Embedded System Design

Stephen A. Edwards

Columbia University

Spring 2016

Spot the Computer







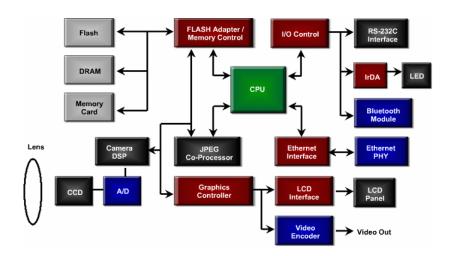
Cars These Days...



Embedded Systems: Ubiquitous Computers



Inside a Digital Camera



Want an Optimal Device that Meets Constraints On



Price



Functionality



Performance



Size



Power



Time-to-market



Maintainability



Safety

Embedded System Technologies



Integrated Circuits



Processing elements



Design tools

IC Technology



1947: First transistor (Shockley, Bell Labs)



1958: First integrated circuit (Kilby, TI)

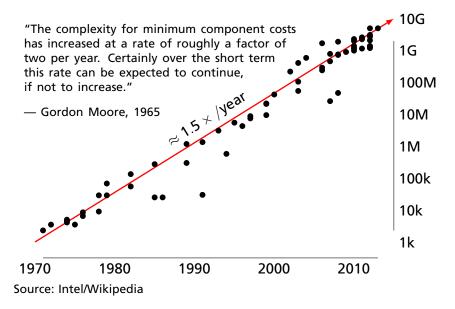


1971: First microprocessor (4004: Intel)

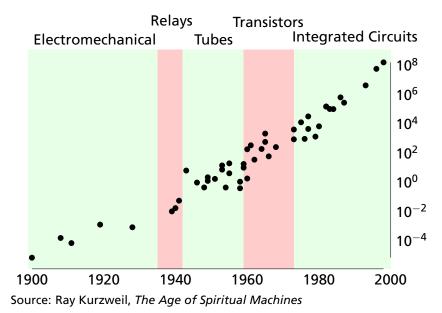


2015: 14 nm features, 13 layers (Intel, Broadwell)

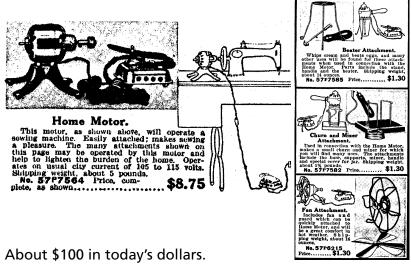
Moore's Law: Transistors per chip



\$1000 Buys You This Many Cycles per Second



1918 Sears Roebuck Catalog

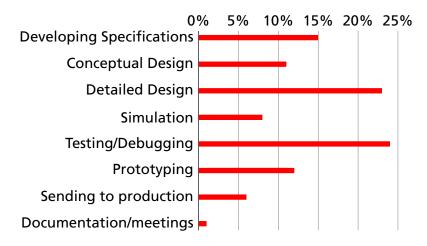


From Donald Norman, The Invisible Computer, 1998.

What Percentage of Time Do You Spend...

09	% 5	5%	10%	15%	20%	25%
Developing Specifications						
Conceptual Design						
Detailed Design						
Simulation						
Testing/Debugging						
Prototyping						
Sending to production						
Documentation/meetings						

What Percentage of Time Do You Spend...



Does Your Current Project Contain FPGAs?

Does Your Current Project Contain FPGAs?

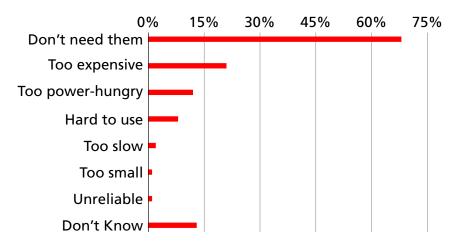
45% Yes

55% No

Why Won't Your *Next* Project Use FPGAs?

00	% 1	5%	30%	45%	60%	75%
Don't need them						
Too expensive						
Too power-hungry						
Hard to use						
Too slow						
Too small						
Unreliable						
Don't Know						

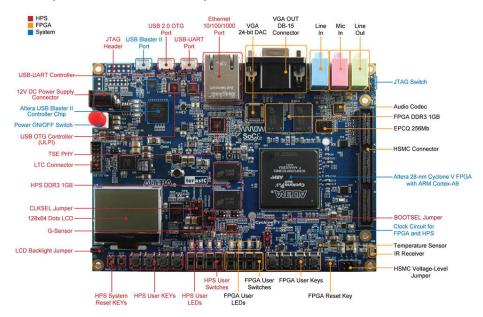
Why Won't Your *Next* Project Use FPGAs?



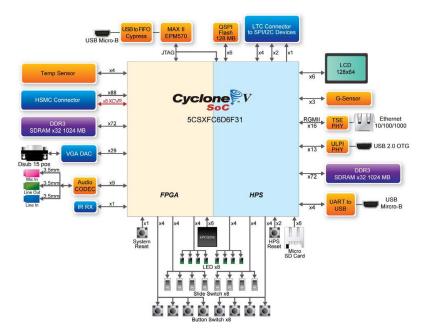
Your Nemesis: The SoCKit Board



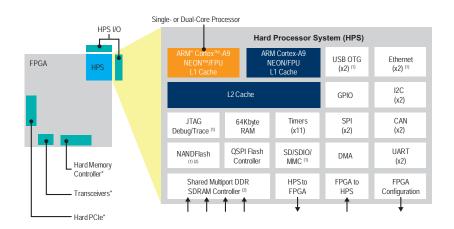
Components and Peripherals



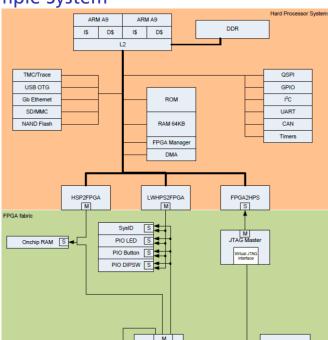
Dual ARM Cortex-A9 and Programmable Logic



Inside the Cyclone V: Dual ARM processors + FPGA

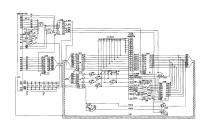


An Example System



Linux + Custom Hardware





Class Structure

Three Introductory Labs: 2 weeks each

Work in pairs

1. Hardware: Access, modify, and display memory

2. Software: A simple Internet chat client

3. HW + SW: A video bouncing ball

The project: **Design-your-own**

Work in groups of four

Broadly: C + SystemVerilog + peripheral(s)

Broad Project Idea: Video Game



Implement graphics in custom hardware
Put game logic in software
Interface with USB HID (Joystick, etc.)

E.g., Pac-man, 2.5D maze game, tank, worms









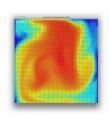








Broad Project Idea: Computational Accelerator



Pick a computationally intensive algorithm

Implement its core in custom hardware

Write software and device drivers that pass data to and from the accelerator

E.g., Smoke simulator, inverse kinematics for robotics, Bitcoin miner

Broad Project Idea: Network Accelerator



Pick a simple network processing problem, e.g., from finance

Implement part of existing software algorithm in hardware

Interface hardware with network controller; processor

E.g., TCP/IP, tickerplant, margin calculations, memcached, FIX protocol parser

More Ideas



Digital tone control



Spectrum analyzer



Internet radio



Speech Synthesizer



MIDI synthesizer



Accelerated JPEG



Game of Life



Pool game

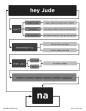


Real-time ray tracer

The Three Main Challenges of Embedded Systems



Coping with Real-World Sensor Data

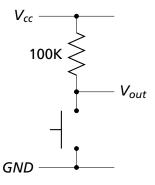


Algorithm Design

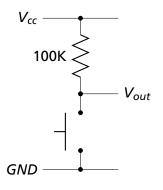


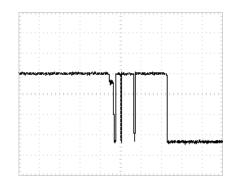
Implementation Details

What Happens When You Press the Switch?

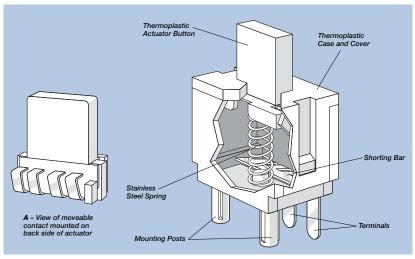


What Happens When You Press the Switch?





Inside a Pushbutton Switch

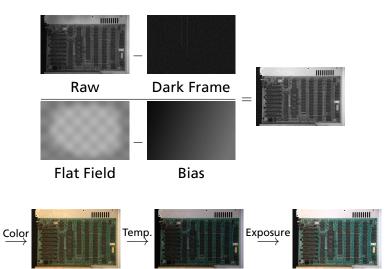


Source: Cherry CS series data sheet

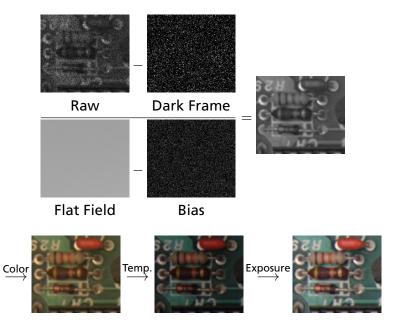
Raw Data from a CCD (zoomed in)

Corrected Image (zoomed in)

Correcting Data from CCDs



Correcting Data from CCDs



Where Does This Noise Come From?

Nikon D300: 23.6 mm \times 15.8 mm 12.3 megapixel CMOS sensor

Pixels are 5.5 μ m on a side

A/D sampling of 12 bits per pixel measures



ISO:	LO	200	400	800	1600	3200
G	7.1	5.5	2.7	1.3	0.65	0.33
В	5.8	4.6	2.3	1.1	0.55	0.27
R	4.7	4.5	2.2	1.1	0.54	0.26

The units: electrons per ADU (digital unit).

Emil Martinec, A comparison of the Nikon D300 and Canon 40D sensors, 2007.

Development Plan

- 1. Obtain some representative raw sensor data
- 2. Develop an algorithmic prototype using your favorite language (e.g., Java, C, Matlab)
- 3. Plan how to implement it
- 4. Implement while constantly testing