updated
Register slice	Added
RES0	Added
RES1	Added
Reserved	Added
RM, was *Round towards Minus infinity (RM) mode*	Updated
RN, was *Round to Nearest (RN) mode*	Updated
Rounding mode	Updated
RP, was *Round towards Plus infinity (RP) mode*	Updated
RZ, was *Round towards Zero (RZ) mode*	Updated
Saved Program Status Register (SPSR)	Updated
Section	Updated
Secure monitor	Updated
Security hole	Updated
Serial Configuration Controller (SCC)	Added
Serial Power Controller (SPC)	Added
Serial Wire or JTAG - Debug Port (SWJ-DP)	Added
Should-Be-One-or-Preserved (SBOP)	Updated
Should-Be-Zero-or-Preserved (SBZP)	Updated
Single Instruction, Multiple Data (SIMD)	Updated
SPC	Added
SPICE	Removed
Stack Pointer (SP)	Updated
Strongly-Ordered Memory	Added
Subnormal value	Updated
T32 instruction, was *Thumb instruction*	Updated
T32 state, was *Thumb state*	Updated
T32EE instruction, was *ThumbEE instruction*	Updated
T32EE state, was *ThumbEE state*	Updated
Test Access Port (TAP)	Updated
Thin Links (TLX)	Added
Tile buffer	Updated
Translation table	Updated

Table A-2 Differences between Issue E and Issue F (continued)

Term	Change
Trap enable bits	Updated
Trigger instruction	Updated
Trigger Interface	Added
Unified Assembler Language (UAL)	Updated
UNKNOWN	Updated
UNPREDICTABLE	Updated
Vectorization	Updated
VFP	Updated
Write completion	Updated
Write interleave capability	Updated
XVC Test Scenario Manager (XTSM)	Updated

a. In general, the glossary does not define industry-standard terms unless the use of the term in ARM documents differs in some way from the usual understanding of the term.