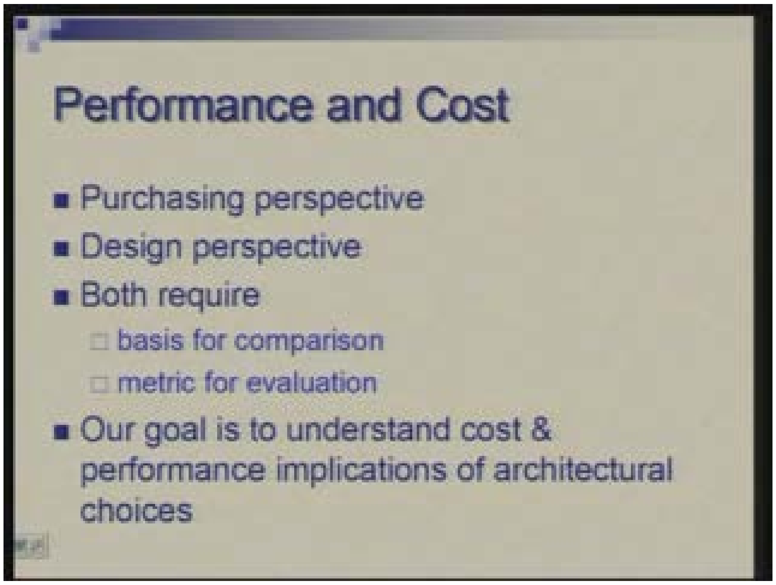
Computer Architecture

Prof. Anshul Kumar
Department of Computer Science and Engineering
Indian Institute of Technology, Delhi

Lecture - 9
Performance

When we see several computers around in the market the natural question which comes up is which is better and which is worse; particularly in terms of performance which performs better and which performs poorly. Also, as a designer if you look deeper into how instructions are organized and how computer is designed the question is that with so many different alternatives which alternative is better from performance point of view. So, in this lecture and the next one we are going to look at the issue of performance; we will define what exactly the term means and what is the relationship between performance and architectural choices.

(Refer Slide Time 1:38)

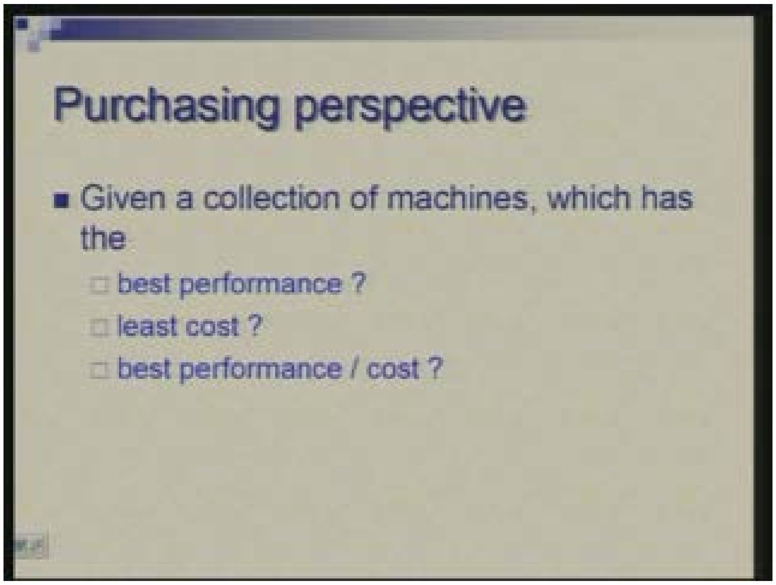


Performance and Cost

- Purchasing perspective
- Design perspective
- Both require
  - □ basis for comparison
  - □ metric for evaluation
- Our goal is to understand cost & performance implications of architectural choices

So the issue of performance….. the question can be asked from a user's perspective. The user may like to know about the performance of the computer which is available and the designer, on the other hand, may try to see from the point of view of design alternatives what is the best. So, for both these perspectives what you need is a basis for comparison; how do you compare one versus other, one computer with the other computer or one deign choice with other design choice and this has to be a quantitative metric so you should be able to say that A is two times better than B or three times better than C and so on. So quantification is a must and therefore a precise definition is required. So although initially I will talk of user's perspective as well as designer's perspective but particularly what is relevant for this course is that we try to understand the relationship between performance and the architecture.

(Refer Slide Time 2:54)



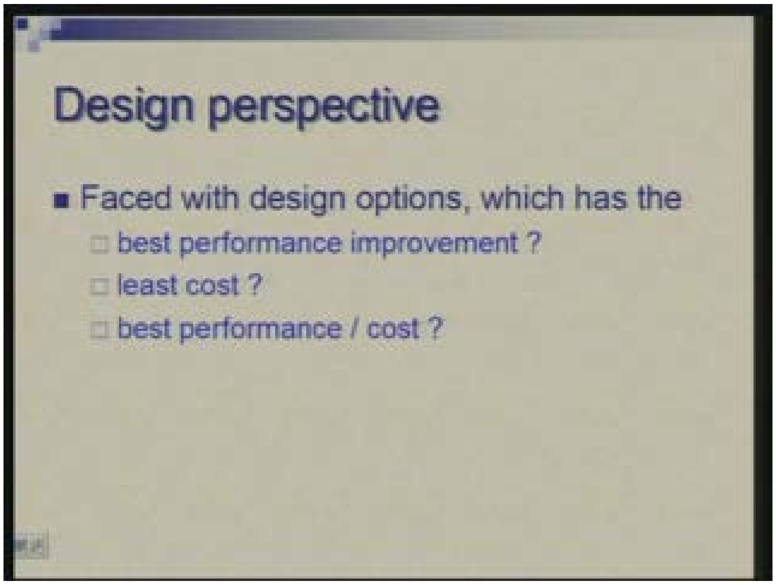
Purchasing perspective

- Given a collection of machines, which has the
  - □ best performance ?
  - □ least cost ?
  - □ best performance / cost ?

So, from user's perspective or a purchaser's perspective suppose you want to buy a computer system for your lab or your lab organization or for yourself as a personal computer you would find that there is a lot of variety. there are different manufacturers, different vendors and for each vendor you will have a set of choices, there is a range of machines so the question you would like to ask his how do they compare performance-wise across the vendors or within the machines of the same vendor and what is the cost implication because possibly as it is intuitive you may not get performance without any cost and which indeed is so that performance and cost may often be having some tradeoff so you put in more cost you can get better performance or you want to save money you will have to settle with the lower performance. So is there a way of putting these two together; you may like to ask the question of what is the best performance for a given rupee; what is the performance price ratio; so which alternative gives you best performance for the same price; so there are different ways in which you can pose this question.

On the other hand, from designer's perspective, we have seen that, first of all, there are so many different types of instruction sets. We have talked about some major styles: load store style or memory memory style, stack style, accumulator type of machine so there are different philosophies of designing an instruction set and also something which we have not studied; given an instruction set there may be many different ways in which you can build the hardware. So you can do the instructions in such a manner, you can execute them in a manner that they take less time or more time and there are implications of those choices on cost and performance.

(Refer Slide Time 5:11)

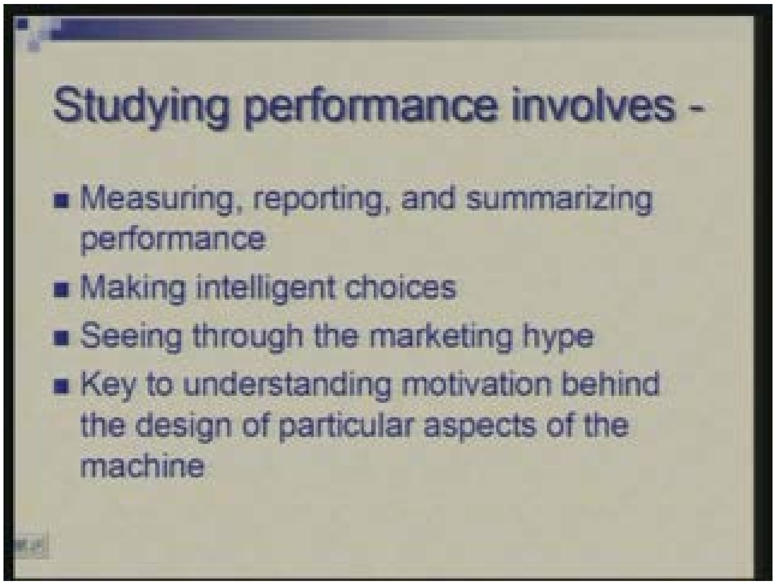


Design perspective

- Faced with design options, which has the
  - □ best performance improvement ?
  - □ least cost ?
  - □ best performance / cost ?

So once again among the different options which you are considering at any stage either at instruction design stage or at hardware stage what gives best performance and what is the cost implication. You can similarly ask the question of performance for a given price. You may fix the price and see what gives you best performance or for a given performance which is a cheapest option so you could work it out in different ways. so the study of performance has several aspects: one is the question of… of course you need to define what performance means in very quantitative and precise terms and there should be a way of measuring the performance; it should be a measurable quantity; so how to measure it, how to report it, how to summarize it because measurement may involve may be several experiments and can you come up with a summary number a number which summarizes all that you do as an experiment.

(Refer Slide Time 5:52)

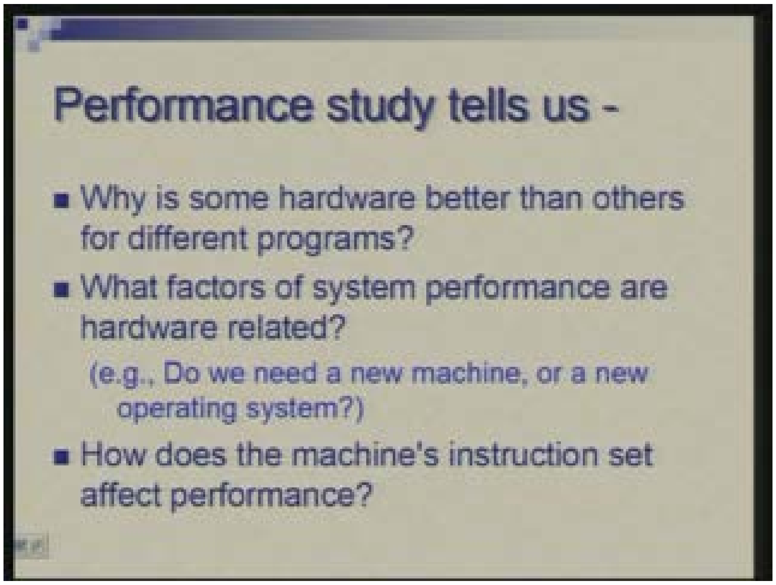


Studying performance involves -

- Measuring, reporting, and summarizing performance
- Making intelligent choices
- Seeing through the marketing hype
- Key to understanding motivation behind the design of particular aspects of the machine

Therefore, based on all this how to make good choices. As I mentioned there are two issues involved: performance and cost so you have to make a judgment on which is the best alternative. you might see that people who are trying to market a system may try to project certain things so there may be hype about this machine does this does this and this is a performance number but you have to see does this match with your definition and your definition may possibly take into account your requirements so you have to have an evaluation or a definition which actually reflects the way performance is going to affect you particularly. So this understanding of performance: method of measurement, method of summarizing, method of comparing is important from the point of view of understanding various design choices which you will study as you go along.

(Refer Slide Time 7:47)



Performance study tells us -

- Why is some hardware better than others for different programs?
- What factors of system performance are hardware related?

  (e.g., Do we need a new machine, or a new operating system?)
- How does the machine's instruction set affect performance?

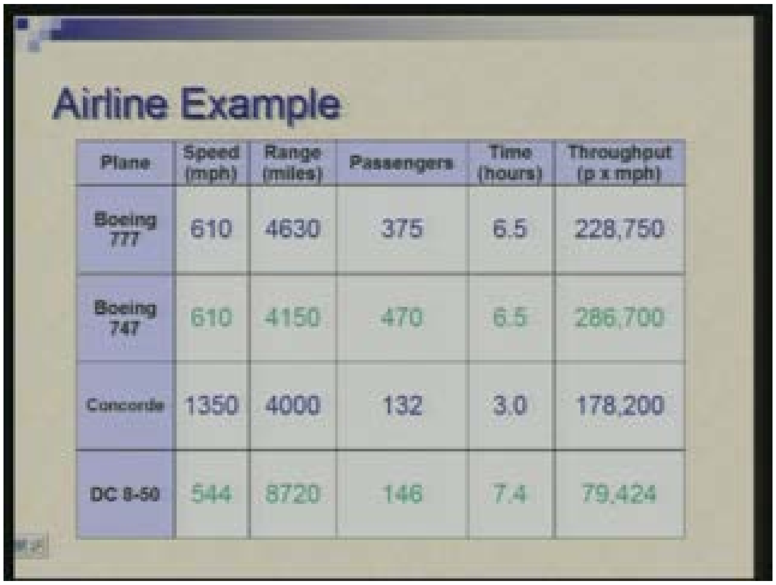
So we would like to understand why certain piece of hardware or certain design choice performs better than the other and we will also see that it may depend upon the program. It may happen that A performs better than B for a given program but for another program it may better. So, for example, you might find that when you are doing word processing A is better than B, when you do emailing B is better than A so those things may happen.

Which factors influencing the performance are related to hardware design or which performance factors are related to architectural issues. There may be factors which are also beyond this beyond the processor design in particular so we would like to get some idea of that also.

What is the influence of instruction set on the performance? Including or not including certain instruction does it have influence on performance or instruction style has some influence on performance so all these questions are there and we should develop certain understanding of these questions.

So, to bring some basic points into attention let me take an example from a different domain. We are talking of although computer performance but the issue of performance could be seen in daily life in many different context and you would see that there are parallels between what we see in this problem and what we would see in computers. So here is some data about a set of aircrafts.

(Refer Slide Time 9:36)



Airline Example

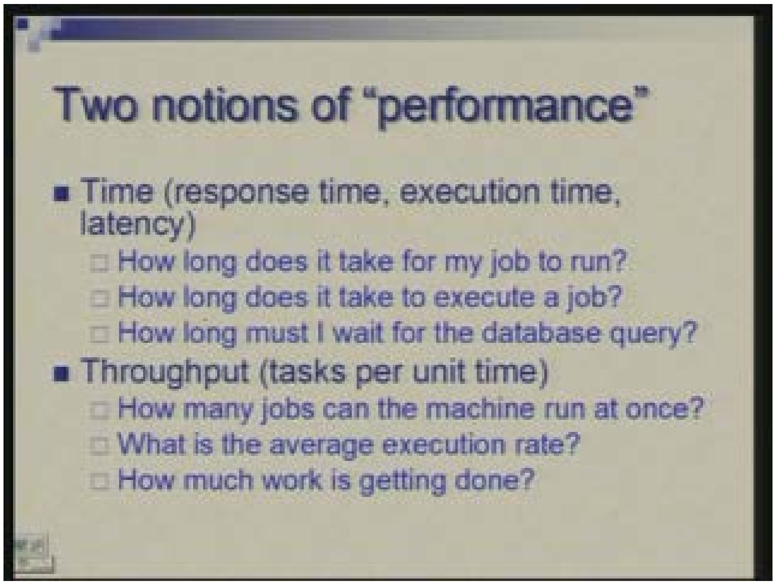
Plane	Speed (mph)	Range (miles)	Passengers	Time (hours)	Throughput (p x mph)
Boeing 777	610	4630	375	6.5	228,750
Boeing 747	610	4150	470	6.5	286,700
Concorde	1350	4000	132	3.0	178,200
DC 8-50	544	8720	146	7.4	79,424

The first column shows set of aircrafts which for example an airline company may be debating as to which one we should buy and the choice may have to be declared by some kind of performance comparison. So the first three columns gives certain data; ignore the last two columns for the moment. the second column says what is the speed of the aircraft expressed in miles per hour here ,what is the range in terms of miles for which the aircraft can go before refueling; the third is the carrying capacity in terms of number of passengers. If you look at these three parameters you would find that it is not a very simple matter of telling which has the best performance. If you simply go by speed Concorde has the maximum speed. If your target was speed then Concorde is the plane you must buy. If you want to look at for example the range; you are interested in long flights without any stoppage in between; suppose you want to connect two airports which are 8000 miles apart then your choice is DC 8-50 others cannot give you a nonstop flight for that distance. Then, on the other hand, if you are talking of passenger capacity you want to carry let us say 450 passengers at a time then your choice is Boeing 747; for the others you will have to make two trips to do the same thing.

This is just to indicate that depending upon how you fix your targets, what is your area of interest, what is that you want to achieve, what your application is, you might find that the performance would be different that with one given criteria A could perform better, with other given criteria B could perform better. So, in this particular column of example you are seeing that time taken in hours to connect two points, let us say, across the Atlantic which are, say about 4000 miles apart then the number of hours you would take is given here. This is another measure which is nothing but inversely proportional to the speed so again Concorde is the best one here.

You could also look at something like throughput. Number of passengers carried into miles per hour. So it is a kind of composite measure of the carrying capacity and the speed so if you multiply these two, the figures you get are shown here and from this point of view (Refer Slide Time: 12:48) Boeing 747 is the best; it is a combination of capacity and the speed.

(Refer Slide Time 13:05)



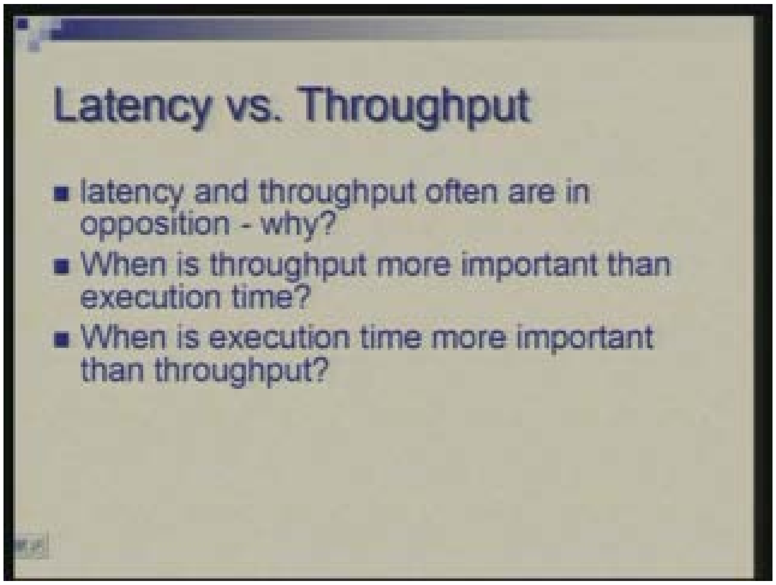
Two notions of "performance"

- Time (response time, execution time, latency)
  - □ How long does it take for my job to run?
  - □ How long does it take to execute a job?
  - □ How long must I wait for the database query?
- Throughput (tasks per unit time)
  - □ How many jobs can the machine run at once?
  - □ What is the average execution rate?
  - □ How much work is getting done?

So another thing which this for example brings to our notice is that there are multiple criteria; multiple machines of performance. Particularly from the context of computers there are two notions which we would be talking of: one is the time, so in aircraft context it is the travel time. So, for computer time would mean response time or execution time or latency. So, for example, to be more precise, you may ask a question how long does it take for my job to run. So, moment I say run it takes certain amount of time and that is what is of concern to me. The other question could be how long does it actually take to run how much time the processor has actually taken to run. the first one is the time I see on the wall clock, the second one would be the time machine has taken to execute the program, third would be, for example, in an interactive environment let us say database query or ATM type of environment where you give a command or you make a request and you get a response so what is the response time.

In different context you may like to ask questions you may like to word it differently but you are talking of basically time in seconds or minutes or hours or whatever it is. The other type of measure is called throughput. so again it say generic term where we are trying to talk of the overall work which is being carried out, the rate at which work is being carried out, the number of tasks per unit time, how many jobs can the machine run at once, what is the average execution rate, how much work is getting done. So you as an individual user may look at time as an important factor. But let us put ourselves in the shoes of the computer system manager who is catering to a user community and the concern there would be how many user programs are being run every hour. it is secondary that a user may have to wait or user may get immediate response so each individual…… user may see his or her own response time but as a manager of a service one would like to see how many programs are being executed by a collection of computers or a single computer as the case is per hour or per day or whatever the unit of time you choose to be.

(Refer Slide Time 16:02)

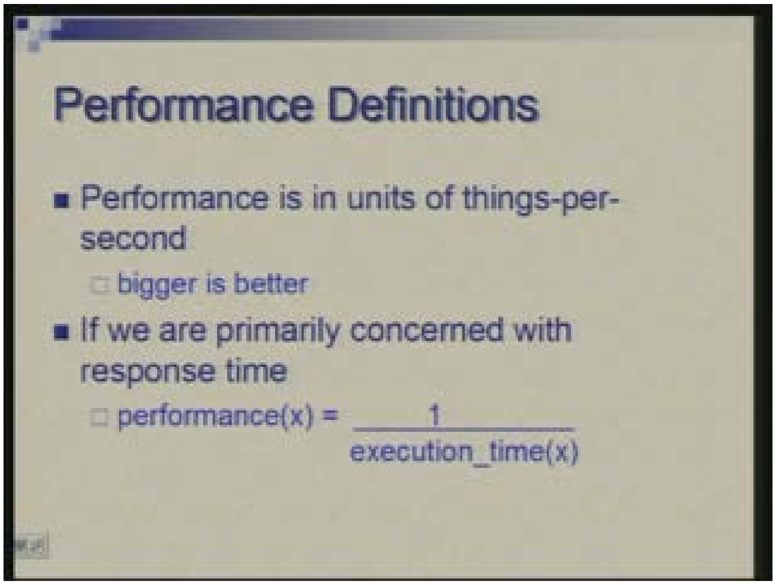


Latency vs. Throughput

- latency and throughput often are in opposition - why?
- When is throughput more important than execution time?
- When is execution time more important than throughput?

These are two measures which sometime may go together or sometime they may contrast with each other. You may, for example, find that in order to get high throughput some waiting times are introduced so some people have to wait; while others may get quick access but some may have to wait and may be different people have to wait at different times but on the whole your attempt to keep the machine busy so that may maximize throughput. But as an individual you may like immediate attention and you may like to get complete grasp of the resource so it may not necessarily lead to better throughput. But of course there are, suppose you replace all computers by better computers it may improve response time it may also improve throughput. So some changes or some policies or some design choices may help both whereas others may help one at the cost of the other. So one has to keep in mind what is objective. So, the question may be when throughput is more important than execution time and vice versa; it depends upon perhaps the person who is asking the question.

(Refer Slide Time 17:33)

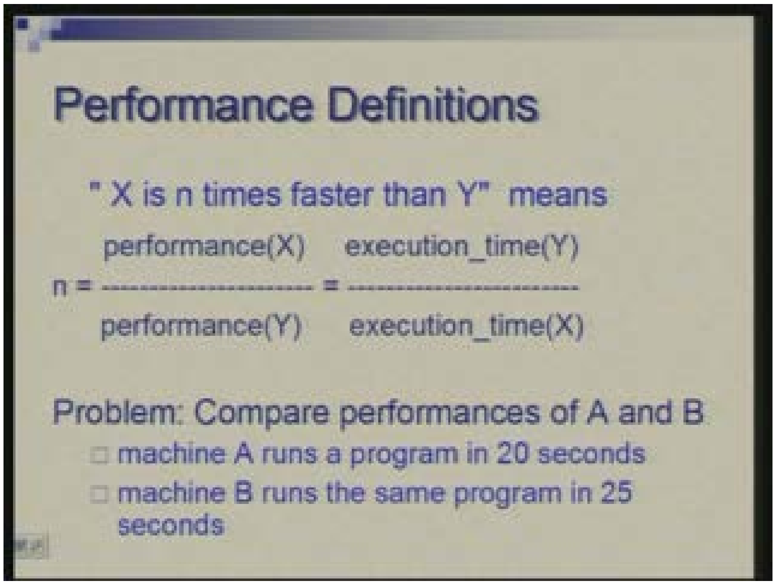


Performance Definitions

- Performance is in units of things-per-second
  - □ bigger is better
- If we are primarily concerned with response time
  - □ performance(x) = $\frac{1}{\text{execution\_time(x)}}$

Now let us try to define performance. We will try to focus on the time aspect and not the throughput aspect. We will predominantly talk of the individual concern of response time or execution time. We want to define performance in a manner that a bigger number a larger number represents better performance. So we could define it as reciprocal of execution time. So we say that performance is one upon execution time. Performance of let us say machine x is one upon execution time for x so you take a machine, you take a program, see how long it takes, measure it and then say, reciprocal of that is your performance number. So it is a very simple and straightforward definition and this can be used to also talk of relative performance.

(Refer Slide Time 18:32)



Performance Definitions

" X is n times faster than Y" means

$$n = \frac{\text{performance(X)}}{\text{performance(Y)}} = \frac{\text{execution\_time(Y)}}{\text{execution\_time(X)}}$$

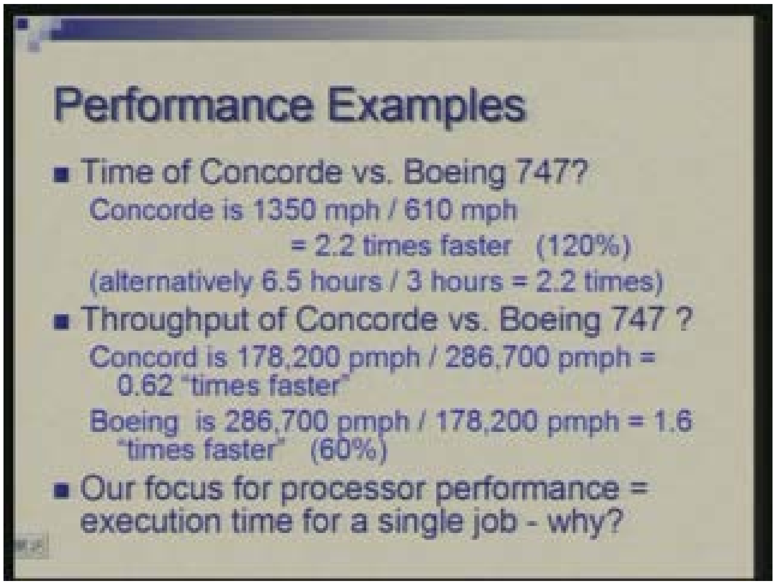
Problem: Compare performances of A and B

- □ machine A runs a program in 20 seconds
- □ machine B runs the same program in 25 seconds

So you want to say x is n times faster than y or x is n times better than y in terms of performance then basically what you are saying is n is the ratio of two performance numbers performance of x or performance of y and in terms of execution time it is reciprocal of this ratio which is execution time of y over execution time of x. So you take the same program, run over two machines A and B, measure the time and take the ratio so that gives you relative performance.

For example, suppose machine A runs the program in 20 seconds and machine B runs the same program in 25 seconds so how many times A is faster than B or how many times B is faster than A. so it is the ratio of these two numbers; A is 25 by 20 times faster than B which is 5 upon 4 or 25 percent faster. Or you could say that B is 0.8 times faster than A which means effectively you are saying it is slower than A.

(Refer Slide Time 19:55)



Performance Examples

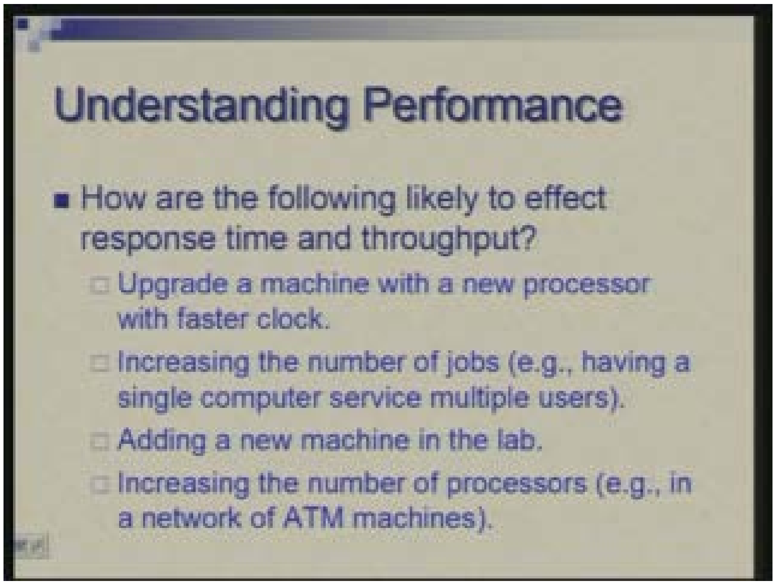
- Time of Concorde vs. Boeing 747?
  Concorde is 1350 mph / 610 mph
  = 2.2 times faster (120%)
  (alternatively 6.5 hours / 3 hours = 2.2 times)
- Throughput of Concorde vs. Boeing 747 ?
  Concord is 178,200 pmph / 286,700 pmph = 0.62 "times faster"
  Boeing is 286,700 pmph / 178,200 pmph = 1.6 "times faster" (60%)
- Our focus for processor performance = execution time for a single job - why?

Now, coming to the aircraft example or airline example you could compare the time; time of Concorde versus Boeing 747. You could take the ratio of speeds one way or ratio of the travel times in the other way. So you could say that Concorde is 1350 mph divided by 610 mph so you take this ratio which means 2.2 times faster or in another words it is 120 percent times faster than Boeing.

Alternatively you could take the ratio of the travel times. So 6.5 hours divided by 3 hours so now you have taken the reciprocal ratio which will again obviously give you the same figure. One could also have compared throughput. Suppose we define throughput as we did in that chart, persons or passengers carried multiplied by miles per hour. So you take the ratio of this pmph figure passenger capacity multiplied by speed which is for Concorde and this is for Boeing (Refer Slide Time: 21:14) so Concorde is 0.62 times faster or alternatively you could say Boeing is so many times or 1.6 or 60 percent time faster than Concorde. So it is now faster in throughput sense whereas the first comparison was faster in speed sense or travel time sense.

Therefore, we will be focusing on the first one of these that is the time because that is of concern to an individual and the current discussion which will have on architecture would be more closely linked with that. So when you understand the question of our performance you may like to understand and ask questions like this, that in order to improve performance you may consider a change like upgrading a machine with a new processor with a faster clock. What will it improve? You have a machine, you take the processor, replace it with a faster processor I mean let us say Pentium 4 2.8 GHz is replaced by Pentium 4 with 3.0 GHz so what changes the throughput changes or the response time changes both will improve.

(Refer Slide Time 22:25)



Understanding Performance

- How are the following likely to effect response time and throughput?
  - □ Upgrade a machine with a new processor with faster clock.
  - □ Increasing the number of jobs (e.g., having a single computer service multiple users).
  - □ Adding a new machine in the lab.
  - □ Increasing the number of processors (e.g., in a network of ATM machines).

Increasing the number of jobs…… Suppose you have a system which is which is running a sequence of jobs so imagine that people come with their programs, run it and go away so increasing the number of jobs what will it improve response time or throughput? It will obviously improve throughput. So, as a policy if you start taking multiple jobs trying to run them in time scheduled manner it would improve the throughput, in fact it can slightly reduce the response time because there may be some overhead of switching from one to the other, overhead of doing time sharing may be there.

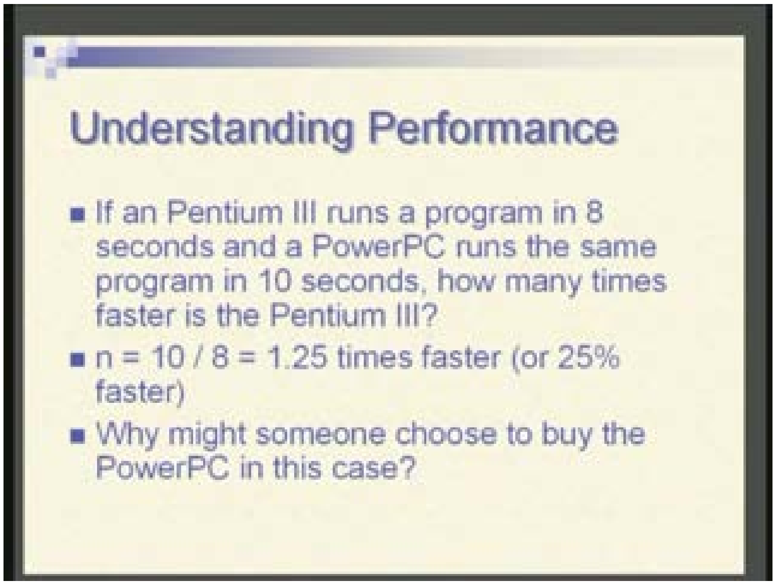
On the other hand, if you knew that there is no time sharing, a processor has to just take one job as it comes and finish it then you are not incurring some overhead which otherwise you are incurring. So response time actually could deteriorate.

Suppose you are in a lab environment where there are couple of machines, five machines are lying there, if you add a couple of more machines what does it improve response time or throughput? Certainly throughput will improve, more jobs can be done. there could be…. if you are counting from the time you are on the machine of course the response time does not change but if you look at the total time you spend from the time you step in the lab and you go out with your work done that may improve because with less machines and more students trying to do it you may have to wait in front of the machine. So something will improve here in terms of time as well if you are talking of total time you have to spend in the lab that could improve but the time which a processor takes to execute your job will not change if you have five machines or ten machines or fifty machines of the same type.

If you, let us say, in an network of ATM machine if you increase the number of processors then suppose there are number of network of ATM machines which are controlled by certain number of processors so if you increase the number of processors which are working at the back end for supporting these ATM machines what will happen? Will you improve throughput or will you improve response time? It will improve both, it will improve throughput more machines are there to do more jobs but again somewhat in the similar analogy to the lab your waiting time may get cut down somewhere t because you may fire a transaction on ATM but the processor is busy so it waits for a time it waits for a while but if there were other processors to take care of your transaction it could respond fast.

So, continuing with this such kind of practical question now suppose you have two processors two machines from different vendors the Pentium 3 and power PC if one takes 8 seconds and the other takes 10 seconds it is clear from what we have discussed which one is faster but could there be reasons to buy the one which takes longer? Yeah, there could be still reasons; it may be costing you less. So you have to see performance not in isolation, but at what price you are getting what performance so the cost factor cannot be altogether ignored.

(Refer Slide Time 26:47)

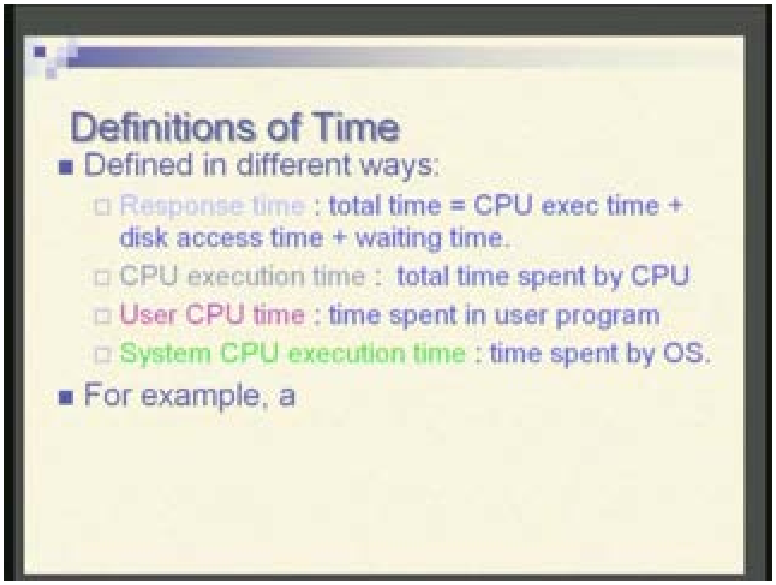


Understanding Performance

- If an Pentium III runs a program in 8 seconds and a PowerPC runs the same program in 10 seconds, how many times faster is the Pentium III?
- n = 10 / 8 = 1.25 times faster (or 25% faster)
- Why might someone choose to buy the PowerPC in this case?

Let us say, again going back to the aircraft situation the Concorde would be faster if you are concerned about the travel time but then the ticket may be more expensive. So if you have a budget limiting budget then you have to make a choice accordingly. So it may be your task may be to get the best performance for a given cost or if your target is of a particular performance you like to see with what minimum cost you can get the same performance.

(Refer Slide Time 28:10)



Definitions of Time

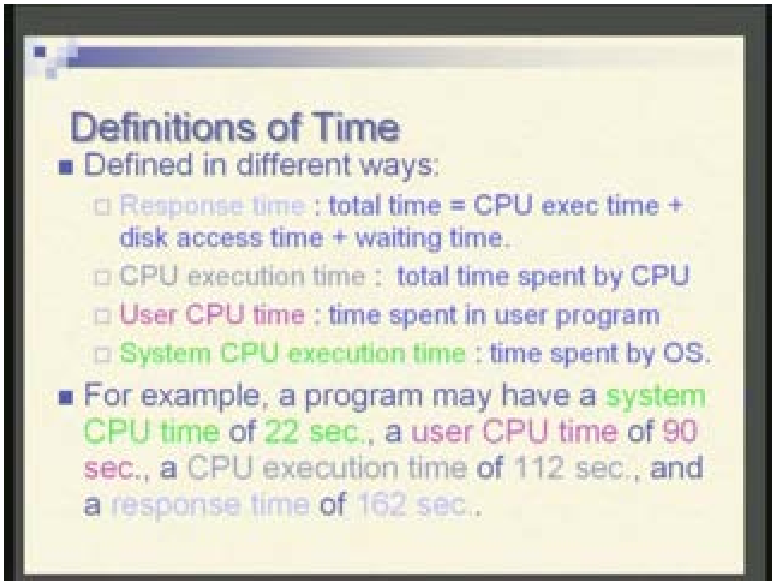
- Defined in different ways:
  - □ Response time : total time = CPU exec time + disk access time + waiting time.
  - □ CPU execution time : total time spent by CPU
  - □ User CPU time : time spent in user program
  - □ System CPU execution time : time spent by OS.
- For example, a

Now, as I mentioned, we will focus on time aspects not so much on the throughput aspect but even time issue could be fairly complicated depending upon how you view it. You could have response time which is a total time taken by the CPU to do the job plus time which for example the disk might take to access files, the time your program may be waiting for, it may wait for some input/output to happen or it may wait for some of the tasks in a multitasking environment there may be a wait involved. So the time which you see ultimately is sum of all these.

On the other hand, the CPU execution time would be the time which the CPU actually spends doing your program so it will exclude the disk access time, it will exclude the waiting time and this itself will be consisting of the time which has been spent actually executing your instructions because there will be a component of OS instruction. There is OS overhead, OS is doing some service for your program; it loads your program, it takes care of IO, it takes care of communication so some overhead which OS incurs is attributable to your program.

So, strictly speaking the total CPU execution time is the sum of these two the time spent on user code and the time spent on OS code and that part of OS code which is executed to serve your program. There are many factors which influence these, we are not talking of OS design, we are not talking of so much of peripheral design at the moment. If you are concerned with the processor architecture then what this will influence is essentially user CPU time and also of course to some extent OS time. But OS time will also be depending upon what kind of OS you are writing, what is the OS scheduling policy and so on. So our immediate concern would be the user CPU time. That means you have user program and how long a CPU will take how long the given architecture will take to run that program.

(Refer Slide Time 30:51)

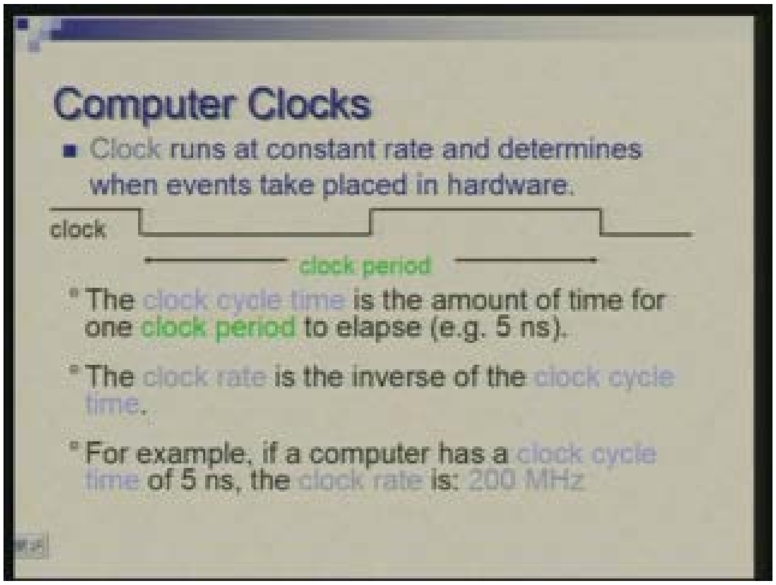


Definitions of Time

- Defined in different ways:
  - □ Response time : total time = CPU exec time + disk access time + waiting time.
  - □ CPU execution time : total time spent by CPU
  - □ User CPU time : time spent in user program
  - □ System CPU execution time : time spent by OS.
- For example, a program may have a system CPU time of 22 sec., a user CPU time of 90 sec., a CPU execution time of 112 sec., and a response time of 162 sec..

So, as an example, to clarify these points, let us say a CPU time for a given program is 22 seconds with sorry the OS component time of the CPU time is 22 seconds, user component is 90 seconds so the total CPU execution time could be 112 seconds which is the sum of these two and with memory time or disk time, waiting time all put together the time could be a total of 162 seconds which is 112 plus some other time which is 50 seconds in this case.

(Refer Slide Time 31:41)



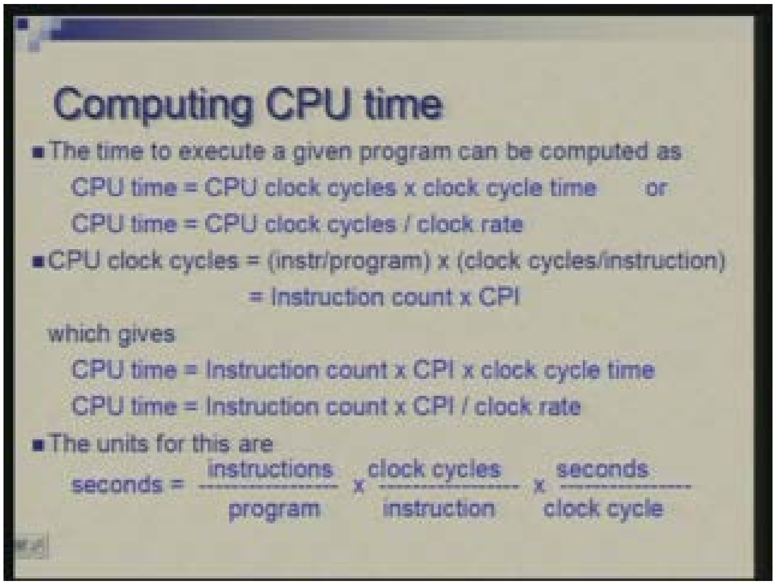
Computer Clocks

- Clock runs at constant rate and determines when events take placed in hardware.

- The clock cycle time is the amount of time for one clock period to elapse (e.g. 5 ns).
- The clock rate is the inverse of the clock cycle time.
- For example, if a computer has a clock cycle time of 5 ns, the clock rate is: 200 MHz

Now, having said that we are looking at the time CPU spent in executing a user program so we like to break it down further and try to express in terms of the clock period. As you know that all processors run with a clock; when you say a processor is running at 2 GHz that means it is running with a periodic signal at the frequency of 2 GHz or half nanoseconds cycle time. So this 2 Gigahertz is the rate at which events takes place within the processor. So any hardware activity in the processor will take place at discrete edges at which clock changes state. So the clock cycle time or the clock period is the reciprocal of the clock frequency. So, for example, if clock frequency is 200 MHz the cycle time or the clock period is 5 nanoseconds.

(Refer Slide Time 32:46)



Computing CPU time

- The time to execute a given program can be computed as

  CPU time = CPU clock cycles x clock cycle time or

  CPU time = CPU clock cycles / clock rate
- CPU clock cycles = (instr/program) x (clock cycles/instruction)

  = Instruction count x CPI

  which gives

  CPU time = Instruction count x CPI x clock cycle time

  CPU time = Instruction count x CPI / clock rate
- The units for this are

$$\text{seconds} = \frac{\text{instructions}}{\text{program}} \times \frac{\text{clock cycles}}{\text{instruction}} \times \frac{\text{seconds}}{\text{clock cycle}}$$

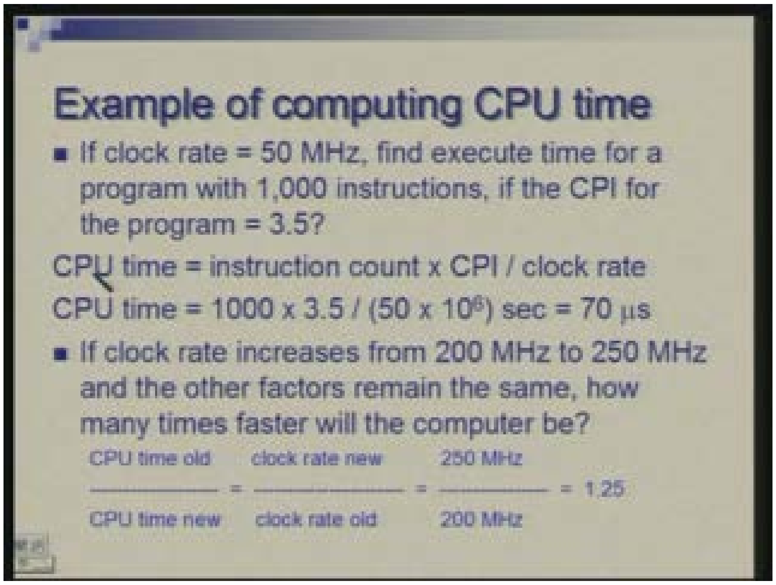
So, relating the time to execute to cycle time we could say that the CPU time is CPU clock cycles into clock cycle time. Suppose your cycle time is 1 nanosecond and you are executing 1 million cycles for doing something then 1 million into 1 nanosecond is how much? 1 millisecond. So CPU will spend 1 millisecond of time.

Alternatively we could say that CPU time is CPU clock cycles divided by clock frequency or clock rate same thing because clock rate and cycle time are reciprocals. It can be further rewritten as CPU clock cycles depending upon how many instructions are there in a program and how many clock cycles are taken per instruction. So the product of instruction count and CPI stands for Cycles Per Instruction. If you know these two figures you can multiply these two to get the idea of the number of cycles which CPU takes to run a program. With that we can write CPU time equal to instruction count multiplied by CPI multiplied by clock cycle time or clock period. Alternatively, it is CPU time equal to instruction count multiplied by CPI divided by clock rate or clock frequency. So units of these quantities can be very easily seen in this equation.

CPU time let us say we are talking of seconds so second CPU time in seconds is equal to instruction count which is instructions per program, CPI is cycles per instruction and clock rate is seconds per clock cycle. So you could see dimensionally how this is balanced; instructions cancel with instruction, cycles get cancel with cycles so what you get is….. basically this should be seconds per program on the left hand side, seconds required to execute a program or seconds per program. So let us illustrate this with an example.

Suppose you have a processor with clock rate of 50 MHz how do you find execution time for a program which has one thousand instructions and given that CPI for the program is 3.5. Now you might wonder when I am talking of cycles everything is happening in cycles why am I talking of a fraction here. So it is a fraction because we are talking of average and I will elaborate on that little later.

(Refer Slide Time 35:54)



Example of computing CPU time

- If clock rate = 50 MHz, find execute time for a program with 1,000 instructions, if the CPI for the program = 3.5?

CPU time = instruction count x CPI / clock rate

CPU time = 1000 x 3.5 / (50 x 10^6) sec = 70 µs

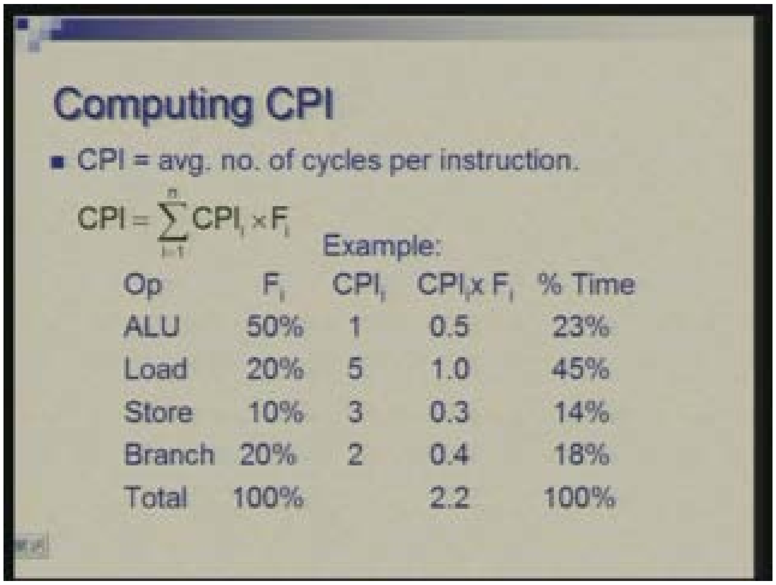
- If clock rate increases from 200 MHz to 250 MHz and the other factors remain the same, how many times faster will the computer be?

$$\frac{\text{CPU time old}}{\text{CPU time new}} = \frac{\text{clock rate new}}{\text{clock rate old}} = \frac{250 \text{ MHz}}{200 \text{ MHz}} = 1.25$$

So let us take this value and try to find CPU time in terms of these instruction count multiplied by CPI divided by the clock rate which was the formula we just saw. So, substituting the value the instruction count is thousand, CPI is 3.5 and clock rate is 50 MHz so 50 into 10 raised power 6. So you can multiply this and get 70 into 10 raised to power minus 6 seconds or 70 microseconds.

Now, suppose we have a situation where everything else remains the same and clock frequency increases I asked this kind of question earlier. So suppose in some case clock changes from 200 MHz to 250 MHz other factors remain the same so how would the time change. So obviously time which is dependent upon three factors while others remaining the same when you take the ratio of old time and new time others will cancel out and basically you will get the inverse ratio of the clock rates or direct ratio of clock periods. So it is 250 MHz by 200 MHz or 1.25. So old time and new time are related with this ratio or that is old time is 1.25 times the new time.

(Refer Slide Time 37:33)



Computing CPI

- CPI = avg. no. of cycles per instruction.

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

Example:

Op	F_i	CPI_i	$CPI_i x F_i$	% Time
ALU	50%	1	0.5	23%
Load	20%	5	1.0	45%
Store	10%	3	0.3	14%
Branch	20%	2	0.4	18%
Total	100%		2.2	100%

Now, coming back to the point of fractional CPI as I mentioned that CPI as we have put there is average because the time taken the number of cycles taken by different instructions may be different and the reason for that would come from the way we implement the hardware. You may notice that hardware takes longer for some instructions and shorter for other instructions. So, given that we need to find an average. so CPI is a essentially a weighted average of CPIs of individual instructions. Suppose there are n different instructions or instruction types and we know CPI of each individual instruction then what we need is a weightage F i which is how frequently this instruction is occurring in a given program.

Therefore, as an example suppose we have these five instructions rather there are five categories of instructions; arithmetic instructions which are present 50 percent of the time in a program that means half the instruction in a program are add, subtract, multiply, divide of that category; 20 percent of instructions are load; 10 percent are store; 20 percent branches and this of course totals up to 100. The CPI is different for these; let us say for ALU instruction CPI is 1 ALU stands for Arithmetic Logic Unit instruction which does arithmetical or logical operation, they take one cycle. Load takes five cycles, store takes three cycles, branch takes two cycles and then weighted average can be obtained by this formula (Refer Slide Time: 29:31).

Hence, we find CPI into F i; instead of percentage I am taking fraction while computing this so 1 into 0.5, this is 5 into 0.2, this is 3 into 0.1 and this is 2 into 0.2. So now you can sum these and you get 2.2 so on the average an instruction spends 2.2 cycles for its execution. So, interestingly you can also find out what is the fraction of time CPU would spend doing ALU instructions; what is the fraction of time CPU will spend doing load instruction and so on so that also could be found out from this data and is shown here. How will you find this what formula have i used can you figure out?

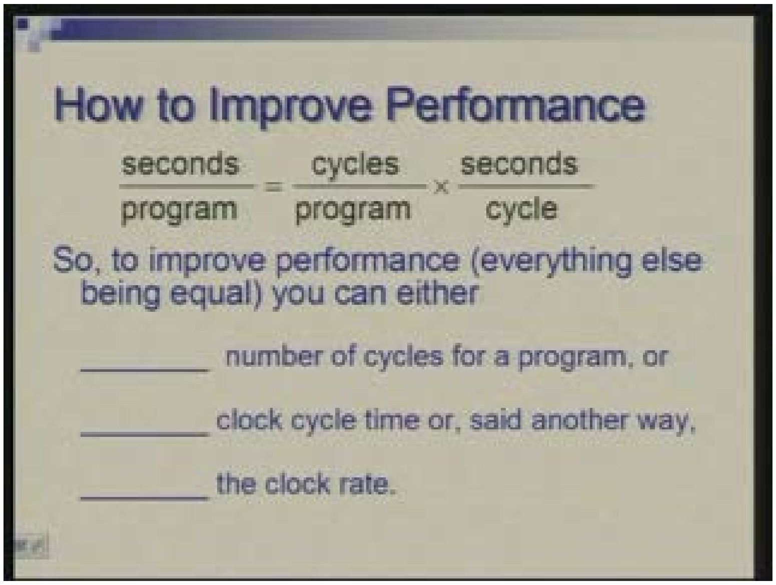
[Conversation between student and Professor: 40:33…….the CPI into F i divided by total CPI] yeah, so basically if you take this as a total what fraction 0.5 is of 2.2 so 0.5 is 23 percent of 2.2 this is what I am saying 1.0 is 45 percent of 2.2, 0.3 is 14 percent of 2.2 and so on. So it gives you an idea of where this processor spends time. in this case it is very clear that such a processor will spend maximum amount of time doing loads and if you were to figure out how to write a program in an efficient manner you will try to you will keep your attention on minimizing the loads.

So here is some explanation of why some instructions take more time or what kind of instructions take longer and what kind of instructions take shorter. So, for example, typically although in this example we group all ALU instructions together but there would be many situations where multiplications and divisions will take longer than addition and subtraction. Now, between integer operations and floating point operations floating point will take longer than integer operations, memory accesses take longer than accessing registers.

So, if a processor has two instructions for adding one picks up operands from memory, one picks up operands from registers then obviously the one taken from the register will be faster. So when you change your cycle time suppose you are trying to redesign a faster clock it can also have an influence on the number of cycles because number of cycles required for doing an instruction depends upon how much work you do in one cycle. So if you make the cycle faster you may take more cycle. So let us say multiply operation. So there is some work to be performed and how you divide into cycles would determine how many cycles you need.

So, if you make your cycles longer you can do more work per cycle and therefore number of cycles may be less and vice versa but what would eventually matter is the product of clock cycle time period multiplied by the number of cycles. So you cannot just attempt to pull down one quantity and hoping that other will not change so one change can influence the other and one has to see the composite effect.

(Refer Slide Time 43:38)



How to Improve Performance

$$\frac{\text{seconds}}{\text{program}} = \frac{\text{cycles}}{\text{program}} \times \frac{\text{seconds}}{\text{cycle}}$$

So, to improve performance (everything else being equal) you can either

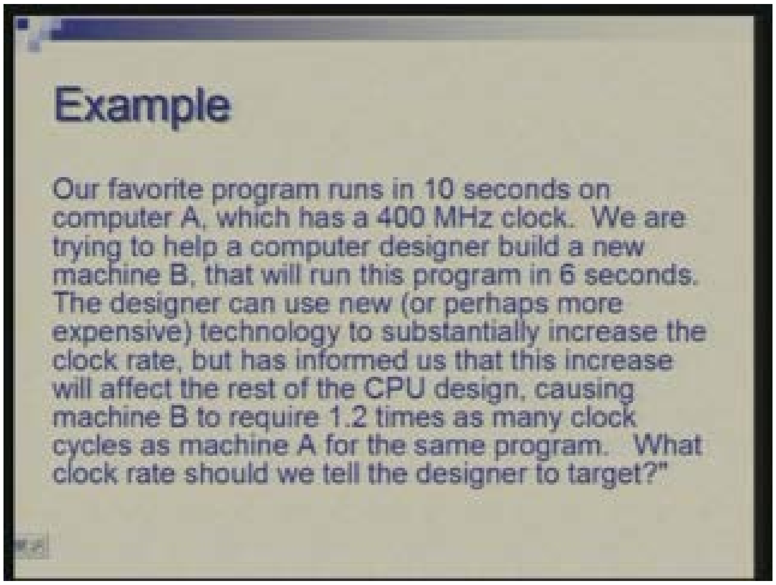
________ number of cycles for a program, or

________ clock cycle time or, said another way,

________ the clock rate.

Now, coming back to this formula seconds per program is cycles per program multiplied by seconds per cycle. We, of course, had divided this into instructions per program multiplied by cycles per instruction but now for the current discussion if you look at this so to improve performance everything else being equal you can either do something to the number of cycles for a program; you can reduce the number of cycles for a program right or do something to the clock cycles or you can reduce this seconds per cycle (Refer Slide Time: 44:31) or equivalently you can increase the clock rate. So these simple equations can tell you very easily which way you must make a move. But if your changes are influencing…… if a change which you do influence more than one factor then you have to see the composite effect; you may be improving one but you may be doing worse in the other.

(Refer Slide Time 45:00)



Example

Our favorite program runs in 10 seconds on computer A, which has a 400 MHz clock. We are trying to help a computer designer build a new machine B, that will run this program in 6 seconds. The designer can use new (or perhaps more expensive) technology to substantially increase the clock rate, but has informed us that this increase will affect the rest of the CPU design, causing machine B to require 1.2 times as many clock cycles as machine A for the same program. What clock rate should we tell the designer to target?"

Before closing let me just present an example to you. It looks like a complicated statement let us go through it gradually.

There is a program which runs in 10 seconds on computer A and the computer has a clock of 400 MHz. So now we are looking for another design we want to help a computer design and build a new machine B which will run this program in 6 seconds. So ultimately we want performance to be improved as far as our program is concerned. the designer can use new or perhaps more expensive technology to substantially increase the clock rate but has informed us that this increase will affect the rest of the CPU design causing machine B to require 1.2 times as many clock cycles as A for the same program. What clock rate should we tell the designer to target?

Pardon [Conversation between student and Professor: (46:08)…….800] how did you…… the answer he is giving is 800 how do you get that?
[Student: Sir, the number of cycles is 1.2 times so the time taken equivalently should be 12seconds so the speed should be doubled reduce it to 6] see, you have to look at these things……..where did it go? So these are the values given for first case A and the time………. let me write it separately. So we get 10 seconds by executing N instructions with certain CPI with the clock rate of 400 MHz. On the other hand, we want to get 6 as a time same number of instructions we are talking of same program; we are assuming that no instruction set is, instruction set is not getting changed; if instruction set changes this figure would change and what we have been given is that CPI is 1.2 times the old CPI so C into 1.2 and divided by a frequency of F MHz. So given that you have to find F.

What you will get is F is equal to 1.2 times 400 multiplied by 10 over 6 so that is the formula effectively used. And you will get 8 MHz as the answer sorry 800 MHz is the answer.

Any questions about this? We will stop at that.
[Student: Sir, what is the difference between system CPU execute N times and OS overhead]………?....... same thing. When you say system means operating system so the question was what is the difference between system CPU time and OS overhead; both are one same thing, just worded differently. Any other questions? Okay, thank you.