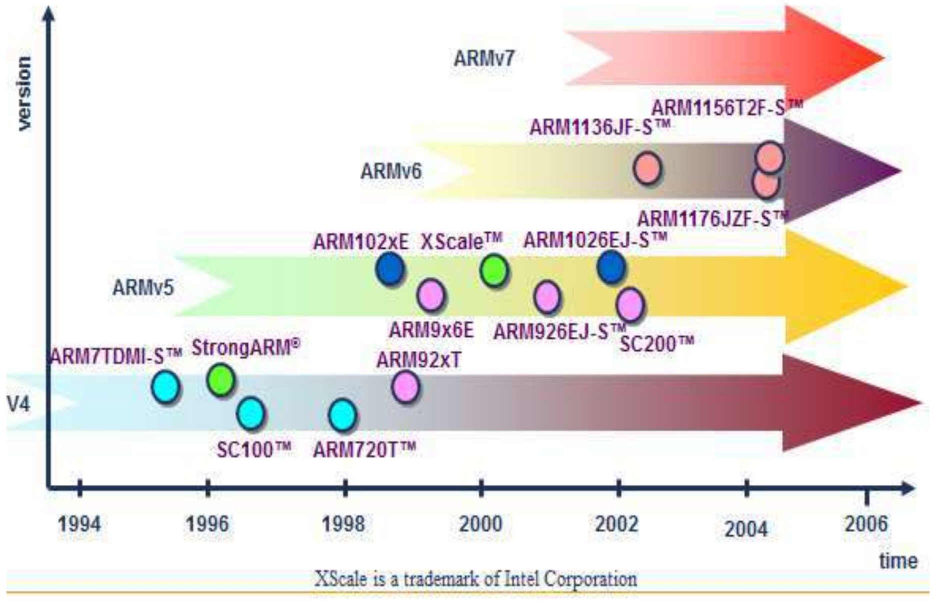
What is ARM?

- Advanced RISC Machine
- First RISC microprocessor for commercial use
- Market-leader for low-power and cost-sensitive embedded applications

The History of ARM

- Developed at Acorn Computers Limited, of Cambridge, England , between 1983 and 1985
- Problems with CISC:
    - Slower memory parts
    - Clock cycles per instruction



Used in:

- Handheld devices
- High end applications involving complex computation
- Robotics
- Automation system
- Consumer electronics

Fundamentals of ARM :

- Load/store architecture
- An orthogonal instruction set.
- Mostly single-cycle execution.
- Enhanced power-saving design.
- 64 and 32-bit execution states for scalable high performance.
- Hardware virtualization support.

Features of ARM :

- High performance, low power, small in size (ideal for embedded sys)
- Large Register File, Small instruction set, Load-Store instructions
- Fixed length instructions, Conditional execution of instructions
- High code density, most instructions executable in single cycle
- 32-bit in-line barrel shifter, built-in circuit for hardware debugging
- DSP enhanced instructions, Jazelle (Java byte code extn. 3rd state)

ARM Nomenclature

A R M {x}{ y}{ z} T D M I E J F S (Example: ARM7-TDMI-S)

ARM	Advanced RISC Machine
x	Series
y	MMU (No. of Memory Management units present)
z	Cache Memory (in terms of KB)
T	Thumb instructions Support
D	Debugger (Debugging via JTAG interface)
M	Multiplier
I	In-Circuit Emulator (ICE) macrocell
E	Enhanced Instructions for DSP related applications
J	Jazelle instructions support for JAVA Codes execution
F	Floating-point unit
S	Synthesizable version

Eg. ARM7TDMI
ARM926EJ-S
ARM1136J(F)-S

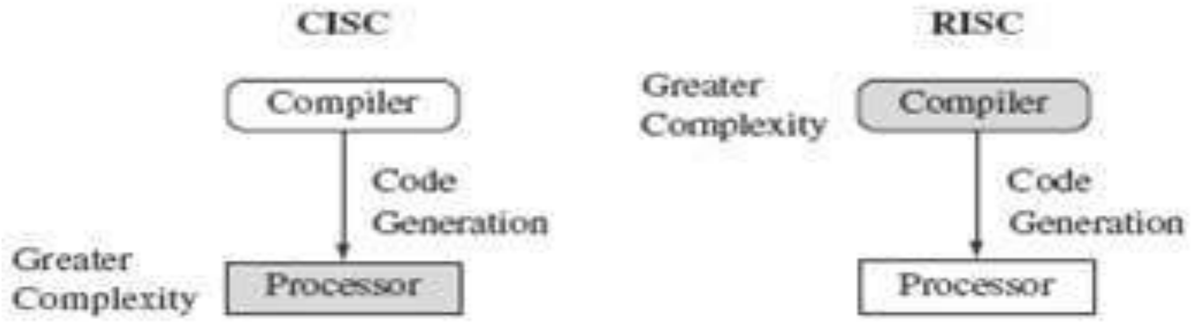
ARM & RISC DESIGN PHILOSOPHY

RISC Processors:

- It is a design philosophy aimed at delivering simple but powerful instruction set that executes within a single cycle at high clock speed.

- CISC and RISC differ in complexities of their instruction sets where CISC is more complex than RISC.

- The smaller instruction set allows a designer to implement a hardwired control unit which runs at a higher clock rate than its equivalent micro sequenced control unit.



RISC	CISC
Simple instruction taking one cycle.	Complex instruction may take one or more clock cycles.
Large register file	Few registers to store data.
Fewer instructions to access memory.	More instructions to access memory
Few addressing modes.	More addressing modes
Instruction Decoder is simple. Hardwired logic is used for the decoder.	The instruction decoder is complex. A decoder using ROM which consists microcode.
Supports pipelining. i.e. overlapping of fetch, decode, execute takes place.	Does not support pipelining.
Fixed instruction size.	Variable instruction size.
Core takes less chip area so more space for cache, MMU.	More chip area is taken by core CPU.
Complexity in software. Compiler design is difficult	Complexity in Hardware. Emphasis is on hardware
Higher clock rates. So faster.	Lower clock rates. So, comparatively slower.
Cache memory is present.	Cache memory is absent or unified cache is present

RISC Philosophy (Four major Rules)

1. Instructions:Less no of instruction classes to provide simple operations that can execute in a single cycle, each instruction is a fixed length to allow the pipeline to fetch future instructions before decoding the current instruction. (Unlike CISC)

2. Pipelines:The processing of instructions is broken down into smaller units that can be executed in parallel by pipelines, instructions can be decoded in one pipeline stage.

3. Registers:RISC machines have a large general-purpose register set & any register can contain either data or an address.

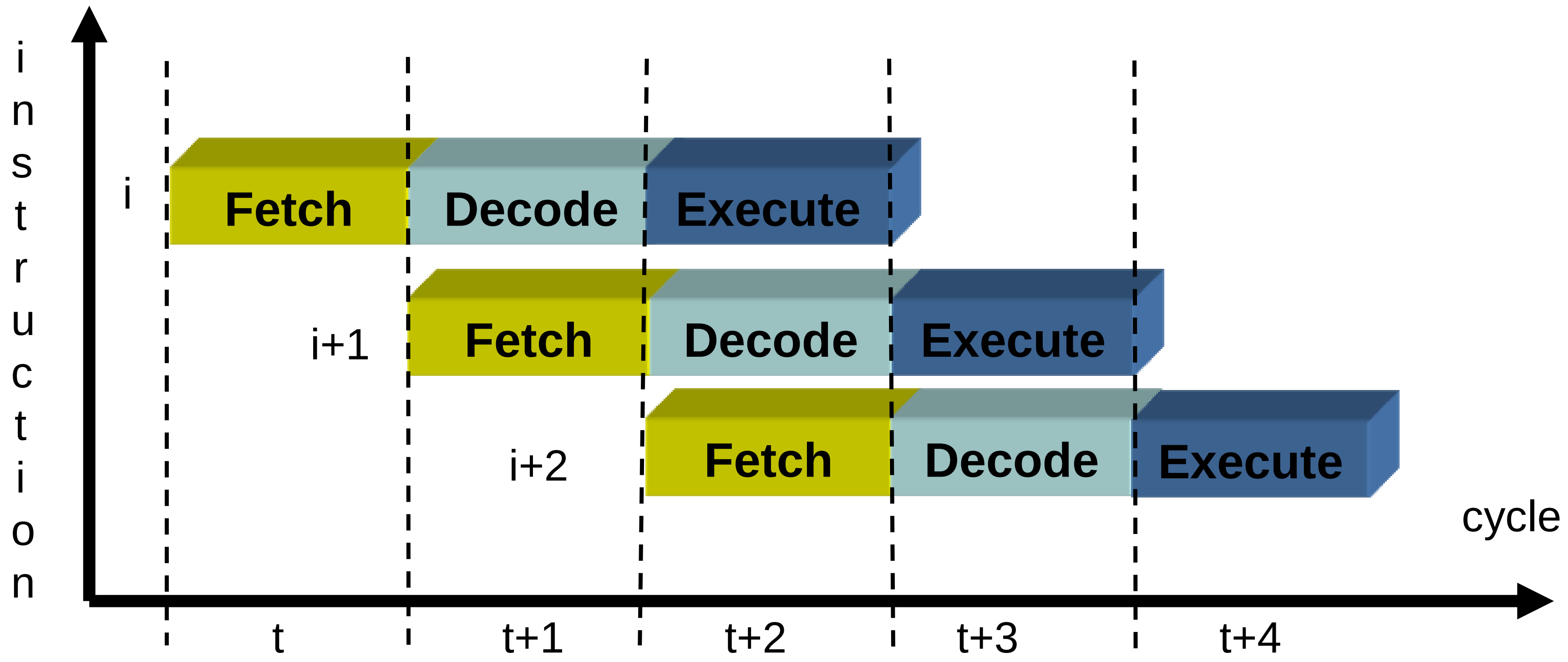
4. Load-Store Architecture: Processor operates on data held in registers. Separate load and store instructions: transfer data between the register bank and external memory. Because memory accesses are costly.

Pipeline Organization

- Increases speed – most instructions executed in single cycle
- Versions:
  - 3-stage (ARM7TDMI and earlier)
  - 5-stage (ARMS, ARM9TDMI)
  - 6-stage (ARM10TDMI)

Pipeline Organization

- 3-stage pipeline: Fetch – Decode - Execute
- Three-cycle latency, one instruction per cycle throughput

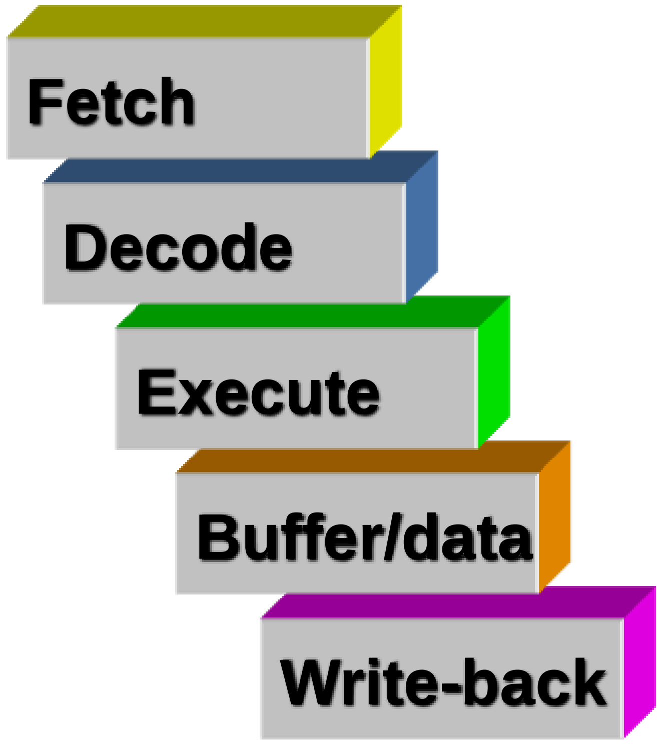


Pipeline Organization

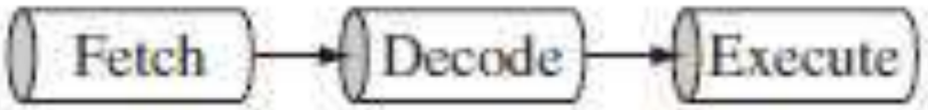
5-stage pipeline:

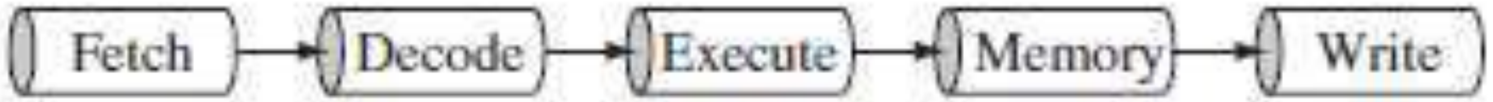
- Reduces work per cycle => allows higher clock frequency
- Separates data and instruction memory => reduction of CPI (average number of clock Cycles Per Instruction)

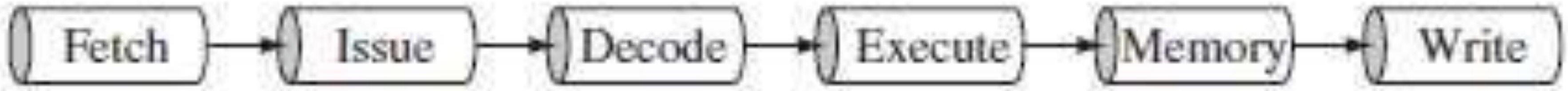
- Stages:



Pipeline Organization







ARM7 FUNDAMENTALS

1. All ARM instructions are 32-bit long & stored word aligned.
2. ARM processor like all RISC processors is a Load Store architecture, Von-Neuman Architecture (same program + data memory).
3. ARM has two special instructions types for transferring data in & data out of processor.
   1. Load Instruction = Copy data from memory to registers in the core.

      (Registers in the processor core <----Memory)
   2. Store Instruction = Copy data from registers to memory
      1. (Registers in processor core ----> Memory)
4. There are no data processing instructions that are directly manipulate data in memory (Hence Data processing is carried out only in registers).
5. ARM core is a 32-bit bit processor most instructions treat the registers ad holding signed or unsigned 32-bit value.
6. Data Types

Word – 32-bit, Halfword – 16-bit, Byte – 8-bit

ARM7 Features:

- 3 stage Pipeline (Fetch, Decode, Execute)
- Operating frequency: 80 MHz
- Power Consumption: 0.06 mW/MHz.
- MIPS is 0.97
- Available in Von-Neumann Architecture.
- Supports both 16- & 32-bit instruction set.

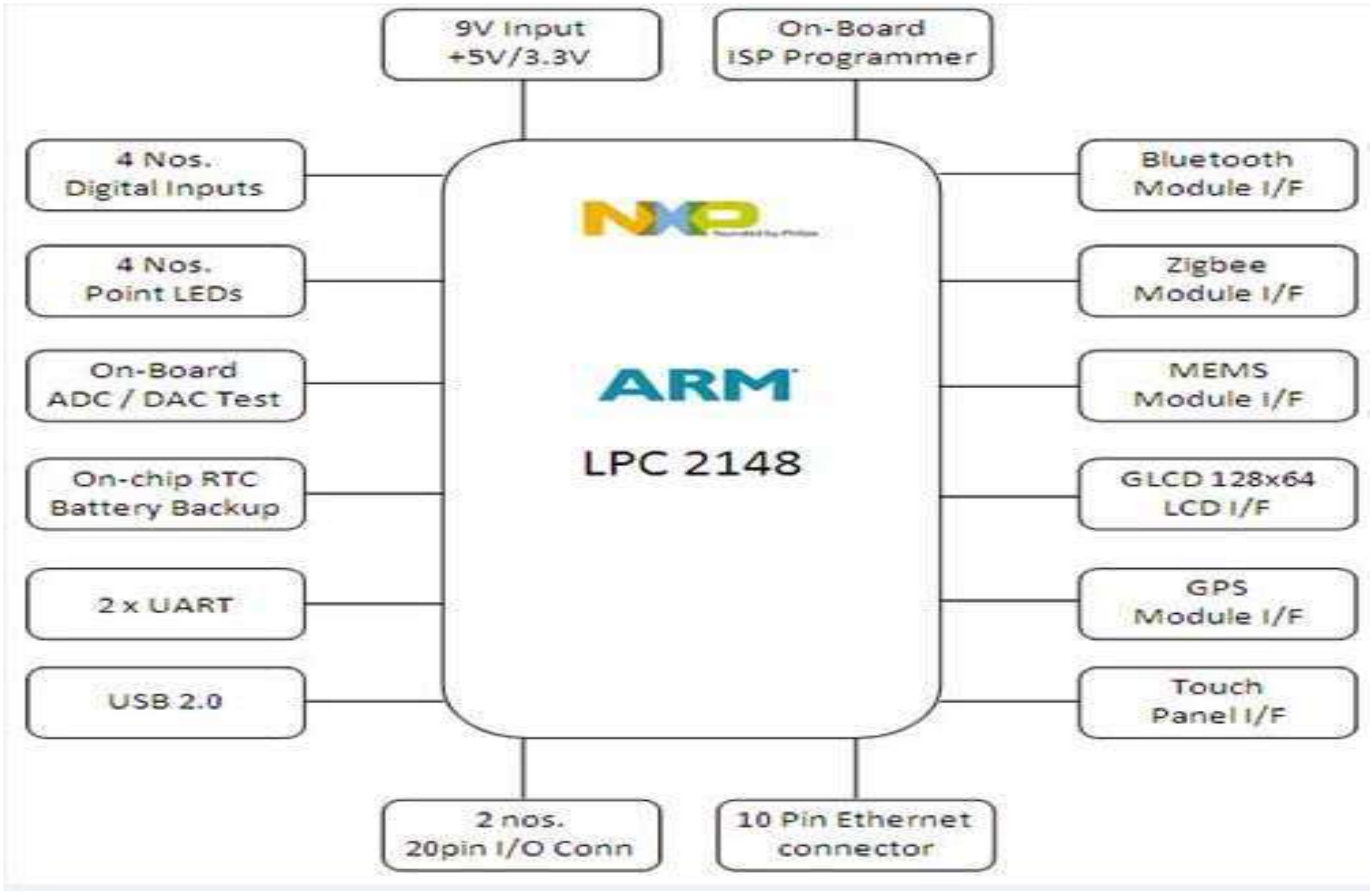
ARM Architecture

- Typical RISC architecture:
  - Large uniform register file
  - Load/store architecture
  - Simple addressing modes
  - Uniform and fixed-length instruction fields
- Enhancements:
  - Each instruction controls the ALU and shifter
  - Auto-increment and auto-decrement addressing modes
  - Multiple Load/Store
  - Conditional execution

ARM Architecture

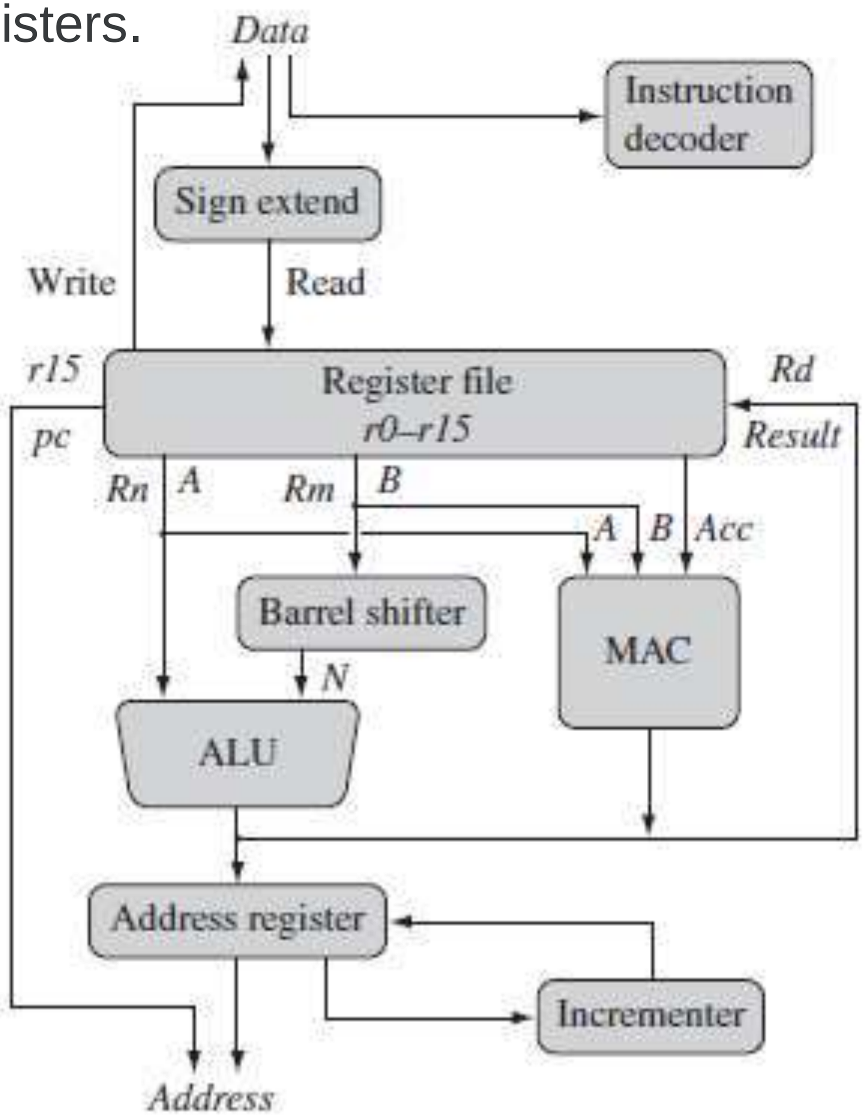
- Current low-end ARM core for applications like digital mobile phones
- TDMI
  - **T**: Thumb, 16-bit instruction set
  - **D**: on-chip Debug support, enabling the processor to halt in response to a debug request
  - **M**: enhanced Multiplier, yield a full 64-bit result, high performance
  - **I**: Embedded ICE hardware
- Von Neumann architecture
- 3-stage pipeline

ARM architecture



Date Flow Model

When an instruction is decoded inside the ARM core and how a particular instruction is executed by interacting with the internal registers file and then send result out of the registers.



Data Sizes and Instruction Sets

- The ARM is a 32-bit architecture.
- When used in relation to the ARM:
  - **Byte** means 8 bits
  - **Half word** means 16 bits (two bytes)
  - **Word** means 32 bits (four bytes)
- Most ARM's implement two instruction sets
  - 32-bit ARM Instruction Set
  - 16-bit Thumb Instruction Set
- Jazelle cores can also execute Java byte code

ARM Registers

- 31 general-purpose 32-bit registers
- Others speed up the exception process
- 16 Data registers (R0-R15) & one status register (CPSR)
- R0 to R13 are orthogonal general-purpose register.
- Orthogonal means, any instruction that you can apply to ro can equally be applied to any of the other register.

  Eg.

  ```
  ADD ro, r1, r2
  ADD r5, r6, r7
  ```

- R13 (stack pointer) and stores the top of the stack in the current processor mode.
- R14(LR) Link Register where the core puts the return address on executing a subroutine.
- R15(PC) Program counter stores the address of next instruction to be executed.
- In ARM state all ARM instruction are 32-bits wide.
- In Thumb state all instructions are 16-bit wide.

Operating Modes

- Seven operating modes:
  - User (Non-Privileged): Only allows read access to the control field in CPSR
  - Privileged: Allow full read and write access of CPSR
    - System (version 4 and above)
    - FIQ
    - IRQ
    - Abort
    - Undefined
    - Supervisor

exception modes

Operating Modes

<u>User mode</u>

- Normal program execution mode
- System resources unavailable
- Mode changed by exception only

<u>Exception mode</u>

- Entered upon exception
- Full access to system resources
- Mode changed freely

Processor Modes

The ARM has seven basic operating modes:

- **User** : unprivileged mode under which most tasks run
- **FIQ** : entered when a high priority (fast) interrupt is raised
- **IRQ** : entered when a low priority (normal) interrupt is raised
- **Supervisor** : entered on reset and when a Software Interrupt instruction is executed
- **Abort** : used to handle memory access violations
- **Undef** : used to handle undefined instructions
- **System** : privileged mode using the same registers as user mode

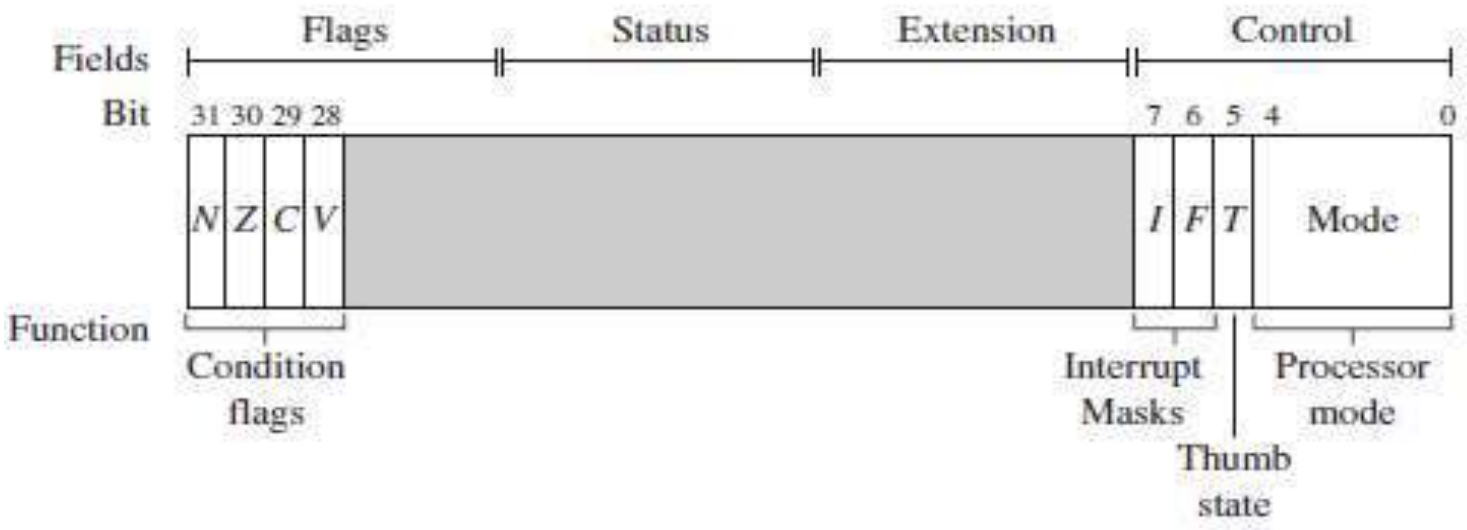
ARM7 Programmer’s Model or Register Model

System & User	FIQ	Supervisor	Abort	IRQ	Undefined
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_fiq	R7	R7	R7	R7
R8	R8_fiq	R8	R8	R8	R8
R9	R9_fiq	R9	R9	R9	R9
R10	R10_fiq	R10	R10	R10	R10
R11	R11_fiq	R11	R11	R11	R11
R12	R12_fiq	R12	R12	R12	R12
R13	R13_fiq	R13_svc	R13_abt	R13_irq	R13_und
R14	R14_fiq	R14_svc	R14_abt	R14_irq	R14_und
R15 (PC)	R15 (PC)	R15 (PC)	R15 (PC)	R15 (PC)	R15 (PC)
CPSR	CPSR	CPSR	CPSR	CPSR	CPSR
	SPSR_fiq	SPSR_svc	SPSR_abt	SPSR_irq	SPSR_und

- In total 17(Visible)+20(Banked Rrgisters)=37
- The active registers available in the user mode are shown below.
- This is protected mode which is normally used while executing applications.
- 16 Data registers & one status register
- r0 to r13 are **orthogonal general purpose register.**
- Orthogonal means, any instruction that you can apply to ro can equally be applied to any of the other register.
  - Eg. ADD ro, r1, r2
  - ADD r5, r6, r7
- **R13 (stack pointer)** and stores the top of the stack in the current processor mode.
- **R14(LR) Link Register** where the core puts the return address on executing a subroutine.
- **R15(PC) Program counter** stores the address of next instruction to be executed.
- In ARM state all ARM instruction are 32-bits wide.
- In Thumb state all instructions are 16-bit wide.
- In ARM state Instruction have to be four byte aligned in the memory. Which implies that the bottom two bits of the PC are always zero(Memory location 1000H,1004,1008H).

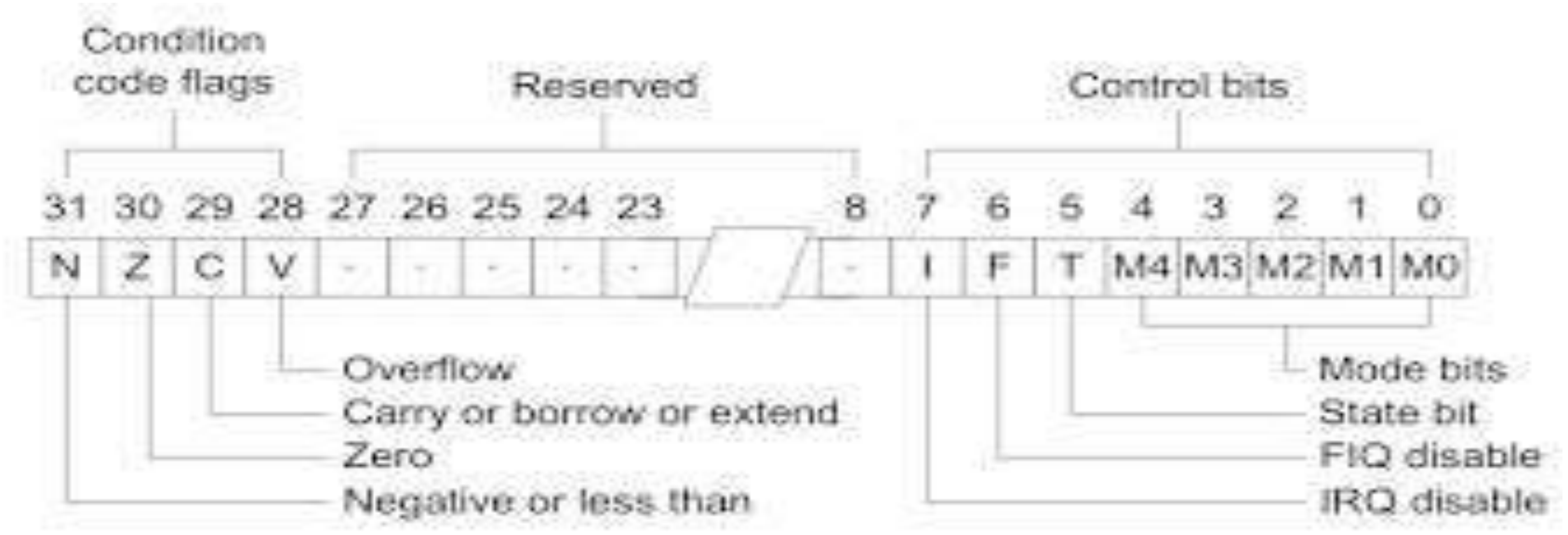
CPSR
CURRENT PROCESSOR STATUS REGISTER

•ARM core uses CPSR to monitor & control internal operations.



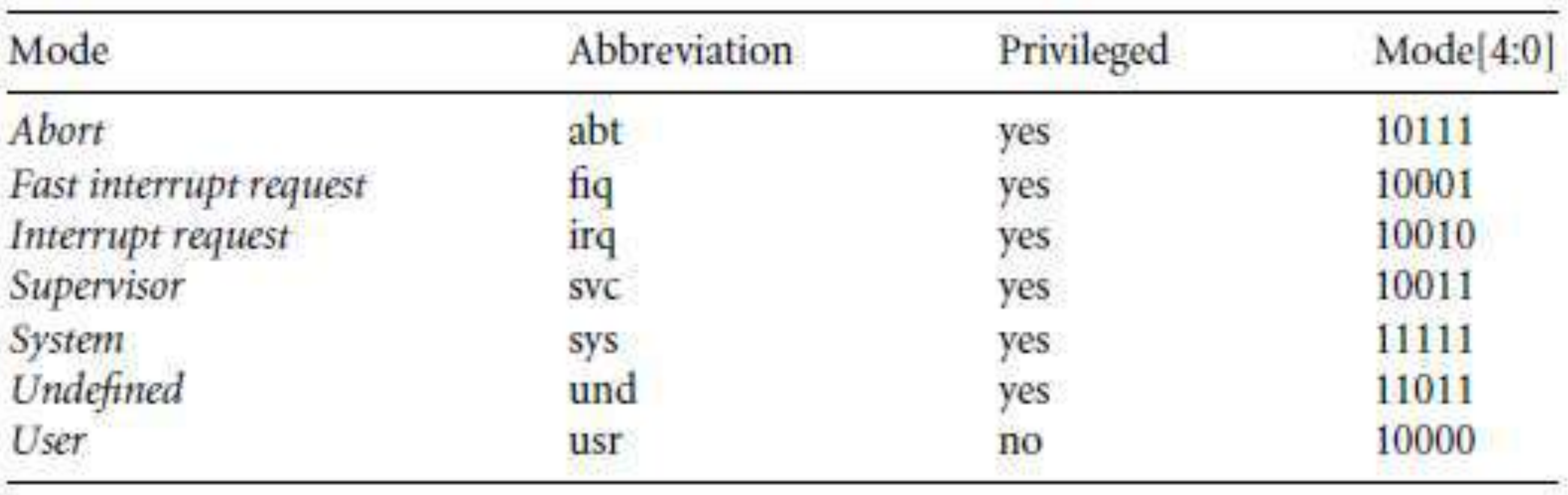
•CPSR fields is divided in to four fields,
each 8-bits wide: flags, status, extension, and control.

•In some ARM processor cores have extra bits allocated J bit (available only on Jazelle enabled processing which execute 8-bit instructions).



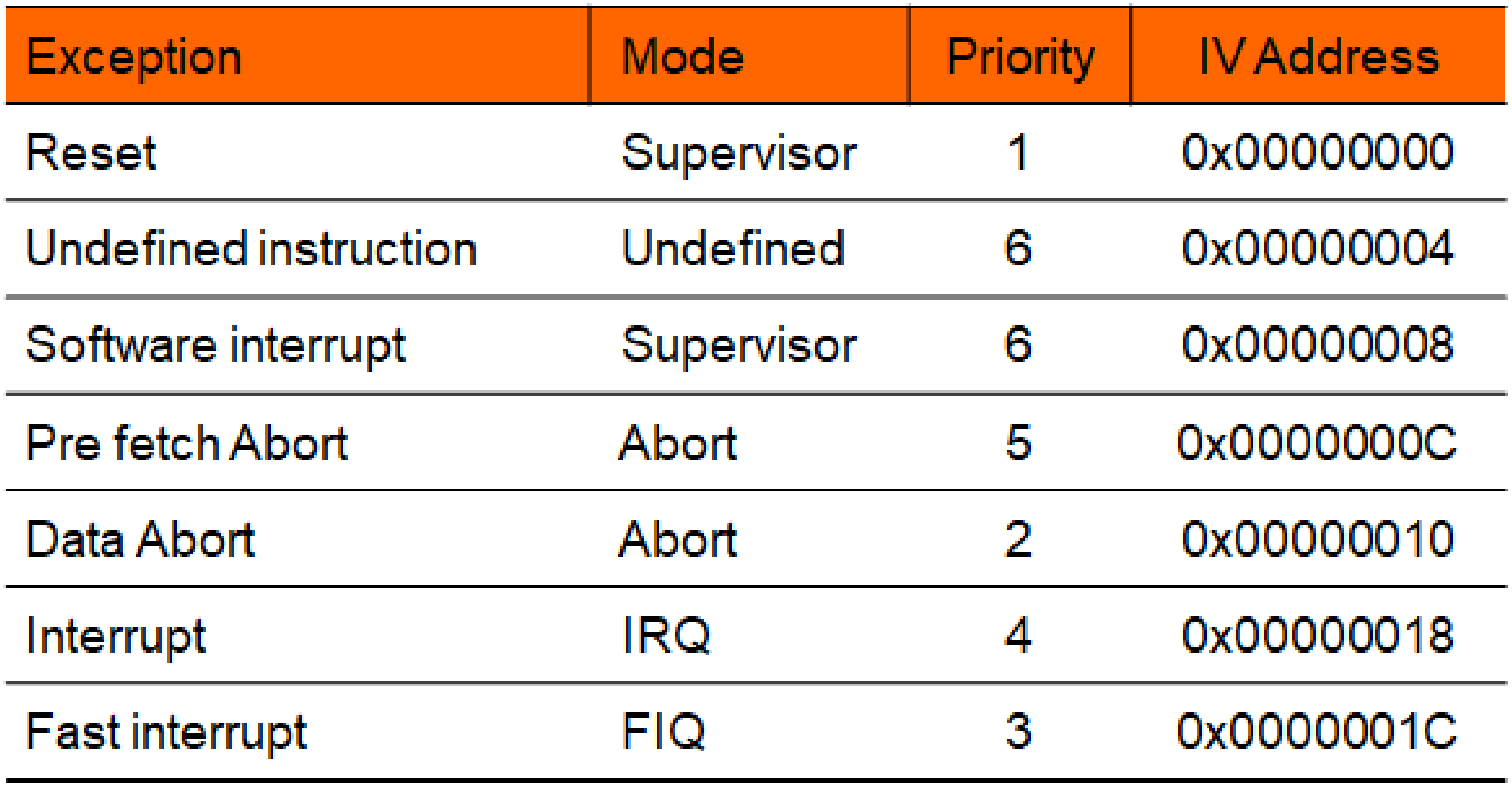
Flag bit	**Sets when**
N- Negative	In case of signed no. operations If result MSB=1 ;Indicates the result of operation is NEGATIVE
Z- Zero	The result of operation is zero
C- Carry	The result causes an unsigned carry(carry out of MSB)
V-Overflow	The result causes a signed overflow
Q- Saturation	The result causes an overflow or saturation
I- Interrupt request Disable	If set interrupt request channel is disabled
F- Fast interrupt request Disable	If set fast interrupt request channel is disabled
J- Jazelle instruction set	If set processor will execute Jazelle instructions
T-Thumb instruction set	If set processor will execute Thumb Instruction set

PROCESSOR MODES



Mode	Abbreviation	Privileged	Mode[4:0]
Abort	abt	yes	10111
Fast interrupt request	fiq	yes	10001
Interrupt request	irq	yes	10010
Supervisor	svc	yes	10011
System	sys	yes	11111
Undefined	und	yes	11011
User	usr	no	10000

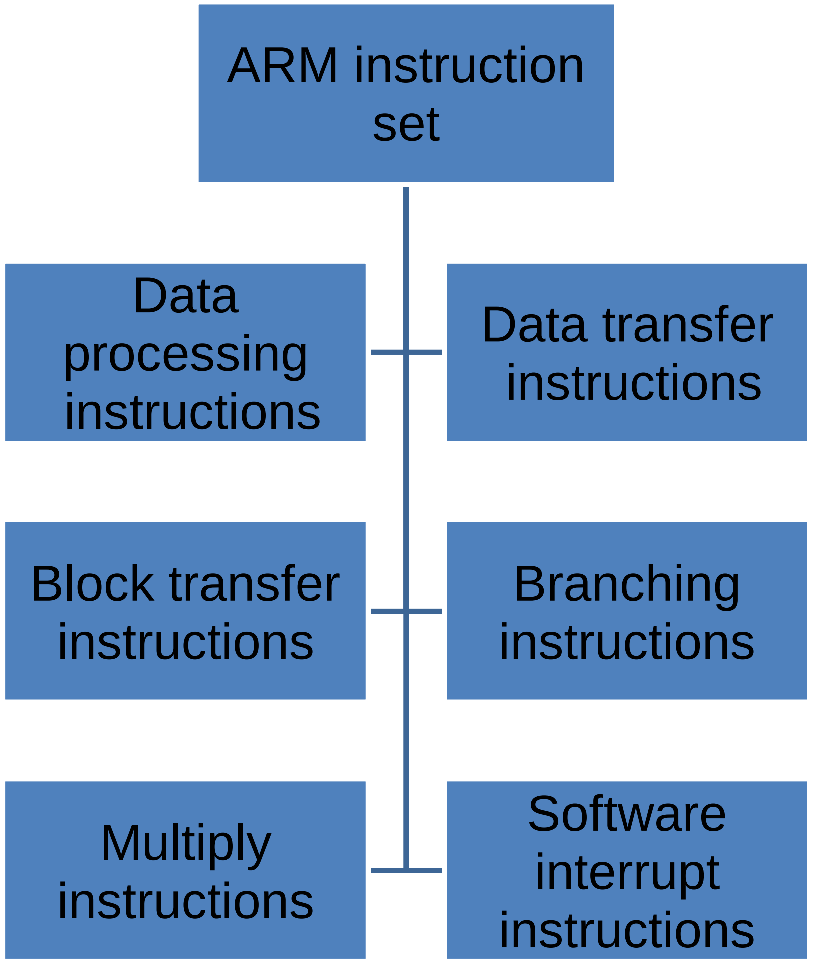
Exceptions



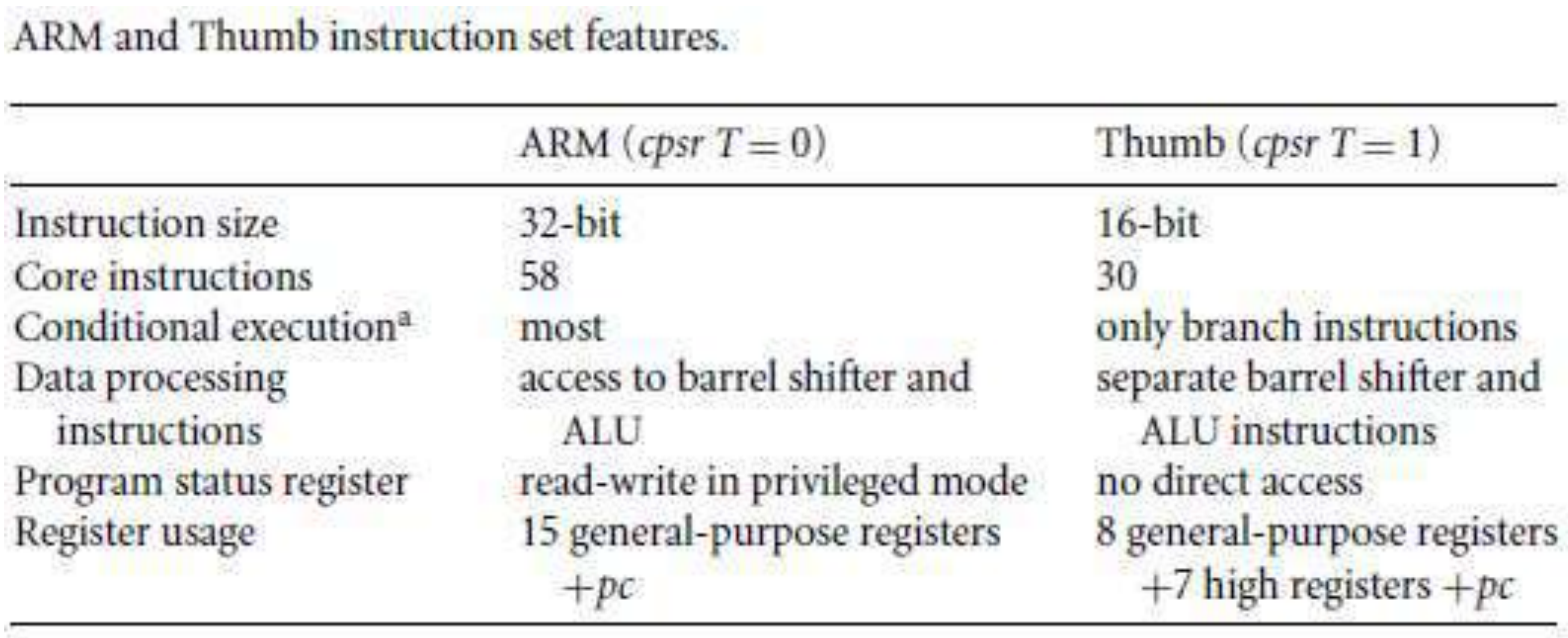
Exception	Mode	Priority	IV Address
Reset	Supervisor	1	0x00000000
Undefined instruction	Undefined	6	0x00000004
Software interrupt	Supervisor	6	0x00000008
Pre fetch Abort	Abort	5	0x0000000C
Data Abort	Abort	2	0x00000010
Interrupt	IRQ	4	0x00000018
Fast interrupt	FIQ	3	0x0000001C

Exception types, sorted by Interrupt Vector addresses

ARM Instruction Set


ARM instruction set
Data processing instructions
Data transfer instructions
Block transfer instructions
Branching instructions
Multiply instructions
Software interrupt instructions

ARM vs THUMB MODE



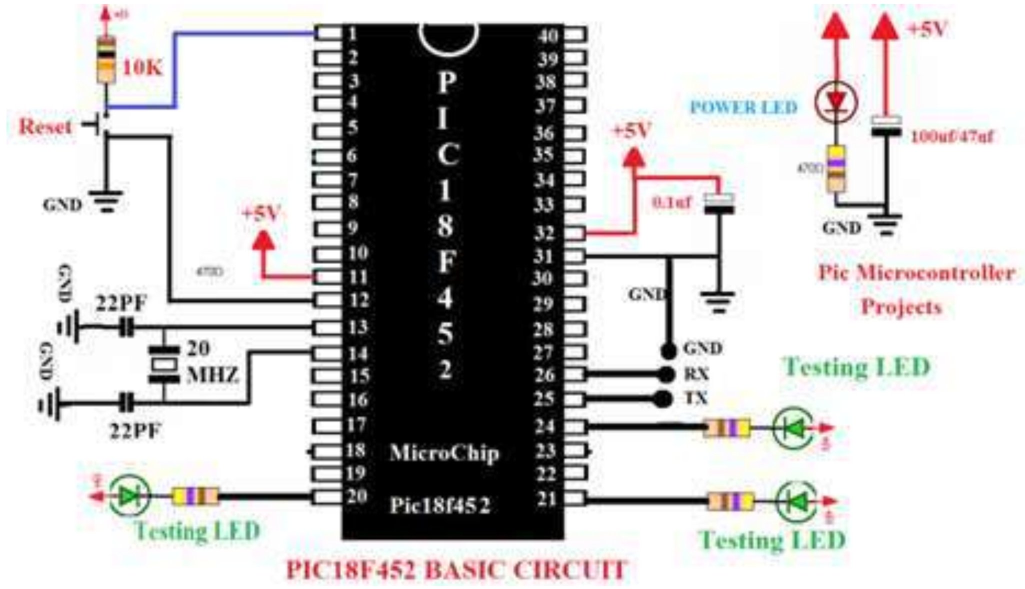
ARM and Thumb instruction set features.

	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*

ARM – QUESTION BANK

- ARM- features/fundamentals.
- ARM & RISC DESIGN PHILOSOPHY
- RISC VS CISC
- PIPELINE
- ARM - DATA FLOW MODEL
- ARM -REGISTERS
- ARM - PROCESSOR MODES
- ARM7 Programmer’s Model or Register Model
- CURRENT PROCESSOR STATUS REGISTER (CPSR)
- ARM vs THUMB MODE
- Describe about ARM Processors & mention the Popular ARM Architectures in detail?

PIC (Peripheral Interface Controller) Microcontrollers



PIC Microcontroller

- PIC microcontroller was developed in the year 1993 by microchip technology.
- The term PIC stands for Peripheral Interface Controller. Initially this was developed for supporting PDP(programmed data processor) computers to control its peripheral devices, and therefore, named as a peripheral interface device.

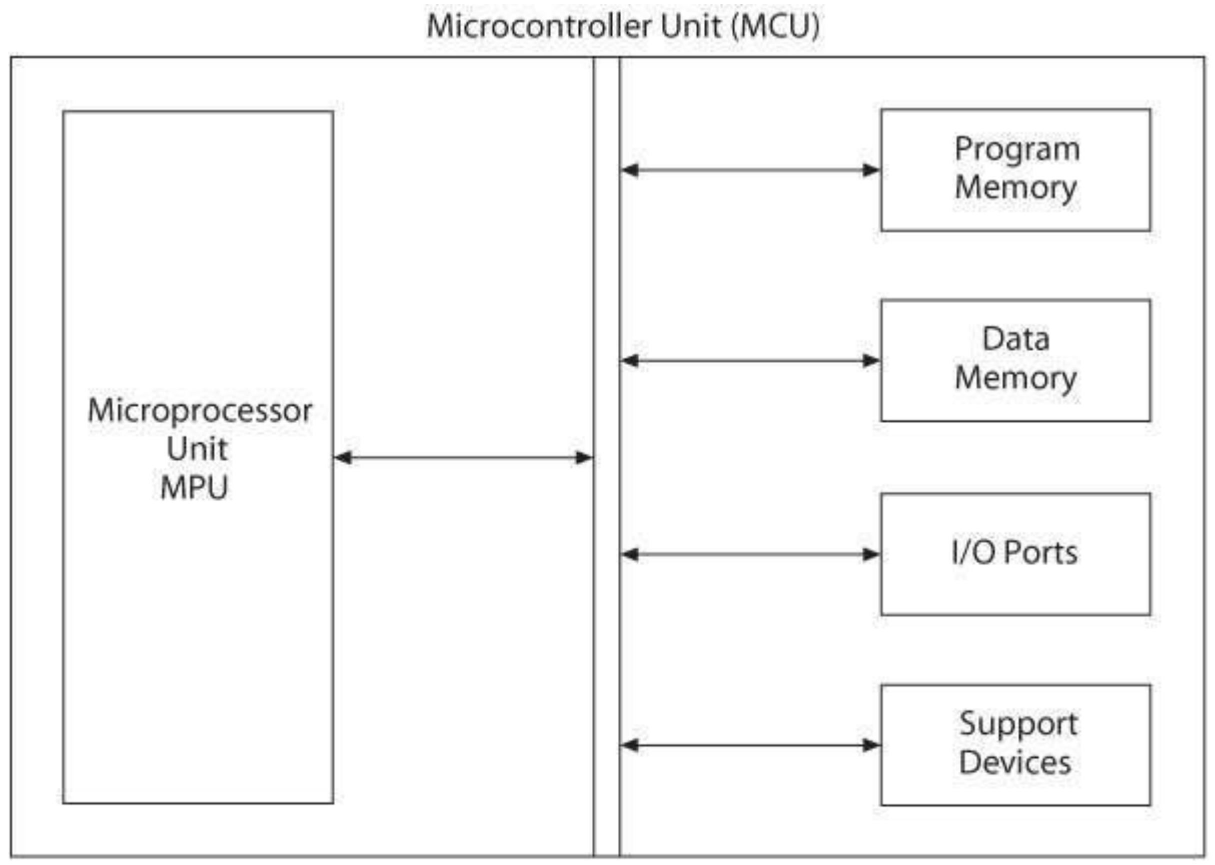
Features of the PIC18 microcontroller

- - 8-bit CPU
- - 2 MB program memory space
- - 256 bytes to 1KB of data EEPROM
- - Up to 3968 bytes of on-chip SRAM
- - 4 KB to 128KB flash program memory
- - Sophisticated timer functions that include: input capture, output compare,
- PWM, real-time interrupt, and watchdog timer
- - Serial communication interfaces: SCI, SPI, I2C, and CAN
- - Background debug mode (BDM)
- - 10-bit A/D converter
- - Memory protection capability
- - Instruction pipelining
- - Operates at up to 40 MHz crystal oscillator

Architecture of PIC18F Microcontrollers

- Microcontroller Unit (MCU)
  - Microprocessor unit (MPU)
  - Harvard Architecture
    - Program memory for instructions
    - Data memory for data
  - I/O ports
  - Support devices such as timers

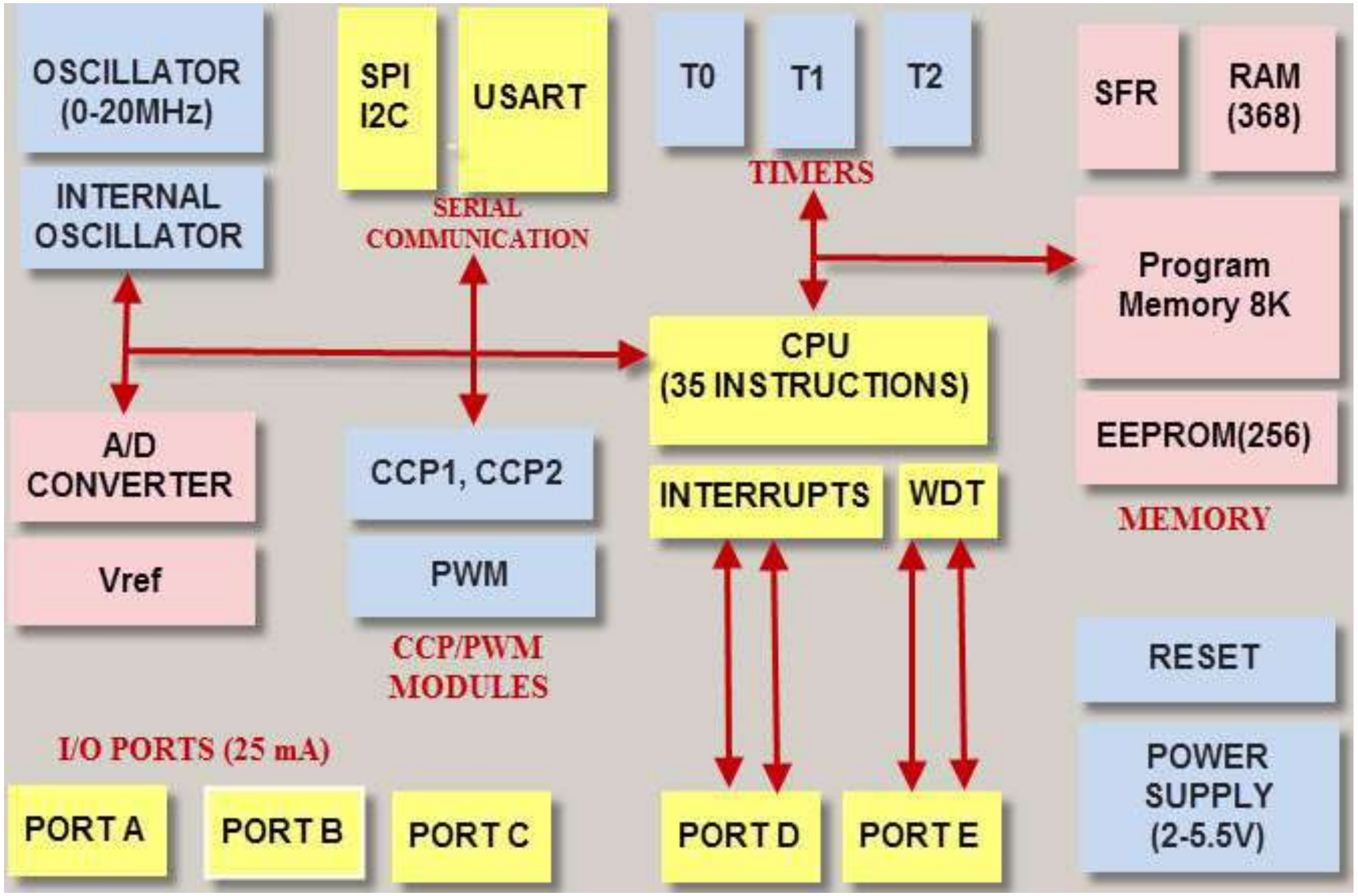
Microcontroller with the Harvard Architecture



Difference between Von Neumann and Harvard Architecture :

VON NEUMANN ARCHITECTURE	HARVARD ARCHITECTURE
It is ancient computer architecture based on stored program computer concept.	It is modern computer architecture based on Harvard Mark I relay based model.
Same physical memory address is used for instructions and data.	Separate physical memory address is used for instructions and data.
There is common bus for data and instruction transfer.	Separate buses are used for transferring data and instruction.
Two clock cycles are required to execute single instruction.	An instruction is executed in a single cycle.
It is cheaper in cost.	It is costly than Von Neumann Architecture.
CPU can not access instructions and read/write at the same time.	CPU can access instructions and read/write at the same time.
It is used in personal computers and small computers.	It is used in micro controllers and signal processing.

Architecture of PIC Microcontroller



Microprocessor Unit

- Includes Arithmetic Logic Unit (ALU), Registers, and Control Unit
  - Arithmetic Logic Unit (ALU)
    - Instruction decoder
      - 16-bit instructions
    - Status register that stores flags
      - 5-bits
    - WREG – working register
      - 8-bit accumulator

Microprocessor Unit

- Registers
  - Program Counter (PC)
    - 21-bit register that holds the Program Memory address
  - Bank Select Register (BSR)
    - 4-bit register used in direct addressing the Data Memory
  - File Select Registers (FSRs)
    - 12-bit registers used as memory pointers in indirect addressing Data Memory
- Control unit
  - Provides timing and control signals
    - Read and Write operations

PIC18F - Address Buses

- Address bus
  - 21-bit address bus for Program Memory
    - Addressing capacity: 2 MB
  - 12-bit address bus for Data Memory
    - Addressing capacity: 4 KB

Data Bus and Control Signals

- Data bus
  - 16-bit instruction/data bus for Program Memory
  - 8-bit data bus for Data Memory
- Control signals
  - Read and Write

The PIC18 Memory Organization

- Data Memory and Program Memory are separated
- - Separation of data memory and program memory makes possible the
- simultaneous access of data and instruction.
- - Data memory are used as general-purpose registers or special function registers
- - On-chip Data EEPROM are provided in some PIC18 MCUs

PIC18F452/4520 Memory

- Program Memory: 32 K
  - Address range: 000000 to $007FFF_H$
- Data Memory: 4 K
  - Address range: 000 to FFF_H
- Data EEPROM
  - Not part of the data memory space
  - Addressed through special function registers

PIC MICROCONTROLLER – QUESTION BANK

- PIC- features/fundamentals.
- VONNEUMANN & HARVARD ARCHITECTURE
- PIC SPECIFICATIONS (MEMORY, SIGNALS, PROCESSOR UNITS etc;)