

Module -2 Question Bank

1. Write a short note on

a. Design space of processors

b. Instruction pipelines with examples

a. Design space of processors:

Clock rate and CPI

- CPU is driven by a clock with a constant cycle time T .
- The inverse of the cycle time is the clock rate.
- It is measured in megahertz.
- The size of a program is determined by its instruction count (I_c), the number of machine instructions to be executed by the program.
- Different machine instruction requires different number of clock cycles to execute.
- It is easy to determine the average number of CPI for a particular processor if the frequency of occurrence of each instruction type is known.

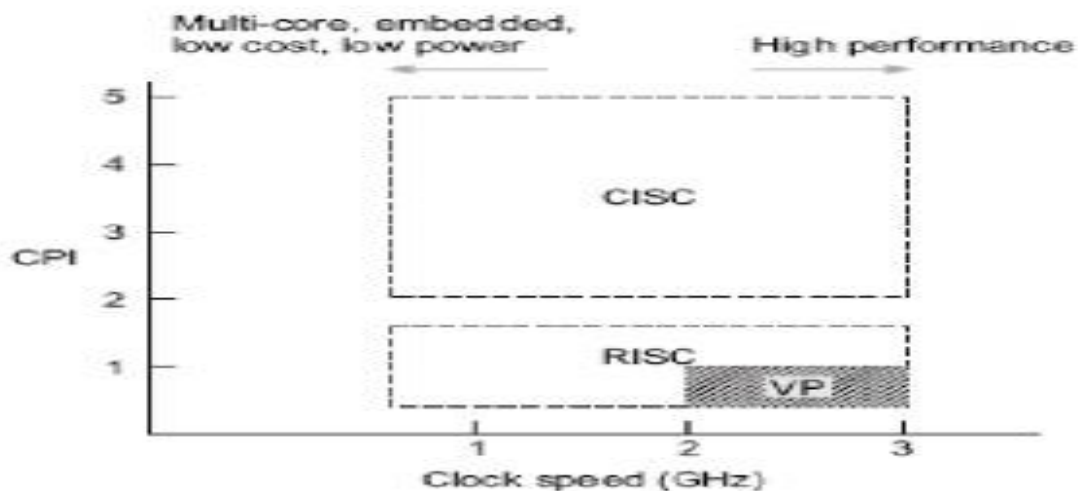


Fig. 4.1 CPI versus processor clock speed of major categories of processors

- Processor designers are trying to lower the CPI rate using hardware and software approaches.
- The complexity of an instruction set is attributed to the instruction formats, data formats, addressing modes, general purpose registers, opcode specifications and flow control mechanism used.

– CISC support

- Different instructions and operand data types.
- many different operand addressing formats and relatively small number of general purpose registers
- Most of the instructions directly match high level language constructions.
- CISC instruction set contains 120 to 350 instructions.
- Processors can be mapped to a space that has clock rate and CPI as coordinates.
- Each processor type occupies a region of this space.
- The term CPI is used with respect to a particular instruction set and a given program mix.

Instruction pipelines

- An instruction pipeline reads consecutive instruction from memory; previous instructions are being executed in other segments.
- Pipeline processing can occur not only in data streams but in the instruction stream as well.
- This causes the instruction fetch and execution phases to overlap and program simultaneously.
- The pipeline must be emptied and all instructions that have been read from memory after the branch instruction must be discarded.
- Typical instruction includes 4 phases
 - Fetch
 - Decode
 - Execute
 - Write back
- These 4 phases are frequently performed in a pipeline or assembly line manner.

Instruction pipeline cycle: The time required for each phase to complete its operation (assuming equal delay in all phases).

Instruction issue latency: The time required between the issuing of 2 adjacent instructions.

Instruction issue rate: The number of instructions issued per cycle.

Simple operation latency: The delay associated with the completion of a simple operation as compared with that of a complex operation.

Resource conflicts: When 2 or more instructions demand use of the same functional units at the same time.

b. Instruction pipelines with examples

- An instruction pipeline reads consecutive instruction from memory, previous instructions are being executed in other segments.
- Pipeline processing can occur not only in data streams but in the instruction stream as well.
- This causes the instruction fetch and execution phases to overlap and program simultaneously.
- The pipeline must be emptied and all instructions that have been read from memory after the branch instruction must be discarded.
- Typical instruction includes 4 phases
 - Fetch
 - Decode
 - Execute
 - Write back
- Pipeline is an implementation technique where multiple instructions are overlapped in execution.
- **Pipeline stage:** The computer pipeline is to divide instruction processing in to stages.
 - Each stage completes a part of an instruction and loads a new part in parallel.
- Throughput of the instruction pipeline is determined by how often an instruction exists in pipeline.
- Pipelining does not decrease the time for individual instruction execution.
- Instead, it increases throughput.
- **Machine cycle:** The time required to move an instruction one step further in the pipeline.
- The length of the machine cycle is determined by the time required for the slowest pipe stage.
- These 4 phases are frequently performed in a pipeline or assembly line manner.

Instruction pipeline cycle: The time required for each phase to complete its operation (assuming equal delay in all phases).

2. Give the architectural distinctions and characteristics of CISC & RISC architectures.

| RISC | CISC |
|--------------------------------------|---------------------------|
| Simple instructions fewer in numbers | Many complex instructions |

| | |
|--|---|
| Fixed length instructions | Variable length instructions |
| Few addressing modes(3-5) | Many addressing modes(12-24) |
| Only load/ store instructions access memory | Many instructions can access memory |
| Complexity in compiler | Complexity in micro code |
| Avg CPI is less than 1.5 | CPI between 2 and 15. |
| Supports large number of general purpose registers | Supports 8 to 24 number of general purpose register |
| Highly pipelined | Less pipelined |

3. Explain VAX8600 processor architecture.

VAX 8600

- Virtual address extension was an instruction set architecture developed by digital equipment corporation (DEC).
- The VAX 8600 implements CISC architecture with micro programmed control.
- It consists of 300 instructions with 20 different addressing modes.
- There are 2 fundamental units for concurrent execution of instruction of integer and floating point instructions.
- There are 16 general purpose registers in the instruction unit instruction pipeline is built with 6 stages.
- The instruction unit prefetches and decodes and handles branching operations.
- The TLB is used in memory for fast generation of the physical address from the virtual address.
- Pipeline concept is used for integer and floating point units.
- **Key features**
 1. A 32 bit CISC instruction set
 2. 4GB virtual memory space
 3. Runs on VMS OS
 4. 16 32 bit registers (r0,r1....r15)
 5. 300+ variable length instructions

6. 22 addressing modes

- **Execution unit:** It is responsible for the overall execution of the machine. It contains most of the micro code of all the complex instructions.
- **Instruction unit:** Performs prefetching and decoding instructions, handles branching operations and supplies operands to the functional units in pipelined manner.
- **Cache memory:** It is of 16kbytes, has virtual address space for data as well as instructions.
- **Memory and I/O control unit:** It performs memory accesses as requested by instruction unit and execution unit. It also has the TLB used for the fast generation of physical address from memory address.

4. Explain MotorolaMC68040 processor architecture.

- The MC68040 is a high performance, 32 bit, 3.3V, static microprocessor that provides a lower power mode of operations.
- A high degree of instructions, execution parallelism is achieved through the use of a full internal Harvard architecture, multiple internal buses and independent execution units.
- It is a 0.8micro m HCMOS microprocessor. It is implemented over 100 instructions using 16 general purpose registers.
- MC68040 processor consist of 4K bytes of data, cache and 4k byte of instruction cache. With separate memory management unit, supported by a address translation cache.
- It is also supports 18 addressing modes, integer unit is organized in 6 stages of instruction pipeline, floating point consist of 3 stage.
- Separate instruction and data buses are provided both addressing and data buses are of 32 bits width.
- The complete memory unit is provided with a virtual demand paged OS. The integer unit which conducts logical and arithmetic operations on the MC 68040V, consists a 6 stage integer execution pipeline.
- The MC 68040V contains independent instructions and data MMUs. Each MMU contains a 64 entry address translation cache used to keep the most recently used translation.
- The full addressing range of the MC68040V is 4 GB.
- The ATCs store recently used logical to physical addresses translation information, as page descriptors, for instructions and data access.
- This cache is 64 entry, 4 ways set associative. Each cache is 4 KB accessed by physical addresses.
- The data cache can be configured as write through or different copy back on page basis.

- This allows for optimizing the system designed for high performance with the deferred copy back writes to system memory.

5. List and brief representative of RISC scalar processors.

- Are designed to issue one instruction per cycle.
- Both RISC and CISC scalar processors should perform about the same if they run with the same clock rate and with equal program length.
- RISC moves less frequent operations in to software , thus dedicating hardware resources to the most frequently used operations.
- RISC scalar processors are
 - SUN SPARC
 - Intel i860
 - Motorola M8810
 - AMD 29000

6. Explain SPARC architecture

- SPARC- Scalable processor architecture
- Formulated at Sun Microsystems in 1985 is based on the RISC I and II design engineered at the university of California at Berkeley from 1980 through 1982.
- SPARC is a CPU ISA, derived from RISC.
- The SPARC architecture is a public property
 - Semiconductor manufacturers are encouraged to produce their own implementation of the SPARC architecture.
- The SPARC processor is divided in to 3 parts
 - Integer Unit(IU)
 - Floating point unit (FPU)
 - Optional coprocessor

7. Differentiate between superscalar and VLIW

| Attributes | Superscalar | VLIW |
|----------------------------|-------------|------|
| Multiple instruction cycle | Y | N |

| | | |
|---|---------------|---------------------|
| Multiple operations/instructions | N | Y |
| Instruction stream passing | Y | N |
| Run time analysis of register dependencies | Y | N |
| Run time analysis of memory dependencies | May be | Commercially |
| Run time instruction reordering | May be | N |
| Run time register allocation | May be | May be |

8. What are the virtual memory models for multiprocessor system?

- Virtual memory is stored in the secondary storage device and helps to extend additional memory capacity.
- Work with primary memory to load applications.
- Reduces the cost of expanding the capacity of physical memory.
- Implementations differ from one OS to other.
- Each process address space is partitioned into parts and used for code, data and stack.
- Parts are loaded into primary memory when needed and written back to secondary storage otherwise.
- The logical address space is referred to as virtual memory.
- Virtual memory is much larger than the physical memory.
- Virtual memory uses: Virtual address and Physical address.
- CPU translates Virtual address to Physical address.
- Virtual memory system uses paging.

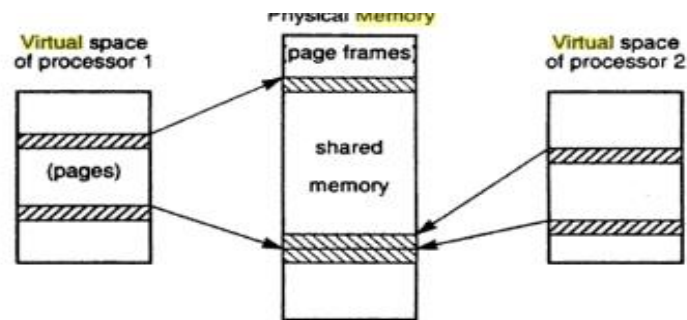
An Abstract Model of Virtual Memory

- The address translation between virtual and physical memory is done by the CPU using page tables which contain all the information that the CPU needs.
- To translate a virtual address into a physical one, the CPU must first work out the addresses virtual PFN and the offset within that virtual page.
- Address space is the amount of memory allocated for all possible addresses for a computational entity, such as
 - a device

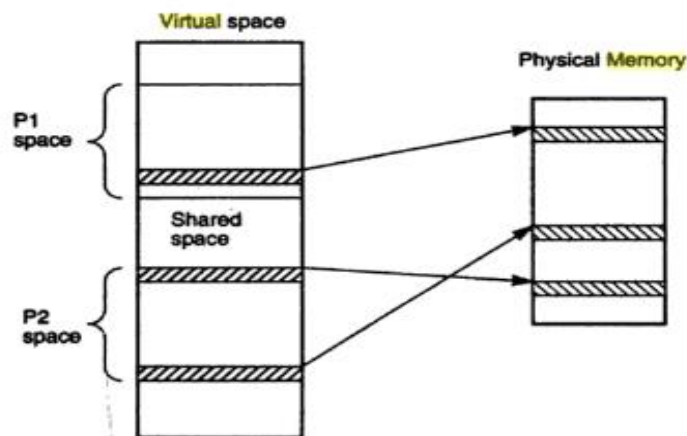
- a file
 - a server
 - a networked computer.
- **Address space** may refer to a range of either physical or virtual addresses accessible to a processor or reserved for a process.
 - A related choice is the granularity of **address mapping**, which is defined as the smallest unit of addressed data (from the persistent store) that can be mapped independently to an area of the virtual address space.

Private virtual memory

- In the full-protection model, each process is given its own private virtual memory, which spans to 2 or 3.5 gigabytes (depending on the CPU).
- This is accomplished by using the CPU's MMU.
- The performance cost for a process switch and a message pass will increase due to the increased complexity of obtaining addressability between two completely private address spaces.



(a) Private virtual memory spaces in different processors



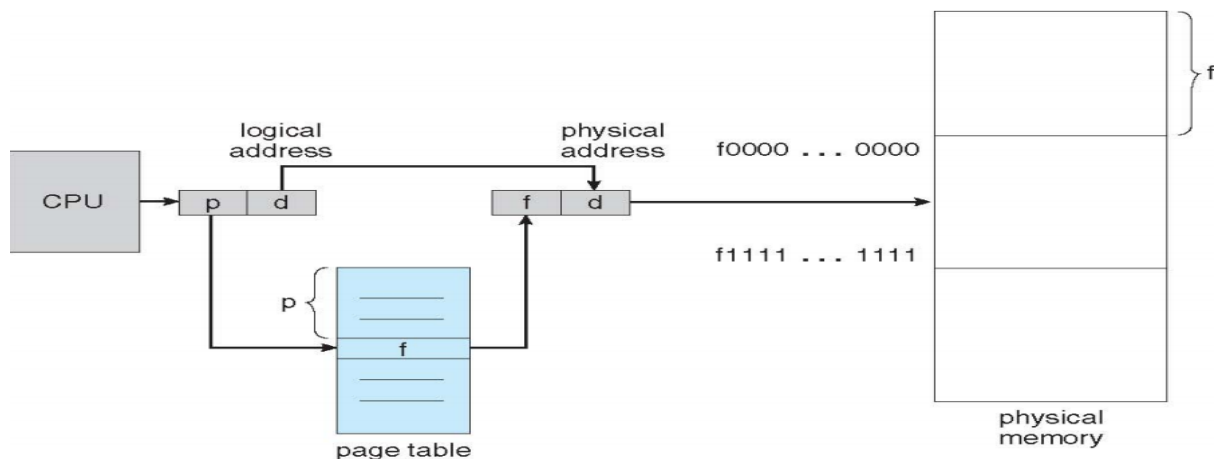
(b) Globally shared virtual memory space

Figure 4.20 Two virtual memory models for multiprocessor systems. (Courtesy of Dubois and Briggs, tutorial, *Annual Symposium on Computer Architecture*, 1990)

9. Explain address translation mechanism using TLB and page table.

- The TLB is used to reduce the time taken to access the memory locations in the page-table method.
- A TLB is a memory cache that stores recent translations of virtual memory to physical addresses for faster retrieval.

Paging hardware

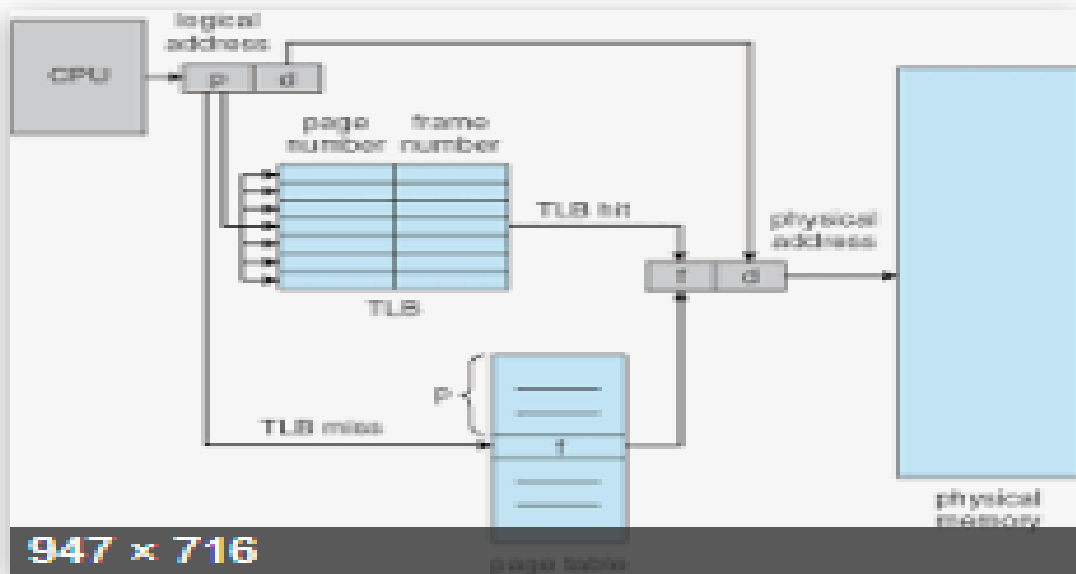


Address Translation Scheme

■ Address generated by CPU is divided into:

- **Page number (p)** – used as an index into a **page table** which contains base address of each page in physical memory
- **Page offset (d)** – combined with base address to define the physical memory address that is sent to the memory unit

| page number | page offset |
|-------------|-------------|
| p | d |
| m - n | n |



TLB

- **TLB** is a memory cache that stores recent translations of virtual memory to physical addresses for faster retrieval.
- When a virtual memory address is referenced by a program, the search starts in the CPU.

Paged memory

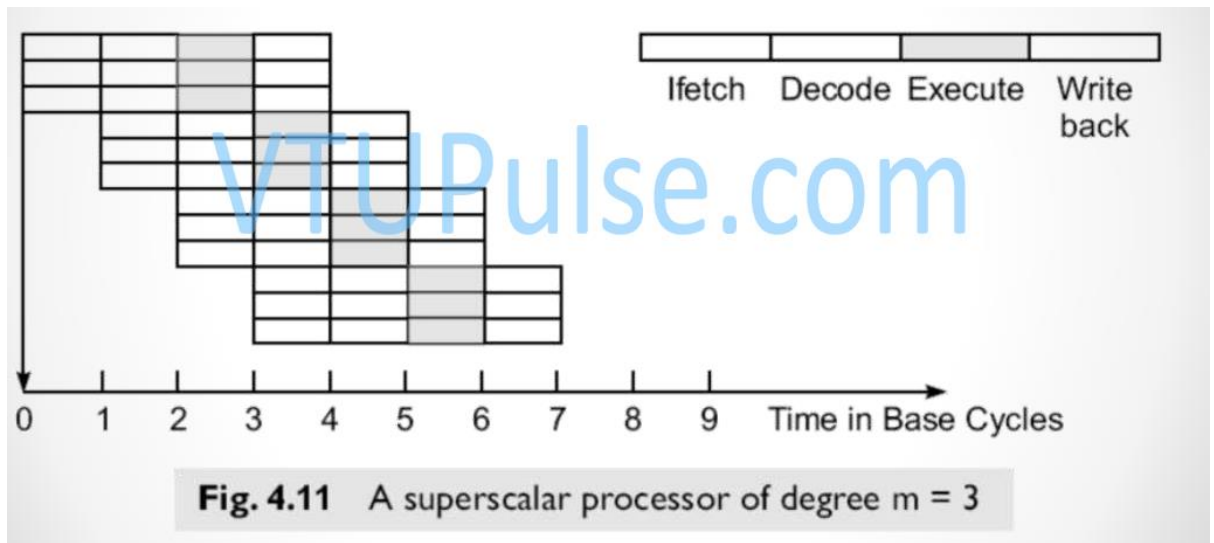
- In computer operating systems, paging is a memory management scheme by which a computer stores and retrieves data from secondary storage for use in main memory.
- In this scheme, the OS retrieves data from secondary storage in same-size blocks called pages.
- Paging is a method of writing data to and reading it from, secondary storage for use in primary storage, also known as main memory.
- Paging plays a role in memory management for a computer's OS.

10. Explain superscalar RISC processor architecture.

Superscalar processor

- Is the ability to initiate multiple instructions during the same clock cycle.
- It fetches and decodes the incoming instructions stream several instructions at a time.
- It exploits the potential of instruction level parallelism.

- Pipelining overlaps various stages of instruction execution to active performance.
- At a high level of abstraction, an instruction can be executed while the next one is being decoded and the next one is being fetched.



- The keys to superscalar execution are
 - IFU that can fetch more than one instruction at a time from cache, instruction decoding logic that can decide when instructions are independent and executed simultaneously and sufficient execution units to be able to process several instructions at one time.
- Ex : IBM RS/6000
- 3 major subsystems of the processor
 - i) Instruction fetch unit(FPU)
 - ii) Integer processor
 - iii) Floating point processor
- IFU 2 stage pipeline
 - i) Packet of 4 instructions is fetched from instruction cache
 - ii) Instructions are routed to the integer processor or floating point processor
- It executes branch instructions.
- There is no overhead from branching.
- IU executed branches while the data units are computing values.
- IU-4 stage pipeline
- It executes data processing instructions and processing for the FPU.

- FPU-6 stages.

11. Explain inclusion, coherence and locality properties.

Inclusion Property

- The inclusion property is stated as $M1 \subseteq M2 \subseteq M3 \subseteq \dots \subseteq Mn$.
- The set inclusion relationship implies that all information items are originally stored in the outermost level Mn .
- During the processing, subsets of Mn are copied into $Mn-1$.
- Similarly, subsets of $Mn-1$ are copied into $Mn-2$, and so on.
- If an information word is found in Mi , then copies of the same word can be also found in all upper levels $Mi+1, Mi+2, \dots, Mn$.
- If a word stored in $Mi+1$ may not be found in Mi .
- A word miss in Mi implies that it is also missing from all lower levels $Mi-1, Mi-2, \dots, M1$.

Coherence Property

- The coherence property requires that copies of the same information item at successive memory levels be consistent.
- If a word is modified in the cache, copies of that must be updated immediately or eventually at all higher levels.
- The hierarchy should be maintained as such.
- Frequently used information is found in the lower levels in order to minimize the effective access time of the memory hierarchy.
- In general, there are two strategies for maintaining the coherence in a memory hierarchy.

Write-through (WT), which demands immediate update in $Mi+1$ of a word is modified in Mi , for $i = 1, 2, \dots, n-1$.

Write-back (WB), which delays the update in $Mi+1$ until the word being modified in Mi is replaced or removed from Mi .

Locality of References/Principle of locality

- The memory hierarchy was developed based on a program behavior known as locality of references.
- Memory references are generated by the CPU for either instruction or data access.

- These accesses tend to be clustered in certain regions in time, space, and ordering.
- In other words, most programs act in favor of a certain portion of their address space at any time window.
- There are 3 dimensions of the locality property:
 - Temporal
 - Spatial
 - Sequential.
- During the lifetime of a software process, a number of pages are used dynamically.
- The references to these pages vary from time to time, however, they follow certain access patterns.
- These memory reference patterns are caused by the following locality properties:

1) Temporal locality

- Recently referenced items (instruction or data) are likely to be referenced again in the near future.
- This is often caused by special program constructs such as iterative loops, process stacks, temporary variables, or subroutines.
- Once a loop is entered or a subroutine is called, a small code segment will be referenced repeatedly many times.
- Thus temporal tends to cluster the access in the recently used areas.

2) Spatial locality

- This refers to the tendency for a process to access items whose addresses are near one another.
- For example, operations on tables or arrays involve accesses of a certain clustered area in the address space.
- Program segments, such as routines and macros, tend to be stored in the same neighborhood of the memory space.

3) Sequential locality

- In typical programs, the execution of instructions follows the program order unless branch instructions create out-of-order executions.
- The ratio of in-order execution to out-of-order execution is roughly 5 to 1 in ordinary programs.
- The access of a large data array also follows a sequential order.

12. Write note on

a. Inverted paging

b. Memory replacement policies

a. Inverted paging

- The direct paging described above works well with a small virtual address space such as 32 bits
- Inverted Page Table structure consists of one page table entry for every frame of the main memory.
- So the number of page table entries in the Inverted Page Table reduces to the number of frames in physical memory
- A single page table is used to represent the paging information of all the processes.

b. Memory replacement policies

- FIFO
- LRU
- Optimal page replacement
- Registers reside directly on the processor chip.
- No latency, referenced directly in instructions.
- Capacity is low
- Number of CPU registers are dependent on the architectural design of the CPU.


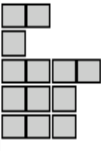

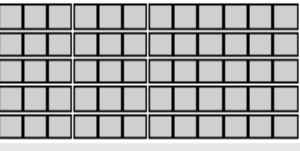
13. Explain Instruction set architecture (ISA)

Instruction set architecture (ISA)

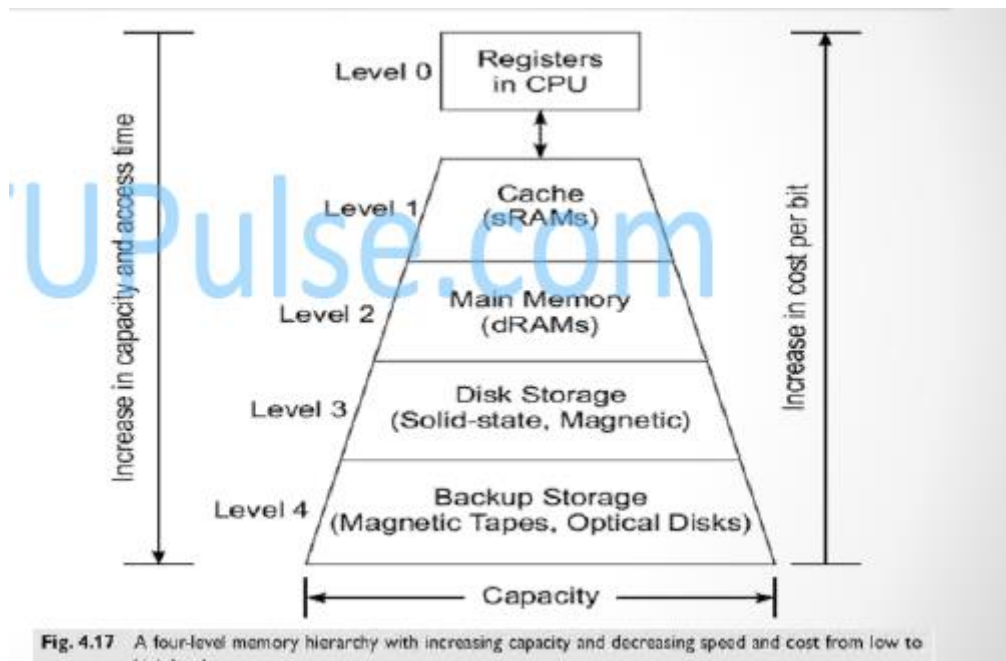
- ISA is the structure of a computer that a machine language programmer or a compiler must understand to write a correct program for that machine.
- Instruction set specifies a processor's functionality
 - What operation it supports?
 - What storage mechanisms it has and how they are accessed?
 - Programmer/compiler uses instruction set to communicate programs to processors.
- ISA is the part of the processor that is visible to the programmer or compiler writer.
- ISA serves as the boundary/ interface between software and hardware.

- Figure: position of ISA
- ISA also provides a mechanism by which the software tells the hardware what should be done.
- The complexity of an instruction set attributes to the
 - instruction formats
 - Data formats
 - Addressing modes
 - General purpose registers
 - Opcode specifications
 - Flow control mechanisms used.

14. Compare VLIW, RISC and CISC

| ARCHITECTURE CHARACTERISTIC | CISC | RISC | VLIW |
|--|---|---|---|
| INSTRUCTION SIZE | Varies | One size, usually 32 bits | One size |
| INSTRUCTION FORMAT | Field placement varies | Regular, consistent placement of fields | Regular, consistent placement of fields |
| INSTRUCTION SEMANTICS | Varies from simple to complex; possibly many dependent operations per instruction | Almost always one simple operation | Many simple, independent operations |
| REGISTERS | Few, sometimes special | Many, general-purpose | Many, general-purpose |
| MEMORY REFERENCES | Bundled with operations in many different types of instructions | Not bundled with operations, i.e., load/store architecture | Not bundled with operations, i.e., load/store architecture |
| HARDWARE DESIGN FOCUS | Exploit microcoded implementations | Exploit implementations with one pipeline and no microcode | Exploit implementations with multiple pipelines, no microcode & no complex dispatch logic |
| PICTURE OF FIVE TYPICAL INSTRUCTIONS  = 1 BYTE |  |  |  |

15. Explain with a neat diagram, Memory hierarchy technology.



In computer architecture, the memory hierarchy separates computer storage into a hierarchy based on response time. Memory hierarchy affects performance in computer architectural design, algorithm predictions and lower level programming constructs involving locality of reference.