Introduction

ECE 355

Microprocessor-Based Systems

Daler N. Rakhmatov
daler@ece.uvic.ca

Welcome to **ECE 355**…

- Course administration
  - Office hours: Tuesdays, 12:30 – 14:00, EOW 327
  - Lectures: TWF, 10:30 – 11:20, HSD A240
  - Labs (check your section schedule):

Mon	16:30 – 19:20	ELW B328	B01
Wed	13:30 – 16:20	ELW B328	B03
Thu	16:30 – 19:20	ELW B328	B07/08
Fri	13:30 – 16:20	ELW B328	B05/06

  - Grading

Homework	5%	
Lab	30%	
Midterm	20%	**October 25**
Final	45%	

Welcome to **ECE 355**…

- Reading
  - Hamacher/Vranesic/Zaky/Manjikian, ***Computer Organization and Embedded Systems***, 6/E © 2011 McGraw-Hill
  - ECE 355 lecture notes and lab materials
  - On-line references (see course website)

- Course website
  - www.ece.uvic.ca/~daler/courses/ece355
  - Notes, assignments, solutions, links, etc…

- ***You are responsible for all the announcements made in class!***

ECE 355 Objectives

- Develop a general understanding of the operation, design, application, and programming of typical microprocessor-based systems
  - Hardware-software interplay and tradeoffs
  - System integration and interfacing issues
- Learn specific details using the 32-bit **Cortex-M0**®
architecture as a demonstration platform
  - Hardware-software interfacing
  - I/O programming
- Ultimate goal:
  - Be able to apply studied concepts to any advanced embedded system

ECE 355 Topics

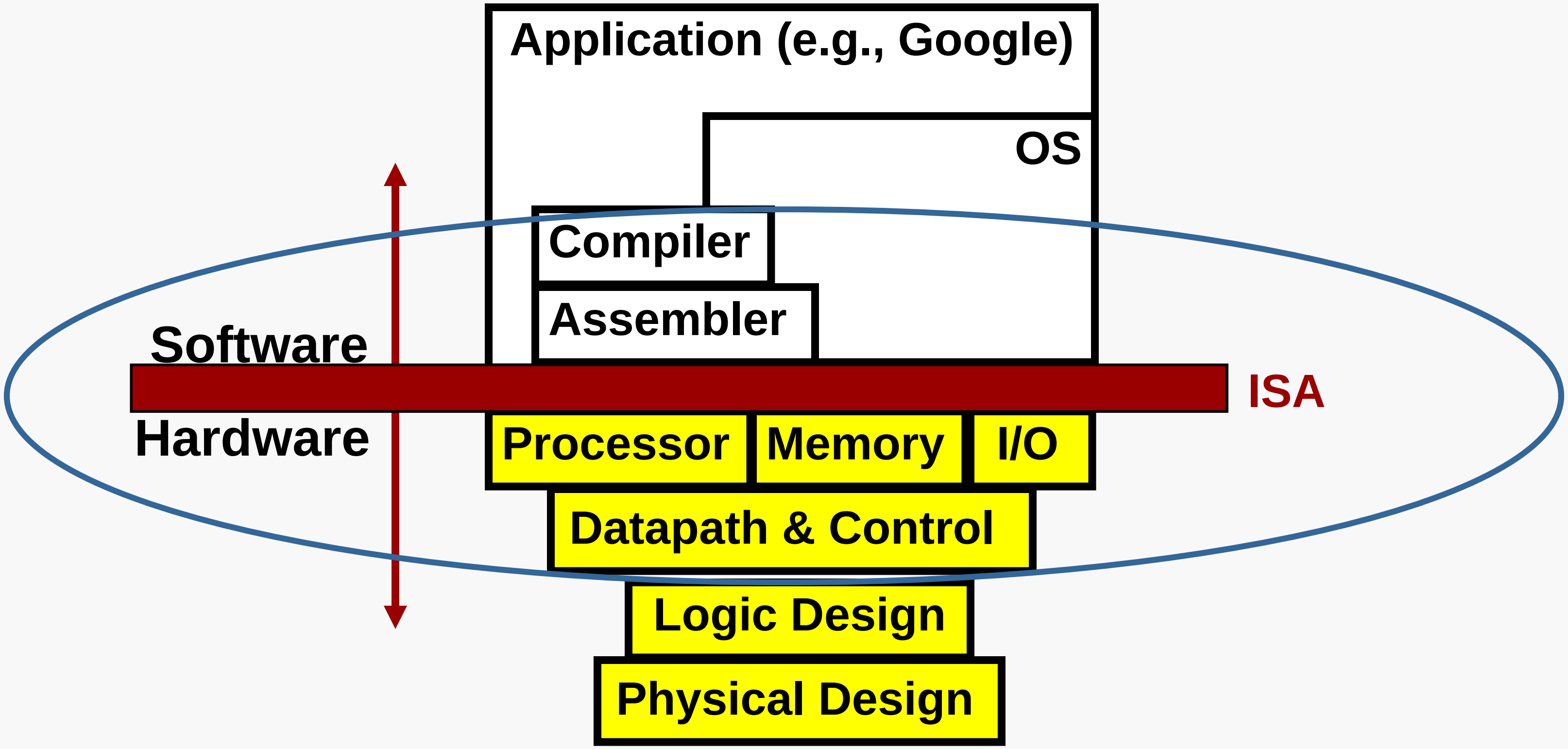
- Embedded systems
  - Applications, technologies, trends
- Computer organization
  - Microprocessors
  - Memory hierarchy
  - I/O interfacing and control
  - Internal and external communication
- Embedded software
  - C programming
  - System and application software

Acknowledgements

- Lecture notes are modified educational materials originally developed by:

Textbook authors	Toronto and Queen’s
David Patterson	Berkeley
John Hennessy	Stanford
Randal Bryant	CMU
S.P. Dandamudi	Carleton
Frank Vahid	Riverside
James Armstrong	Virginia Tech
Ronald Hoelzeman	Pittsburgh
Charles Kime	Wisconsin
Donald Givone	SUNY-Buffalo
Vincent Heuring	Colorado
Etc…	

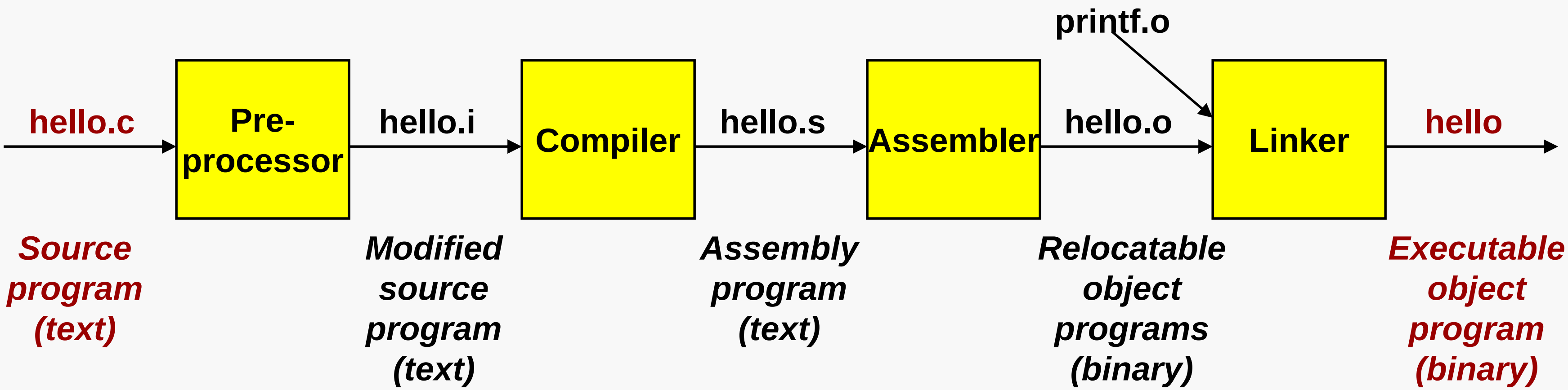
Instruction Set Architecture (ISA)



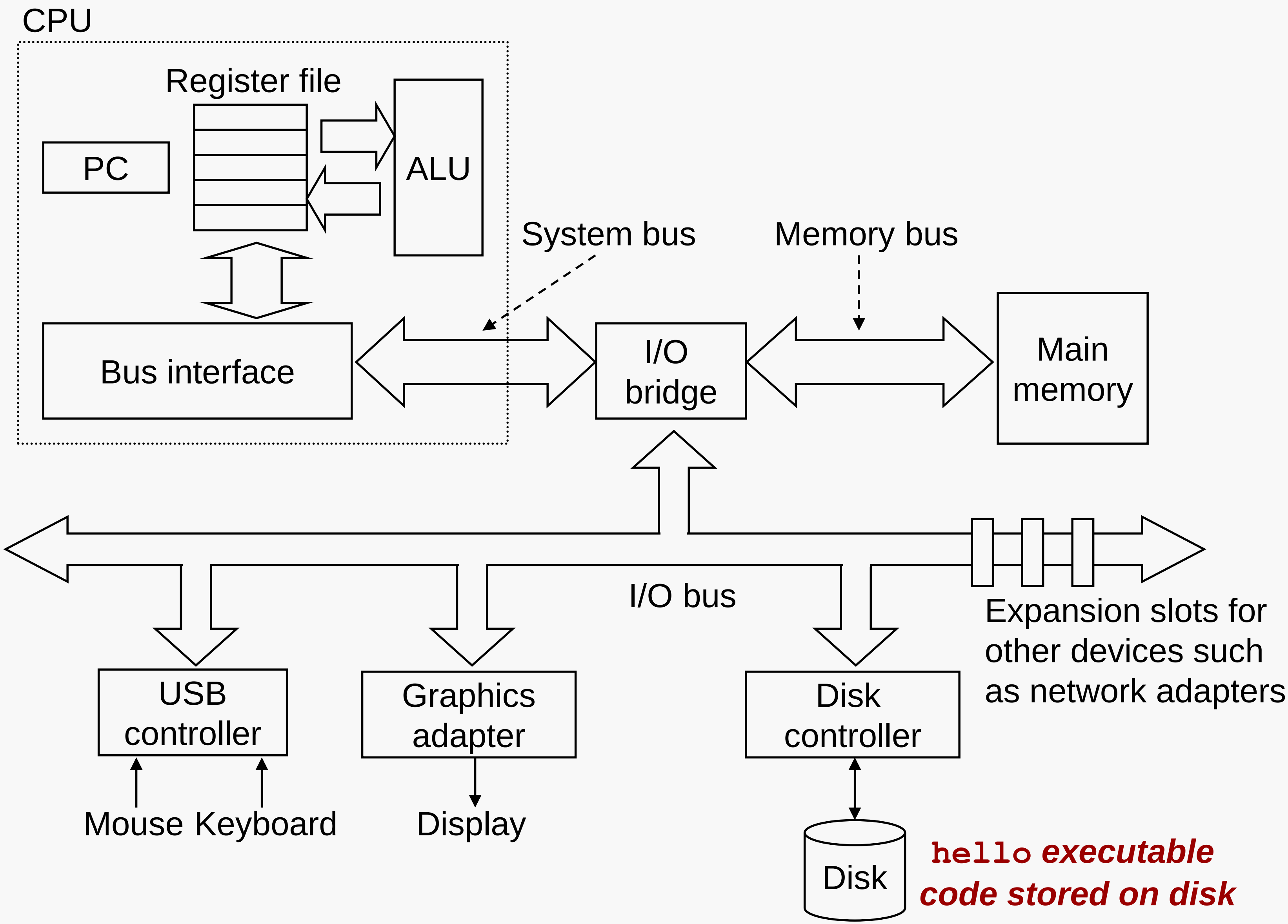
C Program: hello.c

```
#include <stdio.h>

int main() {
     printf("Hello World");
     return 0;
}
```

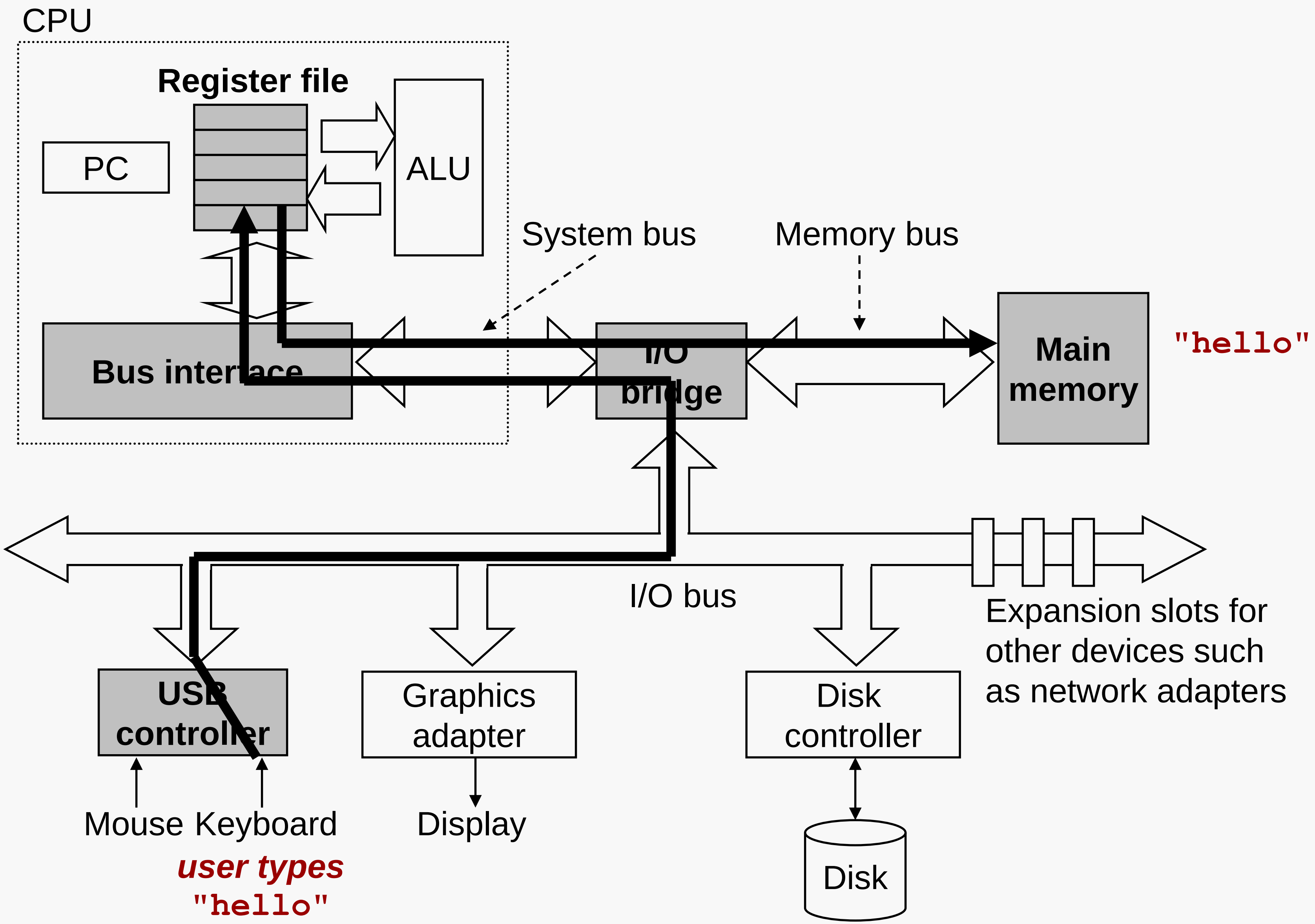


General-Purpose Computer

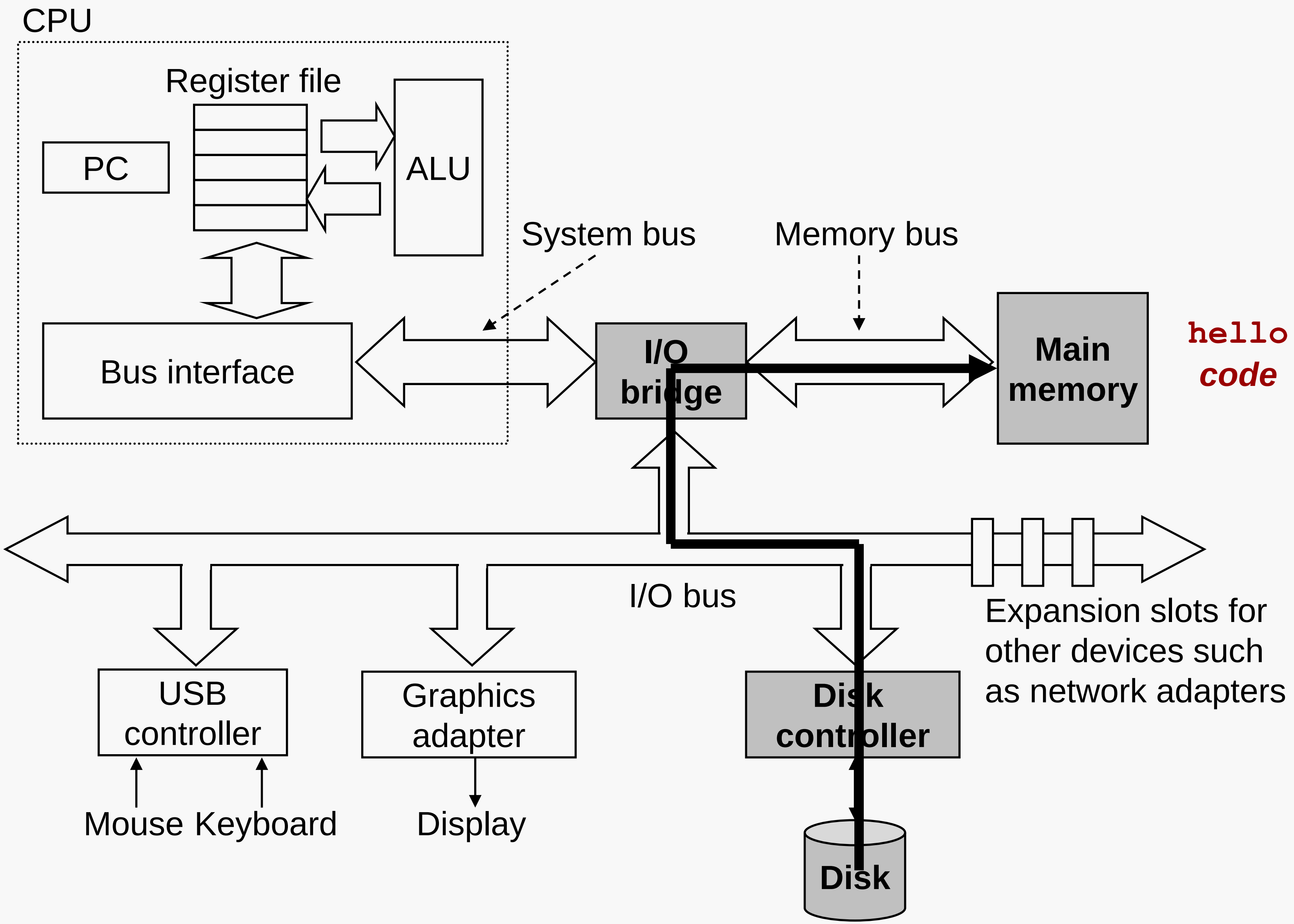


`hello` executable code stored on disk

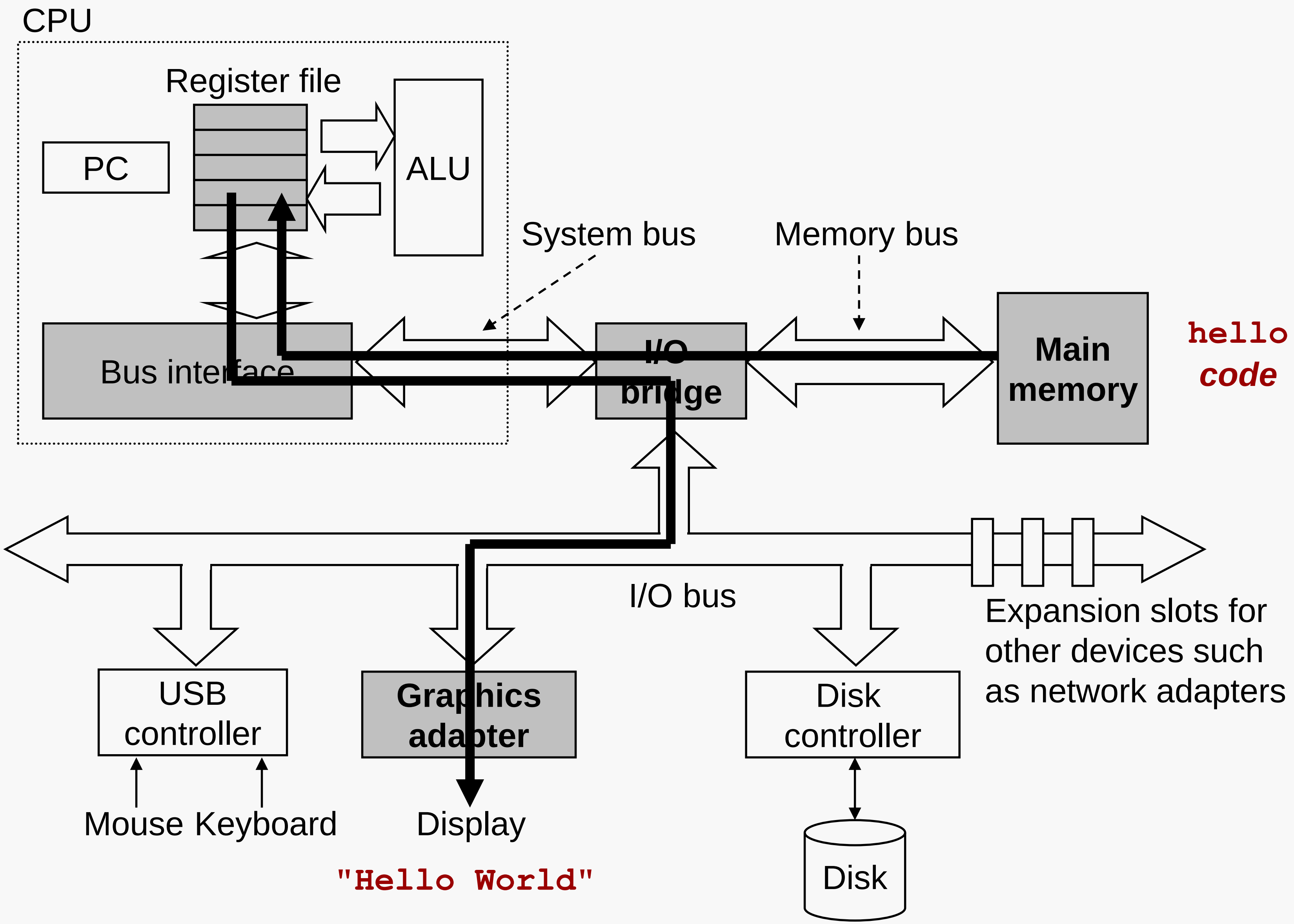
Reading Input



Loading Executable



Execution, Writing Output

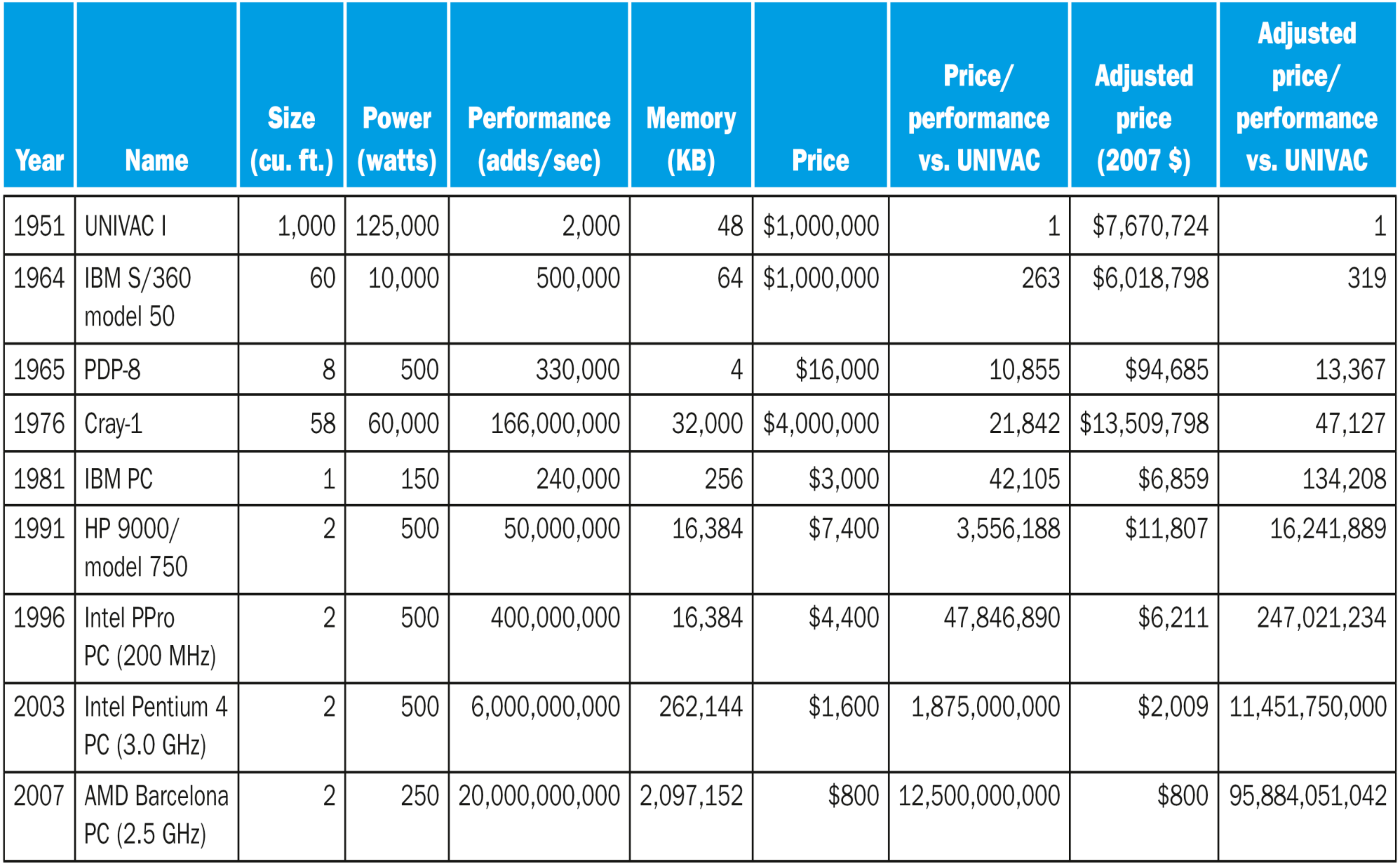


I/O Devices

- *"Computer without I/O is like a car without wheels – great technology, but won't get you anywhere"*
- Typical I/O devices:

Device	Behavior	Partner	Rate (Mbit/s)
Keyboard	Input	Human	0.0001
Mouse	Input	Human	0.0038
Printer	Output	Machine	3.2000
Optical Disk	Storage	Machine	80.0000
Network/LAN	In/Out	Machine	100-1000
Display	Output	Human	800-8000

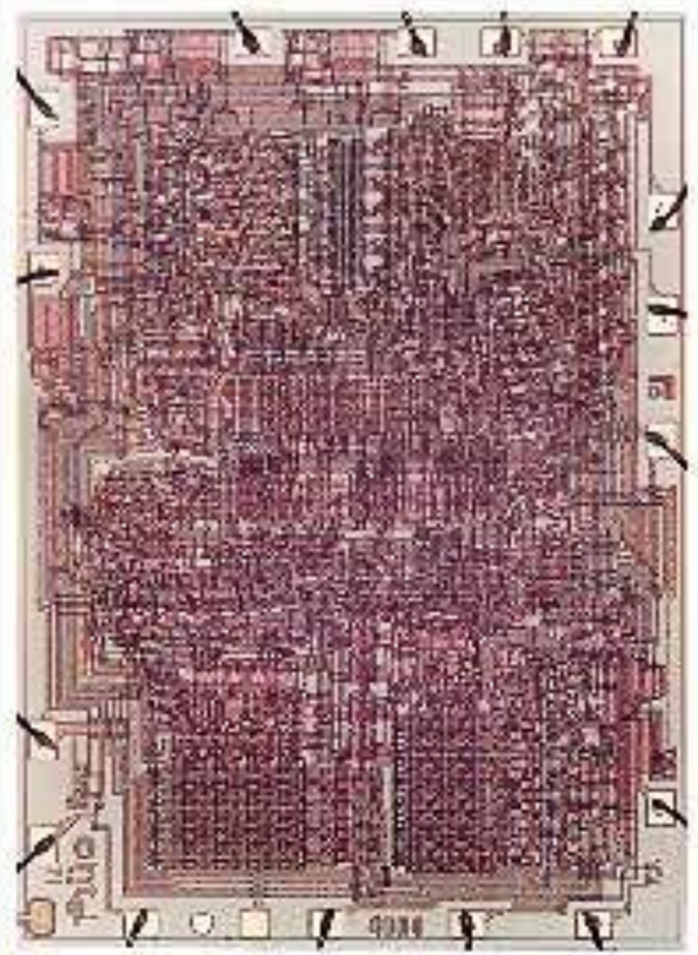
Historical Perspective

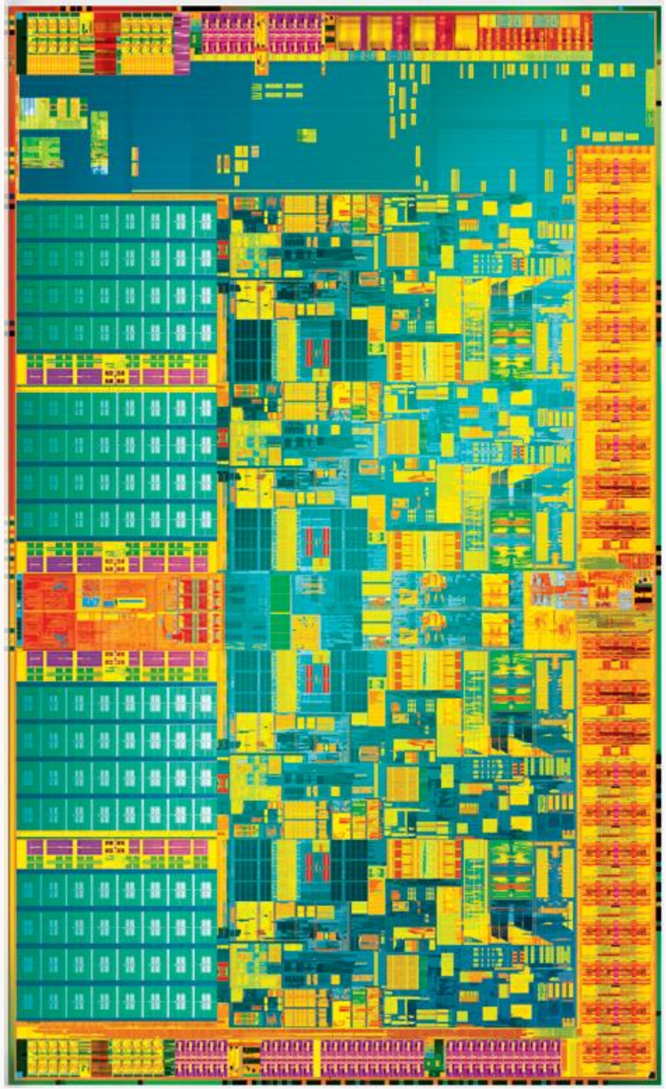


Year	Name	Size (cu. ft.)	Power (watts)	Performance (adds/sec)	Memory (KB)	Price	Price/ performance vs. UNIVAC	Adjusted price (2007 $)	Adjusted price/ performance vs. UNIVAC
1951	UNIVAC I	1,000	125,000	2,000	48	$1,000,000	1	$7,670,724	1
1964	IBM S/360 model 50	60	10,000	500,000	64	$1,000,000	263	$6,018,798	319
1965	PDP-8	8	500	330,000	4	$16,000	10,855	$94,685	13,367
1976	Cray-1	58	60,000	166,000,000	32,000	$4,000,000	21,842	$13,509,798	47,127
1981	IBM PC	1	150	240,000	256	$3,000	42,105	$6,859	134,208
1991	HP 9000/ model 750	2	500	50,000,000	16,384	$7,400	3,556,188	$11,807	16,241,889
1996	Intel PPro PC (200 MHz)	2	500	400,000,000	16,384	$4,400	47,846,890	$6,211	247,021,234
2003	Intel Pentium 4 PC (3.0 GHz)	2	500	6,000,000,000	262,144	$1,600	1,875,000,000	$2,009	11,451,750,000
2007	AMD Barcelona PC (2.5 GHz)	2	250	20,000,000,000	2,097,152	$800	12,500,000,000	$800	95,884,051,042

Microprocessor Evolution (Intel)

i4004 (1971): 10 μm, 750 KHz, 0.5 W, 2.3 KTx



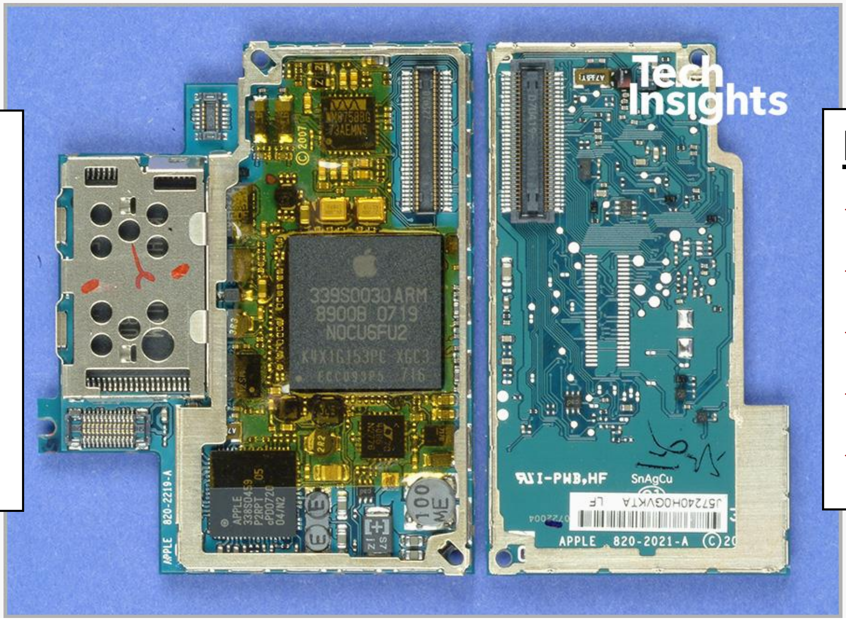


i7 (2008): 45 nm, 3.3 GHz, 130 W, 731 MTx

Embedded Systems (ES)

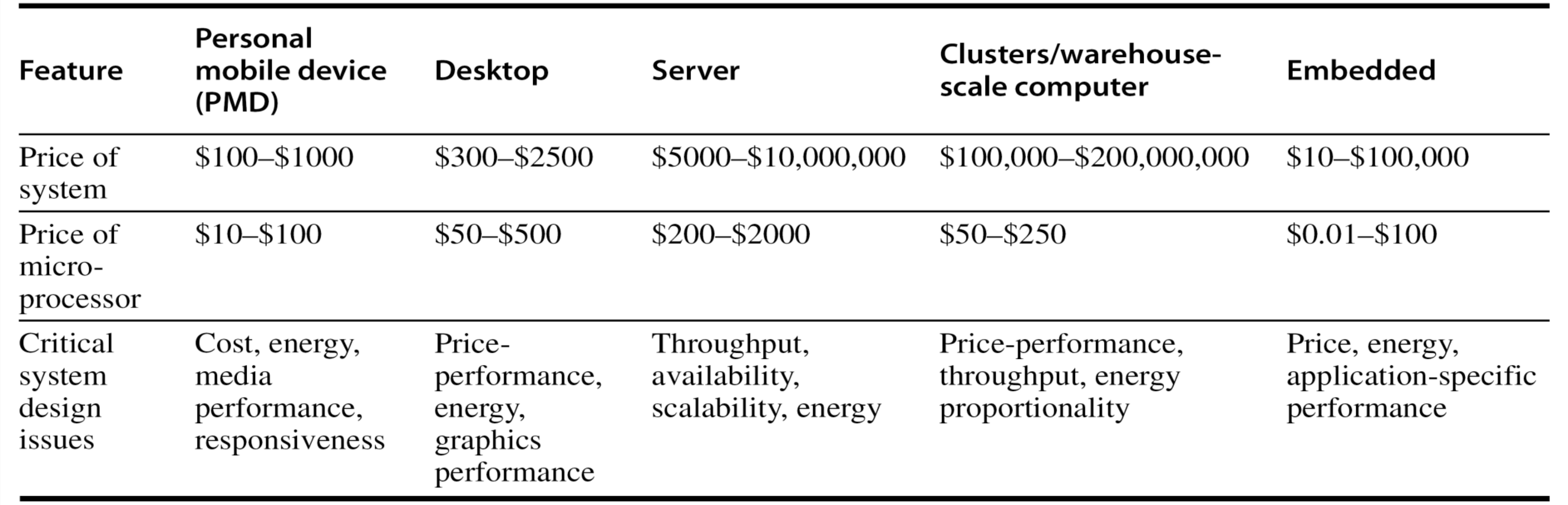
Markets:

- ✓ Consumer electronics
- ✓ Industrial automation
- ✓ Telecommunications
- ✓ Automotive/avionics
- ✓ Medical equipment



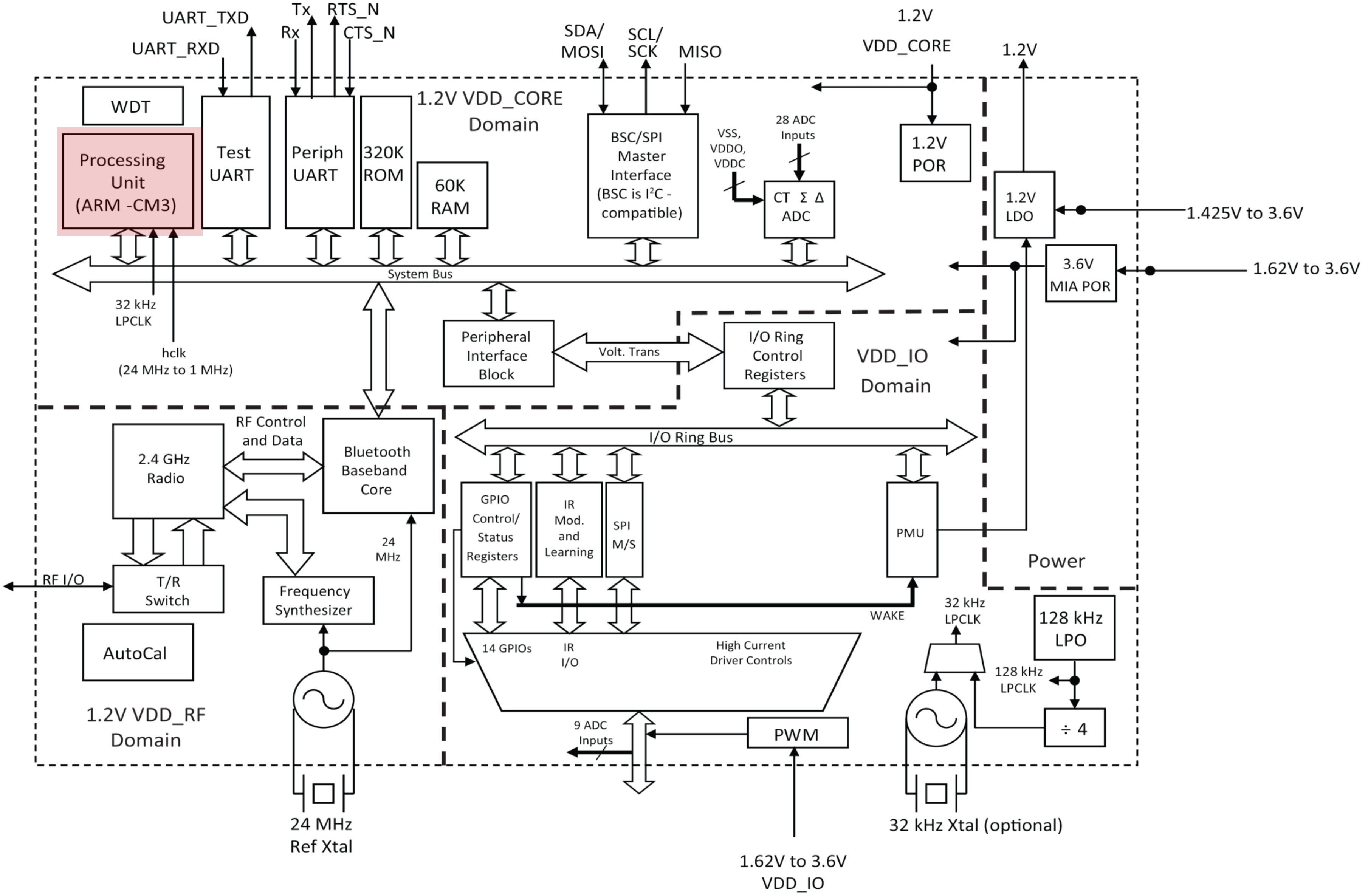
Features:

- ✓ Tightly constrained
- ✓ Tightly integrated
- ✓ Function-specific
- ✓ Real-time
- ✓ Reactive

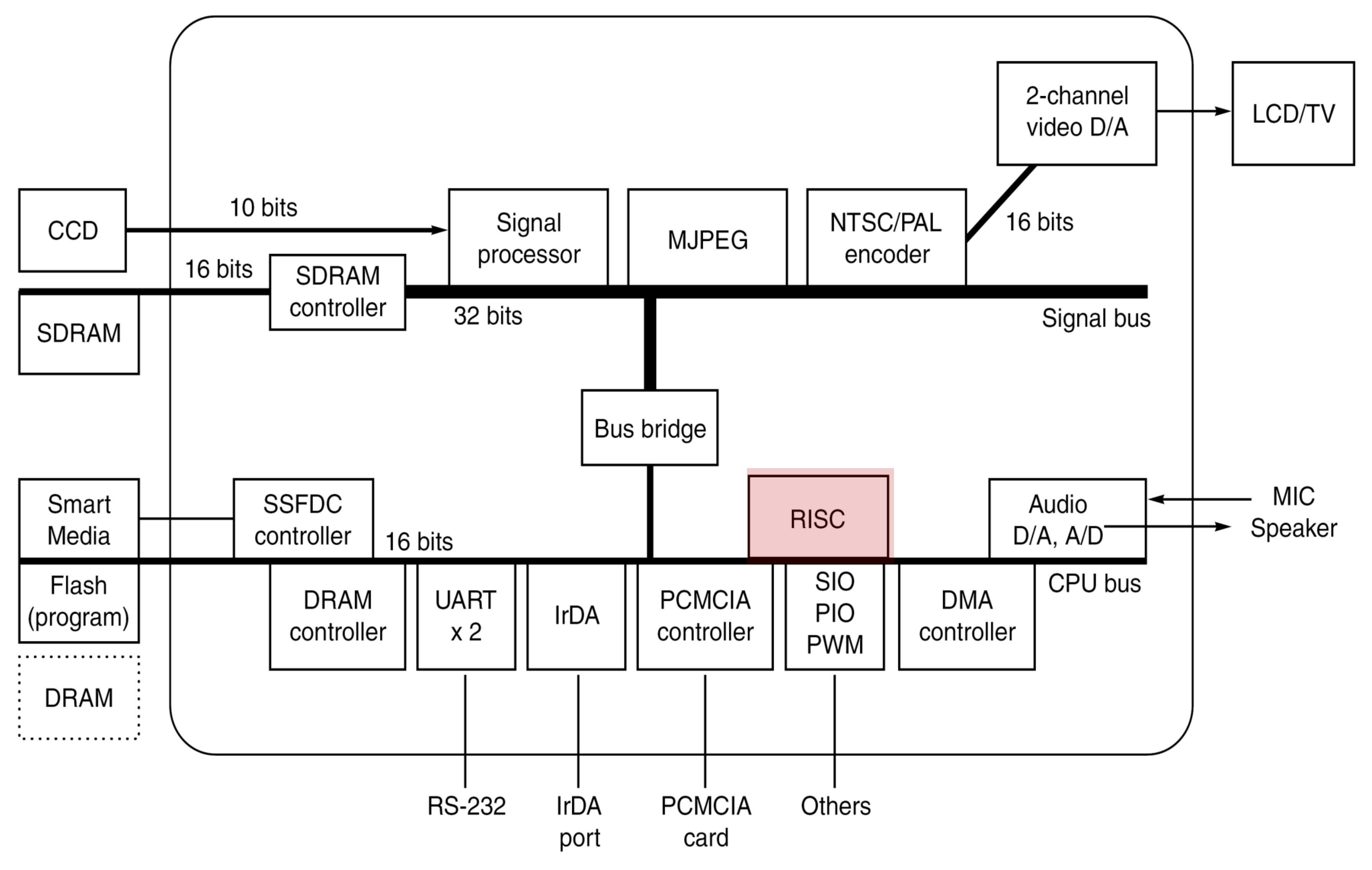


Feature	Personal mobile device (PMD)	Desktop	Server	Clusters/warehouse-scale computer	Embedded
Price of system	$100–$1000	$300–$2500	$5000–$10,000,000	$100,000–$200,000,000	$10–$100,000
Price of micro-processor	$10–$100	$50–$500	$200–$2000	$50–$250	$0.01–$100
Critical system design issues	Cost, energy, media performance, responsiveness	Price-performance, energy, graphics performance	Throughput, availability, scalability, energy	Price-performance, throughput, energy proportionality	Price, energy, application-specific performance

Telecom ES Example



Consumer ES Example



Optimizing Design Criteria

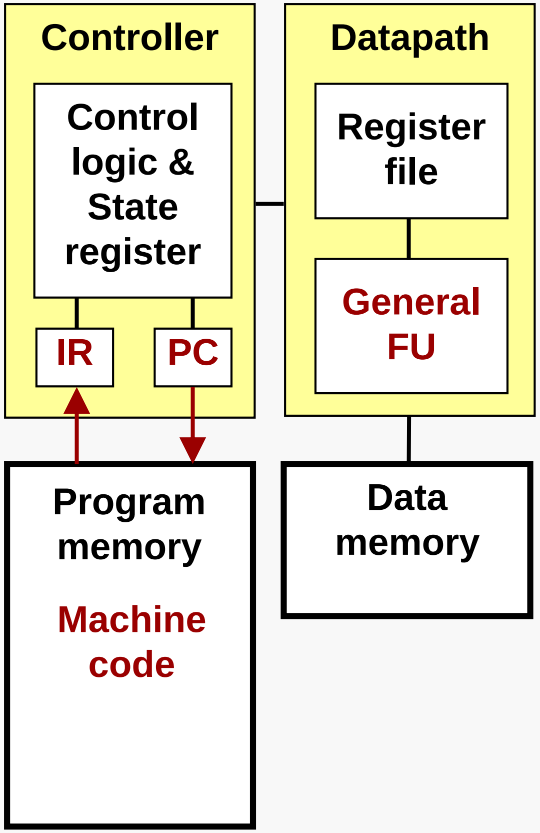
- Challenge: simultaneously optimize multiple (often conflicting) design criteria, while constructing a correct implementation with desired functionality
  - Design criterion: measurable feature of a system's implementation
    - **Size**, **performance**, **power/energy consumption**
    - **Unit cost:** monetary cost of manufacturing each copy of the system, excluding Non-Recurring Engineering (NRE) cost
    - **NRE cost:** one-time monetary cost of designing the system
    - **Time-to-market**, **time-in-market**
  - Other important criteria:
    - **Flexibility:** ability to change system's functionality without incurring heavy NRE costs
    - **Maintainability**, **safety**, etc…

ES Technologies

- Technology
  - A manner of accomplishing a task, using technical processes, methods, or knowledge
- Three key technologies for embedded systems
  - Processor technology
    - The architecture of the computational engine used to implement a system’s desired functionality
  - IC technology
    - The manner in which an implementation is realized as an integrated circuit (IC)
  - Design technology
    - The manner in which we perform a mapping of our desired system functionality to an implementation

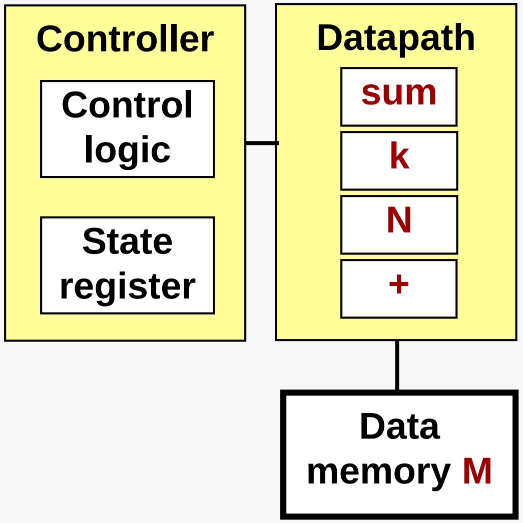
Processor Technology I

- General-purpose processors (GPP)
  - Programmable “microprocessor” used in a variety of applications
  - Features
    - Program memory
    - Program Counter (**PC**) and Instruction Register (**IR**)
    - General datapath with large register file and Functional Unit (**FU**)
  - Benefits
    - Fast time-to-market, low NRE cost, high flexibility



Processor Technology II

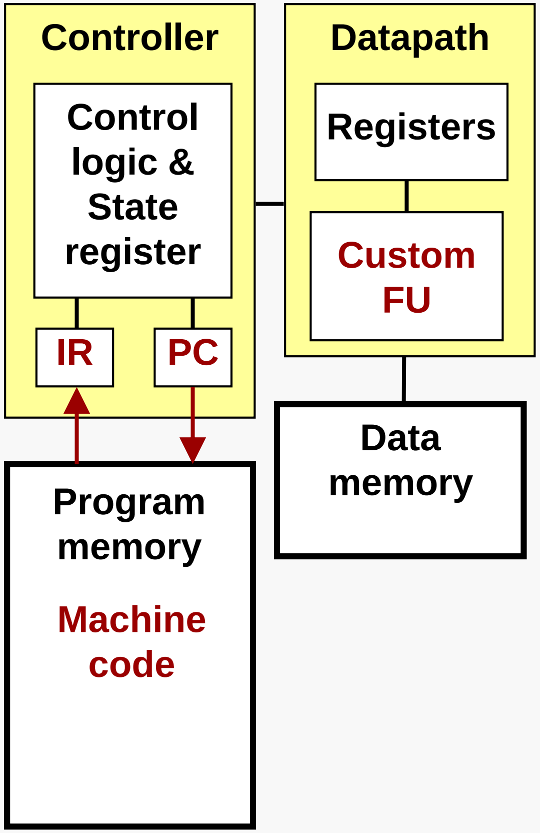
- Single-purpose processor (SPP)
  - Dedicated digital circuit designed to execute exactly one application
  - Features
    - Contains only the components needed to execute a single program
    - No program memory
  - Benefits
    - High performance, low power, small size



```
sum = 0;
for (k=0; k<N; k++)
   sum += M[k];
```

Processor Technology III

- Application-specific instruction processor (ASIP)
  - Programmable processor optimized for a particular class of applications having common characteristics
    - Compromise between GPP and SPP
  - Features
    - Program memory
    - Optimized datapath
    - Custom (specialized) **FU**
  - Benefits
    - Good flexibility, performance, size, power

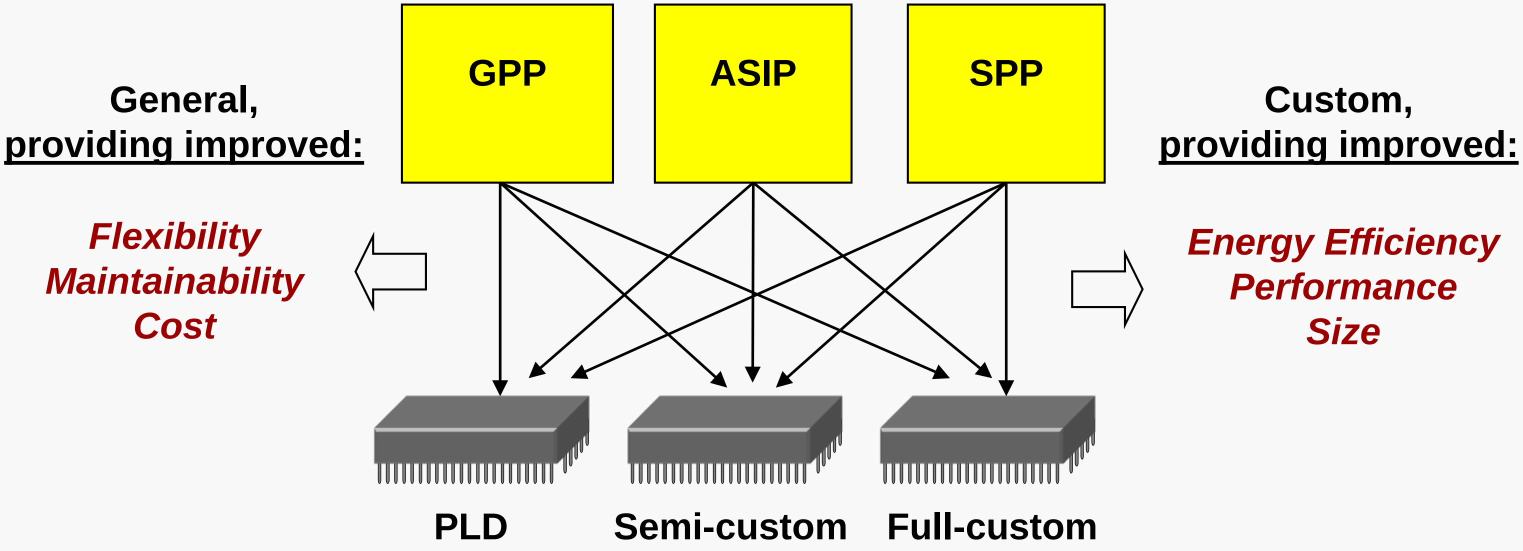


IC Technology

- Full-custom
  - All layers optimized for a particular implementation
    - Designers can control transistor sizing/placement, wire routing
    - Benefits: high performance, small size, low power
- Semi-custom
  - Some layers are only partially designed
    - Designers can control only wire routing, maybe block placement
    - Benefits: good performance, size, power
- PLD (Programmable Logic Device):
  - All layers already pre-fabricated
    - Designers can purchase a reconfigurable IC (e.g., FPGA) and program its internal connections and logic functions
    - Benefits: low NRE cost, reconfigurability

Processor and IC Technologies

- Basic tradeoff: general vs custom
- Processor and IC technologies are **independent**



General,
<u>providing improved:</u>

Flexibility
Maintainability
Cost

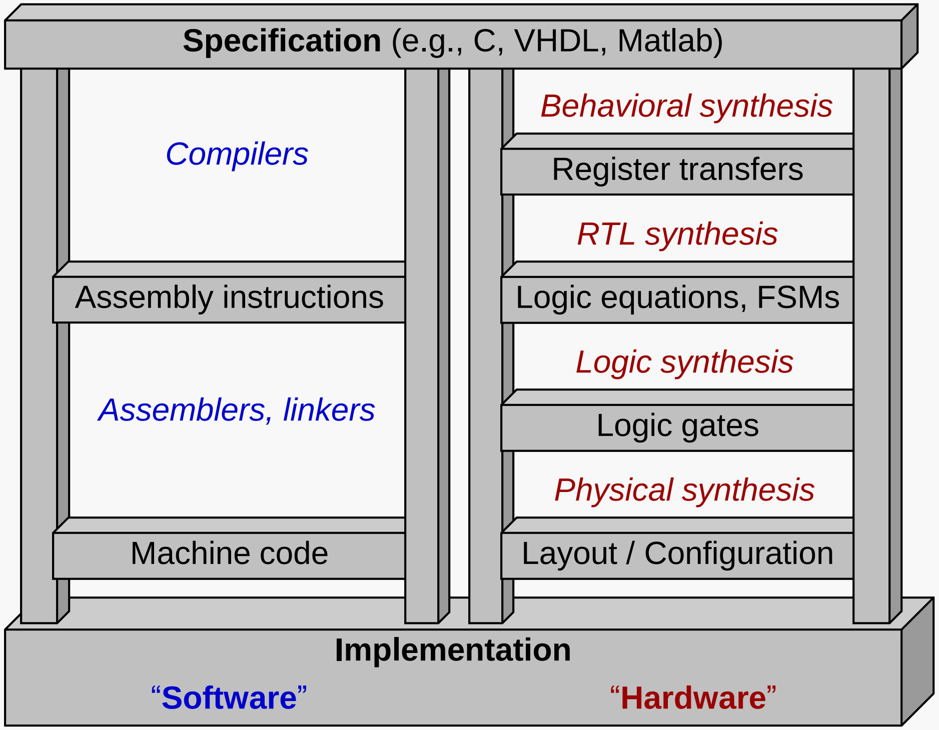
GPP **ASIP** **SPP**

PLD **Semi-custom** **Full-custom**

Custom,
<u>providing improved:</u>

Energy Efficiency
Performance
Size

Design Technology



There is **no fundamental difference** between what hardware or software can implement.

Selecting a **hardware** or a **software** implementation is merely a **tradeoff** among various design criteria: e.g., cost, size, power, performance, etc.

- Expertise with both **software** and **hardware** is needed to optimize multiple design criteria
  - System designers must apply various technologies in order to find the best mapping for a given application and constraints

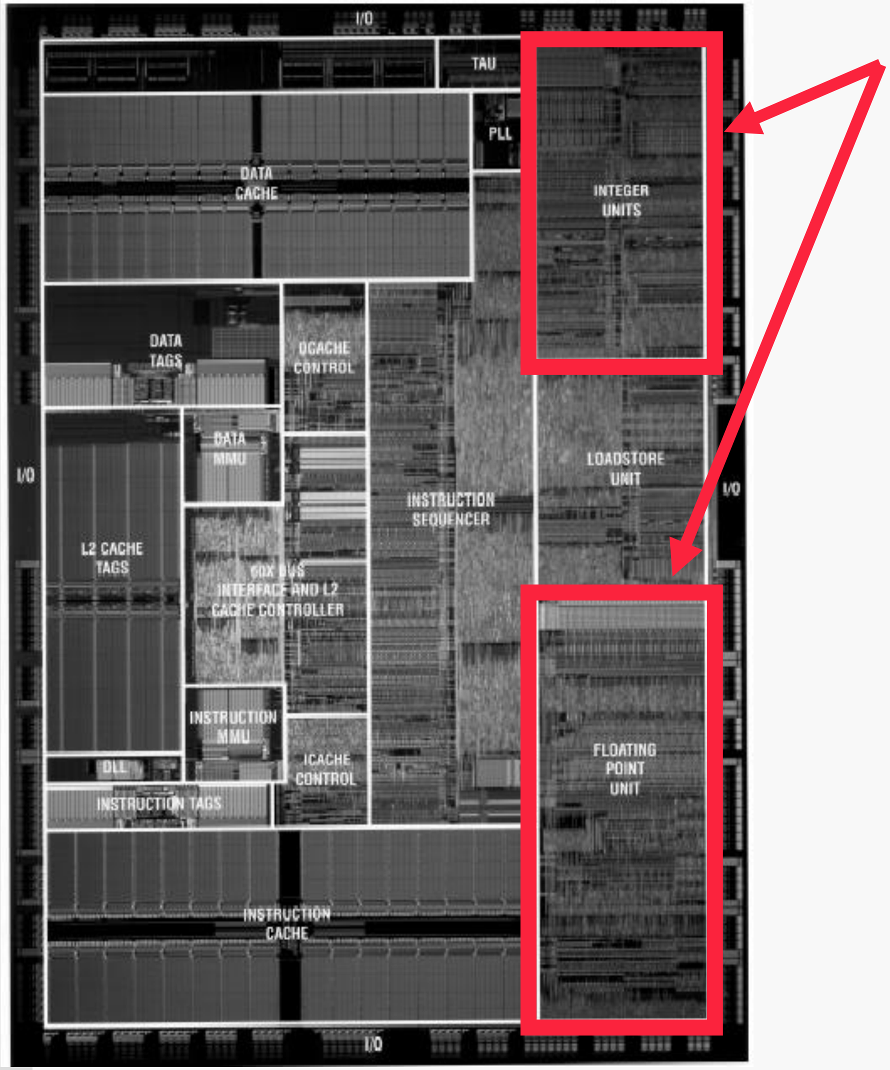
Example: Efficiency Criteria

- High **performance** (speed) is one of the most important design criteria
  - Higher performance usually implies higher **power** dissipation and higher **energy** consumption
  - Higher performance usually requires larger silicon **area** (e.g., to enable parallel execution)

Energy Efficiency = (Useful Computations) / (Energy Required)
= (Number of Operations) / nanoJoule = **OP/nJ**
= (Million OP/Sec) / (mJ/sec) = **MOPS/mW**
= **Power Efficiency**

Area Efficiency = (Useful Computations/Sec) / (Silicon Area)
= (Million OP/Sec) / mm^2 = **MOPS/mm^2**

Microprocessor Example



This is the only circuitry which supports "useful computations" (all the rest is the overhead to support application flexibility)

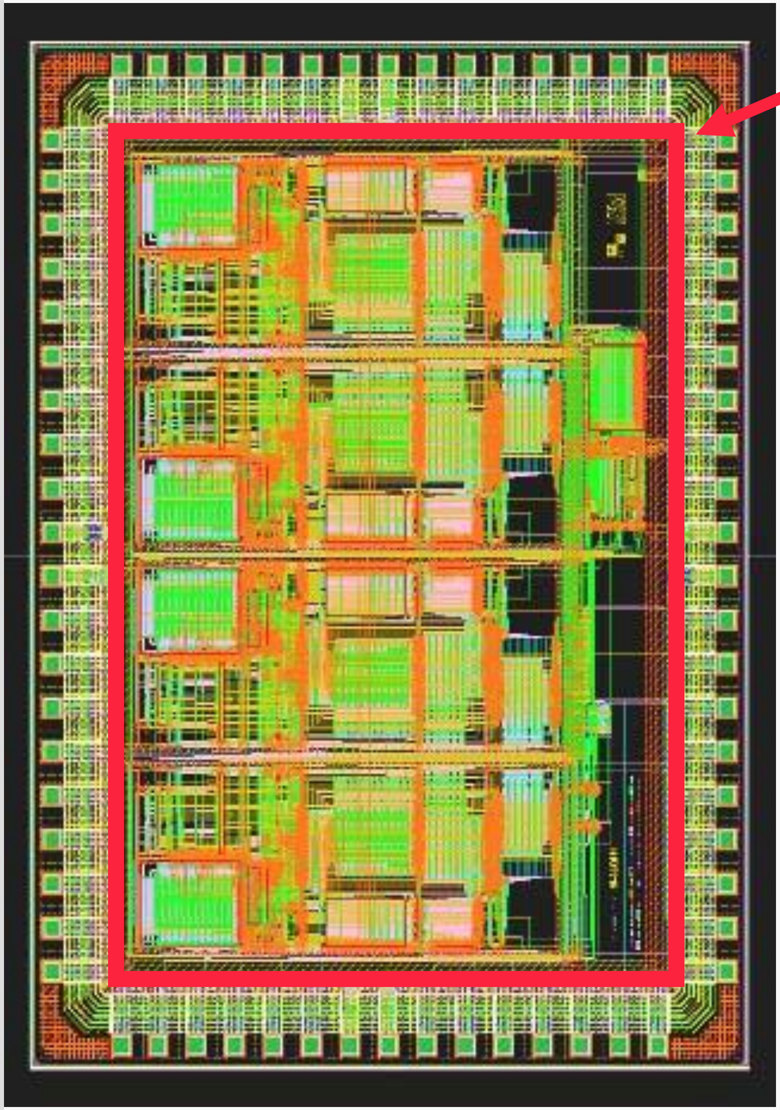
- 84 mm^2 area
- 2 OP per clock cycle
- 450 MHz clock frequency

Performance = 900 MOPS

Power = 7000 mW

900 / 7000 = **0.13** MOPS/mW
900 / 84 = **10.7** MOPS/mm^2

Dedicated Hardware Example



Dedicated adaptive correlator circuit (no application flexibility)

- 5.4 mm^2 area
- 96 OP per clock cycle
- 25 MHz clock frequency

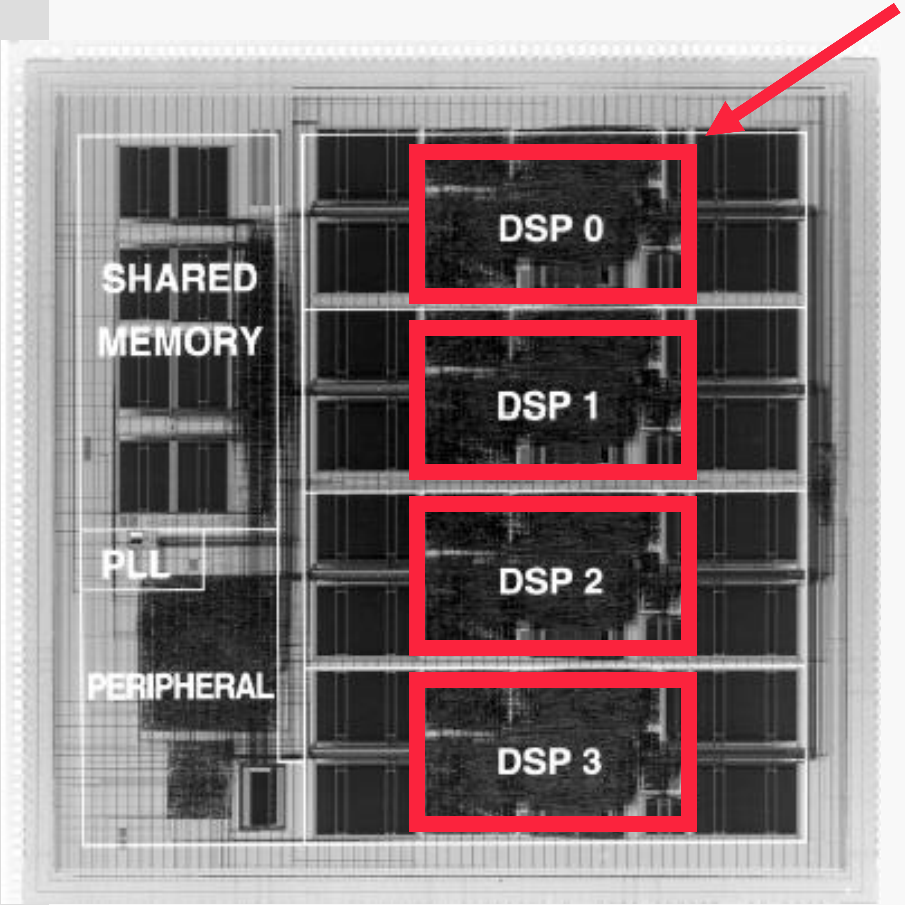
Performance = 2400 MOPS

Power = 12 mW

2400 / 12 = **200** MOPS/mW

2400 / 5.4 = **444.4** MOPS/mm^2

Digital Signal Processor Example



It is similar to a microprocessor, but exploits more parallelism and specialized computations (less application flexibility)

- 84.8 mm^2 area
- 16 OP per clock cycle
- 50 MHz clock frequency

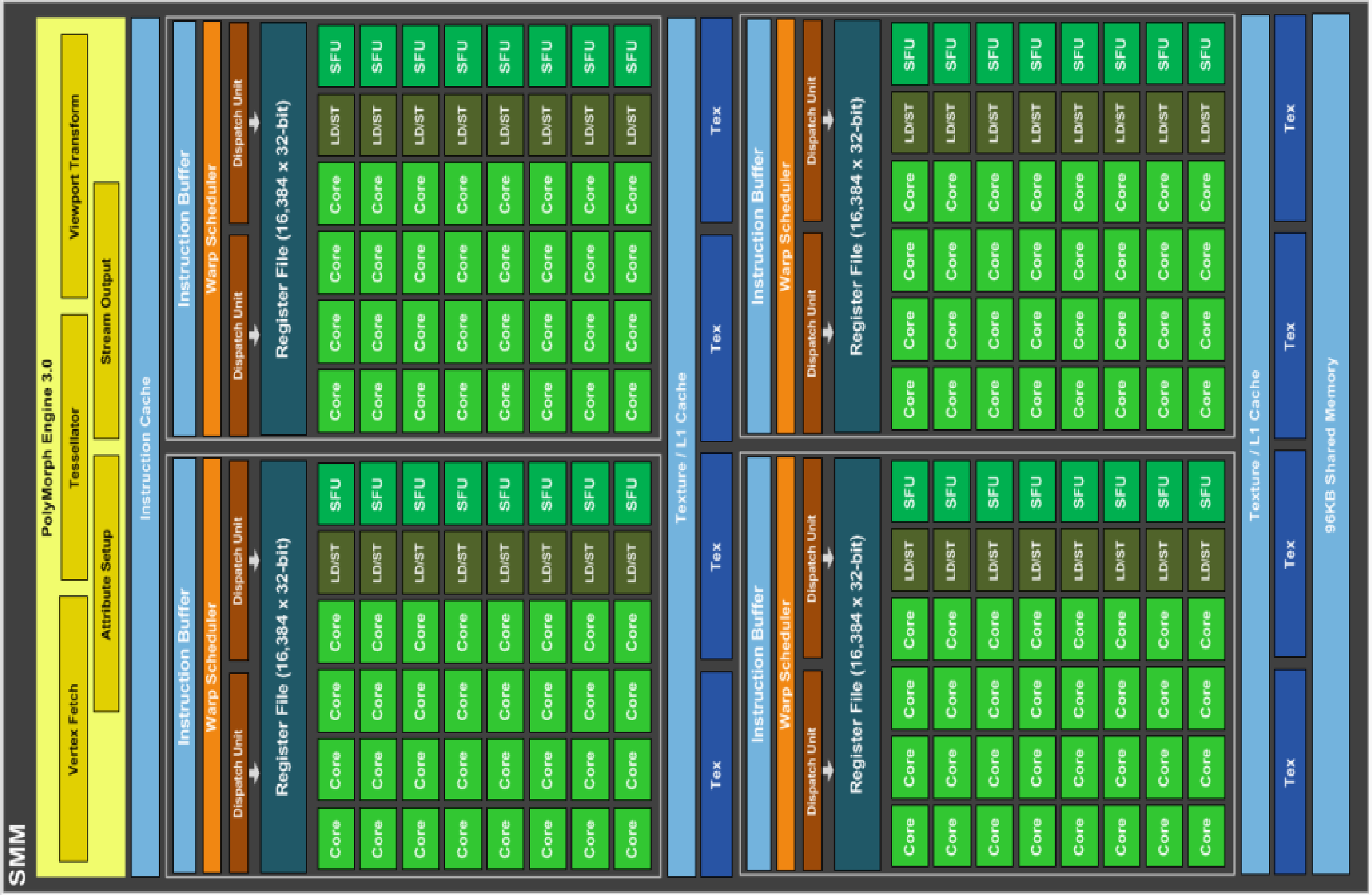
Performance = 800 MOPS

Power = 110 mW

800 / 110 = **7.3** MOPS/mW

800 / 84.5 = **9.5** MOPS/mm^2

Many Cores → High Performance



Amdahl's Law

- **Speedup = 1/(1 – $f_{enhanced}$ + $f_{enhanced}/S_{enhanced}$)**
  - $S_{enhanced}$ – ideal speedup achievable due to enhancement
  - $f_{enhanced}$ – fraction of computational time improved due to enhancement
    - Fraction of computational time NOT improved: **1 – $f_{enhanced}$**
- Enhancement: 1 processor → **P** processors
  - Assume: **f** = parallelizable fraction of computation
  - **Speedup = 1/(1 – f + f/P) = P/(P – f(P – 1))**
- Example:
  - **P** = 8, **f** = 0.75, **Speedup** = 2.9 (much less than 8)
  - **P** = 128, **f** = 0.75, **Speedup** = 3.9 (much less than 128)
    - Limited return-on-investment (ROI) ← **f** is very important!

Why Microprocessors?

- The **performance/power/area** advantages of custom hardware over general microprocessors are so big that there are no technical reasons to use microprocessors to do useful computations
  - However, **risk/time/cost** of implementing dedicated hardware solutions can be prohibitive
- There are economic reasons (e.g., MOPS/$) that demand flexibility (hence, microprocessors)
  - One design for many customers – more sales
    - Customers able to provide added value and uniqueness
  - Uncertain specifications – soft design decisions
    - Engineers able to make changes quickly and cheaply
  - Challenge: maintainability and reusability of software-based products