

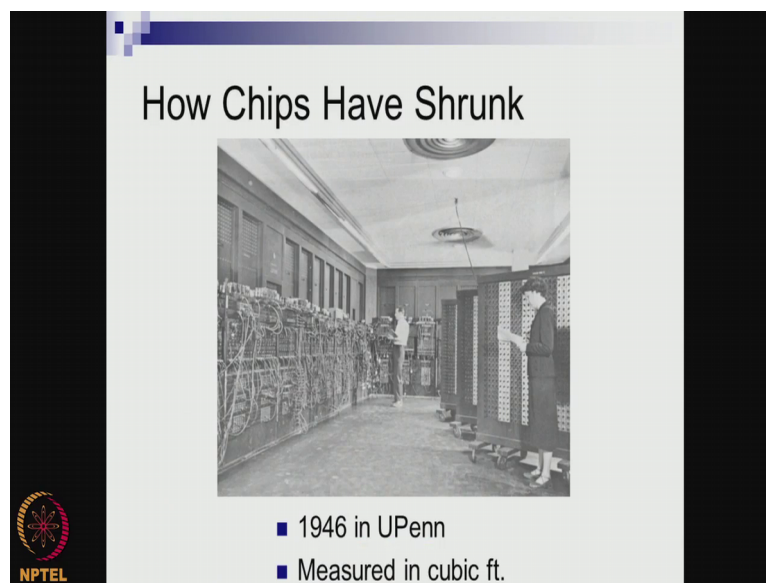
Computer Organization and Architecture
Prof. V. Kamakoti
Department of Computer Science and Engineering
Indian Institute of Technology, Madras

Lecture – 41

Conclusion – Recent Trends in Computer Organization & Architecture

So, welcome to this final lecture of this course on Computer Organization and Architecture. In this lecture what I will be doing is I will be talking about some of the recent trends in computer architecture why certain things have happened in the last decade and a half and then I will also be giving you some I will be showing you a video basically from Google on YouTube on what is the next computing, trend in computing that the world is actually headed towards before we some of this course.

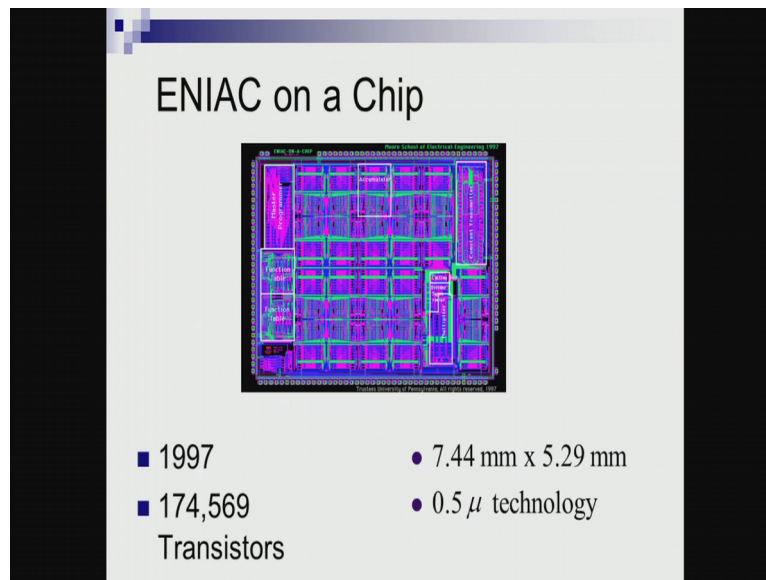
(Refer Slide Time: 00:57)



This is the computer made in 1946 in University of Pennsylvania it is measured in cubic feet and this was not bigger than today's glorified calculator the functionality that this could execute was very close to what today's calculator can take.

This was basically called the ENIAC, it had you know in it had several vacuum tubes and several connections that were going on lot of wires that were running.

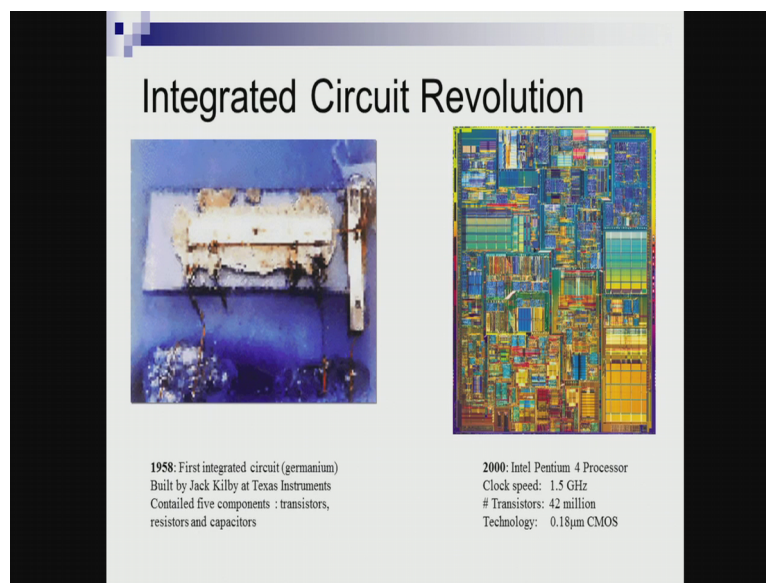
(Refer Slide Time: 01:27)



And the same chip the same ENIAC with the same functionality in 1997 was basically built using 174,569 transistors and the size of the chip was 7.44 millimeter into 5.29 millimeter and it was using a 0.5 micron technology 0.5 into 10 power minus 6 meter technology. What is 0.5 micron in sense that if you assume that a transistor it is a very rough answer what do you mean by 0.5 micron the rough answer is that if you assume a transistor is a square then the side of the square is 0.5 into 10 power minus 6 meters. So, this is roughly what we can say.

Now what made this possible that is you know from this in 1946 to this in 1997, 51 years what was it that made it possible from cubic feet to less you know to a couple of millimeters square.

(Refer Slide Time: 02:31)

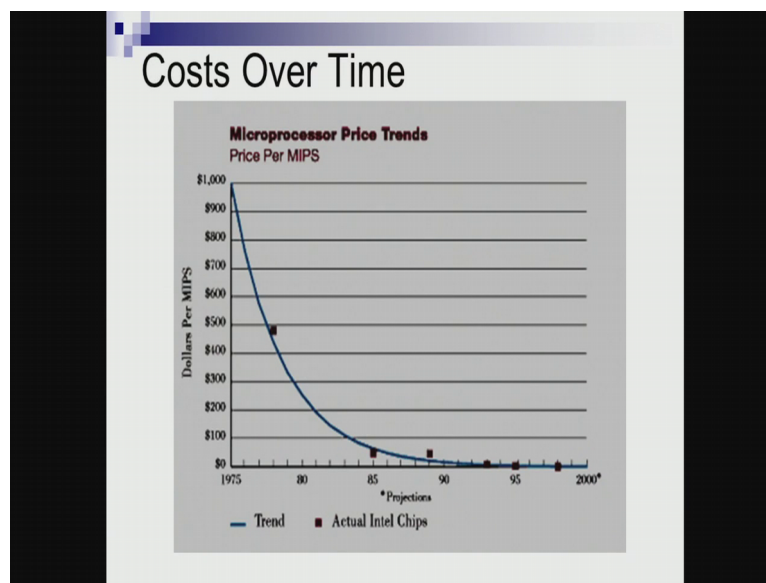


The first thing that we should note is the major integrated circuit revolution which came in 1958 from Jack Kilby of Texas instruments where he showed that on a same substrate more than one component can coexist. So, he made one substrate where it had 5 components there a couple of transistors resistors and capacitors all of them existed on the same substrate what you see on your left hand side in your screen.

So, this basically; that means, that I could get the circuits different components of the circuit put on the same substrate and that formed the basis of what we call as an integrated circuit where we integrate multiple varieties of components different types of transistors etcetera on the same substrate to realize certain functionality that integrated circuit evolution had grown so well that in 2000 what you see this is these are some of the images which Intel releases this is a Intel Pentium 4 processor in 2000 which had close to forty two million transistors and 1.5 gigahertz speed and they were using this 0.18 micron Cmos technology right. So, this is a 180 nanometer technology.

So, from 1958 where we had 5 components on a substrate and we have gone up to 42 million components on the substrate and this is the basic integrated circuit revolution.

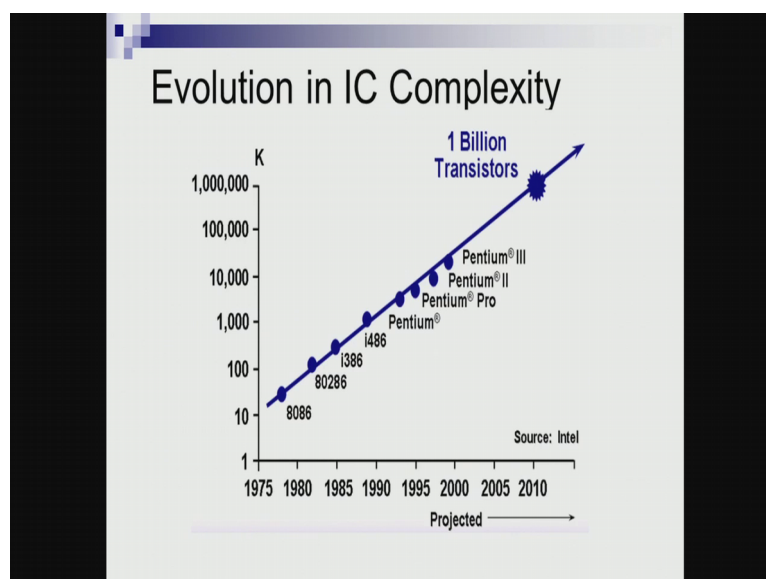
(Refer Slide Time: 04:08)



So, what happened because of this integrated circuit revolution the amount of money that we pay per million instructions per second or mips has decreased today for 1 million instruction per second we pay very less than we pay almost zero cost.

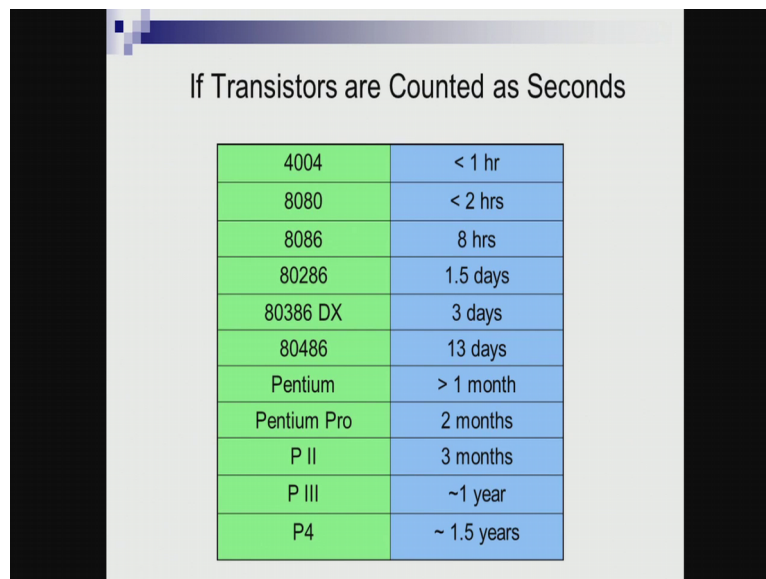
So, when you compare with 1975 for a 1 million instruction per second computer they are paying 1000 dollars today it has to be sub 1 dollar right. So, this is the trend that you see and what you see is there is a growth in the IC complexity in terms of number of transistors I put on a single chip.

(Refer Slide Time: 04:36)



What you see on the left hand side what you see on the y axis is a logarithmic scale right. So, you know on a logarithmic scale if I get a line then essentially it is an exponential growth and what has happened is that one bill the number of transistors we could put in unit area has keep growing exponentially over the last several years and that is what this has shown this is showing.

(Refer Slide Time: 05:14)

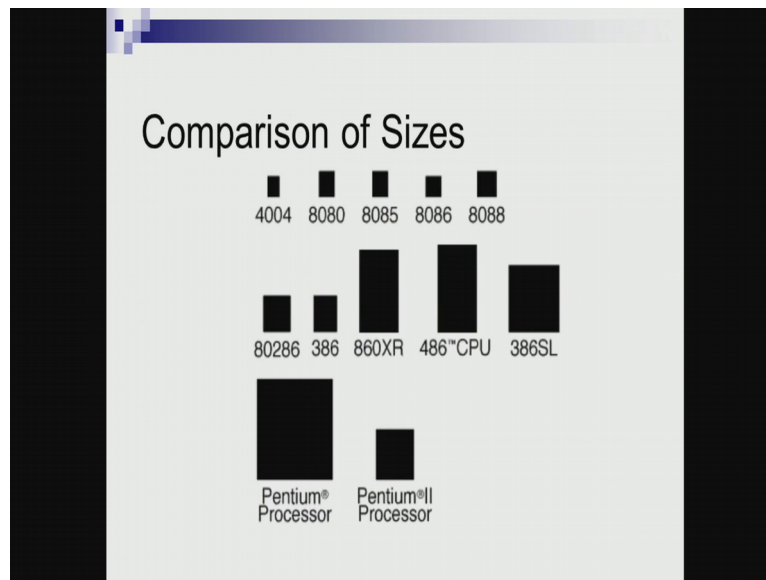


If Transistors are Counted as Seconds

4004	< 1 hr
8080	< 2 hrs
8086	8 hrs
80286	1.5 days
80386 DX	3 days
80486	13 days
Pentium	> 1 month
Pentium Pro	2 months
P II	3 months
P III	~1 year
P4	~ 1.5 years

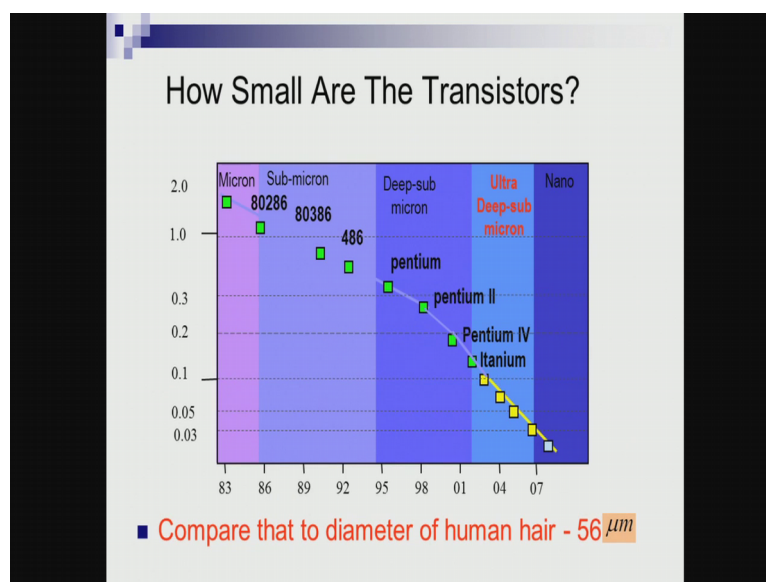
Now, if you actually count the number of transistors as second this is the Intel release data 4004 has less than 1 hour between they are less than 3600 transistors 80, 80 is what to us while you take the p four which is around 1.5 years. So, number of 1.5 into the you know 365 into 24 into 60 into 60 that many seconds that many transistors were there in p 4 as you see here.

(Refer Slide Time: 05:45)



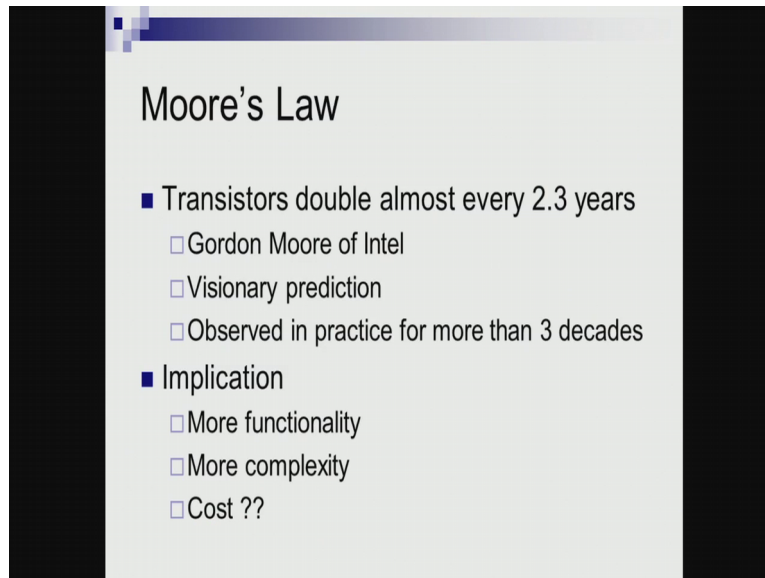
And when you see the comparison of sizes of the die please note that the die size has not grown very big, but the number of transistors that we put inside a die that this transistors per unit area has significantly increased and that is the integrated circuit revolution.

(Refer Slide Time: 06:05)



Now, how small are the transistors we have seen something till 2003 where we are seeing it is sub 0.03 micron. So, you know that the you may know that the human hair a well maintained human hair the diameter of well maintain human hair is 56 micrometer now you see something that is 0.03 micrometer. So, that is the size of transistor today.

(Refer Slide Time: 06:37)



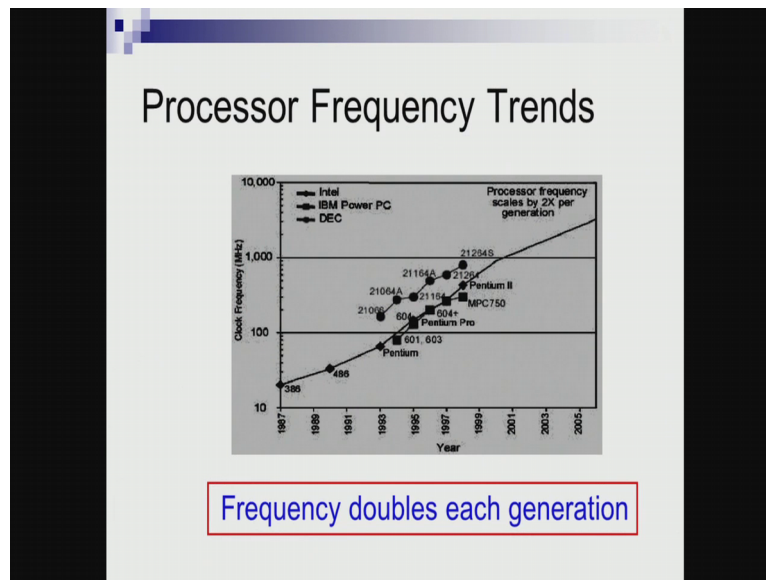
Moore's Law

- Transistors double almost every 2.3 years
 - Gordon Moore of Intel
 - Visionary prediction
 - Observed in practice for more than 3 decades
- Implication
 - More functionality
 - More complexity
 - Cost ??

So, one of the important things one patent law that was basically predicting this was Moore's law which basically said that the transistor density will double almost every 2.3 years what it means that number of transistors I can put in a unit area will almost double every 2.3 years right and that is what we have seen. So, the doubling is what you have seen the exponential growth in the number of transistors as you see here right within the same almost the same area we were able to get twice the number of transistors then what was say two point three years before.

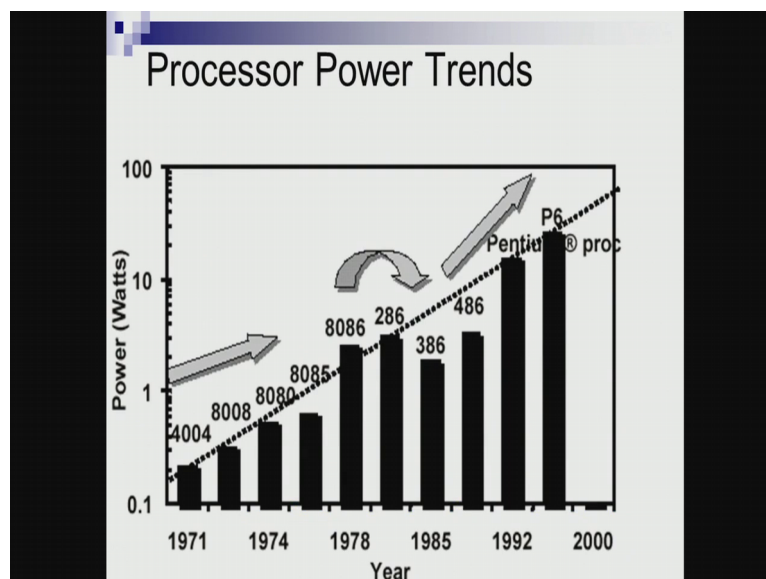
Now, if you have more and more transistors what is the implication like I have more workers right every transistor can implement some functionality. So, if I have more and more workers that; that means, I have more functionality and; that means, I can put more complexity into the system. The other way of looking at it more and more transistors it becomes very complex to manufacture complex to verify and if I put more functionality your complexity of the chip will increase. So, I can do complicated operations complex operations on the system at the same time the design also becomes complex. So, these all these things will have an implication on the cost of manufacturing the system.

(Refer Slide Time: 07:59)



On the other hand the frequency also started doubling right. So, this means that I could get more and more fast clocks and assuming that the number of operations remain the same, number of cycles per operation remain the same we have a higher frequency then I can start executing instructions faster than the previous generation. And again you see a log scale on your way axis here and a linear growth in the frequency which is means of the frequency was growing exponentially over time.

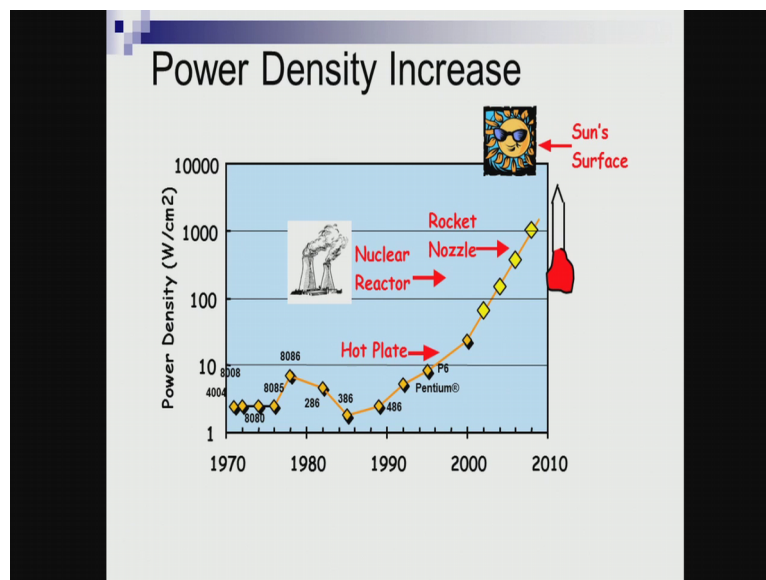
(Refer Slide Time: 08:30)



Nothing comes free of cost as your frequency started increasing your power also started

rising exponentially the power consumed by your processor. So, this is what we see you know we have already seen a hot plate and if that trend that continued post you know what you see on the y axis is again a log scale which is having power density which is the watt per centimeter square you know consumed by the chip or dissipated by the chip.

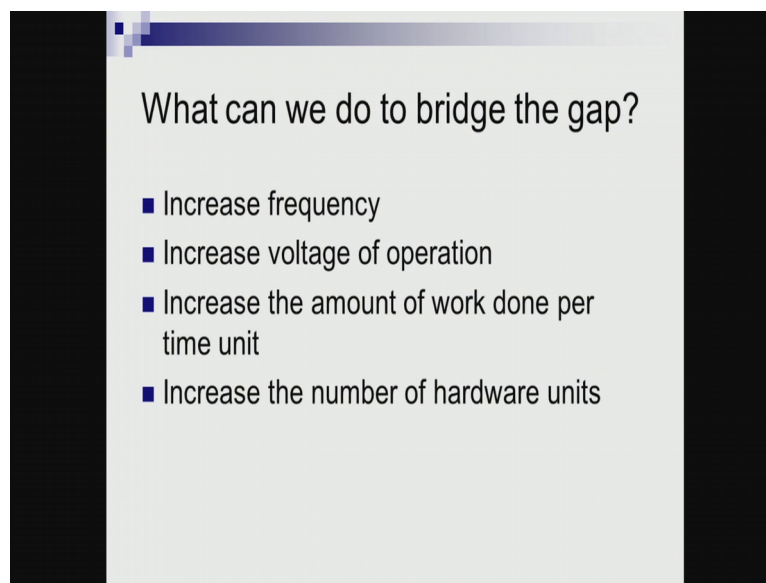
(Refer Slide Time: 08:40)



So, now, because of that your temperature starts increasing. So, if you had allowed the trend to continue you would have reached a nuclear reactor by you know some around mid of 2000 to 2010 and then a rocket nozzle by 2009 and half almost you would have get some surface by 2010.

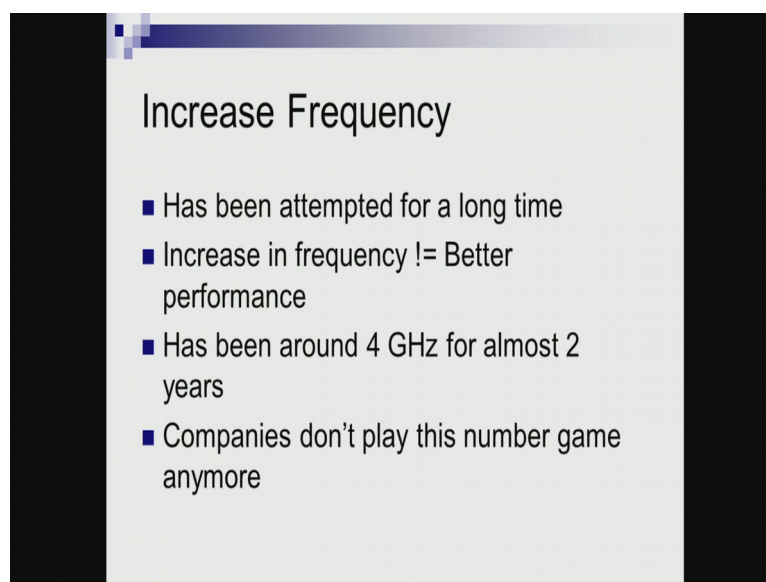
So, this is very complicated because has a frequency increases you have power consumption also increases your power density amount of power consumed per centimeter square also increases and that can reach to lot of power temperature issues and this is something which we cannot even imagine. So, we can maximum go up to a hot plate and that is where the things have stopped as you see in the graph it all stopped at p 6 20 empire we have touched hot plate in terms of temperature dissipation.

(Refer Slide Time: 09:56)



So, what can we do to bridge this gap? We can increase the frequency you can increase the voltage of operation we can increase the amount of work done per time unit to these three and we can increase the number of hardware units.

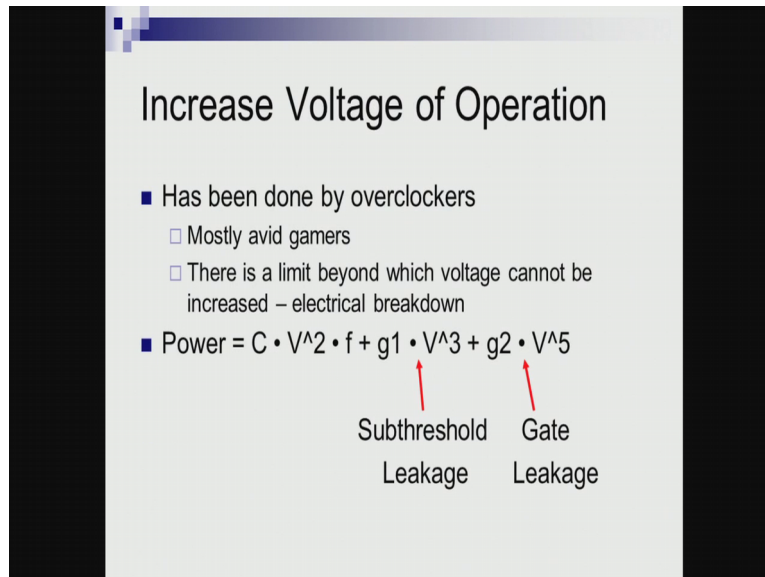
(Refer Slide Time: 10:17)



These are the four different ways by which I could make the computer more and more faster. If I start increasing the frequency already this has been attempted for a long time the increase in frequency need not necessarily give you a better performance. So and increasing the frequency also increase the power consumption. If I increase the voltage of operation

certainly your power also gets increased by large quantity.

(Refer Slide Time: 10:27)

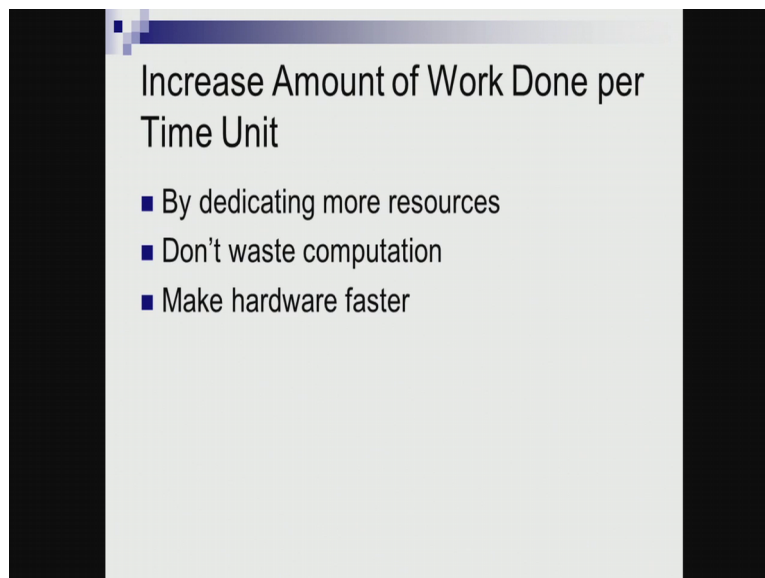


Increase Voltage of Operation

- Has been done by overclockers
 - Mostly avid gamers
 - There is a limit beyond which voltage cannot be increased – electrical breakdown
- $\text{Power} = C \cdot V^2 \cdot f + g_1 \cdot V^3 + g_2 \cdot V^5$
 - Subthreshold Leakage (pointing to $g_1 \cdot V^3$)
 - Gate Leakage (pointing to $g_2 \cdot V^5$)

So, the waste thing is to increase amount of work done per time unit. So, the way, that is how many companies including Intel came out.

(Refer Slide Time: 10:35)

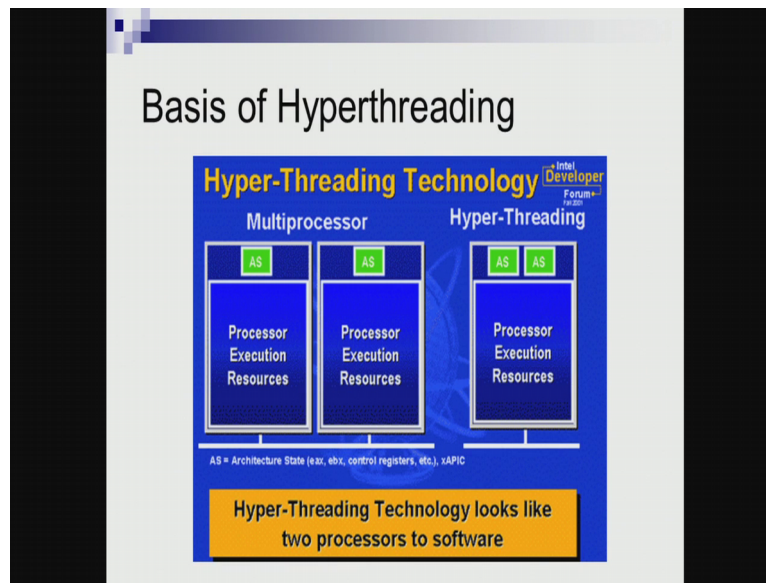


Increase Amount of Work Done per Time Unit

- By dedicating more resources
- Don't waste computation
- Make hardware faster

With the notion of multi threading or hyper threading in a multiprocessor system.

(Refer Slide Time: 10:41)



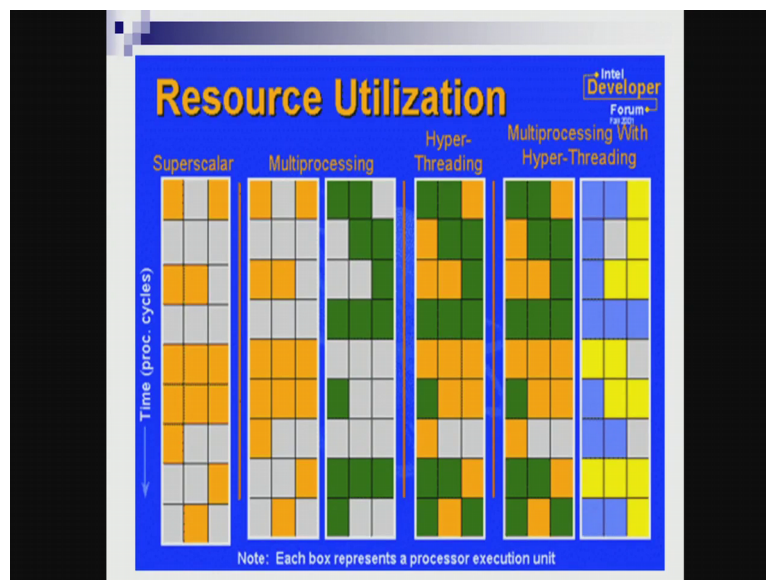
There will be two separate CPUs each will have its own architectural state meaning the general purpose registers and all those things that you need to run the process they have two separate architectural state. So, they can run two independent process, but one of the biggest drawback as you would see in the subsequent slide about this is that if one of the processor whatever you see on the left hand side there are two process of execution unit with two architectural state registers. If one of the processor is executing an integer program the entire floating point resource that you have put on the one of the process going to be waste while the other processor if it is going to use floating point programs then the integer arithmetic there is going to be a waste.

So, I could have two classes of program one using floating point another using integer arithmetic when, if in the case of the multi processor that you see on the left hand side both of processor will have both the floating point and the integer unit. So, one processor when it is executing the integer program the floating point unit will be idle there and another processor the integer unit will be idle. So, in sense we are not using we are not effectively or efficiently using the hardware given to us. So, that from that multi code that that is why there is a notion of hyper threading which was introduced where in there will be two different architectural state, but the processor execution resources would be shared that is what you see on the right hand side in this slide.

So, what happens here is that in the context of me having one integer program and one

floating point program there will two different architecture state of course, for each of these processes. But the processor execution resources will be more effectively used because both the integer and the floating point units that are part of the right hand side the single processor execution resource will be effectively used here. So, from going from a multi processor to a multi threading or hyper trading environment this served as a basic motivation. Not every program will go and do every type of computation there will be some program which will do integer there will be some program which we doing floating point. So, if I have one processor execution resource which will have both integer and floating point I could run multiple processes at the same time this is the biggest motivation for people to start looking at hyper threading slash multi threading.

(Refer Slide Time: 13:24)



So, this is a very clear thing whatever we see. So, the this is a graph actually whatever you see on the y axis is the time and whatever you see on the x axis is whether a particular unit is occupied or not as you see the first we have already seen in a superscalar. Now in the first cycle there are let us assume there are three units inside the superscalar processor now in the first cycle two of the units are occupied one is empty in the second cycle all the three are empty in the third cycle two are occupied etcetera and showing on the left most side of your thing just superscalar.

Now, only one process the orange process as I see here may be you see it as yellow something yellow oranges type you know works here only one process can execute when I

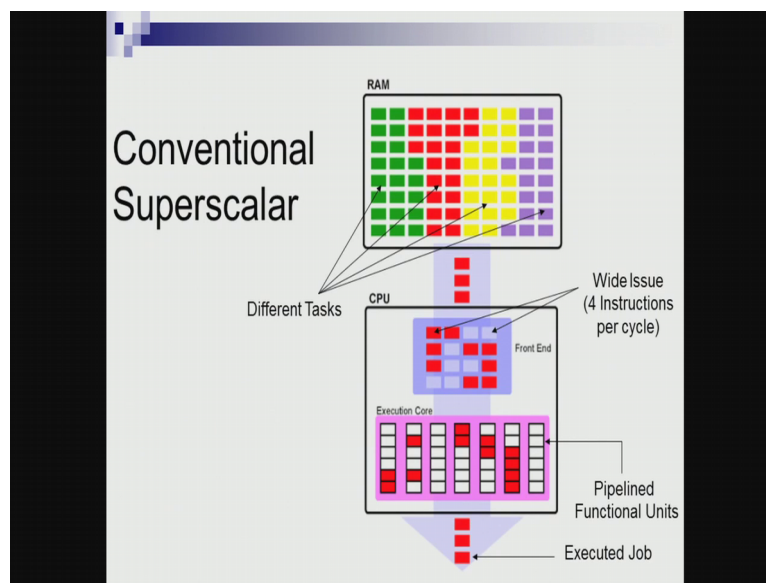
have a multi processing environment where I have two CPUs as you see on the next from your left the second from your left I could run both the orange or yellow instruction and also the green instructions at the same time right.

Now there is something called, so this two course one running one yellow process another running green process in the superscalar thing that you see on the leftmost the first one you cannot run the green process until your yellow processes process is executing. Now if you have multi core if you have multi CPU I can run both the things, but what I have done have a reduced a number of empty slots actually by going from superscalar to multi processing I have doubled the number of empty slots.

Next comes the you know the hyper threading where we can start using some amount of resources right. So, the hyper threading basically has two architecture state that I shown here, but the execution resource will be shared. So, what you see on the third column that is under hyper threading in this slide you see that whatever we (Refer Time: 15:24) multicores the execution resource resources have merged there.

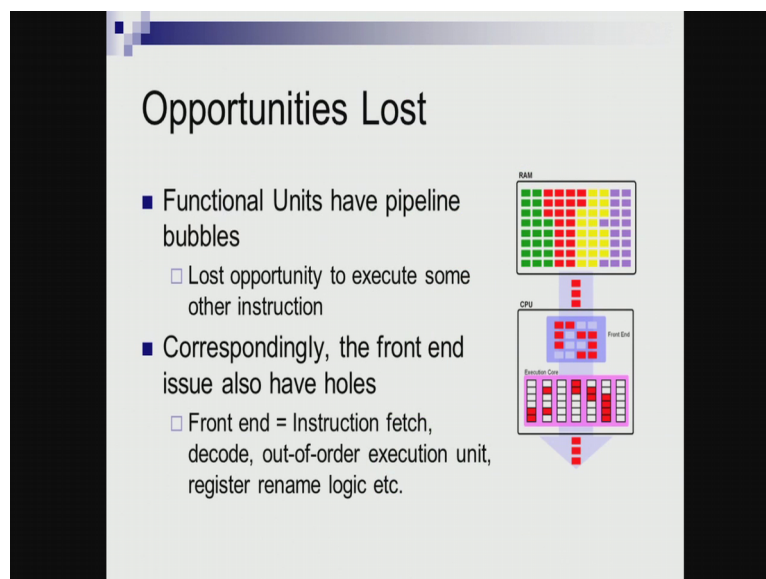
Now, you see the number of white squares on your third column is much less. Now today we can we are looking at multi processing with hyper threading that makes much more sense where in you have green yellow orange or and blue and yellow set of four type of programs. As you see as you move from the left to the right the number of white blocks actually significantly reduces and this is one of the way by which I could get better performance for my workloads.

(Refer Slide Time: 16:03)



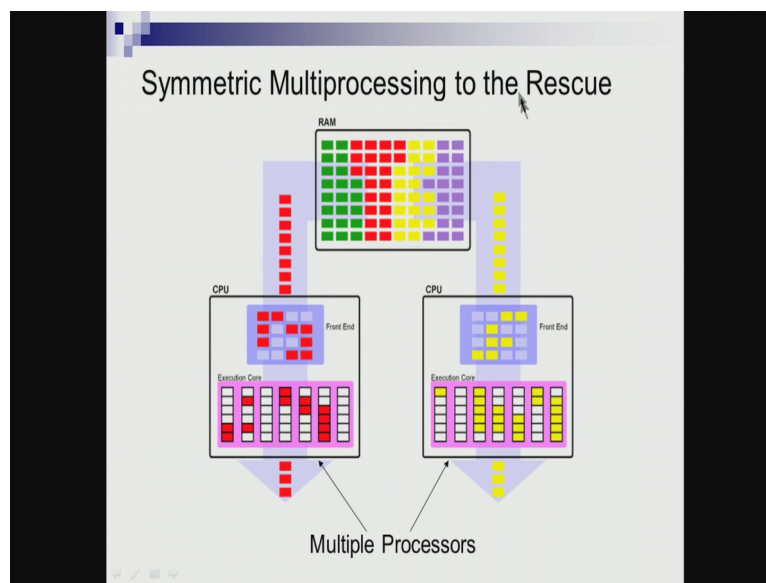
So, a conventional superscalar as we had seen what happens it takes one red program executes it and finish right.

(Refer Slide Time: 16:12).



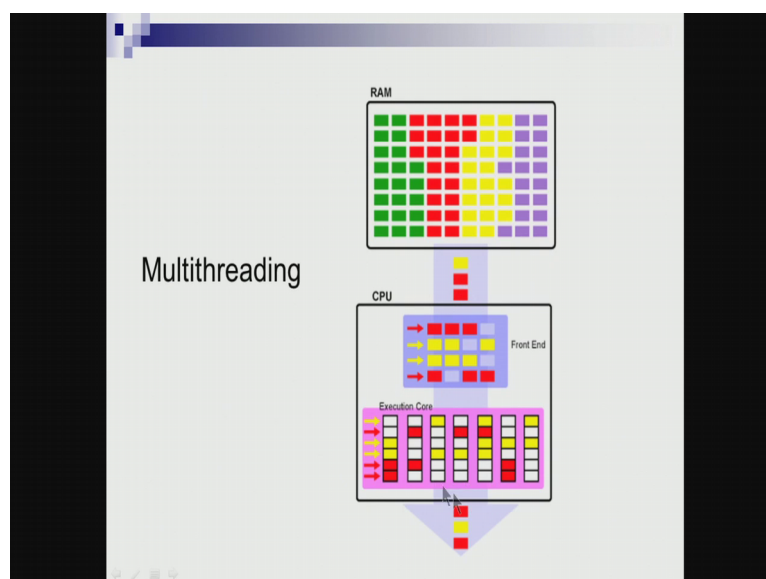
So, what happens is that if I do that you see you see in this slide as I move the mouse on top of it you see there are many white slots here, this units are under replaced.

(Refer Slide Time: 16:28)



When we move to the symmetric multiprocessing where I have two course at the same time two different course the number of white still actually doubles. So, I have more hardware and lot more hardware when compared to the single one namely this compare this only these slots are wasted, but here it has doubled this slot plus this yellow slot. So, when I go for symmetric multiprocessing yes I can do two processes at the same time, but then the wastage of resource is still comfortable the wastage of resource is still significant here.

(Refer Slide Time: 17:03)

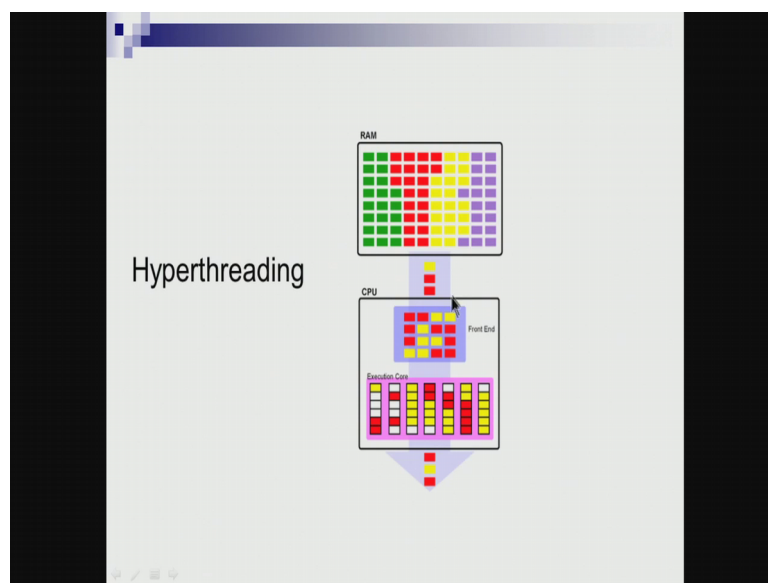


So, that then came the notion of a multi threading as you see here. So, I have 4 types of

programs in the ram as a show through the most one green red yellow and purple and in multi threading what we do we just first execute two of these four threads let us assume. So, the red and the yellow gets executed, but what happens is at one time slot as I move every row in this particular front end as I show here will have only either the red instructions or will have the yellow instructions. By this what happens is that the number of you know wastage of units does not significantly reduced.

So, this is multi threading where I have a single CPU and I execute two threads at the same time, but every entry of every thread we will occupy exactly one. So, so every row here will not have more than one instruction or more than one process that is basically involved. So, in one slot I would have either the red or the yellow, but not both that is what we call as multi threading.

(Refer Slide Time: 18:20)



Now, when we go to hyper threading please note that in one slot I could have both red and yellow instructions, so yellow instructions will work and then immediately you see that the number of white spots here have reduced. So, this is how processors got evolved say till last for 5 years where in we brought in notion of hyper threading multi threading etcetera basically to start utilizing the resources today, I cannot keep increasing the frequency because if I start increasing the frequency beyond this point your system will start burning we cannot have take that much amount of temperature.

So, the idea is to have slightly low frequency or very low frequency CPUs, but many of them.

So, if I have one say four gigahertz CPU, but I have 3 1.5 gigahertz CPU then certainly that that 3 into 1.5 is four 0.5 gigahertz if I am able to use those 3 CPUs underlying 3 CPUs effectively then automatically that will overshoot the 2 gigahertz circuit.

The question that we want to do is that given a hardware with the limited frequency, but multiple of them how can I use all of them together to go and exploit the performance and that is what the last part of the course before this lecture I gave you the a notions of different parallel random access memory like your EREW CRCW and CREW which basically teaches you how to write programs. So, that things like hyper threading can basically get a full you know a set of threads so that it can process it faster and you get the best benefit of having such type of an hyper threading environment.

So, with this, this is how the super scalar and this top level processors are going. So, to conclude what video says is something much different than what we covered in course, but that is going to be the future of computing. So, what is needed is yeah completely low very low power embedded processors capable of doing certain minimum functionalities in terms of graphics processing, in terms of communication and in terms of you know embedded computing that is one major need that we have. And we cannot say that the processors that could basically achieve what the video showed could be you know small computing systems because the type of processing that has happened there essentially needs lot of computing power, but that computing power may not come from a general purpose system, but it will be custom made processors with certain digital signal processing vector processing capabilities.

So, how do we get to this level is going to be the next quantum jump in the field of computer organization and architectures and that is going to be a very interesting revolution that is going to happen in the next few decades and its going to be pretty interesting. So, computer organization and architecture will never have its that we have to it will grow and it will grow in phenomenally in different directions and this one what we saw in the video is certainly one direction in which the next generation computing processors have to look at.

With this word I conclude this course I hope you enjoyed this course and I look for feedback from you and more interaction and a good carrier hopefully in the area of computer architecture and organization.

Thank you very much.