ARM946E-S (Rev1)

System-on-Chip DSP enhanced processor

Product Overview

Applications

- Embedded applications running an RTOS
- Mass storage
  - HDD & DVD
- Speech coders
- Networking applications
  - G.723.1 for voice-over IP
- Automotive control
  - Cruise control, ABS, etc.
  - Hands-free interfaces
- Modems and soft-modems
- Audio decoding
  - Dolby AC3 digital
  - MPEG MP3 audio
- Speech recognition and synthesis.

Benefits

- System-on-Chip ready, allowing rapid integration with short time-to-market
- ARM946E-S provides a single chip DSP and microcontroller solution
- Reduced die size and chip complexity
- Fast interrupt response
- Reduced programming complexity
  - No need to partition DSP and control code
- No duplication in on-chip memory system, busing, debug, and trace resources.

The ARM946E-S™

The ARM946E-S™ is a synthesizable macrocell combining an ARM9E-S™ processor core with instruction and data caches, tightly-coupled instruction and data SRAM memory with protection units, write buffer, and an AMBA™ (Advanced Microprocessor Bus Architecture) AHB (Advanced High-performance Bus) interface. It is a member of the ARM9E-S Thumb® family of high-performance 32-bit *System-on-Chip* (SoC) processors, and it is well suited to a wide range of embedded applications. The size of the instruction and data cache, and the instruction and data SRAM is individually configurable allowing you to tailor hardware to the embedded application. The ARM946E-S provides a complete high-performance processor solution, offering considerable savings in chip complexity and area, chip system design, power consumption, and time-to-market.

Compatible with ARM7™ and StrongARM®

The ARM946E-S™ processor is backwards compatible with the ARM7 Thumb Family and the StrongARM processor families, giving designers software-compatible processors with a range of price/performance points from 60 MIPS to 400 MIPS. Support for the ARM® architecture today includes:

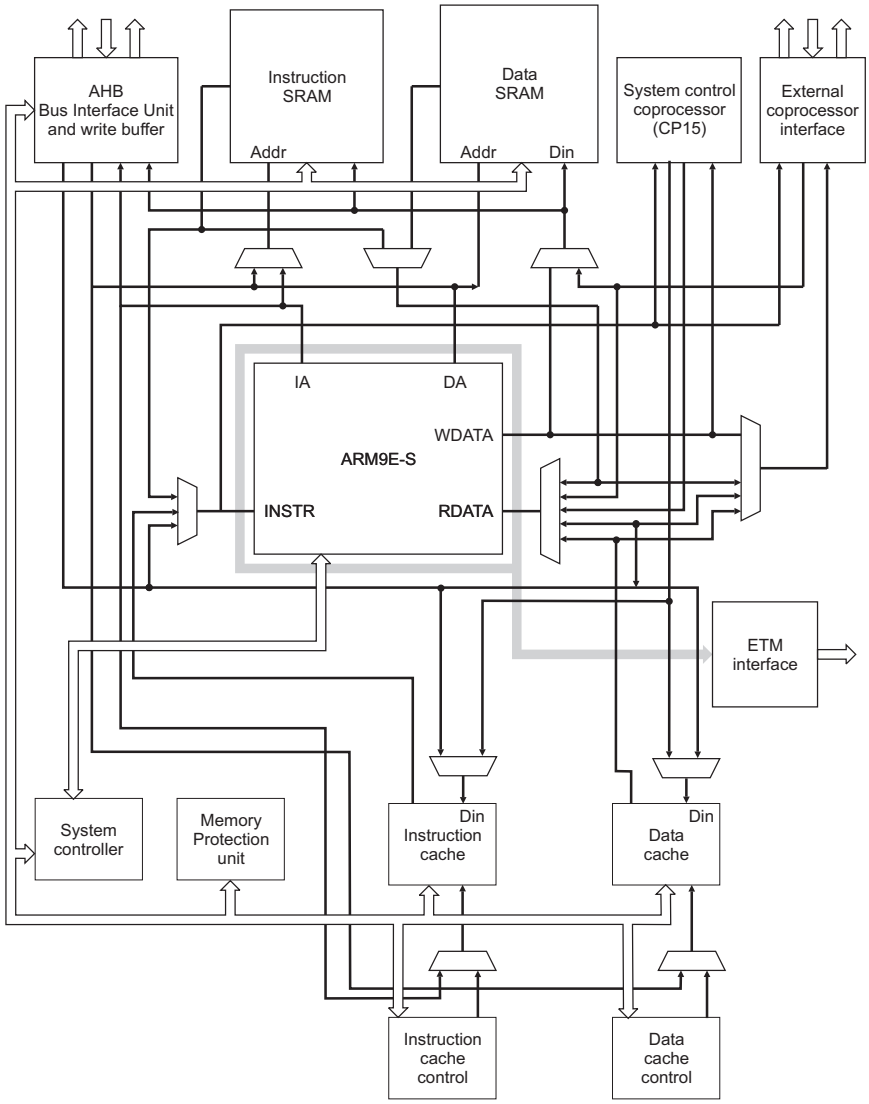
- EPOC, JavaOS
- Linux operating systems & WindowsCE
- 40-plus Real Time Operating Systems
- Co-simulation tools from leading EDA vendors
- Industry supported third-party software development tools.

DSP enhancements

The ARM946E-S processor core offers the full advantage of the enhanced DSP capability of the ARM9E-S processor core. The ARM9E-S processor core executes the ARM5vTE instruction set which includes new multiplier and saturating arithmetic functions. Multiply instructions are processed faster using a single-cycle 32x16 implementation. There are 32x16 and 16x16 multiply instructions, and the pipeline allows one multiply to start each cycle. New saturating arithmetic improves efficiency by automatically selecting saturated behavior during execution. Saturating arithmetic is used to set limits on signal processing calculations to minimize the effect of noise or signal errors. All of these instructions are beneficial for algorithms such as those which implement GSM protocols, *Fast Fourier Transforms* (FFT) and state space servo control for HDDs.

ARM946E-S™ processor

The ARM946E-S™ processor uses the ARM9E-S™ synthesizable macrocell. This macrocell combines the ARM9™ processor core with the powerful features and instruction set extensions which assist DSP applications. The architecture of the processor core or integer unit, is described in more detail on page 9.



System controller

The system controller oversees the interaction between the Instruction Cache, Instruction RAM, Data Cache, Data RAM, and the Bus Interface Unit. It controls internal arbitration between the blocks and stalls appropriate blocks when required.
The system controller arbitrates between instruction and data access to schedule single or simultaneous requests to the cache controllers and the Bus Interface Unit. The system controller receives acknowledgement from each resource to allow execution to continue.

Control coprocessor (CP15)

The CP15 allows configuration of both the caches and the tightly coupled SRAMs, the write buffer, and other ARM946E-S functions.
Several registers within CP15 are available for program control, providing access to features such as:

- big or little-endian operation
- low power state
- memory partitioning and protection
- full memory BIST (Built-in Self Test).

Protection unit

The protection unit allows memory to be partitioned and individual attributes set for each protection region. Both the instruction and data address space can be divided into eight regions of variable size.
The protection attributes for each region can specify the properties of cachable, bufferable, user access, supervisor access, etc. The protection unit is programmed from the CP15 registers.

Caches

Two caches are implemented, one for instructions, the other for data, both with an eight-word line size. Each cache is constructed from SRAM. The caches connect to the ARM9E-S processor core through 32-bit buses, to allow one instruction to be passed into the instruction prefetch unit every cycle, and to allow load and store multiple instructions to transfer one register every cycle.

Cache lock-down

Cache lock-down is provided to allow critical code sequences to be locked into the cache to ensure predictability

for real-time code. The cache replacement algorithm can be selected by the operating system as either pseudo random or round-robin. Both caches are four-way set-associative. Lock down operates on a per-set basis

Cache features

ARM946E-S instruction and data cache sizes can be selected independently from the following range of cache sizes:

- 0KB
- 4KB
- 8KB
- 16KB
- 32KB
- 64KB
- 128KB
- 256KB
- 512KB
- 1MB.

The caches are four-way set associative, with a cache line length of eight words. Cache entries are allocated on a read miss basis.

Write buffer

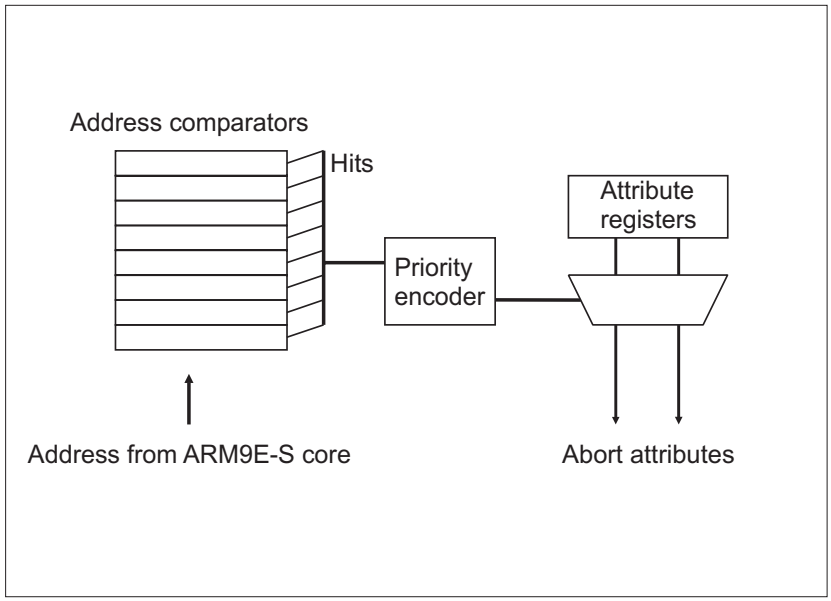
ARM946E-S™ also incorporates a 16-entry write buffer, to avoid stalling the processor when writes to external memory are performed.

Tightly Coupled Memory

ARM946E-S supports SRAM for the *Tightly Coupled Memory* (TCM).

The minimum size, when TCM is present, is 4KB incrementing in powers of 2 (e.g. 8KB, 16KB) up to 1MB. Therefore, the instruction and data SRAM memories can have unique sizes from 0KB to 1MB.

The memory is capable of returning data to the ARM9E-S core in a single cycle.



Protection unit block Diagram

Registers

The ARM9E-S™ processor core consists of a 32-bit datapath and associated control logic. That datapath contains 31 general-purpose registers, coupled to a full shifter, Arithmetic Logic Unit, and multiplier. At any one time 16 registers are visible to the user. The remainder are synonyms used to speed up exception processing. Register 15 is the *Program Counter* (PC) and can be used in all instructions to reference data relative to the current instruction. R14 holds the return address after a subroutine call. R13 is used (by software convention) as a stack pointer.

Modes and exception handling

All exceptions have banked registers for R14 and R13. After an exception, R14 holds the return address for exception processing. This address is used both to return after the exception is processed and to address the instruction that caused the exception. R13 is banked across exception modes to provide each exception handler with a private stack pointer. The fast interrupt mode also banks registers eight to 12 so that interrupt processing can begin without the need to save or restore these registers. A seventh processing mode, System mode, does not have any banked registers. It uses the User mode registers. System mode runs tasks that require a privileged processor mode and allows them to invoke all classes of exceptions.

Status registers

All other processor states are held in status registers. The current operating processor status is in the *Current Program Status Register* (CPSR). The CPSR holds:

- four ALU flags (Negative, Zero, Carry, and Overflow),
- two interrupt disable bits (one for each type of interrupt),
- a bit to indicate ARM or Thumb execution,
- and five bits to encode the current processor mode.

All five exception modes also have a *Saved Program Status Register* (SPSR) which holds the CPSR of the task immediately before the exception occurred.

Exception types

ARM9E-S supports five types of exception, and a privileged processing mode for each type. The types of exceptions are:

- fast interrupt (FIQ)
- normal interrupt (IRQ)
- memory aborts (used to implement memory protection or virtual memory)
- attempted execution of an undefined instruction
- software interrupts (SWIs).

Conditional execution

All ARM instructions (with the exception of BLX) are conditionally executed. Instructions optionally update the four condition code flags (Negative, Zero, Carry, and Overflow) according to their result. Subsequent instructions are conditionally executed according to the status of flags. Fifteen conditions are implemented.

Four classes of instructions

The ARM and Thumb instruction sets can be divided into four broad classes of instruction:

- data processing instructions
- load and store instructions
- branch instructions
- coprocessor instructions.

Data processing

The data processing instructions operate on data held in general purpose registers. Of the two source operands, one is always a register. The other has two basic forms:

- an immediate value
- a register value optionally shifted.

If the operand is a shifted register the shift amount might have an immediate value or the value of another register. Four types of shift can be specified. Most data processing instructions can perform a shift followed by a logical or arithmetic operation. Multiply instructions come in two classes:

- normal - 32-bit result
- long - 32-bit result variants.

Both types of multiply instruction can optionally perform an accumulate operation.

Load and store

The second class of instruction is load and store instructions. These instructions come in two main types:

- load or store the value of a single register or register pair
- load and store multiple register values.

Load and store single register instructions can transfer a 32-bit word, a 16-bit halfword and an

eight-bit byte between memory and a register. Byte and halfword loads may be automatically zero extended or sign extended as they are loaded. A preload ‘hint’ instruction is available to help minimize memory system latency. Swap instructions perform an atomic load and store as a synchronization primitive.

Addressing modes

Load and store instructions have three primary addressing modes

- offset
- pre-indexed
- post-indexed.

They are formed by adding or subtracting an immediate or register-based offset to or from a base register. Register-based offsets can also be scaled with shift operations. Pre-indexed and post-indexed addressing modes update the base register with the base plus offset calculation. As the PC is a general purpose register, a 32-bit value can be loaded directly into the PC to perform a jump to any address in the 4GB memory space.

Block transfers

Load and store multiple instructions perform a block transfer of any number of the general purpose registers to or from memory. Four addressing modes are provided:

- pre-increment addressing
- post-increment addressing
- pre-decrement addressing
- post-decrement addressing.

The base address is specified by a register value (which can be optionally updated after the transfer). As the subroutine return address and the PC values are in general purpose registers, very efficient subroutine calls can be constructed.

Branch

As well as allowing any data processing or load instruction to change control flow (by writing the PC) a standard branch instruction is provided with 24-bit signed offset, allowing forward and backward branches of up to 32MB.

Branch with Link

There is a Branch with Link (BL) which allows efficient subroutine calls. BL preserves the address of the instruction after the branch in R14 (the Link Register or LR). This allows a move instruction to put the LR in to the PC and return to the instruction after the branch.

The third type of branch (BX and BLX) switches between ARM and Thumb instruction sets optionally with the return address preserving link option.

Coprocessor

There are three types of coprocessor instructions:

- coprocessor data processing instructions are used to invoke a coprocessor specific internal operation.
- coprocessor register transfer instructions allow a coprocessor value (word or double word) to be transferred to or from an ARM register (or register pair).
- coprocessor data transfer instructions transfer coprocessor data to or from memory, where the ARM calculates the address of the transfer.

The V5TE ARM Instruction Set including DSP extensions

Mnemonic	Operation	Mnemonic	Operation
MOV	Move	MVN	Move Not
QADD	Saturated add	QDADD	Saturated add with double
QSUB	Saturated subtract	QDSUB	Saturated subtract with double
RSB	Reverse Subtract	RSC	Reverse Subtract with Carry
CMP	Compare	CMN	Compare Negated
TST	Test	TEQ	Test Equivalence
AND	Logical AND	BIC	Bit Clear
EOR	Logical Exclusive OR	ORR	Logical (inclusive) OR
MUL	Multiply	MLA	Multiply Accumulate
SMULL	Sign Long Multiply	SMLAL	Signed Long Multiply Accumulate
UMULL	Unsigned Long Multiply	UMLAL	Unsigned Long Multiply Accumulate
CLZ	Count Leading Zeroes	BKPT	Breakpoint
MRS	Move From Status Register	MSR	Move to Status Register
B	Branch		
BL	Branch and Link	BLX	Branch and Link and Exchange
BX	Branch and Exchange	SWI	Software Interrupt
LDR	Load Word	STR	Store Word
LDRH	Load Halfword	STRH	Store Halfword
LDRB	Load Byte	STRB	Store Byte
LDRSH	Load Signed Halfword	LDRSB	Load Signed Byte
LDMIA	Load Multiple	STMIA	Store Multiple
SWP	Swap Word	SWPB	Swap Byte
CDP	Coprocessor Data Processing		
MRC	Move From Coprocessor	MCR	Move to Coprocessor
LDC	Load To Coprocessor	STC	Store From Coprocessor
LDRD	Load Double Word	PLD	Soft preload
STRD	Store Double Word	MRRC	Move Double Word from Coprocessor
MCRR	Move Double Word to Coprocessor		

The Thumb Instruction Set

Mnemonic	Operation	Mnemonic	Operation
MOV	Move	MVN	Move Not
ADD	Add	ADC	Add with Carry
SUB	Subtract	SBC	Subtract with Carry
RSB	Reverse Subtract	RSC	Reverse Subtract with Carry
CMP	Compare	CMN	Compare Negated
TST	Test	NEG	Negate
AND	Logical AND	BIC	Bit Clear
EOR	Logical Exclusive OR	ORR	Logical (inclusive) OR
LSL	Logical Shift Left	LSR	Logical Shift Right
ASR	Arithmetic Shift Right	ROR	Rotate Right
MUL	Multiply	BKPT	Breakpoint
B	Unconditional Branch	Bcc	Conditional Branch
BL	Branch and Link	BLX	Branch and Link and Exchange
BX	Branch and Exchange	SWI	Software Interrupt
LDR	Load Word	STR	Store Word
LDRH	Load Halfword	STRH	Store Halfword
LDRB	Load Byte	STRB	Store Byte
LDRSH	Load Signed Halfword	LDRSB	Load Signed Byte
LDMIA	Load Multiple	STMIA	Store Multiple
PUSH	Push Registers to stack	POP	Pop Registers from stack

Modes and Registers

User and System Mode	Supervisor Mode	Abort Mode	Undefined Mode	Interrupt Mode	Fast Interrupt Mode
R0	R0	R0	R0	R0	R0
R1	R1	R1	R1	R1	R1
R2	R2	R2	R2	R2	R2
R3	R3	R3	R3	R3	R3
R4	R4	R4	R4	R4	R4
R5	R5	R5	R5	R5	R5
R6	R6	R6	R6	R6	R6
R7	R7	R7	R7	R7	R7
R8	R8	R8	R8	R8	R8_FIQ
R9	R9	R9	R9	R9	R9_FIQ
R10	R10	R10	R10	R10	R10_FIQ
R11	R11	R11	R11	R11	R11_FIQ
R12	R12	R12	R12	R12	R12_FIQ
R13	R13_SVC	R13_ABORT	R13_UNDEF	R13_IRQ	R13_FIQ
R14	R14_SVC	R14_ABORT	R14_UNDEF	R14_IRQ	R14_FIQ
PC	PC	PC	PC	PC	PC

CPSR	CPSR	CPSR	CPSR	CPSR	CPSR
	SPSR_SVC	SPSR_ABORT	SPSR_UNDEF	SPSR_IRQ	SPSR_FIQ

Mode-specific banked registers

ARM9E-S™ processor core

The ARM9E-S processor core is an implementation of the ARM *Instruction Set Architecture* (ISA) Version 5TE with enhanced DSP performance. ARMv5TE is a superset of the ARMv4 ISA implemented by the StrongARM™ processors and the ARMv4T ISA implemented by the ARM7™ Thumb and ARM9™ Thumb Family processors.

Performance and code density

ARM9E-S executes two instruction sets:

- 32-bit ARM instruction set
- 16-bit Thumb instruction set.

The ARM instruction set allows a program to achieve maximum performance with the minimum number of instructions.

The majority of ARM9E-S instructions are executed in a single cycle. These are shown in the ARM9E-S instruction execution timing table on page 10. The table shows for each instruction class:

- Issues Cycles:
  Number of cycles in execute stage.
- Result Delay:
  Number of cycles after execute until data available.

The simpler Thumb instruction set offers much increased code density for code that does not require maximum performance. Code can switch between the ARM and Thumb instruction sets on any procedure call.

ARM9E-S integer pipeline stages

The integer pipeline consists of five stages to maximize instruction throughput on ARM9E-S™:

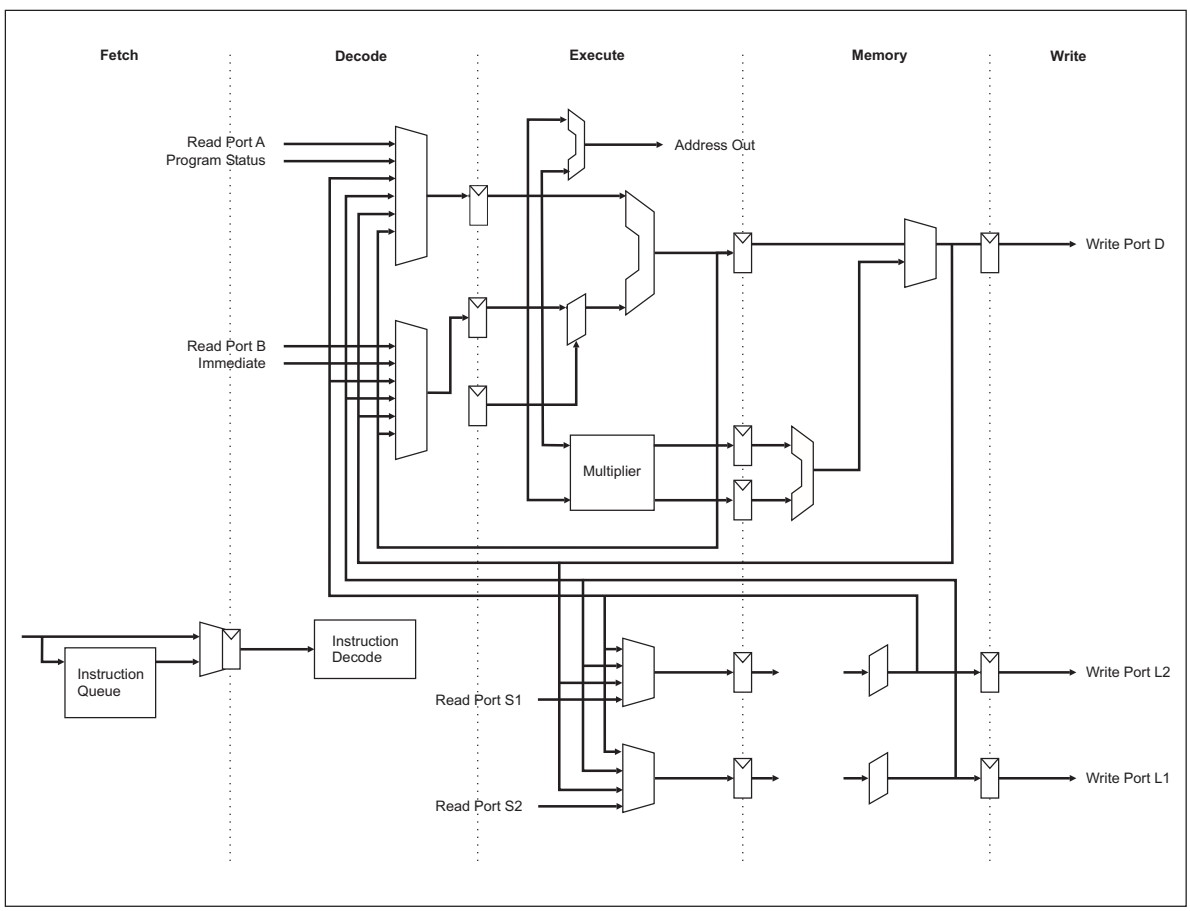
F: Instruction Fetch

D: Instruction Decode and Register Read

E: Execute Shift and ALU, or Address Calculate, or Multiply

M: Memory Access and Multiply

W: Register Write



ARM9E-S integer pipeline

Pipelining

By overlapping the various stages of execution, ARM9E-S maximizes the clock rate achievable to execute each instruction. It delivers a throughput approaching one instruction per cycle.

32-bit data buses

ARM9E-S provides 32-bit data buses between the processor core and the instruction and data caches, and between coprocessors and the processor core. This allows ARM9E-S to achieve very high performance on many code sequences, especially those that require data movement in parallel with data processing.

Coprocessors and pipelines

The ARM9E-S coprocessor interface allows full independent processing in the ARM execution pipeline and supports up to16 independent coprocessors.

Debug features

The ARM9E-S processor core also incorporates a sophisticated debug unit to allow both software tasks and external debug hardware to perform hardware and software breakpoint, single stepping, register and memory access. This functionality is made available to software as a coprocessor and is accessible from hardware via the JTAG port. Full-speed, real-time execution of the processor is maintained until a breakpoint is hit. At this point control is passed either to a software handler or to JTAG control.

Table 1: ARM9E-S instruction execution timing

Instruction class	Issue Cycles	Result delay
Branch	3	NA
Condition failed	1	NA
ALU instruction	1	0
ALU instruction with register specified shift	2	0
MOV PC, Rx	3	NA
ALU instruction dest = PC	3	0
MUL (flags modified)	4 to 5	0
MUL (flags unmodified)	1 to 3	1
MSR (flags only)	1	0
MSR (mode change)	3	0
MRS	1	0
LDR (loaded value)	1	1
STR	1	NA
LDM	Number of words	0 (Not last register in list)
LDM	Number of words	1 (Last register in list)
STM	1	NA
SWP	2	1
CDP	1	NA
MRC	1	1
MCR	1	NA
LDC	Number of words	NA
STC	Number of words	NA
LDRD	2	0 (first word)
	2	1 (second word)
STRD	2	NA
MRRC	2	0 (first word)
	2	1 (second word)
MCRR	2	NA
PLD	1	NA

System Design and Third Party Support

Embedded Trace Macrocell

The *Embedded Trace Macrocell* (ETM) monitors the ARM946E-S processor core address, data, and status signals and generates a compressed trace data stream of processor activity. The ETM provides non-intrusive real-time instruction trace by broadcasting compressed instruction addresses at branches. Data tracing is supported by broadcasting load and store operations. The unique triggering and data capabilities of the ETM means that complex events can be easily traced.

AMBA Bus Architecture

The ARM9 Thumb Family processors are designed for use with the AMBA multi-master on-chip bus architecture. AMBA includes an *Advanced High-performance Bus* (AHB) connecting processors and high-bandwidth peripherals and memory interfaces, and a low-power peripheral bus allowing a large number of low-bandwidth peripherals. The AHB is re-used to allow efficient production test of the ARM946E-S processor macrocell.

The ARM946E-S AHB implementation provides a 32-bit address bus and a 32-bit data bus for high-bandwidth data transfers made possible by on-chip memory and modern SDRAM and RAMBUS memories.

Everything you need

ARM provides a wide range of products and services to support its processor families, including software development tools, development boards, models, applications software, training, and consulting services.

The ARM Architecture today enjoys broad third party support. The ARM9 Thumb Family processors' strong software compatibility with existing ARM families ensures that its users benefit immediately from existing support. ARM is working with its software, EDA, and semiconductor partners to extend this support to use new ARM9 Family features.

Current support

Support for the ARM Architecture today includes:

- Software toolkits available from ARM - *ARM Developer Suite* (ADS), Cygnus/GNU, Microtec, Greenhills, JavaSoft, MetaWare, and Windriver allowing software development in C, C++, Java, FORTRAN, Pascal, Ada, and assembler.
- *ARM Developer Suite* (ADS) includes:
  - Integrated development environment
  - C, C++, assembler, simulators and windowing source-level debugger
  - Available on Windows95, WindowsNT, and Unix.
  - ARM Multi-ICE™ JTAG interface
  - Allows EmbeddedICE software debug of ARM processor systems
  - integrates with the ADS.
  - ARMulator instruction-accurate software simulator.
- ARM and third party Development boards.
- Application software components: DSP, Speech and image compression, software modems, Chinese character input, network protocols, for example.
- Design Sign-off Models provide quality ASIC- simulation.
- 40+ Real Time Operating Systems including Windriver VxWorks, Sun Microsystems Chorus and JavaOS, Microtec VRTX, JMI, Embedded SystemProducts RTXC, and Integrated Systems pSOS.
- Major OS including Microsoft WindowsCE, PSION EPOC, NetBSD and Linux UNIX, Geoworks.
- Hardware/software co-simulation tools from leading EDA Vendors.

For more information, see www.arm.com

Contacting ARM

America

ARM INC.
750 University Avenue
Suite 150,
Los Gatos CA 95032
USA

Tel: +1-408-579-2209
Fax:+1-408-399-8854
Email: info@arm.com

Austin Design Center
ARM
1250 Capital of Texas Highway
Building3, Suite 560
Austin
Texas 78746
USA

Tel: +1-512-327-9249
Fax:+1-512-314-1078
Email: info@arm.com

Seattle
ARM
10900 N.E. 8th Street
Suite 920
Bellevue
Washington 98004
USA

Tel: +1-425-688-3061
Fax:+1-42-454-4383
Email: info@arm.com

Boston
ARM
300 West Main St., Suite 215
Northborough
MA 01532
USA

Tel: +1-508-351-1670
Fax:+1-508-351-1668
Email: info@arm.com

England
ARM Ltd.
110 Fulbourn Road
Cherry Hinton
Cambridgeshire
CB1 9Nj
England

Tel: +44 1223 400400
Fax:+44 1223 400410
Email: info@arm.com

France
ARM France
12, Avenue des Prés
BL 204 Montigny le Bretonneux
78059 Saint Quentin en Yvelines
Cedex
Paris
France

Tel: +33 1 30 79 05 10
Fax: +33 1 30 79 05 11
Email: info@arm.com

Germany
ARM Ltd.
Otto Hahn Str. 13B
85521 Ottobrunn - Riemerling
Munich
Germany

Tel: +49-89-608-75545
Fax:+49-89-608-75599
Email: info@arm.com

Japan
ARM K.K.
Plustaria Building 4F
3-1-4 Shin-Yokohama
Kohoku-ku,
Yokohama-shi
Kanagawa 222-0033

Tel: +81 45 477 5260
Fax: +81 45 477 5261
Email: info-armkk@arm.com

Korea
ARM
Room #1115, Hyundai Building
9-4, Soonae-Dong, Boondang-Ku
Sungnam, Kyunggi-Do
Korea 463-020

Tel: +82-342 712 8234
Fax: +82 432 713 8225
Email: info@arm.com