

Introduction

EE685, Fall 2026

Hank Dietz

<http://aggregate.org/hankd/>

Course Overview

- Sort-of like CPE380/480, but broader & deeper
- There will be *Verilog*, and you'll design stuff
- A lot more of the *advanced* stuff
 - Fancy things inside processors
 - Lots of memory & parallel architecture
- Higher-level models and/or simulation

```

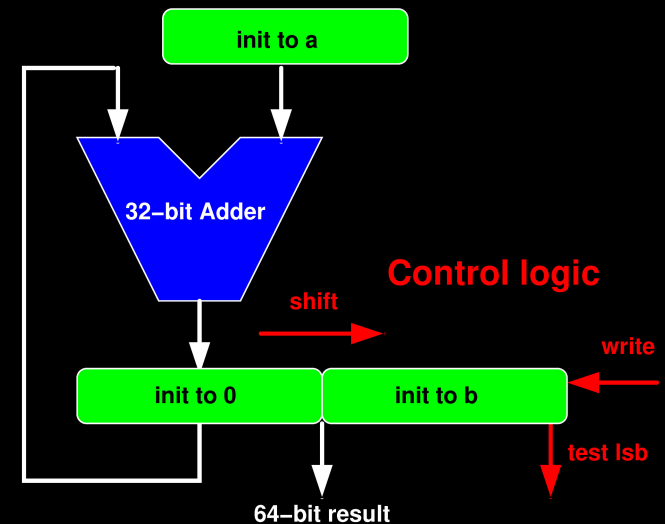
module mul(ready, c, a, b, reset, clk);

parameter BITS = 32;
input [BITS-1:0] a, b;
input reset, clk;
output reg [BITS*2-1:0] c;
output reg ready;
reg [BITS-1:0] d;
reg [BITS-1:0] state;
reg [BITS:0] sum;

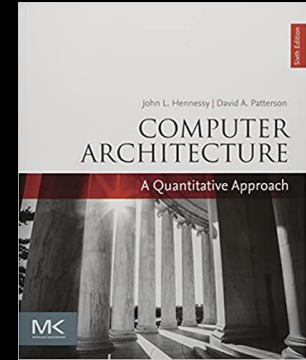
always @(posedge clk or posedge reset) begin
    if (reset) begin
        ready <= 0;
        state <= 1;
        d <= a;
        c <= {{BITS{1'b0}}, b};
    end else begin
        if (state) begin
            sum = c[BITS*2-1:BITS] + d;
            c <= (c[0] ? {sum, c[BITS-1:1]} :
                (c >> 1));
            state <= {state[BITS-2:0], 1'b0};
        end else begin
            ready <= 1;
        end
    end
end
endmodule

```

Verilog 32-bit Multiplier



Textbook



- The text is:
*Computer Architecture, 6th Edition:
A Quantitative Approach,*
by Patterson & Hennessy
- Same text as was used in CPE480...
and *optionally* referenced here
- Lots of additional materials...

Grading & Such

- Four assignments, ~10% each
- Midterm exam, ~20%
- Final exam, ~40%
- Material cited from the **text**, from **lectures**, from **canvas**, or from the **course URL**:
<https://aggregate.org/EE685/>
- You are expected to regularly attend class
- I try not to curve much; always in your favor

Course Content

- Precise content depends on you:
 - How many of you took CPE380? When?
 - How many of you took CPE480? When?

This course is sort-of ++CPE480...

- The syllabus is posted, but it is a little vague because things will vary significantly based on the backgrounds of students in the course

Course Content

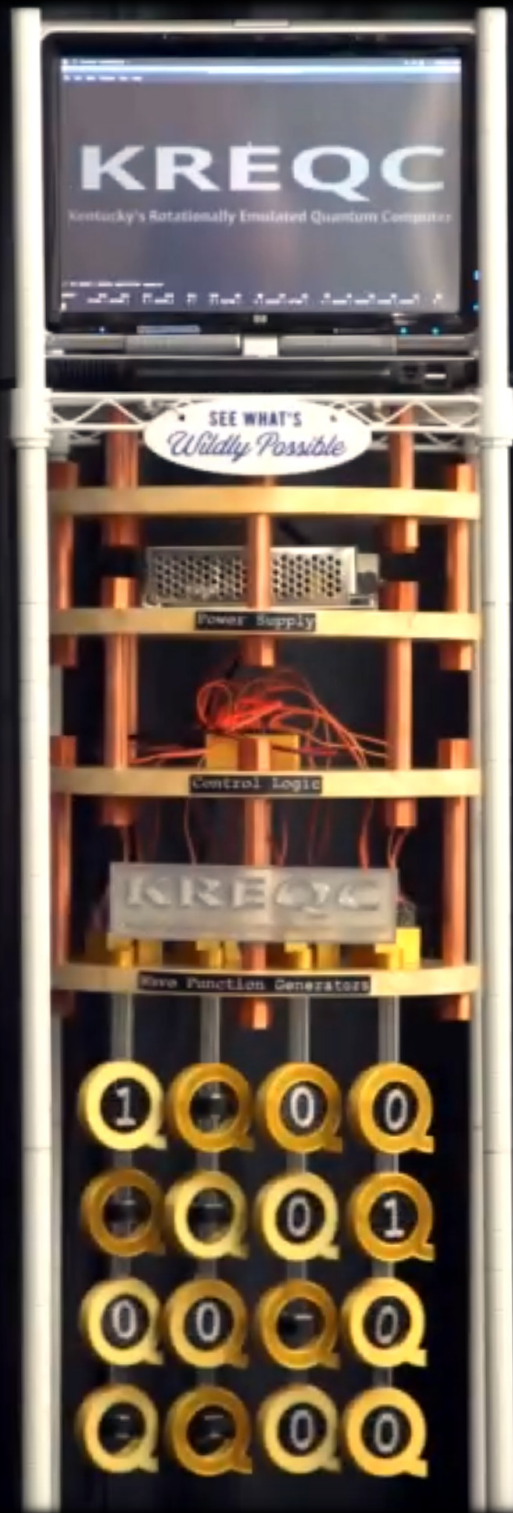
Lectures	Topic
1	Introduction
3	Verilog (project)
6	Pipelined RISC machine (project)
3	Instruction-level parallelism
1	Review for midterm
1	In-class midterm exam
3	Advanced processor internals (assignment)
5	Memory hierarchy and protection (assignment)
4	Scalable parallel processing and networking
1	Review for final exam

Me (and why I'm biased)

- **Hank Dietz**, ECE Professor and James F. Hardymon Chair in Networking
- Office: **203 Marksbury**
- Research in parallel compilers & architectures:
 - Built 1st Linux PC cluster supercomputer
 - Antlr, AFNs, SWAR, FNNs, MOG, ...
 - Various awards & world records for best price/performance in supercomputing
- Lab: **108/108A Marksbury** – I have **TOYS!**

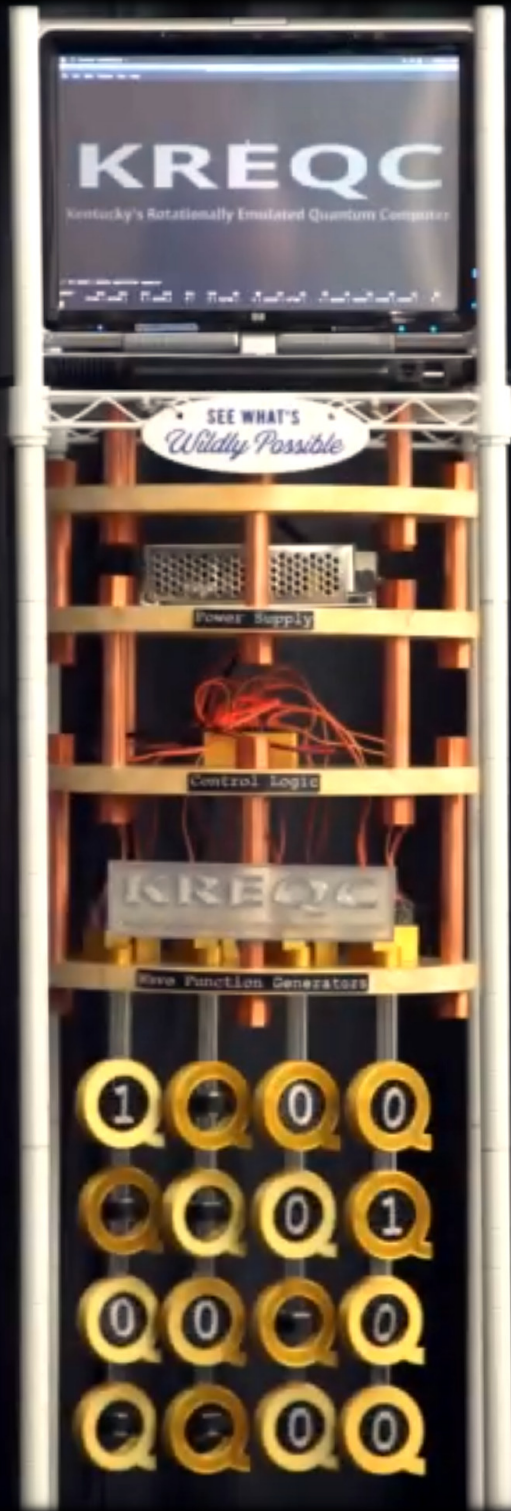
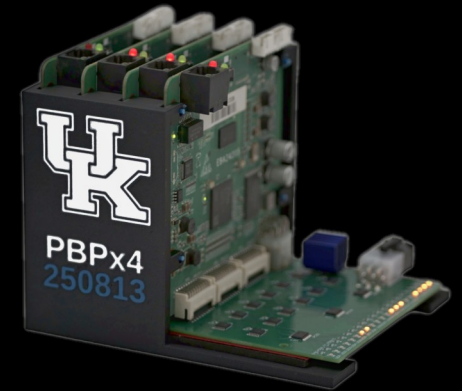


Electrical & Computer Engineering



**This cluster was
Finally retired in
Summer 2025!**

Electrical & Computer Engineering



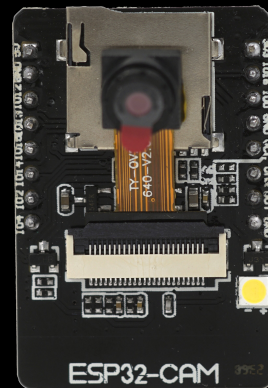


El Capitan supercomputer:
11,039,616 cores, 2.82 Exaflop/s,
1.8 Exaflops/s Linpack, 29.7 MW power



LineShine supercomputer (Chinese tech, no GPUs):
13,789,440 cores, 2.735 Exaflop/s,
2.2 Exaflop/s Linpack, 42.22 MW power

What is advanced stuff?

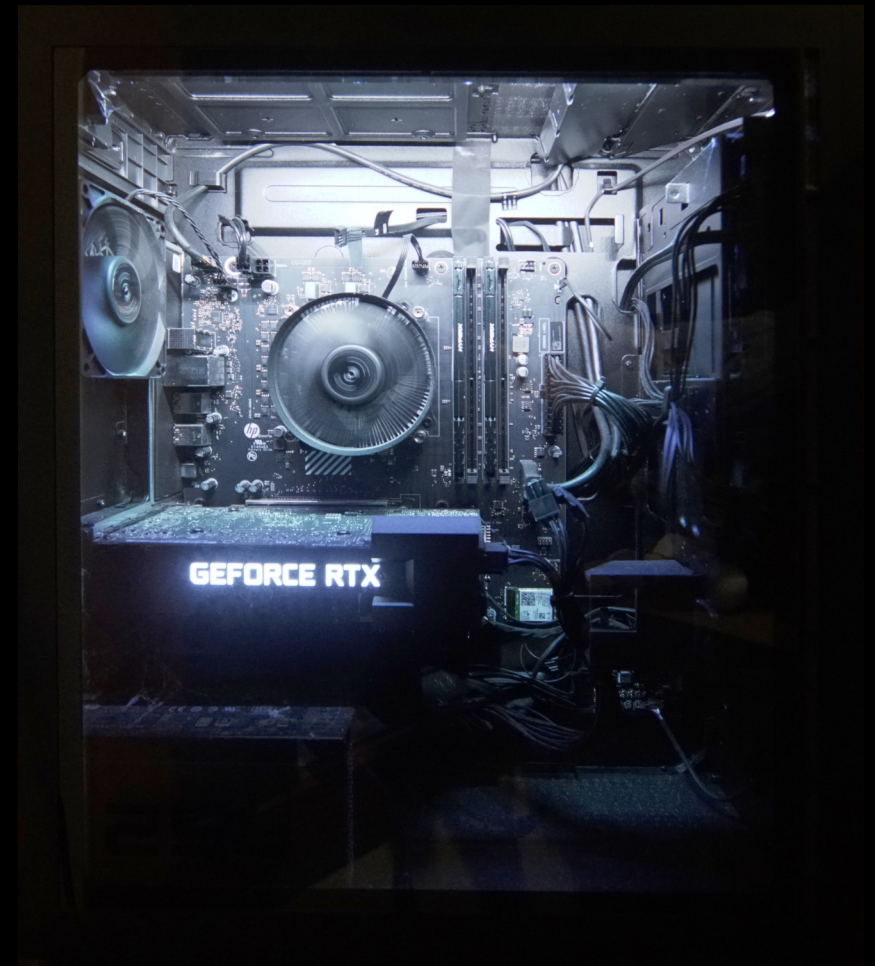
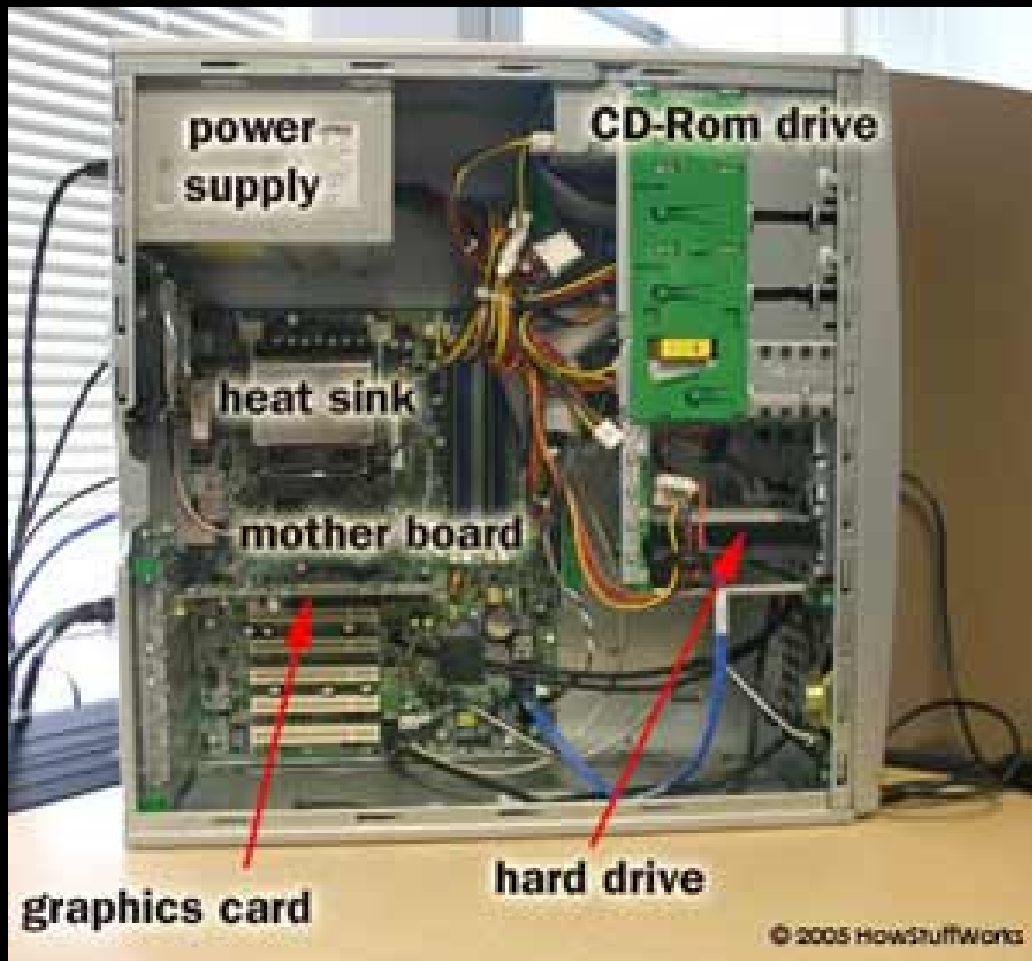


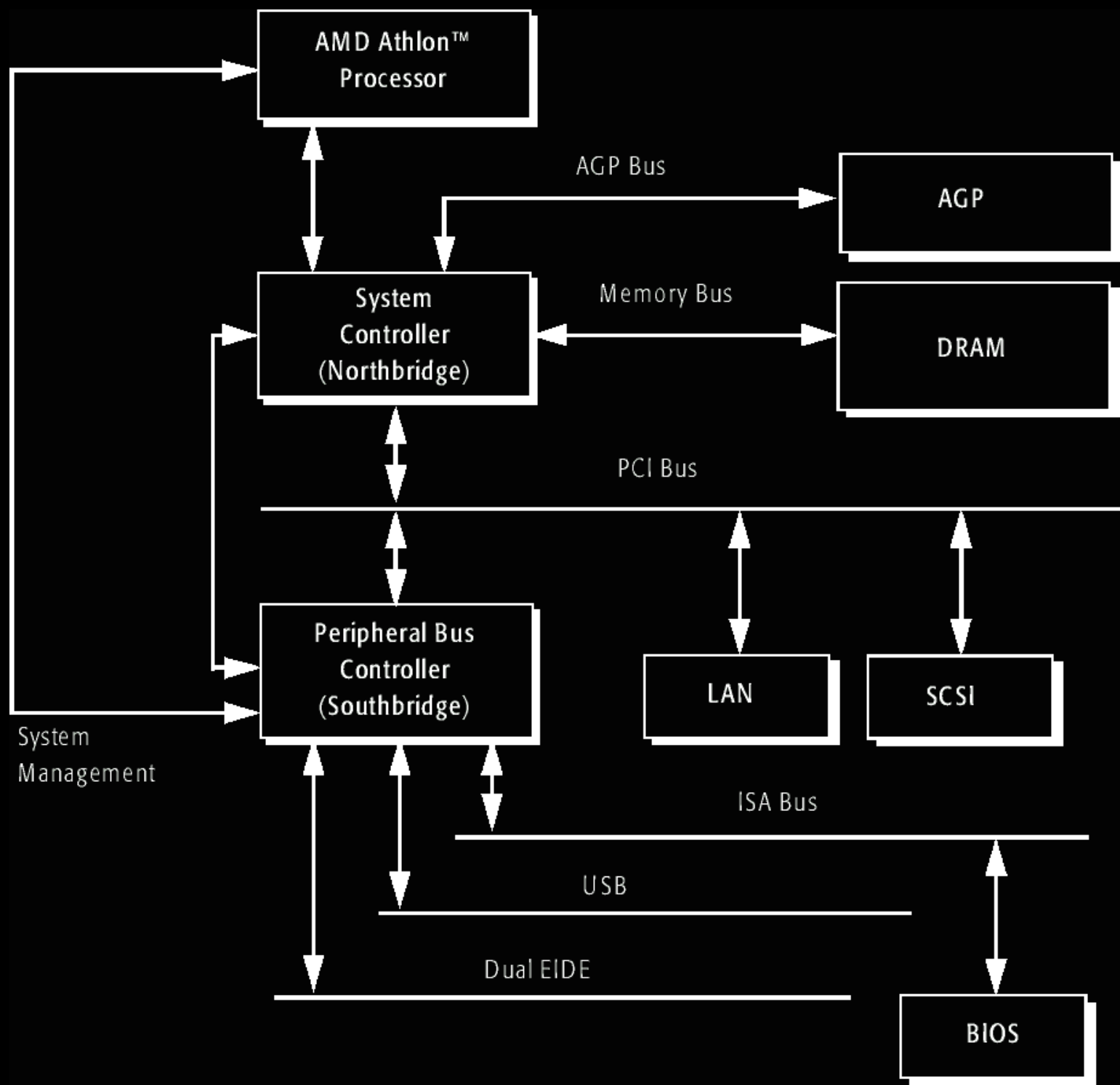
ESP32-CAM microcontroller:

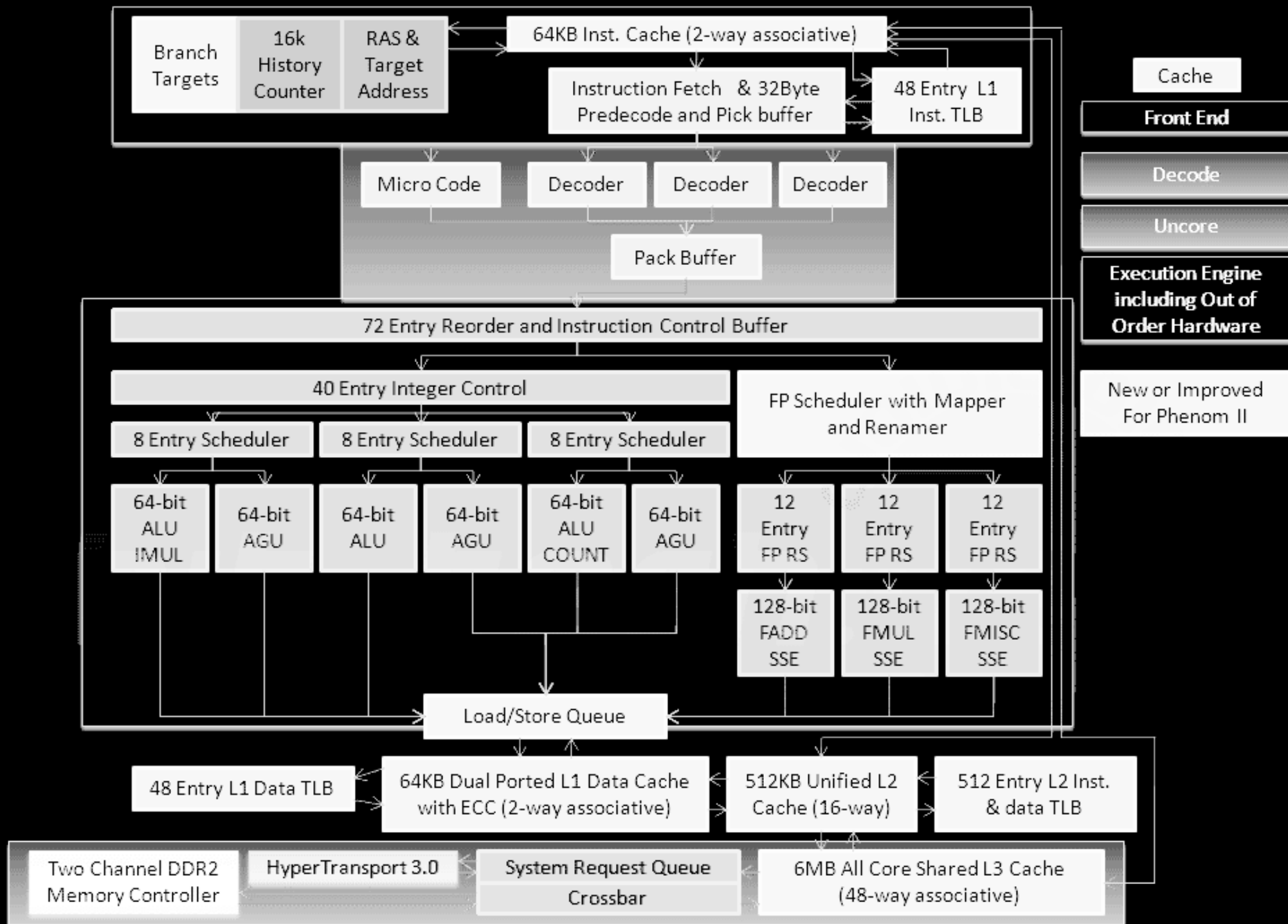
2+ULP cores, 802.11+Bluetooth, TF, 2MP camera

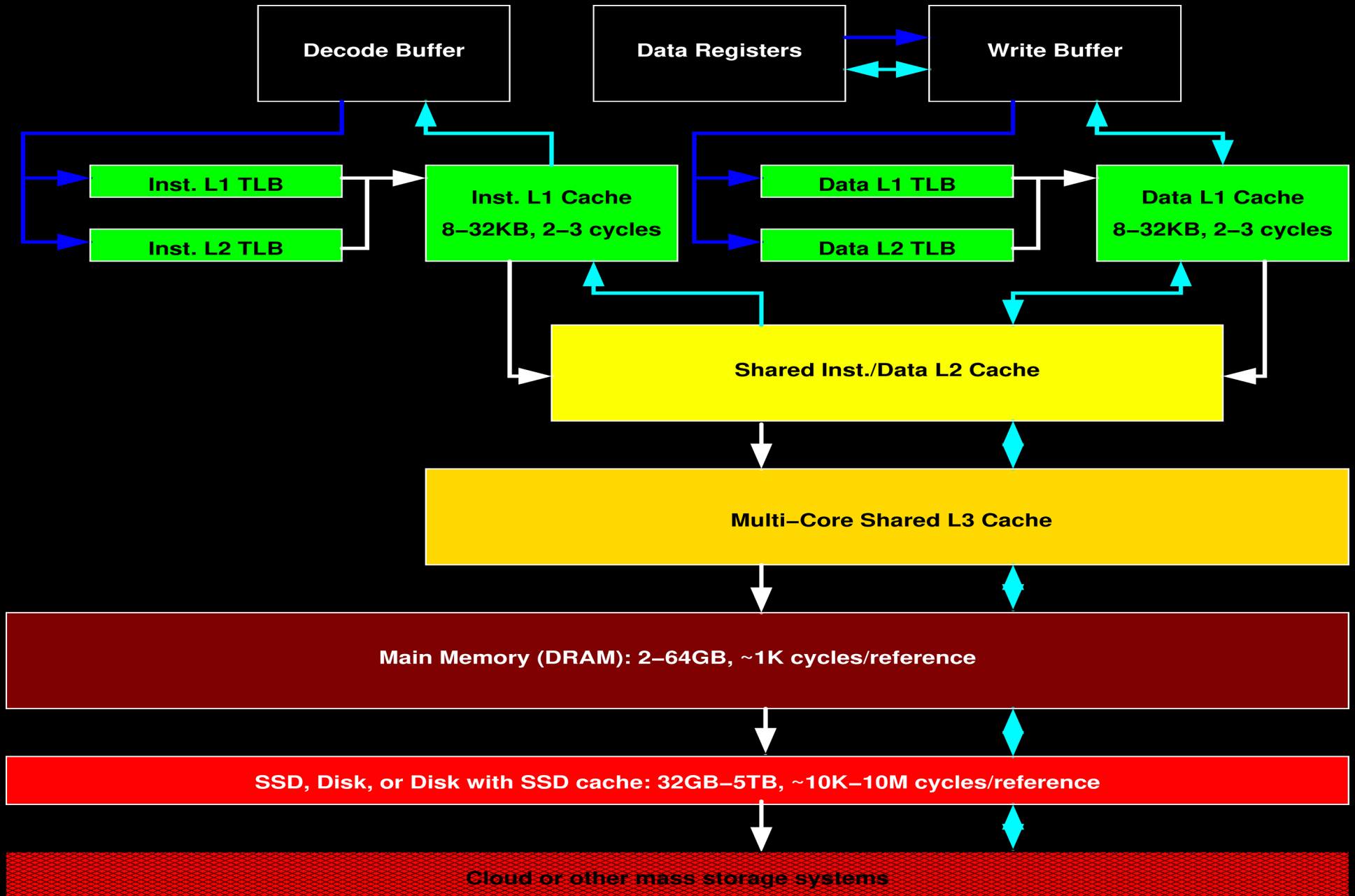
Cost approx. \$7, 5 μ A sleep current

What's inside a PC?



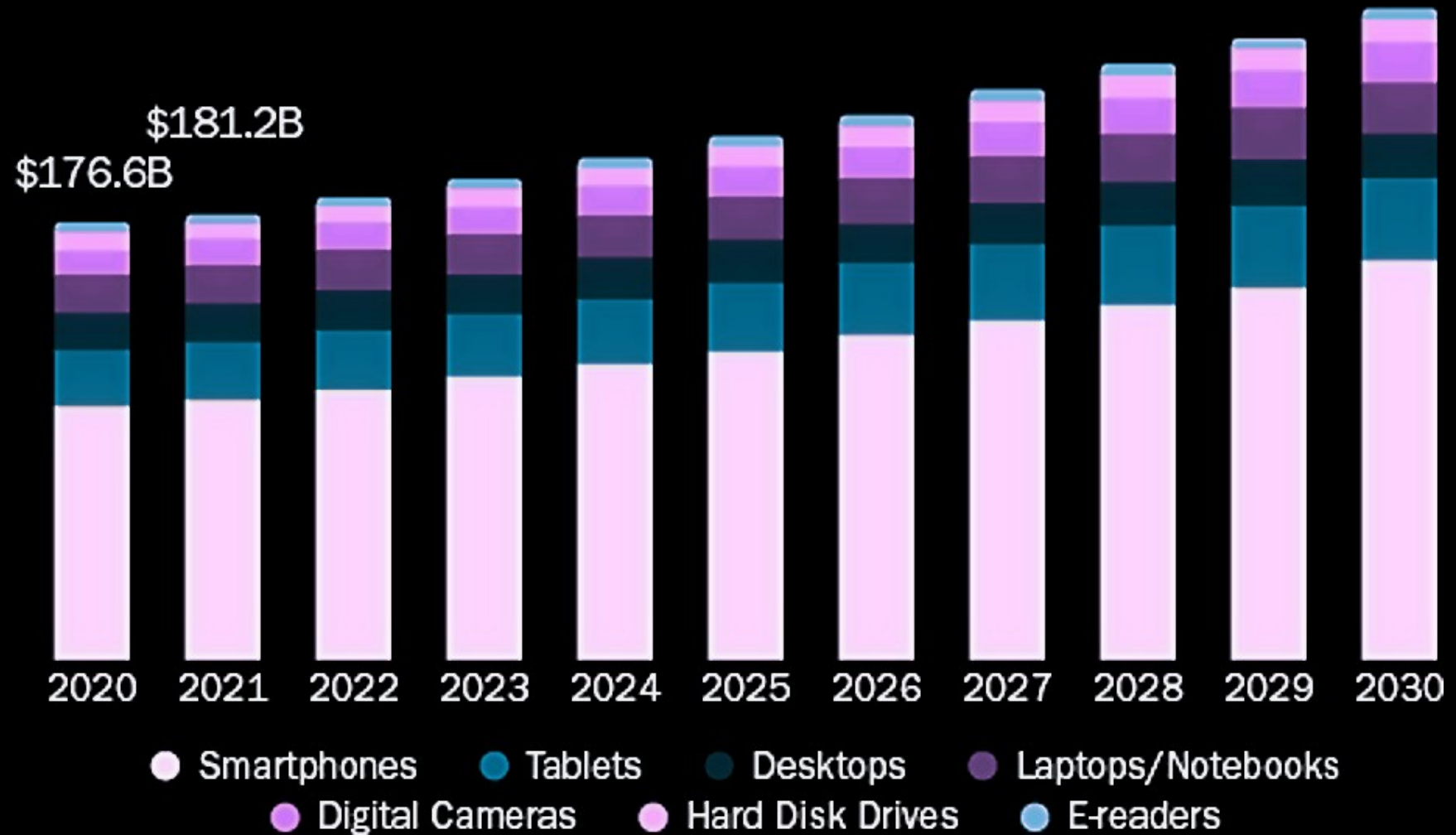




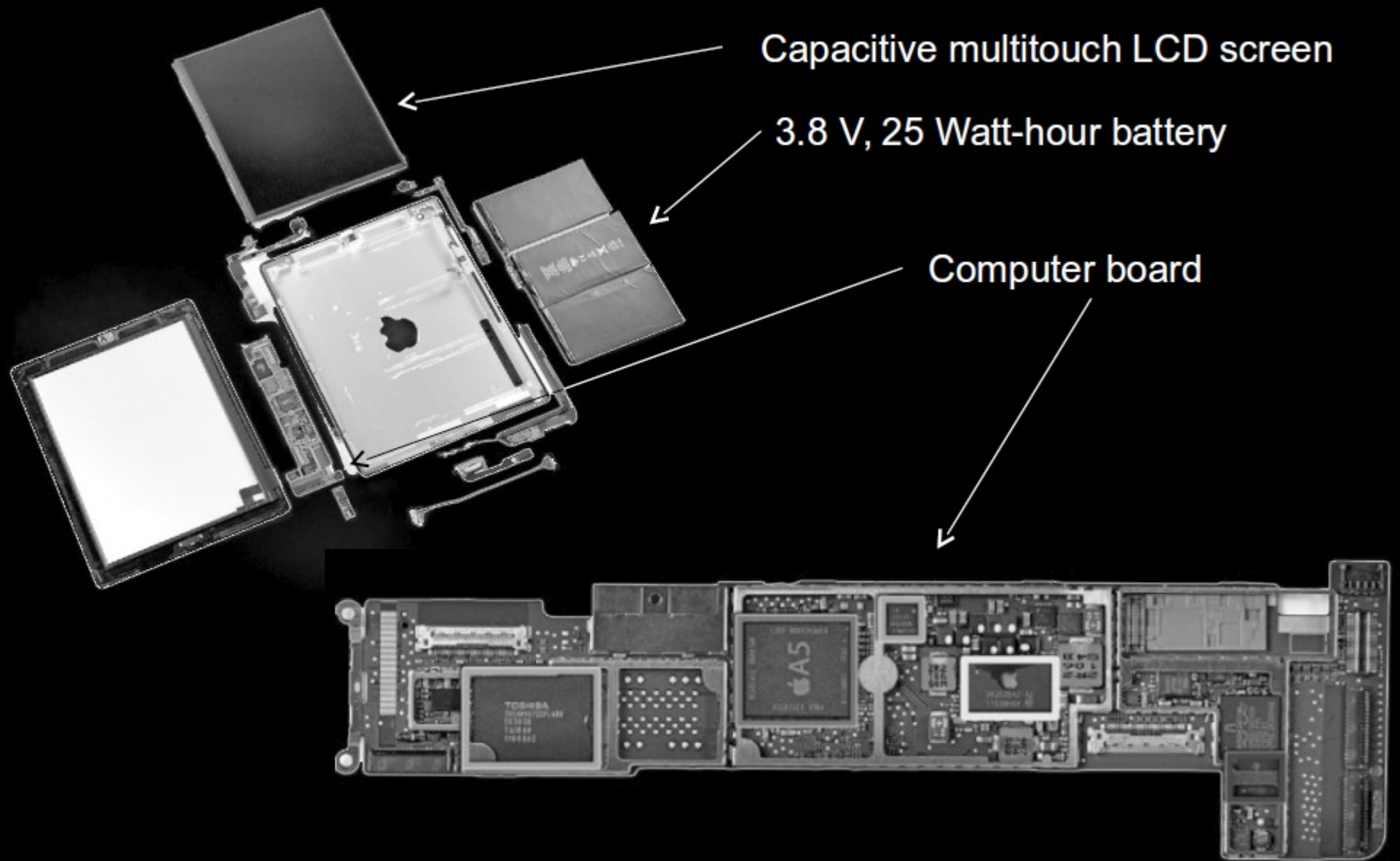


U.S. Consumer Electronics Market

Size, by Product, 2020 - 2030 (USD Billion)

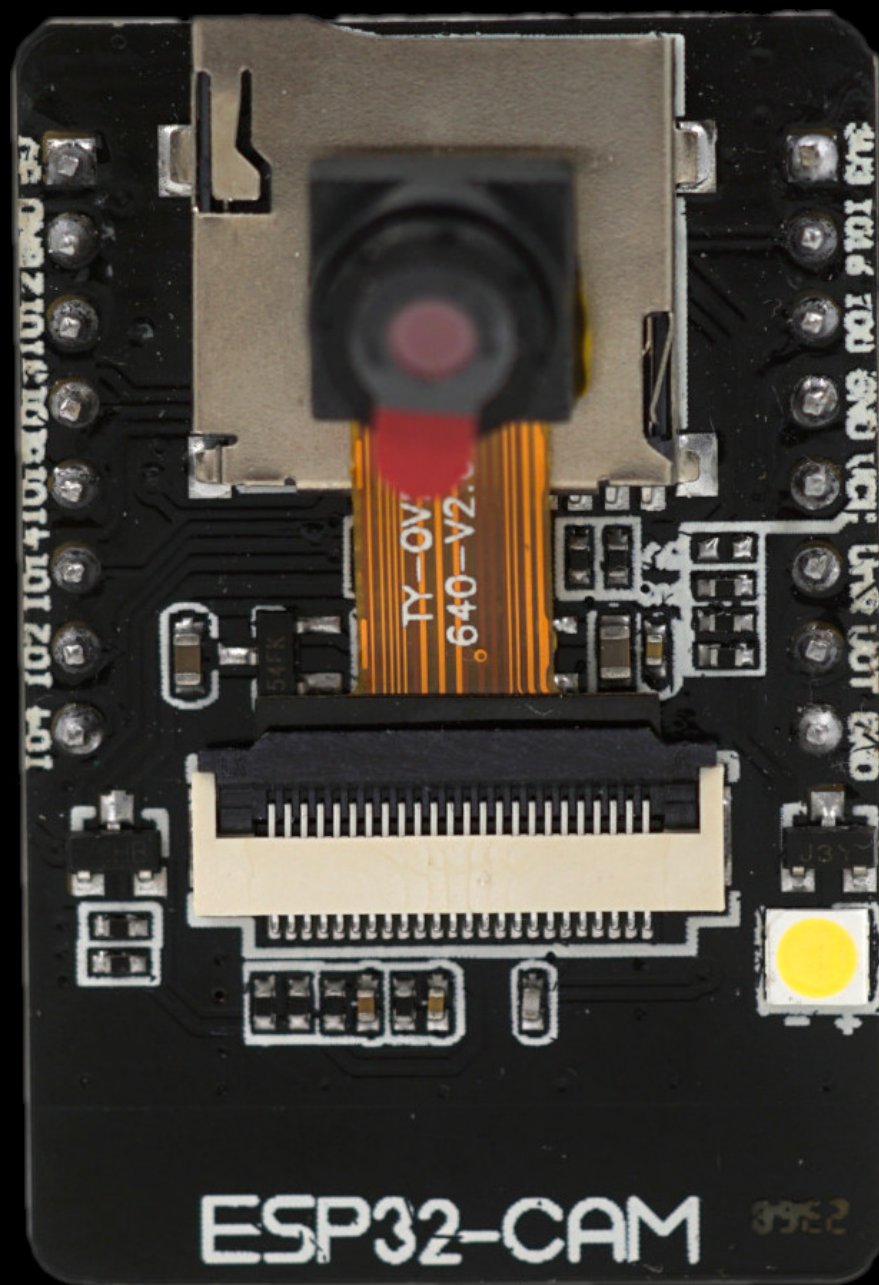








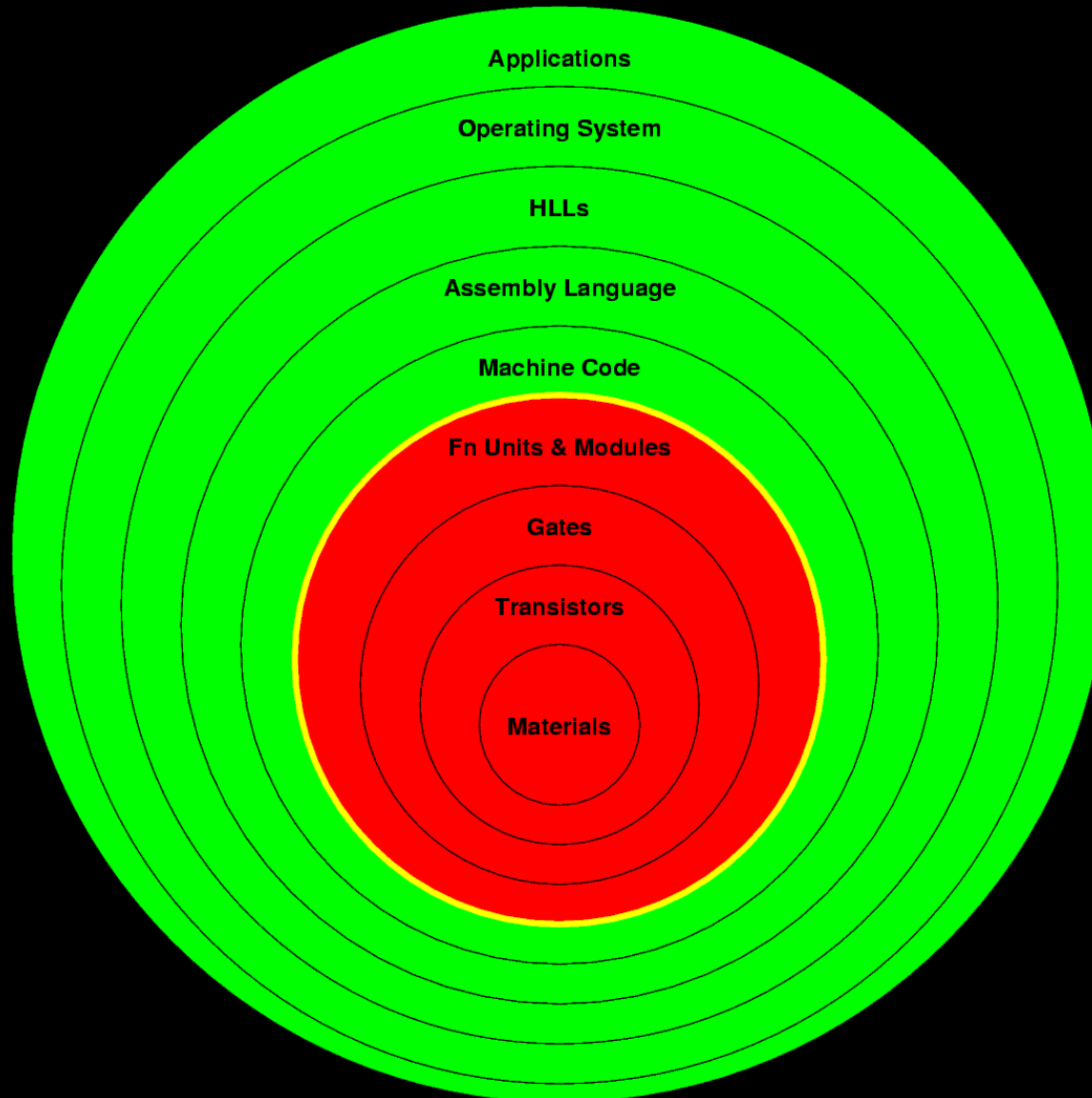




Complexity is Increasing!

- Lots of things you use every day have **BILLIONS** of components!
- You don't live long enough to know it all

Abstraction “Onion”



Software Layers

- Applications...
- Operating Systems (OS)...
- High-Level Languages (HLLs)
Aka, High Order Languages (HOLs)
 - Designed for humans to write & read
 - Modularity
 - Abstract data types, type checking
 - Assignment statements
 - Control constructs
 - I/O statements

Instruction Set Architecture

- ISA defines HW/SW interface
- **Assembly Language**
 - Operations match hardware abilities
 - Relatively simple & limited operations
 - Mnemonic (human readable?)
- **Machine Language**
 - Bit patterns – 0s and 1s
 - Actually executed by the hardware

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 MIPS)

```
swap:
    muli $2, $5, 4
    add  $2, $4, $2
    lw   $15, 0($2)
    lw   $16, 4($2)
    sw   $16, 0($2)
    sw   $15, 4($2)
    jr   $31
```

Assembler

Binary machine
language
program
(for MIPS)

```
000000001010000100000000000011000
000000000000110000001100000100001
100011000110001000000000000000000
100011001111001000000000000000100
101011001111001000000000000000000
101011000110001000000000000000100
00000011111000000000000000001000
```

Hardware Layers

- Function-block organization (CPE380 stuff)
- Gates & Digital Logic (CPE282 stuff)
- Transistors
 - Used as bi-level (saturated) devices
 - Amplifiers, not just on/off switches
- Materials & Integrated Circuits
 - Implementation of transistors, etc.
 - Analog properties

Who Does What?

- Instruction Set Design, by *Architect*
 - Machine & Assembly Languages
 - “**Computer Architecture**”
 - **Instruction Set Architecture** / Processor
- Computer Hardware Design, by *Engineer*
 - Logic Design & Machine Implementation
 - “**Processor Architecture**”
 - “**Computer Organization**”

How does EE685 see this?

- A lot like CPE380/CPE480...
- Like CPE380, Verilog runs through it
- Focus is on the hardware/software interface
 - Models that guide design and use
 - Modules that enable or boost performance

8 Great Ideas

- Design for Moore's Law
- Abstraction
- Make the common case fast
- Pipelining
- Parallelism
- Prediction
- Hierarchy of memories
- Dependability via redundancy



SI Terminology Of Scale

1000^1	kilo	k	1000^{-1}	milli	m
1000^2	mega	M	1000^{-2}	micro	u
1000^3	giga	G	1000^{-3}	nano	n
1000^4	tera	T	1000^{-4}	pico	p
1000^5	peta	P	1000^{-5}	femto	f
1000^6	exa	E			

- 1000^x vs. 1024^x
- 1 **Byte** (**B**) is 8-10 bits (**b**), 4 bits in a **Nybble**
- Hertz (**Hz**) is frequency (vs. period)

Main topics for EE685

- Start with Verilog
- Review pipelined RISC (where CPE380 ends)
- Add instruction-level parallelism (SWAR, VLIW, SuperScalar...)
- Add advanced processor internals (floating point, interrupts)
- Add memory hierarchy (caches, virtualization)
- Add scalable parallelism (SIMD/GPU, MIMD, networking)