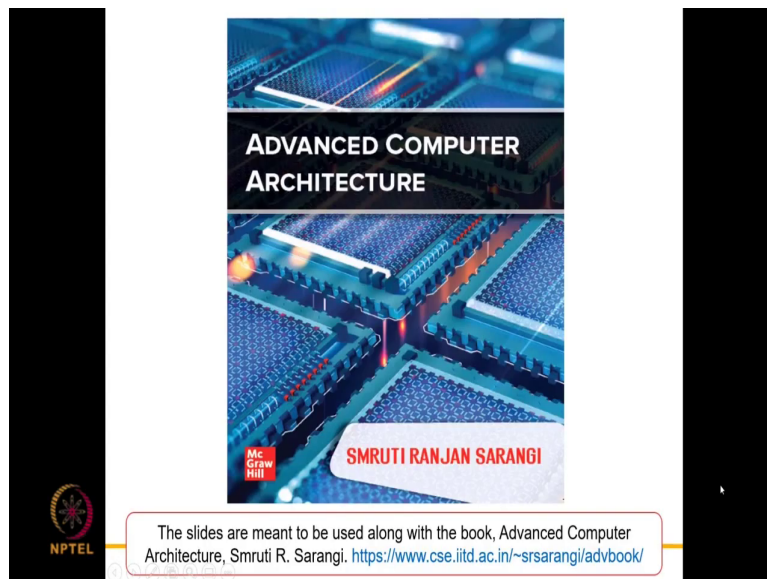


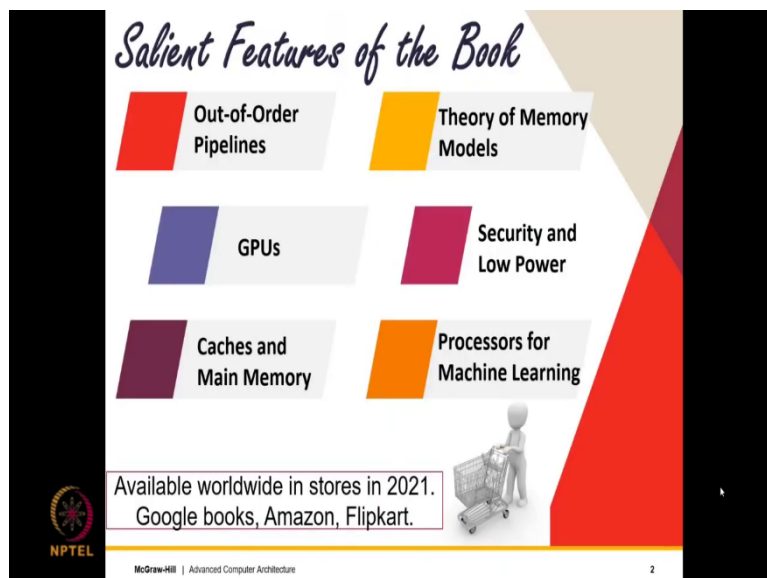
Advanced Computer Architecture
Prof. Smruti R. Sarangi
Department of Computer Science and Engineering
Indian Institute of Technology, Delhi

Lecture - 01
Introduction

(Refer Slide Time: 00:17)



(Refer Slide Time: 00:23)



Welcome to chapter 1 of the book Advanced Computer Architecture.

(Refer Slide Time: 00:39)

Background Required to Understand this Book

- Assembly Languages
- Basic Processor Design
- Basic Pipeline Design

<http://www.cse.iitd.ac.in/~srsarang/archbooksoft.html>

NPTEL

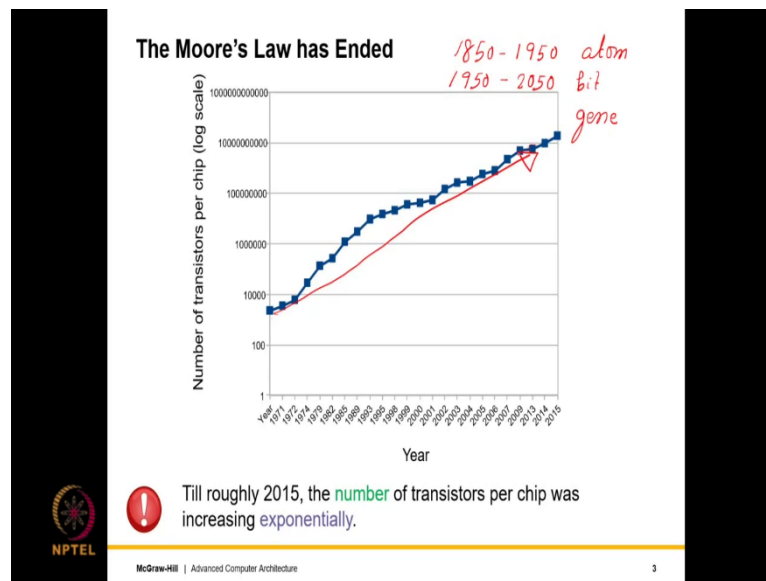
McGraw-Hill | Advanced Computer Architecture

2

So, this chapter is primarily introductory, but to understand this entire book. It is necessary to have an idea of some of the basic concepts of computer architecture. These include assembly languages. So, it can be any assembly language ARM, x86 anything, but at least the basic idea of assembly languages should be there, basic processor design this is also something that should be known not a lot. But at least the basics and basic pipeline design. so, this is what is required.

And readers, viewers of this video are pointed to the link to my previous book which is a book on undergraduate Computer Architecture. The book along with all the preprint chapters, are available at this link. So, I will just pause for two seconds such that readers can take a screenshot and then, navigate to this link 1, 2. So, assuming that the viewers have adequate background, we will move to the next slide.

(Refer Slide Time: 01:51)



So, the way that we will motivate modern computer architecture is like this that since the last 50 years, if something has really progressed and changed human life. It is the advent of computers motively, better, faster and more power efficient computers. So, this is something that has completely changed the fabric of society, industry and human life in general.

So, if you can if you think about it, maybe the era from let's say 1900 or maybe the 1850s to 1950 was the year of the atom. So, that is when most of the advances in atomic physics, materials and so on were happening. From 1950 to hopefully the next 100 years will be the era of the bit and after that people say will be the era of the gene where a lot of what we are doing will be taken over by biology.

But we are not there yet and I cannot see that far yet in the future, but definitely what has changed the discourse and basically the trajectory of human civilization has essentially been the atom, the bit and the gene.

In the bit part of the story, we have seen a near exponential increase of the number of transistors per chip which has rightly predicted by Gordon Moore. One of the co-founders of Intel. so, he had initially predicted that every year the number of transistors per chip will double. Initially, it was every year gradually, it slowed down to every 2 years, but clearly on a

log scale, you can see from 1970 to 2015 this has roughly held. So, it is linear in the log scale which means its exponential.

So, the Moore's law has basically given us an excess of 10 million transistors per chip today which can actually do a lot of interesting things along with having a supercomputer like power, a single chip can recognize faces, do computations, connect you to social networks and do everything.

(Refer Slide Time: 04:07)

The slide is titled "Moore's Law and Dennard Scaling". It contains two main sections, each with a blue header box. The first section, "Moore's Law", has a blue bullet point and text stating: "The number of transistors per chip will **double** roughly every 1-2 years. By 2023 this trend will altogether **stop**." Handwritten in red ink next to this text is "7nm → 5 → 3". The second section, "Dennard Scaling", has a red bullet point and text stating: "The power-density remains constant." and "The performance per Watt increases exponentially (tracking Moore's law)." Handwritten in red ink next to this text is a small diagram of a square with "20" above it and "150-400 mm²" below it. At the bottom center of the slide is a red octagonal "STOP" sign. The slide also features the NPTEL logo on the bottom left and the McGraw-Hill logo on the bottom right.

Moore's Law and Dennard Scaling

Moore's Law

The number of transistors per chip will **double** roughly every 1-2 years. By 2023 this trend will altogether **stop**. *7nm → 5 → 3*

Dennard Scaling

- The power-density remains constant.
- The performance per Watt increases exponentially (tracking Moore's law). *20*
150-400 mm²

STOP

NPTEL McGraw-Hill | Advanced Computer Architecture 4

So, there have been two laws in computer architecture which are essentially empirical observations, which have continued to hold, which are very important. The first as we just discussed is the Moore's law. It says that the number of transistors per chip will double roughly every 1 to 2 years. It has more or less held up till now, but now it is expected to come to an abrupt stop. The reason is that transistors as we speak which is 2021 are already very small.

So, this feature size which is the size of the smallest structure that we can fabricate in silicon that has come down to about 7 nanometers. So, less than that it is going to be very hard, in the sense, we may go down to 5 or 3 considering the fact that a silicon atom is around roughly speaking 200 nanometers, this is not more than 35 to 40 atoms placing atoms so close by or having such accurate fabrication at the atomic scale is increasingly is exceedingly

difficult that is why maybe transistors might reduce in size. So, the feature size might go down from 7 to 5 or who knows maybe 3, but that is it.

So, Moore's law has come to a stop, but essentially what this book will look at is that in spite of the Moore's law coming to a stop, which basically means in spite of transistor, sizes etcetera remaining constant. How do we get the maximum possible performance from computers? So, this is what modern computer architecture is all about.

And another sister law; let us call it the sister of Moore's law, many people may not like that is called Dennard scaling. So, this basically says is that the power density remains constant. So, for various fabrication reasons, the size of a chip has remained more or less constant which is roughly you can say a regular server chip is 2 cm X 2 cm or 20 X 20 mm.

So, this size of let us say 400-millimeter square for a server chip slightly lowered for a desktop or laptop chip. Let's say 150 to 400 millimeter square. so, the sizes remain the same mainly for yield reasons. So, what Dennard had predicted is that the power density will remain the same which means the total power dissipated will remain the same.

If we combine this with Moore's law, what it would also imply is that the performance which increases exponentially. The performance per Watt will also increase exponentially mainly because the number of Watts remains more or less constant. It is determined by other factors such as the maximum rate of heat removal and so on from the chip.

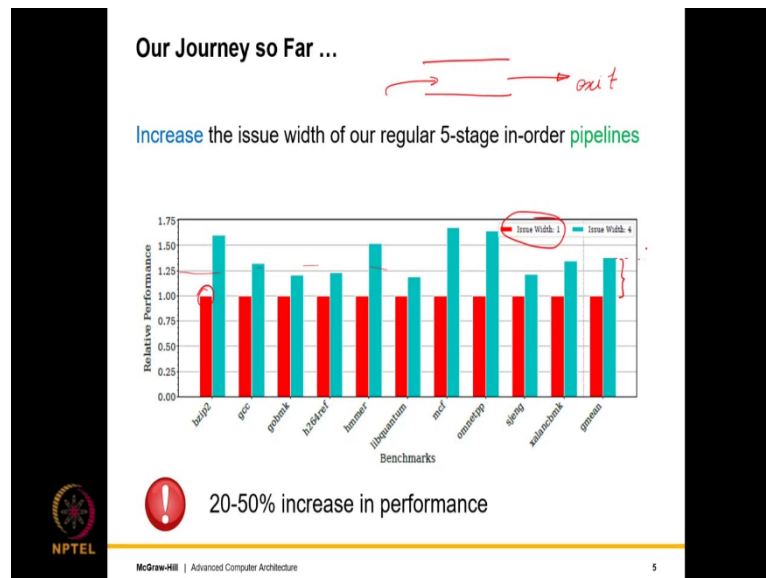
But more or less, the performance per Watt will increase exponentially which means that for every Watt of power expanded. We will get an increasing amount of performance in fact, exponential increase.

The performance increase per Watt has also stopped and the reason is that primarily because Moore's law has also slowed down, and it is expected to fully stop within a few years. So, this is also why Dennard's scaling has ended in fact, it ended in 2006 when this exponential increase in the performance per Watt this gradually started breaking down.

Nonetheless, computer architecture has moved on from there. So, previously most of the innovations that were happening were happening only because we could add more transistors

per chip, but the thinking in computer architecture post 2006 which is the last 15 years has changed substantially.

(Refer Slide Time: 08:21)



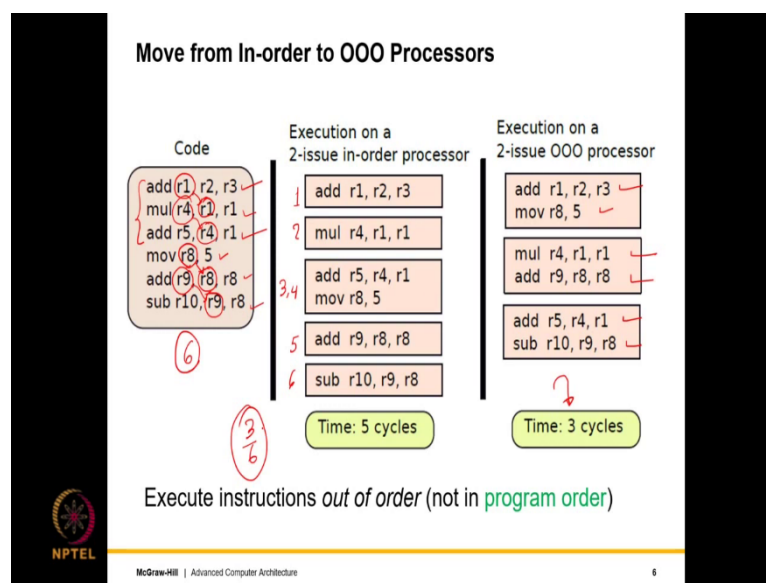
So, let us look at some of the basic results which sort of conclude a discussion on undergraduate computer architecture. The first is that if we take a regular 5 stage in order pipeline where basically if we look at the pipeline as a structure like this, instructions flow in, and they go through the 5 stages so, no instruction overtakes the other and then, they exit.

If we consider this to be our basic structures, then we always have the option of issuing which means sending down the pipeline one instruction or several instructions together. If we send several instructions together of course, the dependencies between the instructions matter and the additional scheduling and forwarding overheads are there. Nevertheless, there is a chance that the performance will increase mainly because of the additional parallels.

If we look at the SPEC benchmarks. so, SPEC is the standard performance evaluation cooperative. So, if you look at the SPEC benchmarks which is the traditional benchmarks that are used to measure performance and let's say that the performance is normalized to an issue width of 1 as you can see over here. If you have issue width of 4 in the sense of issue 4 instructions per cycle, we do not get the kind of speed up that we expect in the sense that the average speed up as you can see.

So, typically in these cases, a geometric mean is taken is actually not that great so, it is not even 50 %, I will just draw a line straight, straight, straight over here so, you will see this to be around at the ballpark of around 35 to 40% which is not much. So, the net increase for any benchmark is between 20 to 50% roughly and of course, bzip2, mcf and omnetpp show a higher performance increase. But that is said and done for most of the benchmarks the performance increase is not much and this much of performance increase one would get anyway with slightly faster transistors if 1 waits for 2, 3 or 4 years.

(Refer Slide Time: 10:37)



So, the next paradigm that was introduced which is something that we will talk very extensively in this book because an out-of-order processing is typically not a part of an undergraduate book, but it is part of a postgraduate book. So, I am assuming some familiarity with assembly language.

If you look at this piece of code over here, we will see that there are read after write dependencies in the sense here r1 is produced, r1 is consumed, register r1 that is and the convention being used here is that add r1, r2, r3, r2 + r3 is set to r1. And similarly, we can say that we are creating the value of r4 and then, reading it. So, there is a clear chain of dependencies here and there is another chain of dependencies over here.

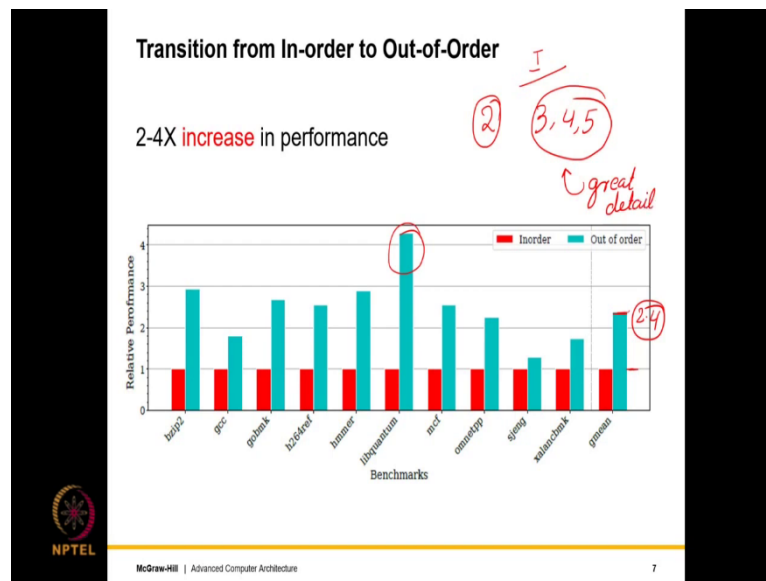
So, the traditional in order pipeline even if we are limited to let's say 2-issue in-order processor, we will execute the 1st instruction, then the 2nd, the only parallelism that we see is with the 3rd and 4th instructions, then again, the 5th and 6th instructions are executed without any other parallel instruction executing with the total execution time is 5 cycles.

If I were to execute the same piece of code on a regular in order pipeline one after the other that would have taken 6 cycles. So, the reduction in time even if we execute more issues in parallel assuming it is possible to do so is only 1, 6th which is not much. So, we do not like it, but if you are able to change the order of execution of instructions as long as any producer-consumer or any read-write dependencies are not violated so, then, we will have a 2-issue, out-of-order processor.

So, what we can do? is we can take this instruction and this instruction, see this and this. Similarly, in the next cycle, we can pick up this and this. How we pick up and so on is a separate matter that is a totally separate matter. We will discuss this in very main detail in chapters 4 and 5 of this book and then, we will pick up this and this, this and this.

So, if we look at it if we just break the in-order assumption and we execute instructions out-of-order, then the entire execution can finish in 3 cycles. So, if we execute them out-of-order as you can see out-of-order means not in program order, not in order in which they appear, then all that we require is 3 cycles which is a 2x speedup and this is substantial. So, we can see that we can even go slightly more than that with additional tricks.

(Refer Slide Time: 13:37)

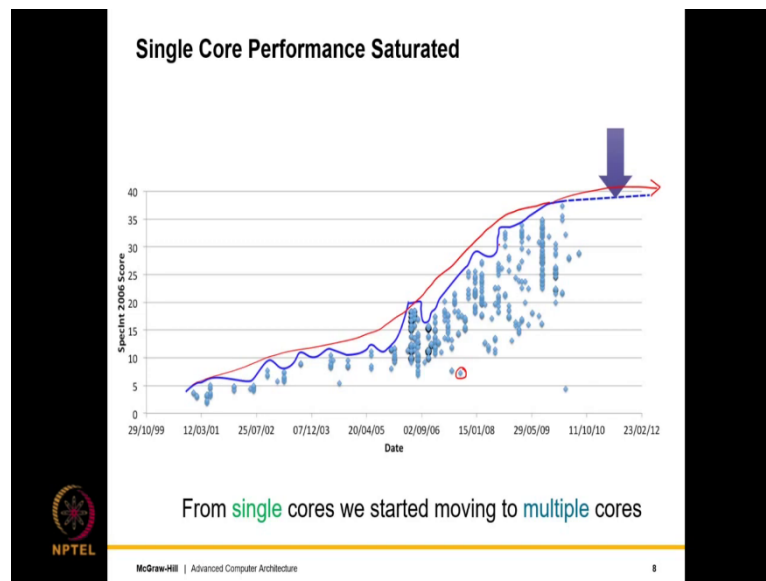


So, now, if we just compare the difference between in-order and out-of-order execution, you will see that the performance numbers are much higher. So, they are much much higher, they are not limited to the 20 to 50% range, instead if let's say this is 1, the average performance gain is 2.4. So, this is 2.4 times which is quite substantial and for some benchmarks, it is actually even more than that.

So, what you can basically see is that there is a huge performance gain, a massive performance gain something that is not seen otherwise and the reason it is not seen otherwise is because the modern out-of-order processor has many tricks that an in-order processor does not use.

So, in this book of course, the introductory chapter was chapter 1, we will motivate out of order process in chapter 2, then chapters 3, 4 and 5, we will discuss which is the 1st part of the book will discuss out-of-order chapters in great detail. So, in excruciating detail, we will discuss all the aspects of out-of-order execution in chapters 3, 4 and 5.

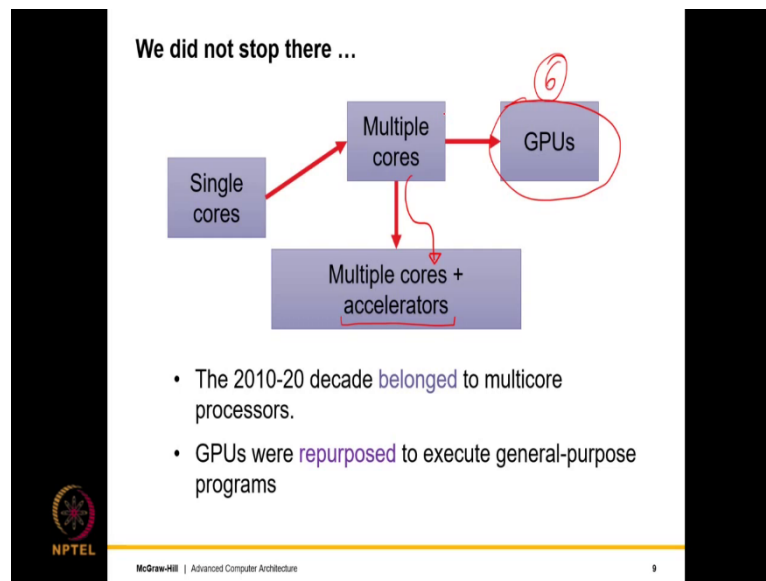
(Refer Slide Time: 14:55)



So, what we will observe towards the end of chapter 5 is that we meaning the entire industry and academic community has invested a lot of effort. A lot of design effort, a lot of thinking effort in increasing the performance of out-of-order processors. So, we started for clock from 2001 and go right up till 2012 where each point represents one processor that was released in this time frame and the y-axis is a specInt score, what we will see is that the performance gradually climbs up, up, up, up, up, up, up until you see a degree of saturation.

So, after 2012, you as you can see the performance is roughly saturated for a single core, a core we will define it more formally, but now think of it as a pipeline and a cache. The combo of a pipeline and a cache. so, a single core or a single processor, its performance saturated way back that is a decade ago and after that pretty much that area got stalled, development in that area got stalled because of saturation.

(Refer Slide Time: 16:10)

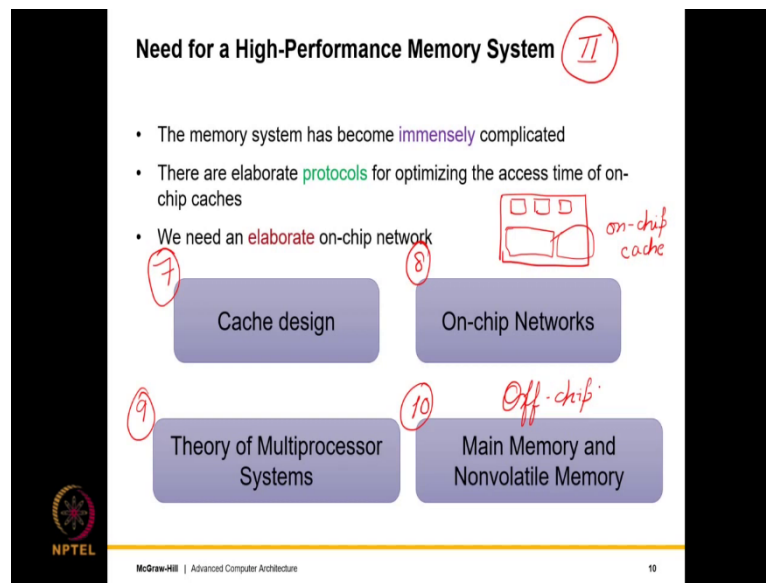


So, what people moved to was basically multicore processors. So, from single cores, they moved up, they moved up the hierarchy, up the chain and they went to multicore processors. So, instead of a single processor kind of on a chip, you have multiple small processors on a chip where each one of them can execute a program.

Recently from multicore processors, we have gone to multicore plus accelerators. For accelerator is a specialized processor that can do something specialized. For example, it can execute neural networks, it can do speech processing, it can process xml documents, encryption, compression. So, accelerators are for specific jobs and then, a very important subclass of multicore processors emerged which took off on its own, it had a life of its own which you will discuss in chapter 6, they are GPUs or graphics processors.

So, the 2010 to 20 decade belong to multicore processors of various use and colors. So, they were regular multicores with accelerators or GPUs, a lot of work was done in this space. So, GPUs again started as separate processors whose job was to only process graphics intensive tasks, but they gradually changed their color and they were repurposed to execute general purpose programs as well.

(Refer Slide Time: 17:43)



Then, our part I of the book comes to an end and we enter part II. So, part I was all about processing so, we finished that. Now, in part II, we will discuss the memory system. So, over the years in a multicore system, we need a very high bandwidth and low latency memory system and as a result, a lot of architectural tricks have to go in to designing the memory system and consequently, the memory system has become immensely complicated.


There are elaborate protocols for everything. particularly, for accessing, optimizing the access of on chip caches. So, I will tell you in a second what I am talking about? So, if we consider a chip, we have a large number of cores in here, we also have a large amount of on-chip memory. So, the on-chip memory is known as an on-chip cache and optimizing the behavior of the on-chip cache is exactly what is or let's say what has been driving innovation in this space. So, we will discuss cache design in chapter 7.

And if we have a large sea of caches, they need to be connected so, we need an on-chip network which is also a very deep field in its own right. So, I will discuss that in chapter 8. Then, when we consider multiprocessor memory systems, they get quite complicated. So, we will discuss them in chapter 9 and finally, in chapter 10, we will discuss off chip memory. So, the first 3 were for on-chip memory, but then, we will discuss off-chip memory.

So, off chip memory in general, they are DRAM memory models, but of late, other technologies are coming in such as ReRAM, STT memory, flash and so on. So, these are non-volatile memory units which means that they can continue to hold their state even if there is a power outage even if we turn the computer off, they will continue to hold their data; so, that is for non-volatile memory or NVMs. So, these 4 chapters broadly are a part of the second part of the book.

(Refer Slide Time: 20:08)

Assorted Issues *III part*

- 1** Power and temperature are **major** issues.
- 2** Reliability: transient faults and [hard errors]. *12* →
- 3** Parameter variation: variations in the process, voltage, and temperature
- 4** Security  *13*

NPTEL
McGraw-Hill | Advanced Computer Architecture 11

The 3rd part of the book, we will discuss assorted issues. So, it will 1st start with chapter 11 which talks about power and temperature. Power and temperature are major issues no doubt. So, they have stopped the further scaling of processors primarily because we cannot increase the temperature or increase the complexity any further because the power dissipation and the resultant temperature rises will become prohibited.

They cause reliability problems; they are a major reason power and temperature is maybe the single largest reason for causing reliability problems. So, they cause either short term faults or hard errors and along with that, we will discuss a few more interesting mechanisms via which processors can fail. So, you will find that particles generated in outer space can come as cosmic rays and then, they can cause faults in our processor and the fault mechanism is

quite interesting. so, I do not want to steal the fun, you will get to read more about it in chapter 12.

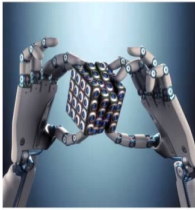
And chapter 12 also contains some phenomenon called parameter variation which means that when you are trying to fabricate a transistor or operate it, our operating conditions or fabricating conditions may be non-ideal, or the results may be non-ideal. So, as a result, a transistor that is seen in an electron microscope after it has been fabricated, its specs may differ from ideal specs and of course.

If you bring in slightly fluctuating voltages and temperature, then transistors are operating in quite an unstable and variable environment. Security everything needs to be secure these days; security is a major challenge we will discuss this in chapter 13.

(Refer Slide Time: 22:08)

Architectures for Machine Learning

(14)



camera
↓
processor

- This is the future.
- Dedicated on-chip accelerators for accelerating computer vision, machine learning and speech processing.

NPTEL

McGraw-Hill | Advanced Computer Architecture

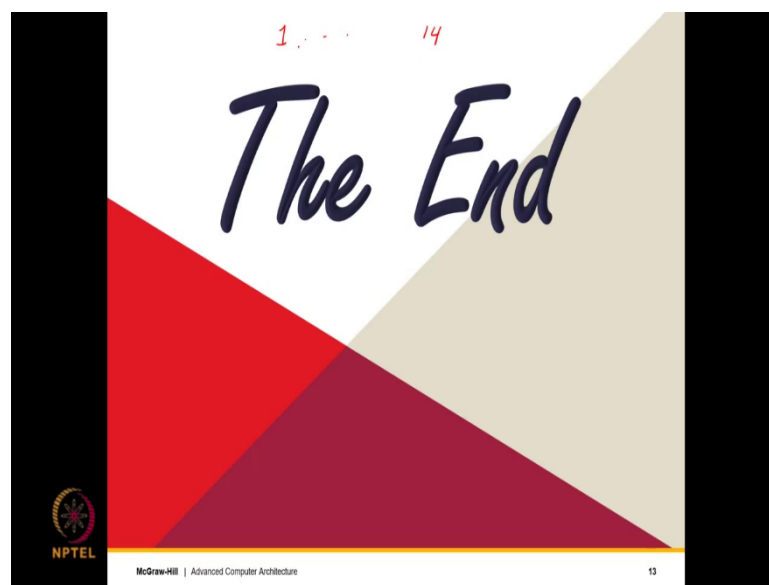
12

And finally, the last chapter, which is chapter 14, will capture an aspect of processor design which is expected to dominate in the next decade. So, next decade is actually the current decade which is 2022 to 2030. This is the future in the sense; we will have large chips with a lot of on-chip accelerators for different tasks such as computer vision, machine learning, speech processing to essentially add many features of intelligent computing into regular processors such that they can be used for all kinds of applications future of an advanced nature.

What would be advanced natures? Think of gesture based computing which basically means that you know I stand in front of the TV, I wave my hand and the channel changes, the way that that happens is there is a small camera that captures the input, sends it to the processor. So, you just look at the way that this happens, there is a small camera, it captures the input, it sends it to the processor.

So, clearly the amount of processing required is too much for a regular processor, but it has a small machine learning accelerator which is quite powerful, it has been built for this purpose. It processes the result and sends the result to the host program. The host program is a program that controls the channel of the TV. so, it increments the channel or decrements the channel.

(Refer Slide Time: 23:54)



So, we will be seeing, increasingly seeing innovations of this type particularly in different machine learning areas not just included to vision and speech. So, let's see what the future may hold, but at least we will look at these 14 chapters of this book. In the series of video lectures and as I said they are divided into 3 parts; with the Ist part is processing. The IInd part is the design of the memory system and the IIIrd part looks at assorted issues that include security as well as machine learning.