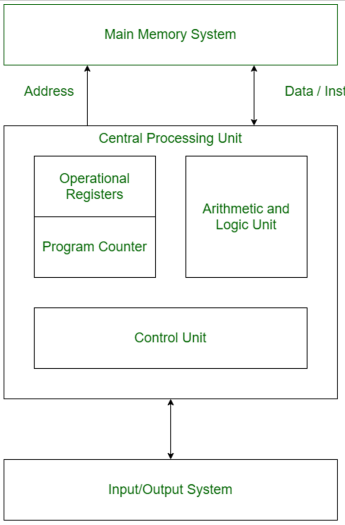
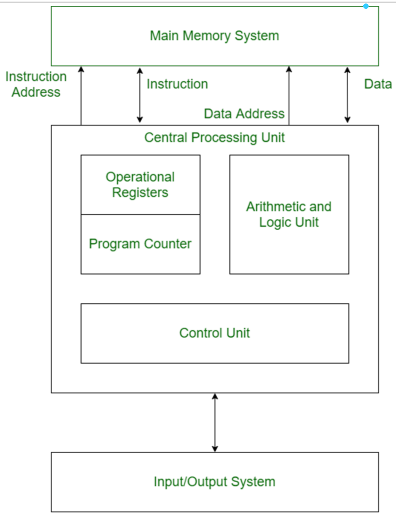
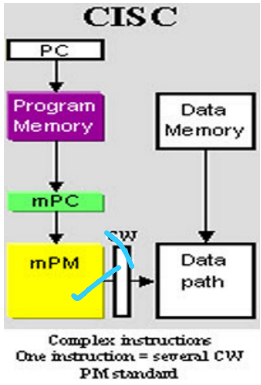
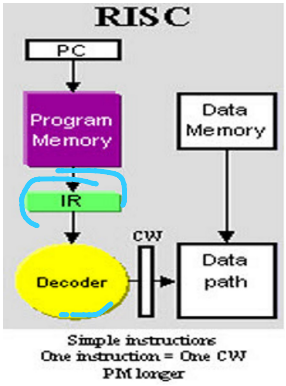
PRINCETON, HARVARD, RISC AND CISC

18 August 2020 22:44

VON NUEMANN ARCHITECTURE Princeton architecture	HARVARD ARCHITECTURE
 Von Neumann Architecture	 Harvard Architecture
1945	1947
Memory, ALU, control unit, input and output connected by buses	Memory, ALU, control unit, input and output Connected by buses
Single memory holds both data and instruction	Separate locations for data and instruction
Serial access of data	Parallel access of data
Simpler since one bus and also cheaper. Control unit development is faster.	Control unit for 2 buses is complicated. It's development takes time and is expensive
Any error can rewrite the program and crash execution	Program can't write itself.
	Both memories can have different size
One bus is a bottleneck.	Free data memory can't be used for instruction
For general purpose, desktops, workstations etc.	For smaller embedded systems, specific purpose
ADVANTAGES: • *Data and instruction are received by CU in same way. Simpler design and development* • *Data from memory or devices accessed same way* • *Memory organisation is in hands of programmer.*	**ADVANTAGES:** • *Parallel access of data* • *Both memories can use different size.*
DISADAVNATGES: • *No parallel execution.* • *One bus is a bottleneck.* • *Instructions can be rewritten.*	**DISADAVNATGES:** • *Free data memory can't be use for instruction or vice-versa.* • *Expensive.*

BASED ON INSTRUCTION SET ARCHITECTURE

CISC	RISC
Complex Instruction Set Computer	Reduced Instruction Set Computer
capacity to perform multi-step operations or addressing modes within one instruction set. The average clock cycle per instruction (CPI) is in the range of 2 and 15.	simple commands that can be divided into several instructions that achieve low-level operation within a single CLK cycle. The average clock cycle per instruction (CPI) is 1.5
	
multi-step processes or addressing modes in single instructions. huge number of compound instructions, which takes a long time to perform.	Each instruction is of a similar length; these are wound together to get compound tasks done in a single operation.
Maximum instructions are finished in two to ten machine cycles.	Most commands are completed in one machine cycle.
pipelining is not easily implemented.	Utilizes pipelining.
Performance is optimized with more focus on hardware.	Performance is optimized with more focus on software.
It has a memory unit to implement complex instructions.	It has no memory unit and uses separate hardware to implement instructions..
It has a microprogramming unit.	It has a hard-wired unit of programming.
The instruction set has a variety of different instructions that can be used for complex operations.	The instruction set is reduced i.e. it has only a few instructions in the instruction set. Many of these instructions are very primitive.
CISC already supports complex addressing modes and can thus be used to represent higher-level programming language statements more efficiently.	Complex addressing modes are synthesized using the software.
Only has a single register set	Multiple register sets are present
They are normally not pipelined or less pipelined	RISC processors are highly pipelined
The complexity lies in the microprogram	The complexity of RISC lies with the compiler that executes the program
Execution time is very high	Execution time is very less
Code expansion is not a problem	Code expansion can be a problem
Decoding of instructions is complex	The decoding of instructions is simple.
It requires external memory for calculations	It does not require external memory for calculations
Examples of CISC processors are the System/360, VAX, PDP-11, Motorola 68000 family, AMD, and Intel x86 CPUs.	The most common RISC microprocessors are Alpha, ARC, ARM, AVR, MIPS, PA-RISC, PIC, Power Architecture, and SPARC.
CISC architecture is used in low-end applications such as security systems, home automation, etc.	RISC architecture is used in high-end applications such as video processing, telecommunications, and image processing.

AVERAGE CLOCK CYCLE: no. of clocks / no of instructions

CONTROL WORD : CONTROL+ SATUS SIGNALS

EXECUTION TIME= CPI X TIME PERIOD OF CLOCK