ARM Architecture

Hsung-Pin Chang
Department of Computer Science
National Chung Hsing University
Outline

- ARM
- ARM Architectures
- ARM Registers
- ARM Exceptions
- ARM Instruction Set
ARM

- The first RICS microprocessor developed for commercial use
- In 1990, ARM Limited was established
  - A market-leader for low-power and cost-sensitive embedded applications
The ARM is supported by a toolkit

- No processor is particularly useful without hardware and software development tools

- Instruction set emulator, assembler, C and C++ compilers, a linker and a symbolic debugger…
Example: ARM7TDMI

- Developed by Advanced RISC Machines
- 32-bit RISC embedded processor
- Low-end ARM core for applications like digital mobile phones
- Von Neumann, load/store architecture
  - Only 32 bit data bus for both instr. and data.
  - Only the load/store instr (and SWP) access memory.
- 3-stage pipeline
  - Fetch
  - Decode
  - Execute
Example: ARM7TDMI (Cont.)

- **T**: in addition to the 32-bit ARM instruction set, also support 16-bit Thumb instructions
- **D**: on-chip Debug support
  - Enable the processor to halt in response to a debug request
- **M**: an enhanced Multiplier
  - Can yield a full 64-bit result
- **I**: EmbeddedICE hardware to give on-chip breakpoint and watchpoint support
  - The embeddedICE module introduces *breakpoint* and *watchpoint registers* that are accessed using JTAG interface
ARM7TDMI Block Diagram
ARM7TDMI Core Block Diagram
ARM Architecture

- Reduced Instruction Set Computer (RISC) architecture
  - A large set of registers
  - A load-store architecture
    - Process values in registers and place the results into a register
    - Data processing operations only operate on register contents, not directly on memory contents
  - Uniform and fixed-length instruction fields
    - Simplify instruction decode
    - 3-address instruction formats
      - Two source operand registers and the result register are all independent specified
ARM Architecture (Cont.)

- Control over both the Arithmetic Logic Unit (ALU) and shifter in every data-processing instruction
  - Maximize the use of an ALU and a shifter
- Auto-increment and auto-decrement addressing mode
  - Optimize program loops
- Powerful load/store multiple register instructions
  - Maximize data throughputs
- Conditional execution of every instruction
  - Maximize execution throughput
- Open instruction set extension through the coprocessor instruction set
ARM Registers

- ARM has 31 general-purpose 32-bit registers
  - 16 of them are visible at any time
    - User mode registers used by all unprivileged mode
    - Link registers (R14)
      - Hold the address of the next instruction after a Branch and Link (BL) instruction
    - Program counter (R15)
      - A pointer to the instruction which is two instructions after the instruction being executed
  - The other 15 registers are used to speed up exception handling
- Besides, ARM has 6 status registers
ARM Exceptions

- ARM supports five types of exceptions, and a privileged processing mode for each type
  - Fast interrupt
  - Normal interrupt
  - Memory abort
    - Used to implement memory protect or virtual memory
  - Attempted execution of an undefined instruction
  - Software interrupt (SWI) instruction
    - Make a system call to an operating system
ARM Exceptions (Cont.)

- When an exception occurs
  - Some of the standard registers are replaced with registers specific to the exception mode

- When an exception occurs
  - The ARM processor halts execution after the current instructions
  - Then begin execution at a fixed memory address, called *exception vectors*
The ARM instruction set can be divided into six board classes:

- Branch instructions
- Data-processing instructions
- Status register transfer instructions
- Load and store instructions
- Coprocessor instructions
- Exception-generating exceptions
  - Software interrupt instructions (SWI)
  - Software breakpoint instructions (BKPT)