Components of a Processor
Switches, Gates, and some History
To be able to effectively program a modern multiprocessor, we have to understand what it is made up of and how it came to be the way it is today
The Modest Switch

• We have been able to make a “Machine” that can do complex things
  • Weather forecast, design of medicinal drugs
  • Speech recognition, Robotics, Artificial Intelligence..
  • Web browsers, internet communication protocols

• What is this machine based on?
What Are Computers Made of?

• Building blocks of computers
• Why has frequency scaling stalled?
• Conception of parallel computing
• Machine organization
• ........

Complexity of Modern Processors Makes Performance Oriented Programming Challenging
The Modest Switch

• All these capabilities are built from an extremely simple component:
  • A controllable switch

• The usual Electrical switch we use every day
  • Turns current on and off
  • But we need to turn it on and off by hand

• The Controllable Switch
  • Voltage controls if the switch is on or off
  • High voltage at input: switch on
  • Otherwise it is off
Let's use them creatively

Output is high if both the inputs input1 AND input2 are high.

If either of the inputs is low, the output is low.

This is called an AND gate.

Now, can you make an OR gate with switches?

L.V. Kale
OR Gate

Output is high if either Input1 or input2 (or both) is high
Basic Gates

• There are three basic kinds of logic gates

**Operation:**
- **AND of two inputs**
- **OR of two inputs**
- **NOT (complement) on one input**

**Logic gate:**
- AND: \( \begin{array}{c}
  x \\
  y
\end{array} \rightarrow \quad xy \)
- OR: \( \begin{array}{c}
  x \\
  y
\end{array} \rightarrow \quad x + y \)
- NOT: \( \begin{array}{c}
  x
\end{array} \rightarrow \quad x' \)

• Two Questions:
  - How can we implement such switches?
  - What can we build with Gates?

  • Adders, controllers, memory elements, computers!

L.V. Kale
You can make an adder with gates

4 bit adder
How About some memory?

• Switches help us build memory
• How can a circuit “remember” anything on its own?
  • After all, the values on the wires are always changing, as outputs are generated in response to inputs.
• The basic idea is feedback: we make a “loop” in the circuit, so the circuit outputs are inputs as well

Set and Reset inputs...

When S and R are 0, Q is “stable”: whatever it was, it stays in that state. Ergo: memory.

When S is 1 and R is 0, Q becomes 1
When R is 1 and S is 0, Q becomes 0
Components of a Stored-Program computer

Instruction Memory → PC (Program Counter) → ALU → Data Memory

Move to the next location

Register Set

L.V. Kale
How computers are built from switches/gates is matter for other courses. For this course, remember:

Computers are made from some very simple components by putting a large number of them together in interesting ways, and running them really fast.
A Brief Look at Early History of Computers
Making Switches and some History

UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN
How to make switches?

• Use mechanical power
• Use hydraulic pressure
• Use electromechanical switches (electromagnet turns the switch on)
• Current technology:
  • Semiconductor transistors
    • A transistor can be made to conduct electricity depending on the input on the 3rd input
  • CMOS “gates” (actually, switches)
Some old history: Analytical Engine

• In 1837 (!) Charles Babbage designed (but not built) the first machine that is close to a modern computer
  • Ada Lovelace, daughter of Lord Byron, has been called the first programmer for this machine
  • Mechanical computer

Image By Bruno Barral (ByB), CC BY-SA 2.5, https://commons.wikimedia.org/w/index.php?curid=6839854
Some old history: continued

• Konrad Zuse (1941, in Germany) built Z3
  • Electromechanical
  • Lacked conditional branches but could do loops

• 1940: Alan Turing helped design **Bombe**, for decoding German messages, but it wasn’t a full fledged computer

• World war two led to modern designs based on vacuum tubes as switches
  • Colossus Mark I (1943)
  • ENIAC (1946): first modern programmable computer

• 1945-46: *Designs* for general purpose stored-program computer:
  • Alan Turing Designed ACE and von Neumann designed EDVAC
Old history continued

• 1951: ILLIAC I: First academic computer at Univ of Illinois
  • You see parts of it in a showcase my department’s atrium

• 1955 onwards: Vacuum tubes were slowly replaced by transistors

A Chassis from ILLIAC I with vacuum tubes

A Board from ILLIAC II with transistors
Old history continued: microprocessors

• 1960s: Integrated circuits: engineers were able to put multiple switches (transistors) on a single chip

• 1971: First microprocessor. Intel 4004:
  • An entire CPU on a single chip, albeit a 4-bit processor

• 1972: 8008
  • 1974: 8080
  • 1978: 8086

What happens over the next 30 years is a fascinating story, and one that has consequences for our objective of learning how to write fast parallel programs.
Moore’s Law

What fueled performance increases in processors between 1974-2004

UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN
Clock Speeds

• In our simple computer:
  • A single program instruction is executed in a “clock cycle”
Components of a Stored-Program computer

Instruction Memory -> PC (Program Counter) -> CPU -> Data Memory

In one clock cycle: Instruction is fetched from memory, operands fetched, arithmetic performed, results stored back.

L.V. Kale
Clock Speeds

• In our simple computer:
  • A single program instruction is executed in a “clock cycle”
  • Instruction is fetched from memory, operands fetched, arithmetic performed, results stored back
  • Clock time period is selected so that the parts of the computer can finish basic calculations within the cycle

• If we can make transistors smaller
  – Which means smaller capacitances..
    • Imagine filling up “tanks” with “water” (electrons)
• We can turn them on or off faster
  – Which means we can make our computers go faster

Two properties of Switches and Gates: Size, Switching and Propagation delay
The Virtuous Cycle

• If you can make transistors smaller,
  • You can fit more of them on a chip
    • Cost per transistor decreases
    • AND: propagation delays get smaller
      • So they can run faster!

• Can you make them smaller?
  • Technological progress needed, but can be done

• This led to:
  • Cheaper and faster processors every year
Moore’s law

• Commonly (mis) stated as
  • “Computer performance doubles every 18 months”

• Gordon Moore observed in 1965
  • “The complexity... has increased roughly a factor of two per year. [It] can be expected to continue...for at least 10 years”
  • Its about **number of transistors** per chip

• Funny thing is: it held true for 50+ years
  • And still going until 2020
  • “Self fulfilling prophecy”
Clock Speeds Increased

Notice a little trick: x axis goes only to 2003!
Until they stopped increasing!

Why?
Prediction in 1999
From Shekhar Borkar, Intel, at MICRO’99

So, the chips were getting too hot
Source: Herb Sutter, DDJ
http://www.gotw.ca/publications/concurrency-ddj.htm

Also see: https://www.karlrupp.net/2015/06/40-years-of-microprocessor-trend-data/
Number of Transistors/chip?

• Well, they will keep on growing for the next several years
  • May be a bit slowly

• Current technology is 10-14 nanometers
  • AMD EPYC 7401P (19.2 billion transistors on 4 dies on the package)
  • 10 nm

• We may go to 5 nanometers feature size
  • i.e. gap between two wires (as a simple definition)

• For comparison:
  • Distance between a carbon and a Hydrogen atom is 1 Angstrom = 0.1 nanometer!
  • Silicon-Silicon bonds are longer
  • 5 Å³ lattice spacing (image: wikipedia)
    • i.e. 0.5 nanometer
  • So, we are close to atomic units!
Consequence

• We will get to 30-50 billion transistors/chip!
• Even in 2006, Pentium 4 last edition had 185M transistors
• What to do with them?
  • We will learn later about what was done between 1973-2003 with this bounty of transistors to improve performance
  • And its consequences for performance-oriented programming
  • But then, around 2003, all these methods, including increasing on chip memory (caches) etc. all ran to dead end
Moore’s Law and Parallelism
Consequence

• We will get to 30-50 billion transistors/chip!
• Even in 2006, Pentium 4 last edition had 185M transistors
• What to do with them?
  • We will learn later about what was done between 1973-2003 with this bounty of transistors to improve performance
  • And its consequences for performance-oriented programming
  • But then, around 2003, all these methods, including increasing on chip memory (caches) etc. all ran to dead end
A Way Out?

• Put more processors (now called “cores”) on a chip
• Beginning of the multicore era after 2003
  • Number of cores per chip doubles every $X$ years
    • $X = 2? 3?$
A Way Out: Parallelism

• We were used running applications faster with each generation of machine
  • Or at least, run more bloated (well.. Sophisticated) software at the same speed

• But now:

• Individual cores were not getting much faster
  • Clock speed stuck at somewhere near 3 GHz

• But we have multiple cores on a chip

• Must use multiple cores to solve one problem faster
  • I.e. Parallel Programming
Bill Wulf in 1978

- William Wulf is one of the most influential computer scientists

- Visited IISc Bangalore around 1978..
  - When I was a grad student

- Talked about his parallel computing projects
  - C.mmp
  - Stated motivations: Sequential processors cannot keep getting faster forever, because of physical limitations.
    We need many processors working in parallel to compute faster

- But engineers kept making it go faster
  - Until now
Status

To summarize:
- We had been used to computers becoming faster every year. That “change” was a constant.
- The revolution is: that the speeds are no longer changing.
- Multiple processors (cores) on a chip is about the only way to utilize the extra transistors we get via Moore’s law.
Era of Parallel Computing

UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN
Parallel Computing is Inevitable.. No?

• So, parallelism is finally here and will get to hundreds of cores per chip.. No?
Two problems

• Maybe we have all the speed we need..
  • I.e. for all the apps that we need
  • Nyah..

• Maybe 8 cores is all that you need
  • We are still seeing improvements because
    • We use multiple programs on the desktop
    • Browsers can do multiple things: get data, draw pictures, ..
  • But now, we have enough power.. Right?

• So, unless one (or more) parallel “killer app” appears, the market (for multicore chips) will stop growing
Alternative: Parallelism

• If we find killer apps that need all the parallel power we can bring to bear
  • With 50B transistors, at least 100+ processor cores on each chip
  • Already:
• There is a tremendous competitive advantage to building such a killer app
  • So, given our history, we will find it
• What are the enabling factors:
  • Finding the application areas.
  • Parallel programming skills
A Few Candidate Areas

• Simple parallelism:
  • Search images, scan files, ..

• Speech recognition:
  • almost perfect already
  • But speaker dependent, minor training, and needs non-noisy environment
  • Next: speaker independent recognition with noisy environment

• Real-time video processing

• Broadly: Artificial intelligence and Machine Learning
• Data centers (data analytics, queries, cloud computing)

• And, of course, HPC (High Performance Computing)
  • typically for CSE (Computational science and Engineering)
Parallel Programming Skills

• So, all machines will be (are?) parallel
• So, almost all programs will be parallel
  • True?
• There are 10 million programmers in the world
  • Approximate estimate
• All programmers must become parallel programmers
  • Right? What do you think?
Programming Models Innovations

• Expect a lot of novel programming models
  • There is scope for new languages, unlike now
  • Only Java broke through after C/C++/C# for compiled languages

• This is good news:
  • If you are a computer scientist wanting to develop new languages

• Bad news:
  • If you are an application developer

• DO NOT WAIT FOR “AutoMagic” parallelizing compiler!
A Little Peek into the Future

UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN
Consequences of Moore’s Law Ending

• CMOS, the microprocessor’s semiconductor technology, for sure, will stop advancing soon
  • And except for quantum computing for a narrow class of problems, no other technology is in sight

• Let us try to predict what the future will look like
Multicore Computers

• Laptops, (maybe even desktops if they survive), server nodes, cellphones
  • Also, of course, single nodes in a supercomputer, also multicore processors too

• Depends on the killer applications of the near future
• With GPGPUs and other specialized accelerators
Small Clusters

• Probably some of the biggest impact

• Broadening of the market

• Every company/department can afford a very powerful (100 TF? PF?) cluster

• All kinds of activities can be computerized
  • Intel’s example:
    • Fashion designers examining how a cloth will drape over a body, and how it will move
    • Via simulation

• Operations research

• Business strategies via AI support
Supercomputers

• Exascale will be reached by 2022
  • May be 50 MW, and $10^{18}$ ops/s

• I expect
  • They will create breakthroughs in science and engineering of great societal impact
  • Biomedicine, materials
  • Astronomy, physics: theories
  • Engineering design of better artifacts
  • Controlled nuclear fusion (fission) may solve energy problems
  • Climate Modeling?

• If society deems it beneficial, technology can be developed for beyond-exascale (1000 Eflops?)
Next Era: End of Moore’s Law

• 10-ish years from now
  • Maybe 5… but the end-phase will be slowed down
• No more increase in performance from a general purpose chip!
• What can we predict about this era?
  • First, innovation would shift to functional specialization
    • Would have started happening already
  • Next, innovation will shift to application areas, and molecular sciences: biomedical (nanobots?), materials
  • Another 5-10 years, you can develop CSE applications knowing that the machine won’t change under your feet
    • Maybe
Summary of Introduction

• Times are changing:
  • I.e., they are getting more stagnant!

• Those who can “get” parallel, will have an advantage

• If killer parallel app doesn’t arrive, progress on single multiprocessor chips will stall

• Complete “stasis” after 7-10-ish years
  • But then, such things have been predicted before
Caution: Predicting Future

• Remember:
  • 1900 or so: “End of science” predicted
  • 1990 or so: “End of history” predicted