

ARM Arch

Contents

ARM architecture	1
List of ARM microarchitectures	19
ARM7	22
ARM9	26
ARM11	30
ARM Cortex-A	37
XScale	39
Comparison of ARMv7-A cores	44

References


Article Sources and Contributors	46
Image Sources, Licenses and Contributors	47

Article Licenses

License	48
---------	----

ARM architecture

ARM architectures

 The ARM logo	
Designer	ARM Holdings
Bits	32-bit, 64-bit
Introduced	1985
Design	RISC
Type	Register-Register
Branching	Condition code
Open	Proprietary

64/32-bit architecture

Introduced	2011
Version	ARMv8-A
Encoding	AArch64/A64 and AArch32/A32 use 32-bit instructions, T32 (Thumb-2) uses mixed 16- and 32-bit instructions. ARMv7 user-space compatibility
Endianness	Bi (little as default)
Extensions	All mandatory: Thumb-2, NEON, Jazelle, VFPv4-D16, VFPv4
Registers	
General purpose	31x 64-bit integer registers plus PC and SP, ELR, SPSR for exception levels
Floating point	32x 128-bit registers, scalar 32- and 64-bit FP, SIMD 64- and 128-bit FP and integer

32-bit architectures (Cortex)

Version	ARMv8-R, ARMv7-A, ARMv7-R, ARMv7E-M, ARMv7-M, ARMv6-M
Encoding	32-bit except Thumb-2 extensions use mixed 16- and 32-bit instructions.
Endianness	Bi (little as default)
Extensions	Thumb-2 (mandatory since ARMv7), NEON, Jazelle, FPv4-SP
Registers	
General purpose	16x 32-bit integer registers including PC and SP
Floating point	Up to 32x 64-bit registers, SIMD/floating-point (optional)

32-bit architectures (legacy)

Version	ARMv6, ARMv5, ARMv4T, ARMv3, ARMv2
Encoding	32-bit except Thumb extension uses mixed 16- and 32-bit instructions.
Endianness	Bi (little as default) in ARMv3 and above
Extensions	Thumb, Jazelle
Registers	
General purpose	16x 32-bit integer registers including PC (26-bit addressing in older) and SP

ARM is a family of instruction set architectures for computer processors based on a reduced instruction set computing (RISC) architecture developed by British company ARM Holdings.

A RISC-based computer design approach means ARM processors require significantly fewer transistors than typical CISC x86 processors in most personal computers. This approach reduces costs, heat and power use. These are desirable traits for light, portable, battery-powered devices— including smartphones, laptops, tablet and notepad computers, and other embedded systems. A simpler design facilitates more efficient multi-core CPUs and higher core counts at lower cost, providing improved energy efficiency for servers.^{[1][2][3]}

ARM Holdings develops the instruction set and architecture for ARM-based products, but does not manufacture products. The company periodically releases updates to its cores. Current cores from ARM Holdings support a 32-bit address space and 32-bit arithmetic; the ARMv8-A architecture, announced in October 2011, adds support for a 64-bit address space and 64-bit arithmetic. Instructions for ARM Holdings' cores have 32 bits wide fixed-length instructions, but later versions of the architecture also support a variable-length instruction set that provides both 32 and 16 bits wide instructions for improved code density. Some cores can also provide hardware execution of Java bytecodes.

ARM Holdings licenses the chip designs and the ARM instruction set architectures to third parties, who design their own products that implement one of those architectures— including systems-on-chips (SoC) that incorporate memory, interfaces, radios, etc. Currently, the widely used Cortex cores, older "classic" cores, and specialized SecurCore cores variants are available for each of these to include or exclude optional capabilities. Companies that make chips that implement an ARM architecture include Apple, Nvidia, Qualcomm, Samsung Electronics, and Texas Instruments.

Globally ARM is the most widely used instruction set architecture in terms of quantity produced. The low power consumption of ARM processors has made them very popular: over 50 billion ARM processors have been produced as of 2014[4], thereof 10 billion in 2013 and "ARM-based chips are found in nearly 60 percent of the world's mobile devices". In 2008, 10 billion chips had been produced. The ARM architecture (32-bit) is the most widely used architecture in mobile devices, and most popular 32-bit one in embedded systems. In 2005, about 98% of all mobile phones sold used at least one ARM processor. According to ARM Holdings, in 2010 alone, producers of chips based on ARM architectures reported shipments of 6.1 billion ARM-based processors, representing 95% of smartphones, 35% of digital televisions and set-top boxes and 10% of mobile computers.

VLSI produced the first ARM silicon on 26 April 1985. It worked the first time, and was known as ARM1 by April 1985. The first production systems named ARM2 were available the following year.

The first ARM application was as a second processor for the BBC Micro, where it helped in developing simulation software to finish development of the support chips (VIDC, IOC, MEMC), and sped up the CAD software used in ARM2 development. Wilson subsequently rewrote BBC BASIC in ARM assembly language. The in-depth knowledge gained from designing the instruction set enabled the code to be very dense, making ARM BBC BASIC an extremely good test for any ARM emulator. The original aim of a principally ARM-based computer was achieved in 1987 with the release of the Acorn Archimedes. In 1992, Acorn once more won the Queen's Award for Technology for the ARM.

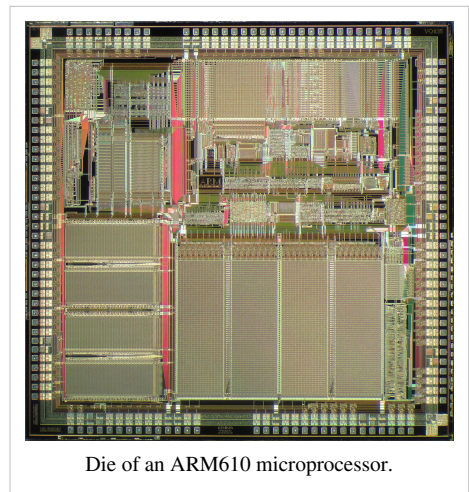
The ARM2 featured a 32-bit data bus, 26-bit address space and 27 32-bit registers. Eight bits from the program counter register were available for other purposes; the top six bits (available because of the 26-bit address space), served as status flags, and the bottom two bits (available because the program counter was always word-aligned), were used for setting modes. The address bus was extended to 32 bits in the ARM6, but program code still had to lie within the first 64 MB of memory in 26-bit compatibility mode, due to the reserved bits for the status flags. The ARM2 had a transistor count of just 30,000, compared to Motorola's six-year-old 68000 model with around 40,000. Much of this simplicity came from the lack of microcode (which represents about one-quarter to one-third of the 68000) and from (like most CPUs of the day) not including any cache. This simplicity enabled low power consumption, yet better performance than the Intel 80286. A successor, ARM3, was produced with a 4 KB cache, which further improved performance.

Apple, DEC, Intel, Marvell: ARM6, StrongARM, XScale

In the late 1980s Apple Computer and VLSI Technology started working with Acorn on newer versions of the ARM core. In 1990, Acorn spun off the design team into a new company named Acorn RISC Machines Ltd., which became ARM Ltd when its parent company, ARM Holdings plc, floated on the London Stock Exchange and NASDAQ in 1998.^[4]

The new Apple-ARM work would eventually evolve into the ARM6, first released in early 1992. Apple used the ARM6-based ARM610 as the basis for their Apple Newton PDA. In 1994, Acorn used the ARM610 as the main central processing unit (CPU) in their RiscPC computers. DEC licensed the ARM6 architecture and produced the StrongARM. At 233 MHz, this CPU drew only one watt (newer versions draw far less). This work was later passed to Intel as a part of

a lawsuit settlement, and Intel took the opportunity to supplement their i960 line with the StrongARM. Intel later developed its own high performance implementation named XScale, which it has since sold to Marvell. Transistor count of the ARM core remained essentially the same size throughout these changes; ARM2 had 30,000 transistors, while ARM6 grew only to 35,000. Wikipedia:Citation needed



Die of an ARM610 microprocessor.

Licensing

See also: ARM Holdings § Licensees

Core license

ARM Holdings' primary business is selling IP cores, which licensees use to create microcontrollers (MCUs) and CPUs based on those cores. The original design manufacturer combines the ARM core with other parts to produce a complete CPU, typically one that can be built in existing semiconductor fabs at low cost and still deliver substantial performance. The most successful implementation has been the ARM7TDMI with hundreds of millions sold. Atmel has been a precursor design center in the ARM7TDMI-based embedded system.

The ARM architectures used in smartphones, PDAs and other mobile devices range from ARMv5, used in low-end devices, through ARMv6, to ARMv7 in current high-end devices. ARMv7 includes a hardware floating-point unit (FPU), with improved speed compared to software-based floating-point.

In 2009, some manufacturers introduced netbooks based on ARM architecture CPUs, in direct competition with netbooks based on Intel Atom. According to analyst firm IHS iSuppli, by 2015, ARM ICs may be in 23% of all laptops.

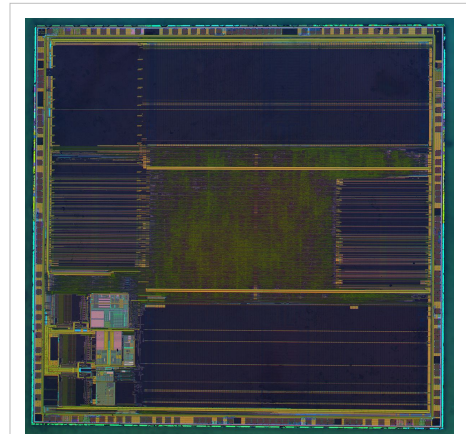
ARM Holdings offers a variety of licensing terms, varying in cost and deliverables. ARM Holdings provides to all licensees an integratable hardware description of the ARM core as well as complete software development toolset (compiler, debugger, software development kit) and the right to sell manufactured silicon containing the ARM CPU.

SoC packages integrating ARM's core designs include Nvidia Tegra's first three generations, CSR plc's Quatro family, ST-Ericsson's Nova and NovaThor, Silicon Labs's Precision32 MCU, Texas Instruments's OMAP products, Samsung's Hummingbird and Exynos products, Apple's A4, A5, and A5X, and Freescale's i.MX.

Fabless licensees, who wish to integrate an ARM core into their own chip design, are usually only interested in acquiring a ready-to-manufacture verified IP core. For these customers, ARM Holdings delivers a gate netlist description of the chosen ARM core, along with an abstracted simulation model and test programs to aid design integration and verification. More ambitious customers, including integrated device manufacturers (IDM) and foundry operators, choose to acquire the processor IP in synthesizable RTL (Verilog) form. With the synthesizable RTL, the customer has the ability to perform architectural level optimisations and extensions. This allows the designer to achieve exotic design goals not otherwise possible with an unmodified netlist (high clock speed, very low power consumption, instruction set extensions, etc.). While ARM Holdings does not grant the licensee the right to resell the ARM architecture itself, licensees may freely sell manufactured product such as chip devices, evaluation boards and complete systems. Merchant foundries can be a special case; not only are they allowed to sell finished silicon containing ARM cores, they generally hold the right to re-manufacture ARM cores for other customers.

ARM Holdings prices its IP based on perceived value. Lower performing ARM cores typically have lower licence costs than higher performing cores. In implementation terms, a synthesizable core costs more than a hard macro (blackbox) core. Complicating price matters, a merchant foundry that holds an ARM licence, such as Samsung or Fujitsu, can offer fab customers reduced licensing costs. In exchange for acquiring the ARM core through the foundry's in-house design services, the customer can reduce or eliminate payment of ARM's upfront licence fee.

Compared to dedicated semiconductor foundries (such as TSMC and UMC) without in-house design services, Fujitsu/Samsung charge two- to three-times more per manufactured wafer. Wikipedia:Citation needed For low to mid



Die of a STM32F103VGT6 ARM Cortex-M3 microcontroller with 1 megabyte flash memory by STMicroelectronics.

volume applications, a design service foundry offers lower overall pricing (through subsidisation of the licence fee). For high volume mass-produced parts, the long term cost reduction achievable through lower wafer pricing reduces the impact of ARM's NRE (Non-Recurring Engineering) costs, making the dedicated foundry a better choice.

Architectural licence

Companies can also obtain an *ARM architectural licence* for designing their own CPU cores using the ARM instruction sets. These cores must comply fully with the ARM architecture.

Cores

Main article: List of ARM cores

Architecture	Bit width	Cores designed by ARM Holdings	Cores designed by third parties	Cortex profile	References
ARMv1	32/26	ARM1			
ARMv2	32/26	ARM2, ARM3	Amber, STORM Open Soft Core		
ARMv3	32	ARM6, ARM7			
ARMv4	32	ARM8	StrongARM, FA526		
<u>ARMv4T</u>	32	<u>ARM7TDMI</u> , <u>ARM9TDMI</u>			
<u>ARMv5</u>	32	<u>ARM7EJ</u> , <u>ARM9E</u> , <u>ARM10E</u>	<u>XScale</u> , FA626TE, Feroceon, PJ1/Mohawk		
ARMv6	32	ARM11			
ARMv6-M	32	ARM Cortex-M0, ARM Cortex-M0+, ARM Cortex-M1		Microcontroller	
ARMv7-M	32	ARM Cortex-M3		Microcontroller	
ARMv7E-M	32	ARM Cortex-M4		Microcontroller	
ARMv7-R	32	ARM Cortex-R4, ARM Cortex-R5, ARM Cortex-R7		Real-time	
<u>ARMv7-A</u>	32	ARM Cortex- <u>A5</u> , ARM Cortex- <u>A7</u> , ARM Cortex- <u>A8</u> , ARM Cortex- <u>A9</u> , ARM Cortex- <u>A12</u> , ARM Cortex- <u>A15</u> , ARM Cortex- <u>A17</u>	Krait, Scorpion, PJ4/Sheeva, Apple A6/A6X (Swift)	Application	
ARMv8-A	64/32	ARM Cortex-A53, ARM Cortex-A57	X-Gene, Denver, Apple A7 (Cyclone), AMD K12	Application	[5][6]
ARMv8-R	32	No announcements yet		Real-time	[7]

A list of vendors who implement ARM cores in their design (application specific standard products (ASSP), microprocessor and microcontrollers) is provided by ARM Holdings.

Example applications of ARM cores

Main article: List of applications of ARM cores

ARM cores are used in a number of products, particularly PDAs and smartphones. Some computing examples are the Microsoft Surface, Apple's iPad and ASUS Eee Pad Transformer. Others include Apple's iPhone smartphone and iPod portable media player, Canon PowerShot A470 digital camera, Nintendo DS handheld game console and TomTom turn-by-turn navigation system.

In 2005, ARM Holdings took part in the development of Manchester University's computer, SpiNNaker, which used ARM cores to simulate the human brain.

ARM chips are also used in Raspberry Pi, BeagleBoard, BeagleBone, PandaBoard and other single-board computers, because they are very small, inexpensive and consume very little power.



Tronsmart MK908, a Rockchip-based quad-core Android "mini PC", with a microSD card next to it for a size comparison.

32-bit architecture

The 32-bit ARM architecture, such as **ARMv7-A**, is the most widely used architecture in mobile devices.

From 1995, the *ARM Architecture Reference Manual* ^[9] has been the primary source of documentation on the ARM processor architecture and instruction set, distinguishing interfaces that all ARM processors are required to support (such as instruction semantics) from implementation details that may vary. The architecture has evolved over time, and version seven of the architecture, ARMv7, that defines the architecture for the first of the Cortex series of cores, defines three architecture "profiles":

- A-profile, the "Application" profile: Cortex-A series
- R-profile, the "Real-time" profile: Cortex-R series
- M-profile, the "Microcontroller" profile: Cortex-M series

Although the architecture profiles were first defined for ARMv7, ARM subsequently defined the ARMv6-M architecture (used by the Cortex M0/M0+/M1) as a subset of the ARMv7-M profile with fewer instructions.

CPU modes

Except in the M-profile, the 32-bit ARM architecture specifies several CPU modes, depending on the implemented architecture features. At any moment in time, the CPU can be in only one mode, but it can switch modes due to external events (interrupts) or programmatically.

- *User mode*: The only non-privileged mode.
- *FIQ mode*: A privileged mode that is entered whenever the processor accepts an FIQ interrupt.
- *IRQ mode*: A privileged mode that is entered whenever the processor accepts an IRQ interrupt.
- *Supervisor (svc) mode*: A privileged mode entered whenever the CPU is reset or when an SVC instruction is executed.
- *Abort mode*: A privileged mode that is entered whenever a prefetch abort or data abort exception occurs.
- *Undefined mode*: A privileged mode that is entered whenever an undefined instruction exception occurs.
- *System mode (ARMv4 and above)*: The only privileged mode that is not entered by an exception. It can only be entered by executing an instruction that explicitly writes to the mode bits of the CPSR.
- *Monitor mode (ARMv6 and ARMv7 Security Extensions, ARMv8 EL3)*: A monitor mode is introduced to support TrustZone extension in ARM cores.

- *Hyp mode (ARMv7 Virtualization Extensions, ARMv8 EL2)*: A hypervisor mode that supports Popek and Goldberg virtualization requirements for the non-secure operation of the CPU.

Instruction set

The original (and subsequent) ARM implementation was hardwired without microcode, like the much simpler 8-bit 6502 processor used in prior Acorn microcomputers.

The 32-bit ARM architecture (and the 64-bit architecture for the most part) includes the following RISC features:

- Load/store architecture.
- No support for unaligned memory accesses in the original version of the architecture. ARMv6 and later, except some microcontroller versions, support unaligned accesses for half-word and single-word load/store instructions with some limitations, such as no guaranteed atomicity.
- Uniform 16× 32-bit register file (including the Program Counter, Stack Pointer and the Link Register).
- Fixed instruction width of 32 bits to ease decoding and pipelining, at the cost of decreased code density. Later, the Thumb instruction set added 16-bit instructions and increased code density.
- Mostly single clock-cycle execution.

To compensate for the simpler design, compared with processors like the Intel 80286 and Motorola 68020, some additional design features were used:

- Conditional execution of most instructions reduces branch overhead and compensates for the lack of a branch predictor.
- Arithmetic instructions alter condition codes only when desired.
- 32-bit barrel shifter can be used without performance penalty with most arithmetic instructions and address calculations.
- Has powerful indexed addressing modes.
- A link register supports fast leaf function calls.
- A simple, but fast, 2-priority-level interrupt subsystem has switched register banks.

Arithmetic instructions

The ARM supports add, subtract, and multiply instructions. The integer divide instructions are only implemented by ARM cores based on the following ARM architectures:

- ARMv7-M and ARMv7E-M architectures always include divide instructions.
- ARMv7-R architecture always includes divide instructions in the Thumb instruction set, but optionally in its 32-bit instruction set.
- ARMv7-A architecture optionally includes the divide instructions. The instructions might not be implemented, or implemented only in the Thumb instruction set, or implemented in both the Thumb and ARM instructions sets, or implemented if the Virtualization Extensions are included.

Registers

Registers across CPU modes

usr	sys	svc	abt	und	irq	fiq
R0						
R1						
R2						
R3						
R4						
R5						
R6						
R7						
R8						R8_fiq
R9						R9_fiq
R10						R10_fiq
R11						R11_fiq
R12						R12_fiq
R13	R13_svc	R13_abt	R13_und	R13_irq	R13_fiq	
R14	R14_svc	R14_abt	R14_und	R14_irq	R14_fiq	
R15						
CPSR						
	SPSR_svc	SPSR_abt	SPSR_und	SPSR_irq	SPSR_fiq	

Registers R0 through R7 are the same across all CPU modes; they are never banked.

R13 and R14 are banked across all privileged CPU modes except system mode. That is, each mode that can be entered because of an exception has its own R13 and R14. These registers generally contain the stack pointer and the return address from function calls, respectively.

Aliases:

- R13 is also referred to as SP, the Stack Pointer.
- R14 is also referred to as LR, the Link Register.
- R15 is also referred to as PC, the Program Counter.

CPSR has the following 32 bits.^[8]

- M (bits 0–4) is the processor mode bits.
- T (bit 5) is the Thumb state bit.
- F (bit 6) is the FIQ disable bit.
- I (bit 7) is the IRQ disable bit.
- A (bit 8) is the imprecise data abort disable bit.
- E (bit 9) is the data endianness bit.
- IT (bits 10–15 and 25–26) is the if-then state bits.
- GE (bits 16–19) is the greater-than-or-equal-to bits.
- DNM (bits 20–23) is the do not modify bits.
- J (bit 24) is the Java state bit.
- Q (bit 27) is the sticky overflow bit.
- V (bit 28) is the overflow bit.

- C (bit 29) is the carry/borrow/extend bit.
- Z (bit 30) is the zero bit.
- N (bit 31) is the negative/less than bit.

Conditional execution

Almost every ARM instruction has a conditional execution feature called predication, which is implemented with a 4-bit condition code selector (the predicate). To allow for unconditional execution, one of the four-bit codes causes the instruction to be always executed. Most other CPU architectures only have condition codes on branch instructions.

Though the predicate takes up four of the 32 bits in an instruction code, and thus cuts down significantly on the encoding bits available for displacements in memory access instructions, it avoids branch instructions when generating code for small `if` statements. Apart from eliminating the branch instructions themselves, this preserves the fetch/decode/execute pipeline at the cost of only one cycle per skipped instruction.

The standard example of conditional execution is the subtraction-based Euclidean algorithm:

In the C programming language, the loop is:

```
while (i != j)
{
    if (i > j)
    {
        i -= j;
    }
    else /* i < j (since i != j in while condition) */
    {
        j -= i;
    }
}
```

In ARM assembly, the loop is:

```
loop:  CMP  Ri, Rj          ; set condition "NE" if (i != j),
                          ;           "GT" if (i > j),
                          ;           or "LT" if (i < j)
      SUBGT Ri, Ri, Rj    ; if "GT" (Greater Than), i = i-j;
      SUBLT Rj, Rj, Ri    ; if "LT" (Less Than), j = j-i;
      BNE  loop          ; if "NE" (Not Equal), then loop
```

which avoids the branches around the `then` and `else` clauses. If `Ri` and `Rj` are equal then neither of the `SUB` instructions will be executed, eliminating the need for a conditional branch to implement the `while` check at the top of the loop, for example had `SUBLE` (less than or equal) been used.

One of the ways that Thumb code provides a more dense encoding is to remove the four bit selector from non-branch instructions.

Other features

Another feature of the instruction set is the ability to fold shifts and rotates into the "data processing" (arithmetic, logical, and register-register move) instructions, so that, for example, the C statement

```
a += (j << 2);
```

could be rendered as a single-word, single-cycle instruction:

```
ADD Ra, Ra, Rj, LSL #2
```

This results in the typical ARM program being denser than expected with fewer memory accesses; thus the pipeline is used more efficiently.

The ARM processor also has features rarely seen in other RISC architectures, such as PC-relative addressing (indeed, on the 32-bit ARM the PC is one of its 16 registers) and pre- and post-increment addressing modes.

The ARM instruction set has increased over time. Some early ARM processors (before ARM7TDMI), for example, have no instruction to store a two-byte quantity.

Pipelines and other implementation issues

The ARM7 and earlier implementations have a three-stage pipeline; the stages being fetch, decode and execute. Higher-performance designs, such as the ARM9, have deeper pipelines: Cortex-A8 has thirteen stages. Additional implementation changes for higher performance include a faster adder and more extensive branch prediction logic. The difference between the ARM7DI and ARM7DMI cores, for example, was an improved multiplier; hence the added "M".

Coprocessors

The ARM architecture provides a non-intrusive way of extending the instruction set using "coprocessors" that can be addressed using MCR, MRC, MRRC, MCRR, and similar instructions. The coprocessor space is divided logically into 16 coprocessors with numbers from 0 to 15, coprocessor 15 (cp15) being reserved for some typical control functions like managing the caches and MMU operation on processors that have one.

In ARM-based machines, peripheral devices are usually attached to the processor by mapping their physical registers into ARM memory space, into the coprocessor space, or by connecting to another device (a bus) that in turn attaches to the processor. Coprocessor accesses have lower latency, so some peripherals—for example an XScale interrupt controller—are accessible in both ways: through memory and through coprocessors.

In other cases, chip designers only integrate hardware using the coprocessor mechanism. For example, an image processing engine might be a small ARM7TDMI core combined with a coprocessor that has specialised operations to support a specific set of HDTV transcoding primitives.

Debugging

All modern ARM processors include hardware debugging facilities, allowing software debuggers to perform operations such as halting, stepping, and breakpointing of code starting from reset. These facilities are built using JTAG support, though some newer cores optionally support ARM's own two-wire "SWD" protocol. In ARM7TDMI cores, the "D" represented JTAG debug support, and the "I" represented presence of an "EmbeddedICE" debug module. For ARM7 and ARM9 core generations, EmbeddedICE over JTAG was a de facto debug standard, though not architecturally guaranteed.

The ARMv7 architecture defines basic debug facilities at an architectural level. These include breakpoints, watchpoints and instruction execution in a "Debug Mode"; similar facilities were also available with EmbeddedICE. Both "halt mode" and "monitor" mode debugging are supported. The actual transport mechanism used to access the debug facilities is not architecturally specified, but implementations generally include JTAG support.

There is a separate ARM "CoreSight" debug architecture, which is not architecturally required by ARMv7 processors.

DSP enhancement instructions

To improve the ARM architecture for digital signal processing and multimedia applications, DSP instructions were added to the set. These are signified by an "E" in the name of the ARMv5TE and ARMv5TEJ architectures. E-variants also imply T, D, M and I.

The new instructions are common in digital signal processor architectures. They include variations on signed multiply–accumulate, saturated add and subtract, and count leading zeros.

SIMD extensions for multimedia

Introduced in ARMv6 architecture.^[9]

Jazelle

Main article: Jazelle

Jazelle DBX (Direct Bytecode eXecution) is a technique that allows Java Bytecode to be executed directly in the ARM architecture as a third execution state (and instruction set) alongside the existing ARM and Thumb-mode. Support for this state is signified by the "J" in the ARMv5TEJ architecture, and in ARM9EJ-S and ARM7EJ-S core names. Support for this state is required starting in ARMv6 (except for the ARMv7-M profile), though newer cores only include a trivial implementation that provides no hardware acceleration.

Thumb

To improve compiled code-density, processors since the ARM7TDMI (released in 1994^[10]) have featured *Thumb* instruction set, which have their own state. (The "T" in "TDMI" indicates the Thumb feature.) When in this state, the processor executes the Thumb instruction set, a compact 16-bit encoding for a subset of the ARM instruction set. Most of the Thumb instructions are directly mapped to normal ARM instructions. The space-saving comes from making some of the instruction operands implicit and limiting the number of possibilities compared to the ARM instructions executed in the ARM instruction set state.

In Thumb, the 16-bit opcodes have less functionality. For example, only branches can be conditional, and many opcodes are restricted to accessing only half of all of the CPU's general-purpose registers. The shorter opcodes give improved code density overall, even though some operations require extra instructions. In situations where the memory port or bus width is constrained to less than 32 bits, the shorter Thumb opcodes allow increased performance compared with 32-bit ARM code, as less program code may need to be loaded into the processor over the constrained memory bandwidth.

Embedded hardware, such as the Game Boy Advance, typically have a small amount of RAM accessible with a full 32-bit datapath; the majority is accessed via a 16-bit or narrower secondary datapath. In this situation, it usually makes sense to compile Thumb code and hand-optimize a few of the most CPU-intensive sections using full 32-bit ARM instructions, placing these wider instructions into the 32-bit bus accessible memory.

The first processor with a Thumb instruction decoder was the ARM7TDMI. All ARM9 and later families, including XScale, have included a Thumb instruction decoder.

Thumb-2

Thumb-2 technology was introduced in the *ARM1156 core*, announced in 2003. Thumb-2 extends the limited 16-bit instruction set of Thumb with additional 32-bit instructions to give the instruction set more breadth, thus producing a variable-length instruction set. A stated aim for Thumb-2 was to achieve code density similar to Thumb with performance similar to the ARM instruction set on 32-bit memory. In ARMv7 this goal can be said to have been met. Wikipedia:Citation needed

Thumb-2 extends the Thumb instruction set with bit-field manipulation, table branches and conditional execution. At the same time, the ARM instruction set was extended to maintain equivalent functionality in both instruction sets. A new "Unified Assembly Language" (UAL) supports generation of either Thumb or ARM instructions from the same source code; versions of Thumb seen on ARMv7 processors are essentially as capable as ARM code (including the ability to write interrupt handlers). This requires a bit of care, and use of a new "IT" (if-then) instruction, which permits up to four successive instructions to execute based on a tested condition, or on its inverse. When compiling into ARM code this is ignored, but when compiling into Thumb it generates an actual instruction. For example:

```
; if (r0 == r1)
CMP r0, r1
ITE EQ          ; ARM: no code ... Thumb: IT instruction
; then r0 = r2;
MOVEQ r0, r2   ; ARM: conditional; Thumb: condition via ITE 'T' (then)
; else r0 = r3;
MOVNE r0, r3   ; ARM: conditional; Thumb: condition via ITE 'E' (else)
; recall that the Thumb MOV instruction has no bits to encode "EQ" or "NE"
```

All ARMv7 chips support the Thumb instruction set. All chips in the Cortex-A series, Cortex-R series, and ARM11 series support both "ARM instruction set state" and "Thumb instruction set state", while chips in the Cortex-M series support only the Thumb instruction set.

Thumb Execution Environment (ThumbEE)

ThumbEE (erroneously called *Thumb-2EE* in some ARM documentation), marketed as Jazelle RCT ^[13] (Runtime Compilation Target), was announced in 2005, first appearing in the *Cortex-A8* processor. ThumbEE is a fourth Instruction set state, making small changes to the Thumb-2 extended Thumb instruction set. These changes make the instruction set particularly suited to code generated at runtime (e.g. by JIT compilation) in managed *Execution Environments*. ThumbEE is a target for languages such as Java, C#, Perl, and Python, and allows JIT compilers to output smaller compiled code without impacting performance.

New features provided by ThumbEE include automatic null pointer checks on every load and store instruction, an instruction to perform an array bounds check, and special instructions that call a handler. In addition, because it utilises Thumb-2 technology, ThumbEE provides access to registers r8-r15 (where the Jazelle/DBX Java VM state is held).^[11] Handlers are small sections of frequently called code, commonly used to implement high level languages, such as allocating memory for a new object. These changes come from repurposing a handful of opcodes, and knowing the core is in the new ThumbEE Instruction set state.

On 23 November 2011, ARM Holdings deprecated any use of the ThumbEE instruction set,^[12] and ARMv8 removes support for ThumbEE.

Floating-point (VFP)

VFP (Vector Floating Point) technology is an *FPU* coprocessor extension to the ARM architecture. It provides low-cost single-precision and double-precision floating-point computation fully compliant with the *ANSI/IEEE Std 754-1985 Standard for Binary Floating-Point Arithmetic*. VFP provides floating-point computation suitable for a wide spectrum of applications such as PDAs, smartphones, voice compression and decompression, three-dimensional graphics and digital audio, printers, set-top boxes, and automotive applications. The VFP architecture was intended to support execution of short "vector mode" instructions but these operated on each vector element sequentially and thus did not offer the performance of true single instruction, multiple data (SIMD) vector parallelism. This vector mode was therefore removed shortly after its introduction, to be replaced with the much more powerful NEON Advanced SIMD unit.

Some devices such as the ARM Cortex-A8 have a cut-down *VFPLite* module instead of a full VFP module, and require roughly ten times more clock cycles per float operation. Other floating-point and/or SIMD coprocessors found in ARM-based processors include FPA, FPE, iwMMXt. They provide some of the same functionality as VFP but are not opcode-compatible with it.

VFPv1

Obsolete

VFPv2

An optional extension to the ARM instruction set in the ARMv5TE, ARMv5TEJ and ARMv6 architectures. VFPv2 has 16 64-bit FPU registers.

VFPv3 or VFPv3-D32

Implemented on the Cortex-A8 and A9 ARMv7 processors. It is backwards compatible with VFPv2, except that it cannot trap floating-point exceptions. VFPv3 has 32 64-bit FPU registers as standard, adds VCVT instructions to convert between scalar, float and double, adds immediate mode to VMOV such that constants can be loaded into FPU registers.

VFPv3-D16

As above, but with only 16 64-bit FPU registers. Implemented on Cortex-R4 and R5 processors.

VFPv3-F16

Uncommon; it supports IEEE754-2008 half-precision (16-bit) floating point.

VFPv4 or VFPv4-D32

Implemented on the Cortex-A12 and A15 ARMv7 processors, Cortex-A7 optionally has VFPv4-D32 in the case of an FPU with NEON. VFPv4 has 32 64-bit FPU registers as standard, adds both half-precision extensions and fused multiply-accumulate instructions to the features of VFPv3.

VFPv4-D16

As above, but it has only 16 64-bit FPU registers. Implemented on Cortex-A5 and A7 processors (in case of an FPU without NEON).

In Debian Linux and derivatives **armhf** (**ARM hard float**) refers to the ARMv7 architecture including the additional VFP3-D16 floating-point hardware extension (and Thumb-2) above.

- Software packages and cross-compiler tools use the armhf vs. arm/armel suffixes to differentiate.

Advanced SIMD (NEON)

The *Advanced SIMD* extension (aka *NEON* or "MPE" Media Processing Engine) is a combined 64- and 128-bit SIMD instruction set that provides standardized acceleration for media and signal processing applications. NEON is included in all Cortex-A8 devices but is optional in Cortex-A9 devices. NEON can execute MP3 audio decoding on CPUs running at 10 MHz and can run the GSM adaptive multi-rate (AMR) speech codec at no more than 13 MHz. It features a comprehensive instruction set, separate register files and independent execution hardware. NEON supports 8-, 16-, 32- and 64-bit integer and single-precision (32-bit) floating-point data and SIMD operations for handling audio and video processing as well as graphics and gaming processing. In NEON, the SIMD supports up to 16 operations at the same time. The NEON hardware shares the same floating-point registers as used in VFP. Devices such as the ARM Cortex-A8 and Cortex-A9 support 128-bit vectors but will execute with 64 bits at a time, whereas newer Cortex-A15 devices can execute 128 bits at a time.

ProjectNe10^[16] is ARM's first open source project (from its inception). The Ne10 library is a set of common, useful functions written in both NEON and C (for compatibility). The library was created to allow developers to use NEON optimizations without learning NEON but it also serves as a set of highly optimized NEON intrinsic and assembly code examples for common DSP, arithmetic and image processing routines. The code is available on GitHub^[17].

Security extensions (TrustZone)

The Security Extensions, marketed as TrustZone Technology, is in ARMv6KZ and later application profile architectures. It provides a low cost alternative to adding an additional dedicated security core to an SoC, by providing two virtual processors backed by hardware based access control. This lets the application core switch between two states, referred to as worlds (to reduce confusion with other names for capability domains), in order to prevent information from leaking from the more trusted world to the less trusted world. This world switch is generally orthogonal to all other capabilities of the processor, thus each world can operate independently of the other while using the same core. Memory and peripherals are then made aware of the operating world of the core and may use this to provide access control to secrets and code on the device.^[13]

Typical applications of TrustZone Technology are to run a rich operating system in the less trusted world, and smaller security-specialized code in the more trusted world (named TrustZone Software, a TrustZone optimised version of the Trusted Foundations Software developed by Trusted Logic Mobility^[19]), allowing much tighter digital rights management for controlling the use of media on ARM-based devices, and preventing any unapproved use of the device. Trusted Foundations Software was acquired by Gemalto. Giesecke & Devrient developed a rival implementation named Mobicore. In April 2012 ARM Gemalto and Giesecke & Devrient combined their TrustZone portfolios into a joint venture Trustonic. Open Virtualization and T6 are open source implementations of the trusted world architecture for TrustZone.

In practice, since the specific implementation details of TrustZone are proprietary and have not been publicly disclosed for review, it is unclear what level of assurance is provided for a given threat model. Wikipedia:Citation needed

No-execute page protection

As of ARMv6, the ARM architecture supports no-execute page protection, which is referred to as *XN*, for *eXecute Never*.^[14]

ARMv8-R

The **ARMv8-R** sub-architecture, announced after the ARMv8-A, shares some features except that it is not 64-bit.

64/32-bit architecture

ARMv8-A

Announced in October 2011, **ARMv8-A** (often called ARMv8 although not all variants are 64-bit such as ARMv8-R) represents a fundamental change to the ARM architecture. It adds a 64-bit architecture, named "AArch64", and a new "A64" instruction set. AArch64 provides user-space compatibility with ARMv7-A ISA, the 32-bit architecture, therein referred to as "AArch32" and the old 32-bit instruction set, now named "A32". The Thumb instruction sets are referred to as "T32" and have no 64-bit counterpart. ARMv8-A allows 32-bit applications to be executed in a 64-bit OS, and a 32-bit OS to be under the control of a 64-bit hypervisor. ARM announced their Cortex-A53 and Cortex-A57 cores on 30 October 2012. Apple was the first to release an ARMv8-A compatible core (Apple A7) in a consumer product (iPhone 5S). AppliedMicro, using an FPGA, was the first to demo ARMv8-A.

To both AArch32 and AArch64, ARMv8-A makes VFPv3/v4 and advanced SIMD (NEON) standard. It also adds cryptography instructions supporting AES and SHA-1/SHA-256.

AArch64 features

- New instruction set, A64
 - Has 31 general-purpose 64-bit registers.
 - Has dedicated SP or zero register.
 - The program counter (PC) is no longer accessible as a register
 - Instructions are still 32 bits long and mostly the same as A32 (with LDM/STM instructions and most conditional execution dropped).
 - Has paired loads/stores (in place of LDM/STM).
 - No predication for most instructions (except branches).
 - Most instructions can take 32-bit or 64-bit arguments.
 - Addresses assumed to be 64-bit.
- Advanced SIMD (NEON) enhanced
 - Has 32× 128-bit registers (up from 16), also accessible via VFPv4.
 - Supports double-precision floating point.
 - Fully IEEE 754 compliant.
 - AES encrypt/decrypt and SHA-1/SHA-2 hashing instructions also use these registers.
- A new exception system
 - Fewer banked registers and modes.
- Memory translation from 48-bit virtual addresses based on the existing LPAAE, which was designed to be easily extended to 64-bit.

Operating system support

32-bit operating systems

Historical operating systems

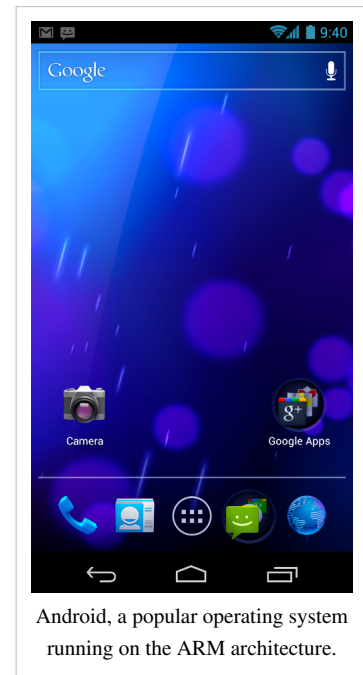
The first ARM-based personal computer, the Acorn Archimedes, ran an interim operating system called Arthur, which evolved into RISC OS, used on later ARM-based systems from Acorn and other vendors. Some Acorn machines also had a Unix port called RISC iX.

Embedded operating systems

The ARM architecture is supported by a large number of embedded and real-time operating systems, including Linux, Windows CE, Symbian, ChibiOS/RT, FreeRTOS, eCos, Integrity, Nucleus PLUS, MicroC/OS-II, PikeOS,^[15] QNX QNX, RTEMS, RTX Quadros, ThreadX, VxWorks, DRYOS, MQX, T-Kernel, OSE, SCIOPTA, OS-9, and RISC OS.

Mobile device operating systems

The ARM architecture is the primary hardware environment for most mobile device operating systems such as iOS, Android, Windows Phone, Windows RT, Bada, Blackberry OS/Blackberry 10, MeeGo, Firefox OS, Tizen, Ubuntu Touch, Sailfish and webOS.



Android, a popular operating system running on the ARM architecture.

Desktop/server operating systems

The ARM architecture is supported by RISC OS and multiple Unix-like operating systems including BSD (NetBSD, FreeBSD), OpenSolaris and various Linux distributions such as Ubuntu and Chrome OS.

64-bit operating systems

Mobile device operating systems

iOS 7 on the 64-bit Apple A7 SOC has ARMv8-A application support.

Desktop/server operating systems

Support for ARMv8-A was merged into the Linux kernel version 3.7 in late 2012. ARMv8-A is supported by a number of Linux distributions. Wikipedia:Citation needed

Windows applications can be recompiled to run on 32-bit or 64-bit ARM in Linux with Winelib.^{[16][17]}

References

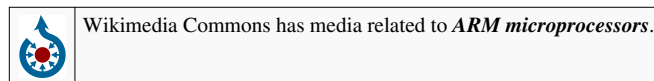
- [1] "Some facts about the Acorn RISC Machine" (<http://groups.google.com/group/comp.arch/msg/269fe7defd51f29e>) Roger Wilson posting to comp.arch, 2 November 1988. Retrieved 25 May 2007.
- [2] "ARM Cores Climb Into 3G Territory" (<http://www.extremetech.com/extreme/52180-arm-cores-climb-into-3g-territory>) by Mark Hachman, 2002.
- [3] "The Two Percent Solution" (<http://www.embedded.com/electronics-blogs/significant-bits/4024488/The-Two-Percent-Solution>) by Jim Turley 2002.
- [4] "ARM Corporate Backgrounder" (<http://www.arm.com/miscPDFs/3822.pdf>), *ARM Technology*.
- [5] ARMv8-A Architecture Webpage; ARM Holdings. (<http://www.arm.com/products/processors/instruction-set-architectures/armv8-architecture.php>)
- [6] ARMv8 Architecture Technology Preview (Slides); ARM Holdings. (http://www.arm.com/files/downloads/ARMv8_Architecture.pdf)
- [7] ARMv8-R Architecture Webpage; ARM Holdings. (<http://www.arm.com/products/processors/instruction-set-architectures/armv8-r-architecture.php>)
- [8] 2.14. The program status registers - Cortex-A8 Technical Reference Manual (<http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.ddi0290g/I27695.html>)

- [9] DSP & SIMD - ARM (<http://www.arm.com/products/processors/technologies/dsp-simd.php>)
- [10] ARM7TDMI Technical Reference Manual (http://www.atmel.com/Images/DDI0029G_7TDMI_R3_trm.pdf) page ii
- [11] "Arm strengthens Java compilers: New 16-Bit Thumb-2EE Instructions Conserve System Memory" (<http://www.arm.com/miscPDFs/10069.pdf>) by Tom R. Halfhill 2005.
- [12] ARM Architecture Reference Manual, ARMv7-A and ARMv7-R edition, issue C.b, Section A2.10, 24 July 2012.
- [13] An Exploration of ARM TrustZone Technology; genode.org (<http://genode.org/documentation/articles/trustzone>)
- [14] "APX and XN (execute never) bits have been added in VMSAv6 [Virtual Memory System Architecture]", ARM Architecture Reference Manual (<http://www.arm.com/miscPDFs/14128.pdf>). Retrieved 2009/12/01.
- [15] "PikeOS Safe and Secure Virtualization" (http://www.arm.com/community/partners/display_product/rw/ProductId/4201/). Retrieved 10 July 2013.
- [16] ARM support (<http://wiki.winehq.org/ARM>)
- [17] ARM64 support (<http://wiki.winehq.org/ARM64>)

Further reading

- *Assembly Language Programming : ARM Cortex-M3*; 1st Edition; Vincent Mahout; Wiley-ISTE; 256 pages; 2012; ISBN 978-1848213296.
- *The Definitive Guide to the ARM Cortex-M3 and Cortex-M4 Processors*; 3rd Edition; Joseph Yiu; Newnes; 600 pages; 2013; ISBN 978-0124080829.
- *The Definitive Guide to the ARM Cortex-M3*; 2nd Edition; Joseph Yiu; Newnes; 480 pages; 2009; ISBN 978-1-85617-963-8. (Online Sample) (http://books.google.com/books?id=mb5d_xeINZEC&printsec=frontcover&dq=isbn:9781856179638)
- *The Definitive Guide to the ARM Cortex-M0*; 1st Edition; Joseph Yiu; Newnes; 552 pages; 2011; ISBN 978-0-12-385477-3. (Online Sample) (<http://books.google.com/books?id=5OZblBzjsJ0C&printsec=frontcover&dq=isbn:9780123854773>)
- Yurichev, Dennis, "An Introduction To Reverse Engineering for Beginners" including ARM assembly. Online book: http://yurichev.com/writings/RE_for_beginners-en.pdf

External links



- Official website (<http://www.arm.com>) , ARM Ltd.
- ARM Virtualization Extensions (<http://www.futurechips.org/understanding-chips/arm-virtualization-extensions-introduction-part-1.html>)

Quick Reference Cards

- Instructions: Thumb (http://infocenter.arm.com/help/topic/com.arm.doc.qrc0006e/QRC0006_UAL16.pdf), ARM and Thumb-2 (http://infocenter.arm.com/help/topic/com.arm.doc.qrc0001m/QRC0001_UAL.pdf), Vector Floating Point (http://infocenter.arm.com/help/topic/com.arm.doc.qrc0007e/QRC0007_VFP.pdf)
- Opcodes: Thumb (<http://re-eject.gbadev.org/files/ThumbRefV2-beta.pdf>), Thumb (<http://www.mechcore.net/files/docs/ThumbRefV2-beta.pdf>), ARM (<http://re-eject.gbadev.org/files/armref.pdf>), ARM (<http://www.mechcore.net/files/docs/armref.pdf>), GNU Assembler Directives (<http://re-eject.gbadev.org/files/GasARMRef.pdf>).

List of ARM microarchitectures

WARNING: Article could not be rendered - ouputting plain text.

Potential causes of the problem are: (a) a bug in the pdf-writer software (b) problematic Mediawiki markup (c) table is too wide

"ARM8" redirects here. For the ARMv8 instruction set architecture, see ARM architecture#ARMv8. This is a list of microarchitectures based on the ARM architectureARM family of instruction sets designed by ARM Holdings and 3rd parties, sorted by version of the ARM instruction set, release and name. ARM provides a summary of the numerous vendors who implement ARM cores in their design. Keil (company)Keil also provides a somewhat newer summary of vendors of ARM based processors. ARM further provides a chart displaying an overview of the ARM processor lineup with performance and functionality versus capabilities for the more recent ARM core families. ARM coresDesigned by ARM ARM family ARM architecture ARM core Feature CPU cacheCache (I / D), Memory management unitMMU Typical Million instructions per secondMIPS @ MHzARM1 ARMv1 ARM1 First implementation None ARM2 ARMv2 ARM2 ARMv2 added the MUL (multiply) instruction None 4 MIPS @ 8 MHz0.33 DMIPS/MHz ARMv2a ARM250 Integrated Memory controllerMEMC (MMU), graphics and I/O processor. ARMv2a added the SWP and SWPB (swap) instructions None, MEMC1a 7 MIPS @ 12 MHz ARM3 ARMv2a ARM3 First integrated memory cache 4 KibibyteKB unified 12 MIPS @ 25 MHz0.50 DMIPS/MHz ARM6 ARMv3 ARM60 ARMv3 first to support 32-bit memory address space (previously 26-bit) None 10 MIPS @ 12 MHz ARM600 As ARM60, cache and coprocessor bus (for FPA10 floating-point unit) 4 KB unified 28 MIPS @ 33 MHz ARM610 As ARM60, cache, no coprocessor bus 4 KB unified 17 MIPS @ 20 MHz0.65 DMIPS/MHz ARM7 ARMv3 ARM700 8 KB unified 40 MHz ARM710 As ARM700, no coprocessor bus 8 KB unified 40 MHz ARM710a As ARM710 8 KB unified 40 MHz0.68 DMIPS/MHz ARM7TDMI ARMv4T ARM7TDMI(-S) 3-stage pipeline, Thumb, ARMv4 first to drop legacy ARM 26-bit Address spaceaddressingnone 15 MIPS @ 16.8 MHz63 DMIPS @ 70 MHz ARM710T As ARM7TDMI, cache 8 KB unified, MMU 36 MIPS @ 40 MHz ARM720T As ARM7TDMI, cache 8 KB unified, MMU with Fast Context Switch Extension 60 MIPS @ 59.8 MHz ARM740T As ARM7TDMI, cache Memory protectionMPUARM7EJARMv5TEJ ARM7EJ-S 5-stage pipeline, Thumb, Jazelle DBX, Enhanced DSP instructions none ARM8 ARMv4 ARM8105-stage pipeline, static branch prediction, double-bandwidth memory 8 KB unified, MMU 84 MIPS @ 72 MHz1.16 DMIPS/MHz ARM9TDMI ARMv4T ARM9TDMI 5-stage pipeline, Thumb none ARM920T As ARM9TDMI, cache 16 KB / 16 KB, MMU with FCSE (Fast Context Switch Extension) Register 13, FCSE PID register ARM920T Technical Reference Manual200 MIPS @ 180 MHz ARM922T As ARM9TDMI, caches 8 KB / 8 KB, MMU ARM940T As ARM9TDMI, caches 4 KB / 4 KB, MPU ARM9E ARMv5TE ARM946E-S Thumb, Enhanced DSP instructions, caches variable, tightly coupled memories, MPU ARM966E-S Thumb, Enhanced DSP instructions no cache, TCMs ARM968E-S As ARM966E-S no cache, TCMs ARMv5TEJ ARM926EJ-S Thumb, Jazelle DBX, Enhanced DSP instructions variable, TCMs, MMU 220 MIPS @ 200 MHz ARMv5TE ARM996HS Clockless processor, as ARM966E-S no caches, TCMs, MPU ARM10E ARMv5TE ARM1020E 6-stage pipeline, Thumb, Enhanced DSP instructions, (VFP) 32 KB / 32 KB, MMU ARM1022E As ARM1020E 16 KB / 16 KB, MMU ARMv5TEJ ARM1026EJ-S Thumb, Jazelle DBX, Enhanced DSP instructions, (VFP) variable, MMU or MPU ARM11ARMv6 ARM1136J(F)-S8-stage pipeline, SIMD, Thumb, Jazelle DBX, (VFP), Enhanced DSP instructions variable, MMU 740 @ 532–665 MHz (i.MX31 SoC), 400–528 MHz ARMv6T2 ARM1156T2(F)-S 8-stage pipeline, SIMD, Thumb-2, (VFP), Enhanced DSP instructions variable, MPU ARMv6Z ARM1176JZ(F)-S As ARM1136EJ(F)-S variable, MMU + TrustZone 965 DMIPS @ 772 MHz, up to 2,600 DMIPS with four processorsARMv6K ARM11 MPCore As ARM1136EJ(F)-S,

1–4 core SMP variable, MMU SecurCore ARMv6-M SC000 0.9 DMIPS/MHz ARMv4T SC100 ARMv7-M SC300 1.25 DMIPS/MHz ARM Cortex-M0Cortex-M ARMv6-M ARM Cortex-M0Cortex-M0 Cortex-M0 Specification Summary; ARM Holdings.Microcontroller profile, Thumb + Thumb-2 subset (BL, MRS, MSR, ISB, DSB, DMB), Cortex-M0/M0+/M1 Instruction set; ARM Holding. hardware multiply instruction (optional small), optional system timer, optional bit-banding memory Optional cache, No TCM, No MPU 0.84 DMIPS/MHz ARM Cortex-M0+Cortex-M0+ Cortex-M0+ Specification Summary; ARM Holdings.Microcontroller profile, Thumb + Thumb-2 subset (BL, MRS, MSR, ISB, DSB, DMB), hardware multiply instruction (optional small), optional system timer, optional bit-banding memory Optional cache, No TCM, optional MPU with 8 regions 0.93 DMIPS/MHz ARM Cortex-M1Cortex-M1 Cortex-M1 Specification Summary; ARM Holdings.Microcontroller profile, Thumb + Thumb-2 subset (BL, MRS, MSR, ISB, DSB, DMB), hardware multiply instruction (optional small), OS option adds SVC / banked stack pointer, optional system timer, no bit-banding memory Optional cache, 0-1024 KB I-TCM, 0-1024 KB D-TCM, No MPU 136 DMIPS @ 170 MHz, (0.8 DMIPS/MHz FPGA-dependent)ARMv7-M ARM Cortex-M3Cortex-M3 Cortex-M3 Specification Summary; ARM Holdings.Microcontroller profile, Thumb / Thumb-2, hardware multiply and divide instructions, optional bit-banding memory Optional cache, No TCM, optional MPU with 8 regions 1.25 DMIPS/MHz ARMv7E-M ARM Cortex-M4Cortex-M4 Cortex-M4 Specification Summary; ARM Holdings.Microcontroller profile, Thumb / Thumb-2 / DSP / optional FPU single-precision Floating-point unitFPU, hardware multiply and divide instructions, optional bit-banding memory Optional cache, No TCM, optional MPU with 8 regions 1.25 DMIPS/MHz ARM Cortex-RCortex-R ARMv7-R Cortex-R4 Cortex-R4 Specification Summary; ARM Holdings.Real-time profile, Thumb / Thumb-2 / DSP / optional VFPv3 Floating-point unitFPU, hardware multiply and optional divide instructions, optional parity & ECC for internal buses / cache / TCM, 8-stage pipeline dual-core running Lockstep (computing)lockstep with fault logic 0–64 KB / 0–64 KB, 0–2 of 0–8 MebibyteMB TCM, opt MPU with 8/12 regions Cortex-R5 (MPCore) Cortex-R5 Specification Summary; ARM Holdings.Real-time profile, Thumb / Thumb-2 / DSP / optional VFPv3 FPU and precision, hardware multiply and optional divide instructions, optional parity & ECC for internal buses / cache / TCM, 8-stage pipeline dual-core running lock-step with fault logic / optional as 2 independent cores, low-latency peripheral port (LLPP), accelerator coherency port (ACP) Cortex-R5 & Cortex-R7 Press Release; ARM Holdings; 31 January 2011.0–64 KB / 0–64 KB, 0–2 of 0–8 MB TCM, opt MPU with 12/16 regions Cortex-R7 (MPCore) Cortex-R7 Specification Summary; ARM Holdings.Real-time profile, Thumb / Thumb-2 / DSP / optional VFPv3 FPU and precision, hardware multiply and optional divide instructions, optional parity & ECC for internal buses / cache / TCM, 11-stage pipeline dual-core running lock-step with fault logic / out-of-order execution / dynamic register renaming / optional as 2 independent cores, low-latency peripheral port (LLPP), ACP0–64 KB / 0–64 KB, ? of 0–128 KB TCM, opt MPU with 16 regions ARM Cortex-ACortex-A ARMv7-A ARM Cortex-A5Cortex-A5 Cortex-A5 Specification Summary; ARM Holdings.Application profile, ARM / Thumb / Thumb-2 / DSP / SIMD / Optional VFPv4-D16 Floating-point unitFPU / Optional NEON / Jazelle RCT and DBX, 1–4 cores / optional MPCore, snoop control unit (SCU), generic interrupt controller (GIC), accelerator coherence port (ACP) 4-64 KB / 4-64 KB L1, MMU + TrustZone 1.57 DMIPS/MHz per core ARM Cortex-A7 MPCoreCortex-A7 MPCore Cortex-A7 Specification Summary; ARM Holdings.Application profile, ARM / Thumb / Thumb-2 / DSP / VFPv4-D16 FPU / NEON / Jazelle RCT and DBX / Hardware virtualization, in-order execution, superscalar, 1–4 SMP cores, Large Physical Address Extensions (LPAE), snoop control unit (SCU), generic interrupt controller (GIC), ACP, architecture and feature set are identical to A15, 8-10 stage pipeline, low-power design32 KB / 32 KB L1, 0–4 MB L2, MMU + TrustZone 1.9 DMIPS/MHz per core ARM Cortex-A8Cortex-A8 Cortex-A8 Specification Summary; ARM Holdings.Application profile, ARM / Thumb / Thumb-2 / VFPv3 FPU / NEON / Jazelle RCT and DAC, 13-stage superscalar pipeline 16-32 KB / 16–32 KB L1, 0–1 MB L2 opt ECC, MMU + TrustZone Up to 2000 (2.0 DMIPS/MHz in speed from 600 MHz to greater than 1 HertzGHz) ARM Cortex-A9 MPCoreCortex-A9 MPCore Cortex-A9 Specification Summary; ARM Holdings.Application profile, ARM / Thumb / Thumb-2 / DSP / Optional VFPv3 FPU / Optional NEON / Jazelle RCT and DBX, out-of-order executionout-of-order speculative executionspeculative issue superscalar, 1–4 SMP cores, snoop control unit (SCU), generic interrupt controller (GIC), accelerator coherence port (ACP) 16–64 KB /

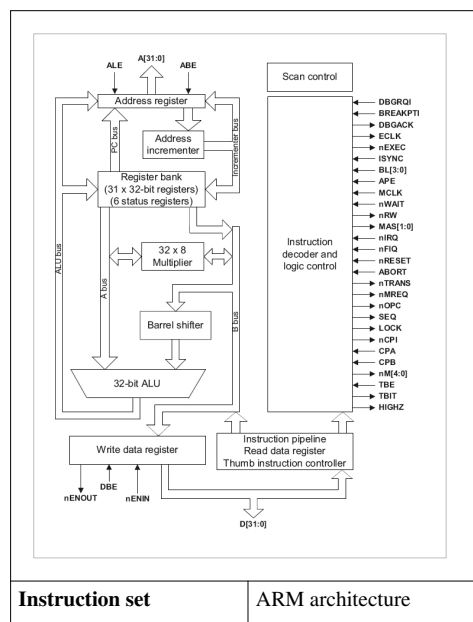
16–64 KB L1, 0–8 MB L2 opt parity, MMU + TrustZone 2.5 DMIPS/MHz per core, 10,000 DMIPS @ 2 GHz on Performance Optimized TSMC 45 nm40G (dual-core) ARM Cortex-A12Cortex-A12 Cortex-A12 Summary; ARM Holdings.Application profile, ARM / Thumb-2 / DSP / VFPv4 FPU / NEON / Hardware virtualization, out-of-order executionout-of-order speculative executionspeculative issue superscalar, 1–4 SMP cores, Large Physical Address Extensions (LPAE), snoop control unit (SCU), generic interrupt controller (GIC), accelerator coherence port (ACP) 32-64 KB / 32 KB L1, 256 KB-8 MB L2 3.0 DMIPS/MHz per core ARM Cortex-A15 MPCoreCortex-A15 MPCore Cortex-A15 Specification Summary; ARM Holdings.Application profile, ARM / Thumb / Thumb-2 / DSP / VFPv4 FPU / NEON / Integer divide / Fused MAC / Jazelle RCT / Hardware virtualization, out-of-order executionout-of-order speculative executionspeculative issue superscalar, 1–4 SMP cores, Large Physical Address Extensions (LPAE), snoop control unit (SCU), generic interrupt controller (GIC), ACP, 15-24 stage pipeline32 KB w/parity / 32 KB w/ECC memoryECC L1, 0–4 MB L2, L2 has ECC, MMU + TrustZone At least 3.5 DMIPS/MHz per core (up to 4.01 DMIPS/MHz depending on implementation) Exclusive : ARM Cortex-A15 "40 Per Cent" Faster Than Cortex-A9 | ITProPortal.comARM Cortex-A17 MPCoreCortex-A17 MPCoreApplication profile, ARM / Thumb / Thumb-2 / DSP / VFPv4 FPU / NEON / Integer divide / Fused MAC / Jazelle RCT / Hardware virtualization, out-of-order executionout-of-order speculative executionspeculative issue superscalar, 1–4 SMP cores, Large Physical Address Extensions (LPAE), snoop control unit (SCU), generic interrupt controller (GIC), ACP MMU + TrustZone Cortex-A50 ARMv8-A Cortex-A53 Application profile, AArch32 and AArch64, 1-4 SMP cores, Trustzone, NEON advanced SIMD, VFPv4, hardware virtualization, dual issue, in-order pipeline 8-64 KB w/parity / 8-64 KB w/ECC L1 per core, 128 KB-2 MB L2 shared, 40-bit physical addresses 2.3 DMIPS/MHz ARM Cortex-A57Cortex-A57 Application profile, AArch32 and AArch64, 1-4 SMP cores, Trustzone, NEON advanced SIMD, VFPv4, hardware virtualization, multi-issue, deeply out-of-order pipeline 48 KB w/DED parity / 32 KB w/ECC L1 per core, 512 KB-2 MB L2 shared, 44-bit physical addresses At least 4.1 DMIPS/MHz per core (up to 4.76 DMIPS/MHz depending on implementation) ARM family ARM architecture ARM core Feature Cache (I / D), Memory management unitMMU Typical Million instructions per secondMIPS @ MHz Designed by third parties These cores implement the ARM instruction set, and were developed independently by companies with an architectural license from ARM. Family Instruction setMicroarchitecture Feature Cache (I / D), Memory management unitMMU Typical Million instructions per secondMIPS @ MHz StrongARMARMv4 SA-110 5-stage pipeline 16 KB / 16 KB, MMU 100–206 MHz1.0 DMIPS/MHz SA-1100 derivative of the SA-110 16 KB / 8 KB, MMU Faraday TechnologyFaraday ARMv4 FA510 6-stage pipeline up to 32 KB / 32 KB Cache, MPU 1.26 DMIPS/MHz 100–200 MHz FA526 up to 32 KB / 32 KB Cache, MMU 1.26 MIPS/MHz 166-300 MHz FA626 8-stage pipeline 32 KB / 32 KB Cache, MMU 1.35 DMIPS/MHz 500 MHz ARMv5TE FA606TE 5-stage pipeline no cache, no MMU 1.22 DMIPS/MHz200 MHz FA626TE 8-stage pipeline32 KB / 32 KB Cache, MMU 1.43 MIPS/MHz800 MHz FMP626TE 8-stage pipeline, SMP 1.43 MIPS/MHz500 MHz FA726TE 13 stage pipeline, dual issue 2.4 DMIPS/MHz1000 MHz XScale ARMv5TE XScale 7-stage pipeline, Thumb, Enhanced DSP instructions 32 KB / 32 KB, MMU 133–400 MHz BulverdeWireless MMX (instruction set)MMX, Wireless SpeedStep added 32 KB / 32 KB, MMU 312–624 MHz MonahansWireless MMX2 added 32 KB / 32 KB (L1), optional L2 cache up to 512 KB, MMU up to 1.25 GHz Marvell Technology GroupMarvell Sheeva ARMv5 Feroceon 5-8 stage pipeline, single-issue 16 KB / 16 KB, MMU 600–2000 MHz Jolteon 5-8 stage pipeline, dual-issue 32 KB / 32 KB, MMU PJ1 (Mohawk) 5-8 stage pipeline, single-issue, Wireless MMX2 32 KB / 32 KB, MMU 1.46 DMIPS/MHz1.06 GHz ARMv6 / ARMv7-A PJ4 6-9 stage pipeline, dual-issue, Wireless MMX2, SMP 32 KB / 32 KB, MMU 2.41 DMIPS/MHz1.6 GHz Snapdragon (system on chip)Snapdragon ARMv7-A Scorpion (CPU)Scorpion Qualcomm's New Snapdragon S4: MSM8960 & Krait Architecture Explored; Anandtech. 1 or 2 cores. ARM / Thumb / Thumb-2 / DSP / SIMD / VFPv3 Floating-point unitFPU / NEON (128-bit wide) 256 KB L2 per core 2.1 DMIPS/MHz per core Krait (CPU)Krait 1, 2, or 4 cores. ARM / Thumb / Thumb-2 / DSP / SIMD / VFPv4 Floating-point unitFPU / NEON (128-bit wide) 4 KB / 4 KB L0, 16 KB / 16 KB L1, 512 KB L2 per core 3.3 DMIPS/MHz per core Apple A6,Apple A6X ARMv7-A Swift 2 cores. ARM / Thumb / Thumb-2 / DSP / SIMD / VFPv4 Floating-point unitFPU / NEON L1: 32 KB / 32 KB, L2: 1 MB 3.5 DMIPS/MHz per core Apple A7

ARMv8-A Cyclone 2 cores. ARM / Thumb / Thumb-2 / DSP / SIMD / VFPv4 Floating-point unit FPU / NEON / TrustZone / AArch64 L1: 64 KB / 64 KB, L2: 1 MB Applied Micro Circuits Corporation X-Gene ARMv8-A X-Gene 64-bit, quad issue, SMP Cache, MMU, virtualization 3 GHz Project Denver Denver ARMv8-A Denver 64-bit 128KB I/64KB D Up to 2.5GHz Cavium ThunderX ARMv8-A ThunderX 8-16 / 24-48 cores (x2 w/two chips). 64-bit up to 2.5 GHz ARM core timeline The following tables lists each core by the year it was announced. ARM Company Milestones. ARM Press Releases. Year ARM7 cores 1998 ARM7TDMI(-S) Year ARM8 cores 1996 ARM810 Year ARM9 cores 1997 ARM9TDMI 2003 ARM966E-S 2003 ARM968E-S 2006 ARM996HS Year ARM11 cores 2002 ARM1136J(F)-S 2003 ARM1156T2(F)-S ARM1176JZ(F)-S Year Cortex cores Embedded Real-time Application 2004 Cortex-M3 2005 Cortex-A8 2007 Cortex-M1 Cortex-A9 2009 Cortex-M0 Cortex-A5 2010 Cortex-M4 Cortex-A15 2011 Cortex-R4 Cortex-R5 Cortex-R7 Cortex-A7 2012 Cortex-M0+ Cortex-A53 Cortex-A57 2013 Cortex-A12 2014 Cortex-A17 References Further reading Digital Signal Processing and Applications Using the ARM Cortex M4; 1st Edition; Donald Reay; Wiley; 250 pages; 2014; ISBN 978-1118859049. Assembly Language Programming : ARM Cortex-M3; 1st Edition; Vincent Mahout; Wiley-ISTE; 256 pages; 2012; ISBN 978-1848213296. The Definitive Guide to the ARM Cortex-M3 and Cortex-M4 Processors; 3rd Edition; Joseph Yiu; Newnes; 600 pages; 2013; ISBN 978-0124080829. The Definitive Guide to the ARM Cortex-M0; 1st Edition; Joseph Yiu; Newnes; 552 pages; 2011; ISBN 978-0-12-385477-3. (Online Sample)

ARM7

This article is about the ARM7 family of ARM processor cores. For the ARMv7 instruction set architecture, see ARM architecture.

ARM7



ARM7 is a generation of ARM processor designs (see List of ARM microprocessor cores).

Overview

Year	ARM7 Cores
1998	ARM7TDMI(-S)

This generation introduced the Thumb 16-bit instruction set providing improved code density compared to previous designs. The most widely used ARM7 designs implement the ARMv4T architecture, but some implement ARMv3 or ARMv5TEJ. All these designs use a Von Neumann architecture, thus the few versions comprising a cache do not separate data and instruction caches.

Some ARM7 cores are obsolete. One historically significant model, the **ARM7DI**^[1] is notable for having introduced JTAG based on-chip debugging; the preceding ARM6 cores did not support it. The "D" represented a JTAG TAP for debugging; the "I" denoted an ICEBreaker debug module supporting hardware breakpoints and watchpoints, and letting the system be stalled for debugging. Subsequent cores included and enhanced this support.

It is a versatile processor designed for mobile devices and other low power electronics. This processor architecture is capable of up to 130 MIPS on a typical 0.13 μm process. The ARM7TDMI processor core implements ARM architecture **v4T**. The processor supports both 32-bit and 16-bit instructions via the ARM and Thumb instruction sets.

ARM licenses the processor to various semiconductor companies, which design full chips based on the ARM processor architecture.

Cores

ARM7

The original ARM7 was based on the earlier ARM6 design and used the same ARMv3 instruction set. The ARM710 variant was used in a CPU module for the Acorn Risc PC, and the first ARM based System on a Chip designs ARM7100 and ARM7500 used this core.

ARM7TDMI

The **ARM7TDMI** (ARM7+16 bit Thumb+j tag Debug+fast Multiplier+enhanced ICE) processor is a 32-bit RISC CPU designed by ARM, and licensed for manufacture by an array of semiconductor companies. In 2009 it remains one of the most widely used ARM cores, and is found in numerous deeply embedded system designs. Texas Instruments licensed the ARM7TDMI, which was designed into the Nokia 6110. The **ARM7TDMI-S** variant is the synthesizable core.

ARM7EJ

The **ARM7EJ** is a version of the ARM7 implementing the ARMv5TE instruction set originally introduced with the more powerful ARM9E core.

Chips

This list is incomplete; you can help by expanding it ^[2].

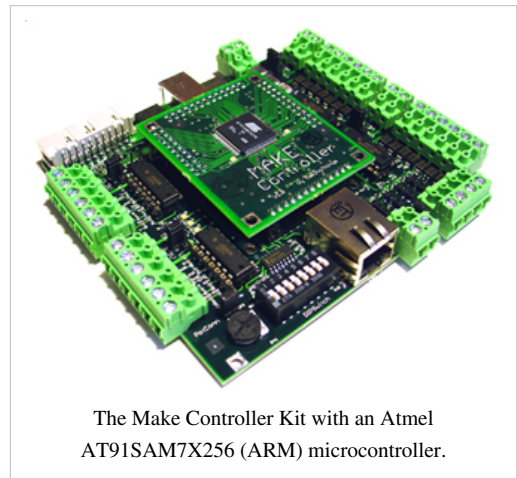
- ADMtek ADM8628
- Atmel AT91SAM7, AT91CAP7, AT91M, AT91R
- NXP LPC2100, LPC2200, LPC2300, LPC2400, LH7
- STMicroelectronics STR7
- Samsung S3C46Q0X01-EE8X, S3C44B0X

- NetSilicon NS7520
- Cirrus Logic CL-PS7110

Products

Perhaps the most common pieces of electronic equipment to have used this processor are:

- Danger Hiptop
- Audio controller in the Sega Dreamcast
- D-Link DSL-604+ Wireless ADSL Router^[2]
- iPod from Apple
- iriver portable digital audio players (the H10 uses a chip with this processor)
- Juice Box
- Lego Mindstorms NXT (Atmel AT91SAM7S256)
- Most of Nokia's mobile phone range.
- Game Boy Advance / Game Boy Advance SP / Game Boy Micro (main-processor) from Nintendo
- Nintendo DS (co-processor) from Nintendo
- PocketStation
- Psion Series 5
- Roomba 500 series from iRobot
- Sirius Satellite Radio receivers
- The main CPU in Stern Pinball S.A.M System games.
- In Building Automation, the American Auto-Matrix BBC-SD^[4] (BACnet Touchscreen Display) uses an ARM7 TDMI core
- In tournament waterski and wakeboard towboats, Perfect Pass speed control
- Many automobiles embed ARM7 cores.
- Samsung microSD cards contain an ARM7TDMI controller with 128 KB of code.



The Make Controller Kit with an Atmel AT91SAM7X256 (ARM) microcontroller.

Documentation

The amount of documentation for all ARM chips is daunting, especially for newcomers. The documentation for microcontrollers from past decades would easily be inclusive in a single document, but as chips have evolved so has the documentation grown. The total documentation is especially hard to grasp for all ARM chips since it consists of documents from the IC manufacturer and documents from CPU core vendor (ARM Holdings).

A typical top-down documentation tree is: high-level marketing slides, datasheet for the exact physical chip, a detailed reference manual that describes common peripherals and other aspects of physical chips within the same series, reference manual for the exact ARM core processor within the chip, reference manual for the ARM architecture of the core which includes detailed description of all instruction sets.

Documentation tree (top to bottom)

1. IC manufacturer marketing slides.
2. IC manufacturer datasheets.
3. IC manufacturer reference manuals.
4. ARM core reference manuals.
5. ARM architecture reference manuals.

IC manufacturer has additional documents, including: evaluation board user manuals, application notes, getting started with development software, software library documents, errata, and more.

References

- [1] "ARM7DI Data Sheet"; Document Number ARM DDI 0027D; Issued: Dec 1994.
- [2] 090506 expansys.se

External links

ARM Holdings

- Official website (<http://www.arm.com/products/processors/classic/arm7/index.php>)

Quick Reference Cards

- Instructions: Thumb (1 (http://infocenter.arm.com/help/topic/com.arm.doc.qrc0006e/QRC0006_UAL16.pdf)), ARM and Thumb-2 (2 (http://infocenter.arm.com/help/topic/com.arm.doc.qrc0001m/QRC0001_UAL.pdf)), Vector Floating Point (3 (http://infocenter.arm.com/help/topic/com.arm.doc.qrc0007e/QRC0007_VFP.pdf))
- Opcodes: Thumb (1 (<http://re-eject.gbadev.org/files/ThumbRefV2-beta.pdf>), 2 (<http://www.mechcore.net/files/docs/ThumbRefV2-beta.pdf>)), ARM (3 (<http://re-eject.gbadev.org/files/armref.pdf>), 4 (<http://www.mechcore.net/files/docs/armref.pdf>)), GNU Assembler Directives 5 (<http://re-eject.gbadev.org/files/GasARMRef.pdf>).

Other

- ARM7TDMI Microcontroller Development Resources (<http://www.arm-development.com/>) - header files, schematics, CAD files, etc..
- Source and binaries (<http://www.uclinux.org/pub/uClinux/ports/arm7tdmi/>) for running uClinux on the ARM7TDMI
- ARM Microcontroller Development HOWTO (<http://martin.hinner.info/ARM-Microcontroller-HOWTO/ARM-Microcontroller-HOWTO.html>) - Document describing development environment for ARM7 Microcontrollers on Linux.
- ARM Assembly Intro (<http://www.coranac.com/tonc/text/asm.htm>) A starter's tutorial on ARM Assembly
- Yurichev, Dennis, "An Introduction To Reverse Engineering for Beginners" including ARM assembly. Online book: http://yurichev.com/writings/RE_for_beginners-en.pdf

ARM9

ARM9 is an ARM architecture 32-bit RISC CPU family.

Overview

Year	ARM9 Cores
1997	ARM9TDMI
2003	ARM966E-S
2003	ARM968E-S
2006	ARM996HS

With this design generation, ARM moved from a von Neumann architecture (Princeton architecture) to a Harvard architecture with separate instruction and data buses (and caches), significantly increasing its potential speed. Most silicon chips integrating these cores will package them as modified Harvard architecture chips, combining the two address buses on the other side of separated CPU caches and tightly coupled memories.

There are two subfamilies, implementing different ARM architecture versions.

Differences from ARM7 cores

Key improvements over ARM7 cores, enabled by spending more transistors, include:^[1]

- Decreased heat production and lower overheating risk.
- Clock frequency improvements. Shifting from a three-stage pipeline to a five-stage one lets the clock speed be approximately doubled, on the same silicon fabrication process.
- Cycle count improvements. Many unmodified ARM7 binaries were measured as taking about 30% fewer cycles to execute on ARM9 cores. Key improvements include:
 - Faster loads and stores; many instructions now cost just one cycle. This is helped by both the modified Harvard architecture (reducing bus and cache contention) and the new pipeline stages.
 - Exposing pipeline interlocks, enabling compiler optimizations to reduce blockage between stages.

Additionally, some ARM9 cores incorporate "Enhanced DSP" instructions, such as a multiply-accumulate, to support more efficient implementations of digital signal processing algorithms.

Switching to a Harvard architecture entailed a non-unified cache, so that instruction fetches do not evict data (and vice versa). ARM9 cores have separate data and address bus signals, which chip designers use in various ways. In most cases they connect at least part of the address space in von Neumann style, used for both instructions and data, usually to an AHB interconnect connecting to a DRAM interface and an External Bus Interface usable with NOR flash memory. Such hybrids are no longer pure Harvard architecture processors.

Cores

ARM9TDMI

ARM9TDMI is a successor to the popular ARM7TDMI core, and is also based on the ARMv4T architecture. Cores based on it support both 32-bit ARM and 16-bit Thumb instruction sets and include:

- ARM920T with 16 KB each of I/D cache and an MMU
- ARM922T with 8 KB each of I/D cache and an MMU
- ARM940T with cache and a Memory Protection Unit (MPU)

ARM9E

ARM9E, and its ARM9EJ sibling, implement the basic ARM9TDMI pipeline, but add support for the ARMv5TE architecture, which includes some DSP-esque instruction set extensions. In addition, the multiplier unit width has been doubled, halving the time required for most multiplication operations. They support 32-bit, 16-bit, and sometimes 8-bit instruction sets.

- ARM926EJ-S with ARM Jazelle technology, which enables the direct execution of 8-bit Java bytecode in hardware, and an MMU
- ARM946
- ARM966
- ARM968

Chips

This list is incomplete; you can help by expanding it ^[2].

- Atmel AT91SAM9, AT91CAP9
- CSR Quatro 4300
- Cypress Semiconductor EZ-USB FX3
- Digi NS9215, NS9210 ^[3]
- NXP Semiconductors LPC3200, LPC3100, LPC2900, LH7A
- Freescale Semiconductor i.MX1x and i.MX2x
- Marvell Kirkwood
- Qualcomm Atheros AR6400
- Samsung S3C24xx
- STMicroelectronics STR9 series, ^[2] Nomadik
- Texas Instruments OMAP 1
- Texas Instruments Sitara AM1x
- Via WonderMedia 8505 and 8650
- MediaTek MT6516, MT6573
- Zilog Encore! 32

Products

- Actiontec MI-424WR REV. I MOCA 1.1 WLAN Router (ARM926EJ-S)
- Excito Bubba B1 Server (ARM9 200 MHz)
- Buffalo network-attached storage series Linkstation Pro and KuroBox Pro
- Canon EOS 5D Mark II digital SLR camera
- Chumby
- Cisco Tandberg C20 (ARM926EJ-S)
- Condor Technology/Medion OYO E-book reader
- Conexant 802.11 products
- Data Robotics DroboFS, NAS
- D-Link DNS-321 NAS (ARM926EJ-S)
- Fiat-Chrysler Group Blue&Me
- Freecom Musicpal internet radio (ARM926EJ)
- GP2X Wiz
- Hanlin eReader
- Nintendo DS and DSi
- Nintendo Wii's ATI Hollywood graphic chip (security coprocessor)
- Palm's Z22
- PlayStation Portable's WLAN chip. [5]
- SATEL SATELLAR Digital System radio modem [6]
- Seagate FreeAgent DockStar^[7] STDS10G-RK (ARM926EJ-S rev 1 (v5I))
- Seagate BlackArmor NAS BlackArmor^[8] (ARM926EJ-S rev 1 (v5I))
- Synology's Disk Station (models DS-x07+), Cube Station CS-407, and Rack Station RS-407 [9]
- Texas Instruments TI-Nspire graphing calculator @ 90/150 MHz
- VEX Robotics Robot controller, 802.11 b/g wireless, 200 MHz [10]
- VTech V.Flash educational consoles^[3]
- Western Digital My Book series NAS devices
- Zyxel NSA310^[4]/NSA320^[5] NAS (ARM926EJ-S)
- many mobile phones from
 - HTC (e.g. HTC Wizard)
 - LG (LG Cookie (KP500) at 175 MHz)
 - Nokia (like Nokia 5220 XpressMusic, Nokia N-Gage and almost all N-Series Smartphones at 100-330 MHz)
 - Philips
 - Siemens/BenQ (x65 series and newer)
 - Sony Ericsson (K, M and W series, usually at 208 MHz)
 - G-Tide G70 (200 MHz with coprocessor for TV)
- Sun Service Processor running the ILOM Java stack
- Hewlett Packard Hp 50g @ 75 MHz
- Ingenico i5100 EFTPOS Terminals
- Iomega StorCenter ix2 NAS (ARM926EJ-S)
- Lacie EtherDisk
- Livescribe Pulse and Echo Smartpens
- Logitech Squeezebox (network music player) Radio (ARM926EJ)
- Lego Mindstorms EV3 (Sitara AM1808)

Documentation

The amount of documentation for all ARM chips is daunting, especially for newcomers. The documentation for microcontrollers from past decades would easily be inclusive in a single document, but as chips have evolved so has the documentation grown. The total documentation is especially hard to grasp for all ARM chips since it consists of documents from the IC manufacturer and documents from CPU core vendor (ARM Holdings).

A typical top-down documentation tree is: high-level marketing slides, datasheet for the exact physical chip, a detailed reference manual that describes common peripherals and other aspects of physical chips within the same series, reference manual for the exact ARM core processor within the chip, reference manual for the ARM architecture of the core which includes detailed description of all instruction sets.

Documentation tree (top to bottom)

1. IC manufacturer marketing slides.
2. IC manufacturer datasheets.

3. IC manufacturer reference manuals.
4. ARM core reference manuals.
5. ARM architecture reference manuals.

IC manufacturer has additional documents, including: evaluation board user manuals, application notes, getting started with development software, software library documents, errata, and more.

References

- [1] "Performance of the ARM9TDMI and ARM9E-S cores compared to the ARM7TDMI core", Issue 1.0, dated 9 February 2000, ARM Ltd.
- [2] STR9 Website; STMicroelectronics. (<http://www.st.com/internet/mcu/subclass/828.jsp>)
- [3] VTech V.Flash product page from ARM (http://www.arm.com/markets/emerging_applications/armpp/15800.html)
- [4] <http://zyxel.nas-central.org/wiki/Category:NSA-310>
- [5] <http://zyxel.nas-central.org/wiki/Category:NSA-320>

External links

ARM Holdings

- Official website (<http://www.arm.com/products/processors/classic/arm9/index.php>)

Quick Reference Cards

- Instructions: Thumb (1 (http://infocenter.arm.com/help/topic/com.arm.doc.qrc0006e/QRC0006_UAL16.pdf)), ARM and Thumb-2 (2 (http://infocenter.arm.com/help/topic/com.arm.doc.qrc0001m/QRC0001_UAL.pdf)), Vector Floating Point (3 (http://infocenter.arm.com/help/topic/com.arm.doc.qrc0007e/QRC0007_VFP.pdf))
- Opcodes: Thumb (1 (<http://re-eject.gbadev.org/files/ThumbRefV2-beta.pdf>), 2 (<http://www.mechcore.net/files/docs/ThumbRefV2-beta.pdf>)), ARM (3 (<http://re-eject.gbadev.org/files/armref.pdf>), 4 (<http://www.mechcore.net/files/docs/armref.pdf>)), GNU Assembler Directives 5 (<http://re-eject.gbadev.org/files/GasARMRef.pdf>).
- Yurichev, Dennis, "An Introduction To Reverse Engineering for Beginners" including ARM assembly. Online book: http://yurichev.com/writings/RE_for_beginners-en.pdf

ARM11

ARM11 is a family of ARM architecture 32-bit RISC microprocessor cores.

Overview

Announced	
Year	Core
2002	ARM1136J(F)-S
2003	ARM1156T2(F)-S
2003	ARM1176JZ(F)-S

The ARM11 microarchitecture (announced 29 April 2002) introduced the ARMv6 architectural additions which had been announced in October 2001. These include SIMD media instructions, multiprocessor support and a new cache architecture. The implementation included a significantly improved instruction processing pipeline, compared to previous ARM9 or ARM10 families, and is used in smartphones from Apple, Nokia, and others. The initial ARM11 core (ARM1136) was released to licensees in October 2002.

The ARM11 family are currently the only ARMv6-architecture cores. There are however ARMv6-M cores (Cortex-M0 and Cortex-M1), addressing microcontroller applications;^[1] ARM11 cores target more demanding applications.

Differences from ARM9

In terms of instruction set, the ARM11 builds on the preceding ARM9 generation. It incorporates all ARM926EJ-S features and adds the ARMv6 instructions for media support (SIMD) and accelerating IRQ response.

Microarchitecture improvements in ARM11 cores^[2] include:

- SIMD instructions which can double MPEG-4 and audio digital signal processing algorithm speed
- Cache is physically addressed, solving many cache aliasing problems and reducing context switch overhead.
- Unaligned and mixed-endian data access is supported.
- Reduced heat production and lower overheating risk
- Redesigned pipeline, supporting faster clock speeds (target up to 1 GHz)
 - Longer: 8 (vs 5) stages
 - Out-of-order completion for some operations (e.g. stores)
 - Dynamic branch prediction/folding (like XScale)
 - Cache misses don't block execution of non-dependent instructions.
 - Load/store parallelism
 - ALU parallelism
- 64-bit data paths

JTAG debug support (for halting, stepping, breakpoints, and watchpoints) was simplified. The EmbeddedICE module was replaced with an interface which became part of the ARMv7 architecture. The hardware tracing modules (ETM and ETB) are compatible, but updated, versions of those used in the ARM9. In particular, trace semantics were updated to address parallel instruction execution and data transfers.

ARM makes an effort to promote good Verilog coding styles and techniques. This ensures semantically rigorous designs, preserving identical semantics throughout the chip design flow, which included extensive use of formal verification techniques. Without such attention, integrating an ARM11 with third party designs could risk exposing hard-to-find latent bugs. Due to ARM cores being integrated into many different designs, using a variety of logic

synthesis tools and chip manufacturing processes, the impact of its register-transfer level (RTL) quality is magnified many times.^[3] The ARM11 generation focused more on synthesis than previous generations, making such concerns be more of an issue.

Cores

There are four ARM11 cores:

- ARM1136^[4]
- ARM1156, introduced Thumb2 instructions
- ARM1176, introduced security extensions^[5]
- ARM11MPcore, introduced multicore support

Chips

This list is incomplete; you can help by expanding it ^[6].

- Ambarella A5s, A7, A7L
- Broadcom BCM2835 (Raspberry Pi), BCM21553
- Cavium ECONA CNS3000 series ^[6]
- CSR Quatro 4230, Quatro 4500 series, Quatro 5300 series
- Freescale Semiconductor i.MX3x series, such as i.MX31, i.MX35
- Nintendo 1048 0H
- Infotmic IMAPX200, IMAPX210, IMAPX220
- Nvidia Tegra
- PLX Technology NAS7820, NAS7821, NAS7825
- MediaTek MTK6573_S01 / MTK6573_S00
- Qualcomm MSM720x, MSM7x27
- Qualcomm Atheros AR7400
- Samsung S3C64x0, S5P6422
- Telechips [tcc8902]
- Texas Instruments OMAP2 series, with a TMS320 C55x or C64x DSP as a second core

Products

Amazon.com

- Kindle 2

Apple Inc.

- iPhone
- iPhone 3G
- iPod Touch
- iPod Touch 2G

Aluratek

- Cinepad AT107F

Alcatel

- One Touch 991

Commtiva

- Z71
-

GeeksPhone

- GeeksPhone ONE ^[8]

HTC

- HTC Aria
- HTC Dream
- HTC Wildfire
- HTC Hero
- HTC Magic
- HTC Legend
- HTC Touch Diamond
- HTC Touch Pro
- HTC Touch Diamond 2 (or Topaz)
- HTC Touch Pro 2 (or Rhodium)
- HTC TyTN II
- HTC Wildfire S

Janam

- Janam XM60+
- Janam XM66

Huawei

- Huawei U8160
- Huawei U8500 with Qualcomm MSM7225
- Huawei U8510 Ideos X3
- Huawei U8650 Sonic

LG

- LG A230/A235 brava
- LG GW620
- LG Optimus Me P350
- LG Optimus S
- LG Optimus One
- LG Optimus Net P690
- LG Optimus GT540
- LG Optimus Hub

Micromax

- A44
- A45
- A50
- A60
- A70
- A73
- A75

Microsoft

- Zune HD
- Kin One

Motorola

- Motorola Backflip
- Motorola RIZR Z8
- Motorola Q9 Series
- Motorola XT300
- Motorola XT311
- Motorola XT316
- Motorola XT317
- Motorola Fire XT
- Motorola ES400^[7]

Nintendo

- Nintendo 3DS
- Nintendo 2DS

Nokia

- Nokia 9500 Communicator
- Nokia 3710 Fold
- Nokia 500
- Nokia 5230
- Nokia 5320 XpressMusic
- Nokia 5233
- Nokia 5700 XpressMusic
- Nokia 5800 XpressMusic
- Nokia 5530 XpressMusic
- Nokia X6
- Nokia 6120 Classic
- Nokia 6210 Navigator
- Nokia 6220 Classic
- Nokia 6290
- Nokia 6700 Classic
- Nokia 6710 Navigator
- Nokia 6720 Classic
- Nokia 603
- Nokia 700
- Nokia 701
- Nokia C5-00
- Nokia C5-03
- Nokia C5-05
- Nokia C6-00
- Nokia C6-01
- Nokia C7
- Nokia E5
- Nokia E51
- Nokia E52
- Nokia E55
- Nokia E63
- Nokia E71
- Nokia E72

- Nokia E73 Mode
- Nokia E75
- Nokia E7-00
- Nokia E90 Communicator
- Nokia N79
- Nokia N81
- Nokia N82
- Nokia N85
- Nokia N86 8MP
- Nokia N93
- Nokia N95
- Nokia N97
- Nokia N97 mini
- Nokia N8
- Nokia N800 Internet tablet
- Nokia N810 Internet tablet
- Nokia Oro
- Nokia 808 PureView

Ouku

- Ouku Horizon/P801W

Palm

- Palm Pixi
- Pandigital R7T40WWHF1 Novel

Raspberry Pi Foundation

- Raspberry Pi

Ritroid T1

Roku

- Roku LT (Some models)
- Roku 2 HD
- Roku 2 XD
- Roku 2 XS
- Roku Streaming Stick

Samsung

- Samsung SGH-i627
 - Samsung Galaxy Ace
 - Samsung Galaxy Europa
 - Samsung Galaxy Spica/Portal/Lite
 - Samsung Galaxy 3/Apollo
 - Samsung i7500 Galaxy
 - Samsung Behold II
 - Samsung Omnia II
 - Samsung OmniaPRO B7330
 - Samsung Moment
 - Samsung M910 Intercept
 - Samsung Galaxy Mini
-

- Samsung Galaxy Y
- Samsung Galaxy Gio
- Samsung Galaxy Pocket

Smart Devices

- SmartQ 5
- SmartQ V5
- SmartQ V5II
- SmartQ V7
- SmartQ N7
- SmartQ T7
- SmartQ R10
- Pandigital Novel (white version)

Sony Ericsson

- Sony Ericsson Xperia X1
- Sony Ericsson Xperia X2
- Sony Ericsson Xperia X8
- Sony Ericsson Xperia X10 mini
- Sony Ericsson Elm
- Sony Ericsson W995

Videocon

- Zeus/V7500

Zeebo

- Zeebo (game console)

ZTE

- ZTE Blade (also sold as ZTE San Francisco, Orange San Francisco, Lutea San Francisco, Dell XCD35, ...)
- ZTE Skate (also sold as Orange Monte Carlo)
- Some digital picture frames (digital media)

Documentation

The amount of documentation for all ARM chips is daunting, especially for newcomers. The documentation for microcontrollers from past decades would easily be inclusive in a single document, but as chips have evolved so has the documentation grown. The total documentation is especially hard to grasp for all ARM chips since it consists of documents from the IC manufacturer and documents from CPU core designer. (ARM Holdings).

A typical top-down documentation tree is: high-level marketing slides, datasheet for the exact physical chip, a detailed reference manual that describes common peripherals and other aspects of physical chips within the same series, reference manual for the exact ARM core processor within the chip, reference manual for the ARM architecture of the core which includes detailed description of all instruction sets.

Documentation tree (top to bottom)

1. IC manufacturer marketing slides
2. IC manufacturer datasheets
3. IC manufacturer reference manuals
4. ARM core reference manuals
5. ARM architecture reference manuals

IC manufacturer has additional documents, including: evaluation board user manuals, application notes, getting started with development software, software library documents, errata, and more.

References

- [1] not supported by Linux as of version 3.3
- [2] "The ARM11 Microarchitecture", ARM Ltd, 2002
- [3] *The Dangers of Living with an X (bugs hidden in your Verilog)*, Version 1.1 (14 October 2003).
- [4] ARM1136JF-S and ARM1136J-S Technical Reference Manual (http://infocenter.arm.com/help/topic/com.arm.doc.ddi0211k/DDI0211K_arm1136_r1p5_trm.pdf) Revision: r1p5; ARM DDI 0211K
- [5] ARM1176JZF-6 Technical Reference Manual (http://infocenter.arm.com/help/topic/com.arm.doc.ddi0301h/DDI0301H_arm1176jzfs_r0p7_trm.pdf) Revision: r0p7; accessed on 4 October 2012.
- [6] ECONA CNS3XXX ARM Based SoC Processors (http://www.cavium.com/ECONA_CNS3XXX.html) at Cavium.com
- [7] http://www.gsmarena.com/motorola_es400-3409.php

External links

ARM Holdings

- Official website (<http://www.arm.com/products/processors/classic/arm11/>)
- ARM11 Technical Reference Manuals (<http://infocenter.arm.com/help/topic/com.arm.doc.set.arm11/index.html>)
- ARMv6 Architecture Reference Manual (<http://infocenter.arm.com/help/topic/com.arm.doc.ddi0406c/index.html>) (requires registration)

Quick Reference Cards

- Instructions: Thumb (1 (http://infocenter.arm.com/help/topic/com.arm.doc.qrc0006e/QRC0006_UAL16.pdf)), ARM and Thumb-2 (2 (http://infocenter.arm.com/help/topic/com.arm.doc.qrc0001m/QRC0001_UAL.pdf)), Vector Floating Point (3 (http://infocenter.arm.com/help/topic/com.arm.doc.qrc0007e/QRC0007_VFP.pdf))
- Opcodes: Thumb (1 (<http://re-eject.gbadev.org/files/ThumbRefV2-beta.pdf>)), 2 (<http://www.mechcore.net/files/docs/ThumbRefV2-beta.pdf>)), ARM (3 (<http://re-eject.gbadev.org/files/armref.pdf>)), 4 (<http://www.mechcore.net/files/docs/armref.pdf>)), GNU Assembler Directives 5 (<http://re-eject.gbadev.org/files/GasARMRef.pdf>).

Other

- ARM11 lacks an integer hardware division instruction (<http://wanderingcoder.net/2010/07/19/ought-arm/>)
- Yurichev, Dennis, "An Introduction To Reverse Engineering for Beginners" including ARM assembly. Online book: http://yurichev.com/writings/RE_for_beginners-en.pdf

ARM Cortex-A

The **ARM Cortex-A** is a group of 32-bit RISC ARM processor cores licensed by ARM Holdings. The cores are intended for application use, and consists of the ARM Cortex-A5, ARM Cortex-A7, ARM Cortex-A8, ARM Cortex-A9, ARM Cortex-A12, ARM Cortex-A15, ARM Cortex-A17 MPCore.

Overview

Main article: ARM architecture

Announced	
Year	Core
2005	Cortex-A8
2007	Cortex-A9
2009	Cortex-A5
2010	Cortex-A15
2011	Cortex-A7
2013	Cortex-A12
2014	Cortex-A17

ARM license

ARM Holdings does not manufacture nor sell CPU devices based on its own designs, but rather, licenses the processor architecture to interested parties. ARM offers a variety of licensing terms, varying in cost and deliverables. To all licensees, ARM provides an integratable hardware description of the ARM core, as well as complete software development toolset, and the right to sell manufactured silicon containing the ARM CPU.

Silicon customization

Integrated device manufacturers (IDM) receive the ARM Processor IP as synthesizable RTL (written in Verilog). In this form, they have the ability to perform architectural level optimizations and extensions. This allows the manufacturer to achieve custom design goals, such as higher clock speed, very low power consumption, instruction set extensions, optimizations for size, debug support, etc. To determine which components have been included in an ARM IC chip, consult the manufacturer datasheet and related documentation.

Instruction sets

The Cortex-A5 / A7 / A8 / A9 / A12 / A15 / A17 cores implement the ARMv7-A architecture.

Documentation

The amount of documentation for all ARM chips is daunting, especially for newcomers. The documentation for microcontrollers from past decades would easily be inclusive in a single document, but as chips have evolved so has the documentation grown. The total documentation is especially hard to grasp for all ARM chips since it consists of documents from the IC manufacturer and documents from CPU core vendor (ARM Holdings).

A typical top-down documentation tree is: high-level marketing slides, datasheet for the exact physical chip, a detailed reference manual that describes common peripherals and other aspects of physical chips within the same series, reference manual for the exact ARM core processor within the chip, reference manual for the ARM

architecture of the core which includes detailed description of all instruction sets.

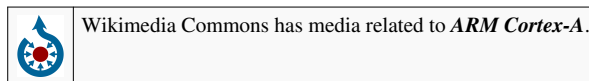
Documentation Tree (top to bottom)

1. IC Manufacturer Marketing Slides.
2. IC Manufacturer Datasheets.
3. IC Manufacturer Reference Manuals.
4. ARM Core Reference Manuals.
5. ARM Architecture Reference Manuals.[□]

IC Manufacturers usually have additional documents, including: evaluation board user manuals, application notes, getting started with development software, software library documents, errata, and more.

References

External links



Official

- ARM Cortex-A official website (<http://www.arm.com/products/processors/cortex-a/index.php/>)
- ARM Cortex-A5 (<http://www.arm.com/products/processors/cortex-a/cortex-a5.php>)
- ARM Cortex-A7 (<http://www.arm.com/products/processors/cortex-a/cortex-a7.php>)
- ARM Cortex-A8 (<http://www.arm.com/products/processors/cortex-a/cortex-a8.php>)
- ARM Cortex-A9 (<http://www.arm.com/products/processors/cortex-a/cortex-a9.php>)
- ARM Cortex-A12 (<http://www.arm.com/products/processors/cortex-a/cortex-a12-processor.php>)
- ARM Cortex-A15 (<http://www.arm.com/products/processors/cortex-a/cortex-a15.php>)
- ARM Cortex-A17 (<http://www.arm.com/products/processors/cortex-a/cortex-a17-processor.php>)

Quick Reference Cards

- Instructions: Thumb (1 (http://infocenter.arm.com/help/topic/com.arm.doc.qrc0006e/QRC0006_UAL16.pdf)), ARM and Thumb-2 (2 (http://infocenter.arm.com/help/topic/com.arm.doc.qrc0001m/QRC0001_UAL.pdf)), Vector Floating-Point (3 (http://infocenter.arm.com/help/topic/com.arm.doc.qrc0007e/QRC0007_VFP.pdf)), arm.com
- Opcodes: Thumb (1 (<http://re-eject.gbadev.org/files/ThumbRefV2-beta.pdf>)), 2 (<http://www.mechcore.net/files/docs/ThumbRefV2-beta.pdf>)), ARM (3 (<http://re-eject.gbadev.org/files/armref.pdf>)), 4 (<http://www.mechcore.net/files/docs/armref.pdf>)), GNU Assembler Directives (5 (<http://re-eject.gbadev.org/files/GasARMRef.pdf>)).

XScale

For the model railroad scale, see *X* scale.

XScale is a microarchitecture for central processing units initially designed by Intel implementing the ARM architecture (version 5) instruction set. XScale comprises several distinct families: IXP, IXC, IOP, PXA and CE (see more below), with some recent models designed as SoCs. Intel sold the PXA family to Marvell Technology Group in June 2006.^[1] Marvell then extended the brand to include processors with other microarchitectures, like ARM's Cortex.

The XScale architecture is based on the ARMv5TE ISA without the floating point instructions. XScale uses a seven-stage integer and an eight-stage memory super-pipelined microarchitecture. It is the successor to the Intel StrongARM line of microprocessors and microcontrollers, which Intel acquired from DEC's Digital Semiconductor division as part of a settlement of a lawsuit between the two companies. Intel used the StrongARM to replace its ailing line of outdated RISC processors, the i860 and i960.

All the generations of XScale are 32-bit ARMv5TE processors manufactured with a 0.18 μm or 0.13 μm (as in IXP43x parts) process and have a 32 KB data cache and a 32 KB instruction cache. First and second generation XScale multi-core processors also have a 2 KB mini-data cache. Products based on the 3rd generation XScale have up to 512 KB unified L2 cache.^[2]

Processor families

The XScale core is used in a number of microcontroller families manufactured by Intel and Marvell, notably:

- Application Processors (with the prefix PXA). There are four generations of XScale Application Processors, described below: PXA210/PXA25x, PXA26x, PXA27x, and PXA3xx.
- I/O Processors (with the prefix IOP)
- Network Processors (with the prefix IXP)
- Control Plane Processors (with the prefix IXC).
- Consumer Electronics Processors (with the prefix CE).

There are also standalone processors: the 80200 and 80219 (targeted primarily at PCI applications).

PXA

PXA210/PXA25x

The PXA210 was Intel's entry-level XScale targeted at mobile phone applications. It was released with the PXA250 in February 2002 and comes clocked at 133 MHz and 200 MHz.

The PXA25x family (code-named **Cotulla**) consists of the PXA250 and PXA255. The PXA250 was Intel's first generation of XScale processors. There was a choice of three clock speeds: 200 MHz, 300 MHz and 400 MHz. It came out in February 2002. In March 2003, the revision C0 of the PXA250 was renamed to PXA255. The main differences were a doubled internal bus speed (100 MHz to 200 MHz) for faster data transfer, lower core voltage (only 1.3 V at 400 MHz) for lower power consumption and writeback functionality for the data cache, the lack of which had severely impaired performance on the PXA250.

PXA26x

The PXA26x family (code-named **Dalhart**) consists of the PXA260 and PXA261-PXA263. The PXA260 is a stand-alone processor clocked at the same frequency as the PXA25x, but features a TPBGA package which is about 53% smaller than the PXA25x's PBGA package. The PXA261-PXA263 are the same as the PXA260 but have Intel StrataFlash memory stacked on top of the processor in the same package; 16 MB of 16-bit memory in the PXA261, 32 MB of 16-bit memory in the PXA262 and 32 MB of 32-bit memory in the PXA263. The PXA26x family was released in March 2003.

PXA27x

The PXA27x family (code-named **Bulverde**) consists of the PXA270 and PXA271-PXA272 processors. This revision is a huge update to the XScale family of processors. The PXA270 is clocked in four different speeds: 312 MHz, 416 MHz, 520 MHz and 624 MHz and is a stand-alone processor with no packaged memory. The PXA271 can be clocked to 13, 104, 208 MHz or 416 MHz and has 32 MB of 16-bit stacked StrataFlash memory and 32 MB of 16-bit SDRAM in the same package. The PXA272 can be clocked to 312 MHz, 416 MHz or 520 MHz and has 64 MB of 32-bit stacked StrataFlash memory.



e-con Systems eSOM270 Marvell XScale PXA270 Computer on module

Intel also added many new technologies to the PXA27x family such as:

- SpeedStep: the operating system can clock the processor down based on load to save power.
- Wireless MMX: 43 new SIMD instructions containing the full MMX instruction set and the integer instructions from Intel's SSE instruction set along with some instructions unique to the XScale. Wireless MMX provides 16 extra 64-bit registers that can be treated as an array of two 32-bit words, four 16-bit halfwords or eight 8-bit bytes. The XScale core can then perform up to eight adds or four MACs in parallel in a single cycle. This capability is used to boost speed in decoding and encoding of multimedia and in playing games.
- Additional peripherals, such as a USB-Host interface and a camera interface.
- Internal 256 KB SRAM to reduce power consumption and latency.

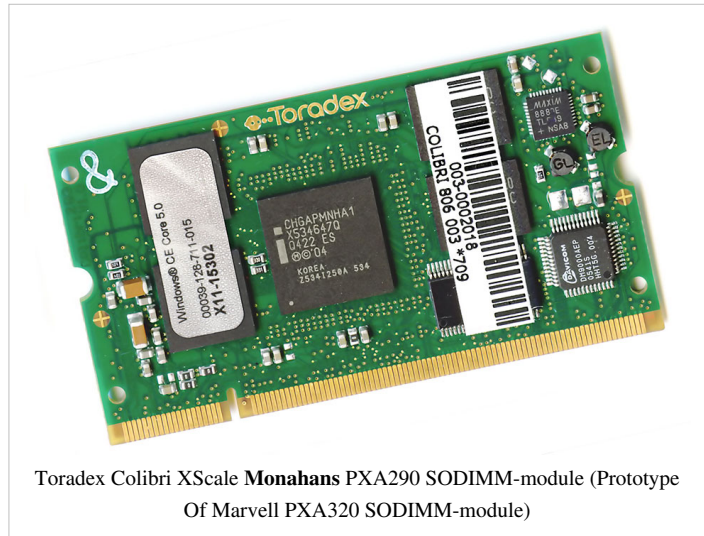
The PXA27x family was released in April 2004. Along with the PXA27x family Intel released the 2700G embedded graphics co-processor.

PXA3xx

In August 2005 Intel announced the successor to **Bulverde**, codenamed **Monahans**.

They demonstrated it showing its capability to play back high definition encoded video on a PDA screen.

The new processor was shown clocked at 1.25 GHz but Intel said it only offered a 25% increase in performance (800 MIPS for the 624 MHz PXA270 processor vs. 1000 MIPS for 1.25 GHz **Monahans**). An announced successor to the 2700G graphics processor, code named Stanwood, has since been canceled. sd features of Stanwood are integrated into **Monahans**. For extra graphics capabilities, Intel recommends third-party chips like the NVIDIA GoForce chip family.



Torodex Colibri XScale **Monahans** PXA290 SODIMM-module (Prototype Of Marvell PXA320 SODIMM-module)

In November 2006, Marvell Semiconductor officially introduced the **Monahans** family as Marvell PXA320, PXA300, and PXA310.^[3] PXA320 is currently shipping in high volume, and is scalable up to 806 MHz. PXA300 and PXA310 deliver performance "scalable to 624 MHz", and are software-compatible with PXA320.

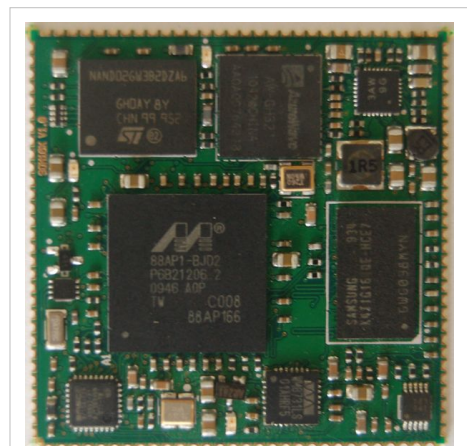
PXA90x

The PXA90x was released by Marvell and combines an XScale Core with a GSM/CDMA communication module.^[4] The PXA90x is build using a 130 nm process^[5]

PXA16x

PXA16x is a processor designed by Marvell, combining the earlier Intel designed PXA SoC components with a new ARMv5TE CPU core named **Mohawk** or **PJ1** from Marvell's **Sheeva** family instead of using wdc Xscale or ARM design. The CPU core is derived from the **Feroceon** core used in Marvell's embedded Kirkwood product line, but extended for instruction level compatibility with the XScale IWMMX.

The PXA16x delivers strong performance at a mass market price point for cost sensitive consumer and embedded markets such as digital picture frames, E Readers, multifunction printer user interface (UI) displays, interactive VoIP phones, IP surveillance cameras, and home control gadgets.^[6]



PXA168 System On Module by tianyeit.com

PXA930/935

The PXA930 and PXA935 processor series were again built using the Sheeva microarchitecture developed by Marvell but upgraded to ARMv7 instruction set compatibility.^[7] This core is a so-called Tri-core architecture codenamed Tavor; Tri-core means it supports the ARMv5TE, ARMv6 and ARMv7 instruction sets.^[8] This new architecture was a significant leap from the old Xscale architecture. The PXA930 uses 65 nm technology^[9] while the PXA935 is build using the 45 nm process.

The PXA930 is used in the Blackberry Bold 9700.

PXA940

Little is known about the PXA940, although it is known to be ARM Cortex-A8 compliant. It is utilized in the Blackberry Torch 9800^{[10][11]} and is built using 45 nm technology.

PXA986/PXA988

After XScale and Sheeva, the PXA98x uses the third CPU core design, this time licensed directly from ARM, in form of dual core Cortex A9 application processors^[12] utilized by devices like Samsung Galaxy Tab 3 7.0^[13]

PXA1088

It is a quad core Cortex A7 application processor with Vivante GPU.^[14]

IXC

IXC1100

The IXC1100 processor features clock speeds at 266, 400, and 533 MHz, a 133 MHz bus, 32 KB of instruction cache, 32 KB of data cache, and 2 KB of mini-data cache. It is also designed for low power consumption, using 2.4 W at 533 MHz. The chip comes in the 35 mm PBGA package.

IOP

The IOP line of processors is designed to allow computers and storage devices to transfer data and increase performance by offloading I/O functionality from the main CPU of the device. The IOP3XX processors are based on the XScale architecture and designed to replace the older 80219 sd and i960 family of chips. There are ten different IOP processors currently available: IOP303, IOP310, IOP315, IOP321, IOP331, IOP332, IOP333, IOP341, IOP342 and IOP348. Clock speeds range from 100 MHz to 1.2 GHz. The processors also differ in PCI bus type, PCI bus speed, memory type, maximum memory allowable, and the number of processor cores.

IXP network processor

The XScale core is utilized in the second generation of Intel's IXP network processor line, while the first generation used StrongARM cores. The IXP network processor family ranges from solutions aimed at small/medium office network applications, IXP4XX, to high performance network processors such as the IXP2850, capable of sustaining up to OC-192 line rates. In IXP4XX devices the XScale core is used as both a control and data plane processor, providing both system control and data processing. The task of the XScale in the IXP2XXX devices is typically to provide control plane functionality only, with data processing performed by the microengines, examples of such control plane tasks include routing table updates, microengine control, memory management.

CE

In April 2007, Intel announced an XScale-based processor targeting consumer electronics markets, the Intel CE 2110.^[15]

Applications

XScale microprocessors can be found in products such as the popular RIM BlackBerry handheld, the Dell Axim family of Pocket PCs, most of the Zire, Treo and Tungsten Handheld lines by Palm, later versions of the Sharp Zaurus, the Motorola A780, the Acer n50, the Compaq iPaq 3900 series and many other PDAs. It is used as the main CPU in the Iyonix PC desktop computer running RISC OS, and the NSLU2 (Slug) running a form of Linux. The

XScale is also used in devices such as PVPs (Portable Video Players), PMCs (Portable Media Centres), including the Creative Zen Portable Media Player and Amazon Kindle E-Book reader, and industrial embedded systems. At the other end of the market, the XScale IOP33x Storage I/O processors are used in some Intel Xeon-based server platforms.

Sale of PXA processor line

On June 27, 2006, the sale of Intel's XScale PXA mobile processor assets was announced. Intel agreed to sell the XScale PXA business to Marvell Technology Group for an estimated \$600 million in cash and the assumption of unspecified liabilities. The move was intended to permit Intel to focus its resources on its core x86 and server businesses. Marvell holds a full Architecture License for ARM, allowing it to design chips to implement the ARM instruction set, not just license a processor core.

The acquisition was completed on November 9, 2006. Intel was expected to continue manufacturing XScale processors until Marvell secures other manufacturing facilities, and would continue manufacturing and selling the IXP and IOP processors, as they were not part of the deal.

The XScale effort at Intel was initiated by the purchase of the StrongARM division from Digital Equipment Corporation in 1998.^[16] Intel still holds an ARM license even after the sale of XScale; this license is at the architectural level.^[17]

References

- [1] Marvell buys Intel's handheld processor unit for \$600 million (<http://www.eet.com/news/latest/showArticle.jhtml?articleID=189601851>)
- [2] 3rd Generation Intel XScale(R) Microarchitecture Developer's Manual - <http://www.intel.com/design/intelxscale/316283.htm>
- [3] Marvell Introduces Next Generation Application Processors (<http://marvell.com/press/pressNewsDisplay.do?releaseID=680>)
- [4] Marvell Communications Processors product page (<http://www.marvell.com/products/cellular/cellular.jsp>)
- [5] Intel XScale PXA900 (Hermon) Application Processor with Modem Datasheet | CPUlist (http://pdadb.net/index.php?m=cpu&id=a900&c=intel_xscale_pxa900). PDADB.net (2012-02-25). Retrieved on 2013-08-02.
- [6] Marvell ARMADA 100 Processors product page (http://www.marvell.com/products/processors/applications/armada_100)
- [7] Google Vertalen (http://translate.google.nl/translate?js=y&prev=_t&hl=nl&ie=UTF-8&layout=1&cof=1&u=http://tweakers.net/reviews/1766/4/blackberry-pearl-3g-klein-en-kiwik-hardware-en-bouwkwiteit.html&sl=nl&tl=en). Translate.google.nl. Retrieved on 2013-08-02.
- [8] Marvell PXA935 (Tavor-P65) Application Processor with Modem Datasheet | CPUlist (http://pdadb.net/index.php?m=cpu&id=a935&c=marvell_pxa935). PDADB.net (2012-02-25). Retrieved on 2013-08-02.
- [9] Marvell PXA930 (Tavor-MG1) Application Processor with Modem Datasheet | CPUlist (http://pdadb.net/index.php?m=cpu&id=a930&c=marvell_pxa930). PDADB.net (2012-02-25). Retrieved on 2013-08-02.
- [10] Blackberry Torch 9800 - Teardown : TechInsights (<http://www.ubmtechinsights.com/reports-and-subscriptions/investigative-analysis/blackberry-torch-9800/teardown/>). Ubmtechinsights.com (2012-10-25). Retrieved on 2013-08-02.
- [11] http://www.ubmtechinsights.com/uploadedImages/Public_Website/Content_-_Primary/Investigative_Analysis/2010/Blackberry_Torch_9800/torch-front-web.jpg
- [12] Fingas, Jon. (2012-08-14) Marvell PXA988, PXA986 chips support 3G for China, the world without reinventing the wheel (or phone) (<http://www.engadget.com/2012/08/14/marvell-pxa988-and-pxa986-chips-support-3g-for-china-and-the-world/>). Engadget.com. Retrieved on 2013-08-02.
- [13] Samsung Galaxy Tab 3 Runs On A Marvell PXA986 Processor (<http://www.ubergizmo.com/2013/05/samsung-galaxy-tab-3-runs-on-a-marvel-pxa986-processor/>). Ubergizmo. Retrieved on 2013-08-02.
- [14] Gorman, Michael. (2013-02-19) Marvell announces PXA1088 quad-core SoC for globetrotting phones and tablets (<http://www.engadget.com/2013/02/19/marvell-pxa1088-quad-core-cortec-a7-soc/>). Engadget.com. Retrieved on 2013-08-02.
- [15] Intel System-On-A-Chip Media Processor Powers New Generation Of Consumer Electronics Devices (http://www.intel.com/pressroom/archive/releases/2007/20070416comp_a.htm)
- [16] http://www.reghardware.co.uk/2006/06/27/intel_sells_xscale/
- [17] AMD Jumps Into The ARM Server Business (<http://www.forbes.com/sites/rogerkay/2012/10/31/amd-jumps-into-the-arm-server-business/>). Forbes. Retrieved on 2013-08-02.

External links

- Intel XScale Technology Overview (<http://www.intel.com/design/intelxscale/>)
- IXP4XX Toolkits (http://www.intel.com/design/network/products/npfamily/ixp400_tpv.htm)
- Intel StrataFlash Memory (<http://www.intel.com/design/flcomp/prodbref/251890.htm>)
- Marvell PXA168 high-performance processor product brief (http://www.marvell.com/products/processors/applications/armada_100/armada_168/pxa_168_pb.pdf)
- Optimized Linux Code for Intel XScale Microarchitecture (<http://download.intel.com/design/iio/applnnts/30851701.pdf>)

Comparison of ARMv7-A cores

This is a table comparing microarchitectures which implement the ARMv7-A instruction set and mandatory or optional extensions of it, the last AArch32.

Table

This list is incomplete; you can help by expanding it ^[1].

Core	Decode width	Execution ports	Pipeline depth	Out-of-order execution	FPU	Pipelined VFP	FPU registers	NEON (SIMD)	Process technology	L0 cache	L1 cache	L2 cache	Core configurations	Speed per core (DMIPS/MHz)
ARM Cortex-A5	1		8	No	VFPv4 (optional)		16 × 64-bit	64-bit wide (optional)			4-64 KB / core		1, 2, 4	1.57
ARM Cortex-A7	2		8	No	VFPv4	Yes	16 × 64-bit	64-bit wide	40/28 nm		8-64 KB / core	up to 1 MB (optional)	1, 2, 4, 8	1.9
ARM Cortex-A8	2		13	No	VFPv3	No	32 × 64-bit	64-bit wide	65/55/45 nm		32 KB + 32 KB	256 or 512 (typical) KB	1	2.0
ARM Cortex-A9	2	3	8	Yes	VFPv3 (optional)	Yes	(16 or 32) × 64-bit	64-bit wide (optional)	65/45/40/32/28 nm		32 KB + 32 KB	1 MB	1, 2, 4	2.5
ARM Cortex-A12	3		11	Yes	VFPv4	Yes	32 × 64-bit	128-bit wide			32-64 KB + 32 KB	256 KB to 8 MB	1, 2, 4	3.0
ARM Cortex-A15	3	7	15/17-25	Yes	VFPv4	Yes	32 × 64-bit	128-bit wide	32/28 nm		32 KB + 32 KB per core	up to 4 MB per cluster, up to 8 MB per chip	2, 4, 8 (4×2)	3.5 to 4.01
ARM Cortex-A17	3		11+	Yes	VFPv4	Yes	32 × 64-bit	128-bit wide			32 KB + 32 KB per core	256 KB up to 8 MB	up to 4	
Qualcomm Scorpion	2		10	non-speculative ^[1]	VFPv3	Yes		128-bit wide	65/45 nm		32 KB + 32 KB	256 KB (single-core) 512 KB (dual-core)	1, 2	2.1
Qualcomm Krait ^[2]	3	7	11	Yes	VFPv4 ^[3]	Yes		128-bit wide	28 nm	4 KB + 4 KB direct mapped	16 KB + 16 KB 4-way set associative	1 MB 8-way set associative (dual-core)/2 MB (quad-core)	2, 4	3.3 (Krait) 3.1 (Krait 200) 3.4 (Krait 300) ^[4] 3.6 (Krait 400)

Apple Swift	3		12	Yes	VFPv4	Yes	32 × 64-bit	128-bit wide	32 nm		32 KB + 32 KB	1 MB	2	3.5
-------------	---	--	----	-----	-------	-----	-------------	--------------	-------	--	---------------	------	---	-----

References

- [1] http://rtcgroup.com/arm/2007/presentations/253%20-%20ARM_DevCon_2007_Snapdragon_FINAL_20071004.pdf
- [2] <http://www.anandtech.com/show/4940/qualcomm-new-snapdragon-s4-msm8960-krait-architecture>
- [3] <http://www.anandtech.com/show/5559/qualcomm-snapdragon-s4-krait-performance-preview-msm8960-adreno-225-benchmarks/2>
- [4] http://www.linleygroup.com/newsletters/newsletter_detail.php?num=4920

Image Sources, Licenses and Contributors

Image:ARM logo.svg *Source:* https://en.wikipedia.org/w/index.php?title=File:ARM_logo.svg *License:* Public Domain *Contributors:* User:SreeBot

Image:ARMSoCBlockDiagram.svg *Source:* <https://en.wikipedia.org/w/index.php?title=File:ARMSoCBlockDiagram.svg> *License:* Creative Commons Attribution-ShareAlike 3.0 Unported *Contributors:* en>User:Cburnett

File:Acorn-ARM-Evaluation-System.jpg *Source:* <https://en.wikipedia.org/w/index.php?title=File:Acorn-ARM-Evaluation-System.jpg> *License:* Creative Commons Attribution-ShareAlike 3.0 *Contributors:* Flibble

File:GPS ARM610 die.JPG *Source:* https://en.wikipedia.org/w/index.php?title=File:GPS_ARM610_die.JPG *License:* Creative Commons Attribution 3.0 *Contributors:* User:Birdman86

Image:STM32F103VGT6-HD.jpg *Source:* <https://en.wikipedia.org/w/index.php?title=File:STM32F103VGT6-HD.jpg> *License:* Creative Commons Attribution 3.0 *Contributors:* Dhx1, Ghouston

File:Quad-core Android "mini PC", with a microSD card next to it for a size comparison.jpg *Source:* [https://en.wikipedia.org/w/index.php?title=File:Quad-core_Android_\"mini_PC\"_with_a_microSD_card_next_to_it_for_a_size_comparison.jpg](https://en.wikipedia.org/w/index.php?title=File:Quad-core_Android_\) *License:* Creative Commons Attribution-ShareAlike 3.0 *Contributors:* Dsimic

File:Android 4.0.png *Source:* https://en.wikipedia.org/w/index.php?title=File:Android_4.0.png *License:* Creative Commons Attribution 2.5 *Contributors:* Android Open Source project

Image:Commons-logo.svg *Source:* <https://en.wikipedia.org/w/index.php?title=File:Commons-logo.svg> *License:* logo *Contributors:* Anomie

File:Diagrama ARM7.jpg *Source:* https://en.wikipedia.org/w/index.php?title=File:Diagrama_ARM7.jpg *License:* Public Domain *Contributors:* Erich Silvestre e Pedro Bachiega

Image:Mck glamor 320.jpg *Source:* https://en.wikipedia.org/w/index.php?title=File:Mck_glamor_320.jpg *License:* Creative Commons Attribution 3.0 *Contributors:* MakeThings LLC

Image:Esom270 computer on module.jpg *Source:* https://en.wikipedia.org/w/index.php?title=File:Esom270_computer_on_module.jpg *License:* Public Domain *Contributors:* Lakshmi

Image:Toradex Colibri XScale Monahans PXA290.jpg *Source:* https://en.wikipedia.org/w/index.php?title=File:Toradex_Colibri_XScale_Monahans_PXA290.jpg *License:* GNU Free Documentation License *Contributors:* Toradex

File:SOM168.jpg *Source:* <https://en.wikipedia.org/w/index.php?title=File:SOM168.jpg> *License:* Creative Commons Attribution-ShareAlike 3.0 *Contributors:* User:Nandcon

License

Creative Commons Attribution-Share Alike 3.0
[//creativecommons.org/licenses/by-sa/3.0/](https://creativecommons.org/licenses/by-sa/3.0/)
