ARM Processor Architecture

Pao-Ann Hsiung
National Chung Cheng University
Brief History of ARM

• ARM is short for Advanced Risc Machines Ltd.
• Founded 1990, owned by Acorn, Apple and VLSI
• Known as computer manufacturer before becoming ARM
• Acorn developed a 32-bit RISC processor for it’s own use Acorn Archimedes
Why ARM here?

- ARM is one of the most licensed and thus widespread processor cores in the world
- Used especially in portable devices due to low power consumption and reasonable performance (MIPS / watt)
- Several interesting extensions available or in development like Thumb instruction set and Jazelle Java machine
ARM

- Processor cores: ARM6, ARM7, ARM9, ARM10, ARM11
- Extensions: Thumb, El Segundo, Jazelle etc.
- IP-blocks: UART, GPIO, memory controllers, etc

<table>
<thead>
<tr>
<th>CPU</th>
<th>Description</th>
<th>ISA</th>
<th>Process</th>
<th>Voltage</th>
<th>Area mm²</th>
<th>Power mW</th>
<th>Clock / MHz</th>
<th>Mips / MHz</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARM7TD MI</td>
<td>Core</td>
<td>V4T</td>
<td>0.18u</td>
<td>1.8V</td>
<td>0.53</td>
<td>&lt;0.25</td>
<td>60-110</td>
<td>0.9</td>
</tr>
<tr>
<td>ARM7TD MI-S</td>
<td>Synthesizable core</td>
<td>V4T</td>
<td>0.18u</td>
<td>1.8V</td>
<td>&lt;0.8</td>
<td>&lt;0.4</td>
<td>&gt;50</td>
<td>0.9</td>
</tr>
<tr>
<td>ARM9TD MI</td>
<td>Core</td>
<td>V4T</td>
<td>0.18u</td>
<td>1.8V</td>
<td>1.1</td>
<td>0.3</td>
<td>167-220</td>
<td>1.1</td>
</tr>
<tr>
<td>ARM920T</td>
<td>Macrocell 16+16kB cache</td>
<td>V4T</td>
<td>0.18u</td>
<td>1.8V</td>
<td>11.8</td>
<td>0.9</td>
<td>140-200</td>
<td>1.05</td>
</tr>
<tr>
<td>ARM940T</td>
<td>Macrocell 8+8kB cache</td>
<td>V4T</td>
<td>0.18u</td>
<td>1.8V</td>
<td>4.2</td>
<td>0.85</td>
<td>140-170</td>
<td>1.05</td>
</tr>
<tr>
<td>ARM9E-S</td>
<td>Synthesizable core</td>
<td>V5TE</td>
<td>0.18u</td>
<td>1.8V</td>
<td>?</td>
<td>~1</td>
<td>133-200</td>
<td>1.1</td>
</tr>
<tr>
<td>ARM1020 E</td>
<td>Macrocell 32+32kB cache</td>
<td>V5TE</td>
<td>0.15u</td>
<td>1.8V</td>
<td>~10</td>
<td>~0.85</td>
<td>200-400</td>
<td>1.25</td>
</tr>
</tbody>
</table>
ARM architecture

- 32-bit RISC-processor core (32-bit instructions)
- 37 pieces of 32-bit integer registers (16 available)
- Pipelined (ARM7: 3 stages)
- Cached (depending on the implementation)
- Von Neuman-type bus structure (ARM7), Harvard (ARM9)
- 8 / 16 / 32 -bit data types
- 7 modes of operation (usr, fiq, irq, svc, abt, sys, und)
- Simple structure -> reasonably good speed / power consumption ratio
ARM7TDMI

• ARM7TDMI is a core processor module embedded in many ARM7 microprocessors, such as ARM720T, ARM710T, ARM740T, and Samsung’s KS32C50100. It is the most complex processor core module in ARM7 series.
  – **T**: capable of executing Thumb instruction set
  – **D**: Featuring with IEEE Std. 1149.1 JTAG boundary-scan debugging interface.
  – **M**: Featuring with a Multiplier-And-Accumulate (MAC) unit for DSP applications.
  – **I**: Featuring with the support of embedded In-Circuit Emulator.

• **Three Pipe Stages**: Instruction fetch, decode, and Execution.
Features

- A 32-bit RSIC processor core capable of executing 16-bit instructions (Von Neumann Architecture)
  - High density code
    - The Thumb’s set’s 16-bit instruction length allows it to approach about 65% of standard ARM code size while retaining ARM 32-bit processor performance.
  - Smaller die size
    - About 72,000 transistors
    - Occupying only about 4.8mm² in a 0.6um semiconductor technology.
  - Lower power consumption
    - dissipate about 2mW/MHZ with 0.6um technology.
Features Cont!

- **Memory Access**
  - Data can be
    - 8-bit (bytes)
    - 16-bit (half words)
    - 32-bit (words)

- **Memory Interface**
  - Can interface to SRAM, ROM, DRAM
  - Has four basic types of memory cycle
    - idle cycle
    - nonsequential cycle
    - sequential cycle
    - coprocessor register cycle
ARM7TDMI Block Diagram
ARM7TDMI on Silicon
Memory Interface

• 32-bit address bus

• 32-bit data bus
  – D[31:0]: Bidirectional data bus
  – DIN[31:0]: Unidirectional input bus
  – DOUT[31:0]: Unidirectional output bus

• Control signals
  – Specify the size of the data to be transferred and the direction of the transfer
Cycle Types (1)

- **Non-sequential cycle**
  - Requests a transfer to or from an address which is unrelated to the address used in the preceding cycle.

- **Sequential cycle**
  - Requests a transfer to or from an address which is either the same as the address in the preceding cycle, or is one word or halfword after the preceding cycle. But should not occur at the page end.

- **Internal cycle**
  - Does not require a transfer.

- **Coprocessor register transfer**
  - Wishes to use the data bus to communicate with a coprocessor, but does not require any action by the memory system.
## Cycle Types (2)

<table>
<thead>
<tr>
<th>nMREQ</th>
<th>SEQ</th>
<th>Cycle type</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>Non-sequential (N-cycle)</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>Sequential (S-cycle)</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>Internal (I-cycle)</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Coprocessor register transfer (C-cycle)</td>
</tr>
</tbody>
</table>
Bus Configuration

• **Pipelined**
  – Interface to DRAM
  – Set $APE$ to 1

• **De-pipelined**
  – Interface to SRAM or ROM
  – Set $APE$ to 0
Pipelined Addresses
De-pipelined Addresses
Coprocessor Interface

• Can have 16 coprocessors
• If the designated coprocessor
  – exists, a coprocessor instruction will be executed by the coprocessor;
  – does not exist, the ARM7TDMI will take the undefined instruction trap and a software will be executed to emulate the coprocessor. The execution of a coprocessor instruction is done by software.
Debug Interface

- Based on IEEE Std. 1149.1-1990, “Standard Test Access Port and Boundary-Scan Architecture”.
- Debug systems
The Concept of Boundary Scan Design
ARM7TDMI Debug Architecture
Enter Debugging State

• ARM7TDMI is forced into debug state after a breakpoint, watchpoint, or debug request has occurred.
  – **Breakpoint**: Set for *a particular instruction*. When this instruction is executed, the machine is forced into debug state.
  – **Watchpoint**: Set for a data access. When data access occurs for *a particular address for a particular data*, the machine is forced into debug state.

• **Setting the breakpoint and watchpoint by**
  – issuing debug request, DBGRQ.
  – ICEBreaker programming
Action In Debug State

• Once entered into debug state
  – The internal states of the machine can be examined.
  – The system’s external state can be examined.
  – The memory bus of the machine (ARM7TDMI) is forced to indicate internal cycles and the machine’s outputs will change asynchronously to the memory system.
  – Then, the internal state of the machine can be scanned out through scan chain for examination.
ARM7TDMI ICEBreaker

- It is programmed through TAP controller.
- It consists of two real-time watchpoint units with a control and status register.
  - Each watchpoint unit can be configured to be a watchpoint or a breakpoint.
- Execution of ARM7TDMI is halted when
  - a match occurs between the values programmed into ICEBreaker and the values currently appearing on the address bus, data bus and various control signals.
# Watchpoint Registers

<table>
<thead>
<tr>
<th>Address</th>
<th>Width</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>00000</td>
<td>3</td>
<td>Debug Control</td>
</tr>
<tr>
<td>00001</td>
<td>5</td>
<td>Debug Status</td>
</tr>
<tr>
<td>00100</td>
<td>6</td>
<td>Debug Comms Control Register</td>
</tr>
<tr>
<td>00101</td>
<td>32</td>
<td>Debug Comms Data Register</td>
</tr>
<tr>
<td>01000</td>
<td>32</td>
<td>Watchpoint 0 Address Value</td>
</tr>
<tr>
<td>01001</td>
<td>32</td>
<td>Watchpoint 0 Address Mask</td>
</tr>
<tr>
<td>01010</td>
<td>32</td>
<td>Watchpoint 0 Data Value</td>
</tr>
<tr>
<td>01011</td>
<td>32</td>
<td>Watchpoint 0 Data Mask</td>
</tr>
<tr>
<td>01100</td>
<td>9</td>
<td>Watchpoint 0 Control Value</td>
</tr>
<tr>
<td>01101</td>
<td>8</td>
<td>Watchpoint 0 Control Mask</td>
</tr>
<tr>
<td>10000</td>
<td>32</td>
<td>Watchpoint 1 Address Value</td>
</tr>
<tr>
<td>10001</td>
<td>32</td>
<td>Watchpoint 1 Address Mask</td>
</tr>
<tr>
<td>10010</td>
<td>32</td>
<td>Watchpoint 1 Data Value</td>
</tr>
<tr>
<td>10011</td>
<td>32</td>
<td>Watchpoint 1 Data Mask</td>
</tr>
<tr>
<td>10100</td>
<td>9</td>
<td>Watchpoint 1 Control Value</td>
</tr>
<tr>
<td>10101</td>
<td>8</td>
<td>Watchpoint 1 Control Mask</td>
</tr>
</tbody>
</table>
ICEBreaker Block Diagram
ARM720T
AMBA

**AMBA: Advanced Microcontroller Bus Architecture**

- It is a specification for an on-chip bus, to enable macrocells (such as a CPU, DSP, Peripherals, and memory controllers) to be connected together to form a microcontroller or complex peripheral chip.
- It defines
  - A high-speed, high-bandwidth bus, the Advanced High Performance Bus (AHB).
  - A simple, low-power peripheral bus, the Advanced Peripheral Bus (APB).
  - Access for an external tester to permit modular testing and fast test of cache RAM
  - Essential house keeping operations (reset/power-up, …)
A Typical AMBA-based Microcontroller
ARM9(TDMI)

- ARM7 microarchitecture is getting old and will be replaced with ARM9
- ARM9 realizes the same (v4T) instruction set as ARM7 and is thus binary compatible
- Pipeline length is 5 stages instead of ARM7 3 stages. This allows for faster clocking.
- Available with TDMI extensions
- ARM92x: ARM9TDMI and caches as a macrocell
- Caches are separate for instructions and data (Harvard Architecture)
ARM9 Processor Core
ARM10(TDMI)

• ARM10TDMI processor core:
• Realizes the ARM instruction set with binary compatibility including the Thumb extension
• Instruction set expanded to version 5 (v5TE), 32x16 MAC-multiplier
• 6-stage pipeline for fixed point instructions
ARM as a standard component

• Even tough ARM is mostly used as a processor core in SoC and other ASICs, some manufacturers have brought ARM-based standard products to market
• Examples of manufacturers: Atmel, Cirrus Logic, Hyundai, Intel, Oki, Samsung, Sharp …
• Most of the products are based on 7TDMI-core, some are based on 720T and 920T-cores
• ARM + FPGA are implemented by Altera and Triscend
• In addition, there are a number of ASSP (Application Specific Standard Product)-chips available
  – for example to communication applications (Philips VWS22100 with ARM7-based GSM baseband chip).
Atmel ARM

- AT91-series:
  - ARM7TDMI-core
  - External bus controller
  - A load of peripherals
  - Variable amount of SRAM on die (up to 2 megabits)
Altera ARM + FPGA

- ARM922T macrocell and programmable logic on same chip
  - System-on-a-programmable-chip

<table>
<thead>
<tr>
<th>Table 1. Current ARM-Based Embedded Processor Device Features (1)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Feature</strong></td>
</tr>
<tr>
<td>Maximum system gates</td>
</tr>
<tr>
<td>Typical gates</td>
</tr>
<tr>
<td>Logic elements (LEs)</td>
</tr>
<tr>
<td>Embedded system blocks (ESBs)</td>
</tr>
<tr>
<td>Maximum RAM bits</td>
</tr>
<tr>
<td>Maximum macrocells</td>
</tr>
<tr>
<td>Maximum user I/Os</td>
</tr>
<tr>
<td>Single-port SRAM</td>
</tr>
<tr>
<td>Dual-port SRAM</td>
</tr>
</tbody>
</table>
Intel ARM derivatives

- **StrongARM**
  - DEC developed ARM variant
  - Being phased out with XScale

- **XScale**
  - ARM v5TE instruction set
  - Intel developed microarchitecture
  - Coprocessor instructions used for extensions
Triscend ARM + FPGA

- Triscend A7:

Block Diagram of the Triscend A7 Configurable System-on-Chip