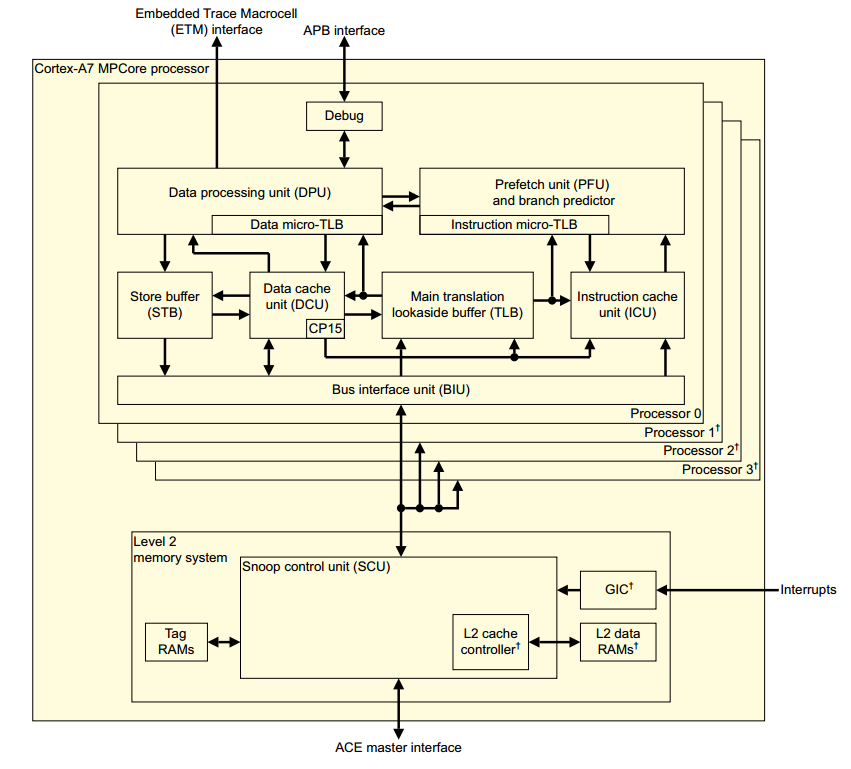
## About the Cortex-A7 MPCore processor

The Cortex-A7 MPCore processor is a high-performance, low-power processor that implements the ARMv7-A architecture. The Cortex-A7 MPCore processor has four processors in a single multiprocessor device with a L1 cache subsystem, an integrated GIC, and an L2 cache controller. Picture 1-1 shows a top-level functional diagram of the Cortex-A7 MPCore processor.



Picture 1-1

It contains Data Processing Unit, System control coprocessor, Instruction side memory system, Data side memory system, L1 memory system, Media Processing Engine, Floating-Point Unit, L2 memory system, Debug and Performance monitoring.

The cortexa7 feature list:

* 4 cores in the cortexa7 MPCore.
* 192 SPI interrupts in which there are 160 external device interrupt.
* 32KB L1 I Cache.
* 32KB L1 D Cache.
* 512KB L2 Cache.
* FPU which is a VFPv4-D16 implementation of the ARMv7 floating-point architecture.
* NEON provide support for the ARMv7 Advanced SIMDv2 and Vectore Floaint-Pointv4 instruction sets.

### Data Processing Unit

The *Data Processing Unit* (DPU) holds most of the program-visible state of the processor, such as general-purpose registers, status registers and control registers. It decodes and executes instructions, operating on data held in the registers in accordance with the ARM Architecture. Instructions are fed to the DPU from the *Prefetch Unit* (PFU). The DPU executes instructions that require data to be transferred to or from the memory system by interfacing to the *Data Cache Unit* (DCU), which manages all load and store operations.

### System control coprocessor

The system control coprocessor, CP15, provides configuration and control of the memory  
system and its associated functionality.

### Instruction side memory system

The instruction side memory system includes instruction cache unit and prefetch unit.

#### Instruction Cache Unit

The *Instruction Cache Unit* (ICU) contains the Instruction Cache controller and its associated linefill buffer. The Cortex-A7 MPCore ICache is two-way set associative and uses *Virtually Indexed Physically Tagged* (VIPT) cache-lines holding up to 8 ARM or Thumb 32-bit instructions or up to 16 Thumb 16-bit instructions.

#### 1.3.2 Prefetch Unit

The *Prefetch Unit* (PFU) obtains instructions from the instruction cache or from external memory and predicts the outcome of branches in the instruction stream, then passes the instructions to the DPU for processing. In any given cycle, up to a maximum of four instructions can be fetched and two can be passed to the DPU.

#### 1.3.3 Branch Target Instruction Cache

The PFU also contains a four-entry deep *Branch Target Instruction Cache* (BTIC). Each entry stores up to two instruction cache fetches and enables the branch shadow of predicted taken B and BL instructions to be eliminated. The BTIC implementation is architecturally transparent, so it does not have to be flushed on a context switch.

#### 1.3.4 Branch Target Instruction Cache

The PFU also contains a eight-entry deep *Branch Target Address Cache* (BTAC) used to predict the target address of certain indirect branches. The BTAC implementation is architecturally transparent, so it does not have to be flushed on a context switch.

#### 1.3.5 Branch prediction

The branch predictor is a global type that uses history registers and a 256-entry pattern history table.

#### 1.3.6 Return stack

The PFU includes an 8-entry return stack to accelerate returns from procedure calls. For each procedure call, the return address is pushed onto a hardware stack. When a procedure return is recognized, the address held in the return stack is popped, and the PFU uses it as the predicted return address. The return stack is architecturally transparent, so it does not have to be flushed on a context switch

### Data side memory system

This section describes Data Cache Unit, Store Buffer and Bus Interface and SCU interface.

#### Data Cache Unit

The *Data Cache Unit* (DCU) consists of the following sub-blocks:

* The *Level 1* (L1) data cache controller, which generates the control signals for the associated embedded tag, data, and dirty memory (RAMs) and arbitrates between the different sources requesting access to the memory resources. The data cache is 4-way set associative and uses a *Physically Indexed Physically Tagged* (PIPT) scheme for lookup which enables unambiguous address management in the system.
* The load/store pipeline that interfaces with the DPU and main TLB.
* The system coprocessor controller that performs cache maintenance operations directly on the data cache and indirectly on the instruction cache through an interface with the ICU.
* An interface to receive coherency requests from the *Snoop Control Unit* (SCU).

The DCU contains a combined local and global exclusive monitor. This monitor can be set to the exclusive state only by a LDREX instruction executing on the local processor, and can be cleared to the open access state by:

* A STREX instruction on the local processor or a store to the same shared cache line on another processor.
* The cache line being evicted for other reasons.
* A CLREX instruction.

The Cortex-A7 MPCore processor uses the MOESI protocol, with ACE modified equivalents of MOESI states, to maintain data coherency between multiple processors. MOESI describes the state that a shareable line in a L1 data cache can be in:

**M** Modified/UniqueDirty (UD). The line is only in this cache and is dirty.

**O** Owned/SharedDirty (SD). The line is possibly in more than one cache and is dirty.

**E** Exclusive/UniqueClean (UC). The line is only in this cache and is clean.

**S** Shared/SharedClean (SC). The line is possibly in more than one cache and is clean.

**I** Invalid/Invalid (I). The line is not in this cache.

The DCU stores the MOESI state of the cache line in the tag and dirty RAMs.

#### Store Buffer

The *Store Buffer* (STB) holds store operations when they have left the load/store pipeline and have been committed by the DPU. From the STB, a store can request access to the cache RAMs in the DCU, request the BIU to initiate linefills, or request the BIU to write the data out on the external write channel. External data writes are through the SCU.

The STB can merge:

* Several store transactions into a single transaction if they are to the same 64-bit aligned address. The STB is also used to queue up CP15 maintenance operations before they are broadcast to other processors in the multiprocessor device.
* Multiple writes into an AXI write burst.

#### Bus Interface Unit and SCU interface

The *Bus Interface Unit* (BIU) contains the SCU interface and buffers to decouple the interface from the cache and STB. The BIU interface and the SCU always operate at the processor frequency.

A write buffer is available to hold:

* Data from cache evictions or non-cacheable write bursts before they are written out to the SCU.
* The addresses of outstanding ACE write transactions to permit hazard checking against other outstanding requests in the system.

#### L1 memory system

The processor L1 memory system includes the following features:

* Separate instruction and data caches.
* Export of memory attributes for system caches.

The caches have the following features:

* Support for instruction and data cache sizes between 8KB and 64KB.
* Pseudo-random cache replacement policy.
* Ability to disable each cache independently.
* Streaming of sequential data from LDM and LDRD operations, and sequential instruction fetches.
* Critical word first linefill on a cache miss.
* All the cache RAM blocks and associated tag and valid RAM blocks if implemented using standard RAM compilers.

#### L2 memory system

The L2 memory system contains:

* The SCU that connects between one to four processors to the external memory system through the ACE master interface. The SCU maintains data cache coherency between the processors and arbitrates L2 requests from the processors.
* L2 cache that:
* Has cache RAM size of 512KB.
* Is 8-way set associative.
* Supports 64-byte cache lines.
* One ACE master interface. All transactions are routed through the interface.

### Generic Interrupt Controller

The integrated GIC manages interrupt sources and behavior, and can route interrupts to individual processors. It permits software to mask, enable and disable interrupts from individual sources, to prioritize, in hardware, individual sources and to generate software interrupts. It also provides support for the Security and Virtualization Extensions. The GIC accepts interrupts asserted at the system level and can signal them to each processor it is connected to. This can result in an IRQ or FIQ exception being taken.

### Media Processing Engine

The optional *Media Processing Engine* (MPE) implements ARM NEON technology, a media and signal processing architecture that adds instructions targeted at audio, video, 3-D graphics, image, and speech processing. Advanced SIMD instructions are available in both ARM and Thumb states.

The Cortex-A7 NEON MPE includes the following features:

* SIMD and scalar single-precision floating-point computation.
* Scalar double-precision floating-point computation.
* SIMD and scalar half-precision floating-point conversion.
* SIMD 8, 16, 32, and 64-bit signed and unsigned integer computation.
* 8 or 16-bit polynomial computation for single-bit coefficients.
* Structured data load capabilities.
* Large, shared register file, addressable as:
* Thirty-two 32-bit S (single) registers.
* Thirty-two 64-bit D (double) registers.
* Sixteen 128-bit Q (quad) registers.

The operations include:

* Addition and subtraction.
* Multiplication with optional accumulation.
* Maximum or minimum value driven lane selection operations.
* Inverse square-root approximation.
* Comprehensive data-structure load instructions, including register-bank-resident table lookup.

See the *Cortex-A7 MPCore NEON Media Processing Engine Technical Reference Manual* for more information.

### Floating-Point Unit

The optional *Floating-Point Unit* (FPU) implements the ARMv7 VFPv4-D16 architecture and includes the VFP register file and status registers. It performs floating-point operations on the data held in the VFP register file.

The Cortex-A7 FPU features are:

* Support for single-precision and double-precision floating-point formats.
* Support for conversion between half-precision and single-precision.
* Support for *Fused Multiply ACcumulate* (FMAC) operations.
* Normalized and denormalized data are all handled in hardware.
* Trapless operation enabling fast execution.

See the *Cortex-A7 MPCore Floating-Point Unit Technical Reference Manual* for more information.

### Debug

The Cortex-A7 MPCore processor has a CoreSight compliant *Advanced Peripheral Bus version 3* (APBv3) debug interface. This permits system access to debug resources, for example, the setting of watch points and breakpoints. The processor provides extensive support for real-time debug and performance profiling.

### Performance monitoring

The Cortex-A7 MPCore processor provides performance counters and event monitors that can be configured to gather statistics on the operation of the processor and the memory system.