The ARM Processor Architecture

Seminar: Multimedia
Vorname: Ioannis
Name: Skazikis
Matr-Nr.: 637868
E-Mail: Ioannis.Skazikis@MNI-Fh-Giessen.De

Table of contents:

1. Some words about ARM:

- ARM designs microprocessor technology that lies at the heart of advanced digital products, from mobile phones and digital cameras to games consoles and automotive systems, and is leading intellectual property (IP) provider of high-performance, low-cost, power-efficient RISC processors, peripherals, and system-on-chip (SoC) designs through involvement with organizations such as the Virtual Socket Interface Alliance (VSIA) and Virtual Component Exchange (VCX). ARM also offers design and software consulting services.

  ARM's architecture is compatible with all four major platform operating systems: Symbian OS, Palm OS, Windows CE, and Linux. As for software, ARM also works closely with with its partners to provide optimized solutions for existing market segments.
  These benefits are making the ARM company a complete solution provider.

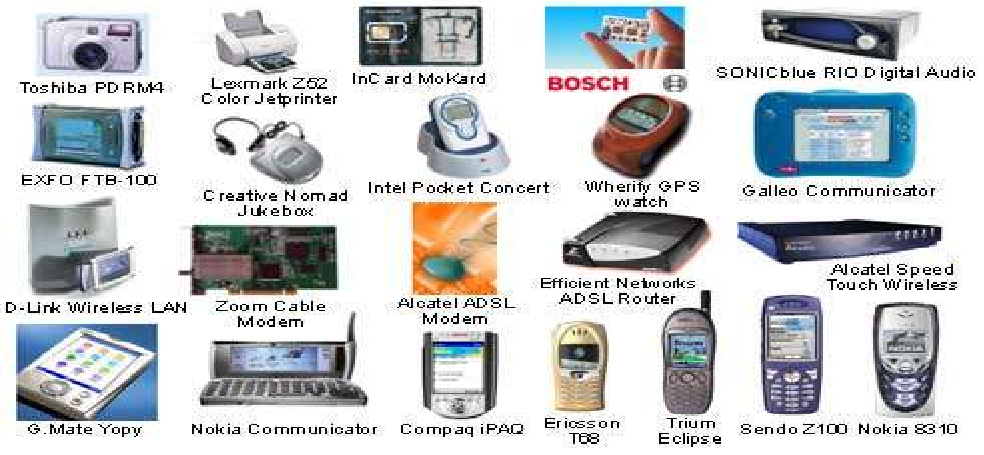


Figure 01: Some products that currently use ARM technology.

- With over forty partners licensed to use its architecture, ARM enables original equipment manufacturers (OEM) to realize an accelerated time-to-market through complete product offerings, such as PrimeCell Peripherals, embedded software IP, development tools, training, and support.

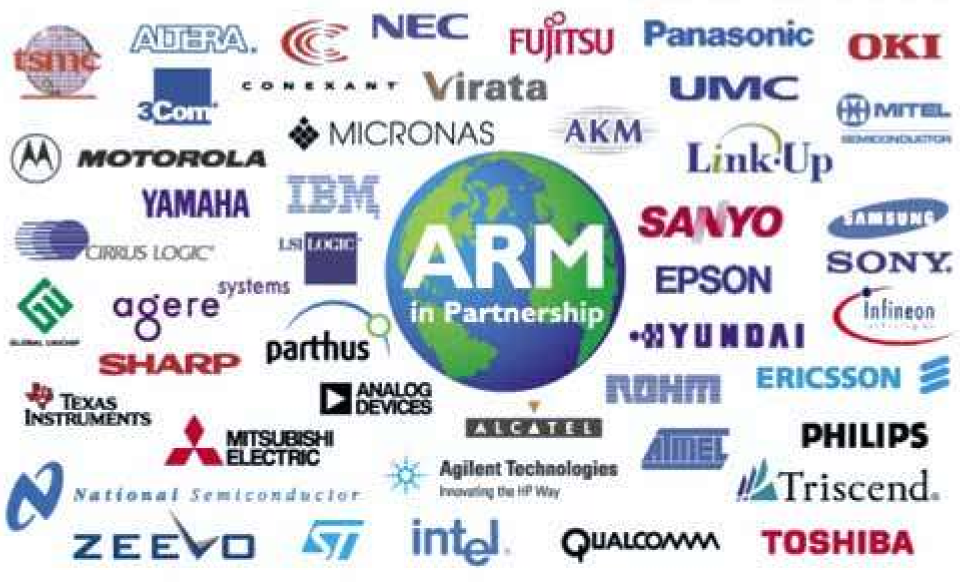


Figure 02: ARM's Partnership Companies

- The company offers a complete solution that is essential to the manufacturing process. Although ARM does not manufacture processors itself, ARM licenses its cores to semi-conductor manufacturers to be integrated into ASIC standards and then the company in using test chips manufactured by its partners to measure and validate the functionality of the core.
  ARM is able to accelerate OEM time-to-market by capitalizing on its architecture. By providing the IP and supporting services, customers can gain a jump on their design cycle and obtain a competitive edge in their targeted market segment. At that point, the architecture is portable to further product generations or applications as all code creation is directly compatible with any future architecture produced by ARM.

- ARM's Global Technology Partner Network is the largest in the industry, spanning from semiconductor manufacturers to distributors. ARM has worked diligently to ensure that the partnerships provide proven solutions in real-time operating systems (RTOS), EDA tools, development systems, applications software, and design consulting, all built around the ARM architecture.

Figure 03: ARM Worldwide Network

- The ARM company is working to establish standards, not just within the company, but across the industry by taking advantage of leadership opportunities in the creation of standards.

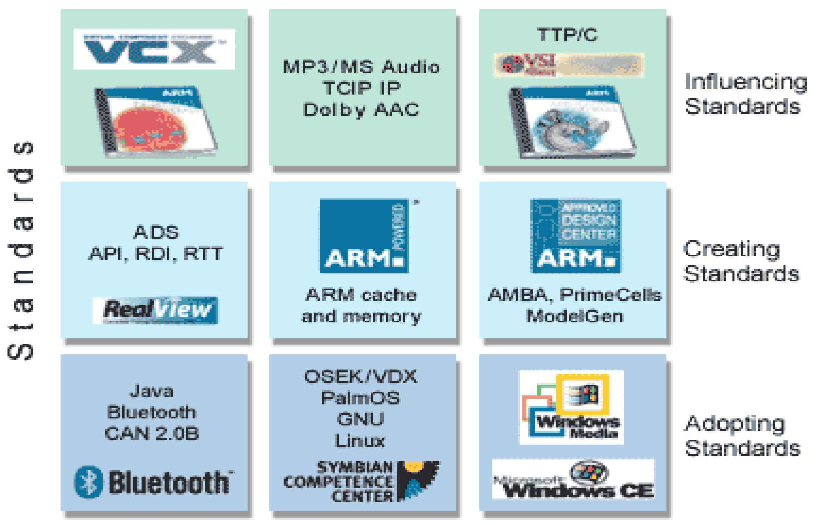


Figure 04: ARM's Worldwide Standarts

- This block diagram describes the ARM solution. The company recognizes that it cannot just present hardened macros and synthesizable CPUs to the industry, but it must also provide the ASIC infrastructure in the form of AMBA, the PrimeCell Peripherals, and models and modeling tools for the cores. There is also the need for ARM to pursue ports for RTOSs, develop debug hardware and software development tools, and, of course, embedded software for "off-the-shelf" integration. ARM combines all these futures together with support and training, to accelerate the design cycle and favour a successful product.

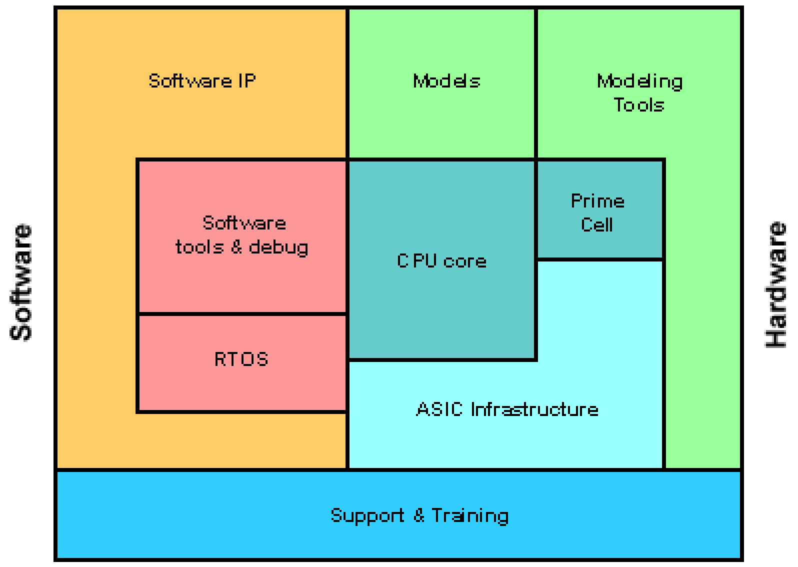


Figure 05: ARM's solution

- Sumury :

  ARM is the industry standard embedded microprocessor architecture, and is a leader in low-power high performance cores. ARM also has a large partner network supporting the entire design and development cycle. ARM is a full-solutions provider, supporting a broad range of applications.

2. Introduction of the ARM's Core Families and their benefits

2.1 Overview of ARM's current families of main cores:

The ARM7 and ARM9 families have contributed to ARM's success. Each core family has several "children" that incorporate many different value-added features and combinations. Essentially, there are four main families available now for license: ARM7, ARM9, ARM9E-S, and ARM10.
The ARM7 family features hardened and synthesizable macrocells with variants that incorporate cache with either a memory protection unit (MPU) or memory management unit (MMU). Other features include real-time debug (RTD) and real-time trace (RTT) technology.
The ARM9 family consists of hardened macrocells with variants also including cache with an MPU or MMU, as well as the RTD and the RTT. Although the ARM9E-S family was released under a different architecture version, ARMv5TE, the fundamental design of the core is based on the ARM9TDMI family. The "E" identifies that the family is a DSP-enhanced architecture and the "S" identifies that the family is synthesizable.
The ARM10 family is the highest performance family to date and will also embody the "E" extensions that were developed for the ARM9E-S family.
Finally, the StrongARM and XScale families are ARM compliant architectures available from Intel.

2.2 The Evolution of the ARM architecture:

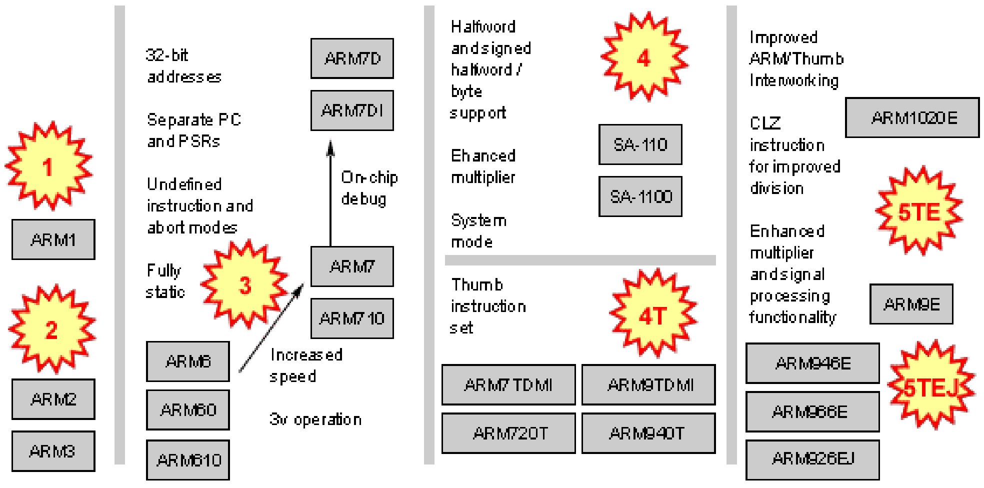


Figure 06: Evolution of the ARM Architecture

Architecture V1 was implemented only in the ARM1 CPU and was not utilized in a commercial product. Architecture V2 was the basis for the first shipped processors. These two architectures were developed by Acorn Computers before ARM became a company in 1990.
After that introduced ARM the Architecture V3, which included many changes over its predecessors. These changes resulted in an extremely small and power-efficient processor suitable for embedded systems.
Architecture V4, co-developed by ARM and Digital Electronics Corporation, resulted in the Strong ARM series of processors. These processors are very performance-centric and do not include the on-chip debug extensions. This architecture was further developed to include the Thumb 16-bit instruction set architecture enabling a 32-bit processor to utilize a 16-bit system. Today, ARM only licenses cores based on Architecture V4T or above.
The latest architectures, version 5TE and 5TEJ, embody added instructions for DSP applications and the Jazelle-Java extensions, respectively. Currently, the ARM9E and 10E family of processors are the only implementations of these architectures. Details on these architectures and cores will be provided later in the course.

2.3 Development Value:

From a development standpoint, ARM cores offer the advantage of a fully 32-bit processor designed specifically for embedded applications. An important feature is the embedded core debug facilities, which reduce the debugging stage of development. In some cases, this can be two-thirds of the overall development cycle.
Architecture compatibility allows code re-use and results in reduced design time. This in turn leads to reduced system cost, by eliminating investment in a second set of development tools to write code for a new processor architecture. The modular approach of the advanced micro-controller bus architecture, (AMBA), enables design reuse. This lowers the complexity of system on-chip (SoC) designs and reduces future design costs.
ARM and third parties offer the developer proven compiler technology and debug solutions. Multiple RTOSs and silicon sources mean that the developer will not need to change the preferred vendor in order to migrate to this architecture.

2.4 Redusing System Costs:

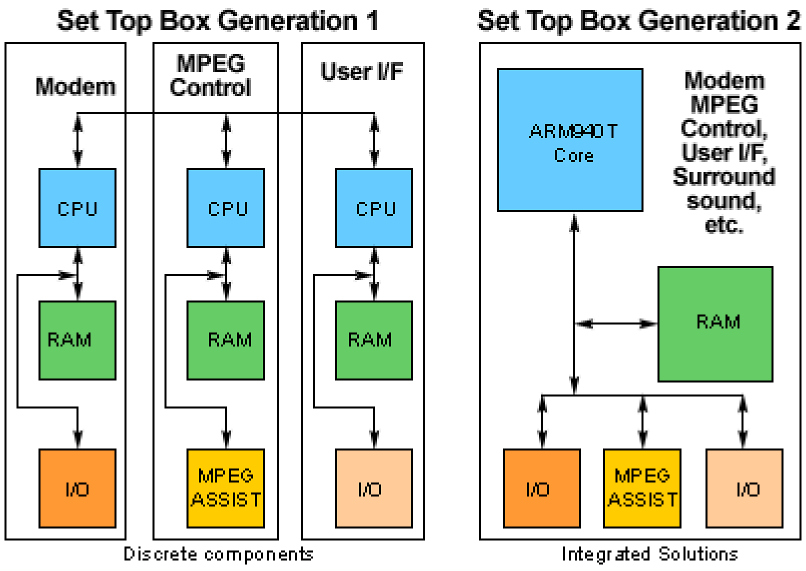


Figure 07: Example of Redusing Sytems Costs

Figur's 07 diagram shows the advantages of combining the functions performed by separate CPUs into a single, high-performance System-on-Chip based on an ARM processor. In the integrated solution, there is no duplication of memory, memory controllers, buses, and pins. Savings can be very high if off-chip memory subsystems can be replaced by a single memory system using commodity DRAM or SDRAM.

2.5 The ARM product roadmap:

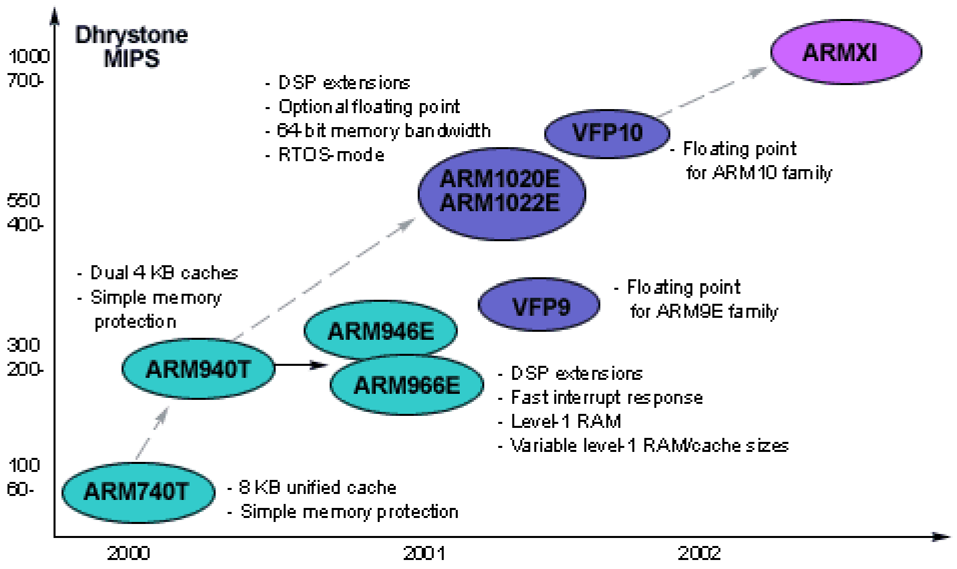


Figure 08: ARM's Product Roadmap

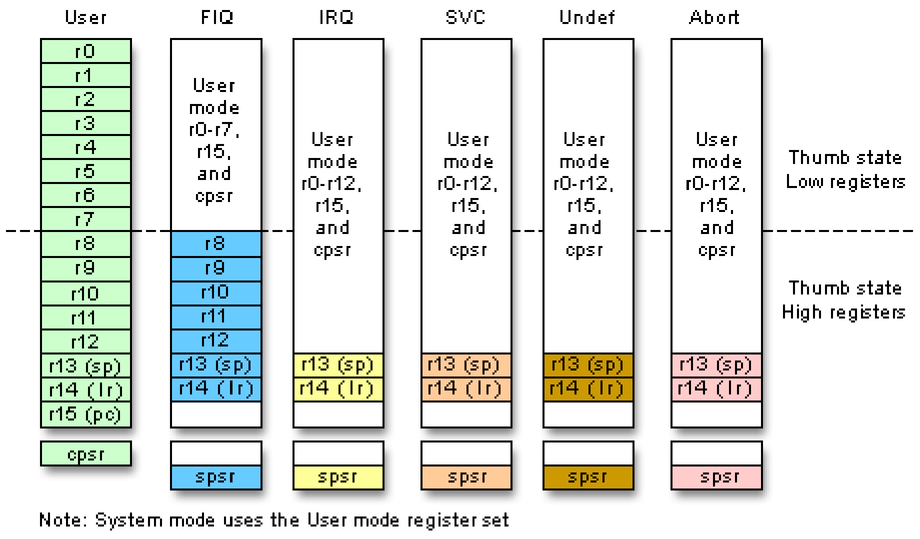
Since the introduction of the ARM7 architecture, there has been huge leaps in core processing performance. As shown here, ARM families provide a wide range of performance, from 100 MIPS to 1000 MIPS.
This increase in performance can be attributed to two main driving factors. The most obvious factor is the advances that have been made in new process technologies. The other is the engineering changes implemented in each subsequent generation of ARM processors and architectures. Specific examples include a new pipeline in the ARM9 family, and the implementation of a Harvard bus architecture in the ARM 9 over the Von Neumann architecture in the ARM7. The result is that the ARM9 family doubles the performance of the ARM7 family.
Recent developments include DSP and Jazelle-Java extensions to some of the new architectures. These products enable feature rich applications to benefit from the high-performance and low power consumption intrinsic to ARM processor cores.
Because of the fact that true embedded control applications typically require a processor with cache and memory protection to utilize real-time operating systems, ARM has developed a vertical expansion of CPUs to match these requirements. Each processor provides a unique, and in some cases configurable, amount of cache.
For example, the ARM9E-S family offers the ability to configure the size of instruction and data cache, as well as the ability to configure tightly coupled SRAM blocks. These features enable you to custom fit the CPU to specific application requirements.
Many other features can be added via the co-processor interface, such as the Vector Floating Point unit for the ARM10 and ARM9E families.
In other words, ARM has produced architectural families that are compatible, flexible, and encompass the full range of embedded requirements. Each product is designed to allow multi-sourcing at every level of development.

ARM is now the de-facto standard in embedded IP.

3. Explanation of the ARM architecture

3.1 Architecture basics

ARM cores use a 32-bit, Load-Store RISC architecture. That meanins that the core cannot directly manipulate the memory. All data manipulation must be done by loading registers with information located in memory, performing the data operation and then storing the value back to memory. There are 37 total registers in the processor. However, that number is split among seven different processor modes.
The seven processor modes are used to run user tasks, an operating system, and to efficiently handle exceptions such as interrupts. Some of the registers with in each mode are reserved for specific use by the core, while most are available for general use. The reserved registers that are used by the core for specific functions are r13 is commonly used as the stack pointer (SP), r14 as a link register (LR), r15 as a program counter (PC), the Current Program Status Register (CPSR), and the Saved Program Status Register (SPSR).
The SPSR and the CPSR contain the status and control bits specific to the properties the processor core is operating under. These properties define the operating mode, ALU status flags, interrupt disable/enable flags and whether the core is operating in 32-bit ARM or 16-bit Thumb state.


User
FIQ
IRQ
SVC
Undef
Abort
r0
r1
r2
r3
r4
r5
r6
r7
r8
r9
r10
r11
r12
r13 (sp)
r14 (lr)
r15 (pc)
cpsr
User mode r0-r7, r15, and cpsr
User mode r0-r12, r15, and cpsr
spsr
Thumb state Low registers
Thumb state High registers
Note: System mode uses the User mode register set
Figure 09: The ARM's Register Organization

There are 37 total registers divided among seven different processor modes. Fifgure 09 shows the bank of registers visible in each mode.
User mode, the only non-privileged mode, has the least number of total registers visible. It has no SPSR and limited access to the CPSR. FIQ and IRQ are the two interrupt modes of the CPU. Supervisor mode is the default mode of the processor on start up or reset. Undefined mode traps unknown or illegal instructions when they are passed though the pipeline. Abort mode traps illegal memory accesses as a result of fetching instructions or accessing data.
Finally, system mode, which uses the user mode bank of registers, was introduced to provide an additional privileged mode when dealing with nested interrupts.
Each additional mode offers unique registers that are available for use by exception handling routines. These additional registers are the minimum number of registers required to preserve the state of the processor, save the location in code, and switch between modes.
FIQ mode, however, has an additional five banked registers to provide more flexibility and higher performance when handling critical interrupts.
When the ARM core is in Thumb state, the registers banks are split into low and high register domains. The majority of instructions in Thumb state have a 3-bit register specifier. As a result, these instructions can only access the low registers in Thumb, R0 through R7. The high registers, R8 through R15, have more restricted use. Only a few instructions have access to these registers.

3.2 TDMI

T-D-M-I stands for:

- **T**humb, which is a 16-bit instruction set extension to the 32-bit ARM architecture, referred as states of the processor.
- "**D**" and "**I**" together comprise the on-chip debug facilities offered on all ARM cores. These stand for the **D**ebug signals and Embedded**I**CE logic, respectively.
- The M signifies the support for 64-bit results and an enhanced multiplier, resulting in higher performance. This multiplier is now standard on all ARMv4 architectures and above.

3.2.1 Thumb 16-bit Instructions

With growing code and data size, memory contributes to the system cost. The need to reduce memory cost leads to smaller code size and the use of narrower memory. Therefore ARM developed a modified instruction set to give market-leading code density for compiled standard C language.
There is also the problem of performance loss due to using a narrow memory path, such as a 16-bit memory path with a 32-bit processor.
The processor must take two memory access cycles to fetch an instruction or read and write data. To address this issue, ARM introduced another set of reduced 16-bit instructions labeled Thumb, based on the standard ARM 32-bit instruction set. For Thumb to be used, the processor must go through a change of state from ARM to Thumb in order to begin executing 16-bit code. This is because the default state of the core is ARM. Therefore, every application must have code at boot up that is written in ARM. If the application code is to be compiled entirely for Thumb, then the segment of ARM boot code must change the state of the processor. Once this is done, 16-bit instructions are fetched seamlessly into the pipeline without any result.
It is important to note that the architecture remains the same. The instruction set is actually a reduced set of the ARM instruction set and only the instructions are 16-bit; everything else in the core still operates as 32-bit.
An application code compiled in Thumb is 30% smaller on average than the same code compiled in ARM and normally 30% faster when using narrow 16-bit memory systems.

An example: ARM7TDMI Block Diagram

Figure 10 shows the register bank in the center of the diagram, plus the required address bus and data bus. The multiplier, in-line barrel shifter, and ALU are also shown.

In addition, the diagram illustrates the in-line decompression process of Thumb instructions while in the decode stage of the pipeline. This process creates a 32-bit ARM equivalent instruction from the 16-bit Thumb instruction, decodes the instruction, and passes it on to the execute stage.

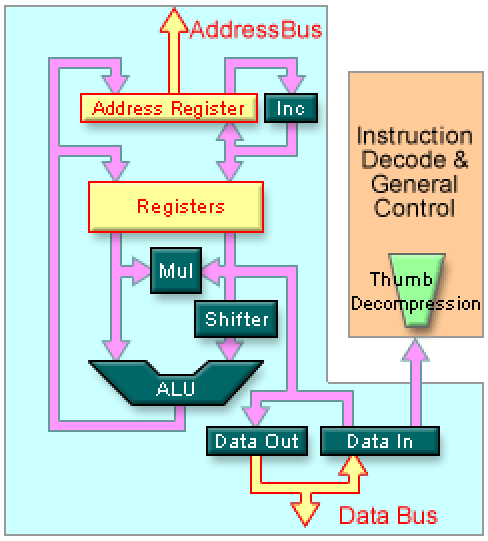


Figure 10: ARM7TDMI Block Diagram

3.2.2 Debug Extensions

The Debug extensions to the core add scan chains to monitor what is occurring on the data path of the CPU. Signals were also added to the core so that processor control can be handed to the debugger when a breakpoint or watchpoint has been reached. This stops the processor enabling the user to view such characteristics as register contents, memory regions, and processor status.

3.2.3 EmbeddedICE Logic

In order to provide a powerful debugging environment for ARM-based applications the EmbeddedICE logic was developed and integrated into the ARM core architecture. It is a set of registers providing the ability to set hardware breakpoints or watchpoints on code or data. The EmbeddedICE logic monitors the ARM core signals every cycle to check if a breakpoint or watchpoint has been hit. Lastly, an additional scan chain is used to establish contact between the user and the EmbeddedICE logic.
Communication with the EmbeddedICE logic from the external world is provided via the test access port, or TAP, controller and a standard IEEE 1149.1 JTAG connection.

The advantage of on-chip debug solutions is the ability to rapidly debug software, especially when the software resides in ROM. This is critical in shortening the development cycle. The use of Multi-ICE and EmbeddedICE provides full debug capabilities for a processor integrated deep inside an ASIC, even in a production version of a consumer product.

4. Archtecture details, features & comparison of the ARM7, ARM9, and ARM10 core families

4.1 ARM7TDMI Processor Core

- Architecture version 4T:
  -- 3-stage pipeline
  -- Unified bus architecture
  -- 32-bit ARM ISA plus 16-bit Thumb extension
- Forward compatible code
- EmbeddedICE on-chip debug
- Hard Macrocell IP
  -- Smallest Die Size: 0.53 mm^2 on 0.18 µm process
- Up to 110 MHz* on TSMC standard 0.18 µm
- Industry leading 0.25 mW/MHz

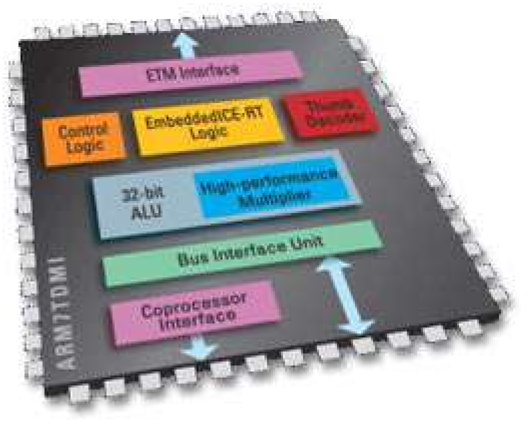


Figure 11: ARM7TDMI Core

The ARM7TDMI has a core based on the fourth version of the ARM architecture. This implementation uses a three stage pipeline - a standard fetch-decode-execute organization.
It features a unified cache, as well as the Thumb extension permitting 32-bit and 16-bit operation. It is completely forward compatible, meaning that any code written for this core will be compatible with any new core releases, such as ARM9 or ARM10. This core also includes the on-chip debug extension discussed in the previous training module.
The core is successful mainly because of the extremely small but high performance processor - slightly more than 70,000 transistors in all an with extremely low power consumption.

4.1.1 ARM7TDMI-S

- Synthesizable RTL compliant with the ARM7TDMI Custom Macrocell:
  -- Fully compatible with the ARMv4T architecture.
  -- Right denied to modify ARM7TDMI instruction set.
  -- Coprocessor interface allows custom functions to be added outside core.
  -- EmbeddedICE support with "Multi-ICE" protocol converter or third party device.
- Supports AMBA interface:
  -- Standard interface, ideal for integration of the core into an ASIC design.
- Supports full-scan and automatic test pattern generator.

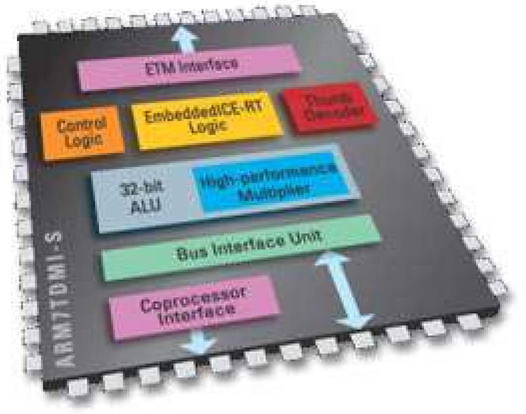


Figure 12: ARM7TDMI-S Core

Figure 12 presents a model of the ARM7 processor that is a synthesizable version of the ARM7TDMI.
This version is fully compatible with the ARMv4T architecture and is functionally identical to the hardened ARM7TDMI macrocell. Although it is a synthesizable solution, the licensee does not have the right to change any feature of this core.

4.1.2 ARM720T

- Cached Macrocell for Platform OS Applications
- ARM7TDMI core:
  -- ARM v4T ISA
  -- THUMB 16-bit instruction set
  -- Rev 3 onwards supports ETM7 for non-stop debug
- 8 KB cache:
  -- High processor performance with low-speed memory interface
- Memory Management Unit:
  -- Full support for WindowsCE and Symbian OS
- ASB bus interface

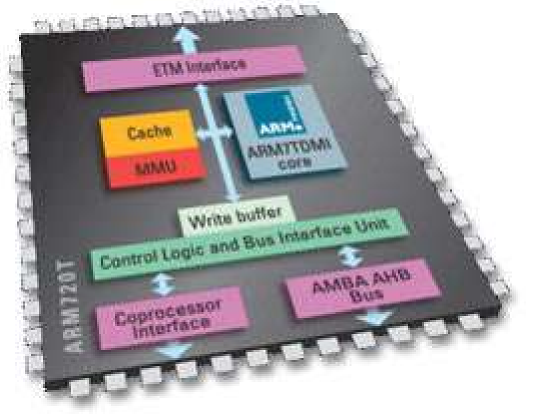


Figure 13: ARM720T Core

ARM720Tcore offers 8 KB of unified instruction and data cache. Also included is a memory management unit (MMU) that offers virtual-to-physical address translation, 64-entry translation look-aside buffer (TLB), two-level page tables stored in memory, and hardware page-table walking. There is also a highly flexible mapping scheme that supports 1 MB sections with permissions, 64 KB large pages with four sets of permissions, and 4 KB small pages with four sets of permissions.
This processor enables up to 16 domains, each with individual access rights. It also features cyclic replacement and lockdown features to lock instructions or data into cache for critical real-time code. The ARM720T was designed to be flexible and application-specific, especially for devices running complex operating systems such as Linux, Windows CE, Symbian OS, or PalmOS. It includes a system control coprocessor for cache and system initialization and the AMBA Advanced System Bus, or ASB, interface.

4.1.3 ARM7EJ

- New Jazelle-enhanced 32-bit processor core
- Thumb, Jazelle and DSP extensions
- Five stage pipeline and high performance multiplier
- Unified instruction and data bus
- v5TEJ architecture
- Real-time trace with the ETM9 macrocell
- Contact ARM for availability and characteristics data

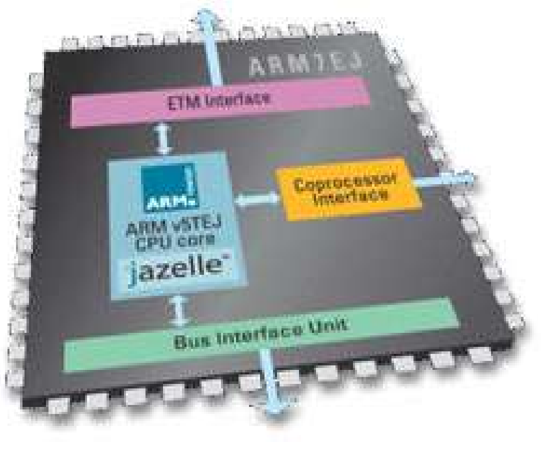


Figure 14: ARM7EJ Core

The ARM7EJ solution is a compact CPU specifically designed for applications demanding low power consumption. It has a memory interface identical to that of the ARM7TDMI-S? core. It features the V5TEJ architecture instructions, including DSP extensions. This core implementation also features a five-stage pipeline similar to that of an ARM9 class processor, and supports easy integration of the Embedded Trace Macrocell-9 for real-time-trace capability.

4.2 ARM SC100 Secure Code

- Optimized processor family for smart card solutions
    - Security enhanced ARM7TDMI design
        -- ARMv4T compliant
        -- Low power, high performance and small die size
        -- Memory Protection Unit (MPU)
        -- Anti-tampering/counterfeiting measures
        -- JavaCard support
        -- Standard coprocessor interface for incorporation of cryptographic solutions.
- SC100 - Small synthesizable IP:
    -- 35K gates - 1 mm^2 area
    -- 66 MHz* on 0.25 mm @2.5 V
    -- Power: 0.7 mW/MHz

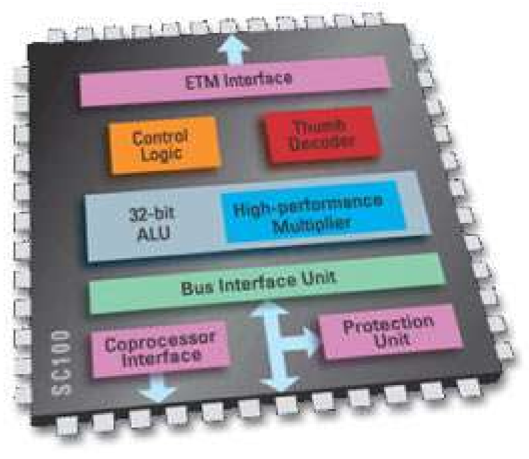


Figure 15: ARM's SC100 Core

The ARM SecurCore family provides unique 32-bit RISC-based solutions for smart card development needs, offering system designers privileged access to ARM processor cores to create fast and secure e-commerce solutions.
The flexible Memory Protection Unit was specifically designed to ensure security of operating system and application data. This enables future generations of smart card solutions having multiple applications running on a single card. Special features in the core have been designed to help obscure processor activity and hide application program signatures, making SecurCore activity difficult to detect and observe.
The SC100 runs all existing ARM JavaCard software implementations. Future SecurCore processors will include ARM's Jazelle technology for direct execution of Java byte codes to enable high-performance low-power JavaCard applications. The advantages over a purely software-emulated JavaCard virtual machine are clear: significant reduction in execution time, improved responsiveness, and significantly lower power consumption.
The SecurCore family of processors also includes a standard coprocessor interface for simple incorporation of cryptographic coprocessors. A coprocessor can be designed for a very specific purpose and can contain as many registers and data paths as needed to implement the specific functions.
To provide one solution, ARM has integrated into the SC100 core a cryptographic accelerator, the Montgomery Multiplier Engine (MME). This engine is optimized for RSA calculations, providing five times the performance of software solutions without any restrictions on key length.
The SecurCore family offers all the benefits of ARM's industry leading high-performance, low-power architecture, with significant design differences that make the ARM approach ideal for secure applications.

4.3 Comparison of the ARM7TDMI with the ARM9TDMI families

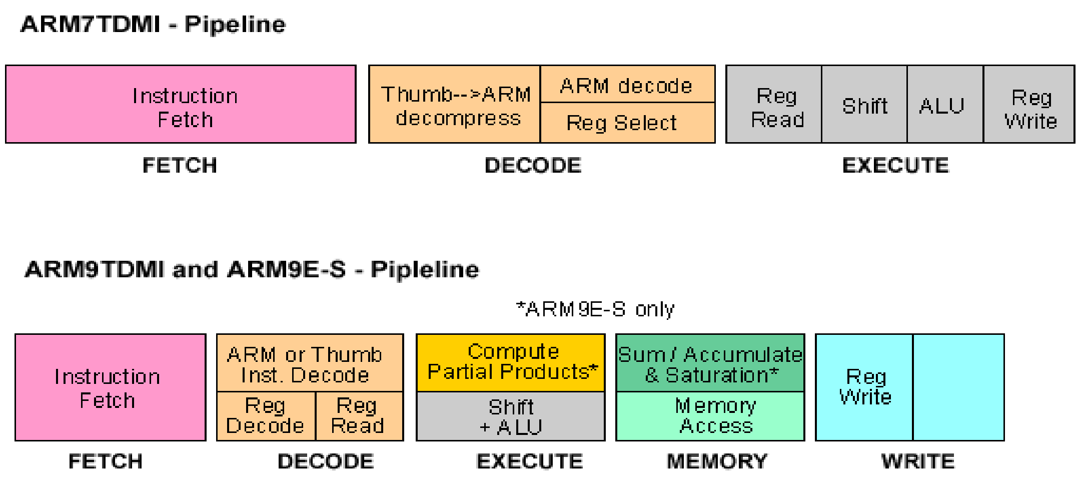


Figure 16: Pipeline Comparison

To increase performance, the pipeline of the ARM9TDMI core was re-engineered from the three-stage system used by the ARM7TDMI family to five stages.
Operations previously performed in the execute stage of ARM7 are spread across four stages in the ARM9 pipeline: decode, execute, memory, and write. The reorganization and removal of these critical paths resulted in a much higher clock frequency.
Another performance improvement is the reduced cycles per instruction rating of the processor. This is due to improved load and store instruction cycle counts. Single load and store instructions are now single-cycle operations. This is an enhancement over the ARM7 operation, which used the execute stage three times: first, to calculate the address; second, to access the memory and cache; and third, to write the data to the register bank. On ARM9, each step has a separate pipeline stage requiring only one cycle, avoiding pipeline stalls.

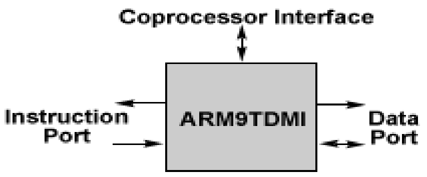
ARM7TDMI Family Applications 0 - 70 MIPS
Cellular
Organizers
MP3 Players
GPS
Mass Storage
ARM9TDMI Family Applications 60 - 200MIPS
Digital STB and NCs
Smartphones
High End Printers and MFP's
Advanced Networking and high-end modems

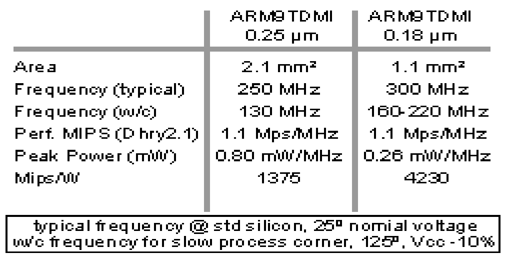
Figure 17: Applications using the ARM Family Cores

The ARM7TDMI family is popular with applications where small die size, high performance, and low power consumption help reduce system costs, especially when the system does not require cache. Applications include cellular phones, MP3 players, and mass storage.
The ARM9TDMI family are used for high performance applications that previously could not be implemented at the same cost. This family of cores was developed with twice the performance of the ARM7TDMI and without changes to the architecture. It is ideally suited for the next generation of cell phones, personal digital assistants, multi-function peripherals and fast printers, and set-top box applications.

4.4 ARM9TDMI Processor Core

- ARM 32-bit and Thumb 16-bit instructions (v4T ISA).
- Very high code compatibility with ARM7TDMI:
  -- Only change is simplified data-abort handler
- Portable to 0.25, 0.18 µm CMOS and below.
- Harvard 5-stage pipeline implementation:
  -- Higher performance from reduced cycle per instruction (1.5)
- Coprocessor interface for on-chip coprocessors:
  -- Allows floating point, DSP, graphics accelerators.
- EmbeddedICE debug capability with extensions:
  -- Hardware single step
  -- Breakpoint on exception.





	ARM9TDMI 0.25 µm	ARM9TDMI 0.18 µm
Area	2.1 mm²	1.1 mm²
Frequency (typical)	250 MHz	300 MHz
Frequency (w/c)	130 MHz	160-220 MHz
Perf. MIPS (Dhry2.1)	1.1 Mps/MHz	1.1 Mps/MHz
Peak Power (mW)	0.80 mW/MHz	0.26 mW/MHz
Mips/W	1375	4230

typical frequency @ std silicon, 25º nomial voltage
w/c frequency for slow process corner, 125º, Vcc -10%

Figure 18: ARM9TDMI Core

The Harvard bus architecture creates separate instruction and data memory interfaces, enabling simultaneous access to instructions and data.
The ARM9TDMI represents a new family of CPU technology. The enhancements made to this core family doubles the performance of the ARM7TDMI family.

4.4.1 ARM940T Macrocell

- Processor for real-time embedded applications:
  -- ARM9TDMI Core (v4T ISA)
  -- 4 KB instruction and data cache with lock-down
  -- Protection unit for RTOS
  -- Code compatible from ARM7 Thumb CPUs
  -- Hard Macro IP:
  -- 4.2 mm^2 on 0.18 µm
  -- Up to 200 MHz (worst case) on TSMC standard 0.18 µm
  -- Power: 0.75 mW/MHz

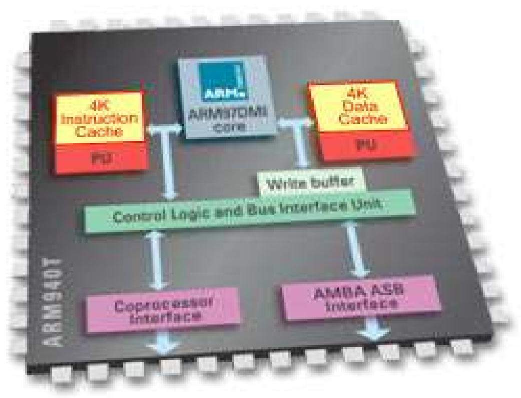


Figure 19: ARM940T Macrocell Core

The ARM940T represents the first sample of a cache-enabled ARM9TDMI core.
This core contains 4 KB each of instruction and data cache, with an MPU for use by real-time operating systems. This system makes the 940T an ideal CPU for embedded control applications, such as wireless networking devices, printers, or automotive control devices.
The protection units allow definition of eight regions of memory, each with independent cache, write buffer enable, and access permissions. The protection unit is configured using on-chip registers, which provides a simple programmer's model. This eliminates the need for page-mapping tables stored in memory.

4.4.2 ARM's 940T Core Structure

The core processor is about one-third of the die size. When other components are incorporated - the system control coprocessor, bus control, memory protection unit, and the cache itself - the integer unit becomes insignificant in total die area. This core has 4 KB caches, the smallest amount of cache used in the entire product family. One can visualize how small this integer unit and cache logic becomes when integrated with synthesized peripherals and the other features that complete the system-on-chip (SoC) design.

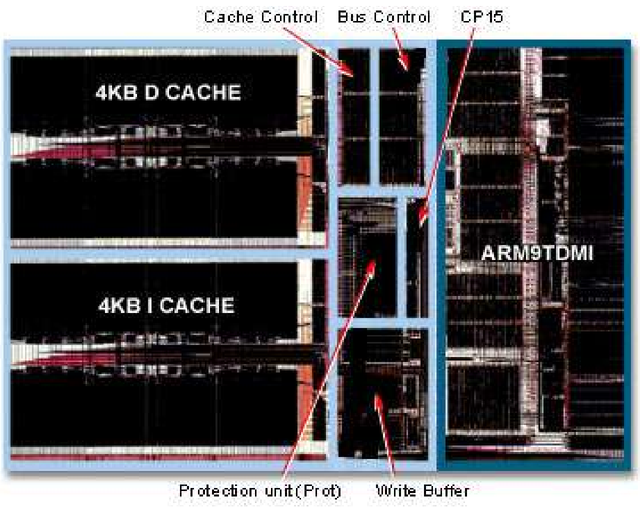


Figure 20: ARM's 940T core structure

4.4.3 ARM920T Macrocell

- Cached processor for platform OS applications:
  - -- 16 KB instruction and data cache
  - -- ARMv4 MMU for Palm OS, Symbian OS, Linux, and Windows CE
  - -- Code compatible from ARM7 Thumb CPUs
  - -- Hard Macro IP:
    - -- 11.8 mm^2 on 0.18 µm
  - -- Up to 200 MHz (worst case) on TSMC standard 0.18 µm
  - -- Power: 0.8 mW/MHz

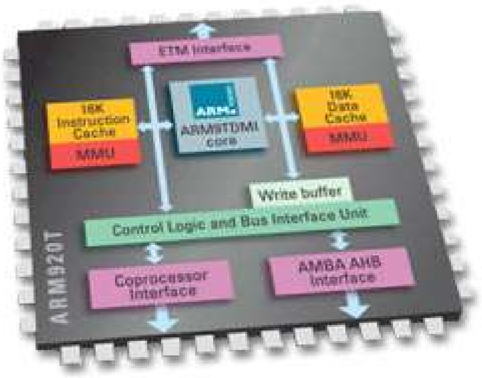


Figure 21: ARM920T Macrocell Core

The ARM920T was created to address the needs of more complex systems using a platform operating system, such as Windows CE or Symbian OS. This core replaces the MPU of the 940T with a full memory management unit, and increases the instruction and data cache sizes to 16 KB for each. The performance, MMU, and cache of this core make it ideal for Personal Digital Assistants, smartphones, and set-top box applications.

4.4.4 ARM922T

Cached processor for Platform OS applications:

- 8 KB instruction and data cache
- ARMv4 MMU for: Palm OS, Symbian OS, Linux, and Windows CE
- Code compatible from ARM7 Thumb CPUs
- Hard Macro IP:
  -- 8.1 mm^2 on 0.18 µm
- Up to 200 MHz (worst case) on TSMC standard 0.18 µm
- Power: 0.8 mW/MHz

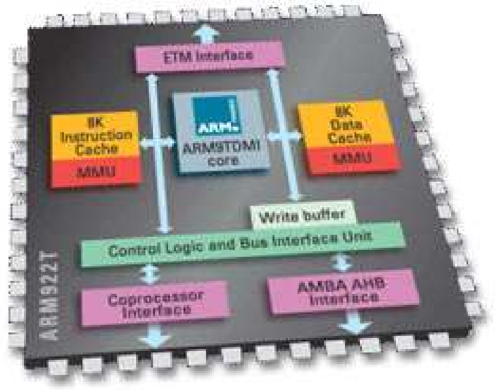


Figure 22: ARM922T Core

The ARM922T core was created with half the amount of instruction and data caches of the 920T, resulting in smaller silicon overhead. Other than this simple difference, the two cores are fundamentally identical.

4.4.5 ARM920T and ARM922T MMU

- Two TLBs:
  - -- 64-entry instruction TLB
  - -- 64-entry data TLB
- Two-level page tables (stored in memory)
- Hardware page-table walking
- Cyclic replacement
- Lockdown features:
  - -- Lock instructions or data into cache for critical real-time code

Both the 920T and 922T core utilize an MMU with the same features. There are two, 64-entry translation look-aside buffers for instruction and data, two-level page tables, hardware page-table walking, and support for random or round robin replacement. Lockdown features are also included to secure critical real-time code. This cache architecture results in two solutions that are simpler to program and minimize power, area, and required memory.

4.4.6 ARM9E Family

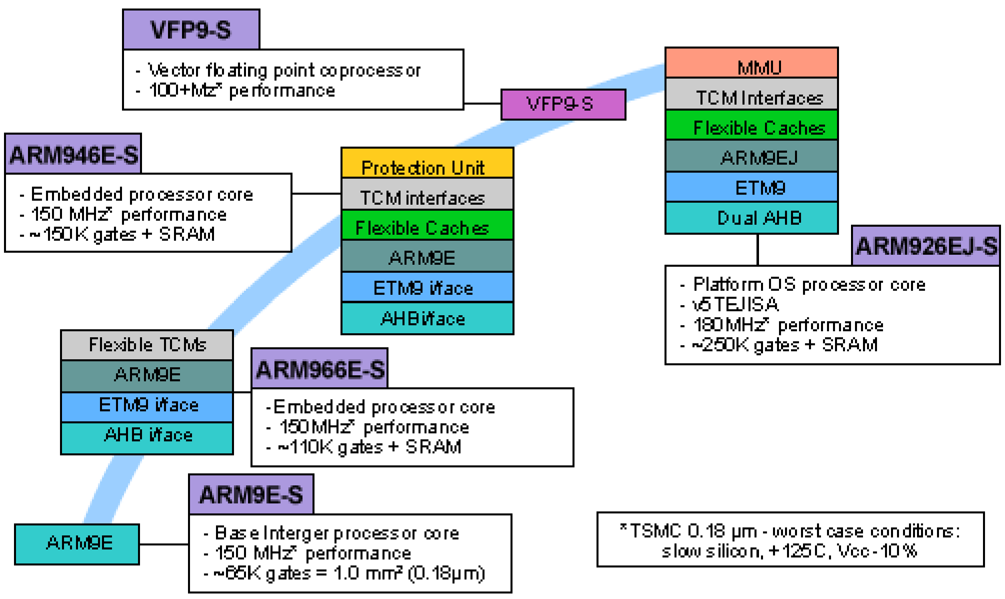


Figure 23: ARM's 9E Core Family

The ARM9E family is currently comprised of four different units. The base ARM9E integer processor offers a high performance and low gate count synthesized solution in its most basic form. The other units offer the true capabilities of the core when coupled with SRAM, cache, vector floating point acceleration, and the Jazelle Java extensions.
As a suite of synthesizable solutions, the final gate count and power consumption statistics of these cores depends on the implementation and the process technology used.

4.4.6.1 ARM9E Core Architecture

- 32-bit load/store RISC architecture
- Efficient 5-stage pipeline
- ARM andThumb instruction sets
- 37 x 32-bit registers
- 32-bit ALU and barrel shifter
- Enhanced 32-bit MAC block
- ETM9 interface
- AMBA AHB interface
- Coprocessor interface
- Synthesizable or soft IP

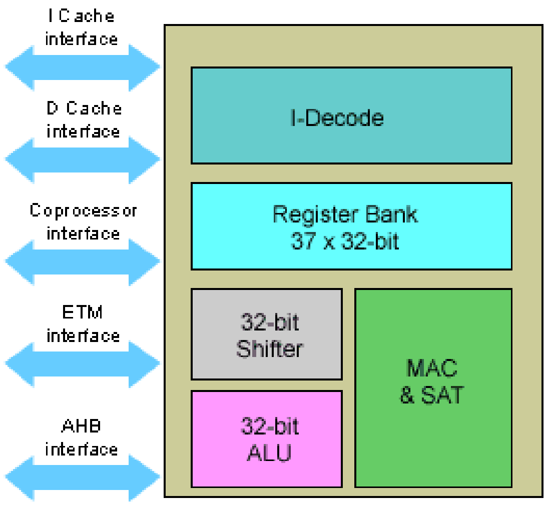


Figure 24: ARM9E Architecture

As mentioned hard macrocells always have been the ultimate answer for optimized performance and die size in any given processor design. But newer synthesized design flows are pushing the envelope for SoC applications.
The ARM9E family was built upon the standard set by the ARM9TDMI family, but it also provides freedom for defining the cache and tightly coupled SRAM configurations used by the core.
It was also the first family of CPUs designed to the AHB bus of the AMBA 2.0 specification.
Another key technological enhancement to this family of CPUs includes DSP extensions for true real-time systems. This improvement to the architecture introduces additional multiply and saturated math instructions for use by complex DSP algorithms. This family is also fully code compatible with ARMv4T architecture cores.
Lastly, to enhance the debug capabilities already common in ARM CPUs, the Embedded Trace Macrocell interface was added. This interface enables real-time debugging of complex real-time systems.

4.4.6.2 ARM966E-S

- Solution for hard real-time applications:
  - ARM9E core (v5TE ISA).
  - I and D TCM memory interfaces with 'wait' signal
  - Selectable size Instruction and Data TCM (0 KB - 64 MB)
  - AMBA AHB bus interface
  - Provides an "off-the-shelf" standard ARM9E solution
  - ETM9 interface for real-time trace with ETM9
  - 150 MHz* on TSMC 0.18 µm

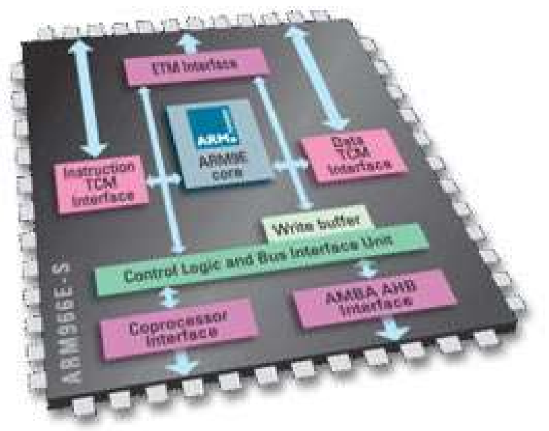


Figure 25: ARM966E-S Core

The ARM966E-S core was designed with hard real-time applications as the primary objective. An example is servo-motor control in hard disk drives. The key feature of this CPU over the base ARM9E-S is the tightly coupled memory interface that allows selectable SRAM sizes of up to 64 MB.

4.4.6.3 ARM946E-S

- Cached processor for embedded real-time applications:
    - MPU to support RTOS: like µITRON and VxWorks
    - Selectable size instruction and data caches and TCMs:(0 KB, 4 KB, 8 KB ... 1 MB)
    - Instruction and data TCM interfaces.
    - 150 MHz* on TSMC 0.18 µm

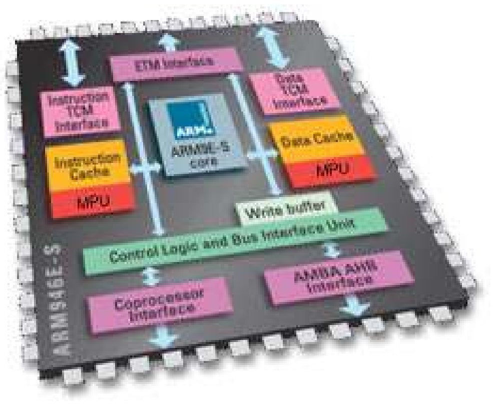


Figure 26: ARM946E-S Core

The ARM946E-S core takes the developments made by the 966E-S and adds selectable instruction and data caches. Since the memory protection unit is integrated with cache, this processor is an excellent high performance solution for embedded real-time applications, such as engine management systems in automobiles and network appliances.

4.4.6.4 ARM946E-S Caches

- Cache is 4-way set associative:
    - Can be built with compiled ASIC RAM.
- Sizes of 0 KB, 4 KB, 8 KB ? 1 MB supported:
    - I and D cache sizes are independently selectable.
- Cache lock-down on per-set basis:
    - Granularity is a quarter of the cache size.
- Software selectable replacement algorithm:
    - Supports pseudo-random and round-robin
- Write through and write back s/w selectable
- Line length fixed at 8 words

The cache memory blocks of this core are selectable up to 1 MB. The cache is 4-way set associative and selectable up to 1 MB. It also features lock-down support on a per-set basis, random and round robin replacement support, software selectable options for write through and write back, and eight-word cache lines.

4.4.7 ARM9EJ-S Core Architecture

- 32-bit load/store RISC architecture
- Efficient 5-stage pipeline:
    - Fetch
    - Decode
    - Execute
    - Memory
    - Writeback
- ARM, Thumb and Java instruction sets
- 31 x 32-bit registers
- 32-bit ALU and barrel shifter
- Enhanced 32-bit MAC block
- ETM9 interface
- Coprocessor interface

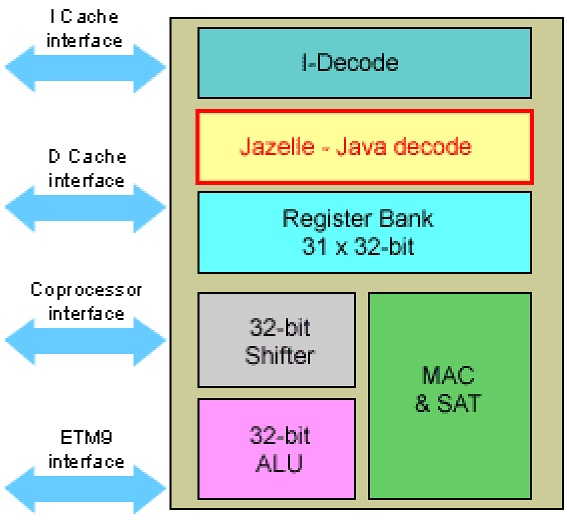


Figure 27: ARM9EJ-S Core Architecture

This family of synthesizable CPUs soon will include an additional enhancement to the architecture: Jazelle Java extensions. The Jazelle technology developed for this new range of cores will in many ways act like the Thumb 16-bit extension. An additional state of the processor is added to support the execution of Java instructions, providing a tremendous performance improvement over current solutions.
This range of cores will have all the characteristics of the other ARM9E cores and will be code-compatible with ARMv4T architecture implementations. It incorporates a separate instruction and data AMBA AHB bus interface, as well an Embedded Trace Macrocell-9 interface.

4.4.7.1 ARM926EJ-S

- Jazelle enhanced cached processor for OS-based platform applications:
    - MMU to support: Symbian OS, Linux, Palm OS, and Windows CE
    - Selectable size instruction and data caches - 4 K, 8 K, 16 K…128 K
    - Tightly coupled memories supported 0 K, 4 K, 8 K…1MB
    - Instruction and data TCM interfaces with wait state support
    - Separate instruction and data AHB buses
    - ETM9 interface for real-time trace with the ETM9 macrocell
    - 180 MHz* on TSMC 0.18 µm

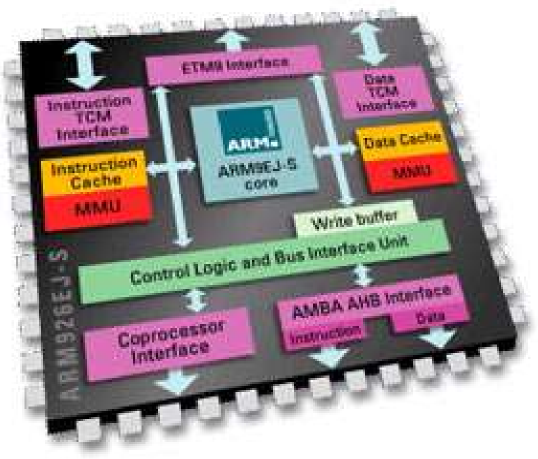


Figure 28: ARM926EJ-S

The ARM926EJ-S core, with full MMU support and selectable tightly coupled memory and cache sizes, introduces a new generation of Internet-enabled devices. For example, set-top-boxes and wireless communications products benefit from this single processor solution. This processor can be compared to the ARM920T or 922T cores in its base functionality and performance.
Now, with the added Jazelle enhancements, Java functions can be performed without the need for complicated coprocessors or slow software implementations.

4.5 ARM10E Architecture Enhancements

ARM10E implements:

- Harvard 6-stage pipeline
- Supports v5TE instruction set
- EmbeddedICE RTII debug logic
- Fully compatible with v4T architecture

- 390-700 MIPS integer performance based on Dhrystone 2.1
- Branch prediction:
    - Eliminates 70% of branches on typical code sequences
- Separate load/store unit:
    - 64-bit path to register bank - load two registers simultaneously
- Hit-under-miss caches:
    - Significantly reduces pipe-line stalls
- Write buffer:
    - Holds up to 8 double-words (16 register values)
- New energy saving power down modes

Anticipating the market's needs for multimedia digital consumer devices, ARM developed the ARM10 family of advanced microprocessor cores with 390-700 MIPS integer performance. To achieve this performance, additional features were added. The pipeline was widened to add an additional stage, and improvements were made to the EmbeddedICE logic to provide support for real-time debug. All the while, compatibility was maintained with ARMv5TE and v4T for ease of code migration.
Performance enhancements include the introduction of branch prediction, hit-under-miss support in the MMU and cache architecture, an improved write buffer that holds up to eight double-words, and a separate load and store unit. These features improve code performance by lowering the average number of cycles per instruction of the processor, and also help when code is heavily dependent on cache operations.
As an added enhancement, the architecture, circuits, layout, and software controlled power-down modes have been developed specifically to achieve low-power operation on high-performance processes. These enhanced features have been optimized to take advantage of clock gating and dynamic power reduction.

4.5.1 High Performance Features

- 64-bit accesses to on-chip I and D caches:
    - Fetch two instructions/cycle
    - Load/store two registers/cycle (LDM/STM)
- Dual 64-bit fast AHB bus:
    - Separate buses for instruction and data
    - >1 Gbyte/sec bandwidth @ 200 MHz (each)
    - Split transaction extensions
- 64-bit coprocessor interface:
    - Load/store double-precision operands in one cycle
- 32-bit integer data path saves area and power

The ARM10E is also the first family of processors designed with a 64-bit data bus. This feature combines the frugal power and die size characteristics of a 32-bit CPU with the bandwidth requirements of high performance systems. The 64-bit coprocessor interface also allows for increased performance of floating point operations when combined with the Vector Floating Point-10 coprocessor.

4.5.2 ARM1020E and ARM1022E

- Highest performance ARM processor cores:
  - 1.3 MIPS/MHz
  - 1.5x ARM9 performance
- Support for High Performance IEEE 754 Floating Point:
  - 600-1200 MFLOPS
- 300MHz (worst case) on TSMC 0.15 µm
- Low Power: 0.7 mW/mips (0.15 µm)
- ARM1020E: 32K I and D cache
  - 17.5 mm^2
- ARM1022E: 16K I and D cache
  - 12 mm^2
- Roadmap to Jazelle enhanced cores

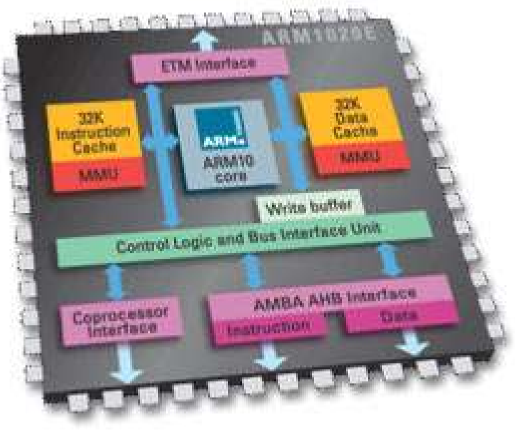


Figure 29: ARM1020E Core

The ARM1020E and ARM1022E processor cores offer the highest performance per unit of power of any 32-bit processor running above 200 MHz. With an unprecedented 0.7 mW/MIPS power consumption ratio, worst case on 0.15 µm process technology, these processors offers ideal solutions for high end platform applications. Examples include MPEG4 videophones, smartphones, and Web pads.
Memory Management and caches are comparable to the ARM920T, ARM922T and ARM720T products - ensuring code portability and protection for existing software investments. Future implementation in this family will also integrate the Jazelle Java enhancements established by the ARM9EJ-S family.

4.6 Vector Floating Point (VFP10)

- High-performance IEEE 754 floating point:
  - Single and double precision
  - Vector operations (up to 8 values per vector)
  - Thirty-two 32-bit (SP) registers (usable as sixteen DP registers)
  - Single cycle FMAC throughput (single precision - double precision FMAC in 2 cycles)
  - 10-100x performance increase over software emulation
- Optional coprocessor:
  - 1.6 mm2 in 0.15 ?m
- Target:
  - Printers, imaging, graphics, embedded control

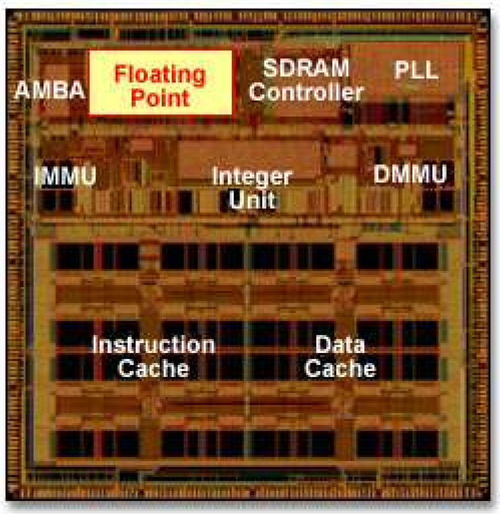
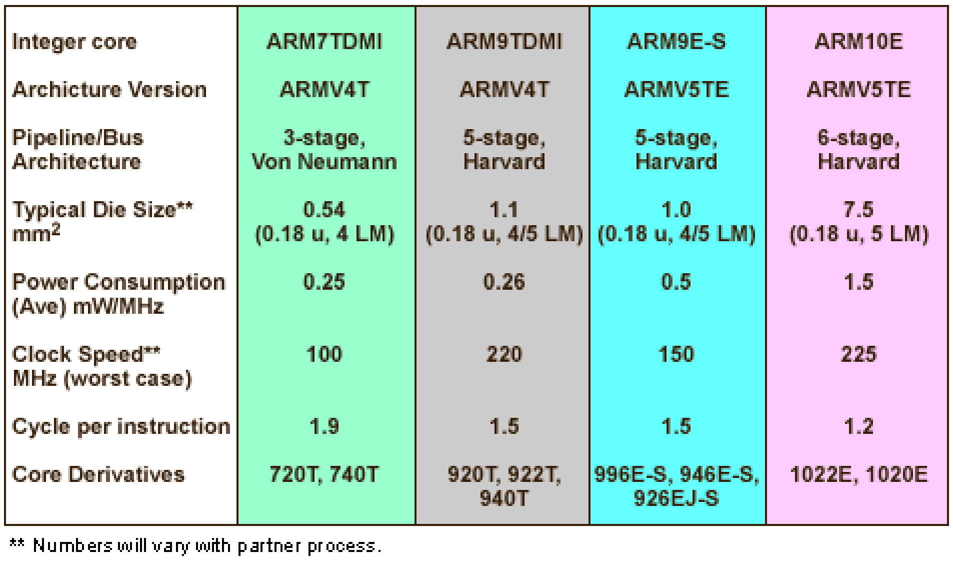


Figure 30: ARM's Vector Floating Point

Many real-time control applications in the industrial and automotive fields benefit from the dynamic range and precision of floating-point offered by the ARM VFP10. Automotive power train, anti-lock braking, traction control, and active suspension systems are examples of mission-critical applications where precision and predictability are essential requirements. Incorporating the ARM VFP10 into a SoC design can reduce development time and provide reliable performance. The vector processing capability of the ARM VFP10 also offers increased performance for imaging applications, such as scaling, transforms, and font generation used in printing, 3D transforms, FFT, and graphic filtering.

4.7 Family Summary



Integer core	ARM7TDMI	ARM9TDMI	ARM9E-S	ARM10E
Archicture Version	ARMV4T	ARMV4T	ARMV5TE	ARMV5TE
Pipeline/Bus Architecture	3-stage, Von Neumann	5-stage, Harvard	5-stage, Harvard	6-stage, Harvard
Typical Die Size** mm^2	0.54 (0.18 u, 4 LM)	1.1 (0.18 u, 4/5 LM)	1.0 (0.18 u, 4/5 LM)	7.5 (0.18 u, 5 LM)
Power Consumption (Ave) mW/MHz	0.25	0.26	0.5	1.5
Clock Speed** MHz (worst case)	100	220	150	225
Cycle per instruction	1.9	1.5	1.5	1.2
Core Derivatives	720T, 740T	920T, 922T, 940T	996E-S, 946E-S, 926EJ-S	1022E, 1020E

** Numbers will vary with partner process.

Figure 30: Family Summary

This table compares various characteristics of the ARM7TDMI, the ARM9TDMI, the ARM9E-S, and the ARM10E integer units. Although these cores will not be necessarily implemented as integer units, especially in the case of the ARM10E, it is useful to illustrate the features that make up each product family. As stated earlier, the goal of developing the ARM9 was to double the performance of the ARM7 while maintaining architectural compatibility. This was achieved by reorganizing and widening the pipeline and migrating to a Harvard bus architecture. Although these changes did result in an increase in die area, the ARM9TDMI is an extremely small core with roughly 120,000 transistors in all.
This philosophy continued with the development of the ARM10E family. The performance level of this family requires the use of caches to meet the needs of merging applications. Paired with the vector floating point coprocessor, the result is a comprehensive solution for high-end SoC implementations never before possible at this mW/MIPS ratio or silicon overhead.

5. Some words about the ARM's AMBA architecture - An open bus standard

The "**A**dvanced **M**icro-controller **B**us **A**rchitecture" on-chip bus is freely available from ARM and offers an established, open specification that serves as a framework for SoC designs.

- Benefits of a System-on-Chip (SoC) solution
    - Low power consumption
    - Small silicon area
    - Low production cost for large quantities

    - Examples include AMBA:
        - EPXA10 (Configurable SoC)
        - EP7209 (MP3)
        - ARM7100 (PDA)

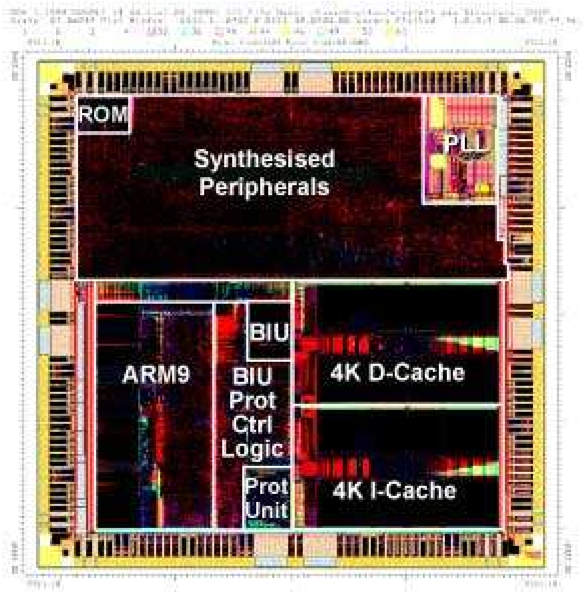


Figure 30: 802.11 Wireless LAN

The purpose of AMBA is to offer pre-built, tested, and proven components for designers to leverage their hard-earned knowledge into future designs. This method enables test methodology to be reapplied with great confidence.

Since the release of the AMBA specification, ARM has developed a suite of enablers to quickly orient engineers on the design-in of this standard bus protocol, the AMBA Design Kit.

This design kit is a licensable product that enables SoC designers to become familiar with the AMBA protocol.

ARM also has developed a suite of system peripheral components called PrimeCells. These are proven AMBA-compliant modules that can be used "off-the-shelf" for integration in SoC designs.

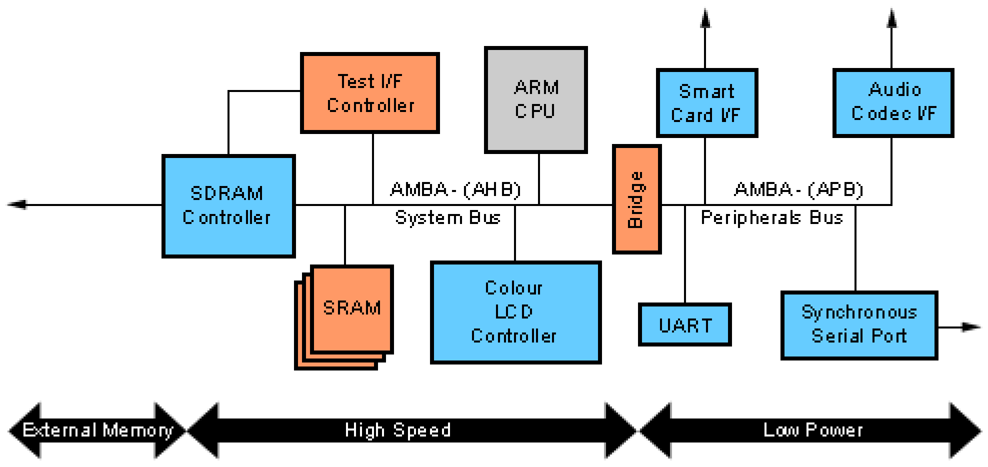


Figure 31: ARM's AMBA architecture

The design of the AMBA bus specification is focused on low power consumption and high performance. A typical AMBA-based SoC consists of an advanced high-performance system bus, or AHB, and an advanced low power peripheral bus, or APB. Figure 31 illustrates a simple implementation of the AMBA bus scheme in a real-world application.

- On the performance critical side of the bus is the ARM core, Memory Controller, Test Interface Controller (TIC), and the LCD Controller.
- On the low power side of the bus is the Smart Card interface, audio codec, UART, and synchronous serial port.
- This is an excellent example of how the AHB and APB buses work in conjunction to provide a complete system solution.

The AMBA test methodology provides a mechanism to give an external tester access to the on-chip AMBA bus. This enables the tester to take control of the bus and check each component separately.

6. Keywords:

AHB:	Advanced Hich-performance Bus
ARM:	Advanced Risk Mashines
APB:	Advanced Peripheral Bus
ASIC:	Application Specific Integrated Circuits
CPSR:	Curent Program Status Register
DSP:	Digital Signal Processor
IP:	Intellectual Property
LR:	Link Register
MMU:	Memory Managmant Unit
MPU:	Memory Protection Unit
OEM:	Original Equipment Manufuctures
PC:	Program Counter
RISK:	Reduced Instruction Set Computing
RTD:	Real Time Debug
RTOS:	Real-Time Operating Systems
RTT:	Real Time Trace
SoC:	System-on-Chip
SP:	Stack Point
SPSR:	Stack Point States Register
TAP:	Test Access Port
VCX:	Virtual Component Exchange
VSIA:	Virtual Socket Interface Alliance

7. Resources:

http://www.arm.com
http://www.arm.com/documentation
http://www.arm.com/aboutarm/multimedia.html
http://www.arm.com/support/training/type4680.html

http://techonline2.techonline.com/
http://www.techonline.com/community/ed_resource
http://www.techonline.com/community/ed_resource/course#Microprocessors
http://www.techonline.com/community/ed_resource/course/13071