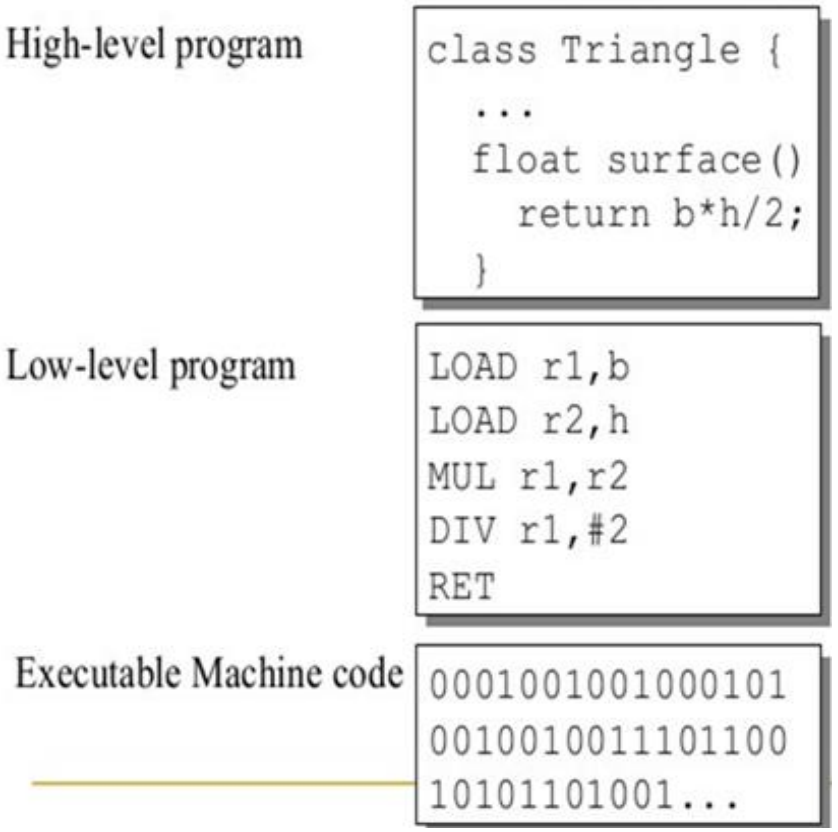
Key Questions

- How to measure performance of a CPU & a computer?
- What are the performance measures used?
- Which is the most acceptable/reliable performance measure? Why?

Factors deciding Performance

- Algorithm
  - Determines number of operations executed
- Programming language, compiler, architecture
  - Determine number of machine instructions executed per operation
- Processor and memory system
  - Determine how fast instructions are executed
- I/O system (including OS)
  - Determines how fast I/O operations are executed



Number of swap operation

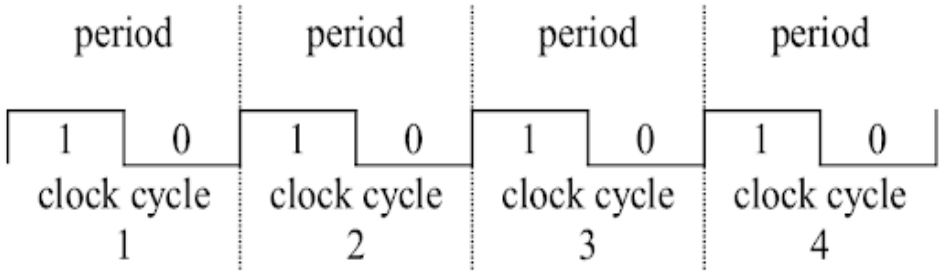
Sort	N=100	N=1000
Bubble Sort	7890	879604
Insertion Sort	3146	243578
Selection Sort	4913	267548

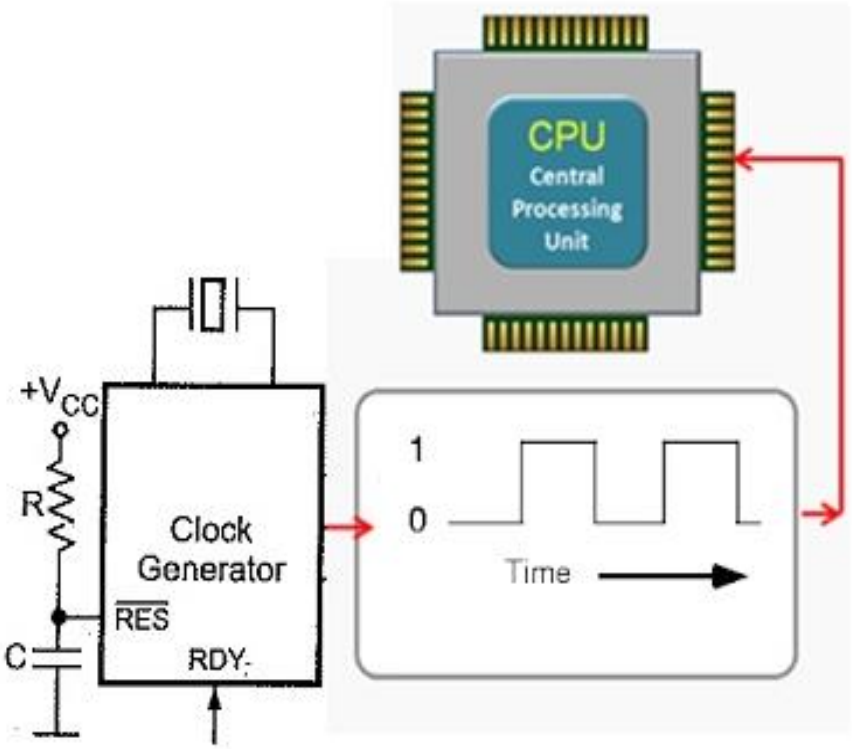
Sorting Algorithm	Time to sort (seconds)			
	100	*1000*	*10000*	*100000*
Bubble Sort	0.06	0.10	7.74	895.7
Insertion Sort	0.09	0.56	5.62	788.4
Selection Sort	0.03	0.34	3.38	491.8

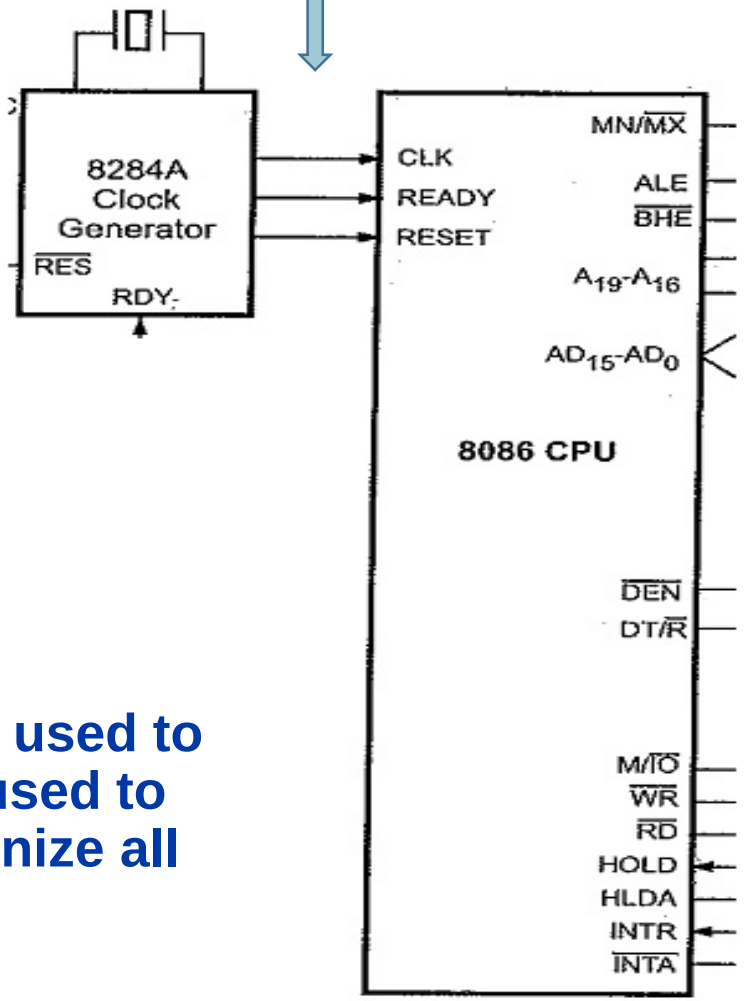
Performance Measures

- Processing time of a single instruction in second
- Processing time of a single instruction in CPU Clock Cycles
- CPU Clock Cycles per Instruction
- CPU Clock Cycles of a Program
- Average CPU Clock Cycles
- Number of Instructions per second
- Processing time of a program
- Processing time of a standard program
- Performance
- Speed-up Factor

CPU CLOCK: A Timing Signal

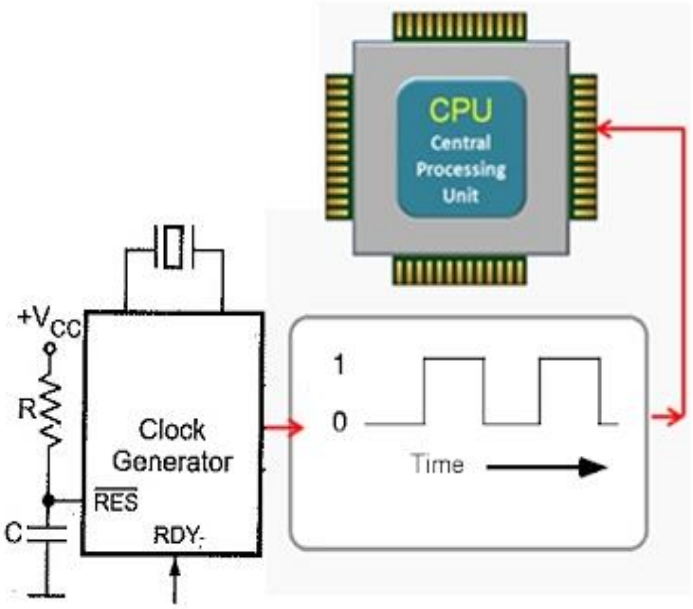


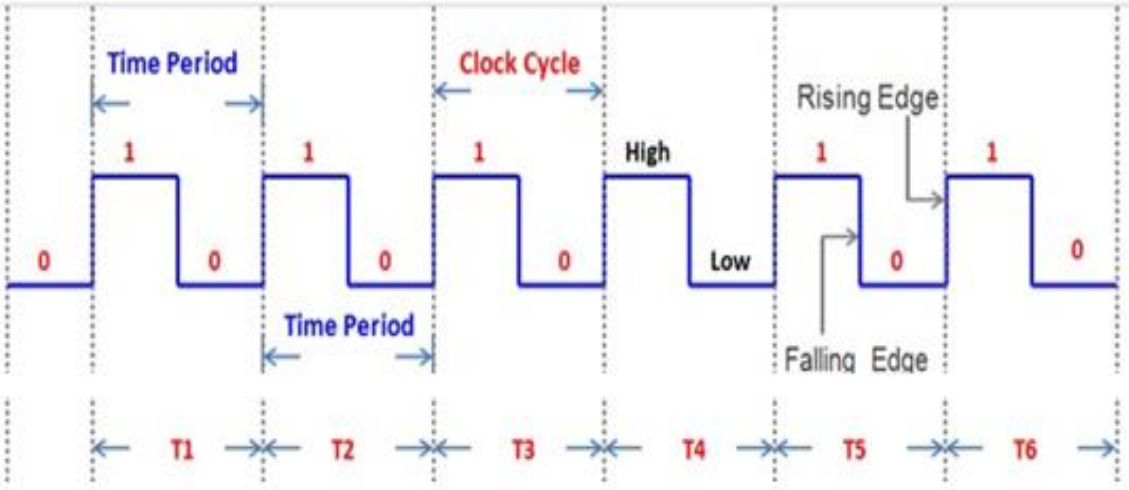




CPU CLOCK: An Electronic Signal used to initiate any basic operation. Also used to sequence next operation. Synchronize all operations

CPU Clocking





Clock signal is defined by two parameters

Clock period: duration of a clock cycle

Measured in Sec, milli-sec, micro-sec, nano-sec, pico-sec etc

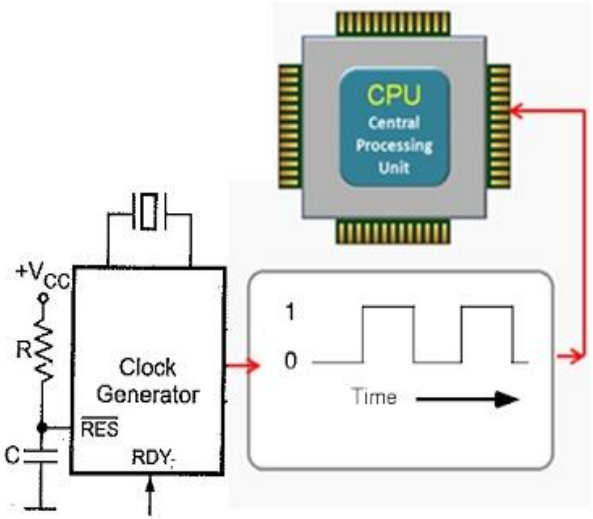
Clock rate or **clock frequency**: clock cycles per second (inverse of clock cycle time)

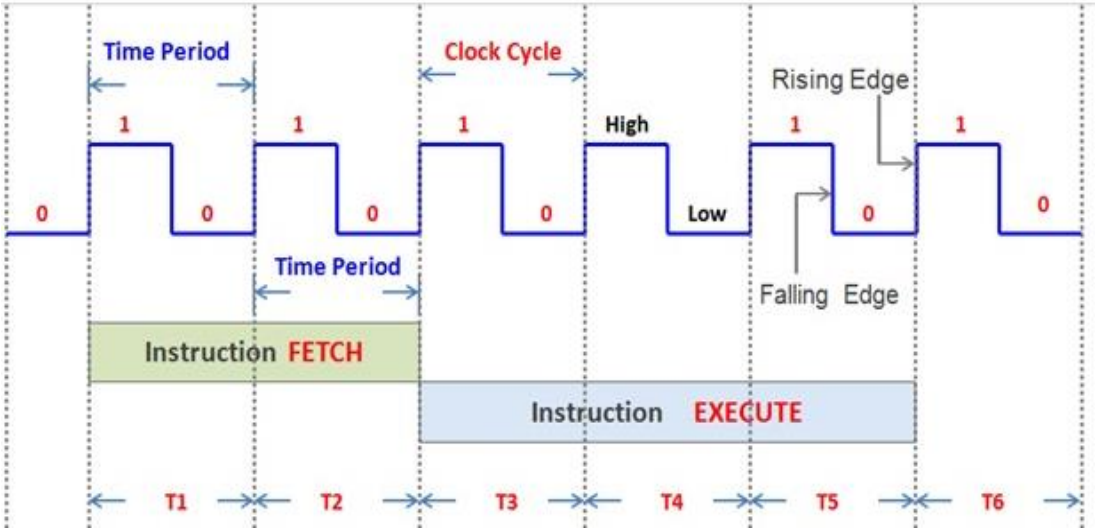
Clock frequency: cycles per second

Measured in Hz, Kilo-Hz(KHz), Mega-Hz(MHz), Giga-Hz(GHz) etc

- Each CPU/Microprocessor requires a highly precise square wave like electronic signal as input, called Clock signal
- This signal is often generated by an external IC and input to processor
- Operation of CPU and computer hardware are strictly controlled by this signal.
- CPU/Computers use this electronic signal to determine when events take place within hardware
- Basic operations and execution of any instruction are controlled by this signal.
- Run time of any instruction or of a program also depend on this electronic signal

CPU Clocking

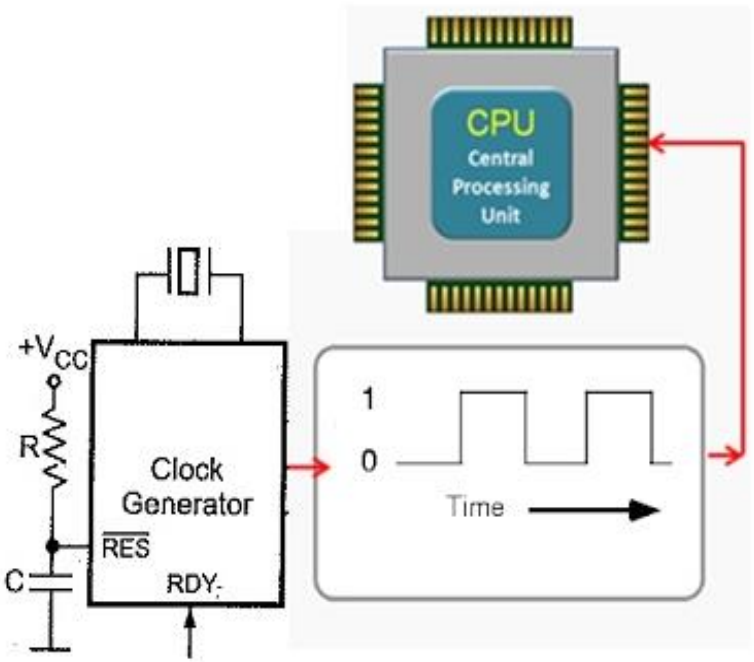


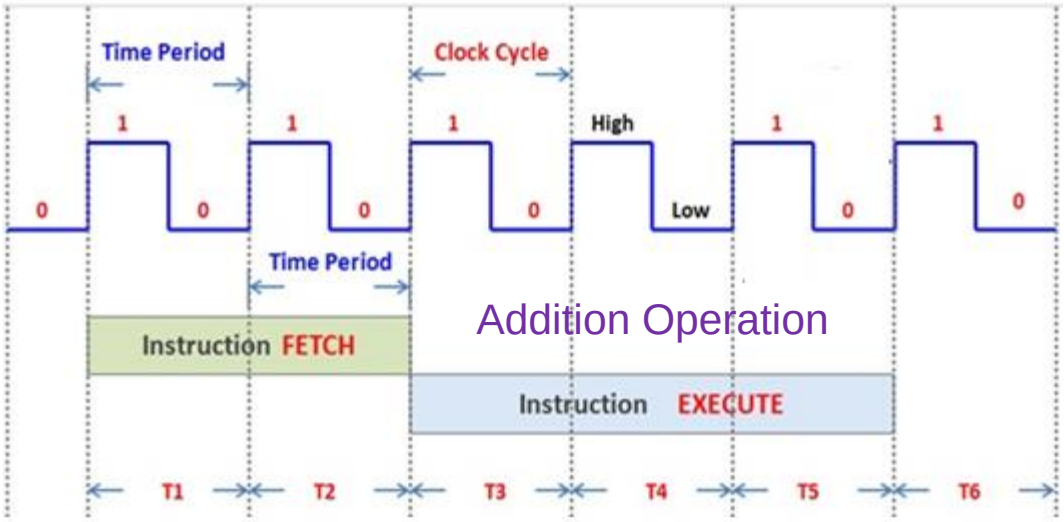


- **If T = 1 Sec ; Clock Frequency = 1/1sec = 1Hz**
- **If T = 1 milli-sec ; Clock Frequency = $1/10^{-3}$ sec = 1000 Hz = 1KHz**
- **If T = 1 micro-sec ; Clock Frequency = $1/10^{-6}$ sec = 10^6 Hz = 1MHz**
- **If T = 1 nano-sec ; Clock Frequency = $1/10^{-9}$ sec = 10^9 Hz = 1GHz**
- **Example:** 3 GigaHertz clock rate means
  clock cycle time = $1/(3 \times 10^9)$ seconds
  = 333 picoseconds (ps)

CPU Time for an Instruction

- Time period: 1ns = 1×10^{-9} s
- Frequency: 1 GHz



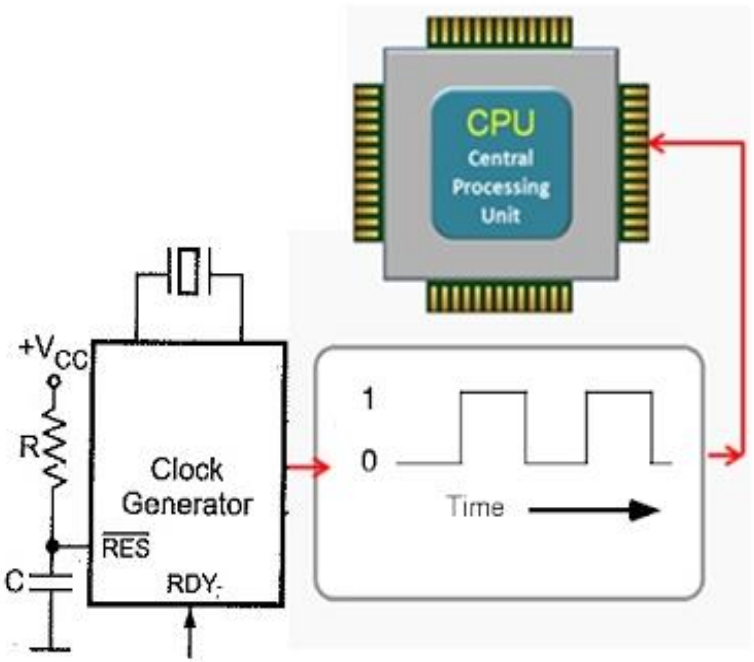


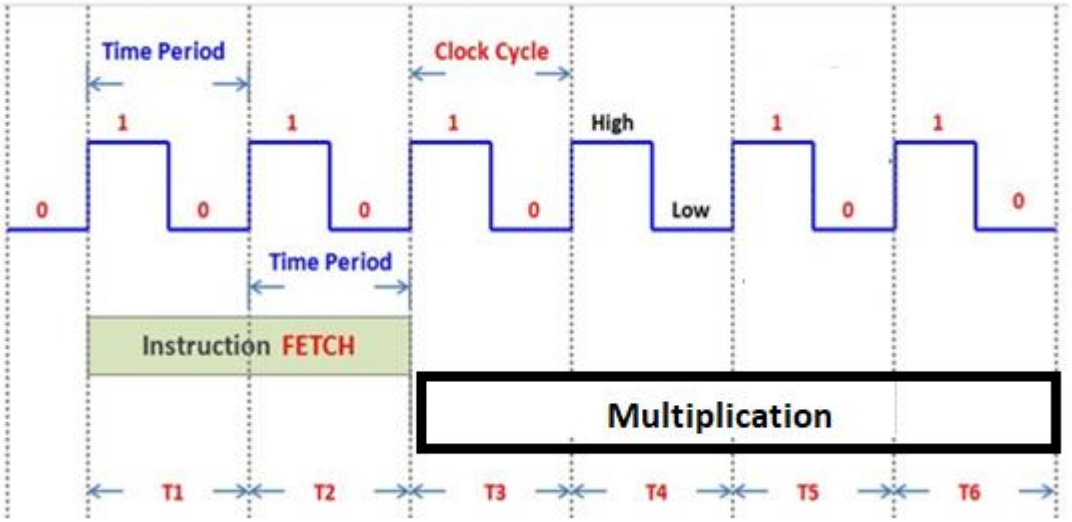
Example: Addition Operation

- CPU Time = 3ns = 3×10^{-9} s
- CPU Time = 3 Cycles per Addition = 3 CPI for Addition
- CPU Time = CPI x Time period of Clock signal **=** 3 x 1ns = 3ns

CPU Time for an Instruction

- Clock period: 1ns = 1×10^{-9} s
- Clock frequency: 1 GHz





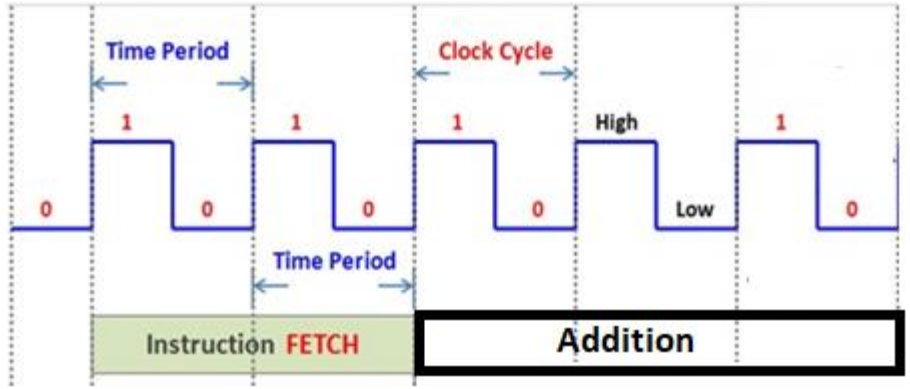
Example: Multiplication Operation

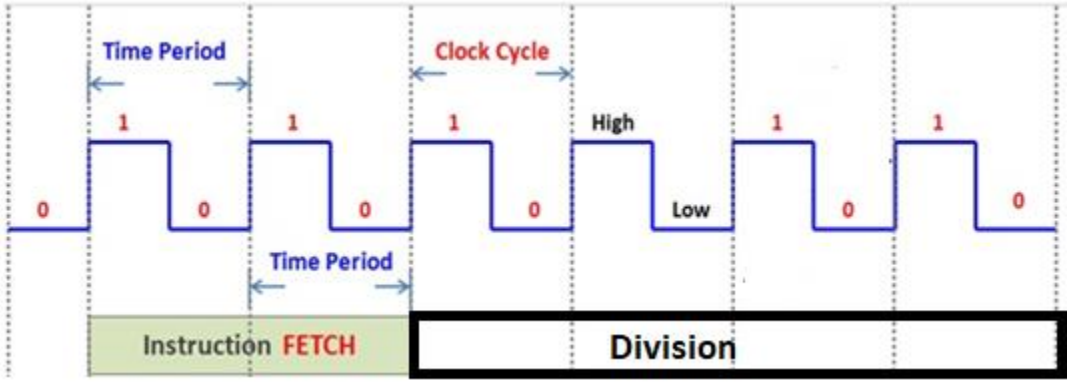
- Time = 4ns = 4×10^{-9} s
- Time: 4 CPU Clock cycles = 4 CPI for Multiplication
- Time = CPI x Clock period = 4 x 1ns = 4ns

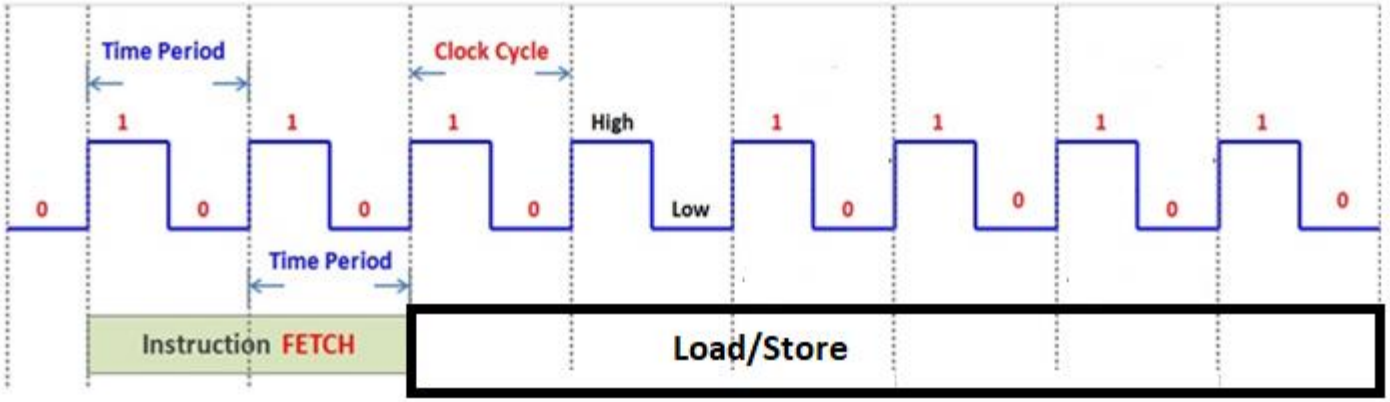
$$\text{CPU Time} = \text{CPU Clock Cycles} \times \text{Clock Cycle Time}$$

$$= \frac{\text{CPU Clock Cycles}}{\text{Clock Rate}}$$

CPI for different Instructions







Clock Cycle:

The clock cycle is the shortest time interval during which a CPU performs a specific operation.
It's the fundamental unit of time for all CPU operations, like a heartbeat for the processor.
Clock speed, measured in Hertz (Hz), indicates how many clock cycles occur per second.
For example, a 2.5 GHz processor has 2.5 billion clock cycles per second.
A higher clock speed generally means the CPU can execute instructions faster.

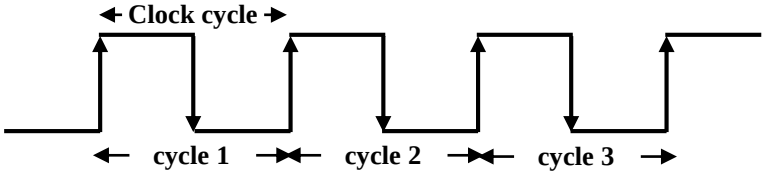
Machine Cycle:

A machine cycle is the time required to complete a single, basic operation within the CPU.
It can be composed of one or more clock cycles.
Examples of machine cycle operations include fetching an instruction from memory or reading/writing data.
The machine cycle is often associated with the steps needed to execute a microcode operation.

Instruction Cycle:

The instruction cycle is the complete sequence of steps needed to fetch, decode, and execute a single instruction from a program.
It typically involves these stages:
Fetch: Retrieving the instruction from memory.
Decode: Interpreting the instruction to understand what operation to perform.
Execute: Performing the operation specified by the instruction.
The instruction cycle can be comprised of multiple machine cycles.
In some architectures, the instruction cycle may also include steps for fetching operands or storing results.

Example: Cycles Per Instruction (CPI)



CPU Clock rate, f =1MHz; Clock Cycle, C = 1 micro second

- **If CPU takes 1 micro second to complete an Instruction, then CPI =1 for that Instruction.**
- **If CPU takes 3 micro seconds to complete another Instruction, then CPI =3 for that Instruction.**

If CPU Clock rate, f =2MHz; Clock Cycle, C = 0.5 micro second

- **If CPU takes 1 micro second to complete an Instruction, then CPI =2 for that Instruction.**
- **If CPU takes 3 micro seconds to complete another Instruction, then CPI =6 for that Instruction.**

Can we find CPI of a Program?

A program contains different types of instructions having different CPI values.

Types of Instruction: Arithmetic, Logical, Data Transfer, Branch etc

Program in Assembly language

```
LOAD R1, A
ADD R1, B
ADD R1, C
MOV R2, R1
OUT 25, R2
```

Given the number of instruction of different types, we can calculate the **Average CPI** of a program.

Average CPI

For a given program executed on a given CPU

$$\text{Avg CPI} = \frac{\text{Total CPU Clock cycles for all instruction of program}}{\text{Instructions count in the program}}$$

Example: A program contains following instructions. CPI for different types of instructions are indicated. Calculate the Average CPI of the CPU for the program.

Instruction type	Instruction Count	CPI
ALU	500	2
Load	200	4
Store	200	4
Branch	100	6

Answer: Avg CPI = (500x2 + 200x4 + 200x4 + 100x6)/1000

Example: Average CPI

A benchmark program is run on a 40 MHz processor. The executed program consists of 100,000 instruction executions, with the following instruction mix and clock cycle count:

INSTRUCTION TYPE	INSTRUCTION COUNT	CYCLES PER INSTRUCTIONS
ARITHMETIC	45000	1
DATA TRANSFER	32000	2
FLOATING POINT	15000	2
CONTROL TRANSFER	8000	2

$$CPI = (45000 \times 1 + 32000 \times 2 + 15000 \times 2 + 8000 \times 2)/100000 = 1.55$$

Example: Average CPI

Consider two different machines, with two different instruction sets, both of which have a clock rate of 200 MHz. The following measurements are recorded on the two machines running a given set of benchmark programs:

MACHINE - A	INSTRUCTION TYPE	INSTRUCTION COUNT	CYCLES PER INSTRUCTIONS
	ARITHMETIC	8	1
	LOAD and STORE	4	3
	BRANCH	2	4
	OTHERS	4	3

.....

MACHINE - B	INSTRUCTION TYPE	INSTRUCTION COUNT	CYCLES PER INSTRUCTIONS
	ARITHMETIC	10	1
	LOAD and STORE	8	2
	BRANCH	2	4
	OTHERS	4	3

Machine-A : Effective CPI: $(8\times1+4\times3+2\times4+4\times3)/18 = 40/18 = 2.22$

Machine-B: Effective CPI: $(10\times1+8\times2+2\times4+4\times3)/24 = 46/24 = 1.92$

CPU Execution Time: The CPU Equation

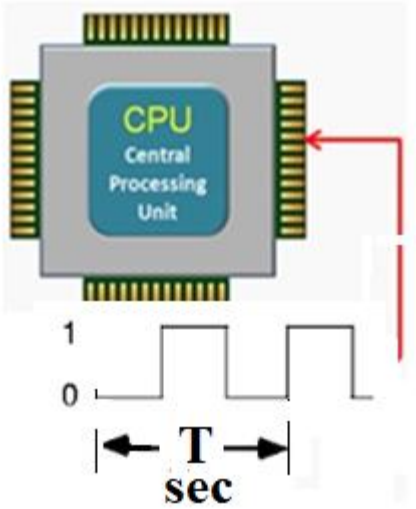
```
1. LOAD r1,b
2. LOAD r2,h
3. MUL r1,r2


I. DIV r1,#2
```

A program is comprised of a I number of instructions
Total number of Instructions = I

The average number of clock *cycles per instruction= CPI*

CPU has a fixed clock cycle time, T = 1/clock frequency



$$\text{CPU time} = \frac{\text{Seconds}}{\text{Program}} = \frac{\text{Instructions}}{\text{Program}} \times \frac{\text{Cycles}}{\text{Instruction}} \times \frac{\text{Seconds}}{\text{Cycle}}$$

$$\text{CPU Time} = I \times CPI \times T$$

execution Time per program in seconds

Number of instructions executed

Average CPI for program

CPU Clock Period

CPU Performance Equation

- For a given program:

$$\textbf{CPU Time} = \textbf{I} \times \textbf{CPI} \times \textbf{T}$$

$$\text{CPU Time} = \text{I} \times \text{CPI} \times \frac{1}{f}$$

CPU Execution Time: Example

- A Program is running on a CPU with the following parameters:
  - Total instruction in program = 10,000,000
  - Average CPI for the program, CPI = 2.5
  - CPU clock frequency, f = 200 MHz.
  - (Time Period, T = 1/f = 5×10^{-9} seconds)
- What is the execution time for this program:

$$\text{CPU time} = \frac{\text{Seconds}}{\text{Program}} = \frac{\text{Instructions}}{\text{Program}} \times \frac{\text{Cycles}}{\text{Instruction}} \times \frac{\text{Seconds}}{\text{Cycle}}$$

CPU time = Instruction count x CPI x Clock period

= 10,000,000 x 2.5 x 1 / frequency

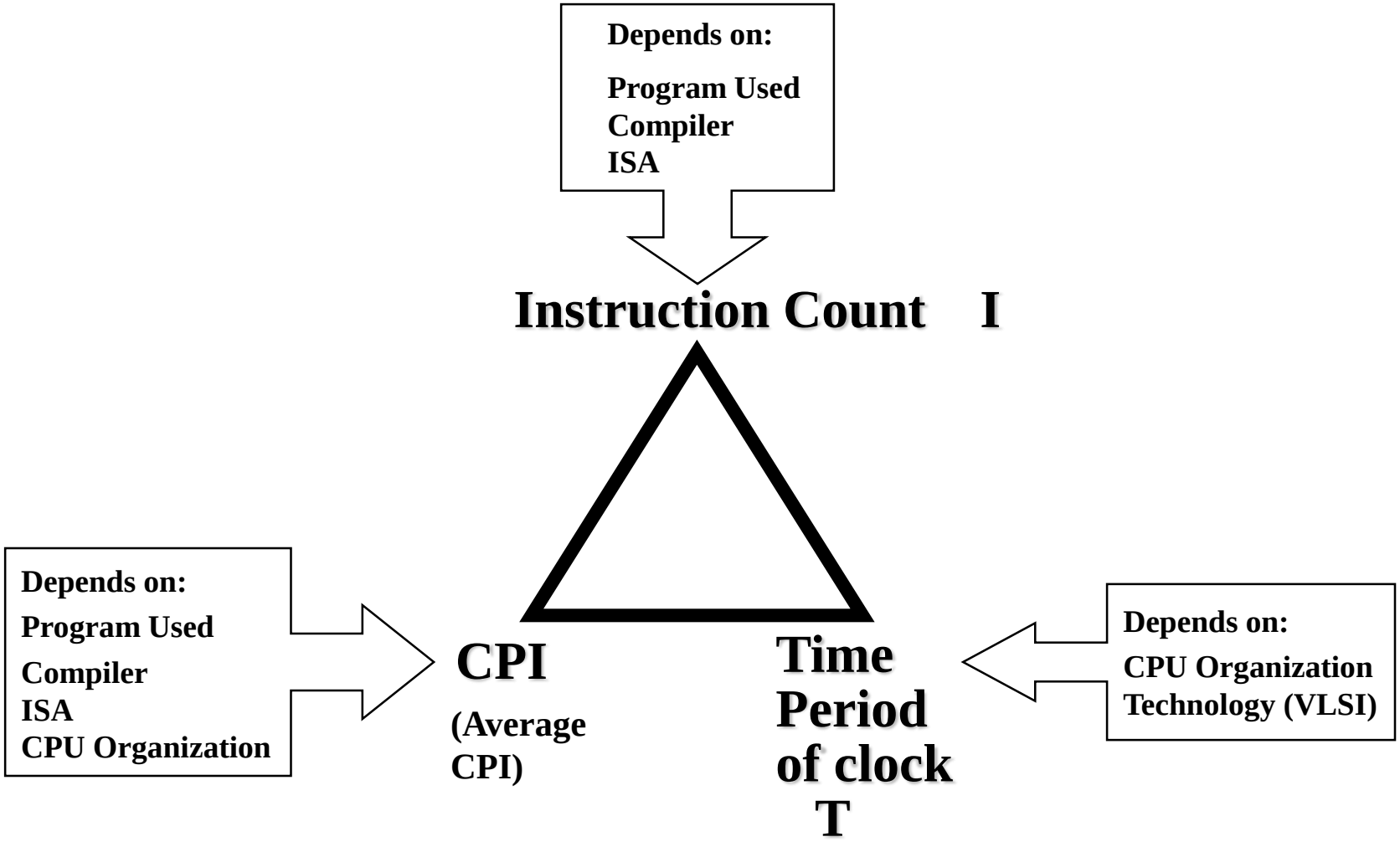
= 10,000,000 x 2.5 x 5×10^{-9}

= 0.125 seconds

Components That Affect Performance Factors

Component (HW/SW)	Factors Affected
Algorithm	Total Instruction Count, CPI
Programming Language	Total Instruction Count, CPI
Compiler	Total Instruction Count, CPI
Instruction Set Architecture	Total Instruction Count, CPI, Clock frequency

CPU Time = Total Instruction x CPI x Clock period



$$\text{CPU time} = \frac{\text{Seconds}}{\text{Program}} = \frac{\text{Instructions}}{\text{Program}} \times \frac{\text{Cycles}}{\text{Instruction}} \times \frac{\text{Seconds}}{\text{Cycle}}$$

	instr. count	CPI	Time Period
Program	X	X	
Compiler	X	X	
Instr. Set Arch.	X	X	
Organization		X	X
Technology			X

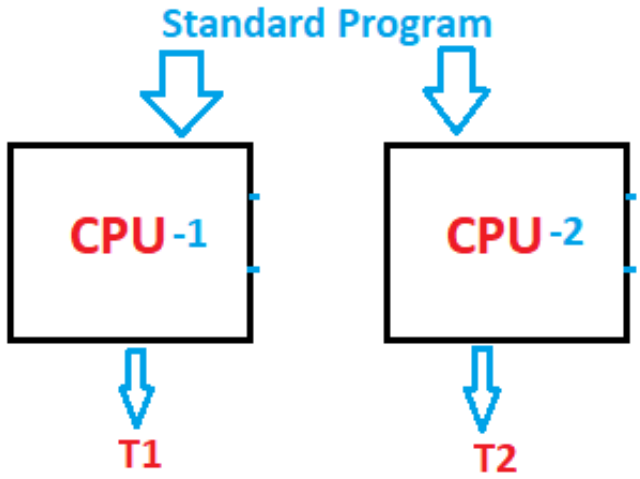
Computer Performance Measures: Program Execution Time

- **How can one measure the performance of CPU running a program?**
  - **CPU is said to be faster or has better performance running this program if the total execution time is shorter.**
  - **Thus the inverse of the total measured program execution time is a possible performance measure or metric:**

Performance = 1 /Execution Time

Measuring CPU Performance

- Execution time
  - Time to complete one task
- Throughput
  - Tasks completed per unit time



$$\text{Performance} = \frac{1}{\text{Execution Time}}$$

$$\frac{\text{Performance}_{(\text{CPU-1})}}{\text{Performance}_{(\text{CPU-2})}} = \frac{\text{Execution Time}_{\text{-T2}}}{\text{Execution Time}_{\text{-T1}}} = N$$

CPU-1 runs N times faster than CPU-2

Comparing Computer Performance Using Execution Time

- To compare the performance of two CPUs "A", "B" running a given specific program:

$$\text{Performance}_A = 1 / \text{Execution Time}_A$$
$$\text{Performance}_B = 1 / \text{Execution Time}_B$$

- CPU A is n times faster than machine B means (or slower? if n < 1) :

$$\text{Speedup} = n = \frac{\text{Performance}_A}{\text{Performance}_B} = \frac{\text{Execution Time}_B}{\text{Execution Time}_A}$$

- Example:

For a given program:

CPU time on A = 1 second

CPU time on B = 10 seconds

$$\text{Speedup} = \text{Performance}_A / \text{Performance}_B = 10 / 1 = 10$$

The performance of CPU A is 10 times the performance of CPU B

Performance Comparison

- A Program is running on a specific machine (CPU) with the following parameters:
  - Total instruction count, I = 10,000,000 instructions
  - Average CPI for the program = 2.5 cycles/instruction.
  - Clock Frequency = 200 MHz.

$T(old) = 1/(200 \times 10^6) = 5 \times 10^{-9}$ seconds

- Using the same program with these changes:
  - A new compiler used: New instruction count, I = 9,500,000

    New CPI: 3.0
  - Faster CPU implementation: New clock frequency = 300 MHz
- What is the speedup with the changes?

$T(new) = 1/(300 \times 10^6) = 3.33 \times 10^{-9}$ seconds

$$\text{Speedup} = \frac{\text{Old Execution Time}}{\text{New Execution Time}} = \frac{I_{old} \times CPI_{old} \times T_{old}}{I_{new} \times CPI_{new} \times T_{new}}$$

$$\text{Speedup} = (10{,}000{,}000 \times 2.5 \times 5 \times 10^{-9}) / (9{,}500{,}000 \times 3 \times 3.33 \times 10^{-9})$$

$$= 0.125 / 0.095 = 1.32$$

Instruction Types & CPI

- **Given a program with *n* types or classes of instructions executed on a given CPU with the following characteristics:**

C_i = Count of instructions of type$_i$ executed

CPI_i = Cycles per instruction for type$_i$ **i = 1, 2, …. n**

Depends on CPU Design

Then:

CPI = CPU Clock Cycles / Instruction Count I

i.e average or effective CPI

Executed

Where:

$$CPU\ clock\ cycles = \sum_{i=1}^{n} \left(CPI_i \times C_i \right)$$

Executed Instruction Count I = $\Sigma\ C_i$

Instruction Types & CPI: An Example

- An instruction set has three instruction classes:

Instruction class	CPI
A	1
B	2
C	3

For a specific CPU design

- Two code sequences have the following instruction counts:

	Instruction counts for instruction class		
Code Sequence	A	B	C
1	2	1	2
2	4	1	1

- CPU cycles for sequence 1 = 2 x 1 + 1 x 2 + 2 x 3 = 10 cycles

  CPI for sequence 1 = clock cycles / instruction count

  = 10 /5 = 2

i.e average or effective CPI

- CPU cycles for sequence 2 = 4 x 1 + 1 x 2 + 1 x 3 = 9 cycles

  CPI for sequence 2 = 9 / 6 = 1.5

$$CPU\ clock\ cycles = \sum_{i=1}^{n} (CPI_i \times C_i)$$

CPI = CPU Cycles / I

Instruction Frequency & CPI

- **Given a program with *n* types or classes of instructions with the following characteristics:**

i = 1, 2, …. n

C_i = Count of instructions of type$_i$ executed

CPI_i = Average cycles per instruction of type$_i$

F_i = Frequency or fraction of instruction type$_i$ executed

= C_i/ total executed instruction count = C_i/ I

Where: Executed Instruction Count I = ΣC_i

Then:

$$CPI = \sum_{i=1}^{n} \left(CPI_i \times F_i \right)$$

i.e average or effective CPI

Fraction of total execution time for instructions of type i = $\dfrac{CPI_i \times F_i}{CPI}$

T = I x CPI x C

Instruction Type Frequency & CPI: A RISC Example

$\frac{CPI_i \times F_i}{CPI}$

Program Profile or Executed Instructions Mix

Base Machine (Reg / Reg)

Depends on CPU Design

Op	Freq, F_i	CPI_i	$CPI_i \times F_i$	% Time
ALU	50%	1	.5	23% = .5/2.2
Load	20%	5	1.0	45% = 1/2.2
Store	10%	3	.3	14% = .3/2.2
Branch	20%	2	.4	18% = .4/2.2
			Sum = 2.2	

Given

Typical Mix

$$CPI = \sum_{i=1}^{n} (CPI_i \times F_i)$$

i.e average or effective CPI

$$CPI = .5 \times 1 + .2 \times 5 + .1 \times 3 + .2 \times 2 = 2.2$$
$$= .5 + 1 + .3 + .4$$

$$T = I \times CPI \times C$$

MIPS (Million Instructions Per Second) Rating

- **For a specific program running on a specific CPU the MIPS rating is a measure of how many millions of instructions are executed per second:**

MIPS Rating = Instruction count / (Execution Time x 10^6)

= Instruction count / (CPU clocks x Cycle time x 10^6)

= (Instruction count x Clock rate) / (Instruction count x CPI x 10^6)

= Clock rate / (CPI x 10^6)

- **Major problem with MIPS rating: As shown above the MIPS rating does not account for the count of instructions executed (I).**
  - **A higher MIPS rating in many cases may not mean higher performance or better execution time. i.e. due to compiler design variations.**
- **In addition the MIPS rating:**
  - **Does not account for the instruction set architecture (ISA) used.**
    - **Thus it cannot be used to compare computers/CPUs with different instruction sets.**
  - **Easy to abuse: Program used to get the MIPS rating is often omitted.**
    - **Often the Peak MIPS rating is provided for a given CPU which is obtained using a program comprised entirely of instructions with the lowest CPI for the given CPU design which does not represent real programs.**

MIPS (Million Instructions Per Second) Rating

- **Under what conditions can the MIPS rating be used to compare performance of different CPUs?**

- **The MIPS rating is only valid to compare the performance of different CPUs provided that the following conditions are satisfied:**

  1 **The same program is used**

  (actually this applies to all performance metrics)

  2 **The same ISA is used**

  3 **The same compiler is used**

  $\Rightarrow$ **(Thus the resulting programs used to run on the CPUs and obtain the MIPS rating are identical at the machine code level including the same instruction count)** **(binary)**

Compiler Variations, MIPS & Performance: An Example

- For a machine (CPU) with instruction classes:

Instruction class	CPI
A	1
B	2
C	3

- For a given high-level language program, two compilers produced the following executed instruction counts:

	Instruction counts (in millions) for each instruction class		
Code from:	A	B	C
Compiler 1	5	1	1
Compiler 2	10	1	1

- The machine is assumed to run at a clock rate of 100 MHz.

Compiler Variations, MIPS & Performance: An Example (Continued)

MIPS = Clock rate / (CPI x 10^6) = 100 MHz / (CPI x 10^6)

CPI = CPU execution cycles / Instructions count

$$CPU\ clock\ cycles = \sum_{i=1}^{n} \left(CPI_i \times C_i \right)$$

CPU time = Instruction count x CPI / Clock rate

- **For compiler 1:**
  - **CPI_1 = (5 x 1 + 1 x 2 + 1 x 3) / (5 + 1 + 1) = 10 / 7 = 1.43**
  - **MIPS $Rating_1$ = 100 / (1.428 x 10^6) = 70.0 MIPS**
  - **CPU $time_1$ = ((5 + 1 + 1) x 10^6 x 1.43) / (100 x 10^6) = 0.10 seconds**
- **For compiler 2:**
  - **CPI_2 = (10 x 1 + 1 x 2 + 1 x 3) / (10 + 1 + 1) = 15 / 12 = 1.25**
  - **MIPS $Rating_2$ = 100 / (1.25 x 10^6) = 80.0 MIPS**
  - **CPU $time_2$ = ((10 + 1 + 1) x 10^6 x 1.25) / (100 x 10^6) = 0.15 seconds**

MIPS rating indicates that compiler 2 is better while in reality the code produced by compiler 1 is faster

MFLOPS (Million FLOating-Point Operations Per Second)

- A floating-point operation is an addition, subtraction, multiplication, or division operation applied to numbers represented by a single or a double precision floating-point representation.
- MFLOPS, for a specific program running on a specific computer, is a measure of millions of floating point-operation (megaflops) per second:

$$\text{MFLOPS} = \text{Number of floating-point operations} \;/\; (\text{Execution time} \times 10^6)$$

- MFLOPS rating is a better comparison measure between different machines (applies even if ISAs are different) than the MIPS rating.
  - Applicable even if ISAs are different
- Program-dependent: Different programs have different percentages of floating-point operations present. i.e compilers have no floating- point operations and yield a MFLOPS rating of zero.
- Dependent on the type of floating-point operations present in the program.
  - Peak MFLOPS rating for a CPU: Obtained using a program comprised entirely of the simplest floating point instructions (with the lowest CPI) for the given CPU design which does not represent real floating point programs.