Digital System Design

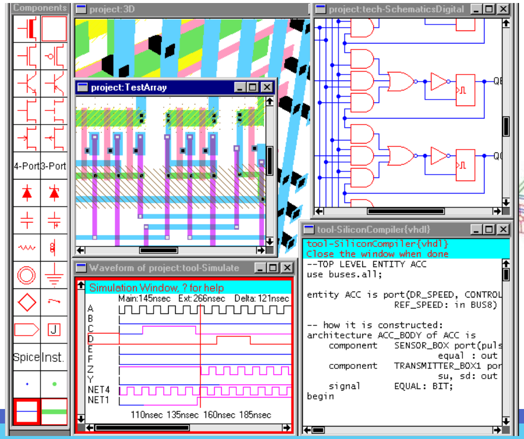
Hajar Falahati

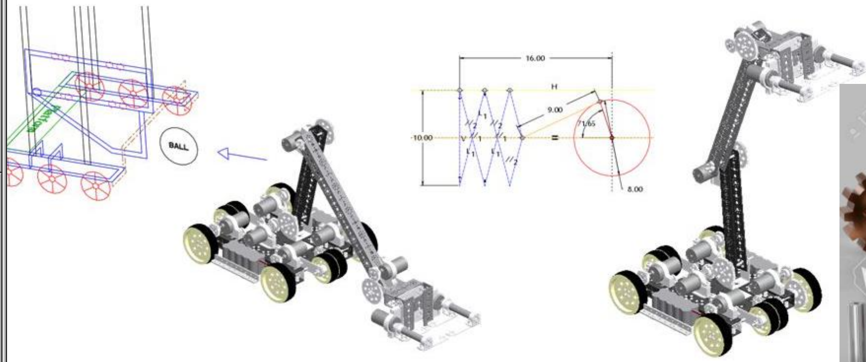
hfalahati@ipm.ir
hfalahati@ce.sharif.edu

DSD

- Digital System Design (DSD)

Computer Aided Design





Outline

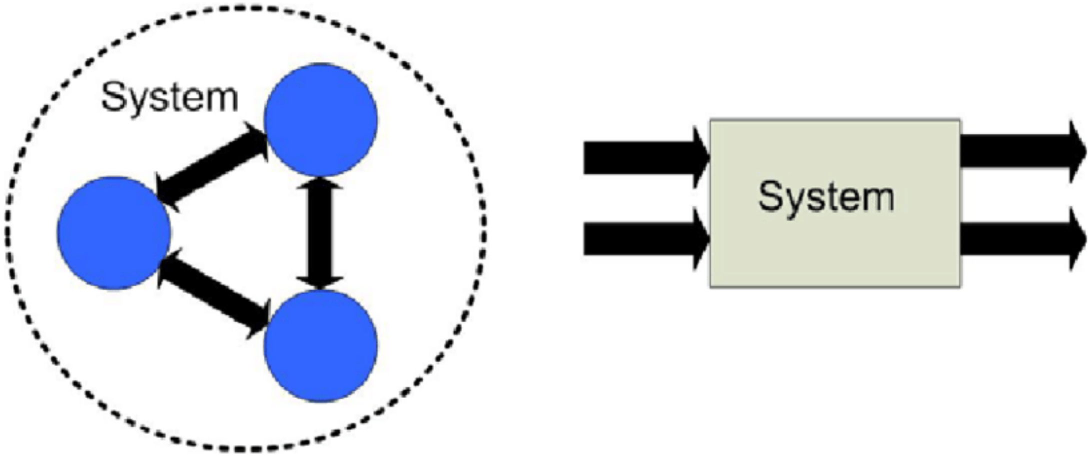
- Digital Computer Systems

Digital Computer Systems

System

- A **set** of **interacting components** that **acts** **as a whole**

- Performs the desired functions
  - Behavior



Computer

- **Brukes Goldsten, Von Neuman**
  - Preliminary discussion of the **logical design** of an **electronic computing instrument**
    - 1946

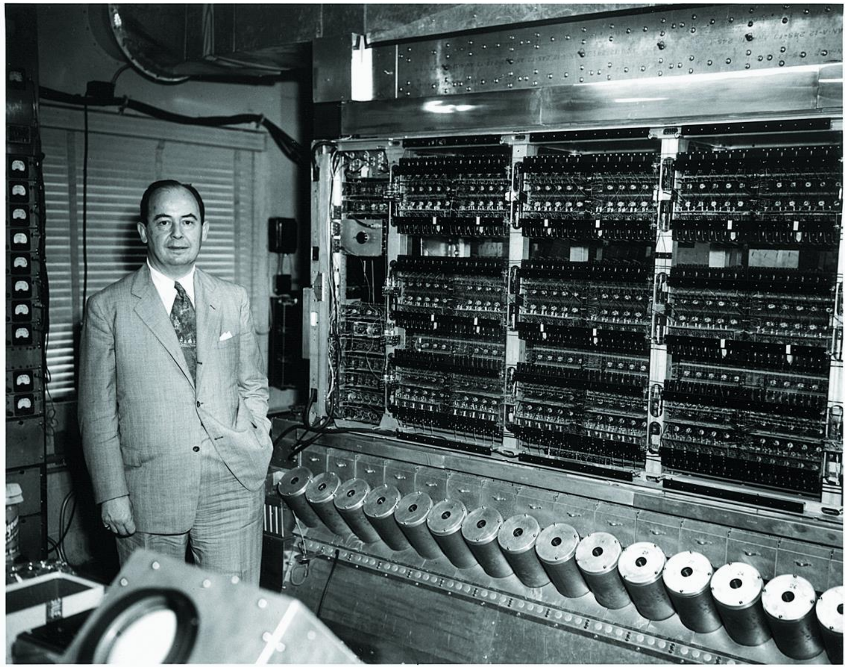
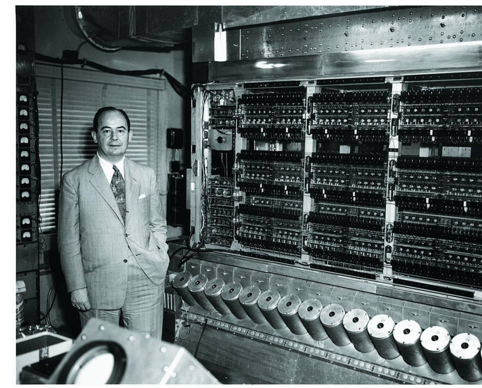


Image source: https://lbsitbytes2010.wordpress.com/2013/03/29/john-von-neumann-roll-no-15/

Components

- **Three key components**
  - Computation
  - Communication
  - Storage / Memory



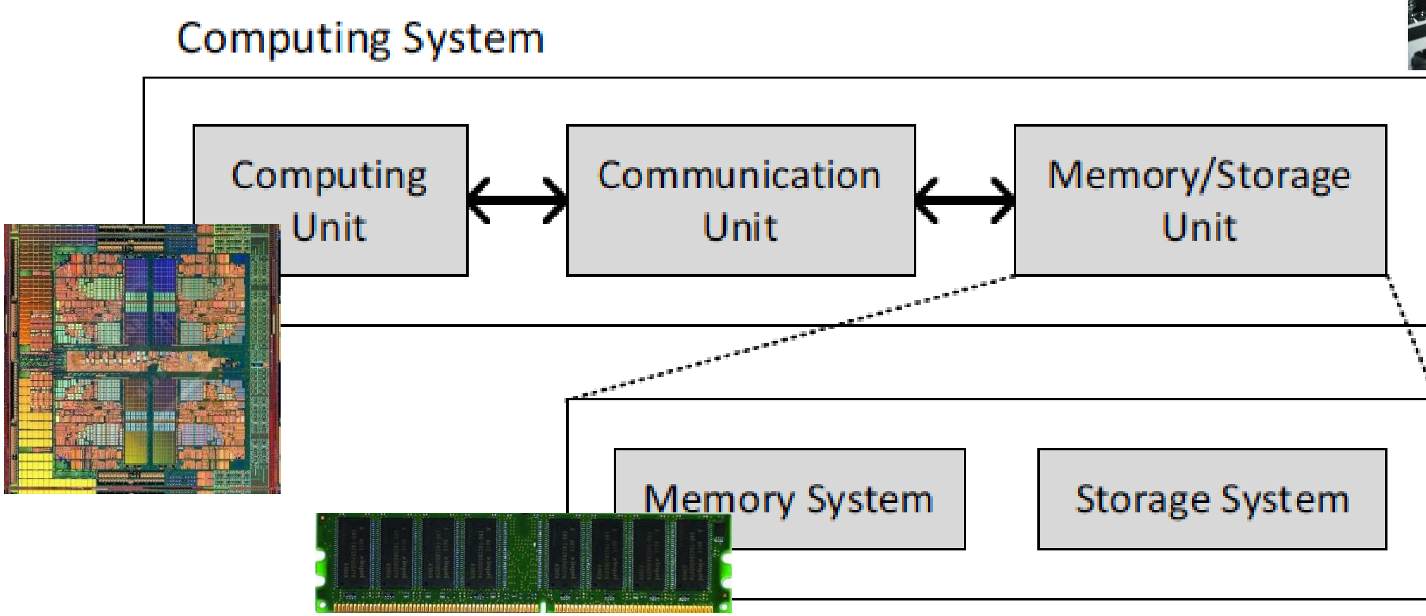
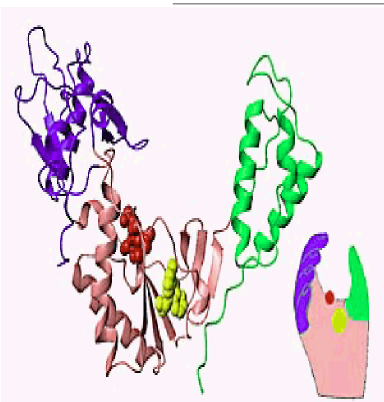
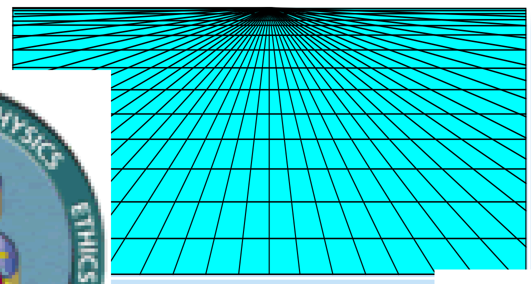


Image source: https://lbsitbytes2010.wordpress.com/2013/03/29/john-von-neumann-roll-no-15/

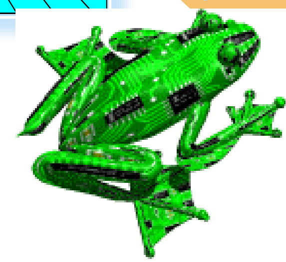
Computer Systems Are Every Where!



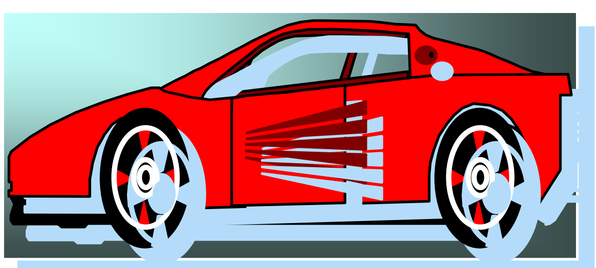

Life Sciences



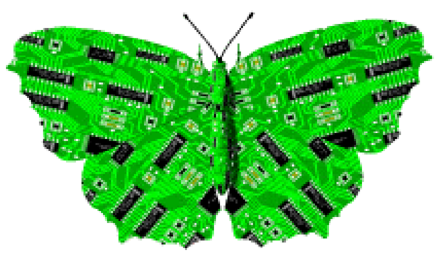
Aerospace



Internet & Ecommerce



CAD/CAM



Digital Biology

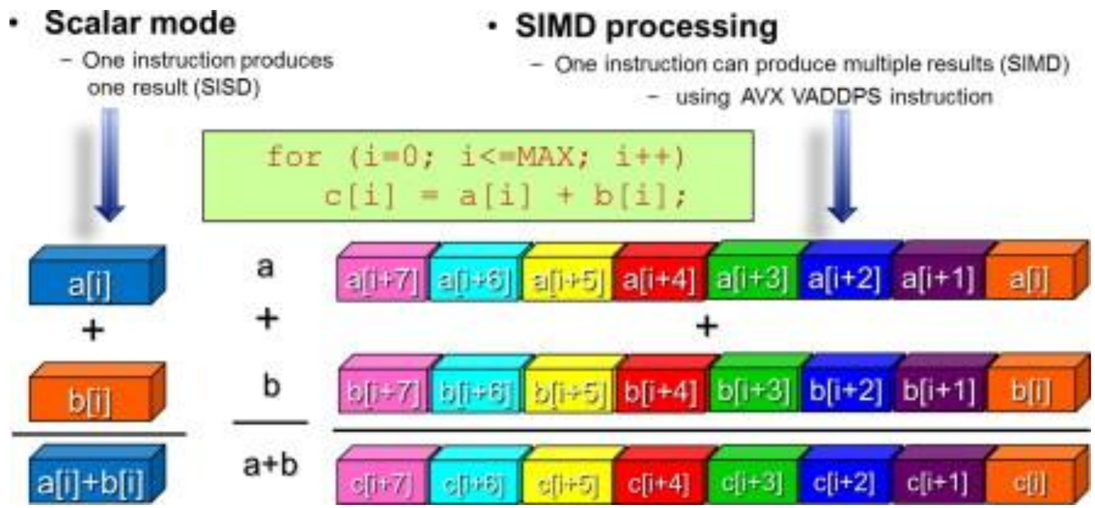
Military Applications

Computer System

- A computer combined with peripheral equipment and software
- Combination of hardware, software, user and data
  - Performs desired functions

Hardware Vs. Software

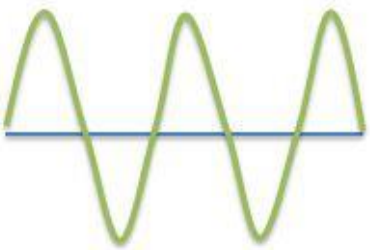
- Software
  - Flexibility
  - Ease of modification
  - Ease of upgrade

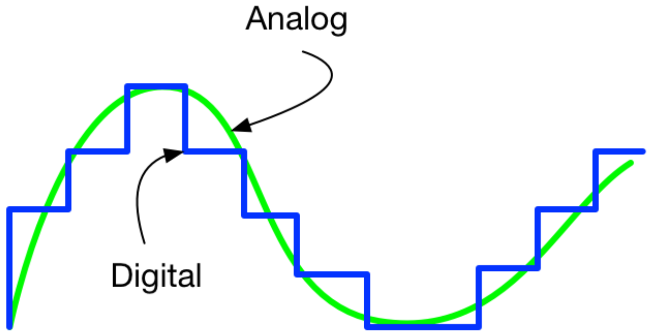


- Hardware
  - High speed
  - Low power consumption!

Digital Vs. Analog

- Analog
  - Time-varying signals
  - Take **any** value across a **continuous time** domains
  - Sensing and actuating environmental values



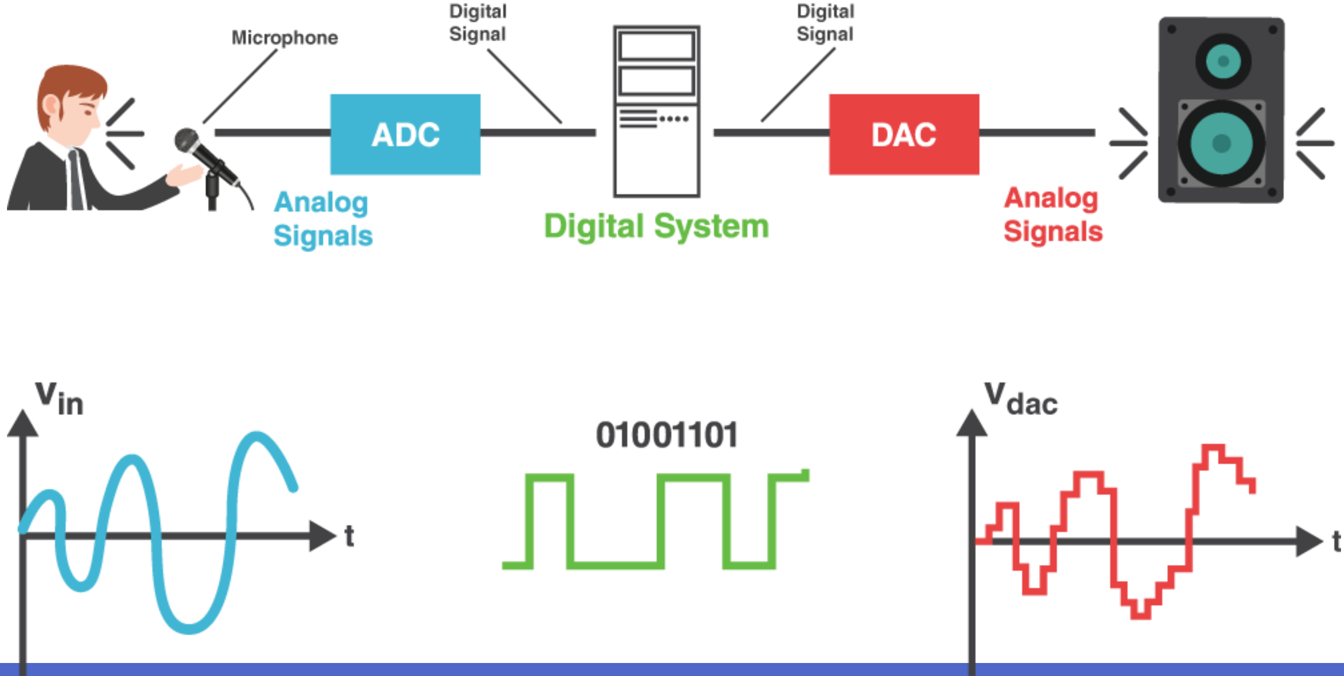


- Digital
  - **Finite** values in **discrete time** domains
  - Algorithmic control
  - Data processing

0100111101

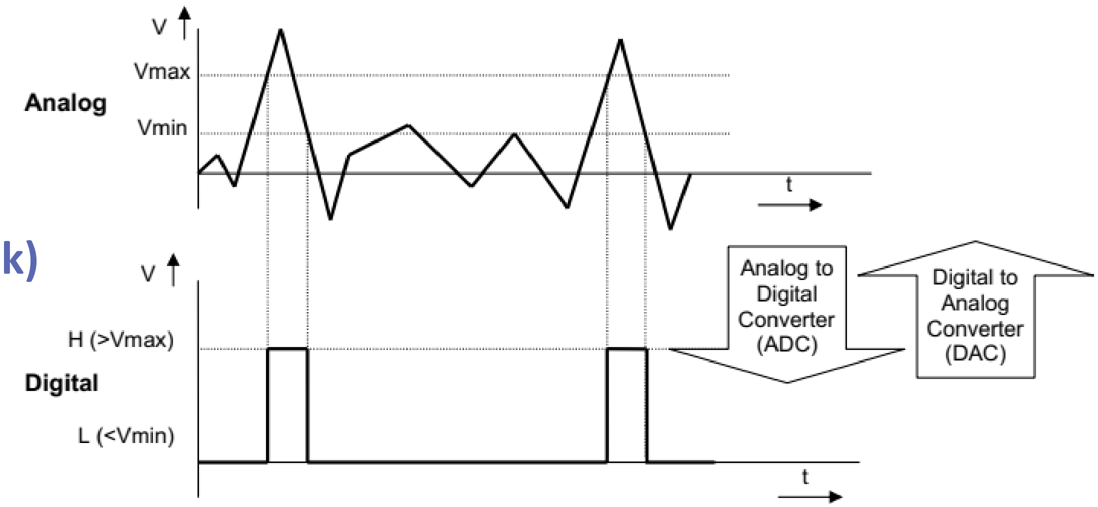
Digital System

- Takes a set of discrete information as inputs
- Takes discrete internal information as system state
- Generates a set of discrete information as outputs



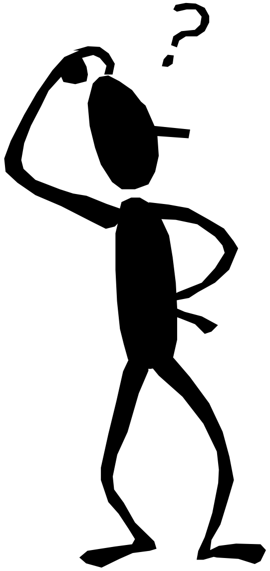
Digital Computer Systems

- Binary values
  - **Digits:** 0,1
  - **Words (symbol):** False (F), True (T)
  - **Words:** Low (L), High (H)
  - **Words:** On, Off
  - **Voltage (CPU)**
  - **Electrical charge (DRAM)**
  - **Magnetic Field Direction (Disk)**
  - **Surface Pits/Lights (CD)**



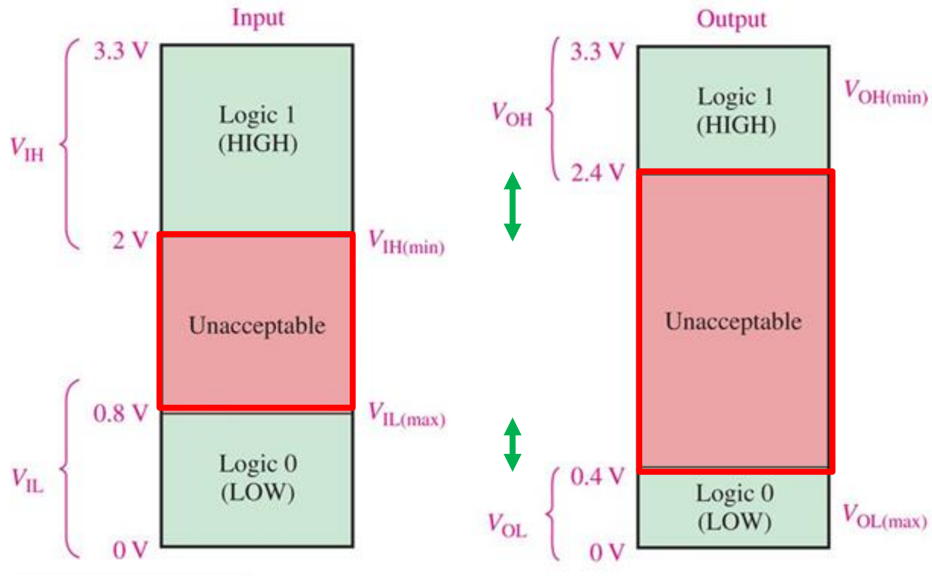
Example of analog and digital representations of human Heart Beat:

Why Digital?



Why Digital?

- High noise immunity
- Better reliability



Why Digital? (cont'd)

- Design and Implementation simplicity
  - No complex mathematics formula and details of physical processes
  - Modular design
  - Easier implementation

- Programmability
  - Easy to program

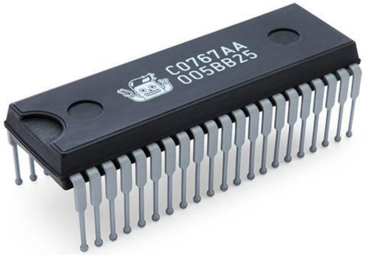
- More flexibility
  - Easy to program and modify

Digital Computer System: Trend

- Non-Electronic Computing Machines
  - Abacus

- Electro-Mechanical Computers

- Electronic Computers



Non-Electronic Computing Machines

- Punch machine
  - Punch cards
  - Presenting digital information by the presence or absence of holes.

Non-Electronic Computing Machines

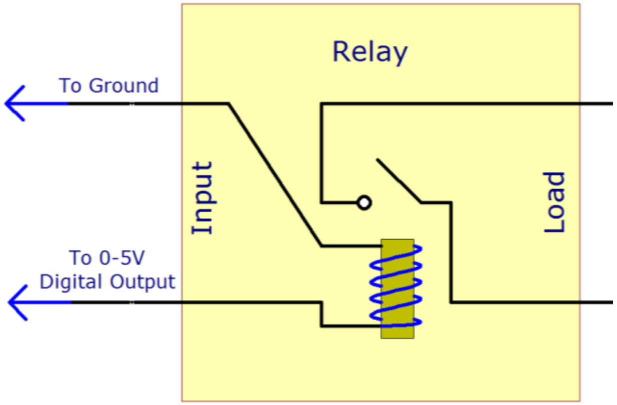
- Difference engine
  - Automatic mechanical calculator
  - Polynomial functions

Non-Electronic Computing Machines

- Analytical engine
  - Programmable
  - Punch cards

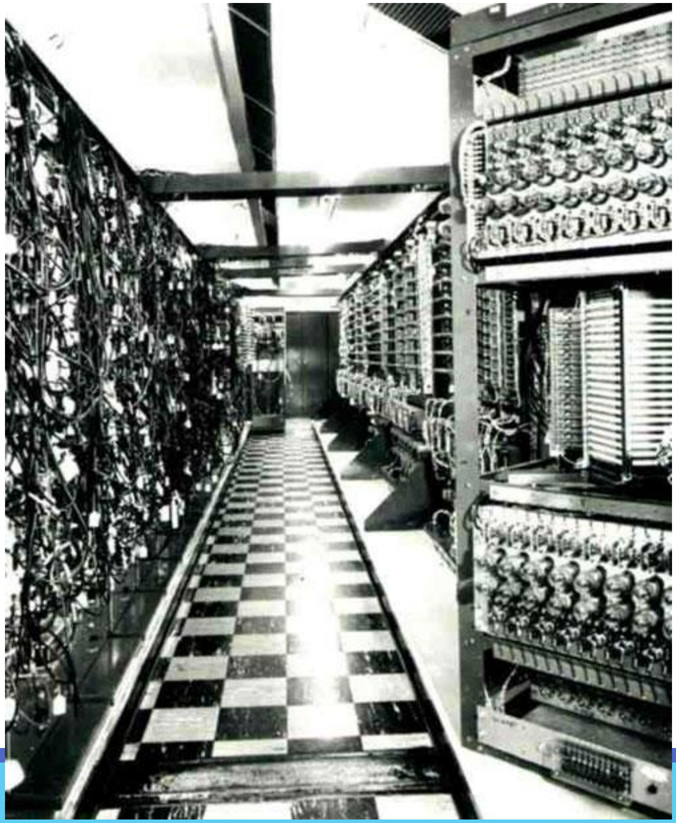
Electro-Mechanical Computers

- Electric switches drove mechanical relays to perform the calculation
- Low operating speed



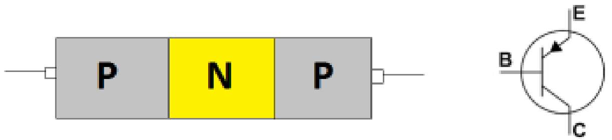
Electronic Computers: 1st Generation

Generation	year	Technology
1st generation	1945-1955	Vacuum tubes

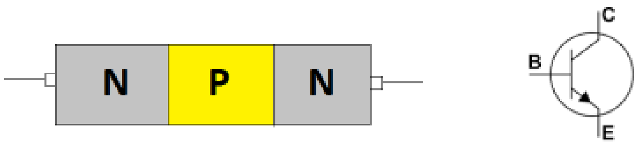


Electronic Computers: 2nd Generation

Generation	year	Technology
1st generation	1945-1955	Vacuum tubes
2nd generation	1955-1965	BJT transistors



PNP Transistor

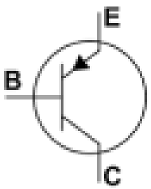


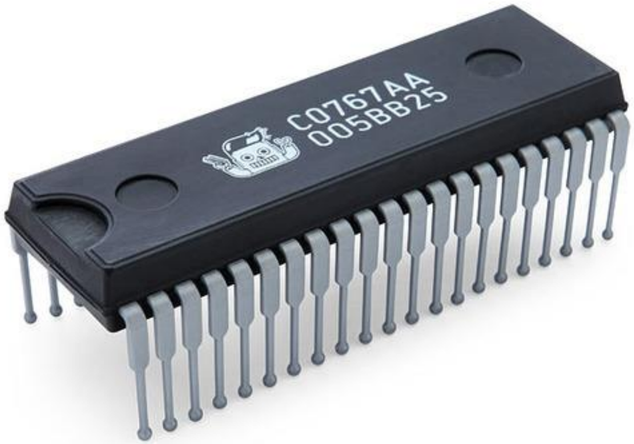
NPN Transistor

Typical Transistor

Electronic Computers: 3rd Generation

Generation	year	Technology
1st generation	1945-1955	Vacuum tubes
2nd generation	1955-1965	BJT transistors
3rd generation	1965-1974	Integrated Circuits

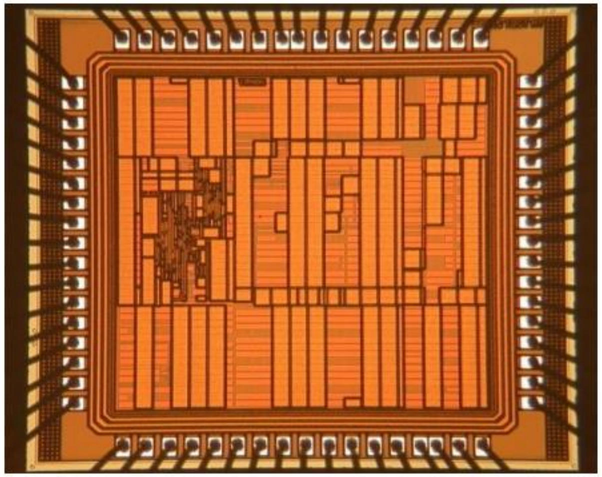




Integrated Circuits (ICs)

- A collection of gates fabricated on a single silicon chip
  - Many Applications
  - Low power
  - Small area
  - High speed

Discrete Circuits



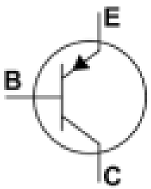
Integrated Circuits

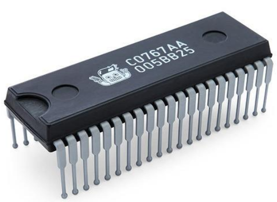
Electronic Computers: Integrated Circuits (ICs)

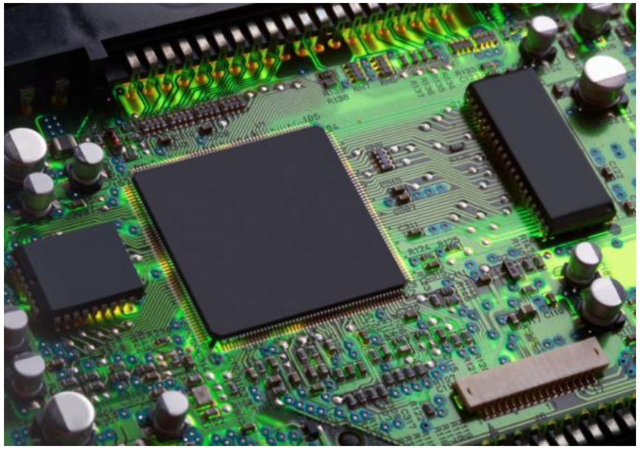
- A collection of gates fabricated on a single silicon chip
  - ***Small Scale Integration (SSI)***
    - A small number of gates
  - ***Medium Scale Integration (MSI)***
    - > 100 gates
    - Decoder, register, counter
  - ***Large Scale Integration (LSI)***
    - > 1000 gates
    - Small memories, PLDs
  - ***Very Large Scale Integration (VLSI)***
    - > 1,000,000 transistors
    - Microprocessors, memories

Electronic Computers: 5th Generation

Generation	year	Technology
1st generation	1945-1955	Vacuum tubes
2nd generation	1955-1965	BJT transistors
3rd generation	1965-1974	Integrated Circuits
4th generation	1974-1989	VLSI
5th generation	1990-present	ULSI

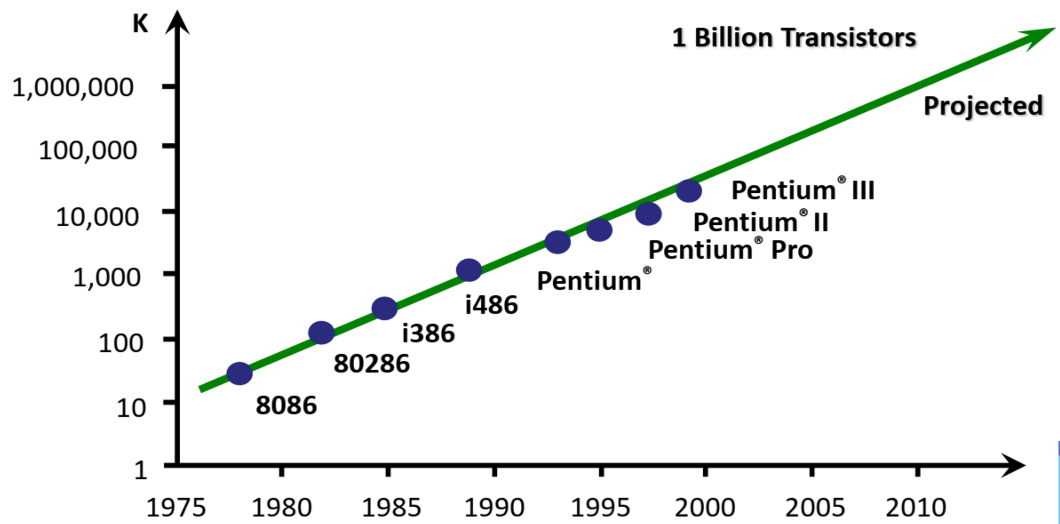






Moore’s Law

- **2x** transistors/chip every **1.5** years
  - First transistor 1947 (Nobel prize)
  - First Silicon transistor 1954
  - First CMOS gate 1963
  - First commercial CMOS chips 1974
    - Digital watches
  - Nowadays, largest chips have **>10’000’000’000** transistors about as many people on the planet



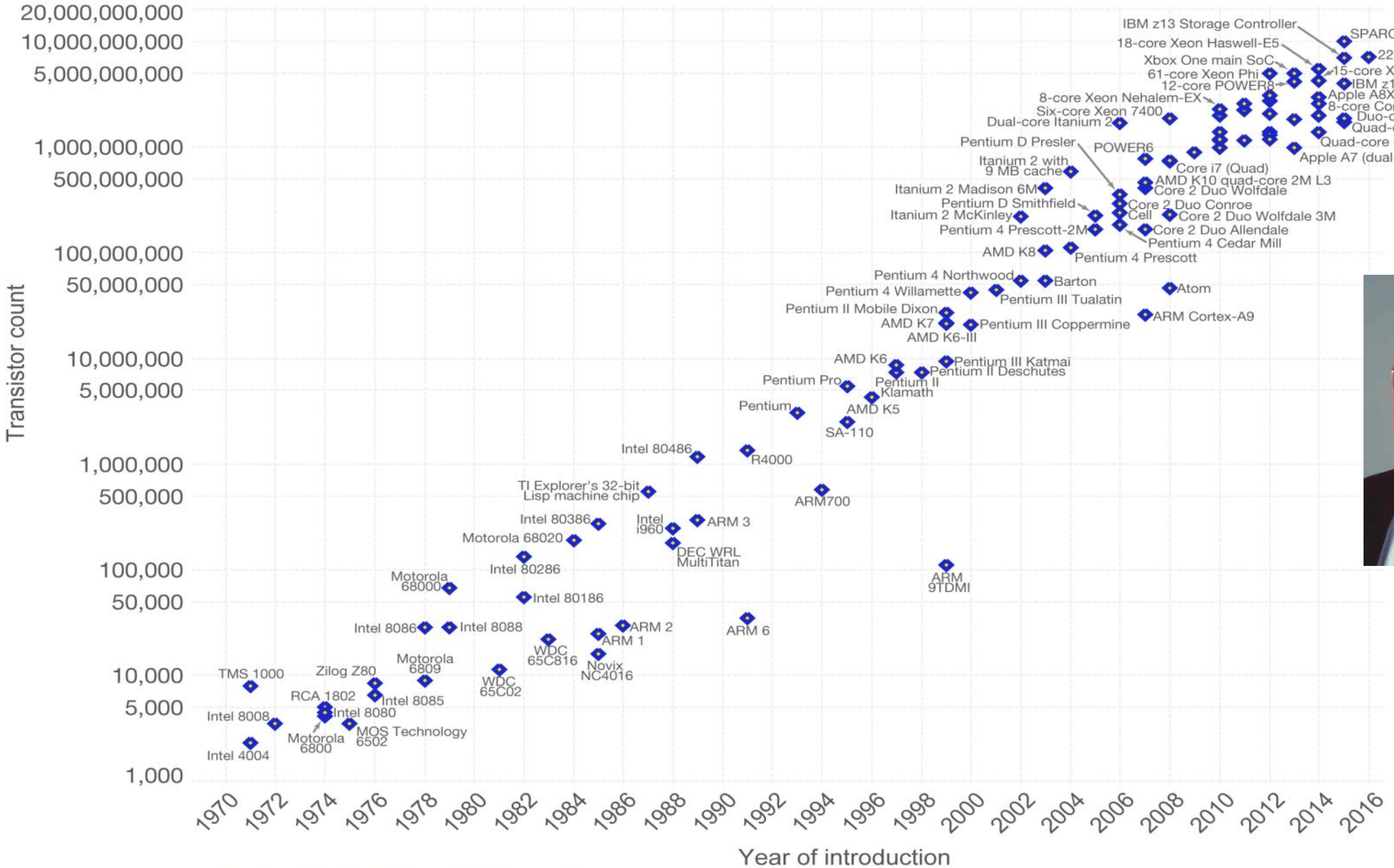
Why Digital Design?

Growing Complexity

Moore's Law – The number of transistors on integrated circuit chips (1971-2016)

Our World in Data

Moore's law describes the empirical regularity that the number of transistors on integrated circuits doubles approximately every two years. This advancement is important as other aspects of technological progress – such as processing speed or the price of electronic products – are strongly linked to Moore's law.



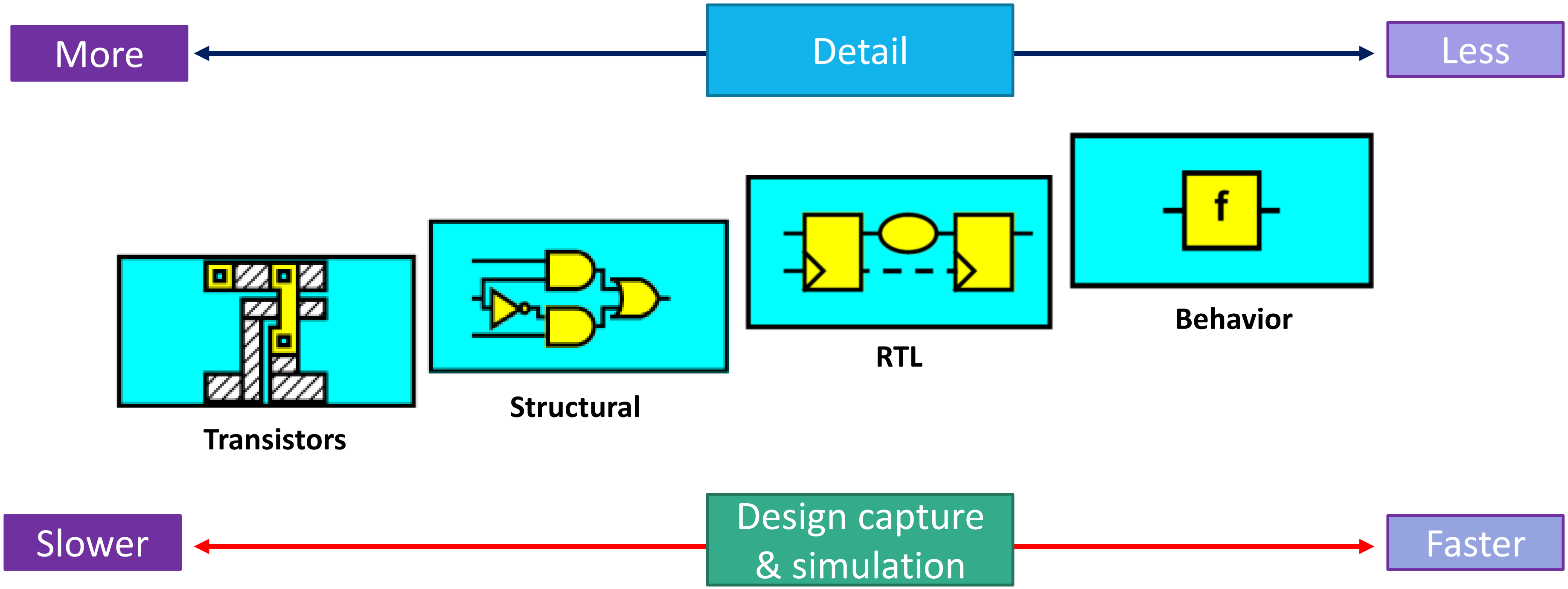
Data source: Wikipedia (https://en.wikipedia.org/wiki/Transistor_count)

The data visualization is available at OurWorldinData.org. There you find more visualizations and research on this topic.

https://www.intel.com/pressroom/kits/events/moores_law_40th/

Simplifying Modeling and Design

- Facilitate **modeling** the digital systems



Emerging Applications

- New requirements
- User experiences
- Rapid change!

Application-Specific Implementation

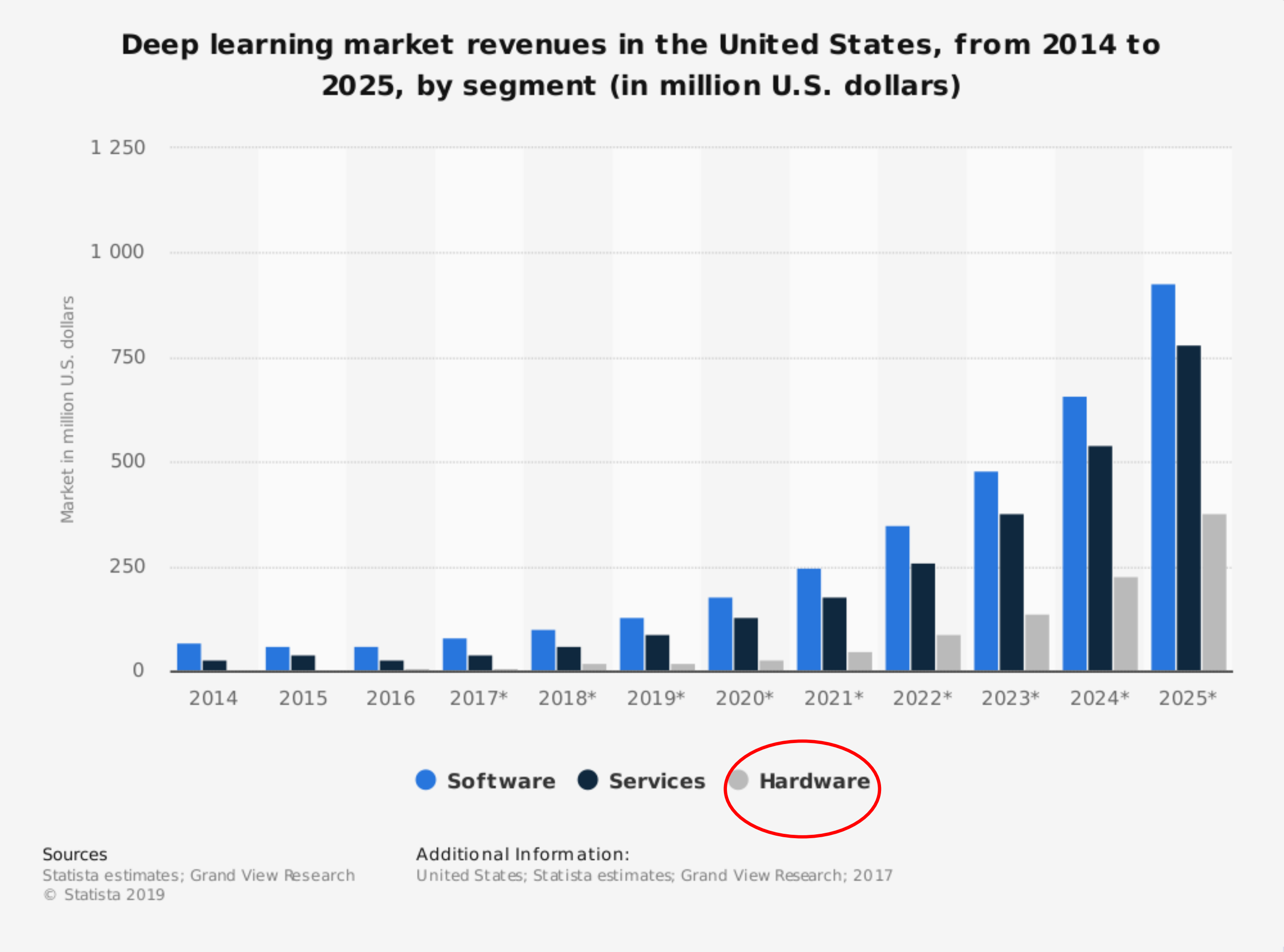
- **The era of domain-specific hardware-software codesign**
  - https://iscaconf.org/isca2018/turing_lecture.html

A New Golden Age for Computer Architecture: Domain-Specific Hardware/Software Co-Design, Enhanced Security, Open Instruction Sets, and Agile Chip Development

John L. Hennessy and David A. Patterson

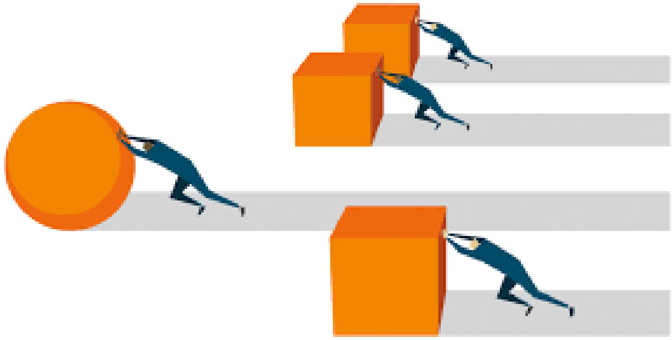
ISCA2018

Deep Learning Market Revenue



Technology Optimizations

- Efficiency parameters
  - Cost
  - Performance
  - Power
  - ....

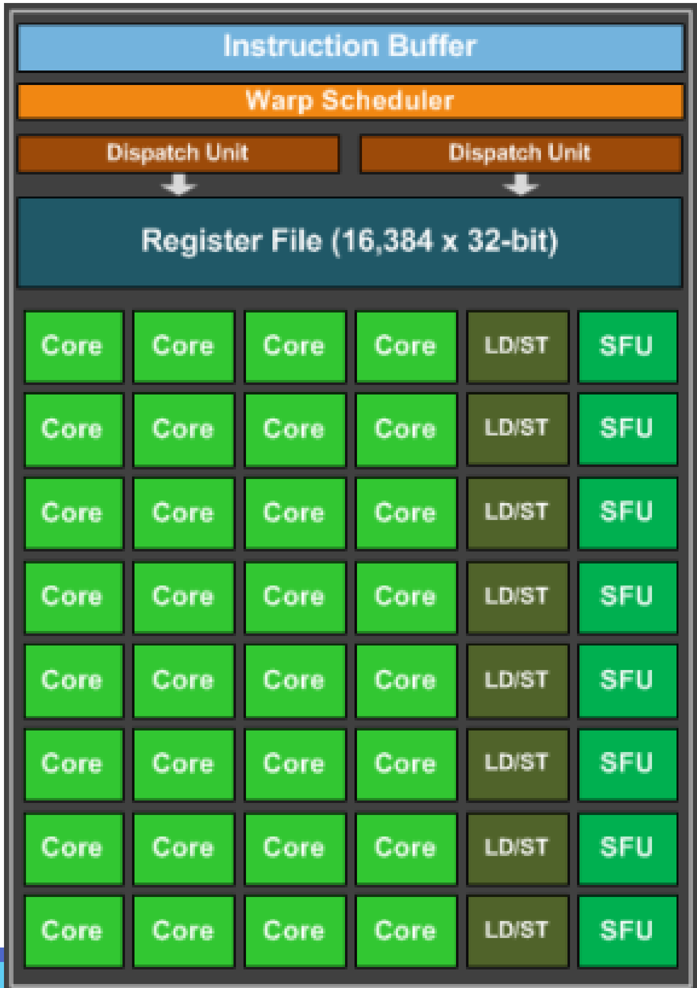


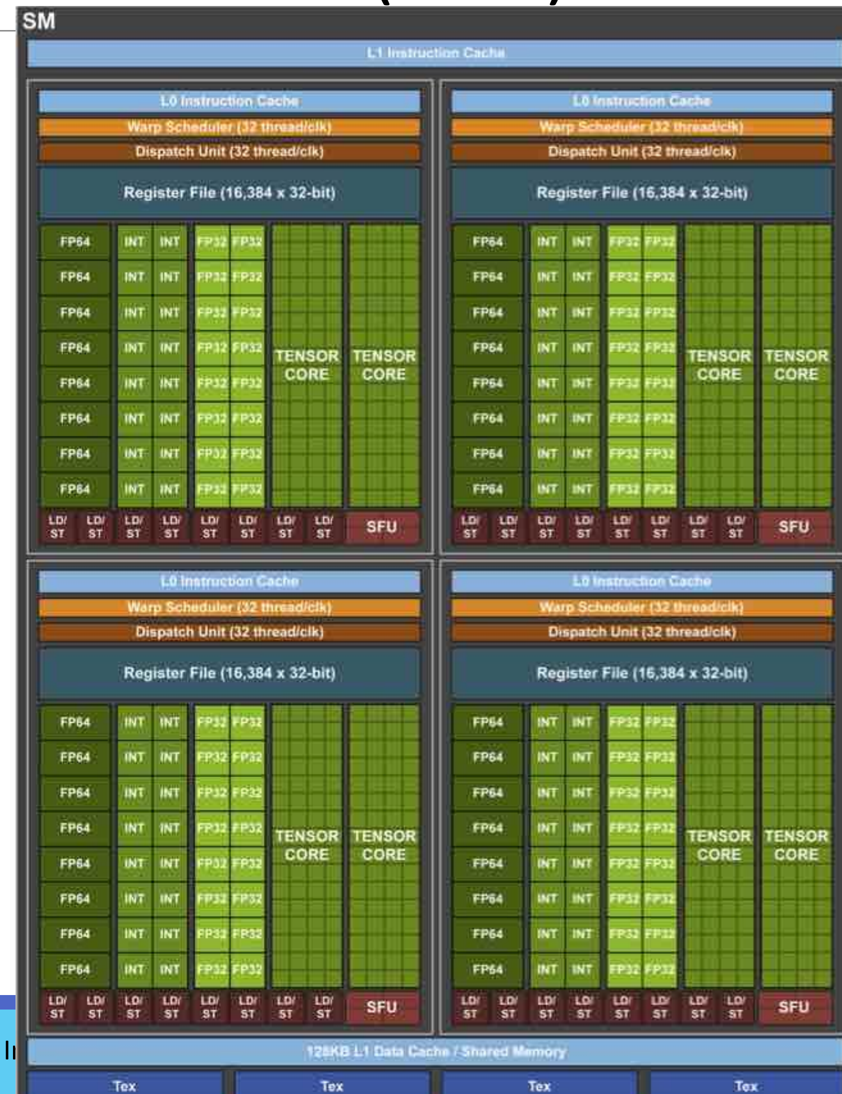
- Need to be **aware of architecture** to write the optimized programs

Let’s consider GPUs as an example

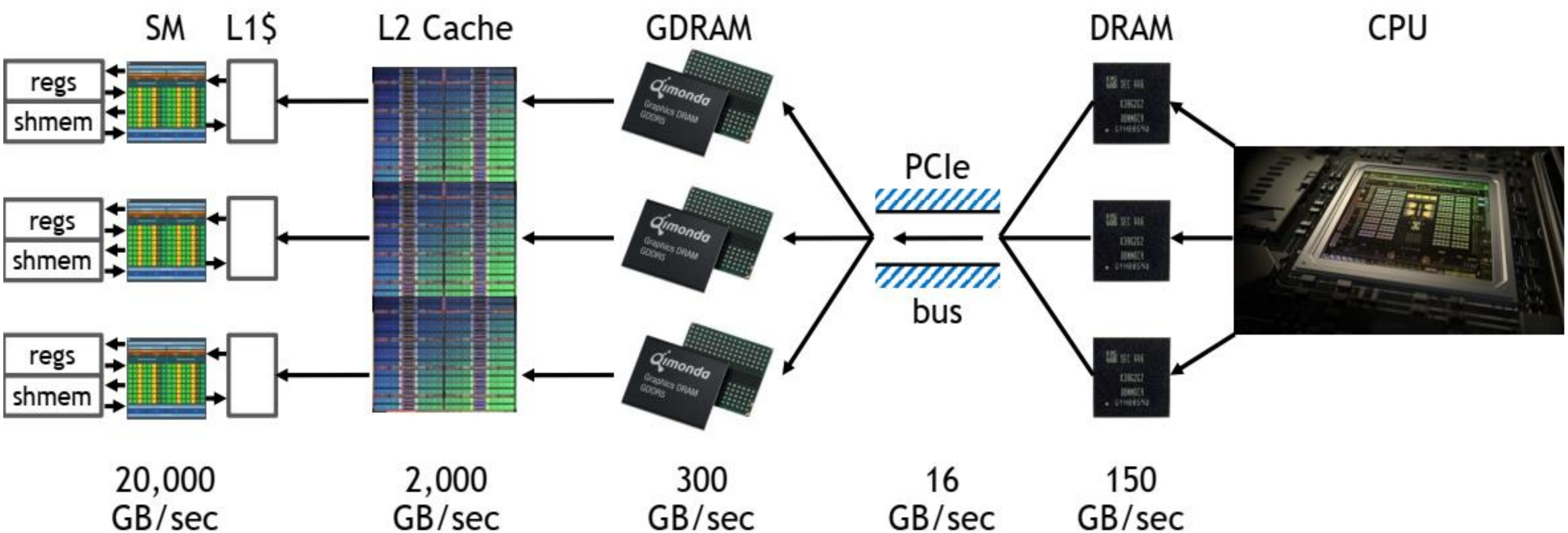
- Graphic Processing Unit (GPU)
  - NVIDIA V100

GPU:
Streaming Multiprocessor (SM)

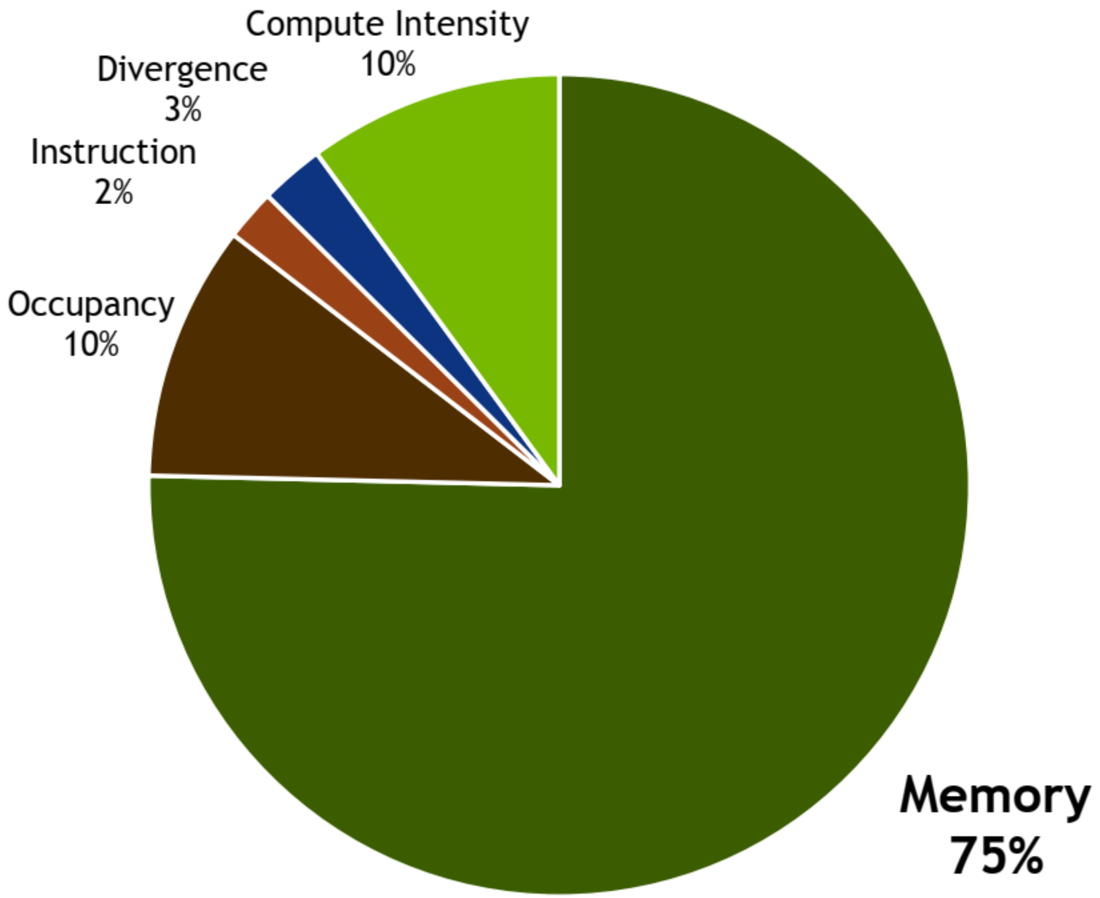




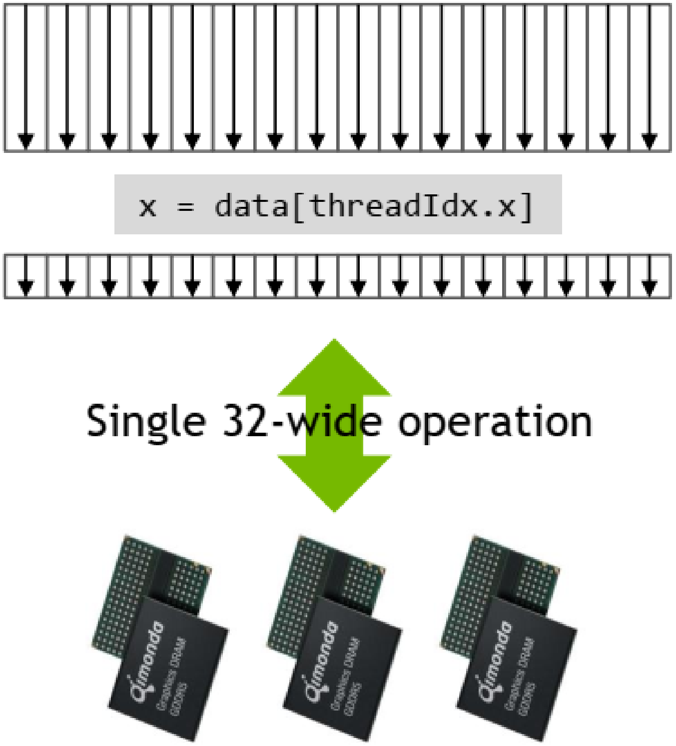
GPU: Memory

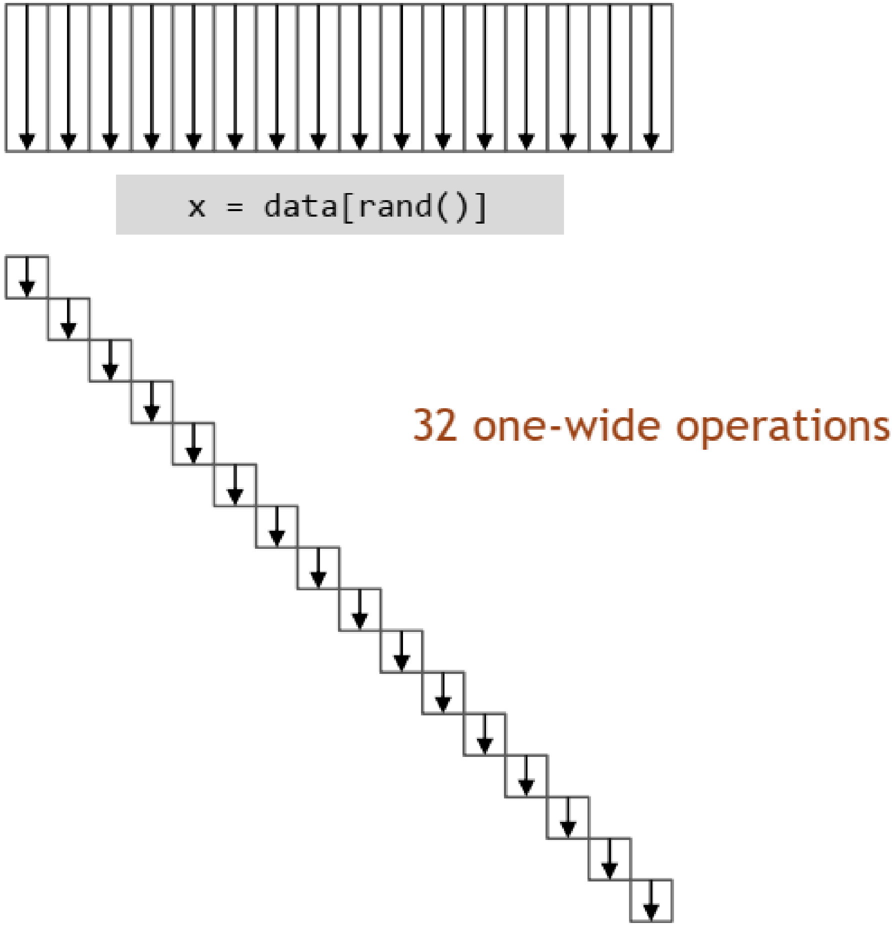

SM
L1$
L2 Cache
GDRAM
DRAM
CPU
regs
shmem
regs
shmem
regs
shmem
PCIe
bus
20,000 GB/sec
2,000 GB/sec
300 GB/sec
16 GB/sec
150 GB/sec

GPU: Performance Constraints



GPU: Coalesced Memory





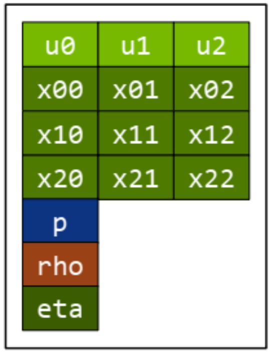
GPU: Sample Code

Array-of-Structures

```
#define NPTS 1024 * 1024

struct Coefficients_AOS {
      double u[3];
      double x[3][3];
      double p;
      double rho;
      double eta;
};

Coefficients_AOS gridData[NPTS];
```

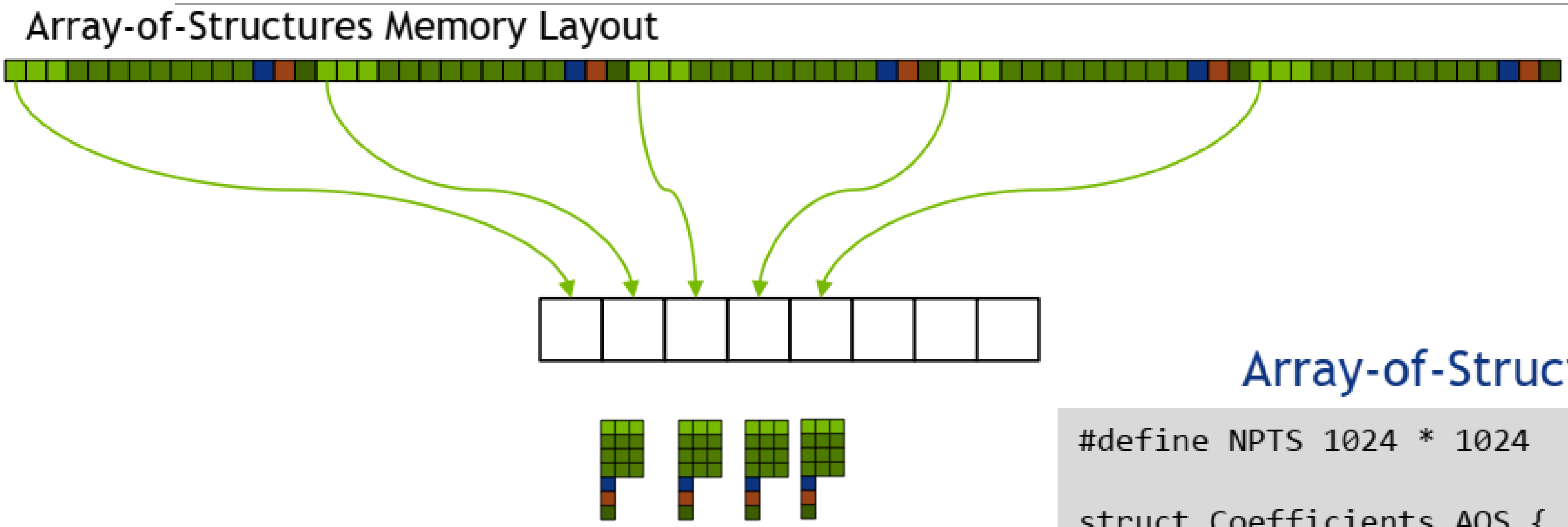
Structure-of-Arrays

```
#define NPTS 1024 *1024

struct Coefficients_SOA {
      double u[3][NPTS];
      double x[3][3][NPTS];
      double p[NPTS];
      double rho[NPTS];
      double eta[NPTS];
};

Coefficients_SOA gridData;
```



Conceptual Layout

AOS

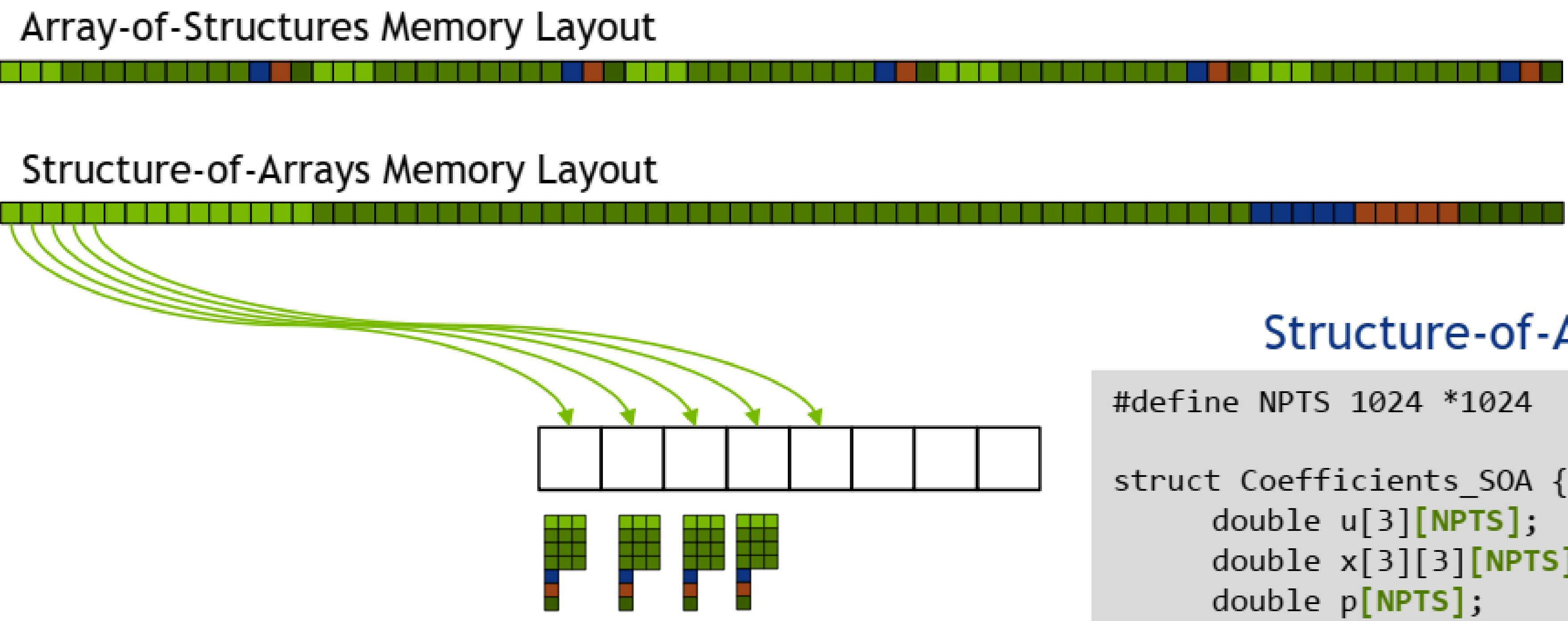



```
double u0 = gridData[threadIdx.x].u[0];
```

Array-of-Structures

```
#define NPTS 1024 * 1024

struct Coefficients_AOS {
      double u[3];
      double x[3][3];
      double p;
      double rho;
      double eta;
};

Coefficients_AOS gridData[NPTS];
```

SOA

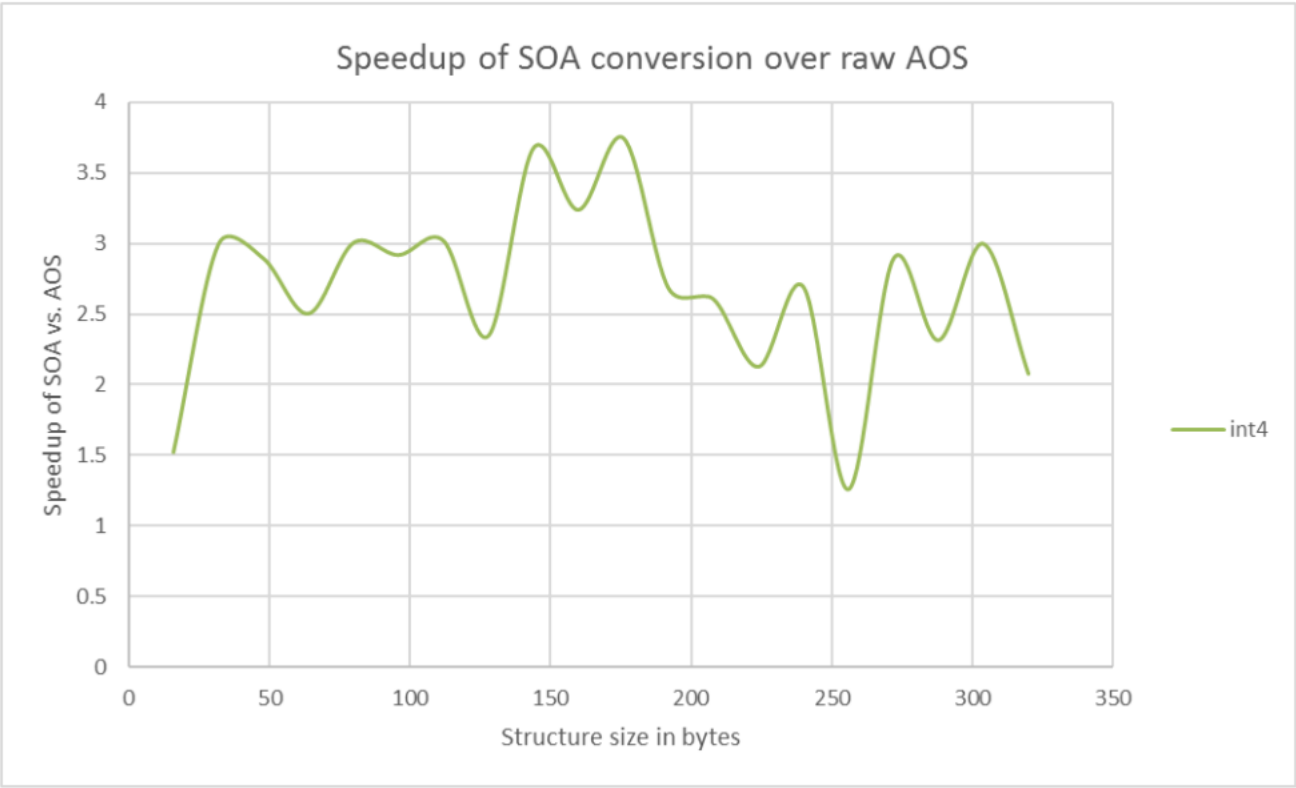



```
double u0 = gridData.u[0][threadIdx.x];
```

Structure-of-Arrays

```
#define NPTS 1024 *1024

struct Coefficients_SOA {
      double u[3][NPTS];
      double x[3][3][NPTS];
      double p[NPTS];
      double rho[NPTS];
      double eta[NPTS];
};

Coefficients_SOA gridData;
```

AOS Vs. SOA



“CUDA Optimization Tips, Tricks and Techniques,” by Stephen Jones, GTC17.

Array-of-Structures

```
#define NPTS 1024 * 1024

struct Coefficients_AOS {
      double u[3];
      double x[3][3];
      double p;
      double rho;
      double eta;
};

Coefficients_AOS gridData[NPTS];
```

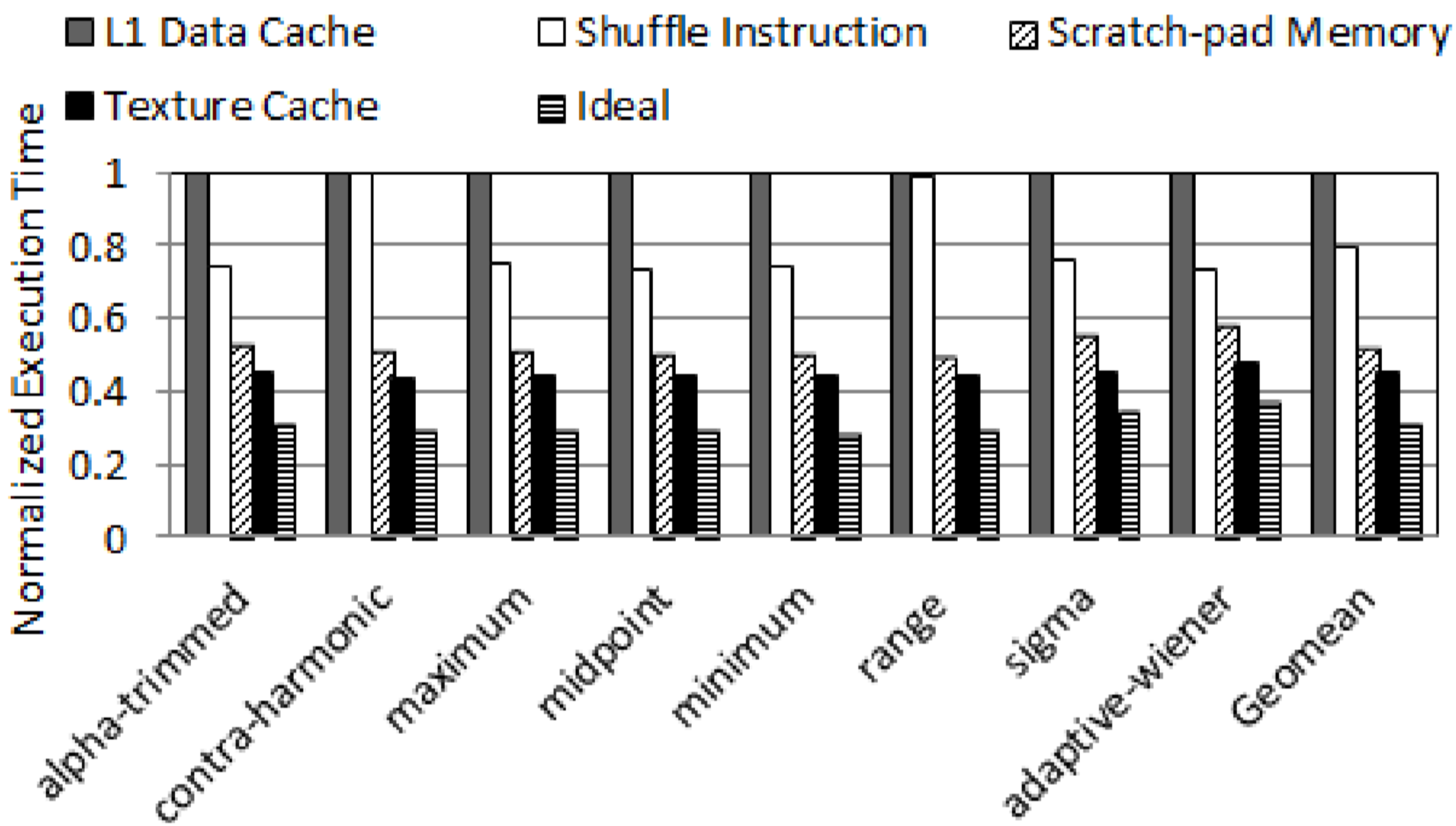
Single-thread code prefers arrays of structures, for cache efficiency

Structure-of-Arrays

```
#define NPTS 1024 *1024

struct Coefficients_SOA {
      double u[3][NPTS];
      double x[3][3][NPTS];
      double p[NPTS];
      double rho[NPTS];
      double eta[NPTS];
};

Coefficients_SOA gridData;
```

SIMT code prefers structures of arrays, for execution & memory efficiency

Execution Time Over Different Techniques



N. Nematollahi, M. Sadrosadati, H. Falahati, M. Barkhordar, H. Sarbazi-Azad, **Neda: Supporting direct inter-core neighbor data exchange in GPUs**. *CAL,* 2018.

Thank You