MODULE-6
ARM Processor Architecture

V.PRAKASH
Asst. Professor(Sr.), SENSE,
VIT Chennai

MODULE-6

ARM Processor Architecture

ARM Design Philosophy; Overview of ARM architecture; States [ARM, Thumb, Jazelle]; Registers, Modes; Conditional Execution; Pipelining; Vector Tables; Exception handling.

INTRODUCTION

INTRODUCTION

- The ARM stands for Advance RISC Machine and it is one of the extensive and most licensed processor cores in the world.
- The ARM processor core uses a RISC architecture.
- The first ARM RISC processor was produced by the Acorn Group of Computers in the year 1985.
- The ARM processor core is a key component of many successful 32-bit embedded systems due to the benefits, such as low power consumption, reasonable performance, etc.,
- ARM processors are specifically used in portable devices like digital cameras, mobile phones, home networking modules and wireless communication technologies etc.,

INTRODUCTION

ABOUT ARM

- ARM: 32-bit RISC-processor core
- 37 pieces of 32-bit integer registers
- Fast interrupt response
- Simple but powerful instruction set
- small core size and power-efficiency
- Pipelined operation
- Von Neuman or Harvard type bus structure
- High performance and High code density
- 8 / 16 / 32 -bit data types
- 7 modes of operation (USR, SYS, FIQ,IRQ, SVC, ABT, UND)
- Good speed / power consumption ratio

INTRODUCTION

ARM VERSIONS

VERSION 1 (ARMv1)
- 26 bit addressing
- Basic data processing, branch & software interrupt instruction
- No commercial product

VERSION 2 (ARMv2)
- Multiply & multiply-accumulate instructions
- Coprocessor support
- Two new banked register in FIQ mode

VERSION 2 (ARMv2a)
- On-chip cache
- Atomic swap inst.
- Coprocessor 15 for cache management

VERSION 3 (ARMv3)
- 32-bit addressing
- CPSR & SPSR introduced
- New 2 processor modes (abort & undefined)
- MMU support

VERSION 3 (ARMv3M)
- Signed and unsigned long multiply instructions

VERSION 4 (ARMv4)
- Half-word load/store inst.
- New mode—system
- 26-bit addressing mode no longer supported

VERSION 4 (ARMv4T)
- 16-bit Thumb (v1) compressed form of instruction introduced

VERSION 5 (ARMv5TE)
- Superset of 4T (v2) adding new instruction with DSP Enhanced
- Adds software breakpoint instruction

VERSION 5 (ARMv5TEJ)
- Same as v5TE with Jazelle technology support
- It enables execution of Java byte codes

VERSION 6
- Developed for media extension
- Improved multiprocessor inst.
- Support 8/9 stage pipeline

VERSION 7
- Mainly used by cortex family
- It uses 13 stage superscalar pipeline

INTRODUCTION

ARCHITECTURE ARMv7 PROFILES

- Application profile (ARMv7-A)
  - Memory management support (MMU)
  - Highest performance at low power
    - Influenced by multi-tasking OS system requirements
  - TrustZone and Jazelle-RCT for a safe, extensible system
  - e.g. Cortex-A5, Cortex-A9
- Real-time profile (ARMv7-R)
  - Protected memory (MPU)
  - Low latency and predictability 'real-time' needs
  - Evolutionary path for traditional embedded business
  - e.g. Cortex-R4
- Microcontroller profile (ARMv7-M)
  - Lowest gate count entry point
  - Deterministic and predictable behavior a key priority
  - Deeply embedded use
  - e.g. Cortex-M3

ARM Architecture Progress

Core	Architecture
ARM1	v1
ARM2	v2
ARM2as, ARM3	v2a
ARM6, ARM600, ARM610	v3
ARM7, ARM700, ARM710	v3
ARM7TDMI, ARM710T, ARM720T, ARM740T	v4T
StrongARM, ARM8, ARM810	v4
ARM9TDMI, ARM920T, ARM940T	V4T
ARM9E-S, ARM10TDMI, ARM1020E	v5TE
ARM10TDMI, ARM1020E	v5TE
ARM11 MPCore, ARM1136J(F)-S, ARM1176JZ(F)-S	v6
Cortex-A/R/M	v7

ARM DESIGN PHILOSOPHY

ARM DESIGN PHILOSOPHY

The ARM RISC is implemented with four major design rules:

1. Instructions:

- RISC processors have a reduced number of instruction classes. These classes provide simple operations that can each execute in a single cycle.
- Each instruction is a fixed length to allow the pipeline

2. Pipelines:

- The processing of instructions is broken down into smaller units that can be executed in parallel by pipelines.

3. Registers:

- RISC machines have a large general-purpose register set.
- Any register can contain either data or an address.

4. Load-store architecture:

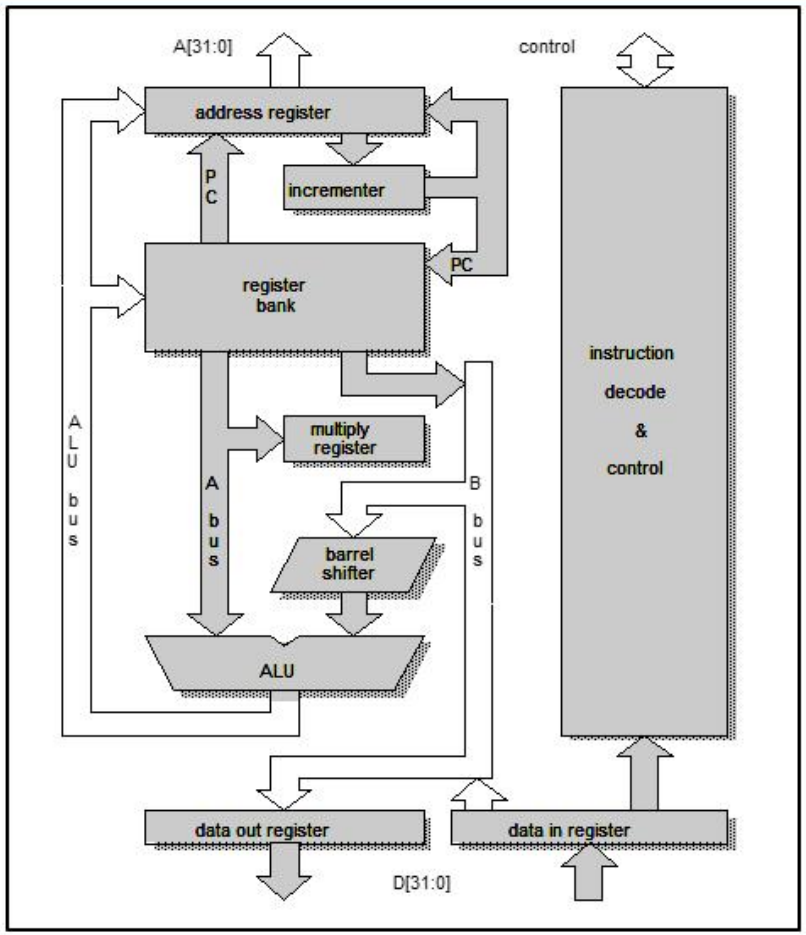
- Separate load and store instructions transfer data between the register bank and external memory.
- Memory accesses are costly, CISC design the data processing operations can act on memory directly.

OVERVIEW OF ARM ARCHITECTURE

OVERVIEW OF ARM ARCHITECTURE

- The ARM processor, like all RISC processors, uses a load-store architecture. This means it has two instruction types for transferring data in and out of the processor.
- Load instructions copy data from memory to registers in the core, and conversely the store instructions copy data from registers to memory.
- There are no data processing instructions that directly manipulate data in memory. Thus, data processing is carried out solely in registers.
- Load and store instructions use the ALU to generate an address to be held in the address register and broadcast on the Address bus.
- Data enters the processor core through the databus and the data may be an instruction to execute or a data item.

OVERVIEW OF ARM ARCHITECTURE



ARM Architecture

OVERVIEW OF ARM ARCHITECTURE

- **Address register:** It store the 32-bit memory address from which the data/instruction to be accessed.
- **Incrementer:** It increments the memory address so as to point to the next instruction when required.
- **Data in and Data out registers:** It is used as a buffer to store the 32-bit data when read/write operation is performed from/into the memory
- **Instruction decoder and control unit:**
  - The instruction decoder translates instructions before they are executed.
  - The operation in the processor is managed and controlled with the help of signals generated to different components in the system from the control unit.

OVERVIEW OF ARM ARCHITECTURE

- **Register bank:**
  - Data items are placed in the register bank - a storage unit made up of 32-bit registers.
  - Also it is used in arithmetic operations, intermediate variable storage, temporary address storage.

- **ALU:**
  - This unit performs the various arithmetic and logical operations on two 32-bits inputs.
  - The primary input comes from the register file using A bus, whereas the other input comes from the barrel shifter using the B bus.
  - After passing through the ALU unit, the result is written back to the register file using the ALU bus.
  - Status registers flags are modified by the ALU outputs.

OVERVIEW OF ARM ARCHITECTURE

- **Barrel shifter:**
  - This is the unique and powerful feature of the ARM processor which has the ability to shift the 32-bit binary pattern in one of the source registers left or right by a specific number of positions before it enters the ALU.
  - This shift increases the power and flexibility of many data processing operations and it is useful for loading constants into a register to achieve fast multiply/division by a power of 2.

- **Multiply register:**
  - This unit performs the operation known as multiply-accumulate (MAC), a fundamental operation in many computing devices, especially in Digital Signal Processing (DSP) application.
  - MAC unit operates in two stages, firstly it computes the product of given numbers and forward the result for the second stage operation i.e. addition/accumulate.
  - Any operation's result can be written back to the register bank, or if the instruction needs memory access, the result is sent to the address register.

STATES
[ARM, THUMB, JAZELLE]

STATES [ARM, THUMB, JAZELLE]

- The state of the ARM core determines which instruction set is being executed. There are three instruction sets: ARM, Thumb, and Jazelle.
- In ARM state the processer executes 32-bit instructions, but in Thumb state the processor is executing purely Thumb 16- bit instructions.
- Jazelle state executes 8-bit instructions and it is a hybrid mix of software and hardware designed to speed up the execution of Java byte codes.
- You cannot intermingle sequential ARM, Thumb, and Jazelle instructions.
- The Jazelle "J" and Thumb "T" bits in the CPSR register reflect the state of the processor.
  - When both J and T bits are 0, the processor is in ARM state
  - When the T bit is 1, then the processor is in Thumb state.

STATES [ARM, THUMB, JAZELLE]

ARM and Thumb instruction set features.

	ARM (*cpsr* $T = 0$)	Thumb (*cpsr* $T = 1$)
Instruction size	32-bit	16-bit
Core instructions	58	30
Conditional execution[a]	most	only branch instructions
Data processing instructions	access to barrel shifter and ALU	separate barrel shifter and ALU instructions
Program status register	read-write in privileged mode	no direct access
Register usage	15 general-purpose registers +*pc*	8 general-purpose registers +7 high registers +*pc*

Jazelle instruction set features.

	Jazelle (*cpsr* $T = 0, J = 1$)
Instruction size	8-bit
Core instructions	Over 60% of the Java bytecodes are implemented in hardware; the rest of the codes are implemented in software.

REGISTERS

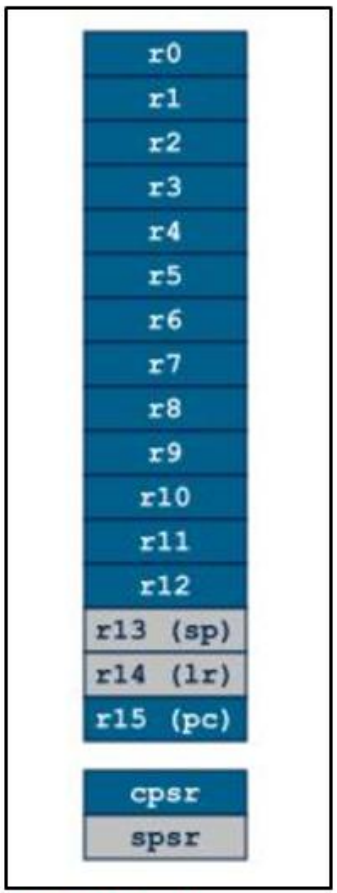
REGISTERS

- In ARM processor, there are up to 18 active registers: 16 data registers (r0 to r15) and 2 program status registers. All the registers are 32 bits in size.
- r0 – r12: Used as General Purpose Registers(GPR) to hold either data or an address.
- r13 – r15 : Used as special function registers
  - r13 is traditionally used as the stack pointer (sp) and stores the head of the stack in the current processor mode.
  - r14 is called the link register (lr) and is where the core puts the return address whenever it calls a subroutine.
  - r15 is the program counter (pc) and contains the address of the next instruction to be fetched by the processor.
- Program Status Registers:
  1. Current Program Status Register (cpsr)
  2. Saved Program Status Register(spsr)

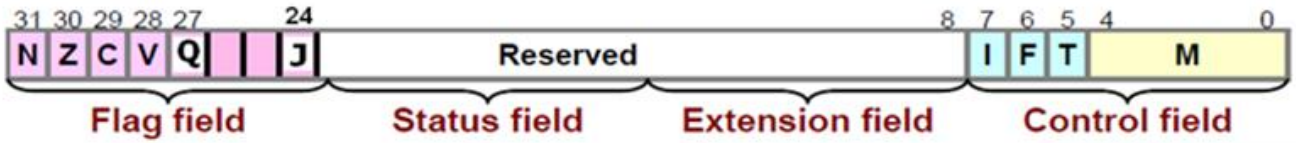
REGISTERS

- **CPSR:** The ARM core uses the CPSR 32-bit register to monitor and control internal operations.
  - The CPSR is divided into four fields, each 8 bits wide:
    - Control field (bit-0 to 7) : contains the processor mode, state, and interrupt mask bits.
    - Extension field (bit-8 to 15): reserved for future use
    - Status field (bit-16 to 23): reserved for future use
    - Flag field (bit-24 to 31): contains the condition flags
  - Some ARM processor cores have extra bits allocated. For example, the J bit, which can be found in the flags field, is only available on Jazelle-enabled processors.
- **SPSR:** It is used to store the current value of the CPSR when an exception(Interrupt) is taken so that it can be restored after handling the exception.
  - It is useful to examine the value that the CPSR had when the exception was taken, for example to determine the instruction set state and privilege level etc.,
  - Each exception handling mode can access its own SPSR but, user mode and System mode do not have an SPSR because they are not exception handling modes.

REGISTERS



ARM Register set



Condition Code Flags	
N	Negative result from ALU
Z	Zero result from ALU
C	ALU operation caused Carry
V	ALU operation oVerflowed
Q	Sticky Overflow
J	Jazelle state bit

Control bits	
I	1: disables IRQ
F	1: disables FIQ
T	1: Thumb, 0: ARM

Mode bits	
M[4:0]	Mode
0b10000	User
0b11111	System
0b10001	FIQ
0b10010	IRQ
0b10011	Supervisor
0b10111	Abort
0b11011	Undefined

Current Program Status Register (CPSR)

MODES

MODES

- The processor mode determines which registers are active and the access rights to the cpsr register itself.
- Each mode has access to its own stack space and a different subset of registers
- Each processor mode is either privileged or non-privileged:
  - A privileged mode allows full read-write access to the CPSR
  - A nonprivileged mode only allows read access to the control field in the CPDR but still allows read-write access to the condition flags.
- There are seven processor modes in total:
  - Six privileged modes (abort, fast interrupt request, interrupt request, supervisor, system, undefined)
  - One non-privileged mode (user).

MODES

- User mode is used for programs and applications.
- System mode is a special version of user mode that allows full read-write access to the cpsr.
- The processor enters abort mode when there is a failed attempt to access memory.
- Fast interrupt request and interrupt request modes correspond to the two interrupt levels available on the ARM processor.
- Supervisor mode is the mode that the processor is in after reset and is generally the mode that an operating system kernel operates in.
- Undefined mode is used when the processor encounters an instruction that is undefined or not supported by the implementation.

MODES

Exception modes

Mode	Description	
Supervisor (SVC)	Entered on reset and when a Supervisor call instruction (SVC) is executed	**Privileged modes**
FIQ	Entered when a high priority (fast) interrupt is raised	
IRQ	Entered when a normal priority interrupt is raised	
Abort	Used to handle memory access violations	
Undef	Used to handle undefined instructions	
System	Privileged mode using the same registers as User mode	
User	Mode under which most Applications / OS tasks run	**Unprivileged mode**

THANK YOU

By
prakash v