What is ARM?

• ARM is a CPU architecture (A family of related CPU architectures).

• If this CPU is put on the chip at by itself, then microprocessor

• If you combine it with ROM, RAM and peripherals on one chip, then microcontroller

• First ARM processor was developed in the year 1978 by Cambridge University. •Project taken up in 1983, to replace 8-bit 6502 microprocessor from BBC computers. First ARM RISC processor was produced by the Acorn Group of Computers in the year 1985. • In 1990, new company - ARM was formed, which was jointed owned by Acorn, Apple and VLSI • RISC - Reduced Instruction Set Computer (Intel used a CISC - Complex Instruction Set Computing) • RISC - use only one cycle to execute a command, reduces functions.

Why ARM ?

• One of the most licensed and thus widespread processor cores in the world – Used in cell phones, multimedia players, handheld game console digital TV and cameras , digital TV and cameras – ARM7: GBA, iPod – ARM9: NDS PSP Sony Ericsson BenQ , PSP, Sony Ericsson, BenQ – ARM11: Apple iPhone, Nokia N93, N800 – 90% of 32-bit embedded RISC processors till 2010 • Used especially in portable or battery-operated devices due to its low power consumption and reasonable performance 7.

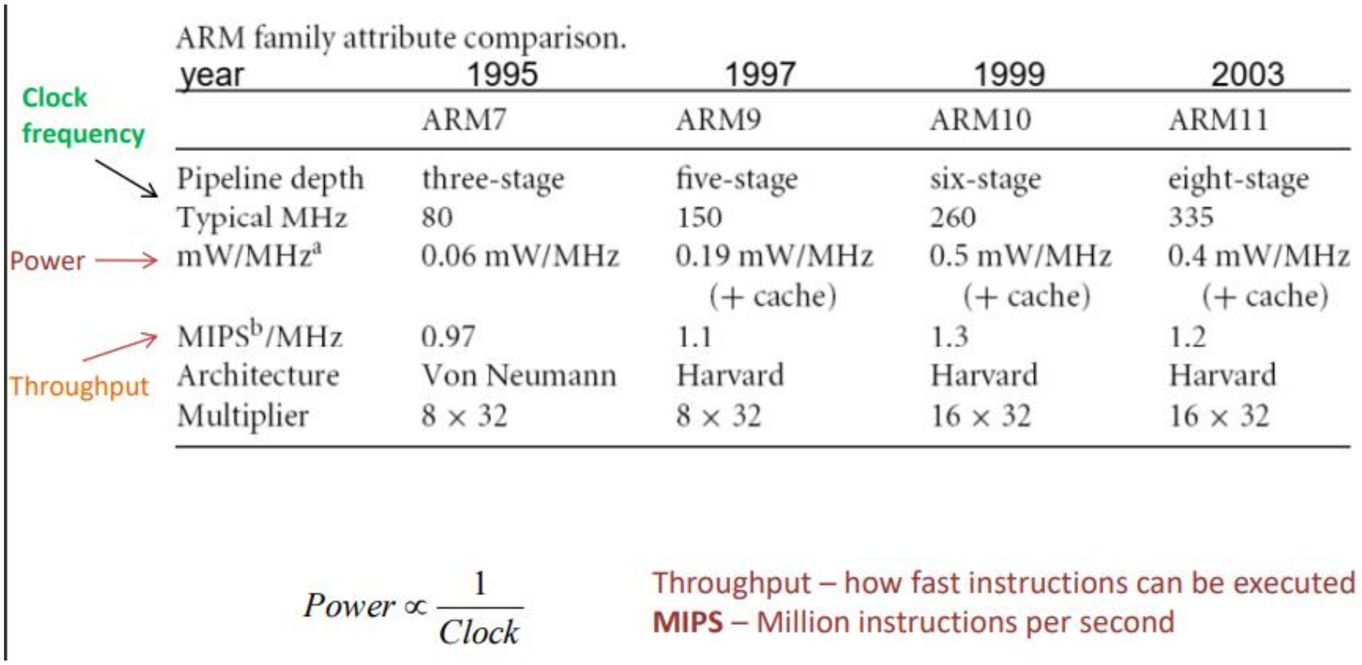
Embedded system is a combination of computer hardware and software designed for a specific function or functions within a larger system

About ARM Processors

A simple RISC-based architecture with powerful design • A whole family of ARM processors exist – share similar design principles and common instruction set (backward compatibility) • Design philosophy Small processor (size) for low power consumption (suitable for embedded applications) High code density (Instruction and Data in same memory, space scarcity) for limited memory and physical size restrictions. Can interface with slow and low-cost memory systems Reduced die size for processor to accommodate more peripherals.

Popular ARM architectures

ARM7TDMI – 3 pipeline stages (fetch/decode/execute) – how instructions are executed – High code density/low power consumption – One of the most used ARM-version for low-end systems – where high power not required – All ARM cores after ARM7TDMI include TDMI even if they do not include TDMI in their labels • ARM9TDMI – Compatible with ARM7 – 5 stages (fetch/decode/execute/memory/write) – Separate instruction and data cache (Instruction and Data in same memory till ARM7) • ARM10 - 6 stages (fetch/issue/decode/execute/memory/write)


Clock frequency
Power
Throughput

ARM family attribute comparison.

year	1995	1997	1999	2003
	ARM7	ARM9	ARM10	ARM11
Pipeline depth	three-stage	five-stage	six-stage	eight-stage
Typical MHz	80	150	260	335
mW/MHz[a]	0.06 mW/MHz	0.19 mW/MHz (+ cache)	0.5 mW/MHz (+ cache)	0.4 mW/MHz (+ cache)
MIPS[b]/MHz	0.97	1.1	1.3	1.2
Architecture	Von Neumann	Harvard	Harvard	Harvard
Multiplier	8 × 32	8 × 32	16 × 32	16 × 32

$$Power \propto \frac{1}{Clock}$$

Throughput – how fast instructions can be executed

MIPS – Million instructions per second

ARM RISC

RISC: simple but powerful instructions that execute within a single cycle at high clock speed. • Four major design rules: – Instructions: reduced set/single cycle/fixed length(decoding easy) – Pipeline: decode in one stage/no need for microcode (complicated program) – Registers: a large set of general-purpose registers(GPRs) (data can be stored temporarily in between calculations) – Load/store architecture: Data processing (ALU)instructions apply to registers only (-they do not access memory); load/store to transfer data between registers and memory • The distinction blurs because modern day CISC (Complex Instruction Set Computer) implements RISC concepts

RISC – Load/Store

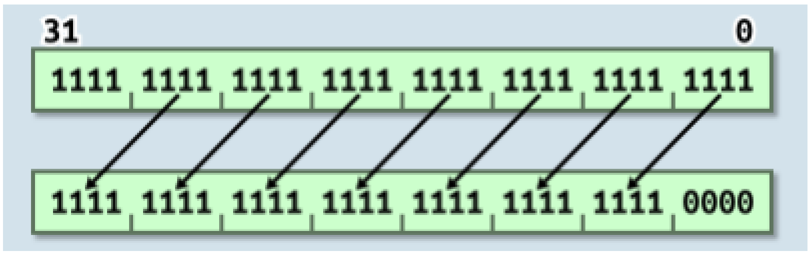
Load-Store architecture: Instructions are classified into 2 categories – Memory access (load and store between memory and registers) and ALU operations (which only occur between registers) For example – In a load–store approach, both operands and destination for an ADD operation must be in registers. In register–memory architecture (for example, a CISC instruction set architecture such as x86) in which one of the operands for the ADD operation may be in memory, while the other is in a register

ARM specific features – which differ from RISC

• Variable cycle execution for certain instructions – (multiple-register load/store for higher code density) • Inline barrel shifter leading to more complex instructions - (improves performance and code density) • Thumb 16-bit instruction set: When 32-bit power is not needed, it can work with 16-bit thumb, resulting in 30% code density improvement (32-bit instructions that can be freely intermixed with 16-bit instructions in a program.) • Conditional execution – reduces branches and improves performance (Add 2 numbers provided ‘0’ flag is Set. This is common in other architectures’ branch or jump instructions but ARM allows its use with most mnemonics.) • Enhanced instructions – additional functions like MULTIPLY and ADD especially for DSP applications (-from voice to audio to sensor hubs to machine learning (ML))

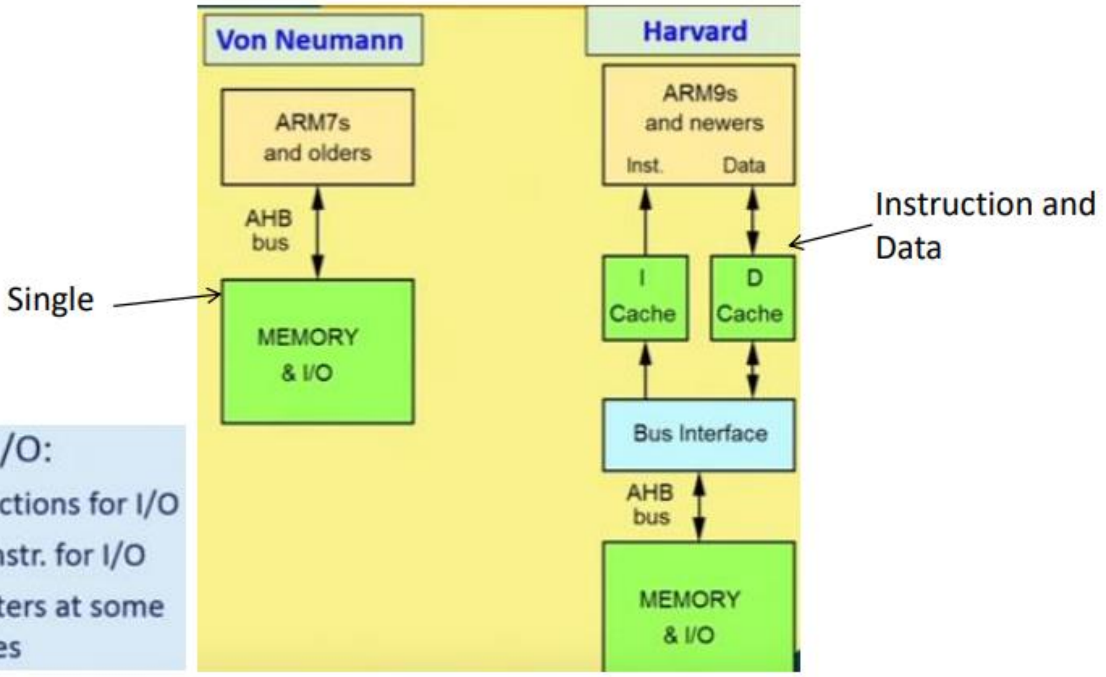
ARM – Barrel Shifting

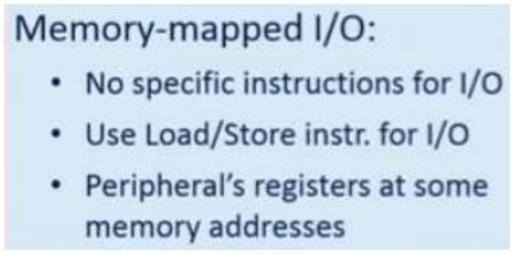
•Barrel shifter is a hardware that allows multiple bit shifting in 1 cycle. •It performs SHIFT and ROTATE operations in ARM processors - 5 types



1. Logic Shift Left (LSL), 2.LSR, 3. Arithmetic Shift Right (ASR), 4. ROR, 5. RRX

Architecture:

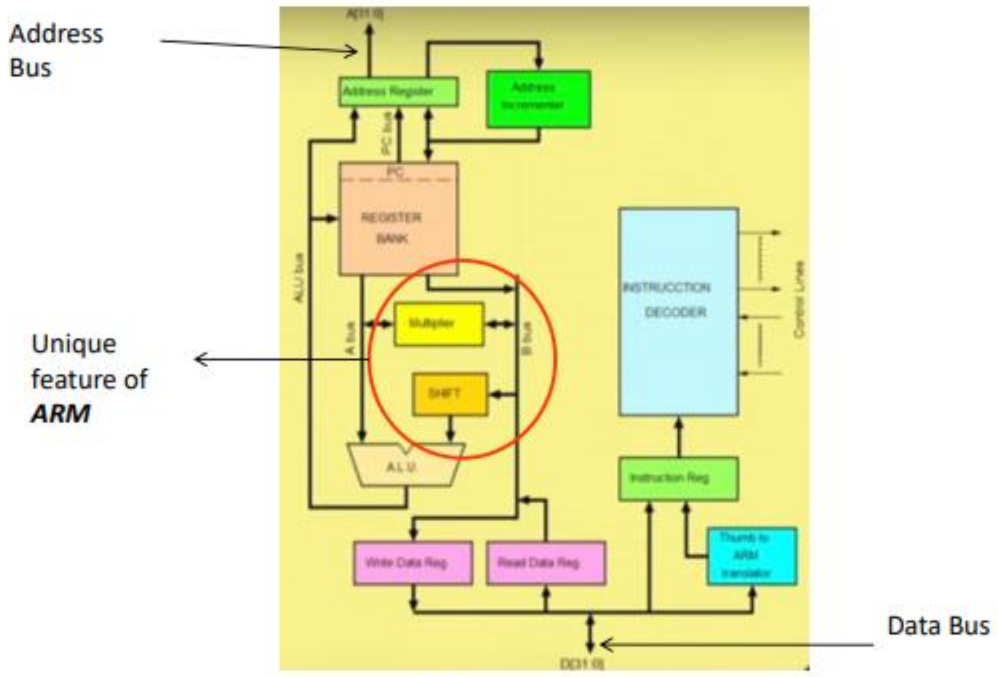



Memory-mapped I/O:

- No specific instructions for I/O
- Use Load/Store instr. for I/O
- Peripheral's registers at some memory addresses

Some part of memory is reserved for I/O

Arm 7 arch



Different modes of ARM Processor

	Mode	Code	Description	Family
	User	usr	Normal program execution, no privileges	All
Interrupt handler routine	FIQ	fiq	Entered when a High priority (Fast) interrupt is raised	All
	IRQ	irq	Entered when a Low priority (Normal) interrupt is raised	All
	Supervisor	svc	Privileged or protected mode for the operating system	All
Memory protection	Abort	abt	Used to handle memory access violations	ARMv3+
Expansion	Undefined	und	Used to define undefined instructions - Facilitates emulation of co-processors in hardware	ARMv3+
	System	sys	Runs privileged operating system tasks	ARMv4+

REGISTERS -I • ARM has 37 registers all of which are 32 bit long •

These registers are –

a) 1 dedicated program counter (PC)

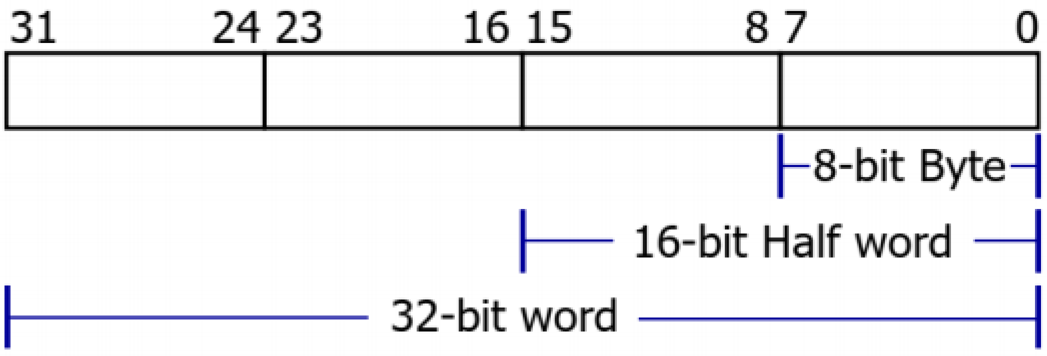
b) 1 dedicated current program status register (CPSR)

c) 5 dedicated saved program status register (SPSR)

d) 30 general purpose registers (GPR)

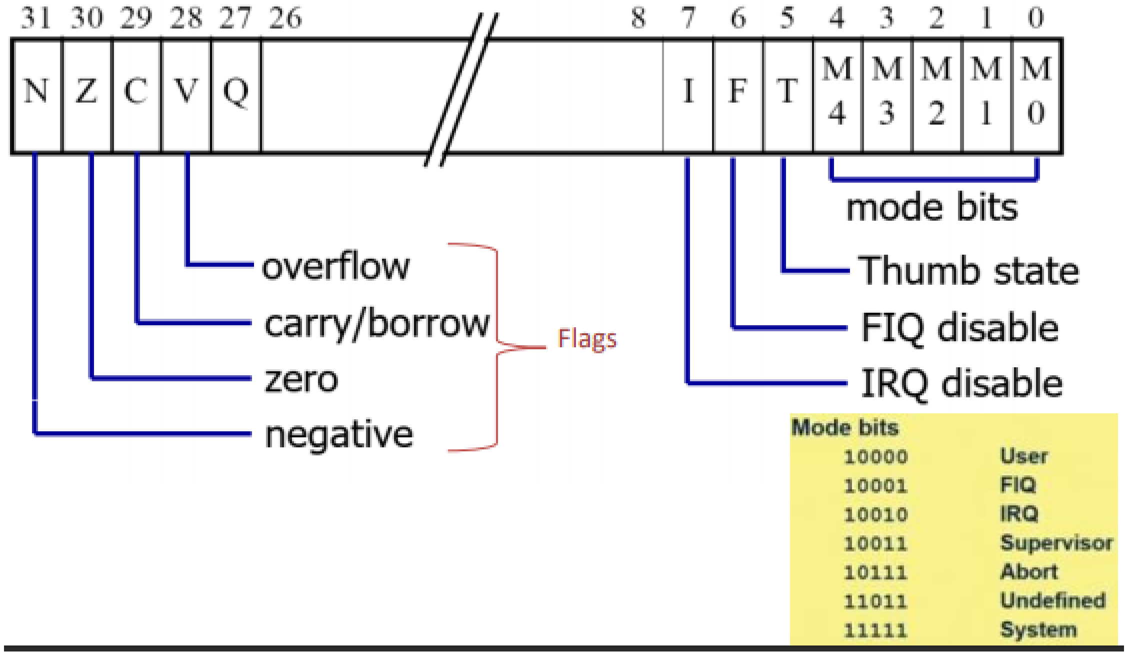
REGISTERS - II • Processor mode governs which of the several register sets is accessible • Only 16 registers are visible to a specific mode of operation. Each mode can access- A particular set of registers (r0-r12) r13 – Stack Pointer (SP) r14 – Link register (LR) r15 – Program counter (PC) Current program status register (CPSR)

General Purpose Registers (GPRs) • 6 data types are supported (signed/unsigned) • Operations that are supported – 8 bit byte, 16 bit half word, 32 bit word • All ARM operations : 32-bit



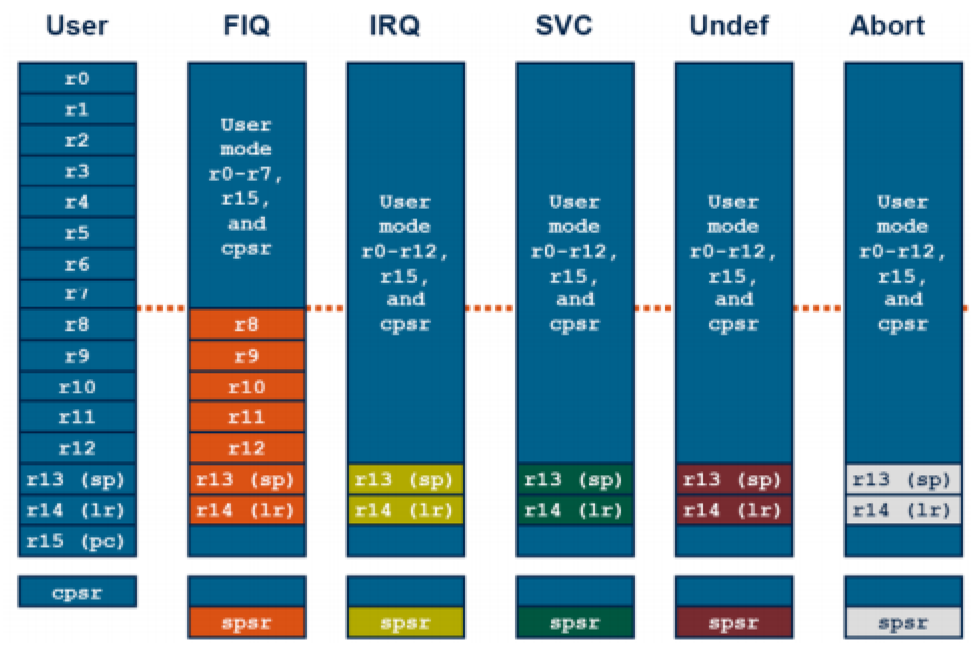
Arithmetic operations – 32 bit, Logical operations – shorter data types

Current Program Status Registers (CPSR)

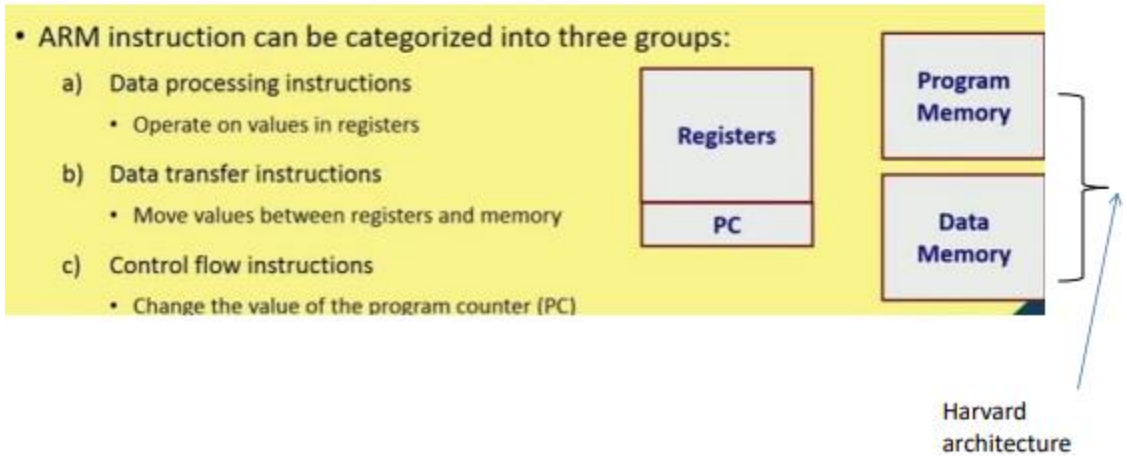


Mode bits	
10000	User
10001	FIQ
10010	IRQ
10011	Supervisor
10111	Abort
11011	Undefined
11111	System

REGISTER ORGANIZATION

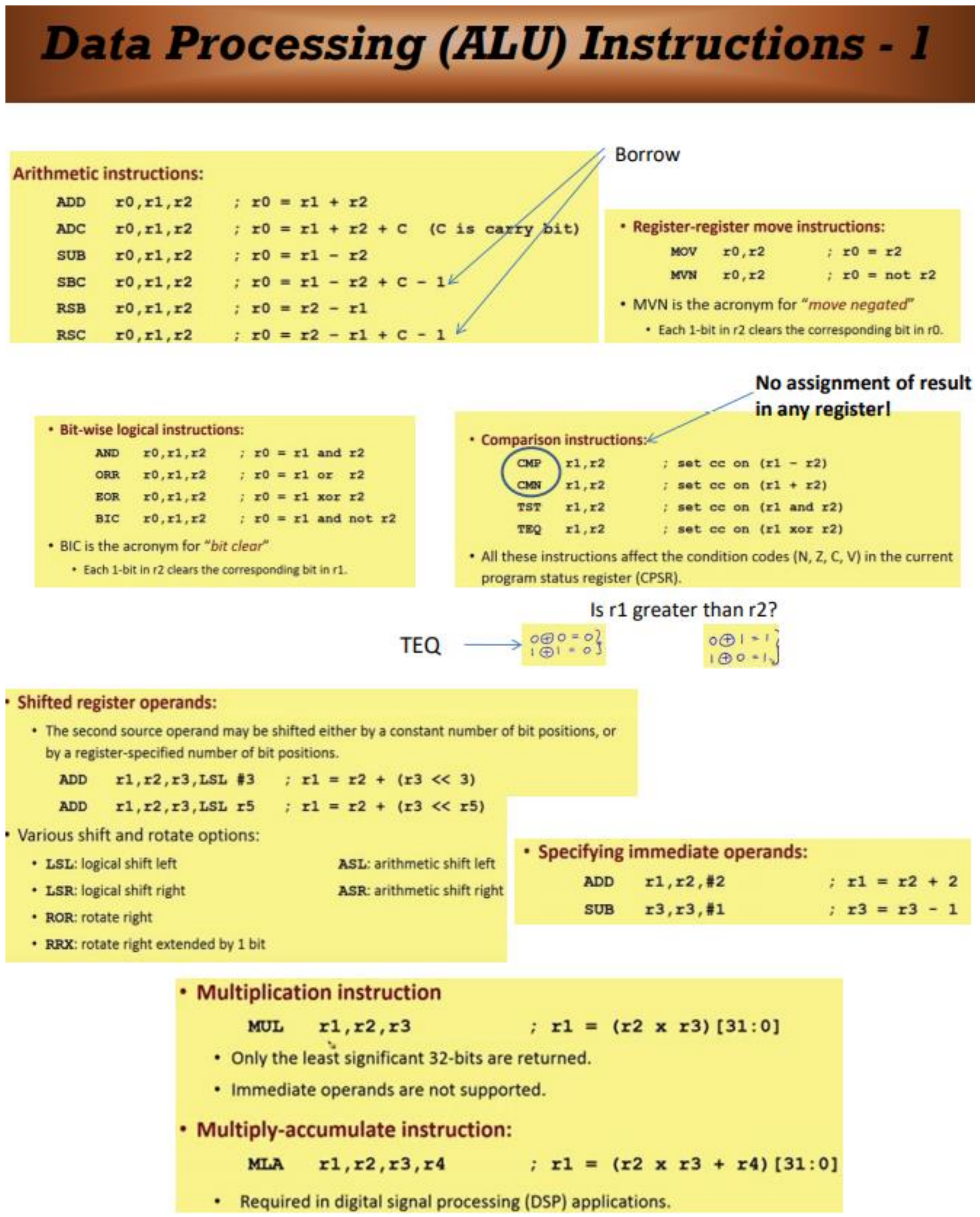


INSTRUCTION SET of ARM Processor



- ARM instruction can be categorized into three groups:
  - a) Data processing instructions
    - Operate on values in registers
  - b) Data transfer instructions
    - Move values between registers and memory
  - c) Control flow instructions
    - Change the value of the program counter (PC)

Registers
PC
Program Memory
Data Memory
Harvard architecture



Data Processing (ALU) Instructions - 1

Borrow

Arithmetic instructions:

```
ADD    r0,r1,r2     ; r0 = r1 + r2
ADC    r0,r1,r2     ; r0 = r1 + r2 + C   (C is carry bit)
SUB    r0,r1,r2     ; r0 = r1 - r2
SBC    r0,r1,r2     ; r0 = r1 - r2 + C - 1
RSB    r0,r1,r2     ; r0 = r2 - r1
RSC    r0,r1,r2     ; r0 = r2 - r1 + C - 1
```

- Register-register move instructions:

```
MOV    r0,r2        ; r0 = r2
MVN    r0,r2        ; r0 = not r2
```

- MVN is the acronym for "*move negated*"
  - Each 1-bit in r2 clears the corresponding bit in r0.

- Bit-wise logical instructions:

```
AND    r0,r1,r2     ; r0 = r1 and r2
ORR    r0,r1,r2     ; r0 = r1 or  r2
EOR    r0,r1,r2     ; r0 = r1 xor r2
BIC    r0,r1,r2     ; r0 = r1 and not r2
```

- BIC is the acronym for "*bit clear*"
  - Each 1-bit in r2 clears the corresponding bit in r1.

No assignment of result in any register!

- Comparison instructions:

```
CMP    r1,r2        ; set cc on (r1 - r2)
CMN    r1,r2        ; set cc on (r1 + r2)
TST    r1,r2        ; set cc on (r1 and r2)
TEQ    r1,r2        ; set cc on (r1 xor r2)
```

- All these instructions affect the condition codes (N, Z, C, V) in the current program status register (CPSR).

Is r1 greater than r2?

TEQ $\longrightarrow$ $0 \oplus 0 = 0$, $1 \oplus 1 = 0$ $\quad$ $0 \oplus 1 = 1$, $1 \oplus 0 = 1$

- Shifted register operands:
  - The second source operand may be shifted either by a constant number of bit positions, or by a register-specified number of bit positions.

```
ADD    r1,r2,r3,LSL #3    ; r1 = r2 + (r3 << 3)
ADD    r1,r2,r3,LSL r5    ; r1 = r2 + (r3 << r5)
```

- Various shift and rotate options:
  - **LSL**: logical shift left $\quad$ **ASL**: arithmetic shift left
  - **LSR**: logical shift right $\quad$ **ASR**: arithmetic shift right
  - **ROR**: rotate right
  - **RRX**: rotate right extended by 1 bit

- Specifying immediate operands:

```
ADD    r1,r2,#2           ; r1 = r2 + 2
SUB    r3,r3,#1           ; r3 = r3 - 1
```

- Multiplication instruction

```
MUL    r1,r2,r3           ; r1 = (r2 x r3)[31:0]
```

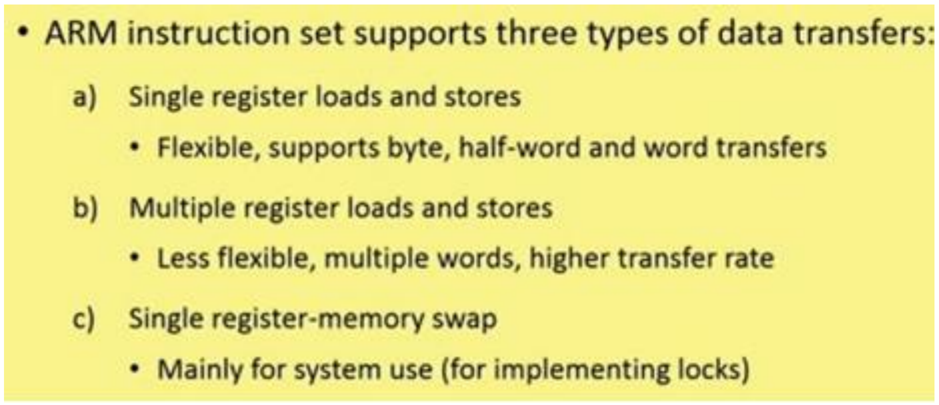
- Only the least significant 32-bits are returned.
- Immediate operands are not supported.

- Multiply-accumulate instruction:

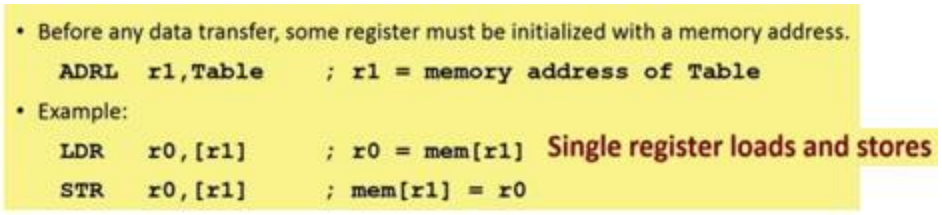
```
MLA    r1,r2,r3,r4        ; r1 = (r2 x r3 + r4)[31:0]
```

- Required in digital signal processing (DSP) applications.

Data Transfer Instructions



- ARM instruction set supports three types of data transfers:
  a) Single register loads and stores
     - Flexible, supports byte, half-word and word transfers
  b) Multiple register loads and stores
     - Less flexible, multiple words, higher transfer rate
  c) Single register-memory swap
     - Mainly for system use (for implementing locks)



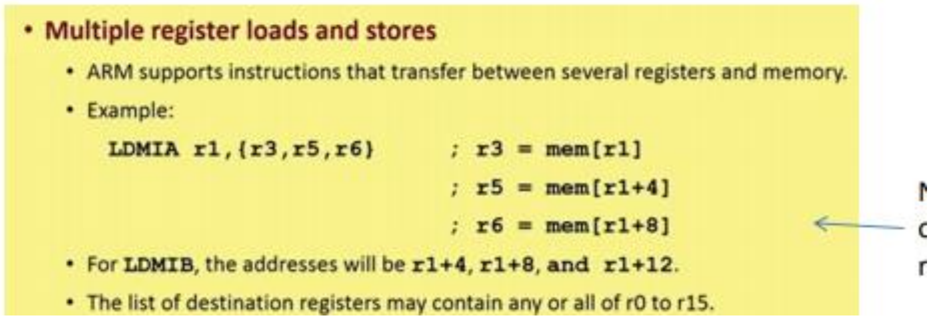
- Before any data transfer, some register must be initialized with a memory address.

```
ADRL  r1,Table     ; r1 = memory address of Table
```

- Example:

```
LDR   r0,[r1]      ; r0 = mem[r1]
STR   r0,[r1]      ; mem[r1] = r0
```

Single register loads and stores

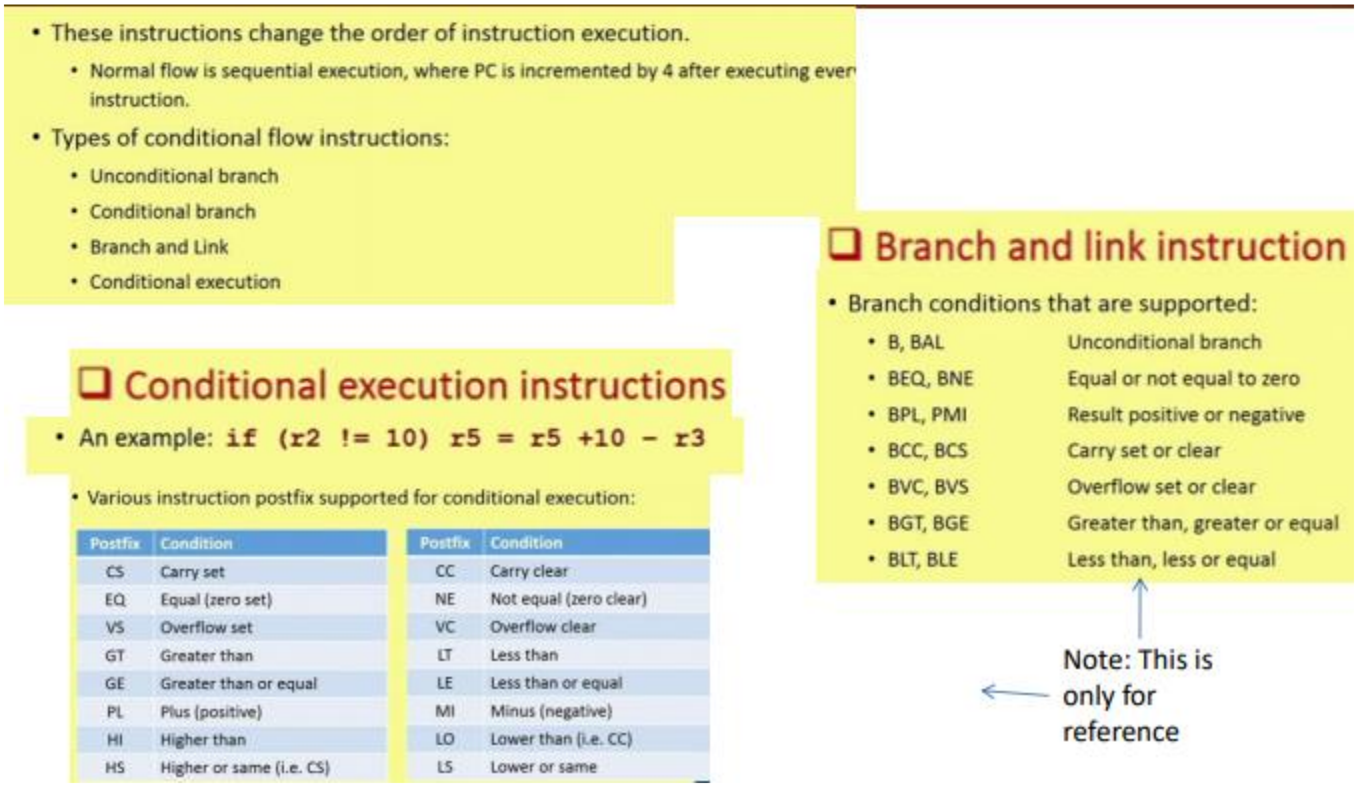


- Multiple register loads and stores
  - ARM supports instructions that transfer between several registers and memory.
  - Example:

```
LDMIA r1,{r3,r5,r6}       ; r3 = mem[r1]
                          ; r5 = mem[r1+4]
                          ; r6 = mem[r1+8]
```

  - For **LDMIB**, the addresses will be **r1+4**, **r1+8**, **and r1+12**.
  - The list of destination registers may contain any or all of r0 to r15.

Control Flow Instructions



- These instructions change the order of instruction execution.
  - Normal flow is sequential execution, where PC is incremented by 4 after executing ever instruction.
- Types of conditional flow instructions:
  - Unconditional branch
  - Conditional branch
  - Branch and Link
  - Conditional execution

❑ Conditional execution instructions

- An example: `if (r2 != 10) r5 = r5 +10 - r3`
  - Various instruction postfix supported for conditional execution:

Postfix	Condition	Postfix	Condition
CS	Carry set	CC	Carry clear
EQ	Equal (zero set)	NE	Not equal (zero clear)
VS	Overflow set	VC	Overflow clear
GT	Greater than	LT	Less than
GE	Greater than or equal	LE	Less than or equal
PL	Plus (positive)	MI	Minus (negative)
HI	Higher than	LO	Lower than (i.e. CC)
HS	Higher or same (i.e. CS)	LS	Lower or same

❑ Branch and link instruction

- Branch conditions that are supported:

B, BAL	Unconditional branch
BEQ, BNE	Equal or not equal to zero
BPL, PMI	Result positive or negative
BCC, BCS	Carry set or clear
BVC, BVS	Overflow set or clear
BGT, BGE	Greater than, greater or equal
BLT, BLE	Less than, less or equal

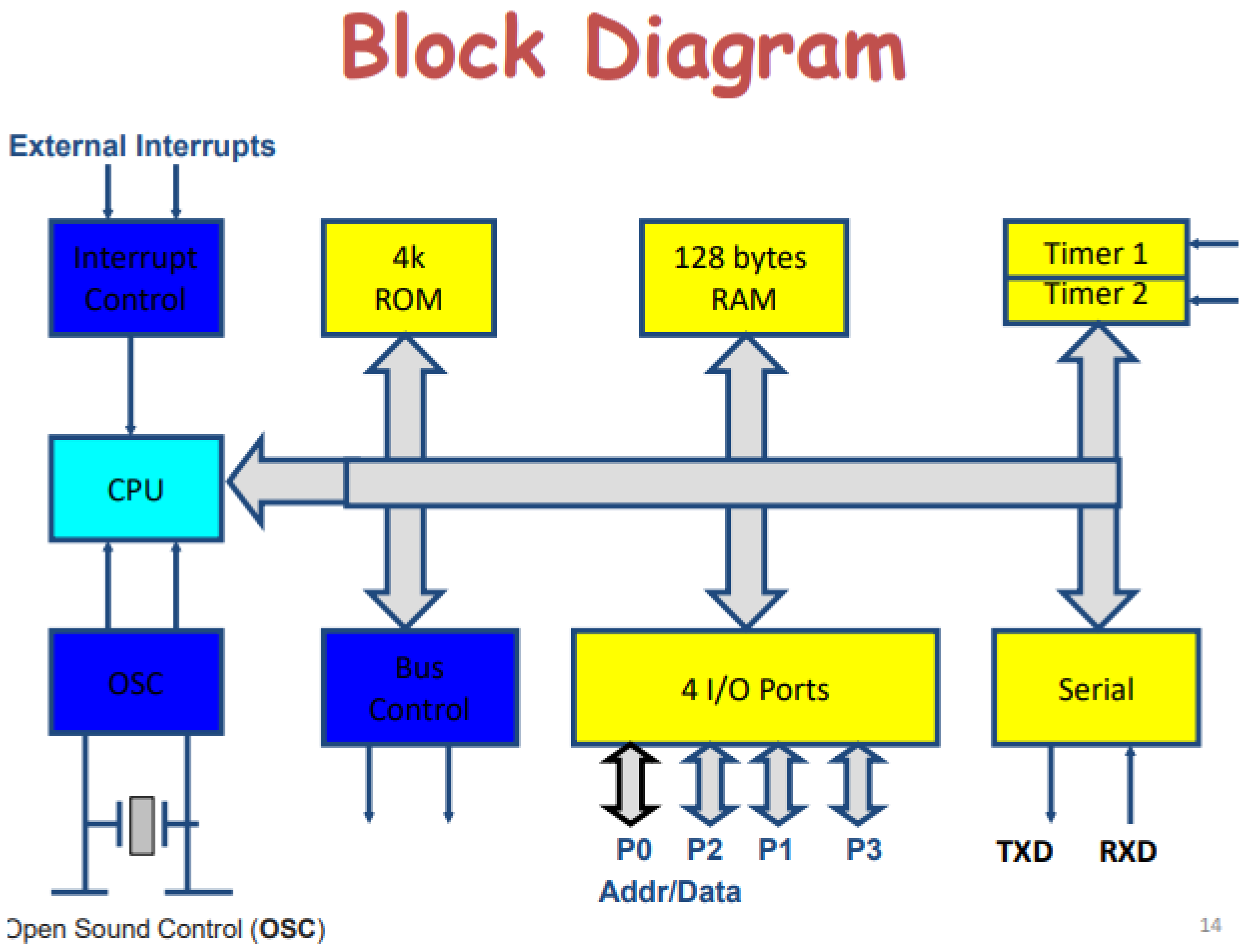
Note: This is only for reference

The 8051 Microcontroller

	Microprocessor	Microcontroller
Application	It used where **intensive processing** is required. It is used in personal computers, laptops, mobiles, video games, etc.	It used where the **task is fixed and predefined**. It is used in the washing machine, alarm, etc.
Structure	It has only the CPU in the chip. Other devices like I/O port, memory, timer is connected externally. The structure of the microprocessor is flexible. Users can decide the amount of memory, the number of I/O port and other peripheral devices.	CPU, Memory, I/O port and all other devices are connected on the single chip. The structure is fixed. Once it is designed the user cannot change the peripheral devices.
Clock speed	The **clock speed** of the microprocessor **is high**. It is in terms of the GHz. It ranges between 1 GHz to 4 GHz.	The **clock speed** of the microcontroller **is less**. It is in terms of the MHz. it ranges between 1 MHz to 300 MHz.
RAM	The volatile memory (RAM) for the microprocessor is in the range of the 512 MB to 32 GB.	The volatile memory (RAM) for the microcontroller is in the range of 2 KB to 256 KB.
ROM	The hard disk (ROM) for the microprocessor is in the range of the 128 GB to 2 TB.	The hard drive or flash memory (ROM) is in the range of the 32 KB to 2 MB.
Peripheral interface	The common peripheral interface for the microprocessor is USB, UART, and high-speed Ethernet.	The common peripheral interface for the microcontroller is I2C, SPI, and UART.
Programming	The program for the microprocessor can be changed for different applications. The programming of the microprocessor is difficult compared to the microcontroller.	The program for the microcontroller is fixed once it is designed.
Bit size	It is available in **32-Bit and 64-bit.**	It is available in **8-bit, 16-bit, and 36-bit.**
Cost	The cost of the microprocessor is high compared to the microcontroller.	It is cheaper.
Power consumption	The power consumption for the microprocessor is high.	The power consumption for the microcontroller is less.
Size	The overall size of the system is large.	The overall size of the system is small.

8051 Basic Component

4K bytes internal ROM • 128 bytes internal RAM • Four 8-bit I/O ports (P0 - P3). • Two 16-bit timers/counters • One serial interface

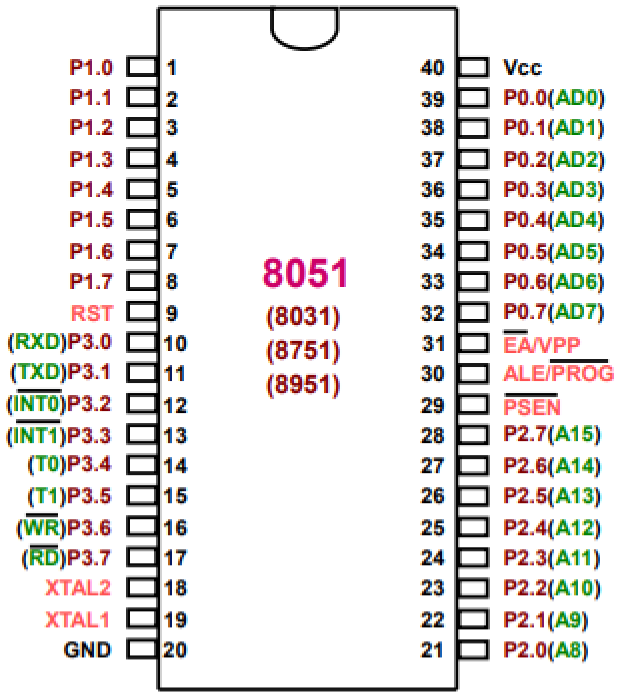


Other 8051 features • only 1 On chip oscillator (external crystal) • 6 interrupt sources (2 external , 3 internal, Reset) • 64K external code (program) memory(only read)PSEN • 64K external data memory(can be read and write) by RD,WR • Code memory is selectable by EA (internal or external) • We may have External memory as data and code

Embedded System (8051 Application) • What is Embedded System? – An embedded system is closely integrated with the main system – It may not interact directly with the environment – For example – A microcomputer in a car ignition control □ An embedded product uses a microprocessor or microcontroller to do one task only □ There is only one application software that is typically burned into ROM

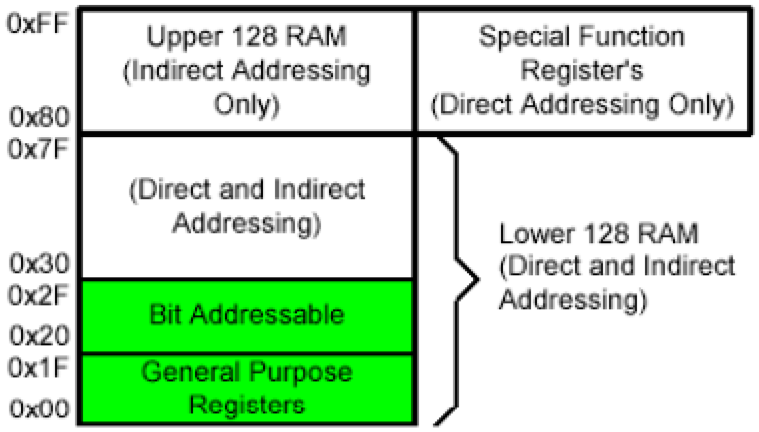
Examples of Embedded Systems • Keyboard • Printer • video game player • MP3 music players • Embedded memories to keep configuration information • Mobile phone units • Domestic (home) appliances • Data switches • Automotive controls

Comparison of the 8051 Family Members • ROM type – 8031 no ROM – 80xx mask ROM – 87xx EPROM – 89xx Flash EEPROM • 89xx – 8951 – 8952 – 8953 – 8955 – 898252 – 891051 – 892051 • Example (AT89C51,AT89LV51,AT89S51) – AT= ATMEL(Manufacture) – C = CMOS technology – LV= Low Power(3.0v)



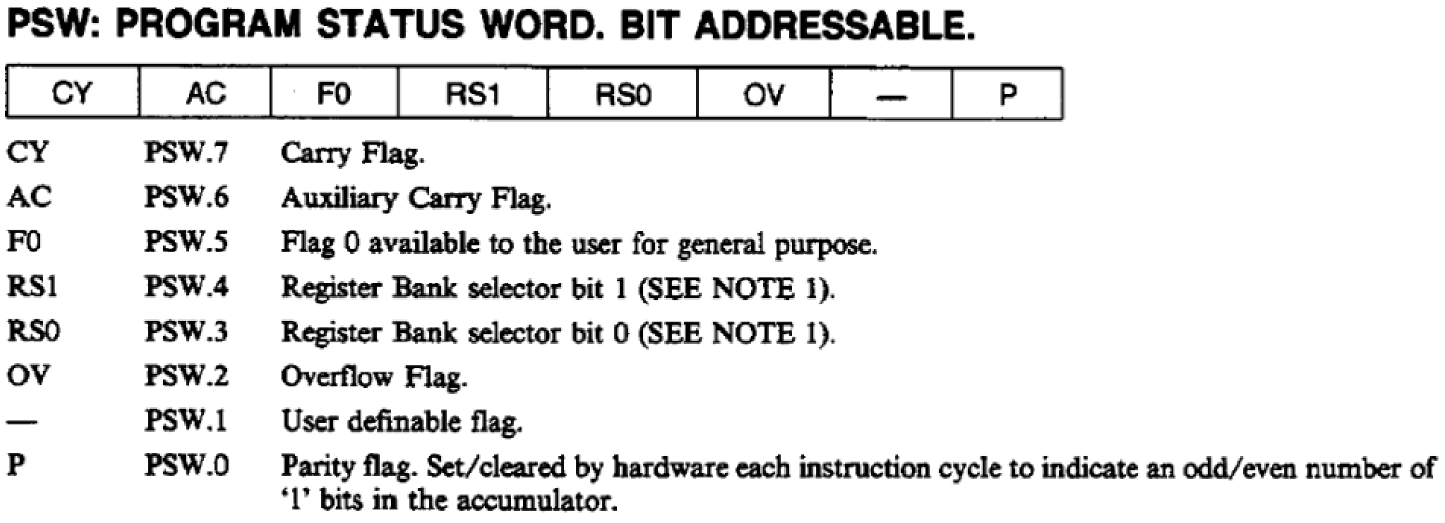
Pins 1 to 8 − These pins are known as Port 1. This port doesn't serve any other functions. It is internally pulled up, bi-directional I/O port. • Pin 9 − It is a RESET pin, which is used to reset the microcontroller to its initial values. • Pins 10 to 17 − These pins are known as Port 3. This port serves some functions like interrupts, timer input, control signals, serial communication signals RxD and TxD, etc. • Pins 18 & 19 − These pins are used for interfacing an external crystal to get the system clock. • Pin 20 − This pin provides the power supply to the circuit. • Pins 21 to 28 − These pins are known as Port 2. It serves as I/O port. Higher order address bus signals are also multiplexed using this port. • Pin 29 − This is PSEN pin which stands for Program Store Enable. It is used to read a signal from the external program memory. • Pin 30 − This is EA pin which stands for External Access input. It is used to enable/disable the external memory interfacing. • Pin 31 − This is ALE pin which stands for Address Latch Enable. It is used to demultiplex the address-data signal of port. • Pins 32 to 39 − These pins are known as Port 0. It serves as I/O port. Lower order address and data bus signals are multiplexed using this port. • Pin 40 − This pin is used to provide power supply to the circuit. 6 IMPORTANT PINS (IO Ports) • One of the most useful features of the 8051 is that it contains four I/O ports (P0 - P3) • Port 0 （pins 32-39） : P0（P0.0～P0.7） – 8-bit R/W - General Purpose I/O – Or acts as a multiplexed low byte address and data bus for external memory design • Port 1 （pins 1-8） : P1（P1.0～P1.7） – Only 8-bit R/W - General Purpose I/O • Port 2 （pins 21-28） : P2（P2.0～P2.7） – 8-bit R/W - General Purpose I/O – Or high byte of the address bus for external memory design • Port 3 （pins 10-17） : P3（P3.0～P3.7） – General Purpose I/O – if not using any of the internal peripherals (timers) or external interrupts. • Each port can be used as input or output (bi-direction) 7

On-Chip Memory Internal RAM


0xFF
Upper 128 RAM (Indirect Addressing Only)
Special Function Register's (Direct Addressing Only)
0x80
0x7F
(Direct and Indirect Addressing)
Lower 128 RAM (Direct and Indirect Addressing)
0x30
0x2F
Bit Addressable
0x20
0x1F
General Purpose Registers
0x00

Register Banks

Active bank selected by PSW [RS1,RS0] bit Permits fast “context switching” in interrupt service routines (ISR)



PSW: PROGRAM STATUS WORD. BIT ADDRESSABLE.

CY	AC	F0	RS1	RS0	OV	—	P

CY	PSW.7	Carry Flag.
AC	PSW.6	Auxiliary Carry Flag.
F0	PSW.5	Flag 0 available to the user for general purpose.
RS1	PSW.4	Register Bank selector bit 1 (SEE NOTE 1).
RS0	PSW.3	Register Bank selector bit 0 (SEE NOTE 1).
OV	PSW.2	Overflow Flag.
—	PSW.1	User definable flag.
P	PSW.0	Parity flag. Set/cleared by hardware each instruction cycle to indicate an odd/even number of ‘1’ bits in the accumulator.

The 8051 Assembly Language

Data Transfer Instructions

• MOV dest, source dest <- source •

Stack instructions

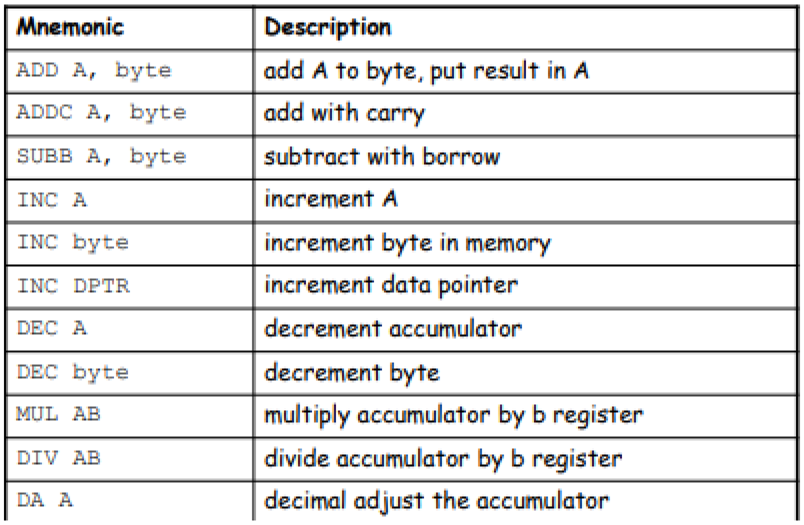
PUSH byte ; increment stack pointer, ;move byte on stack

POP byte ;move from stack to byte, ;decrement stack pointer

• Exchange instructions XCH a, byte ;exchange accumulator and byte

XCHD a, byte ; exchange low nibbles of ;accumulator and byte

Arithmetic Instructions



Mnemonic	Description
ADD A, byte	add A to byte, put result in A
ADDC A, byte	add with carry
SUBB A, byte	subtract with borrow
INC A	increment A
INC byte	increment byte in memory
INC DPTR	increment data pointer
DEC A	decrement accumulator
DEC byte	decrement byte
MUL AB	multiply accumulator by b register
DIV AB	divide accumulator by b register
DA A	decimal adjust the accumulator

CLR - clear RL – rotate left RLC – rotate left through Carry RR – rotate right RRC – rotate right through Carry SWAP – swap accumulator nibbles

PMMC – Permanent Magnet Moving Coil

 Instrument that allows you to measure the current through a coil by observing the coil's angular deflection in a uniform magnetic field. A PMMC meter places a coil of wire (i.e. a conductor) in between two permanent magnets in order to create stationary magnetic field. According to Faraday's Laws of electromagnetic induction, a current carrying conductor placed in a magnetic field will experience a force

The magnitude (strength) of this force will be proportional to the amount of current through the wire. A pointer is attached to the end of the wire and it is put along a scale.

PMMC – Working Principle

D' Arsonval – French physician, physicist, and inventor

Current in coil - produces force or torque (rotational force), called as deflecting torque ….Faraday's law of electromagnetic induction

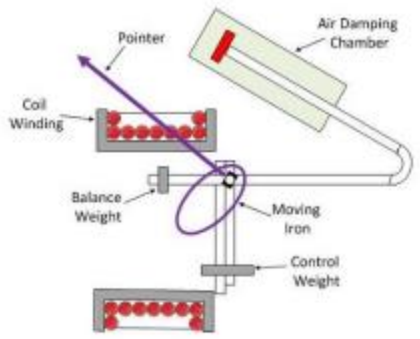
Controlling torque - produced by springs attached to the moving coil – balances the deflecting torque

Deflection of the pointer is calibrated against a scale and it is proportional to the current flowing through the coil.

MI – Moving Iron Instruments

Most primitive measuring instruments These instruments use the effect of attraction or repulsion of a piece of iron towards a magnet (or an electromagnet) CONSTRUCTION: The plate or vane of soft iron is used as the moving element of the instrument. The vane is so placed that it can freely move in the magnetic field of the stationary coil. The conductor makes the stationary coil, and it is excited by the voltage or current whose magnitude is used to be measured. The moving iron instrument uses the stationary coil as an electromagnet. The electromagnet is the temporary magnet whose magnetic field strength increases or decreases with the magnitude of the current passes through it.

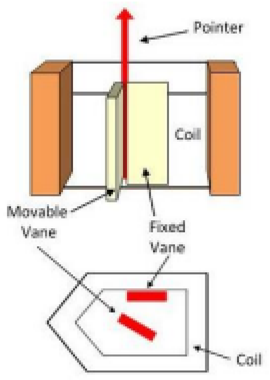
Attraction type MI Instrument



1. Magnetic field of the electromagnet can be easily increased or decreased by controlling the amount of current through the coil.

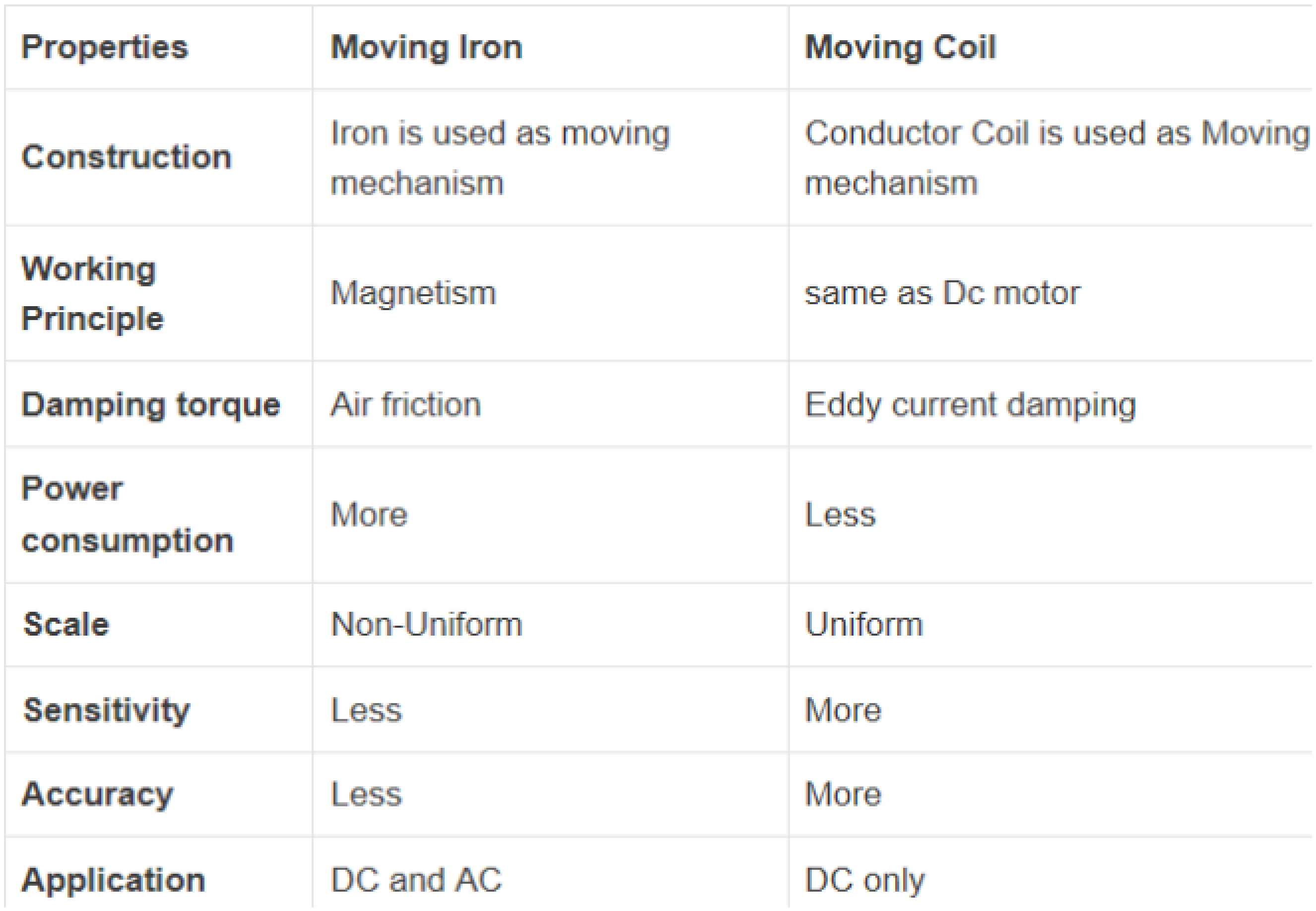
2. The Vane or moving coil is attracted in proportion to the magnetic field and the calibrated pointer moves accordingly.

Repulsion type MI Instrument



 Repulsion type instrument has two vanes or iron plates. One is fixed, and the other one is movable. Repulsion type MI Instrument When current flows through the coil, it magnetizes both the vanes and produce similar polarity at the same end! Thus, a repulsive force is produced Because of a repulsive force, the moving coil starts moving away from the fixed vane. The calibrated pointer, moves in proportion to this repulsive force.

Differences:



Properties	Moving Iron	Moving Coil
Construction	Iron is used as moving mechanism	Conductor Coil is used as Moving mechanism
Working Principle	Magnetism	same as Dc motor
Damping torque	Air friction	Eddy current damping
Power consumption	More	Less
Scale	Non-Uniform	Uniform
Sensitivity	Less	More
Accuracy	Less	More
Application	DC and AC	DC only