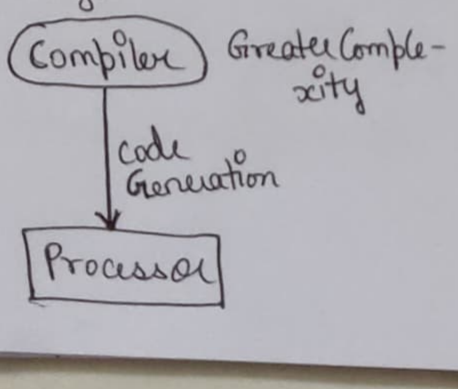
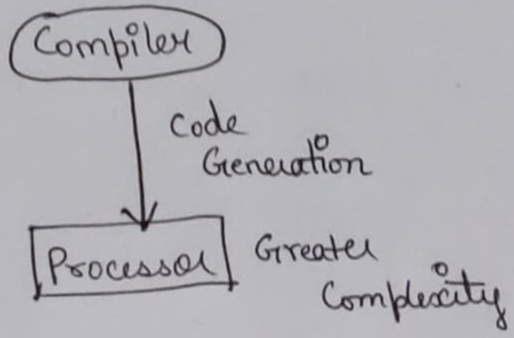
Name :- Pranjali . S
USN :- 1CR19IS107
Section :- B
Subject :- MCES
Assignment 1
Due Date :- 17/5/21

1) Differentiate between RISC and CISC processors?

⟹

RISC	CISC
① Emphasizeson compiler complexity	① Emphasizes on processor complexity
② Simple but powerful instructions	② Instructions are complex
③ Excecutes instuctions in single cycle.	③ Excecutes instructions in multiple cycles.
④ Instructions are of fixed ~~leg~~ length	④ Instructions are of variable length.
⑤ Have a large set of general purpose registers	⑤ Have a limited set of general purpose registers.
⑥ Any register can contain data or address.	⑥ Dedicated registers have special purpose.
⑦ Separate load and store Instructions transfer data between memory and register.	⑦ MOV instructions can be used to transfer data between register & memory
⑧ 	⑧ 

2) Explain the major design rules to implement RISC philosophy?

⇒ The four design rules are:-

① Instructions :- RISC has a reduced number of instruction classes. These classes provide simple operations so that each is executed in a single cycle. Each instruction is of fixed length to allow the pipeline to fetch instructions before decoding the current instruction.

② Pipeline :- The processing of instructions is broken down into smaller units that can be executed in parallel by pipelines.

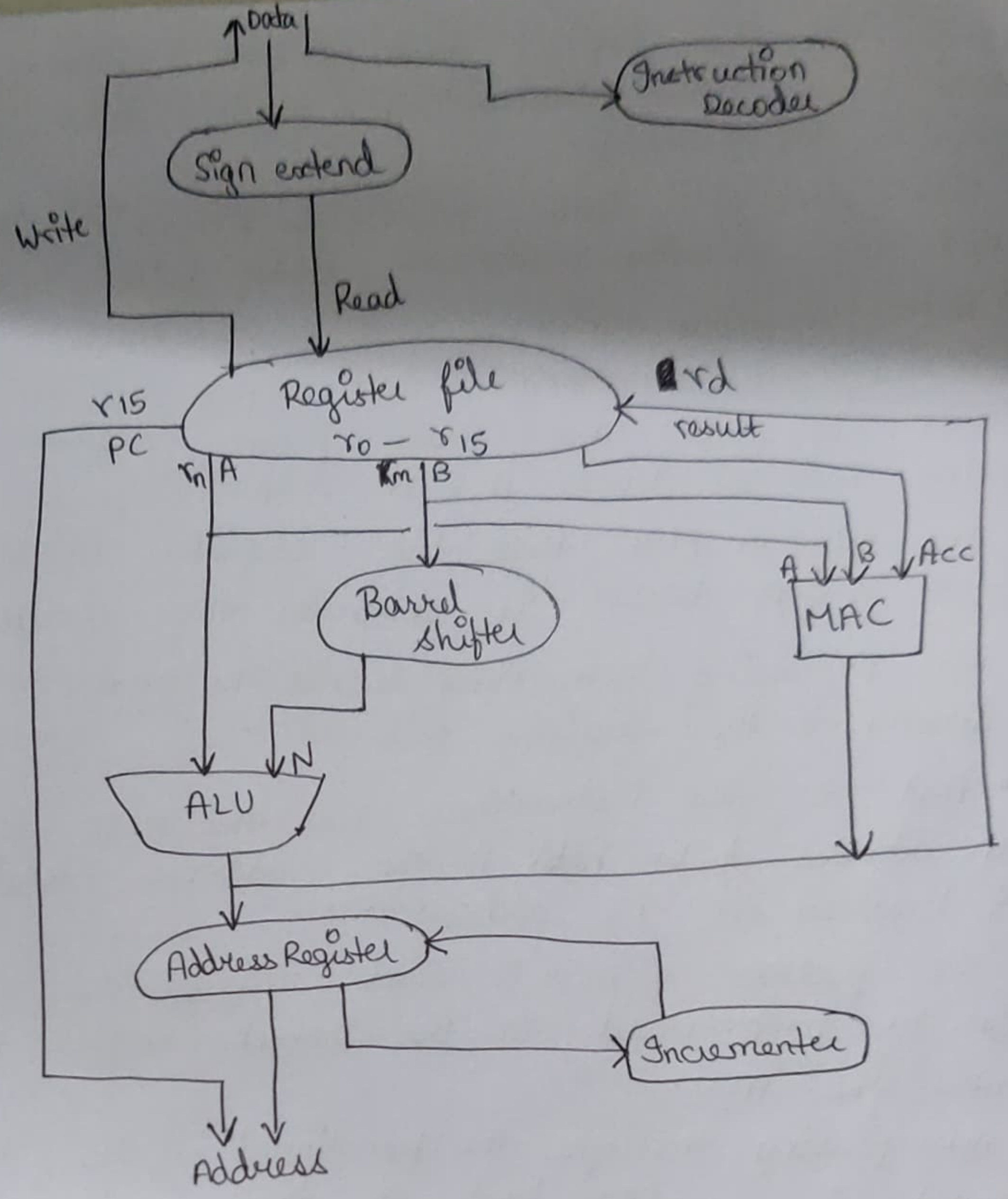
③ Registers :- RISC machines have a large number of general purpose registers which can contain either data or address.

④ Load - Store Architecture :- The processor operates on data held in registers. Separate load and store instructions transfer data between register bank and external memory.

3) Explain the ARM core dataflow model with a neat diagram?

PTO

⇒



- An ARM core has functional units which are connected by using data buses where arrows represent flow of data and the lines represent the buses & the boxes represent the operation unit or storage area.

- The instruction decoder translates instructions before they are executed.

- The ARM processor, like all RISC processors uses a load- store architecture.

- Load instruction copies data from memory to registers & store instruction copies data from registers to memory.
- There are no data ~~manipula~~ processing instructions that can directly manipulate data held in memory.
- ARM instructions typically have 2 source registers rn & rm & a single destination register rd. Source operands are read from register file using internal buses A & B respectively.
- The ALU or MAC take the register values rn & rm from A & B buses & computes the result.
- Data processing instructions write the result in rd directly to the register file.
- Load & store instructions use the ALU to generate an address to be held in the address register & broadcast on the address bus.
- One feature of ARM is that register rm alternatively can be preprocessed in the barrel shifter before it enters the ALU.
- After passing through the functional units, the result in rd is written back to the register file using the result bus.
- After load & store instructions, the incrementer updates address register before the code ~~reads~~ reads or writes the next register value from or to the next sequential memory location.

PTO

• 4) Explain the programmer's model of ARM processor with complete register sets available?

⇒

User & System	fast interrupt request	interrupt request	Supervisor	Undefined	Abort
r0					
r1					
r2					
r3					
r4					
r5					
r6					
r7					
r8	r8_fiq				
r9	r9_fiq				
r10	r10_fiq				
r11	r11_fiq				
r12	r12_fiq				
r13 SP	r13_fiq	r13_irq	r13-svc	r13_undef	r13_abt
r14 LR	r14_fiq	r14_irq	r14-svc	r14-undef	r14_abt
r15 PC					

cpsr					
—	spsr_fiq	spsr_irq	spsr_svc	spsr_undef	spsr_abt

• ARM processor has in total 37 registers in the register file.

• Out of these 37:-

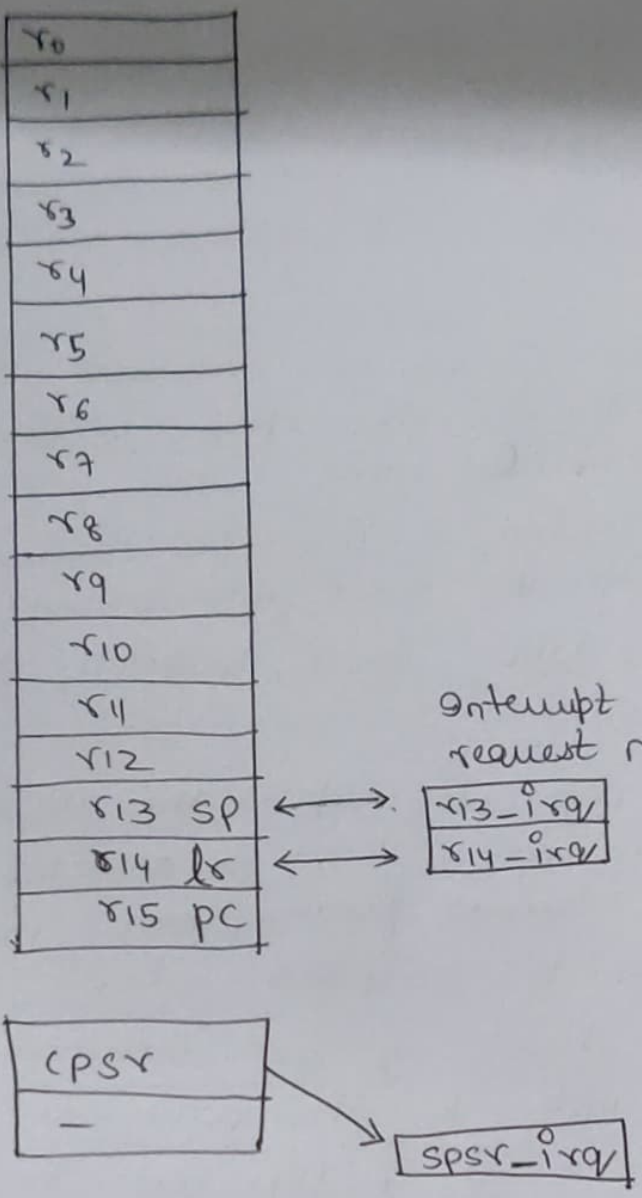
① 20 registers are hidden from program at different times and are called as banked registers.

② • They are available only when processor is in a particular mode.

• Banked registers of a particular mode are ~~undefined~~ denoted by an underline character postfixed to the mode mnemonic.

- Every processor mode except the user mode can change mode by writing directly to themode bit of cpsr.
- All preveliged modes except the system mode have an a set of associated banked registers that are subset of the main 16 registers.
- If the processor mode is changed, a banked register from the new mode will replace an existing register.
- The processor mode can be changed by a program that writes directly to the cpsr when processor core is in privillege mode.
- The following exception & interupts causes a mode change: reset, interupt request, fast interupt request, software interupt, data abort, prefetch abort & undefined instructions.
- Exceptions and interupts suspend the normal execution of sequential instructions & jump to a specific location.
- When the core changes from user mode to interupt request mode which happens when an interupt request occurs due to an external device raising an interupt to the processor core, then the user registers r13 & r14 are to be banked.
- The user registers are replaced with r13_irq & r14_irq.
- r14_irq is link register & r13-irq is stackpointer
- The saved program status register (spsr) which stores the previous mode's cpsr.

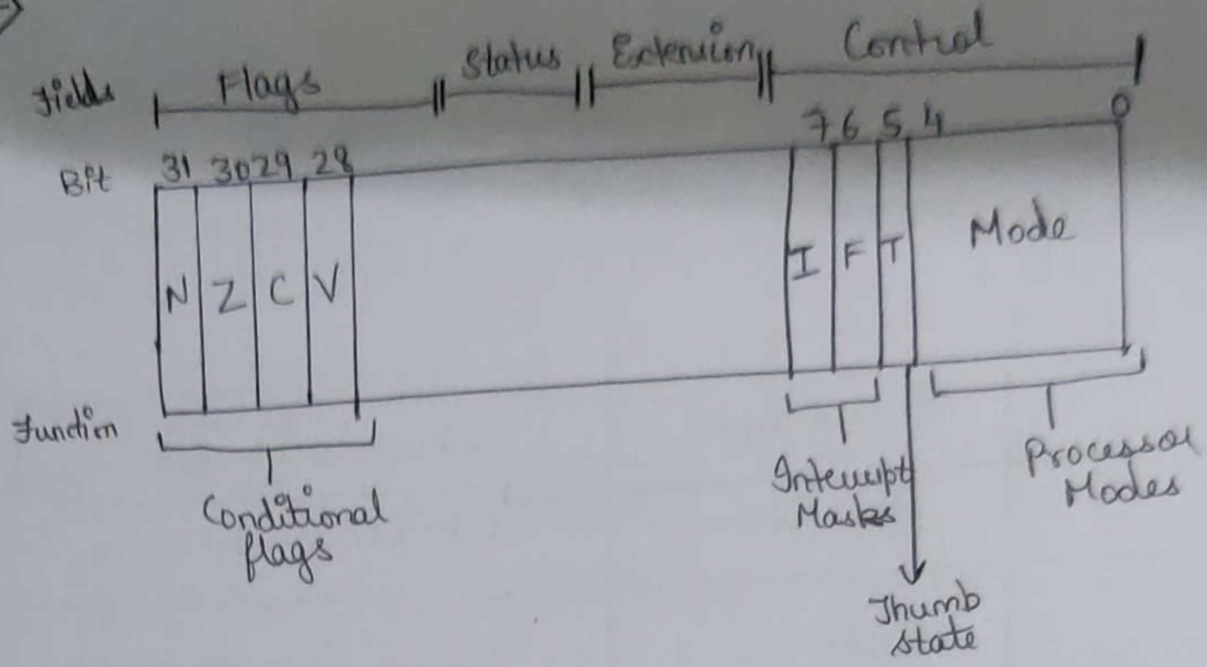
User mode



PTO

5) With a help of a bit layout diagram, explain the current status program register of ARM?

⟹



- The cpsr is divided into 4 fields - each 8 bit ~~long~~ wide:- flags, status, extension and control
- In current designs, the extension and status fields are reserved for future use.
- The control field contains the processor mode, state & interrupt masking bits.
- The flag field contains the conditional flags.

Mode	Bit Order [4:0]
Abort	10111
Interrupt Request	10010
Fast Interrupt Request	10001
Supervisor	10011
System	11111
Undefined	11011
User	10000

- When cpsr bit 5; T=1 then the processor is in thumb state. When T=0, the processor is in ARM state.

- The cpsr has 2 interrupt masking bits 7 & 6 (I & F) which control masking interrupt requests.

- Conditional flags are updated by comparisons & the result of ALU operations that specify the S instruction suffix.

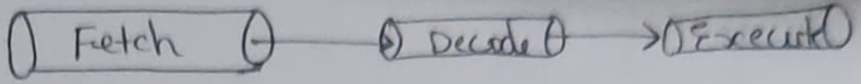
Flag	Flag Name	When is it set
N	Negative	Bit 31 is 1
Z	zero	Result is all zeroes
C	carry	Result causes an unsigned carry
V	Overflow	Result causes a signed overflow

6) What is a pipeline in ARM? Illustrate with an example? Show the pipeline stages of ARM7, ARM9 & ARM10?

PTO

⟹ It is the mechanism to speed up execution of instructions by fetching the next instruction while other instructions are being decoded & executed.

- stages of ARM7 processor:-



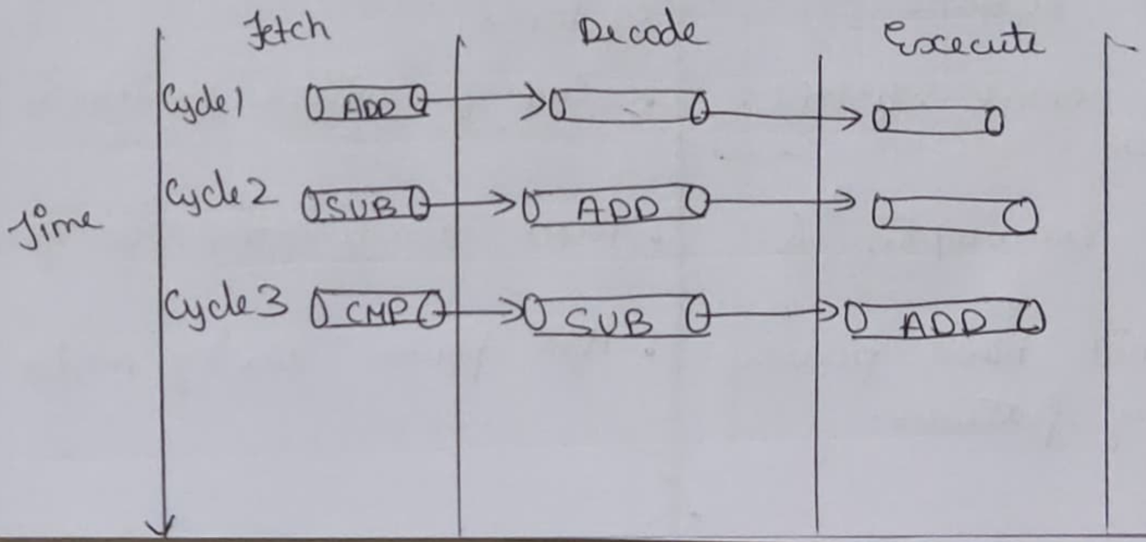
- Fetch loads an instruction from memory.
- Decode identifies the instruction to be executed
- Execute processes the instructions and writes the result back to a register.
- Each instruction takes a single cycle to complete after the pipeline is filled.

↳ In 1st cycle core fetches ADD instruction from memory.

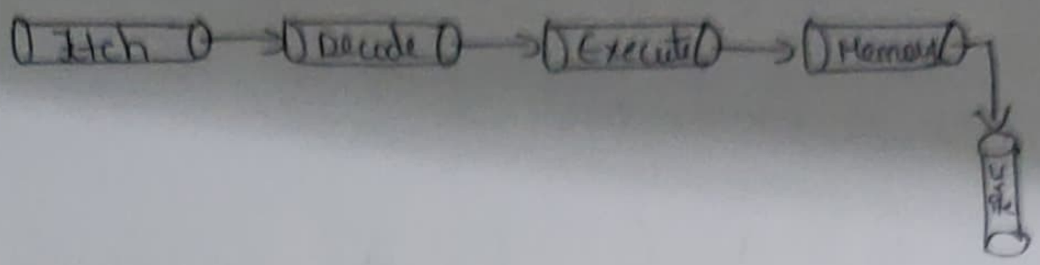
↳ In 2nd cycle core decodes ADD instruction & fetches SUB instruction from memory.

↳ In 3rd cycle core executes ADD instruction, decodes SUB instruction & fetches CMP instruction from memory.

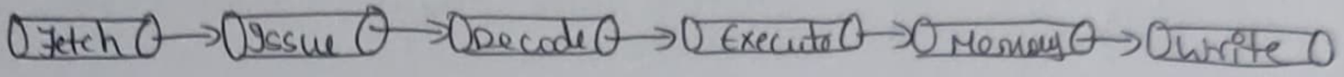
↳ ADD is executed SUB is decoded & CMP is fetched. This is called filling the pipeline.



- Stages of ARM9 processor :-



- Stages of ARM 10 processor :-



7) Compare and Contrast microprocessor and microcontroller?

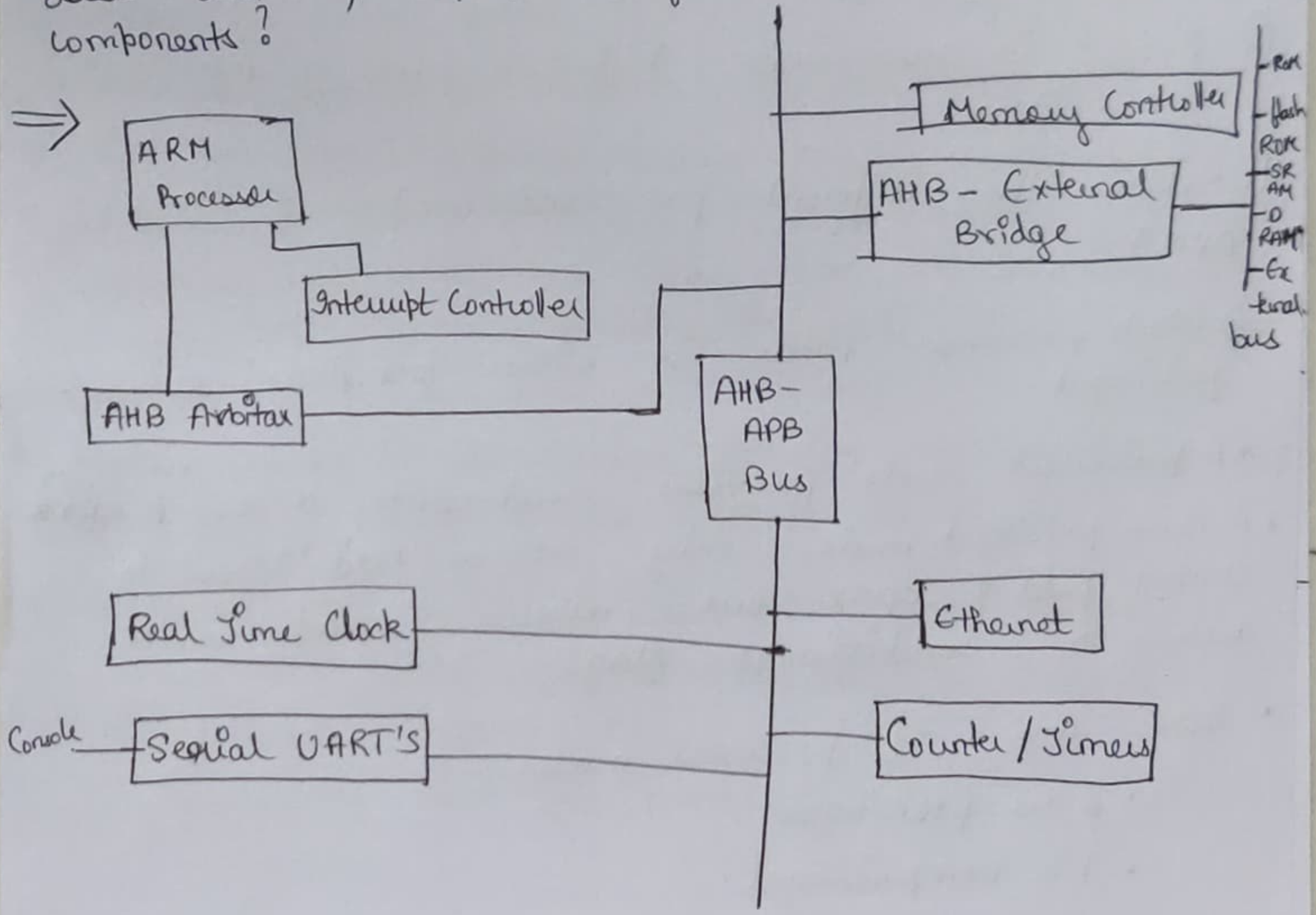
⟹

Microprocessors	Microcontrollers
• Heart of computer systems	• Heart of embedded systems
• Memory and I/o components have to be connected externally.	• ~~Along with~~ It has external memory along with internal memory and I/o compoments
• Circuit becomes large	• Circuit is small.
• Cannot be used in compact systems.	• Can be used in compact systems.
• Cost of entire system increases	• Cost of entire system is low.
• Power consumption is high.	• Power consumption is low.
• Does not have power saving features.	• Has power saving modes.

Microprocessors	Microcontrollers
• Each instruction needs external operation ∴ it is slow	• Most of the operations are internal ∴ it is fast.
• Have less no. of registers ∴ Most of operations are memory-based.	• Have more no. of registers ∴ Programs are easier to write.

8) Along with a neat diagram of an ARM based embedded device, explain the four main hardware components?

⇒



ARM processor based embedded system hardware can be separated into the following four main hardware components :-

① ARM Processor :- It controls the embedded device. Different versions of the processor are available to suite the desired operating characteristics.

② Controllers :- They coordinate important blocks of the system. Two commonly found controllers are memory and interrupt controller.

③ Peripherals :- The peripherals provide all the input - output capability external to the chip and responsible for the uniqueness of the embedded device.

④ Bus :- To communicate different parts of the device.

9) Explain the different processor modes provided by ARM7?

⇒ • Each processor mode is either privileged or non-privileged.

- A privileged mode provides read-write access to CPSR.
- A non-privileged mode only allows read access to control field of CPSR but allows read-write access to conditional flags.

- There are 7 processor modes:-
  - 6 are privileged
  - 1 is non privileged

•

Processor Modes

Privileged Modes

1) Abort
2) Interrupt Request
3) Fast Interrupt Request
4) Supervisor
5) System
6) Undefined

Non-Privileged Modes

1) User Mode

• 1) The processor enters abort mode when there is a failure to attempt to access the memory.

2) Fast Interrupt & Interrupt Request modes correspond to the 2 interrupt levels available on ARM processor.

3) Supervisor mode is the mode that the processor is in after reset & is generally the mode that OS kernel operates in.

4) System mode is a special version of user mode that allows full read-write access to cpsr.

5) Undefined mode is the mode when the processor encounters an undefined instruction or the instruction not supported by the implementation.

6) User mode is used for programs and applications.

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

10) Discuss the ARM design philosophy?

⟹ • ARM processor has been specially designed to be small to reduce power consumption and extend battery operation – essentially for applications such as mobil phones & personal digital assistants.

• High code density is required since embedded systems have limited memory due to cost & physical size restictions. High code density is usefu for applications that have limited on-board memory, such as mobile phones.

• Embedded systems are price sensitive & use low cost memory devices.

PTO

- Another requirement is to reduce the area of die taken up by the processor. For a single chip solution, smaller the area, more available 'space for specialized peripherals'

- ARM has incorporated hardware debug technology within the processor so that software engineers can view what is happening while the processor executes code.

11) Explain ARM bus technology?

⟹ Embedded systems use different bus technologies than those designed for x86 PC's. The most common PC bus technology, the Peripheral Component Inter-Connect (PCI) bus connects devices such as video cards & hard disk controllers to x86 processor bus. this type of technology is external or off-chip & is built into the motherboard of the PC. In contrast embedded devices use an on-chip bus that is internal to the chip & allows different peripheral devices to be interconnected with ARM core. There are 2 different classes of devices attached to the bus.

The ARM processor core is the bus master - a logical device capable of initiating a data transfer with another device across the same bus.

Peripherals tend to be bus slaves - logical devices only capable of responding to a transfer request from the bus master. Bus has 2 architectural levels.
First is the physical level that covers the electrical characteristics & bus width. Second level deals with protocol - set of logical rules that govern the communication between the processor and a peripheral.
ARM is primarily a design company. It seldom improves the electrical characteristics of the bus but it routinely specifies the bus protocol.

12) Describe the conditional execution. Write the different code suffices?

⇒ Conditional execution controls whether or not the core will execute an instruction. Most instructions have a condition attribute that determines if the core will execute it based on the setting of the conditional flags. Prior to execution the processor compares the condition attribute with
13' conditional flags in cpsr. If they match then the ~~attribute is postfixed~~ instruction is executed else it is ignored. The condition attribute is postfixed to the instruction mnemonic which is encoded into the instruction.

Mnemonic	Name	Condition flags
EQ	Equal	Z
NE	Not Equal	z
CSHS	carry set / unsigned higher or same	C
CCLO	carry clear / unsigned lower	c
MI	minus or negative	N
PL	plus or +ve or zero	n
VS	overflow	V
VC	no overflow	v
HI	unsigned higher	zC
LS	unsigned lower or same	Z or c
GE	signed greater or equal	NV or nv
LT	signed less than	Nv or nV
GT	signed greater than	NzV or nzv
LE	signed less than or equal	Z or Nv or nV
AL	always (unconditional)	ignored

13) Briefly describe the concept of exceptions, interrupts & vector table ?

PTO

⟹ When an exception or interrupt occurs, the processor sets the program counter to a specific memory address.

The address is within a specified address range is called vector table.

The entries in vector table are the instructions that branch to specific routines designed to handle particular exception or interrupt.

The memory map address 0x00000000 is reserved for the vector table, a set of 32 bit words.

On some processors the vector table can optionally be located at higher address in memory starting at 0xffff0000.

When an exception or interrupt occurs, the processor suspends normal execution & starts loading instructions from the exception vector table.

Each vector table entry contains a form of branch instruction pointing to start of a specific routine.

- Reset vector is the location of the first instruction executed by processor when power is applied. This branches to initialization code.

- Undefined instruction vector is used when the processor cannot decode the instruction.

- Software interrupt vector is called when SWI instruction is used as mechanism to invoke an OS routine.

- Prefetch Abort vector occurs when the processor attempts to fetch an instruction from an address without correct access permissions

- Data Abort vectors are raised when an instruction attempts to access data memory without access permissions.

- Interrupt request vector is used by external hardware to interrupt the normal execution flow.

- Fast Interrupt request vector is used ~~by~~ for ~~internal hardware to~~ hardwares requiring

PTO

Exceptions / Interupts	Shorthand	Address	High Address
Reset	RESET	0x00000000	0xffff0000
Undefined Instruction	UNDEF	0x00000004	0xffff0004
Software Interupt	SWI	0x00000008	0xffff0008
Prefetch Abort	PABT	0x0000000C	0xffff000C
Data Abort	DABT	0x00000010	0xffff0010
Reserved	—	0x00000014	0xffff 0014
Interupt Request	IRQ	0x00000018	0xffff0018
Fast Interrupt Request	FIQ	0x0000001C	0xffff001C