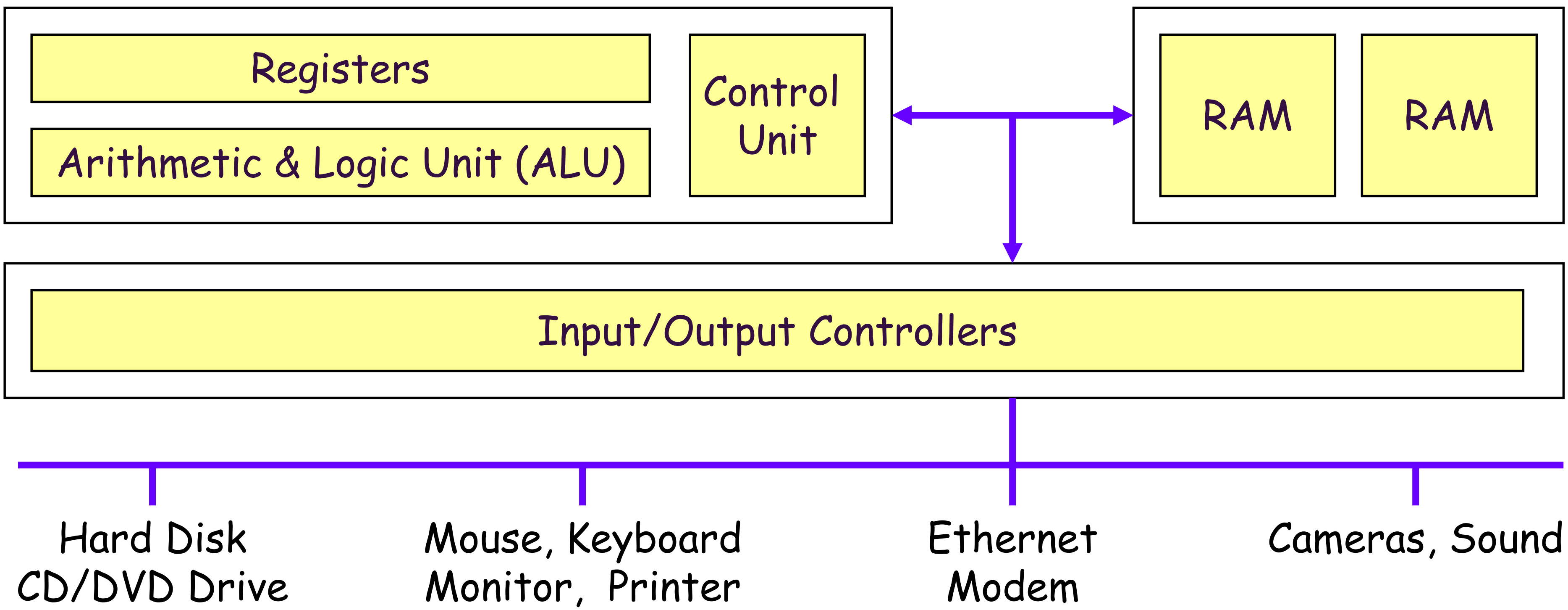
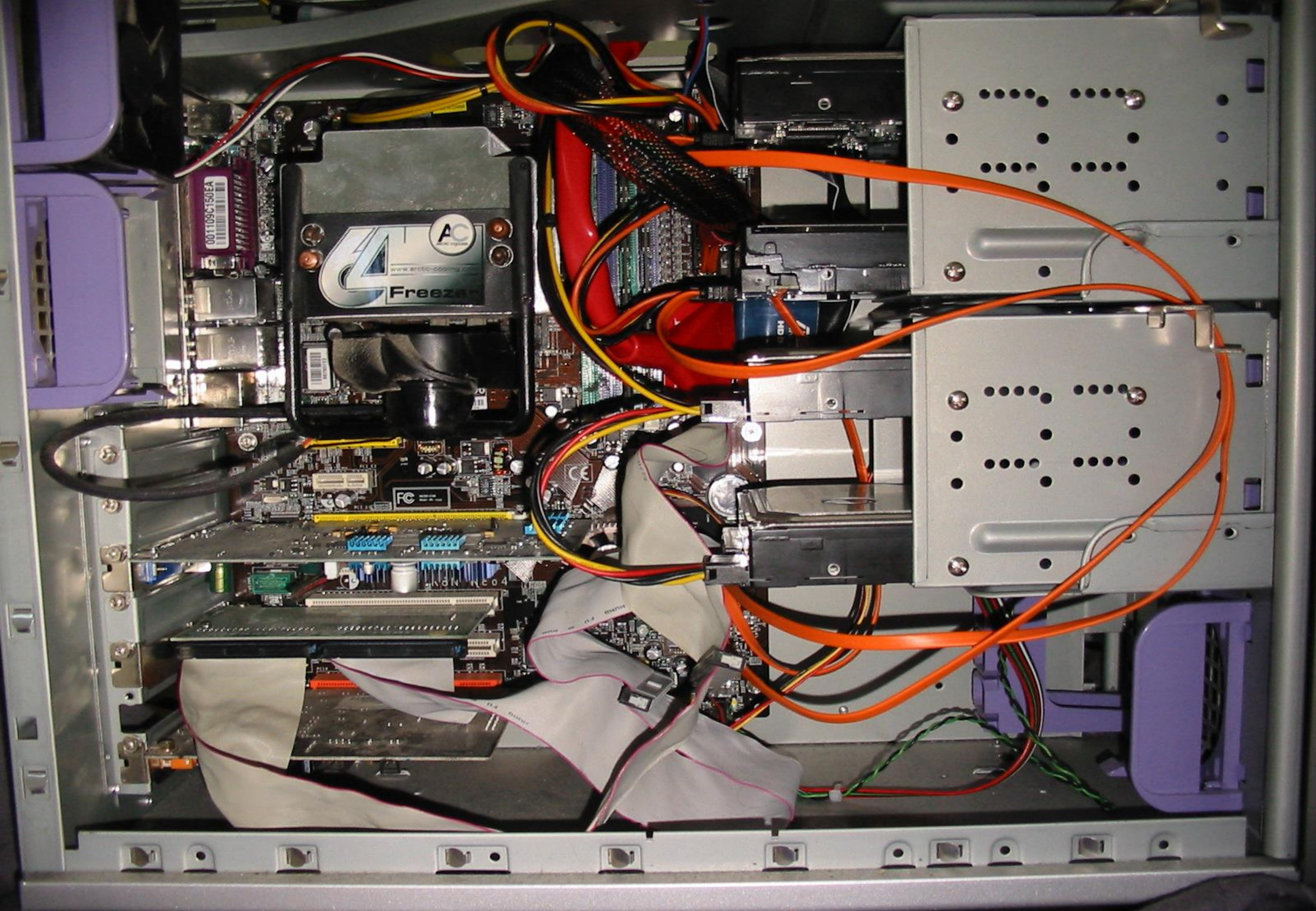
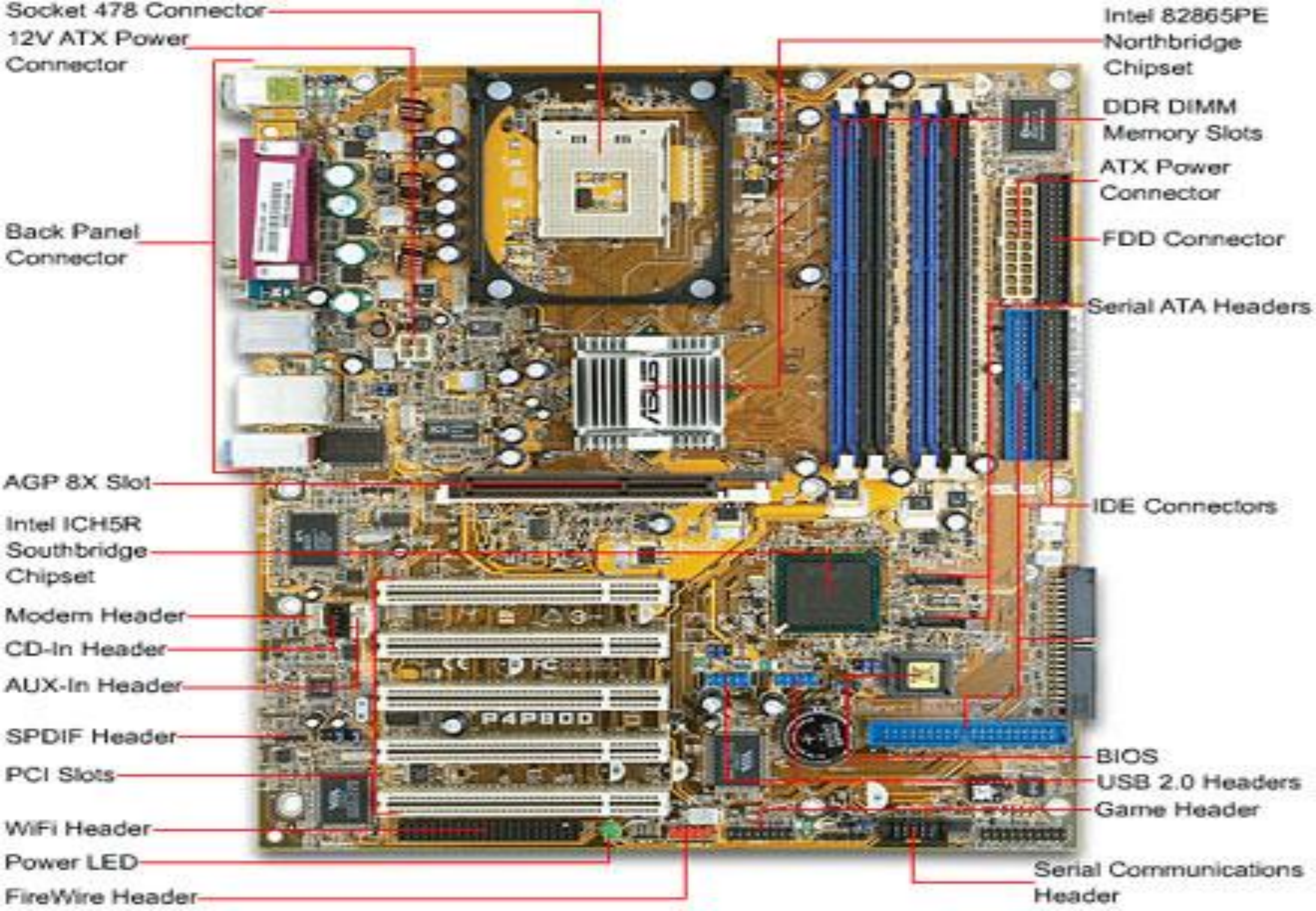
Computer: Key Components




AM/FM
R33008
37S0115679
SPDIF

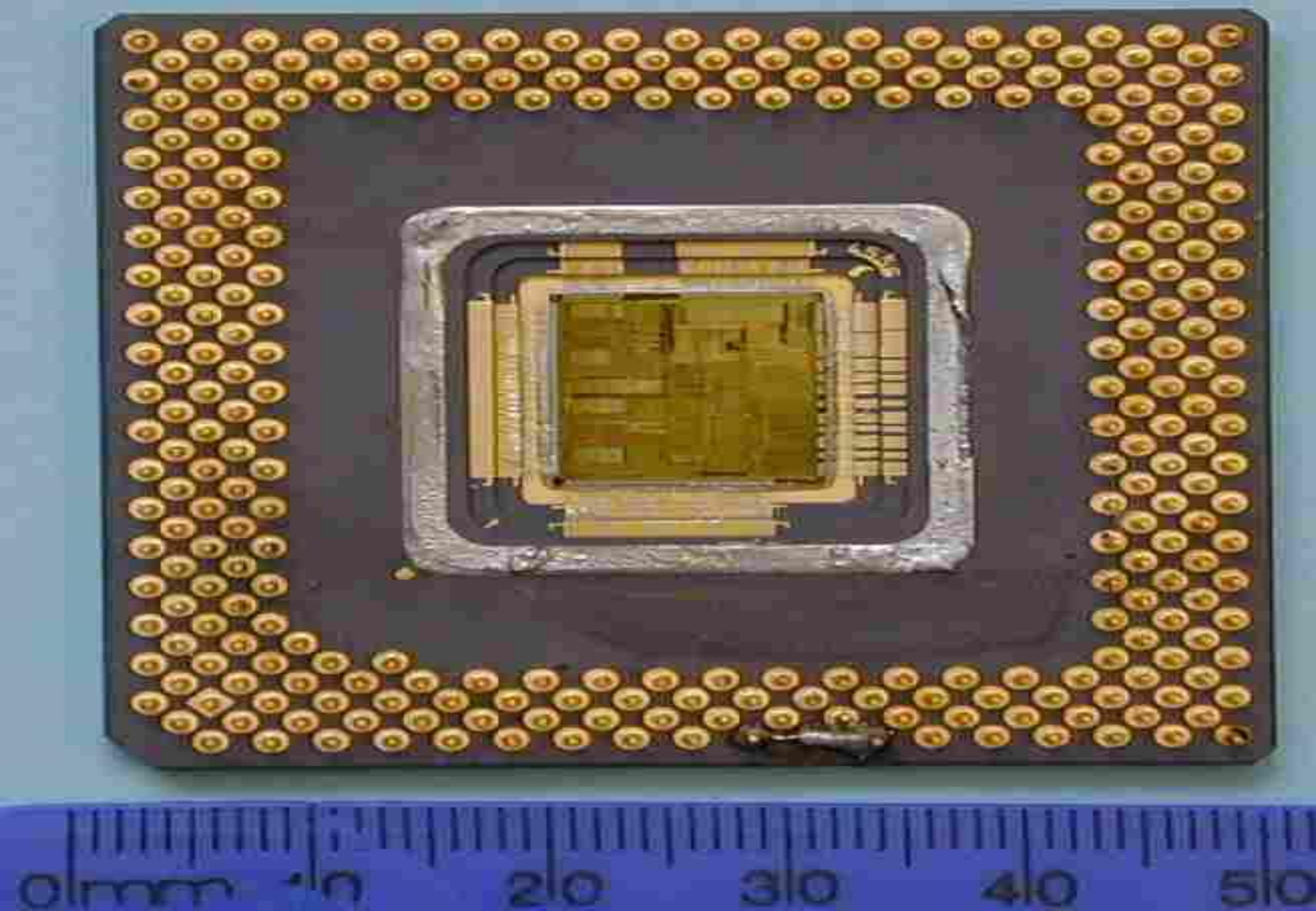

001109C150EA
AC
Freezer
CE


Socket 478 Connector
12V ATX Power Connector
Back Panel Connector
AGP 8X Slot
Intel ICH5R Southbridge Chipset
Modem Header
CD-In Header
AUX-In Header
SPDIF Header
PCI Slots
WiFi Header
Power LED
FireWire Header
Intel 82865PE Northbridge Chipset
DDR DIMM Memory Slots
ATX Power Connector
FDD Connector
Serial ATA Headers
IDE Connectors
BIOS
USB 2.0 Headers
Game Header
Serial Communications Header
P4P800

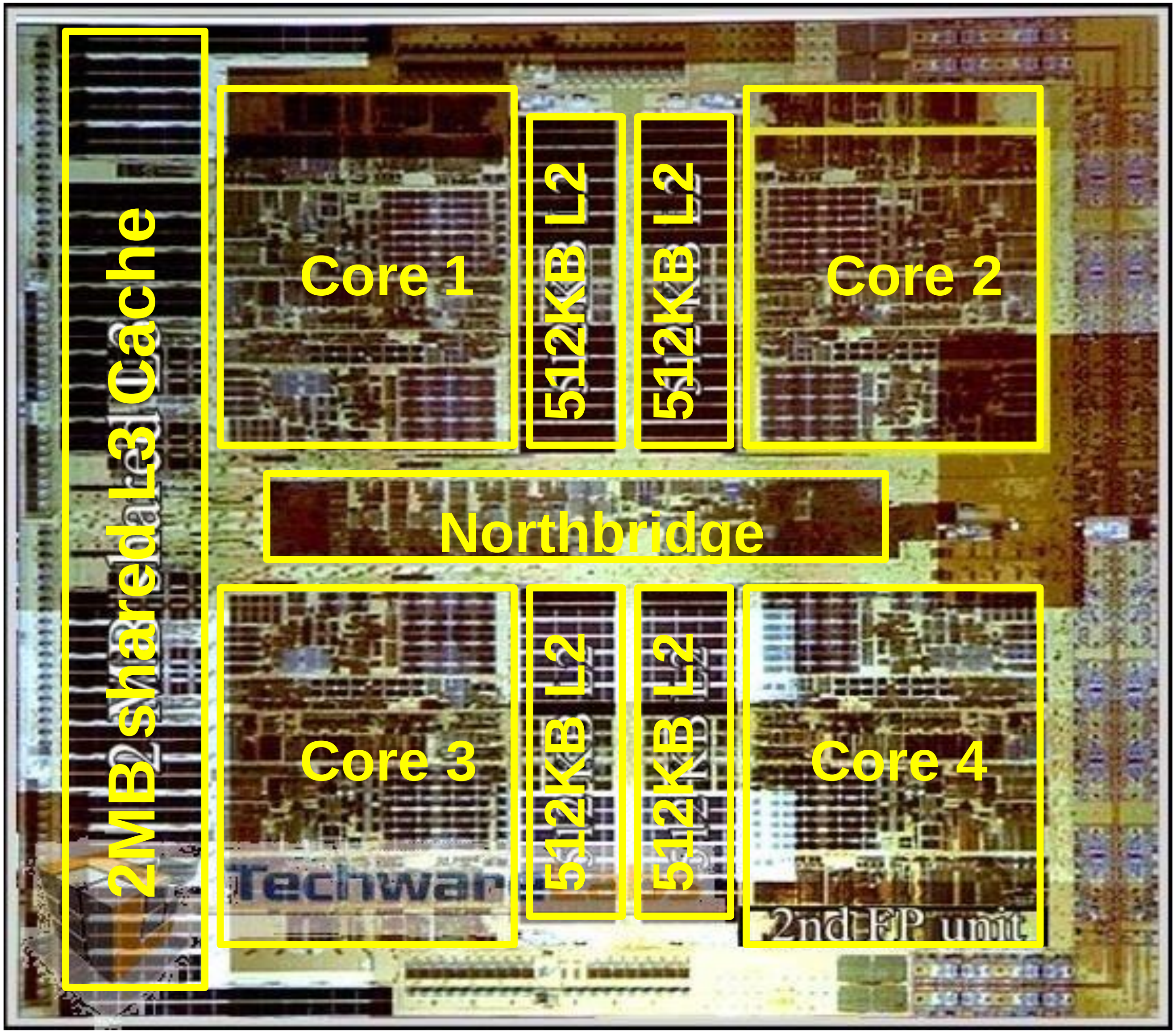

SanDisk
ultra II
2.0GB MagicGate
SanDisk
Memory Stick PRO Duo
1.0GB
Compact Flash Card
2GB
MMC PLUS
512
Kingston
512


Industrial
pqi
TURBO
2.5" SATA
Solid State Disk
64GB

Source: MKP


0 mm
10
20
30
40
50

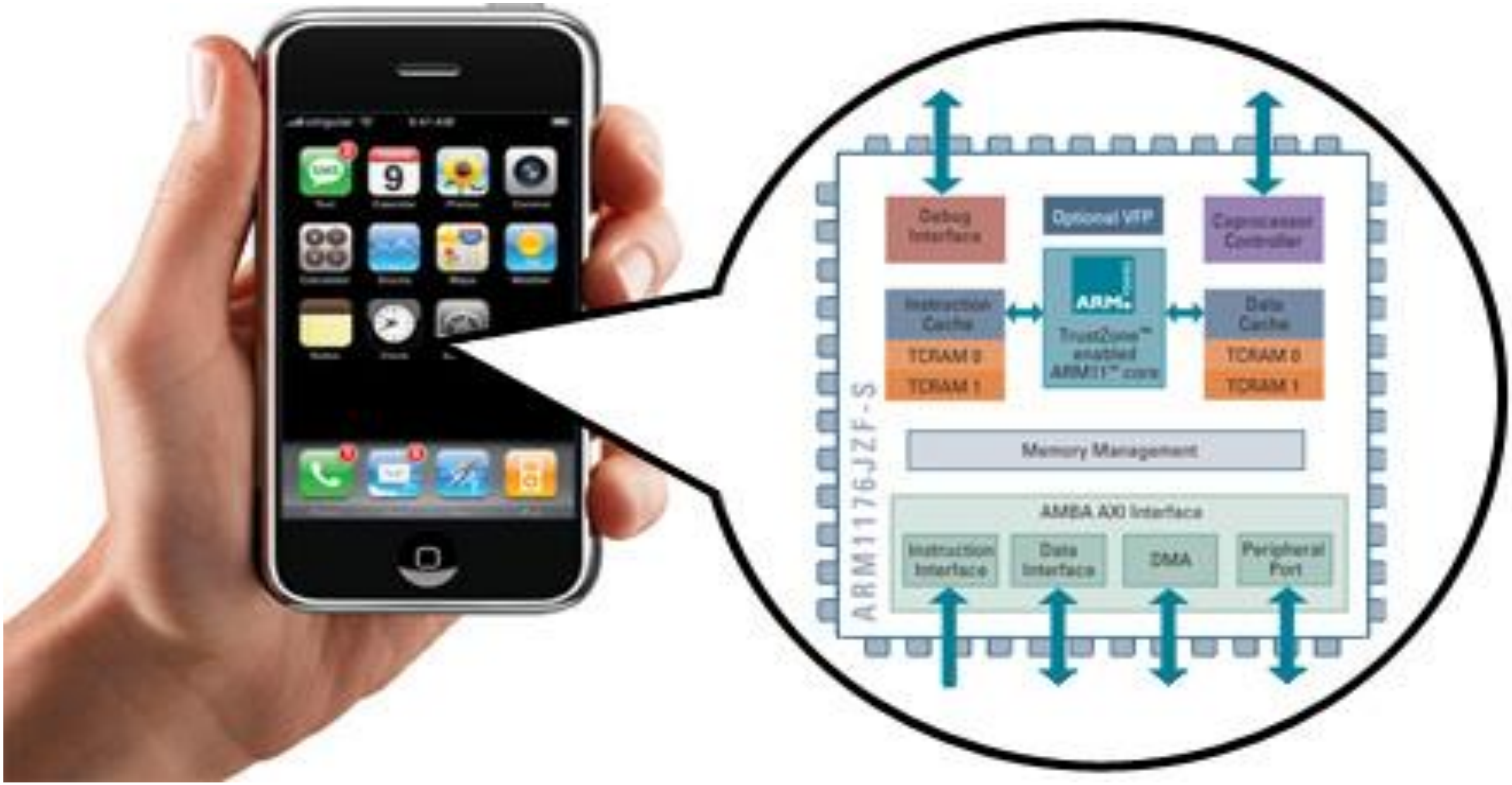
AMD's Barcelona Multicore Processor



- Four out-of-order cores on one chip
- 1.9 GHz clock rate
- 65nm technology
- Three levels of caches (L1, L2, L3) on chip
- Integrated Northbridge

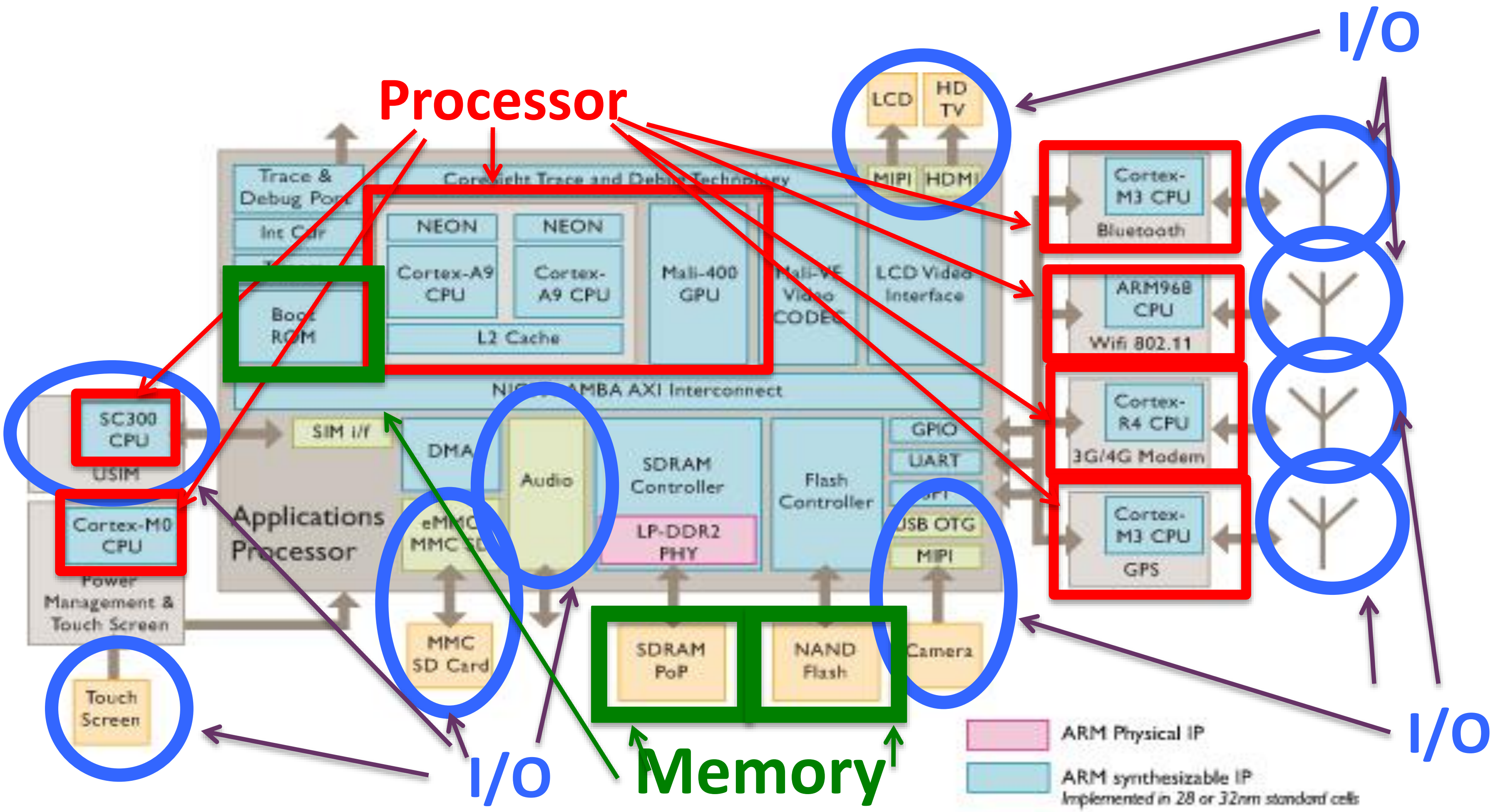
http://www.techwarelabs.com/reviews/processors/barcelona/

iPhone: has System-on-Chip



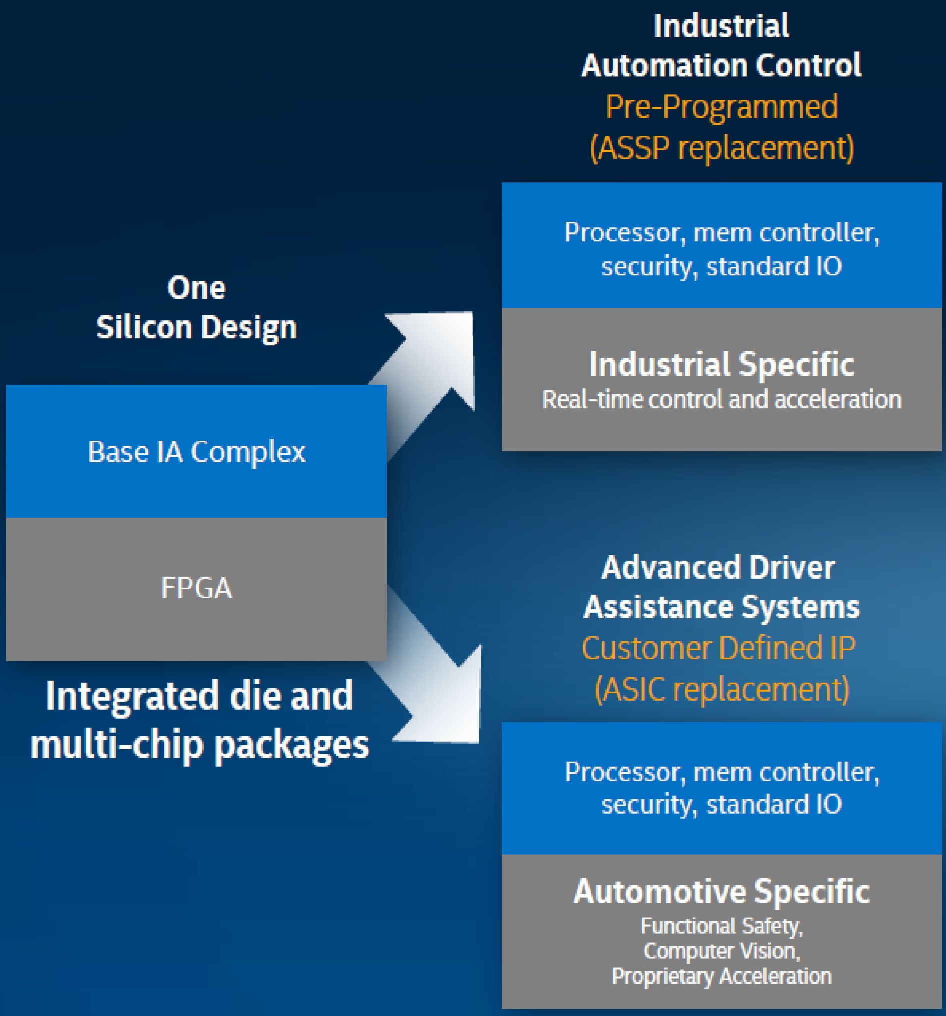
Source: UC Berkeley

iPhone inside



Source: UC Berkeley

Future: Internet of Things (IoT)



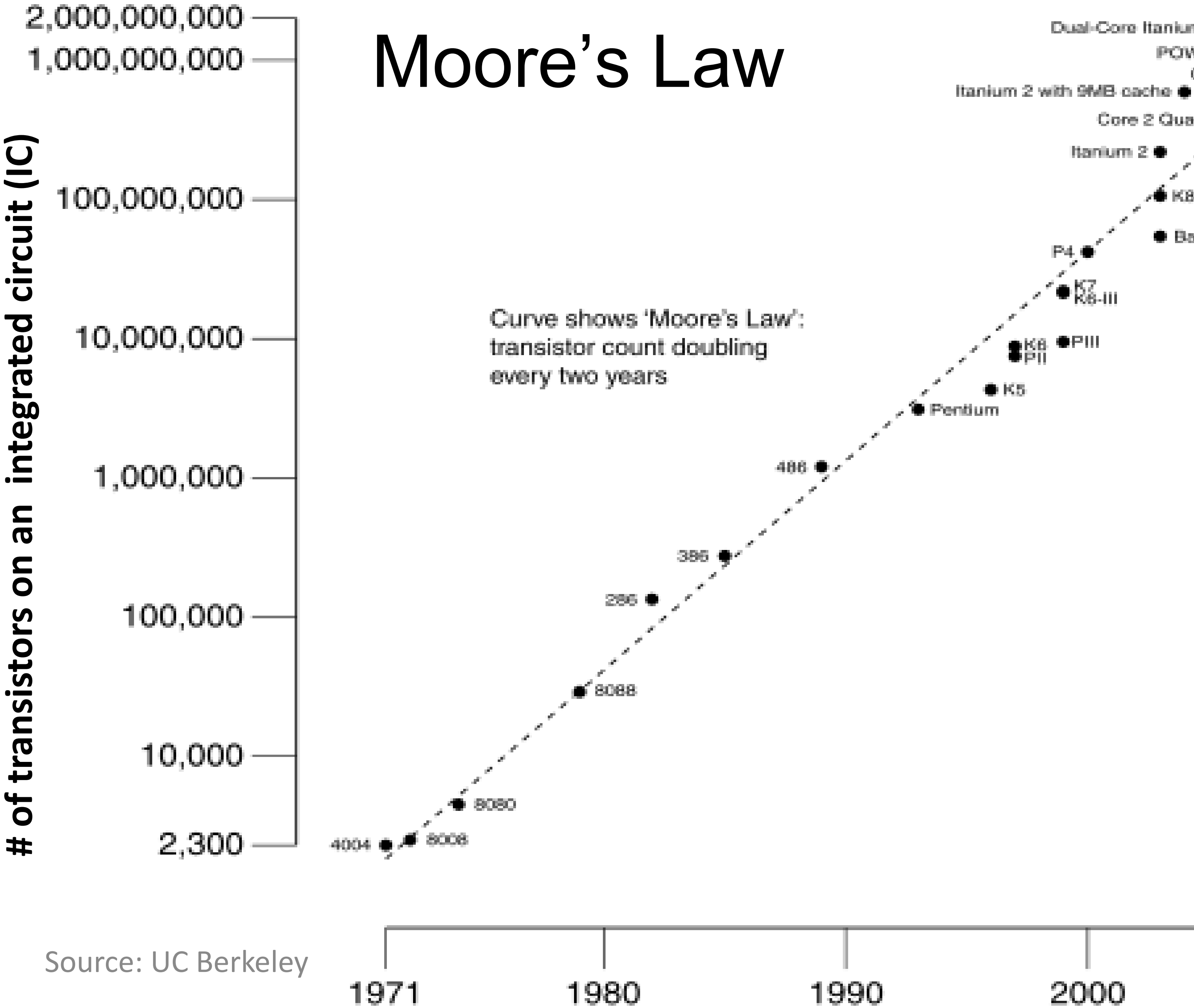
IoT Application Examples

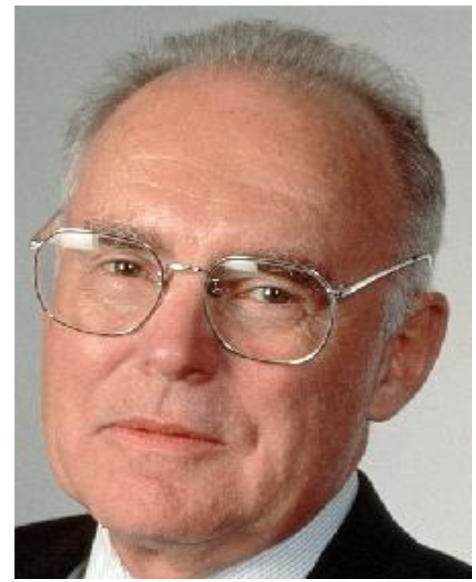
- Integrated solutions accelerate growth in key IoT segments by adding new functionality, improving performance, and lowering cost
- ~$11B incremental SAM by 2020 as integrated FPGAs become cost competitive with ASICs & ASSPs
- Customers can program their own IP, replacing ASICs
- Intel can pre-program industry-specific IP, replacing ASSPs
- Expected to reduce time-to-market by more than 50%

Source: Intel

Predicts: 2X Transistors / chip every 1.5 to 2 years

Moore's Law





Gordon Moore
Intel Cofounder

Source: UC Berkeley

wl 2018 2.11

Key idea: levels of representation/interpretation

High Level Language Program (e.g. Haskell, Java)

Compiler

Assembly Language Program (e.g. ARM or MIPS or x86)

Assembler

Machine Language Program (e.g. ARM or MIPS or x86)

Machine Interpretation

Hardware Architecture Description (e.g. block diagrams)

Architecture Implementation

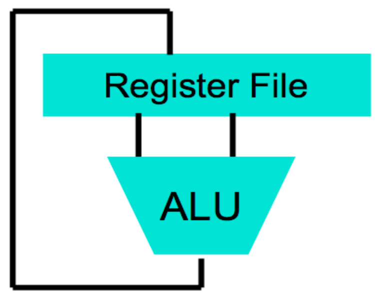
Logic Circuit Description (Circuit Schematic Diagrams)

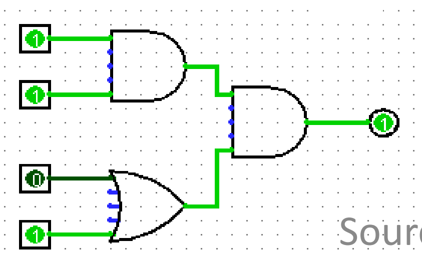
temp = v[k];
v[k] = v[k+1];
v[k+1] = temp;

```
lw   $t0, 0($2)
lw   $t1, 4($2)
sw   $t1, 0($2)
sw   $t0, 4($2)
```

Anything can be represented as a *number*, i.e. data or instructions

```
0000 1001 1100 0110 1010 1111 0101 1000
1010 1111 0101 1000 0000 1001 1100 0110
1100 0110 1010 1111 0101 1000 0000 1001
0101 1000 0000 1001 1100 0110 1010 1111
```





Source: UC Berkeley

Unsigned Binary Integers

- n-bit number

$$x = x_{n-1}2^{n-1} + x_{n-2}2^{n-2} + \cdots + x_1 2^1 + x_0 2^0$$

- range: 0 to $+2^n - 1$
- example
  - $0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 1011_2$
    $= 0 + \ldots + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$
    $= 0 + \ldots + 8 + 0 + 2 + 1 = 11_{10}$
- 32 bits
  - 0 to +4,294,967,295

Source: MKP

Two's-Complement Signed Integers

- n-bit number

$$x = -x_{n-1}2^{n-1} + x_{n-2}2^{n-2} + \cdots + x_1 2^1 + x_0 2^0$$

- range: -2^{n-1} to $+2^{n-1} - 1$
- example
  - $1111\ 1111\ 1111\ 1111\ 1111\ 1111\ 1111\ 1100_2$
    $= -1 \times 2^{31} + 1 \times 2^{30} + \ldots + 1 \times 2^2 + 0 \times 2^1 + 0 \times 2^0$
    $= -2{,}147{,}483{,}648 + 2{,}147{,}483{,}644 = -4_{10}$
- 32 bits
  - –2,147,483,648 to +2,147,483,647

Source: MKP

Two's-Complement Signed Integers

- bit 31 is sign bit
  - 1 for negative numbers
  - 0 for non-negative numbers
- $-(-2^{n-1})$ can't be represented
- non-negative numbers:
  - have the same unsigned and 2s-complement representation
- some specific numbers
  - 0: 0000 0000 … 0000
  - –1: 1111 1111 … 1111
  - most-negative: 1000 0000 … 0000
  - most-positive: 0111 1111 … 1111
- find out: sign & magnitude, excess-n representations

Source: MKP

Signed Negation

- complement and add 1
  - complement means 1 → 0, 0 → 1

$$x + \bar{x} = 1111...111_2 = -1$$

$$\bar{x} + 1 = -x$$

- example: negate +2
  - $+2 = 0000\ 0000\ \ldots\ 0010_2$
  - $-2 = 1111\ 1111\ \ldots\ 1101_2 + 1$
    $= 1111\ 1111\ \ldots\ 1110_2$

Source: MKP

Sign Extension

- representing a number using more bits
  - preserve the numeric value
- replicate the sign bit to the left
  - c.f. unsigned values: extend with 0s
- examples: 8-bit to 16-bit
  - +2: 0000 0010 => 0000 0000 0000 0010
  - –2: 1111 1110 => 1111 1111 1111 1110

Source: MKP

Hexadecimal

- base 16
  - compact representation of bit strings
  - 4 bits per hex digit

0	0000	4	0100	8	1000	c	1100
1	0001	5	0101	9	1001	d	1101
2	0010	6	0110	a	1010	e	1110
3	0011	7	0111	b	1011	f	1111

- example: eca8 6420
  - 1110 1100 1010 1000 0110 0100 0010 0000
- find out: Octal, Binary Coded Decimal (BCD) representations

Source: MKP

Character Data

- byte-encoded character sets
  - ASCII: 7-bit characters, 128 bit patterns
    - 95 graphic, 33 control
  - Latin-1: 256 characters
    - ASCII, +96 more graphic characters

- unicode: 32-bit character set
  - used in Java, C++ wide characters, …
  - most of the world's alphabets, plus symbols
  - UTF-8, UTF-16: variable-length encodings

Source: MKP

Think about

- How can I be sure that my bit-level design works?

- Correctness: with respect to the integer-level operation

- Example: to show bit-level negative, negbit, is correct, we need:
  - negbit :: [Bool] -> [Bool]   -- negbit is the bit-level design
  - negint :: Int   -> Int   -- negint n = -n is the integer-level operation
  - bit2int :: [Bool] -> Int   -- bit2int converts bit-level data to integers

  - To show: bit2int . negbit = negint . bit2int   ("." is function composition)
  - e.g. if negbit is correct, then negbit [F,F,T,T] = [T,T,F,T]   (T=True, F=False)
    LHS: bit2int (negbit [F,F,T,T]) = bit2int [T,T,F,T] = -3
    RHS: negint (bit2int [F,F,T,T]) = negint (3) = -3