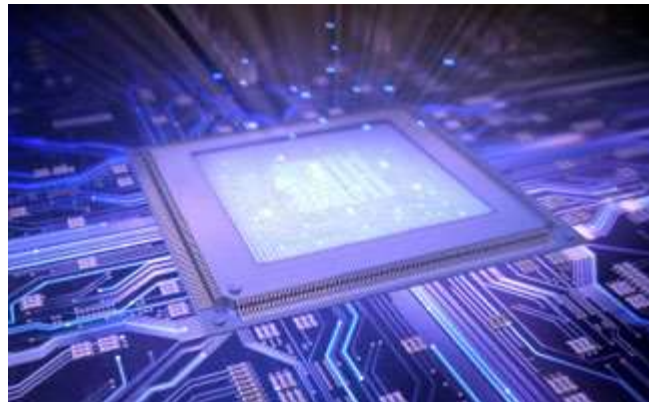




CSCI-UA.0480-003  
**Parallel Computing**

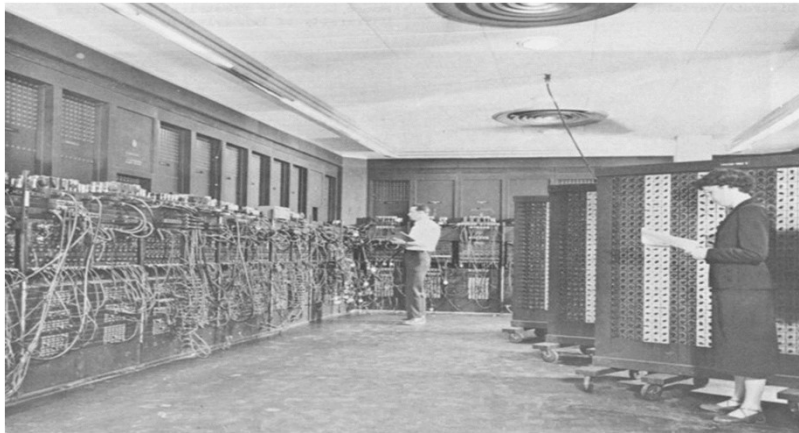
**Lecture 2: Parallel Hardware: Basics**

Mohamed Zahran (aka Z)  
mzahran@cs.nyu.edu  
<http://www.mzahran.com>



# Computer History

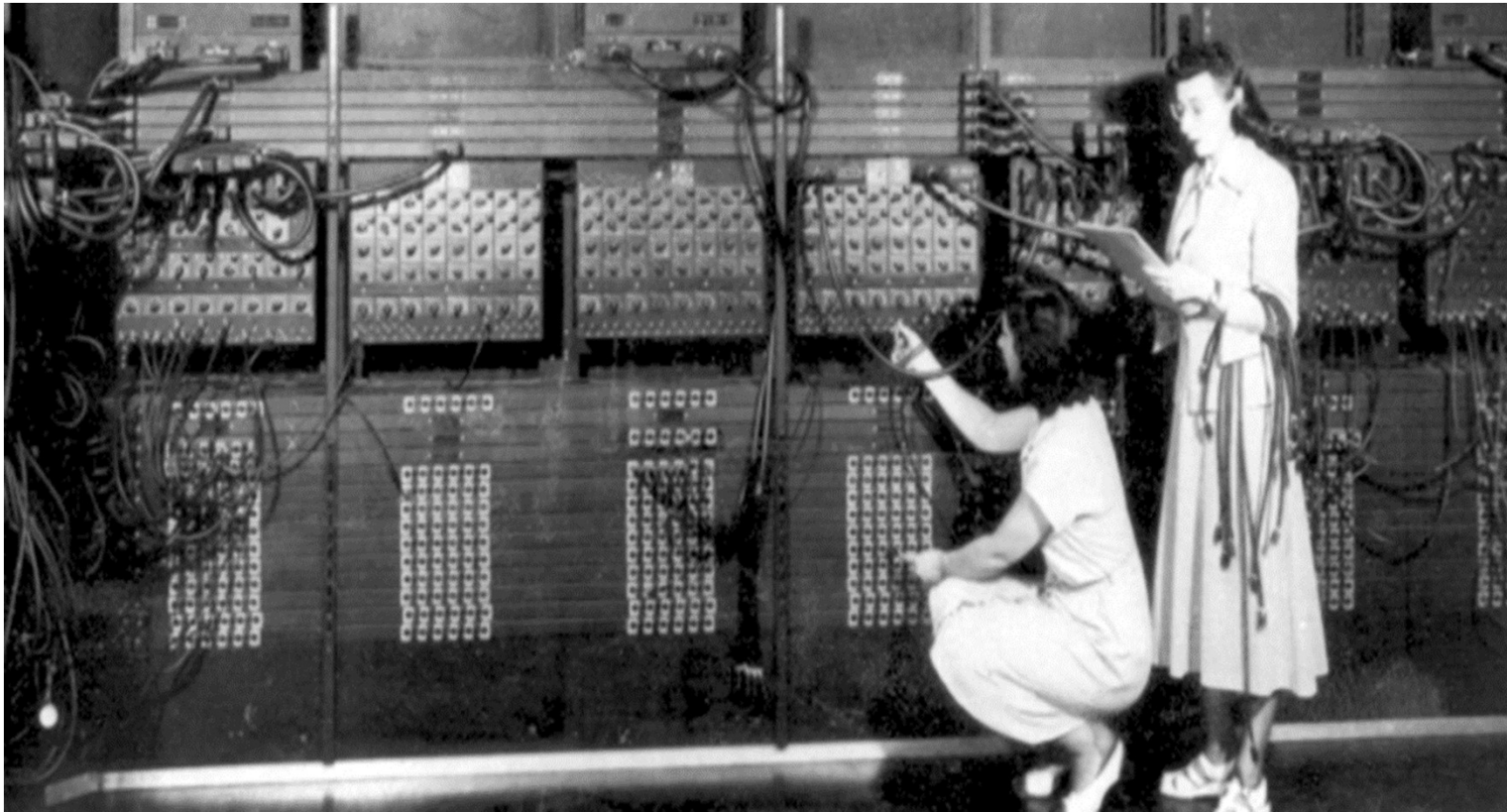
Eckert and Mauchly



ENIAC

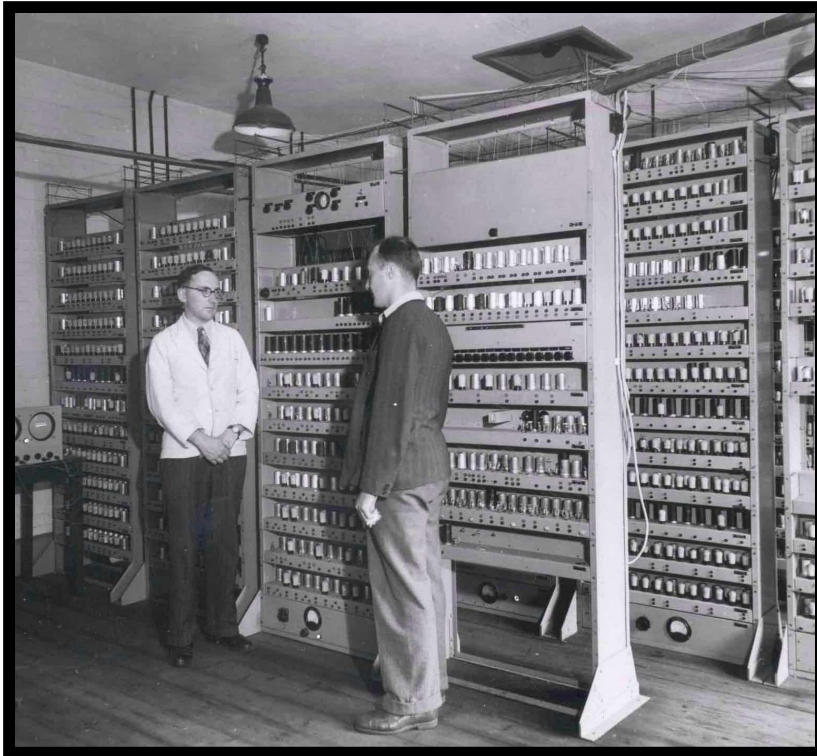
- 1<sup>st</sup> working electronic computer (1946)
- To reprogram it you need to re-arrange the cords
- 18,000 Vacuum tubes
- 1,800 instructions/sec
- 3,000 ft<sup>3</sup>

# Computer History



Programming the ENIAC!

# Computer History



EDSAC 1 (1949)

<http://www.cl.cam.ac.uk/UoCCL/misc/EDSAC99/>

- Von Neumann presented his idea of **stored program concept**.
- Maurice Wilkes built it.



1<sup>st</sup> stored program  
computer  
650 instructions/sec  
1,400 ft<sup>3</sup>

# Computer History

- After the vacuum tubes, **transistors were invented** (1947) → 2<sup>nd</sup> generations of computers



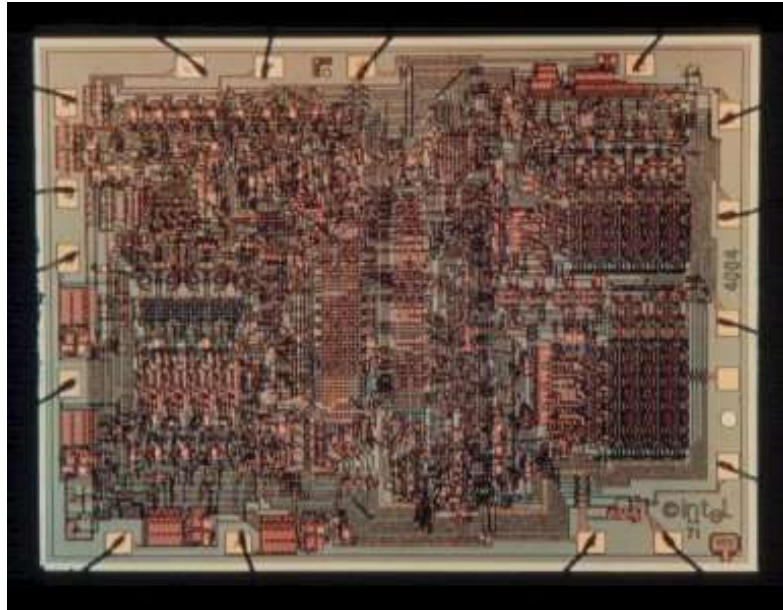
- UNIVAC (UNiversal Automatic Computer)
- Introduced in the 50s



# Computer History

- From transistors to **integrated circuits(IC)** → 3<sup>rd</sup> generation of computers
- One IC can host hundreds (then thousands, then millions) of transistors  
→ computers are getting smaller

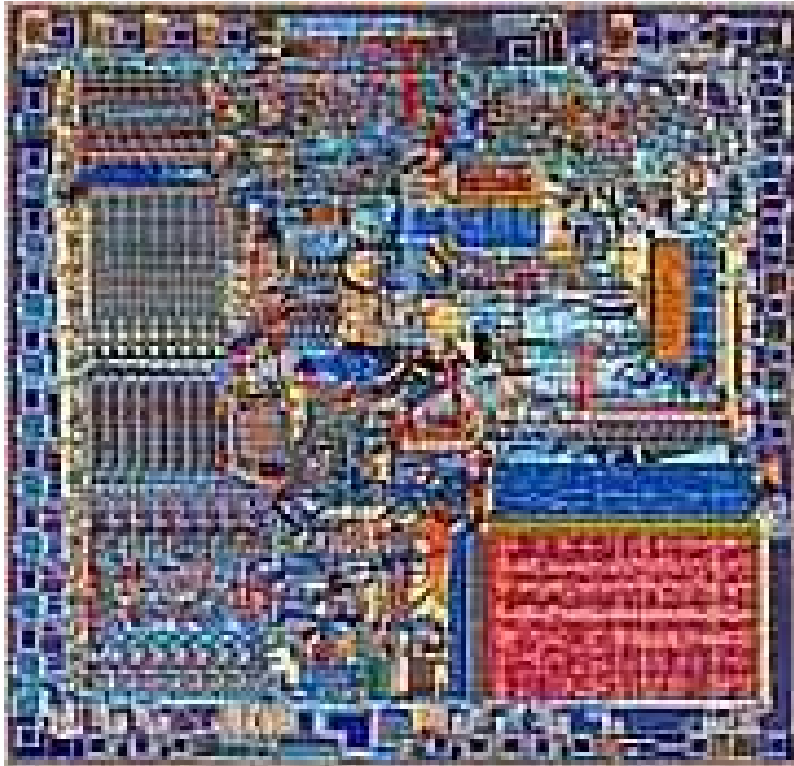
# Intel 4004 Die Photo



- Introduced in 1970
  - First microprocessor
- 2,250 transistors
- 12 mm<sup>2</sup>
- 108 KHz



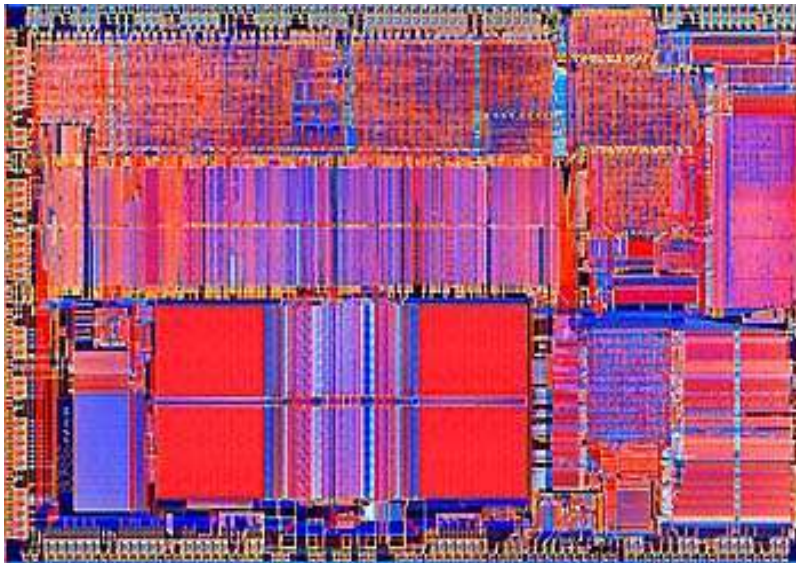
# Intel 8086 Die Scan



- 29,000 transistors
- 33 mm<sup>2</sup>
- 5 MHz
- Introduced in 1979
  - Basic architecture of the IA32 PC

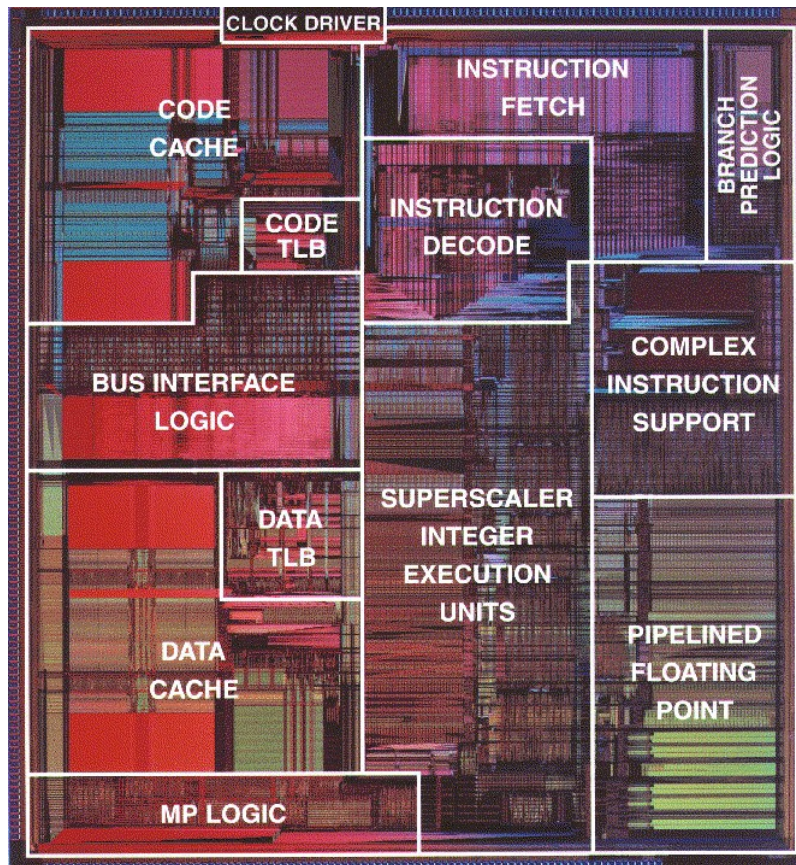


# Intel 80486 Die Scan



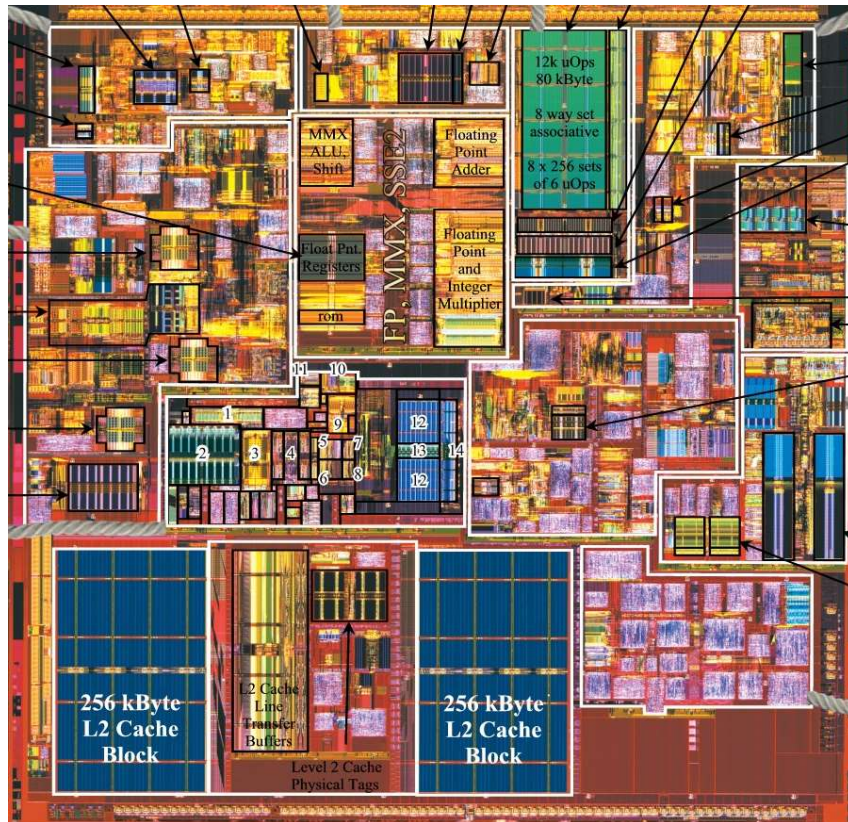
- 1,200,000 transistors
- 81 mm<sup>2</sup>
- 25 MHz
- Introduced in 1989
  - 1<sup>st</sup> pipelined implementation of IA32
  - 1<sup>st</sup> processor with on-chip cache

# Pentium Die Photo



- 3,100,000 transistors
- 296 mm<sup>2</sup>
- 60 MHz
- Introduced in 1993
  - 1<sup>st</sup> superscalar implementation of IA32

# Pentium 4



55,000,000  
transistors

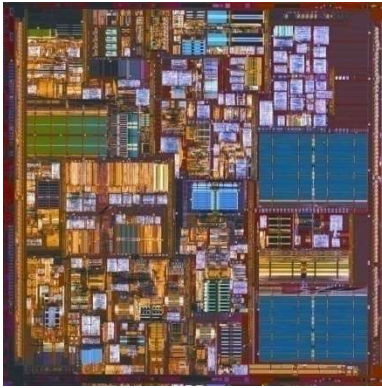
146 mm<sup>2</sup>

3 GHz

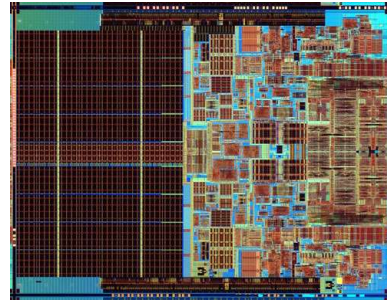
Introduced in 2000

<http://www.chip-architect.com>

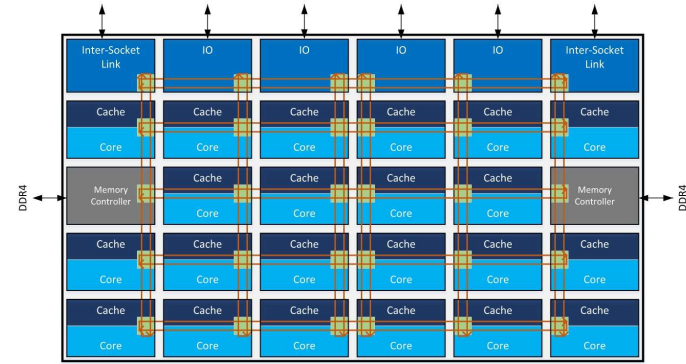




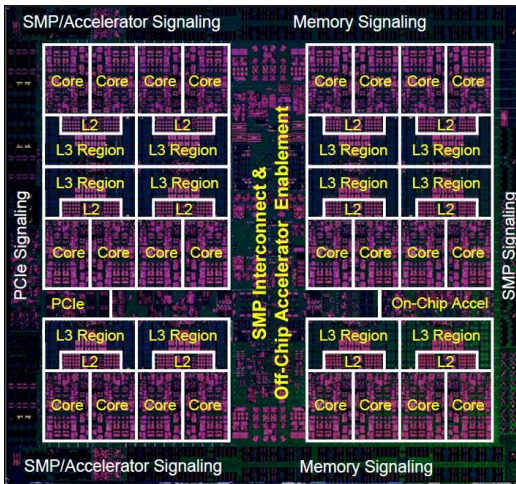
Pentium 4



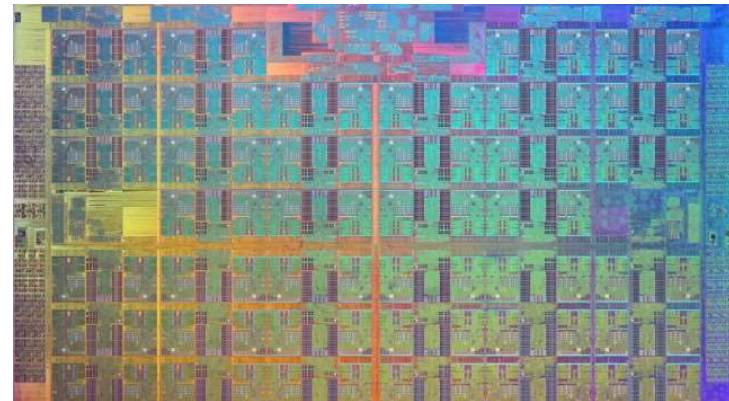
Core 2 Duo (Merom)



Intel Core i9 (Kaby-lake )



IBM Power 9 (24 cores)

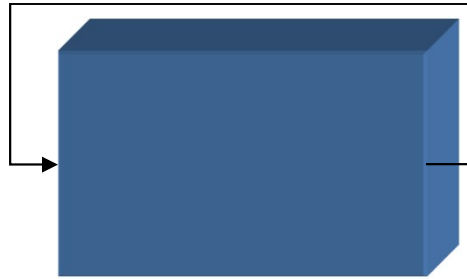


Intel Xeon Phi (72 cores)

How did the hardware evolve like  
that?

Let's look at different waves (generations of architectures)

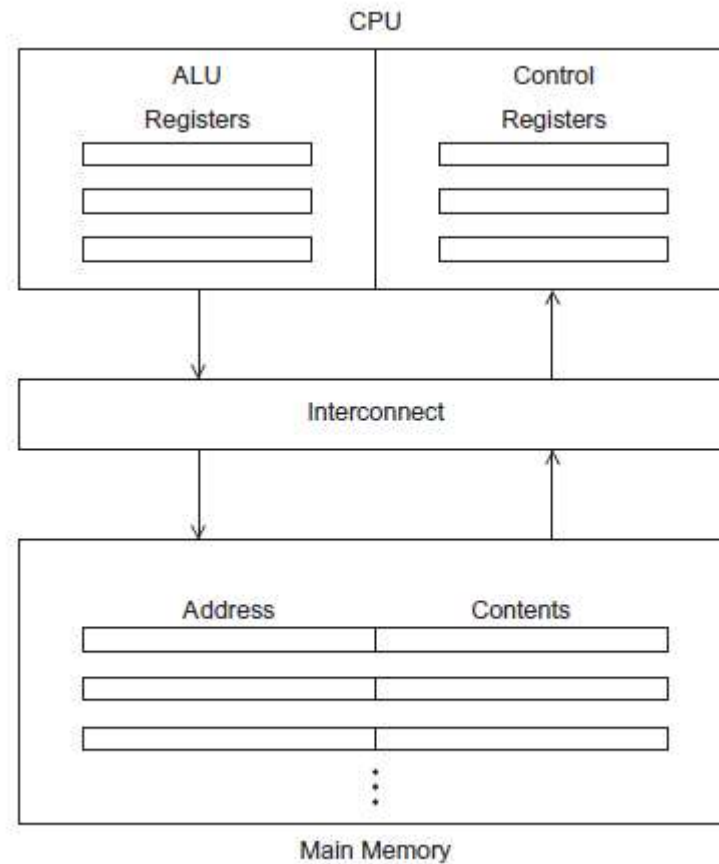
# First Generation (1970s)



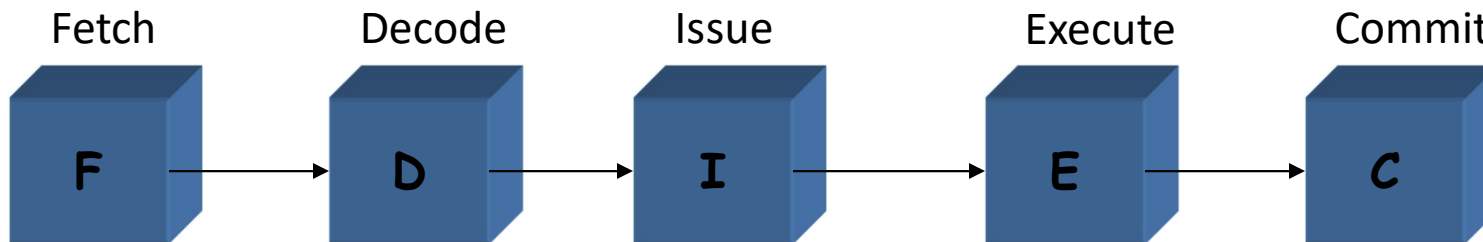
Single Cycle Implementation



# The Von Neumann Architecture



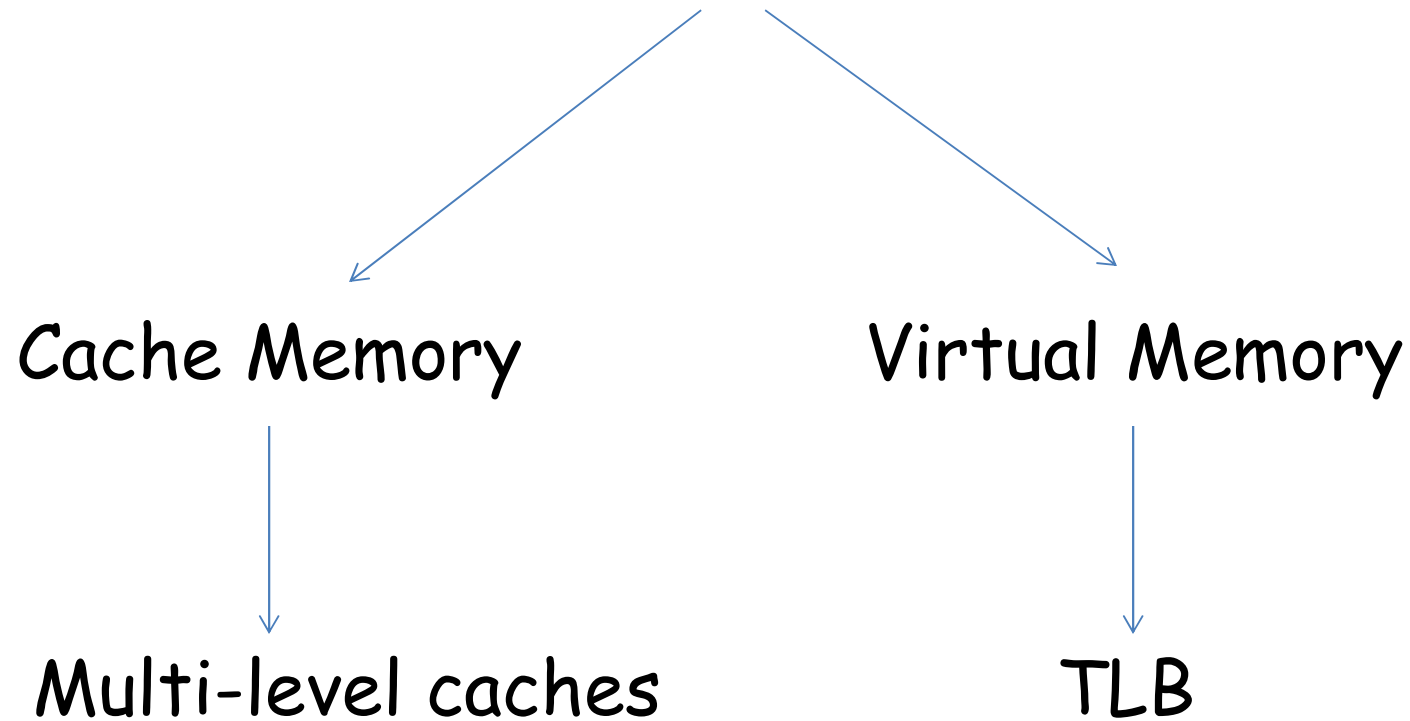
# Second Generation (1980s)



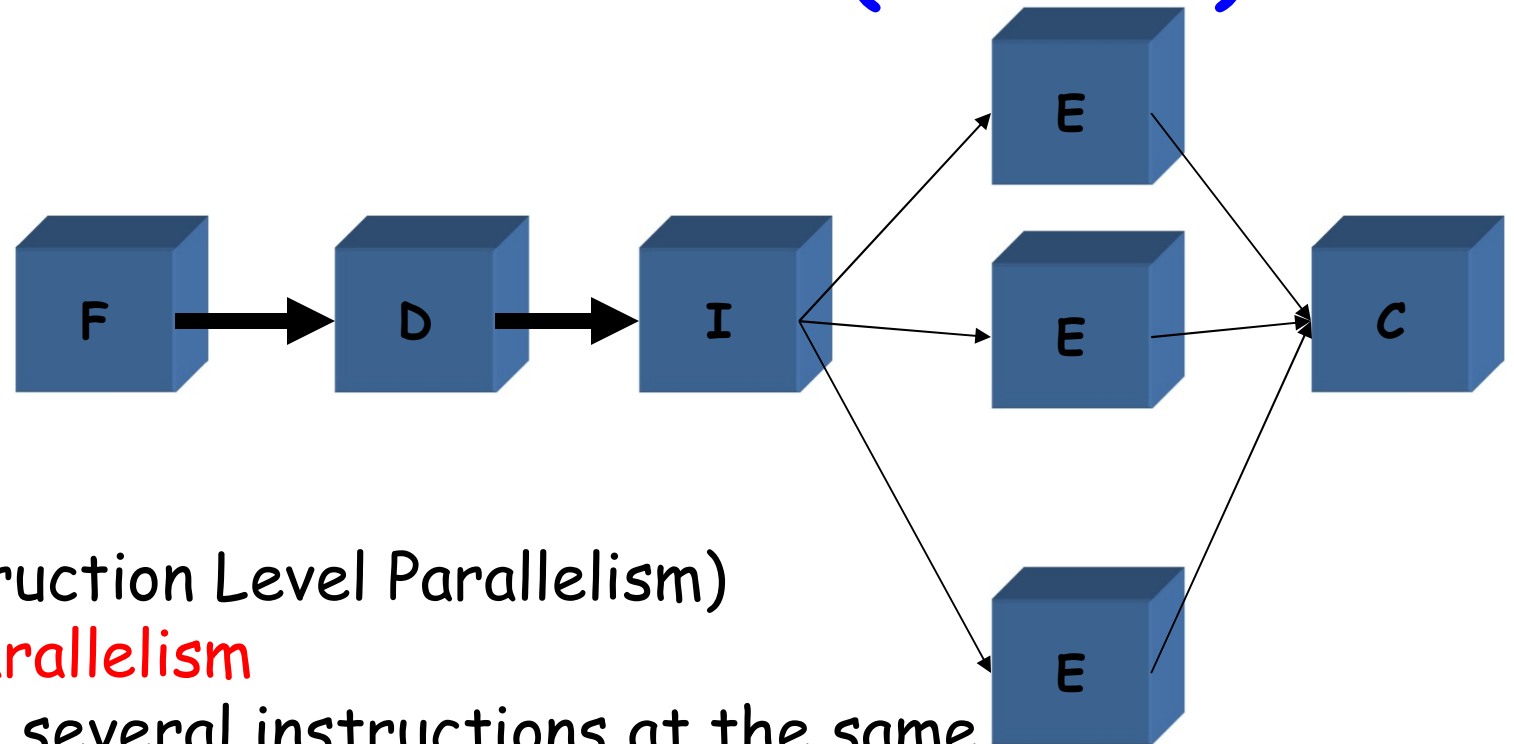
- **Pipelining:**

- the hardware divided into stages
  - **temporal parallelism**
  - Number of stages increases with each generation
- 
- Maximum **CPI** (Cycles Per Instruction) = 1
    - Due to dependencies  
(i.e. an instruction must wait for the result of another instruction)

# Some Enhancements

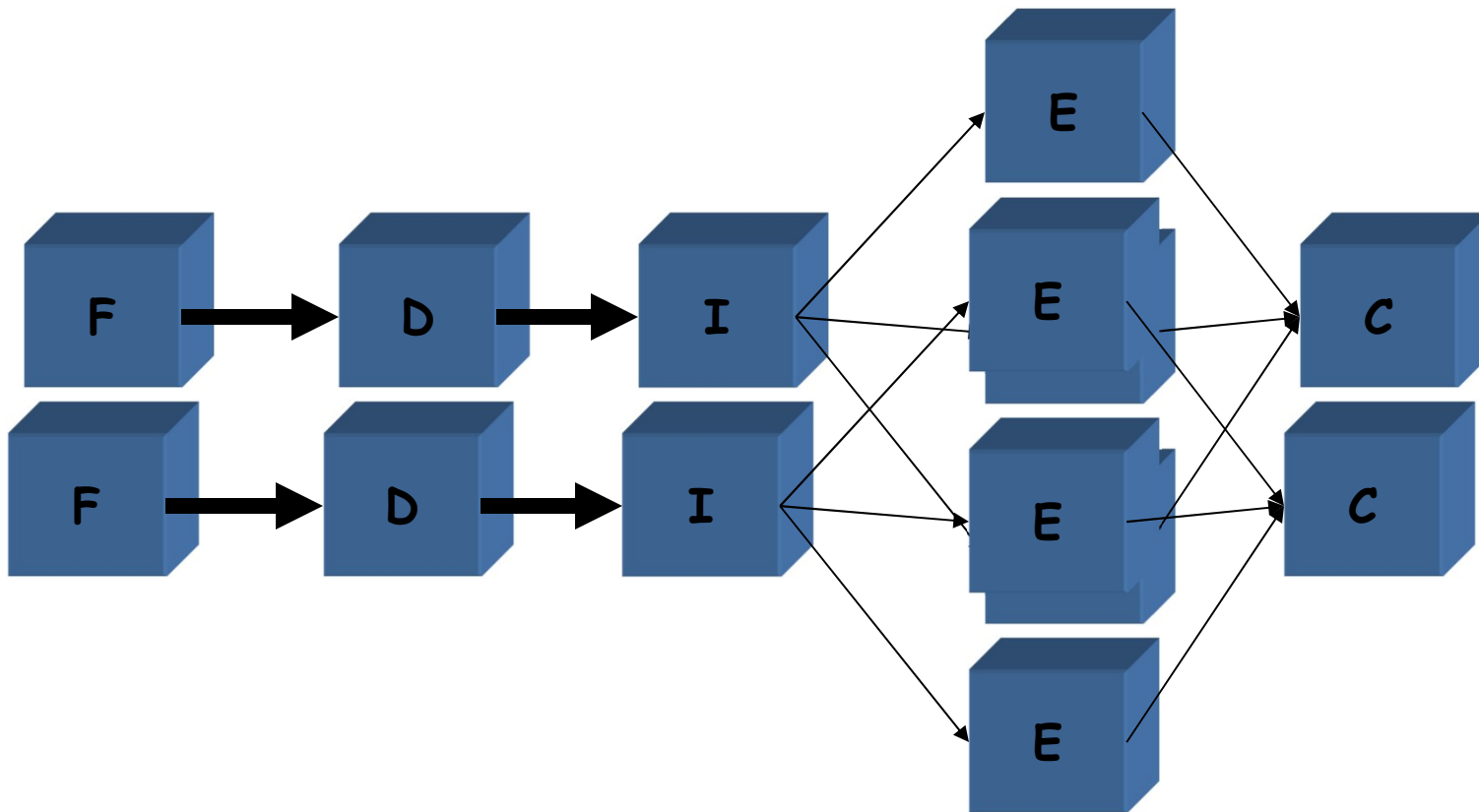


# Third Generation (1990s)



- **ILP** (Instruction Level Parallelism)
- **Spatial parallelism**
- Executing several instructions at the same time is called **superscalar** capability.
- performance = instructions per cycle (**IPC**)
- **Speculative Execution** (prediction of branch direction) is introduced to make the best use of superscalar capability → This can make some instructions execute **out-of-order**!!

# Fourth Generation (2000s)



Simultaneous Multithreading (SMT)  
(aka Hyperthreading Technology)

Some definitions before we proceed



# An operating system "process"

- An instance of a computer program that is being executed.
- Components of a process:
  - The executable machine language program
  - A block of memory
  - Descriptors of resources the OS has allocated to the process
  - Security information
  - Information about the state of the process

# Multitasking

- Gives the illusion that a single processor system is running multiple programs simultaneously.
- Each process takes turns running → **time slice**
- After its time is up, it waits until it has a turn again.

# Threading

- Threads are **contained within processes**.
- They allow programmers to divide their programs into (more or less) independent **tasks**.
- The hope is that when one thread blocks because it is waiting on a resource, another thread will have work to do and can run.

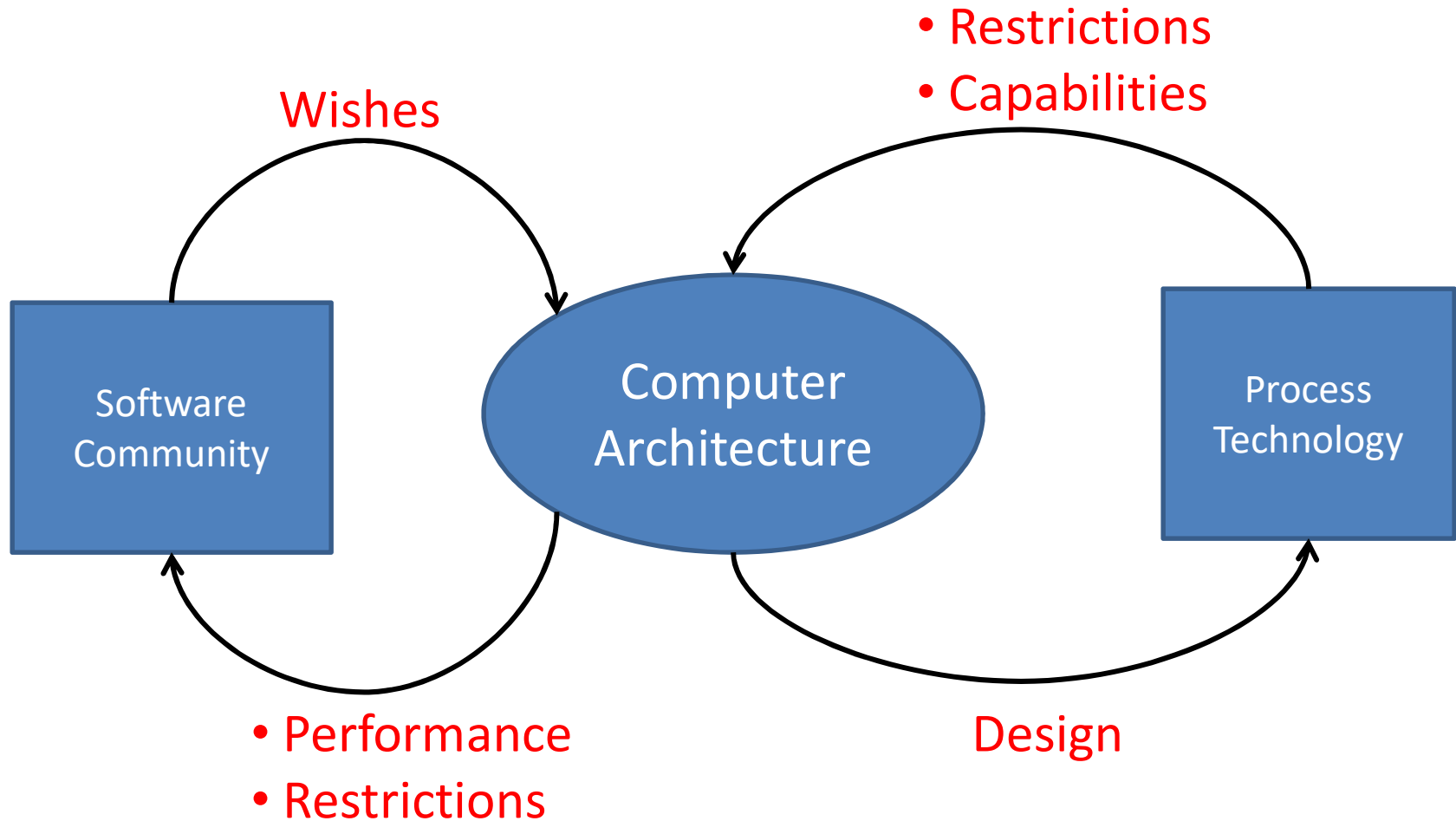
As you can see ...

We can have several processes, executed in a multitasking fashion, and each process can consist of several threads.

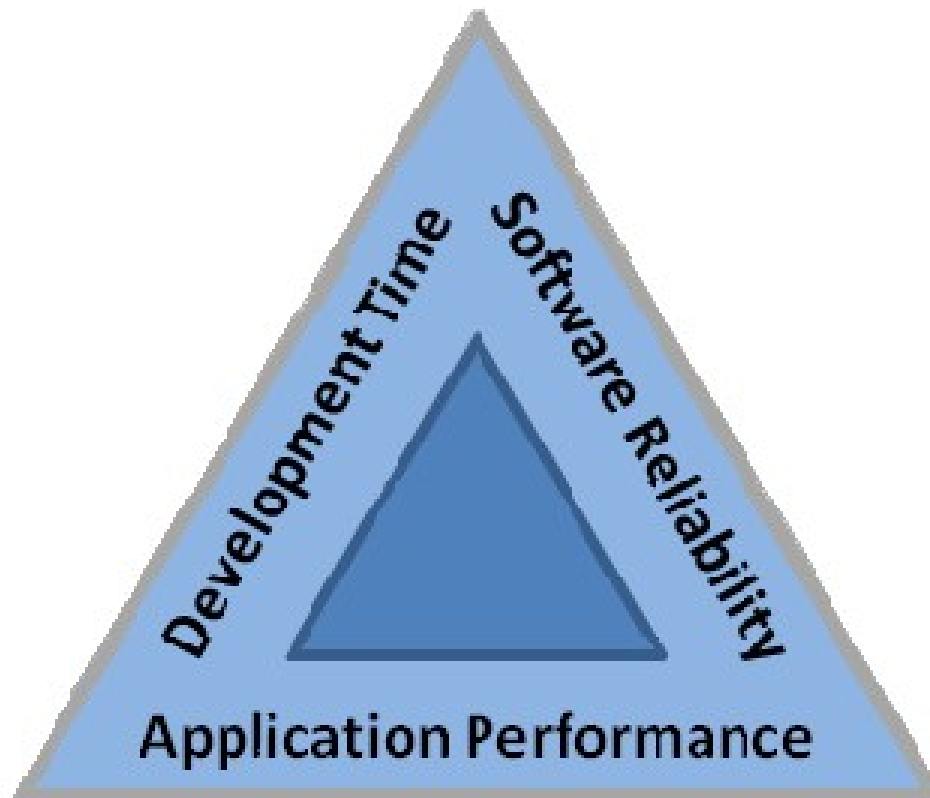
# The Status-Quo

- We moved from single core to multicore to manycore:
  - for technological reasons, as we saw last lecture.
- Free lunch is over for software folks
  - The software will not become faster with every new generation of processors
- Not enough experience in parallel programming
  - Parallel programs of old days were restricted to some elite applications -> very few programmers
  - Now we need parallel programs for many different applications

# How Did These Advances Happen?







The Multicore Software Triad

# Conclusions

- The hardware evolution, driven by Moore's law, was geared toward two things:
  - Exploiting parallelism
  - Dealing with memory (latency, capacity)