CMP 338: *Second Class*

Converting between basis
HW 1 solution
The Basic Processor (data path and control)
Eight recurring concepts in computer architecture
What is an ***interface***? (ex. ISA and API)
What happens to your program?
Moore's law
Integrate circuit manufacture and cost
Memory hierarchy

For next class: HW 2; *read* 1.6–8, 1.10, *reread* 1.6

Integers in Different Bases

Base 10 (***decimal*** — ten fingers)

$$4129_{10} = 4*10^3 + 1*10^2 + 2*10^1 + 9*10^0$$

Base 2 (***binary*** — two fingers)

$$1011_{2} = 1*2^3 + 0*2^2 + 1*2^1 + 1*2^0$$

Base 16 (***hexadecimal*** — sixteen fingers)

$$A3F8_{16} = 10*16^3 + 3*16^2 + 15*16^1 + 8*16^0$$

Conversion between base 2 and base 16 is easy!

0x A3F8 = 0b 1010 0011 1111 0100

Four Hundred and Thirty Seven

110110101_2	437_{10}	$1B5_{16}$
$1 \cdot 2^8 = 256$	$4 \cdot 10^2 = 100$	$1 \cdot 16^2 = 256$
$+ 1 \cdot 2^7 = 128$	$+ 3 \cdot 10^1 = 30$	$+ 11 \cdot 16^1 = 128$
$+ 0 \cdot 2^6 = 0$	$+ 7 \cdot 10^0 = 7$	$+ 5 \cdot 16^0 = 5$
$+ 1 \cdot 2^5 = 32$		
$+ 1 \cdot 2^4 = 16$		
$+ 0 \cdot 2^3 = 0$		
$+ 1 \cdot 2^2 = 4$		
$+ 0 \cdot 2^1 = 0$		
$+ 1 \cdot 2^0 = 1$		

First 16 Non Negative Integers

Decimal		Binary		Hexadecimal	
0	8	0000	1000	0	8
1	9	0001	1001	1	9
2	10	0010	1010	2	A
3	11	0011	1011	3	B
4	12	0100	1100	4	C
5	13	0101	1101	5	D
6	14	0110	1110	6	E
7	15	0111	1111	7	F

Powers of 2

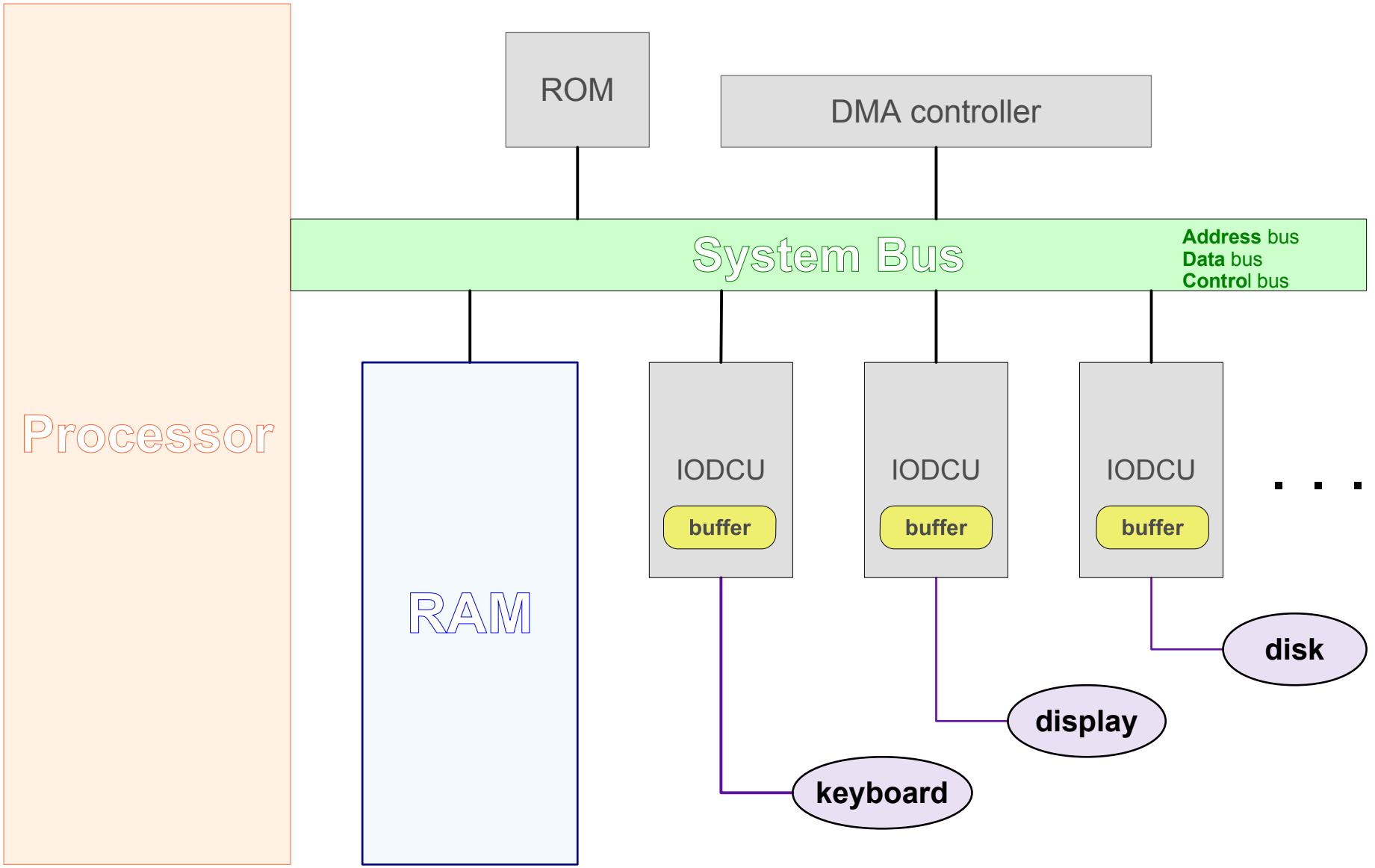
2^0	1	2^8	256
2^1	2	2^9	512
2^2	4	2^{10}	1024
2^3	8	2^{11}	2048
2^4	16	2^{12}	4096
2^5	32	2^{13}	8192
2^6	64	2^{14}	16384
2^7	128	2^{15}	32968

HW 1: Basic Computer Model

In two or three sentences of your own words define, describe, or discuss the following components of the basic computer model:

1. The System Bus
2. ROM Memory
3. RAM Memory
4. IODCU
5. IODCU buffer
6. The Processor
7. The DMA Controller

Basic Computer Model



HW 1: Computer Components

The System Bus

Connects the different components of the computer. (Logically, three separate buses for addresses, data, and control.)

ROM Memory

Read Only Memory — non-volatile memory provided by the computer manufacturer. Contains instructions and data that allows the computer to start up when power is turned on.

HW 1: Computer Components

RAM Memory

***Random Access Memory — Main Memory*, contains both data and instructions for running program(s).** Comprises a sequence of individually addressable bytes of data that can be read or written by the processor. (A *byte* consists of eight bits. Each *bit* can be either 0 or 1.)

Connects to the system bus through a ***Memory Address Register*** and a ***Memory Data Register***

HW 1: Computer Components

IODCU

Input/Output Device Control Unit — Device Controller*, an *interface* between a computer system and its input/output device. Translated information coming from the device into a form the system can use and *vice versa*. *Interrupts the processor when a task is finished.

IODCU Buffer

A chunk of RAM memory that provides a way station for data on its way from the device to the system and *vice versa*. Compensates for the speed mismatch between the system (extremely fast) and the device (relatively slow).

HW 1: Computer Components

The Processor

Fetches instructions from main memory and executes them. User instruction execution may entail loading data from memory, performing operations (addition, etc.) on data, or storing data.

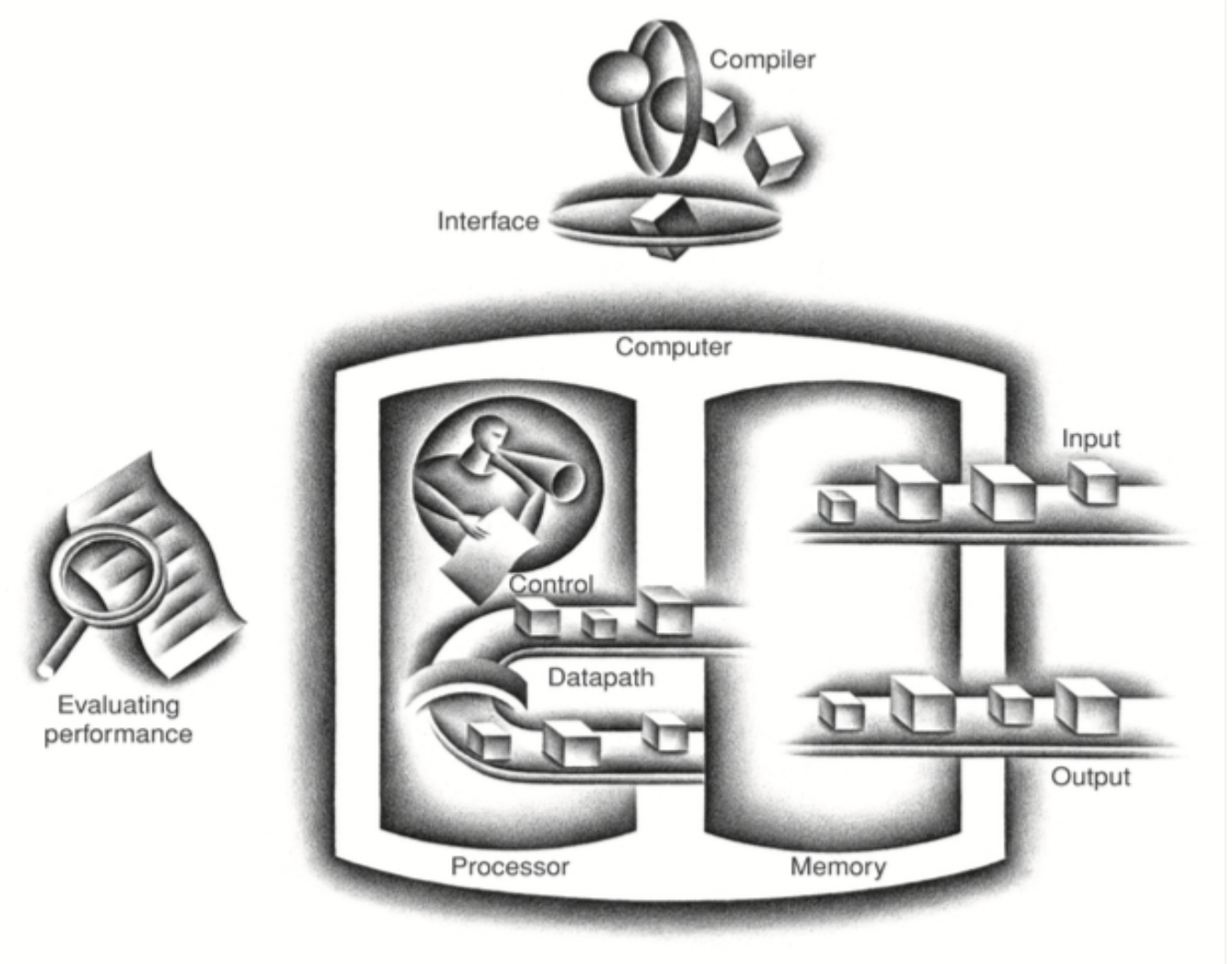
Privileged instructions, executed only for the computer's operating system (often in response to interrupts from device controllers), choreograph the flow of data among the components of the computer over the system bus.

HW 1: Computer Components

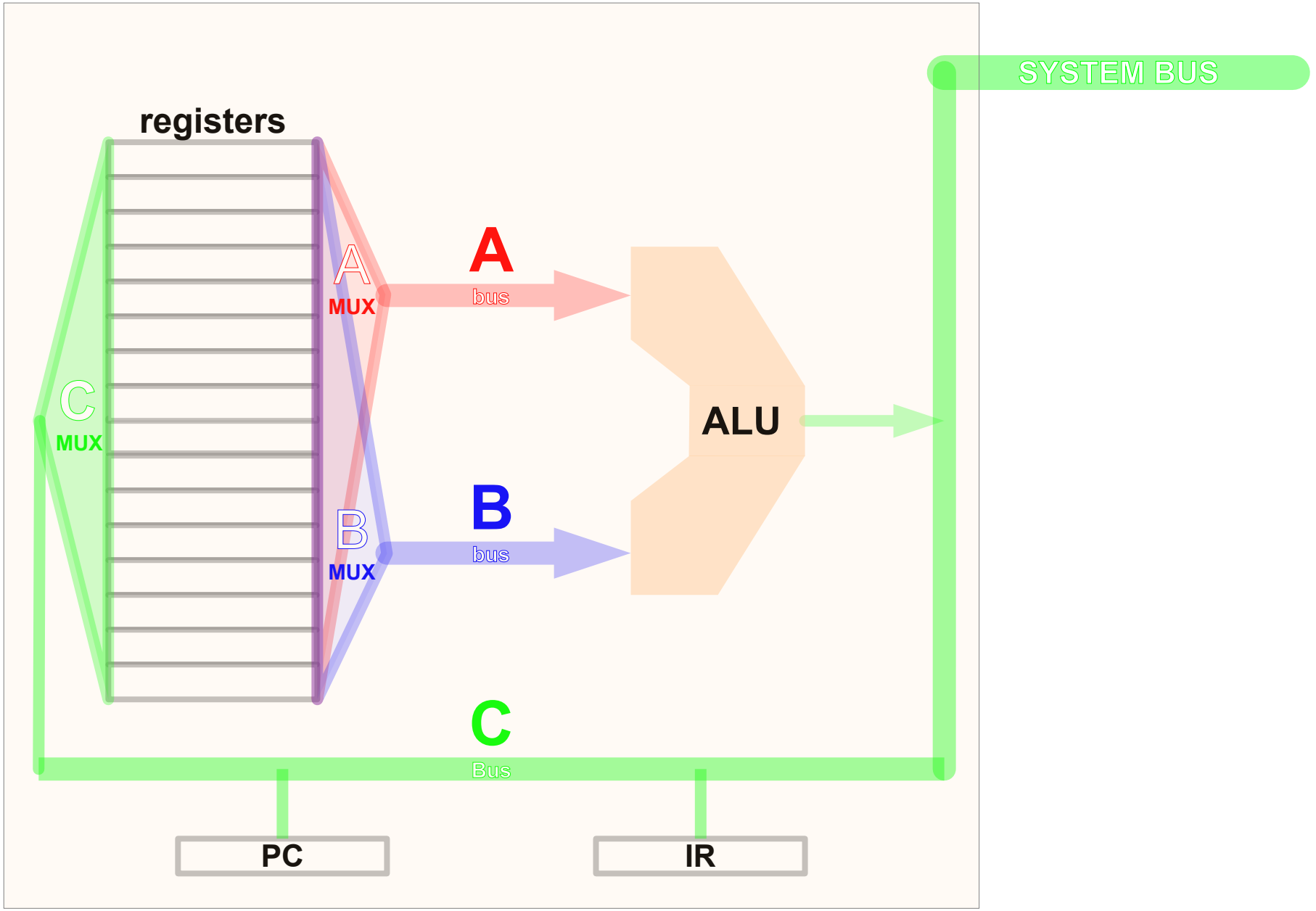
DMA controller

Direct Memory Access **controller — takes over from the processor the task of orchestrating the movement of large chunks of data between main memory and I/O device controller units.**

Organization of a Computer



Basic Processor Model



HW 2: Basic Processor Model

In two or three sentences of your own words define, describe, or discuss the following components of the basic processor model:

1. Register
2. Register File
3. A (B) multiplexer (MUX)
4. A (B) Bus
5. ALU — *Arithmetic / Logical Unit*
6. C Bus
7. Program Counter (PC)
8. Instruction Register (IR)

8 Great Computer Architecture Ideas

Design for ***Moore's Law***

Use ***abstraction*** to simplify design

Make the ***common case fast***

Performance *via* ***parallelism***

Performance *via* ***pipelining***

Performance *via* ***prediction***

Hierarchy of memories

Dependability *via* redundancy

Separation of Concerns

Interface

Boundary – between objects or systems
Protocol – rules for interaction between parties
Contract – formalized expectations

Distribution of Labor

User (consumer) ignores *implementation*
Provider (producer) ignores *application*

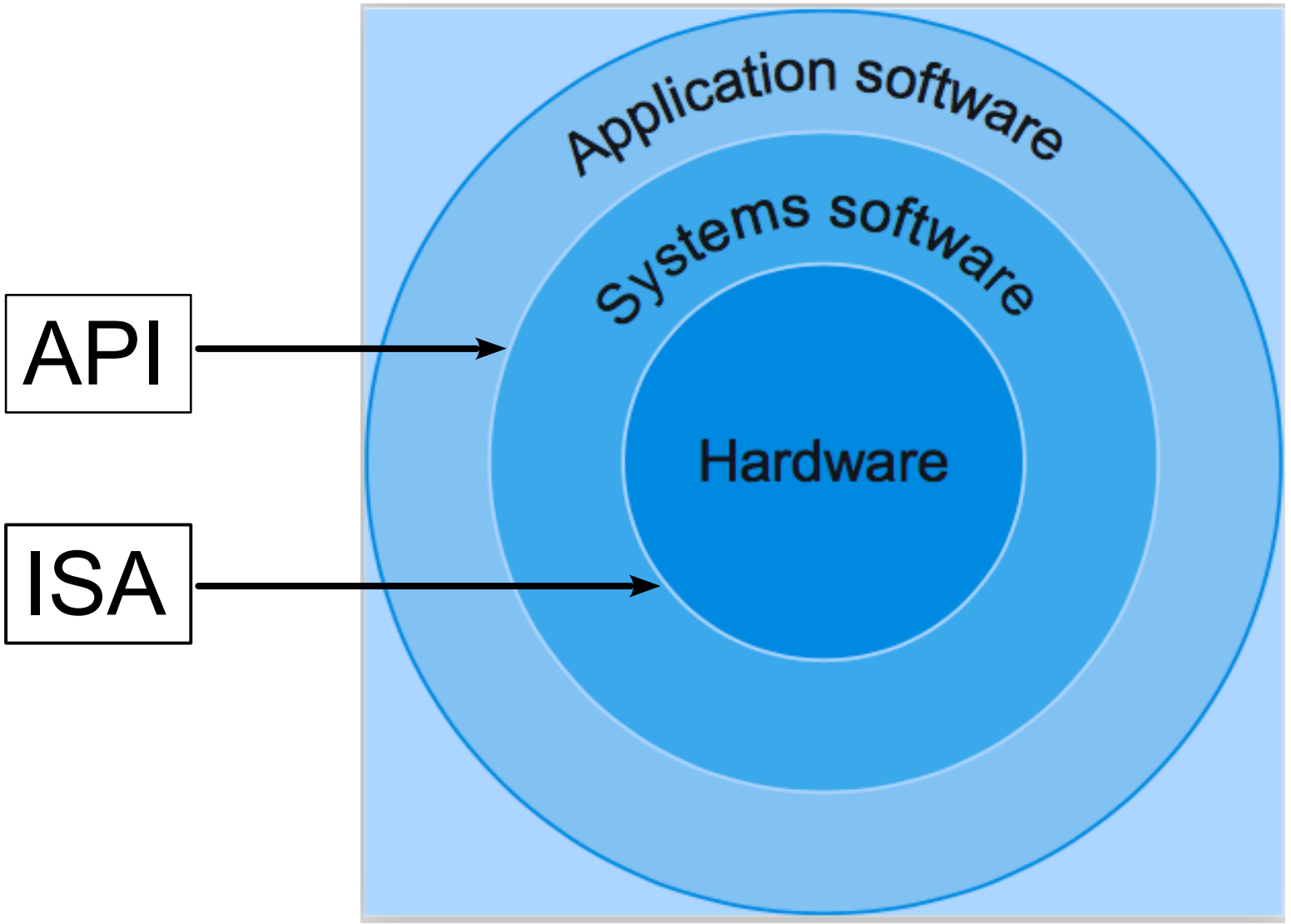
Instruction Set Architecture

Between hardware and software

Application Program Interface

Between user program and operating system

Interface Map



What Happens to Your Program

High-level language program (in C)

```
swap(int v[], int k)
{int temp;
   temp = v[k];
   v[k] = v[k+1];
   v[k+1] = temp;
}
```

Compiler

Assembly language program (for ARMv8)

```
swap:
        LSL  X10, X1,3
        ADD  X10, X0,X10
        LDUR X9, [X10,0]
        LDUR X11,[X10,8]
        STUR X11,[X10,0]
        STUR X9, [X10,8]
        BR   X10
```

Assembler

Binary machine language program (for ARMv8)

```
00000000101000100000000100011000
00000000100000100001000000100001
10001101111000100000000000000000
10001111000010010000000000000100
10101111000010010000000000000000
10101101111000100000000000000100
00000011111000000000000000001000
```

Moore's "Law"

Moore's Observation (1965)

- # gates per chip doubles (about) every two years
- Compute power $\propto$ # gates per chip

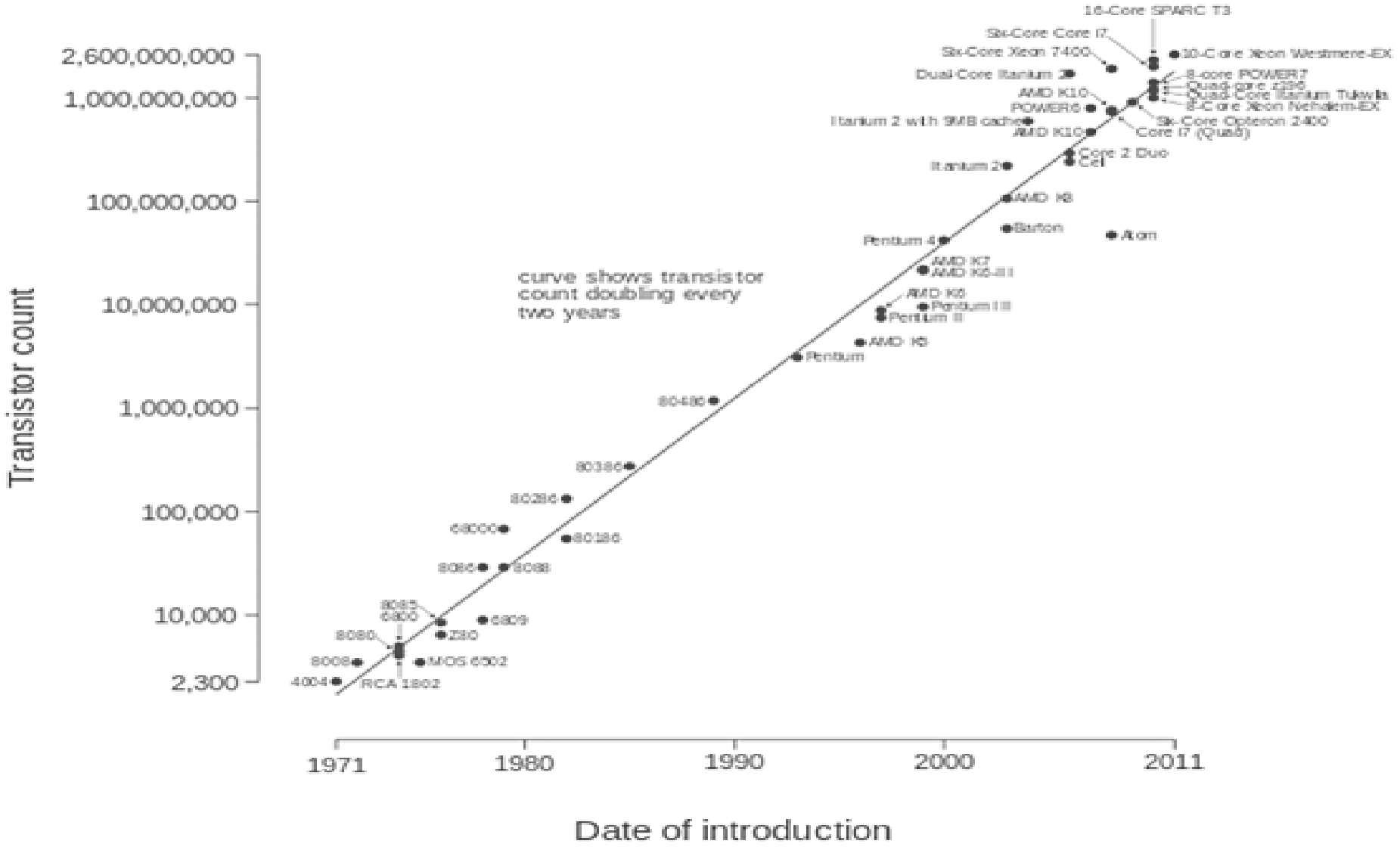
What to do with increasing compute power?

- Until about 2000, faster uniprocessors
- Since 2003, more processors per chip
  - Exploiting increasing parallelism isn't easy

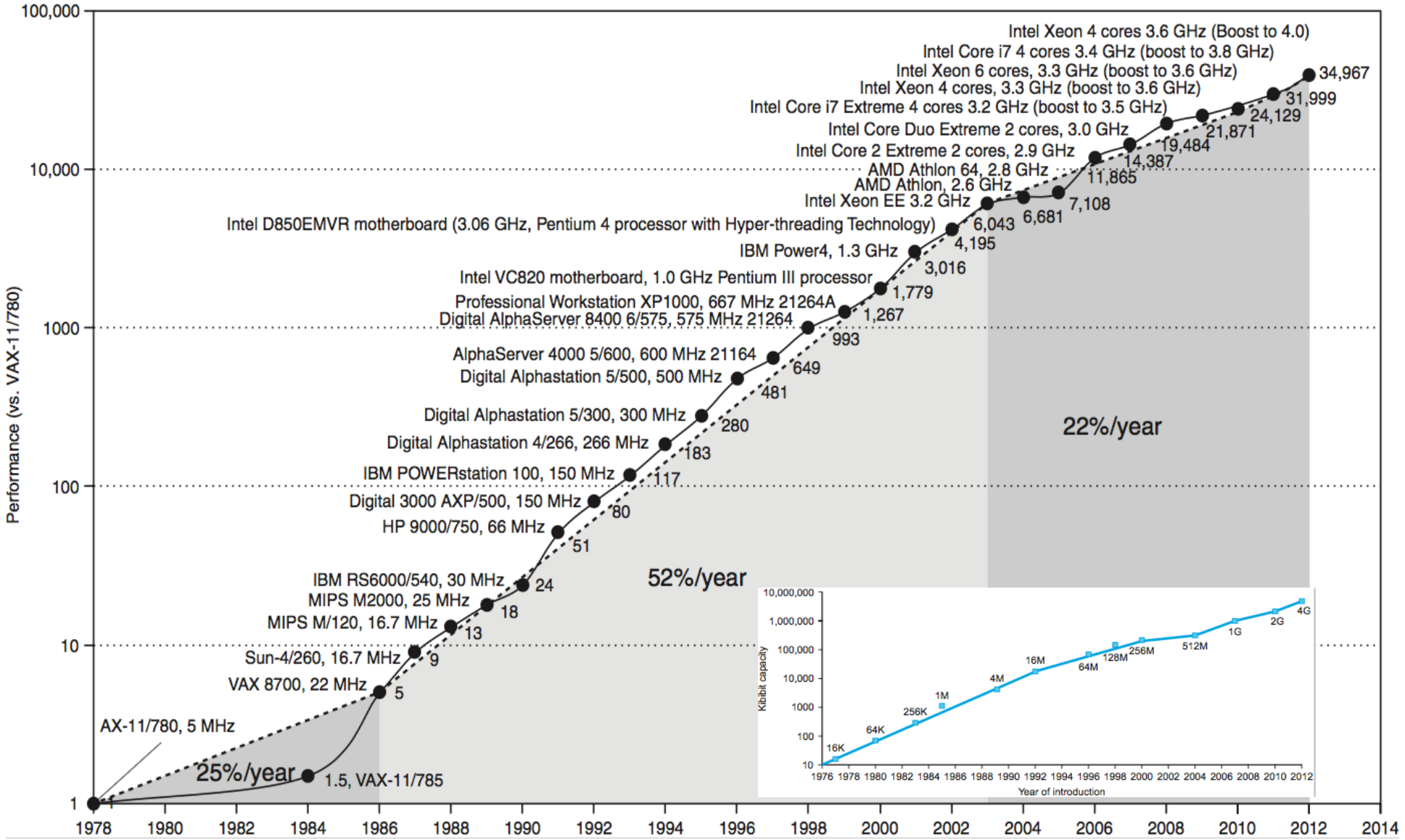
Are we the end of Moore's Law?

- Seems to be slowing down, can't continue forever
- However, it has been pronounced dead before

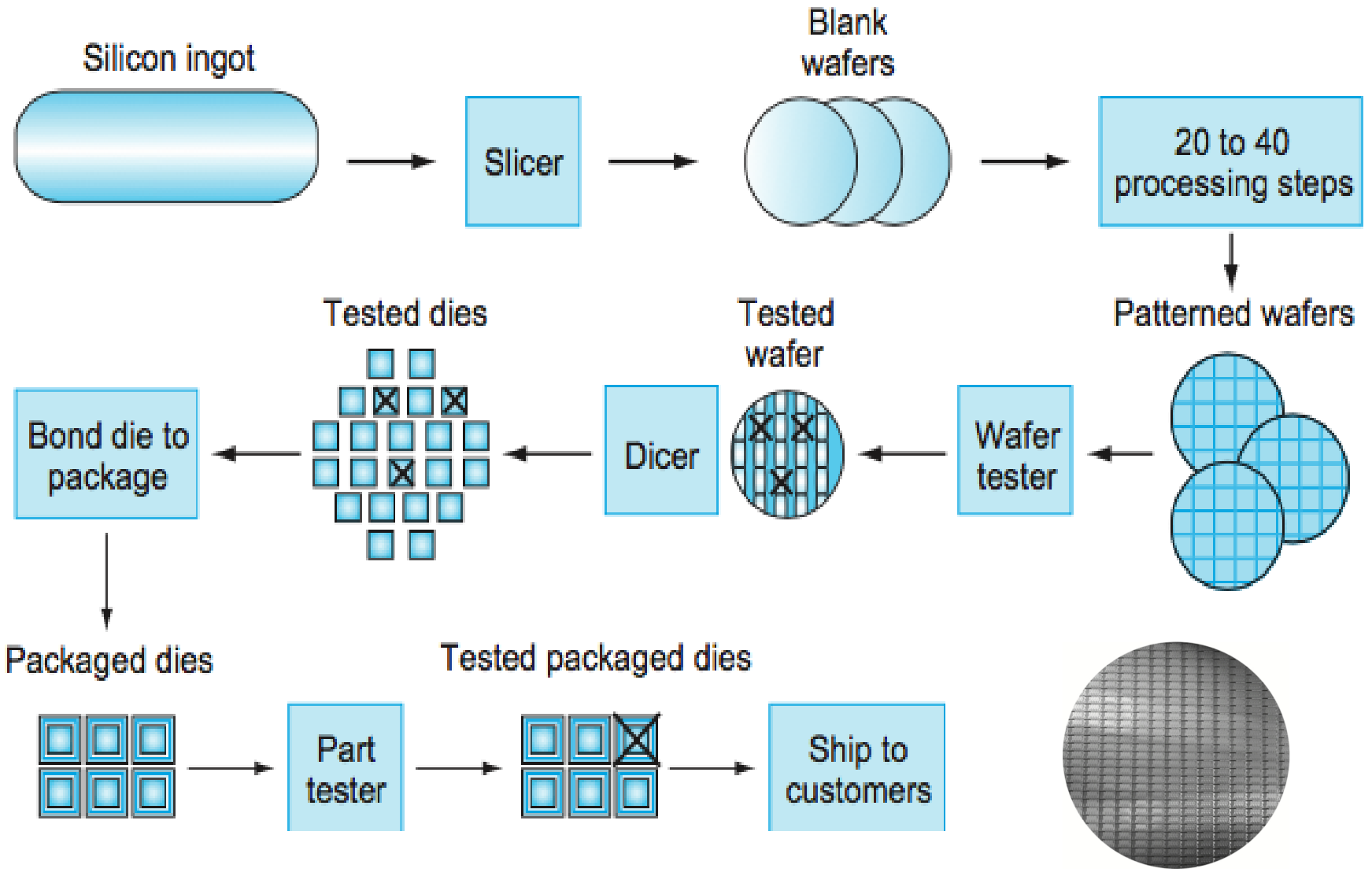
Moore's Law


Microprocessor Transistor Counts 1971-2011 & Moore's Law
Transistor count
2,600,000,000
1,000,000,000
100,000,000
10,000,000
1,000,000
100,000
10,000
2,300
curve shows transistor count doubling every two years
16-Core SPARC T3
Six-Core Core i7
Six-Core Xeon 7400
10-Core Xeon Westmere-EX
Dual-Core Itanium 2
8-core POWER7
Quad-core z196
Quad-Core Itanium Tukwila
AMD K10
POWER6
8-Core Xeon Nehalem-EX
Itanium 2 with 9MB cache
Six-Core Opteron 2400
AMD K10
Core i7 (Quad)
Core 2 Duo
Itanium 2
AMD K8
Barton
Atom
Pentium 4
AMD K7
AMD K6-III
AMD K6
Pentium III
Pentium II
AMD K6
Pentium
80486
80386
80286
68000
80186
8086
8088
8085
6800
6809
8080
Z80
8008
MOS 6502
4004
RCA 1802
1971
1980
1990
2000
2011
Date of introduction

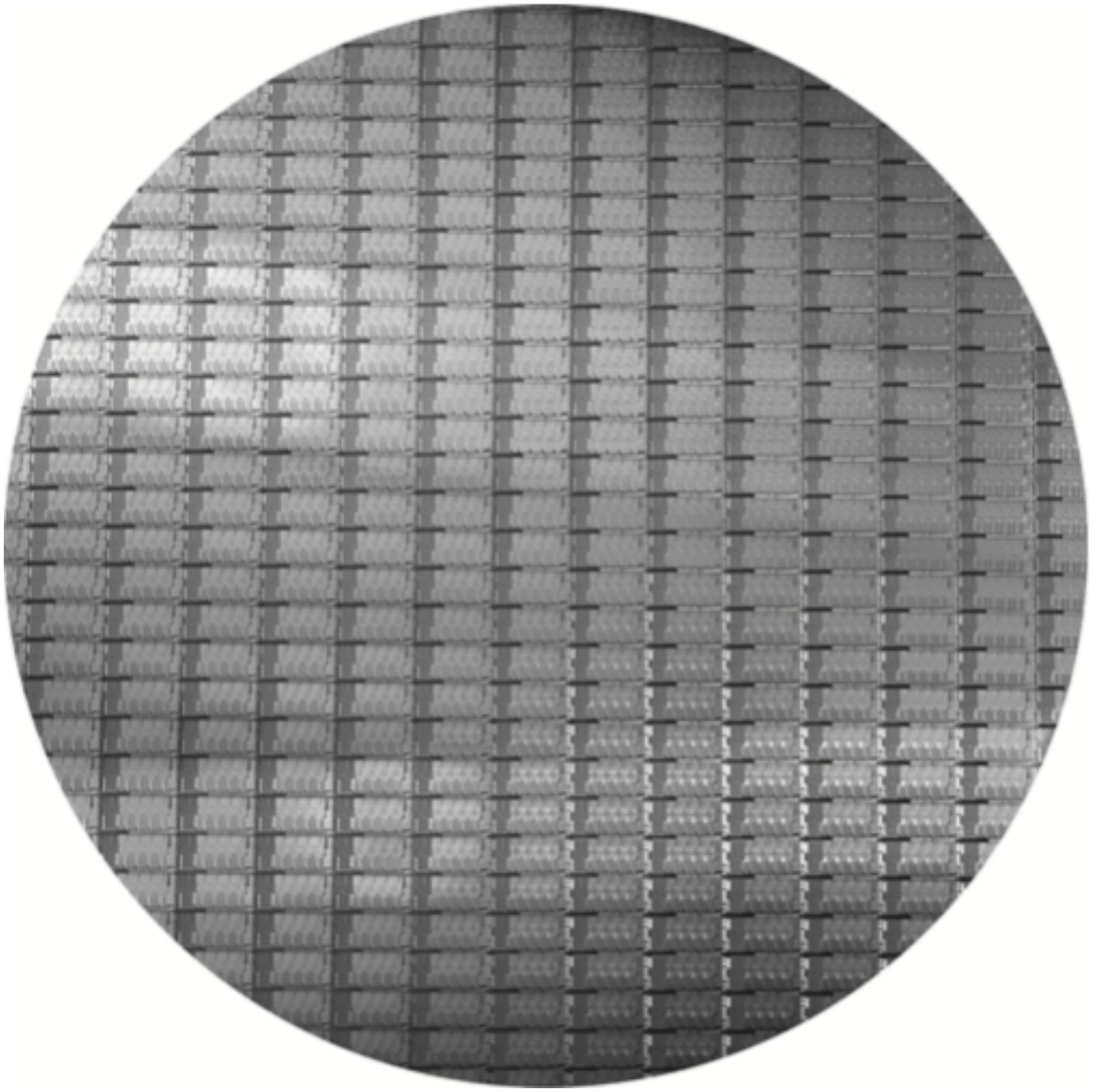
Processing Power over Time



Chip Fabrication



Intel Core I7 Wafer



Integrated Circuit Cost Equations

$$\text{Cost per die} = \frac{\text{Cost per wafer}}{\text{Dies per wafer} \times \text{yield}}$$

$$\text{Dies per wafer} \approx \frac{\text{Wafer area}}{\text{Die area}}$$

$$\text{Yield} = \frac{1}{(1 + (\text{Defects per area} \times \text{Die area}/2))^2}$$

12 inch (300mm) patterned wafer

280 (20.7 x 10.5 mm) dies per wafer

~10% of dies are defective (yield ~= 0.9)

If each chip costs $60 to fabricate

How much does the wafer cost?

Memory Hierarchy

Programmer want fast, large, and cheap memory

Memory technologies vary from very fast, very small, and very expensive to very slow, very large, and inexpensive

The illusion of fast, large, and affordable memory depend upon ***locality of references***:

The memory addresses referenced by a program tend to glom together in time *(**temporal locality**)* and space *(**spacial locality**)*

Boolean Algebra

Constants: **0** and **1**

Operators: $\overline{\ }$ (not), ● (and), **+** (or)

A	$\overline{\mathbf{A}}$
0	1
1	0

A	**B**	**A + B**	**A ● B**
0	0	0	0
0	1	1	0
1	0	1	0
1	1	1	1

Proof by Truth Table

$$(X \bullet \overline{Y}) + (Y \bullet \overline{X}) = (X + Y) \bullet (X + \overline{Y})$$

x	y	$\overline{x}$	$\overline{y}$	$x \bullet \overline{y}$	$y \bullet \overline{x}$	$(x \bullet \overline{y}) + (y \bullet \overline{x})$	$(x+y) \bullet (\overline{y}+\overline{x})$	x+y	$\overline{y}+\overline{x}$
0	0	1	1	0	**0**	**0**	0	0	1
0	1	1	0	0	**1**	**1**	1	1	1
1	0	0	1	1	**0**	**1**	1	1	1
1	1	0	0	0	**0**	**0**	0	1	0

Proof by Truth Table

X • (Y + Z) = (X • Y) + (X • Z)

X	Y	Z	Y + Z	X • (Y + Z)	(X • Y) + (X • Z)	X • Y	X • Z
0	0	0	0	**0**	**0**	0	0
0	0	1	1	**0**	**0**	0	0
0	1	0	1	**0**	**0**	0	0
0	1	1	1	**0**	**0**	0	0
1	0	0	0	**0**	**0**	0	0
1	0	1	1	**1**	**1**	0	1
1	1	0	1	**1**	**1**	1	0
1	1	1	1	**1**	**1**	1	1

Boolean Identities

$$\overline{\overline{X}} = X$$

$X \bullet (Y \bullet Z) = (X \bullet Y) \bullet Z$	$X + (Y + Z) = (X + Y) + Z$
$X \bullet 1 = X$	$X + 0 = X$
$X \bullet \overline{X} = 0$	$X + \overline{X} = 1$
$X \bullet Y = Y \bullet X$	$X + Y = Y + X$
$X \bullet X = X$	$X + X = X$
$X \bullet 0 = 0$	$X + 1 = 1$
$(X \bullet Y) \bullet X = X \bullet Y$	$(X + Y) + X = X + Y$
$X \bullet (Y + Z) = (X \bullet Y) + (X \bullet Z)$	$X + (Y \bullet Z) = (X + Y) \bullet (X + Z)$
$\overline{X \bullet Y} = \overline{X} + \overline{Y}$	$\overline{X + Y} = \overline{X} \bullet \overline{Y}$