Chapter 1

Computer Abstractions and Technology

Outline

- Introduction
- Eight great ideas in computer architecture
- How a program is executed
- Under the cover of computers
- Technologies for building processors and memory

The Computer Revolution

- Progress in computer technology
  - Underpinned by Moore's Law
- Makes novel applications feasible
  - Computers in automobiles
  - Cell phones
  - Human genome project
  - World Wide Web
  - Search Engines
- Computers are pervasive

Classes of Computers

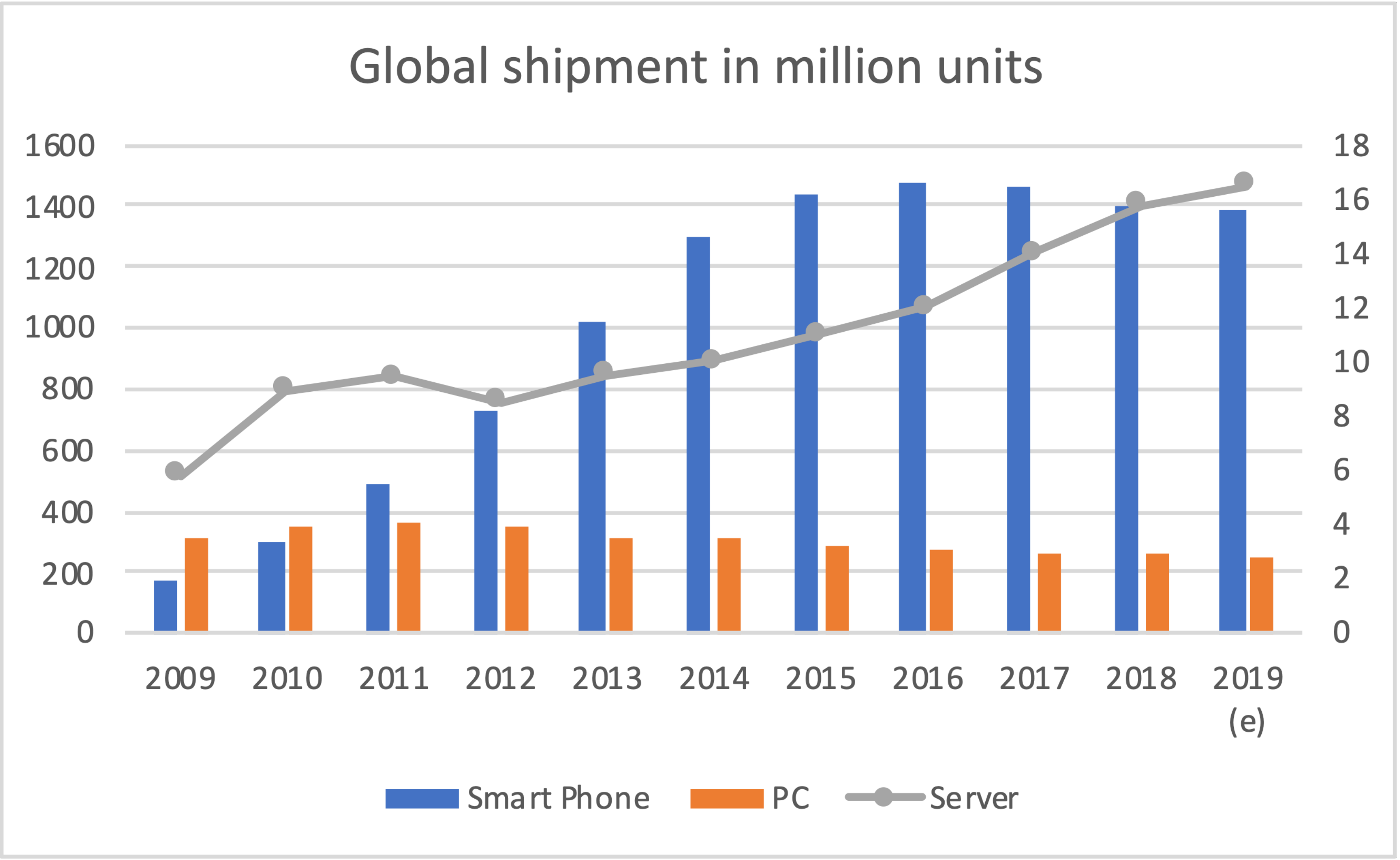
- Personal computers
  - General purpose, variety of software
  - Subject to cost/performance tradeoff

- Server computers
  - Network based
  - High capacity, performance, reliability
  - Range from small servers to building sized

Classes of Computers

- Supercomputers
  - High-end scientific and engineering calculations
  - Highest capability but represent a small fraction of the overall computer market
- Embedded computers
  - Hidden as components of systems
  - Stringent power/performance/cost constraints

Smart Phone vs. PC Sales



The PostPC Era

- Personal Mobile Device (PMD)
  - Battery operated
  - Connects to the Internet
  - Hundreds of dollars
  - Smart phones, tablets, electronic glasses
- Cloud computing
  - Warehouse Scale Computers (WSC)
  - Software as a Service (SaaS)
  - Portion of software run on a PMD and a portion run in the Cloud
  - Amazon and Google

What You Will Learn

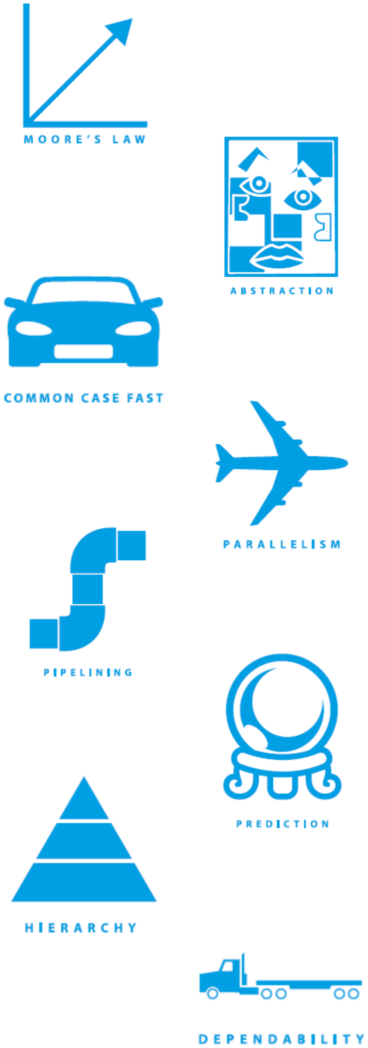
- How programs are translated into the machine language
  - And how the hardware executes them
- How instruction sets work as the hardware/software interface
- What determines program performance
- How hardware designers improve performance
- What is parallel processing

Understanding Performance

- Algorithm
  - Determines number of operations executed
- Programming language, compiler, architecture
  - Determine number of machine instructions executed per operation
- Processor and memory system
  - Determine how fast instructions are executed
- I/O system (including OS)
  - Determines how fast I/O operations are executed

Eight Great Ideas

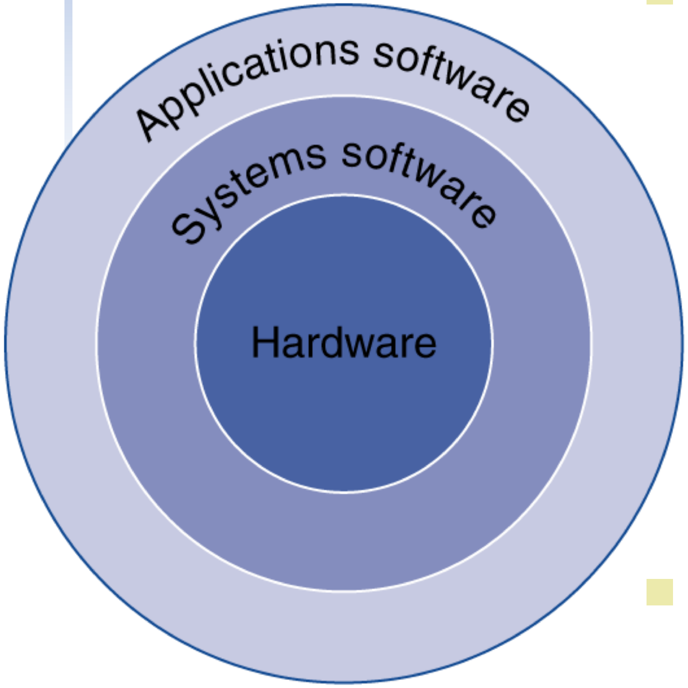
- Design for ***Moore's Law***
- Use ***abstraction*** to simplify design
- Make the ***common case fast***
- Performance *via* ***parallelism***
- Performance *via* ***pipelining***
- Performance *via* ***prediction***
- ***Hierarchy*** of memories
- ***Dependability*** *via* redundancy



- Moore's Law
  - Integrated resources double every 18-24 months.
- Abstraction
  - Lower-level details are hidden with a simpler model
  - Ex. transistor → gate → digital circuit
- Make common case fast
  - Common cases consume most time in a process.
  - 90/10 rule

- Parallelism
  - Ex. more workers to pick fruits in a farm
  - Counter Ex. more engineers on designing one product
- Pipelining
  - Ex. assembly line works
- Prediction
  - Ex. weather forecasting based on current weather, barometric measure, clouds, etc.
  - Cost associated with mis-prediction?
- Memory hierarchy
  - Ex. popular collection section in a library
- Redundancy
  - Ex. backing up your data at different cloud servers

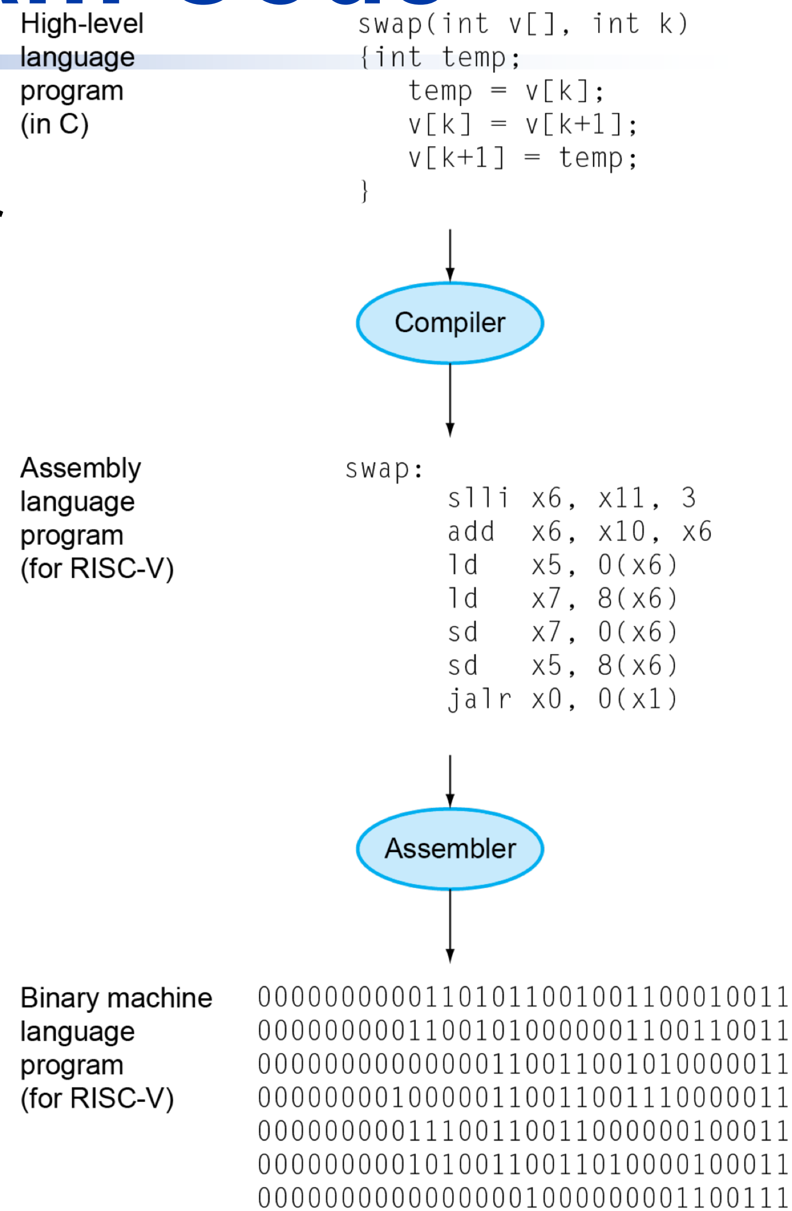
Below Your Program



- Application software
  - Written in high-level language
- System software
  - Compiler: translates high-level code to machine code
  - Operating System: service code
    - Handling input/output
    - Managing memory and storage
    - Scheduling tasks & sharing resources
- Hardware
  - Processor, memory, I/O controllers

Levels of Program Code

- High-level language
  - Level of abstraction closer to problem domain
  - Provides for productivity and portability
- Assembly language
  - Textual representation of instructions
- Hardware representation
  - Binary digits (bits)
  - Encoded instructions and data



Operating Systems

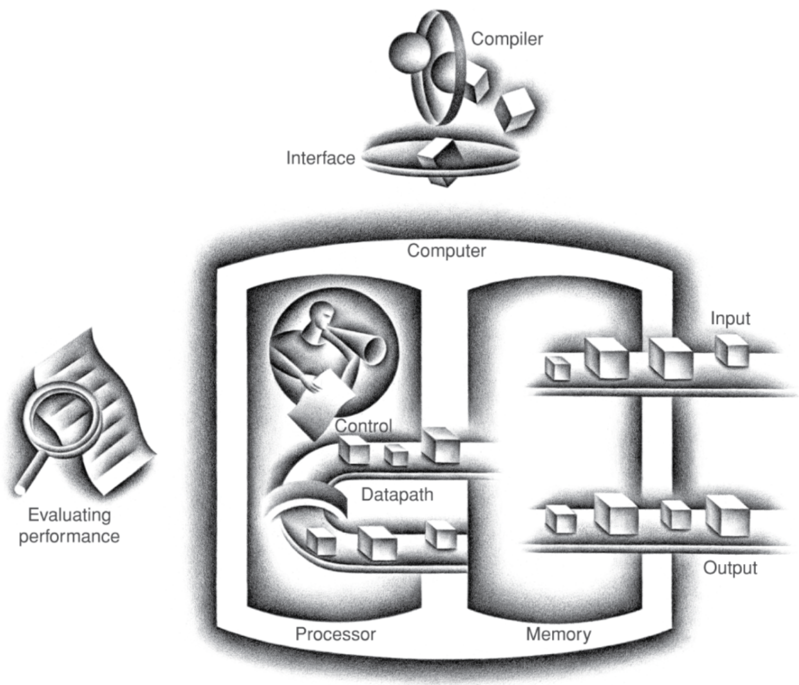
- Load and execute user programs
- Provide input/output interface to programs
- Schedule processes and threads
- Manage memories
- Manage storage
- Manage networking
- Manage system security

Compiler and Toolchain

- **Compiler** will analyze your source codes and translate them into (optimized) machine codes.
- **Linker** will include libraries into the program.
- **Debugger** helps programmer to examine machine and memory states
- **Profiler** measures the performance of code segments

Components of a Computer

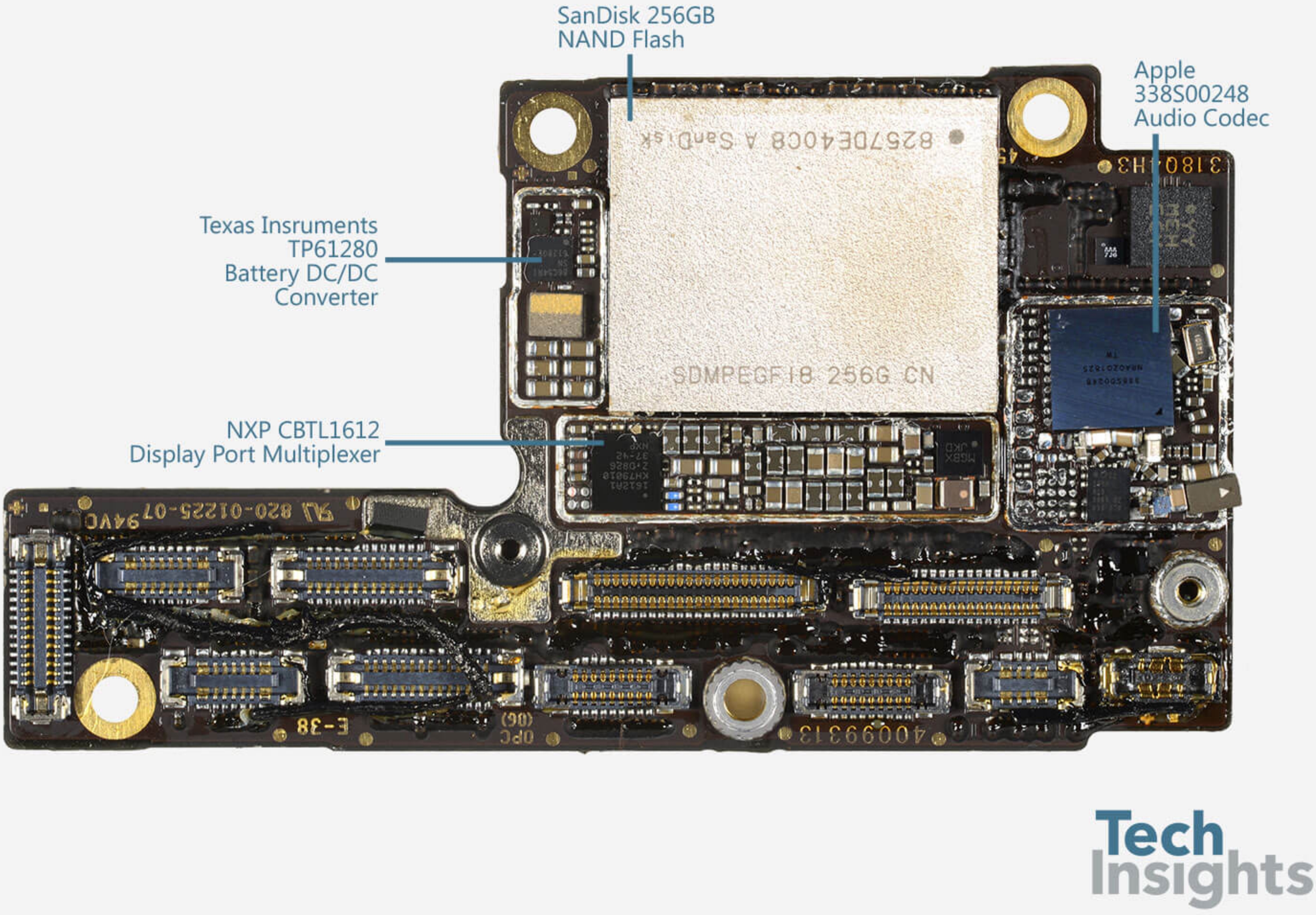
The BIG Picture

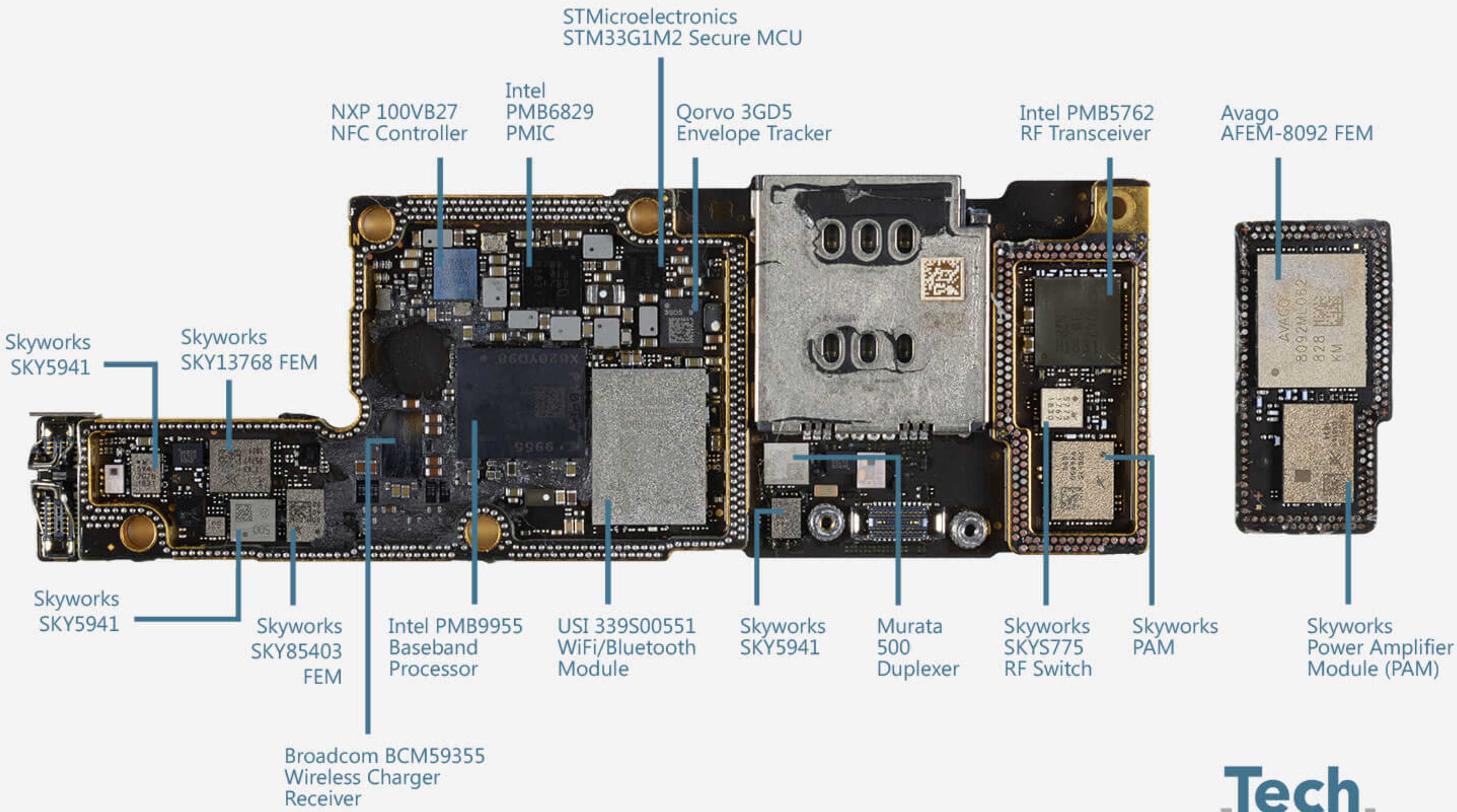


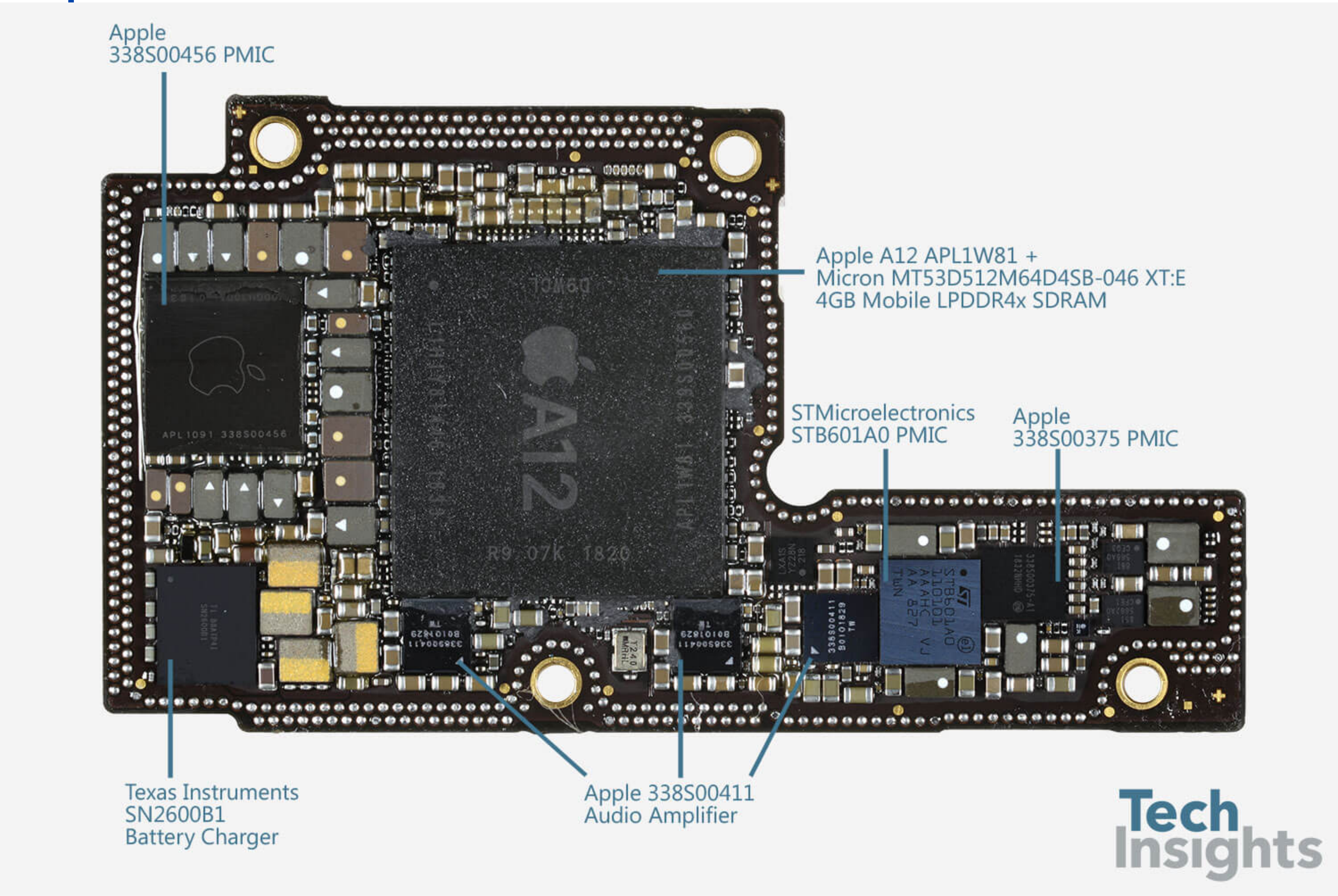
- Same components for all kinds of computer
  - Desktop, server, embedded
- Input/output includes
  - User-interface devices
    - Display, keyboard, mouse
  - Storage devices
    - Hard disk, CD/DVD, flash
  - Network adapters
    - For communicating with other computers

iPhone Xs Max Teardown

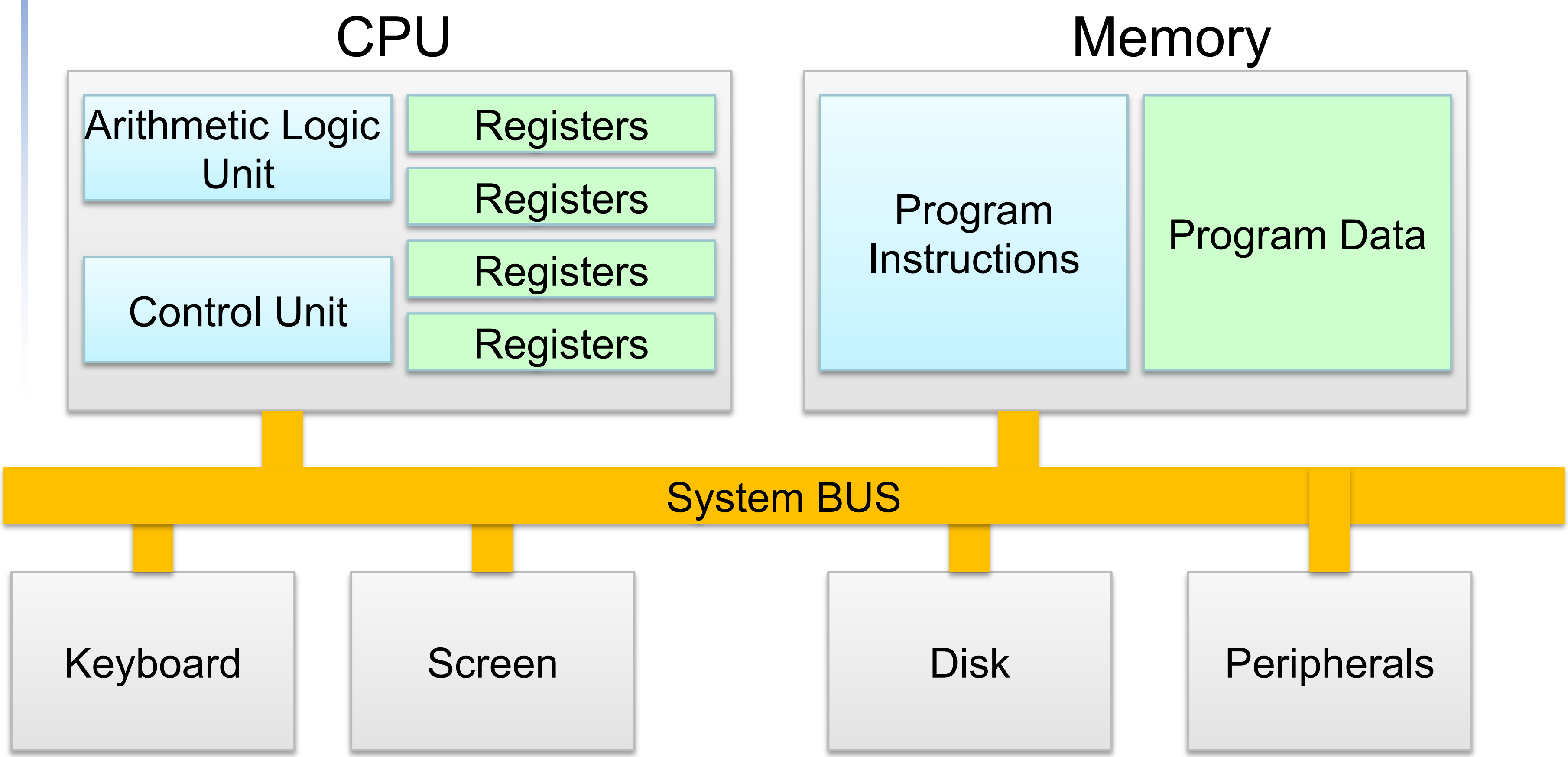
https://www.techinsights.com/blog/apple-iphone-xs-max-teardown
https://www.ifixit.com/Teardown/iPhone+XS+and+XS+Max+Teardown/113021


SanDisk 256GB
NAND Flash
Apple
338S00248
Audio Codec
Texas Insruments
TP61280
Battery DC/DC
Converter
NXP CBTL1612
Display Port Multiplexer
SDMPEGF18 256G CN
TechInsights


STMicroelectronics
STM33G1M2 Secure MCU
NXP 100VB27
NFC Controller
Intel
PMB6829
PMIC
Qorvo 3GD5
Envelope Tracker
Intel PMB5762
RF Transceiver
Avago
AFEM-8092 FEM
Skyworks
SKY5941
Skyworks
SKY13768 FEM
Skyworks
SKY5941
Skyworks
SKY85403
FEM
Intel PMB9955
Baseband
Processor
USI 339S00551
WiFi/Bluetooth
Module
Skyworks
SKY5941
Murata
500
Duplexer
Skyworks
SKYS775
RF Switch
Skyworks
PAM
Skyworks
Power Amplifier
Module (PAM)
Broadcom BCM59355
Wireless Charger
Receiver


Apple
338S00456 PMIC
Apple A12 APL1W81 +
Micron MT53D512M64D4SB-046 XT:E
4GB Mobile LPDDR4x SDRAM
STMicroelectronics
STB601A0 PMIC
Apple
338S00375 PMIC
Texas Instruments
SN2600B1
Battery Charger
Apple 338S00411
Audio Amplifier
A12
TechInsights

Inside a Computer



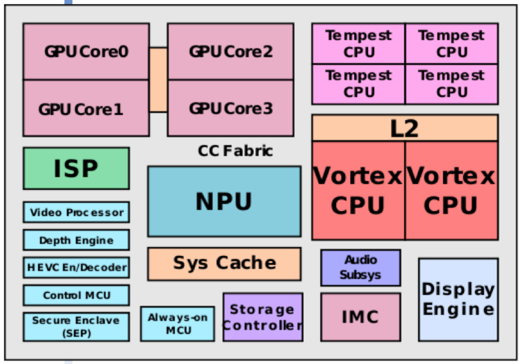
Central Processing Unit (CPU)

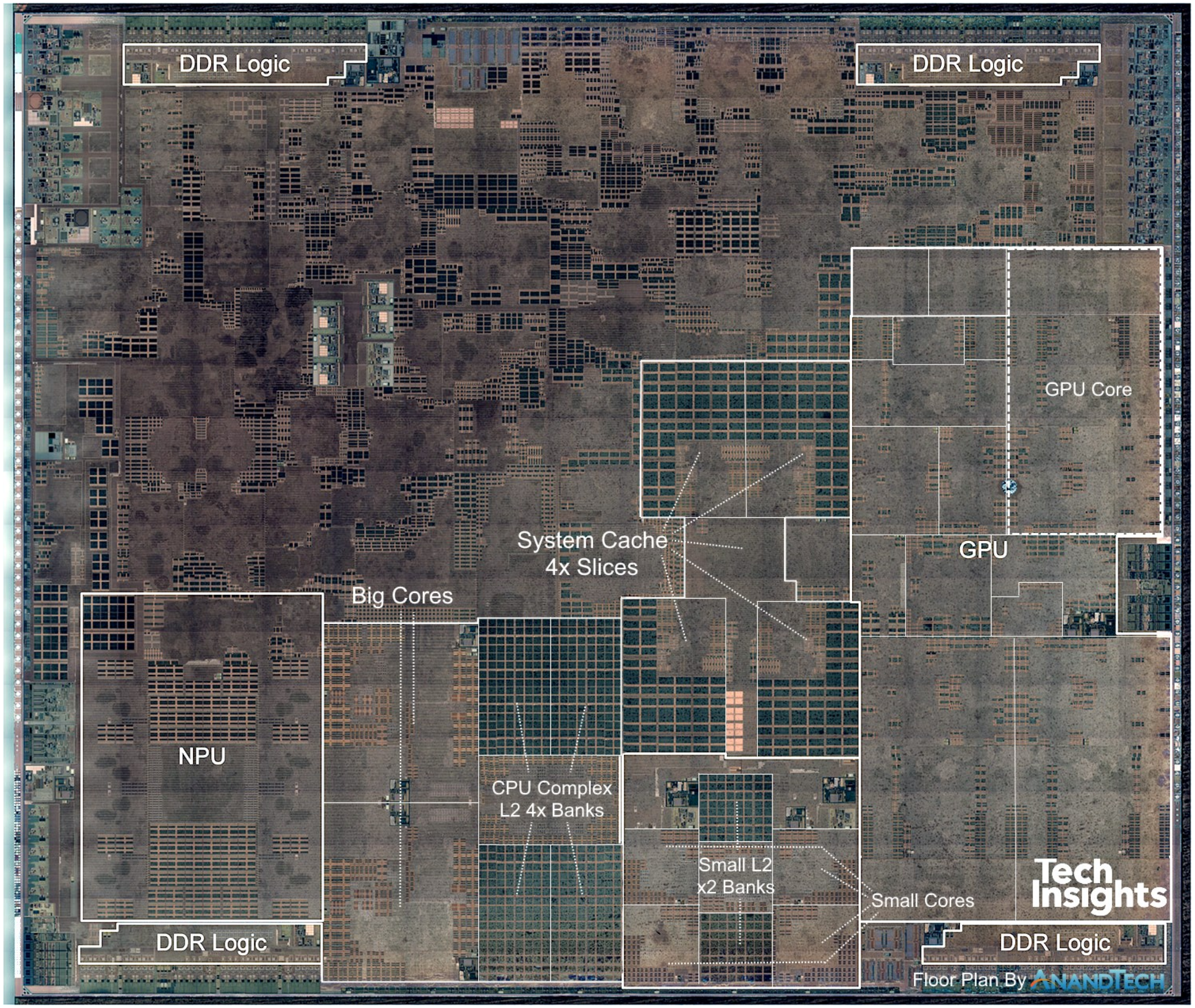
- Control Unit
  - A finite state machine (FSM)
  - Retrieves and decodes program instructions
  - Generate signals to coordinate computer operations: load/store registers, perform ALU functions, take branches, etc…
- Arithmetic & Logic Unit (ALU)
  - Performs mathematical operations

Inside the Processor (CPU)

- Datapath: performs operations on data
- Control: sequences datapath, memory, ...
- Cache memory
  - Small fast SRAM memory for immediate access to data

Apple A12 Bionic Die Photo





- TSMC 7 nm process
- 6.9 billion transistors
- 2 big Vortex cores at up to 2.4 GHz
- 4 little Tempest high-efficiency cores
- 4 core GPU

Data Storage

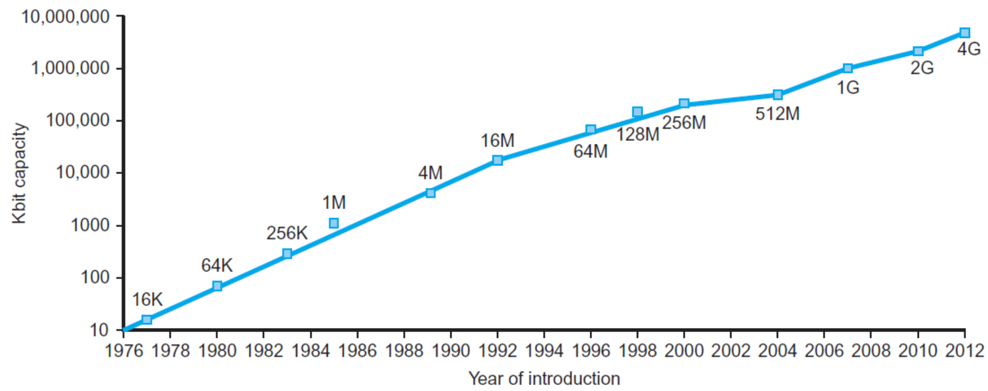
- Volatile main memory
  - Loses data when power off
  - SRAM
  - DRAM
- Non-volatile secondary memory
  - Magnetic disk
  - Flash memory and SSD
  - Optical disk (CDROM, DVD)
  - Tapes

Networking

- Communication, resource sharing, nonlocal access
- Local area network (LAN): Ethernet
- Wide area network (WAN): the Internet
- Wireless network: WiFi, Bluetooth

Technology Trends

- Electronics technology continues to evolve
  - Increased capacity and performance
  - Reduced cost



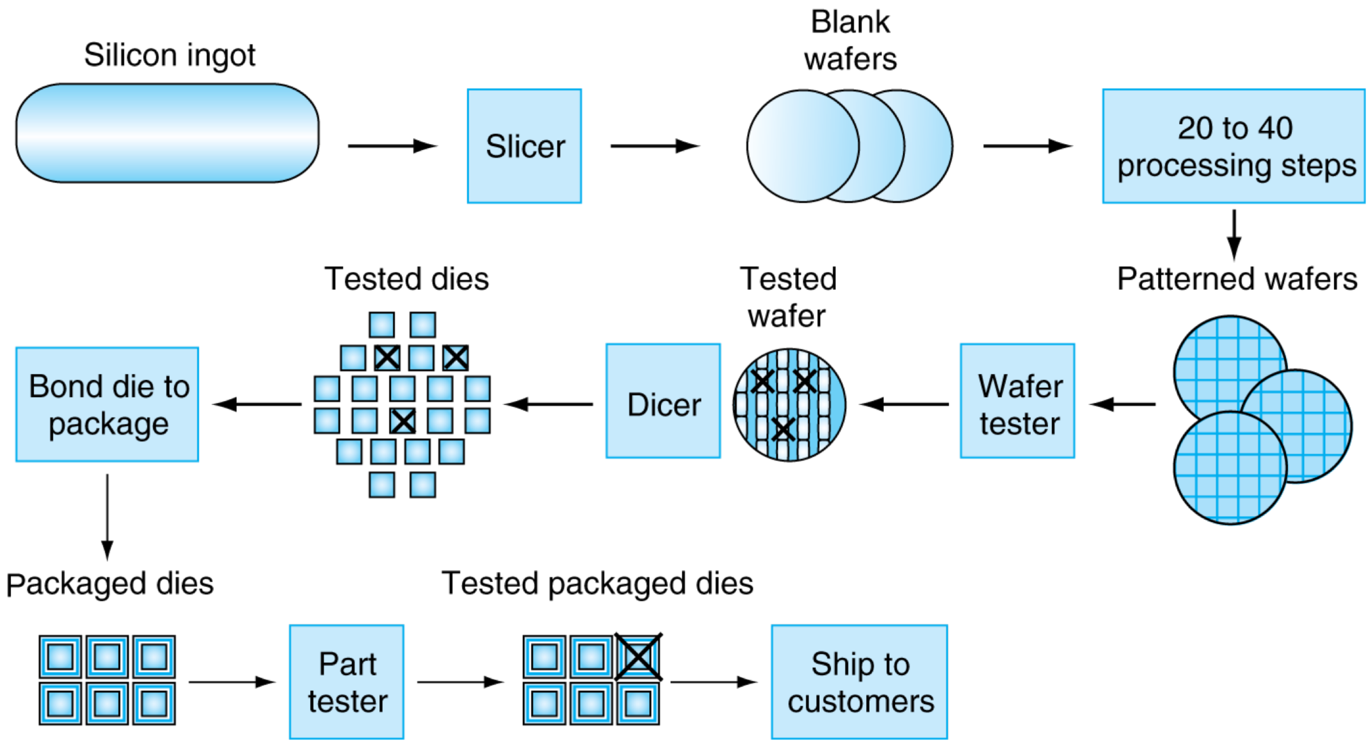
DRAM capacity

Year	Technology	Relative performance/cost
1951	Vacuum tube	1
1965	Transistor	35
1975	Integrated circuit (IC)	900
1995	Very large scale IC (VLSI)	2,400,000
2013	Ultra large scale IC	250,000,000,000

Semiconductor Technology

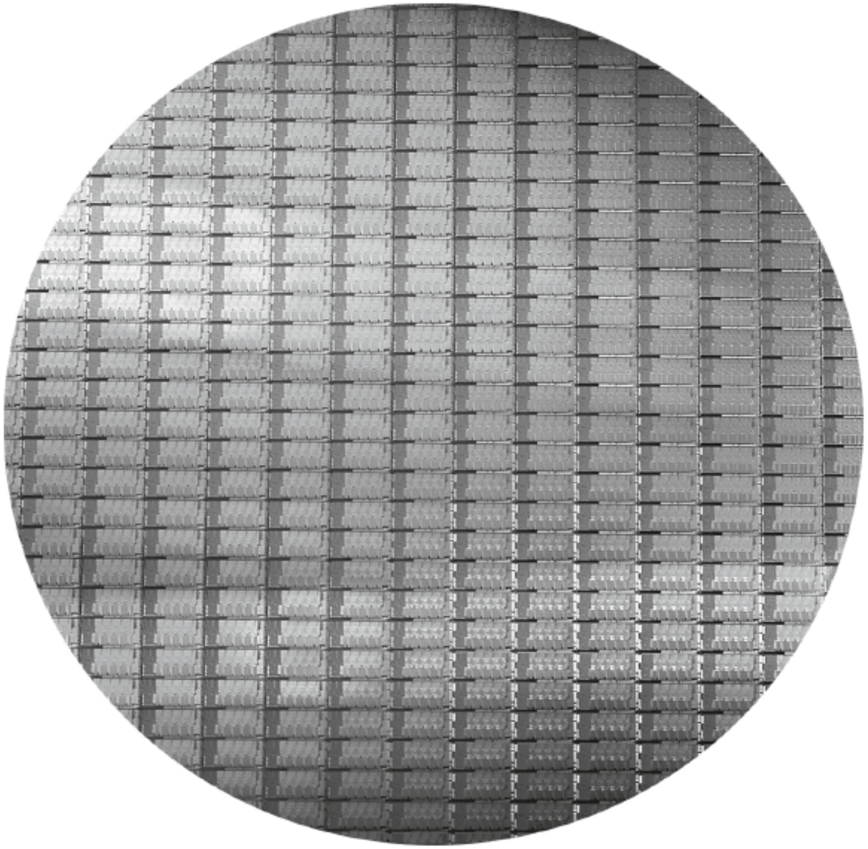
- Use silicon as a base material to create semiconductor properties
  - Controllable conductivity
- Transistors
  - Switch (digital)
  - Amplifier (analog)
- Integrated circuits
  - Use layers of conductors to interconnect transistors

Manufacturing ICs



- Yield: proportion of working dies per wafer

Intel Core i7 Wafer



- 300mm wafer, 280 chips, 32nm technology
- Each chip is 20.7 x 10.5 mm

Integrated Circuit Cost

每塊Wafer的Cost除上可用的Dies個數

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

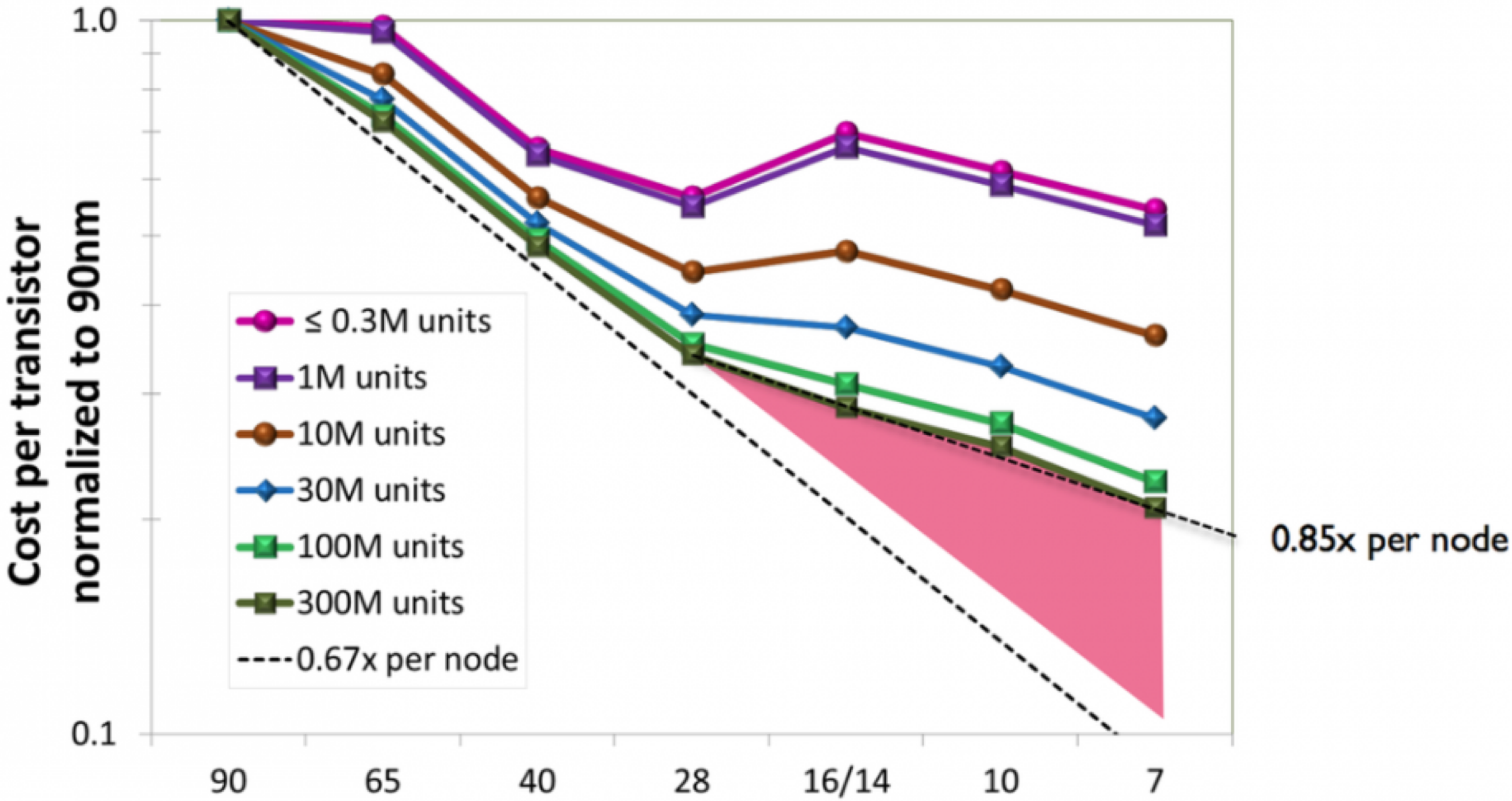
Wafer總共Dies數 x 良率

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

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

- Nonlinear relation to area and defect rate
  - Wafer cost and area are fixed
  - Defect rate determined by manufacturing process
  - Die area determined by architecture and circuit design

Transistor Cost



Source: ARM

[IEDM keynote: cost scaling will swap architectural changes for area]

Big Picture

Abstraction Layers

Application
Algorithm
Software
ISA
Microarchitecture
Circuit
Transistor

Abstractions

The BIG Picture

- Abstraction helps us deal with complexity
  - Hide lower-level detail
- Instruction set architecture (ISA)
  - The hardware/software interface
- Application binary interface
  - The ISA plus system software interface
- Implementation
  - The details underlying and interface