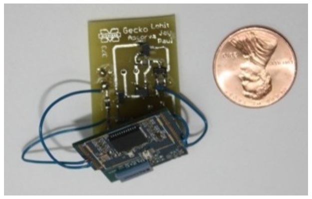
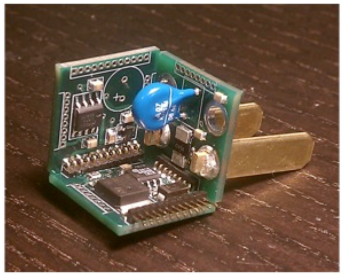
EECS 373

Design of Microprocessor-Based Systems

http://web.eecs.umich.edu/~prabal/teaching/eecs373



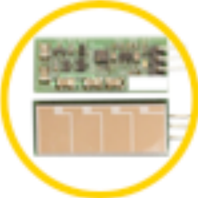


Prabal Dutta

University of Michigan

$100\ cm^3$
[IPSN'12]

$50\ cm^3$
[Sensys'12]

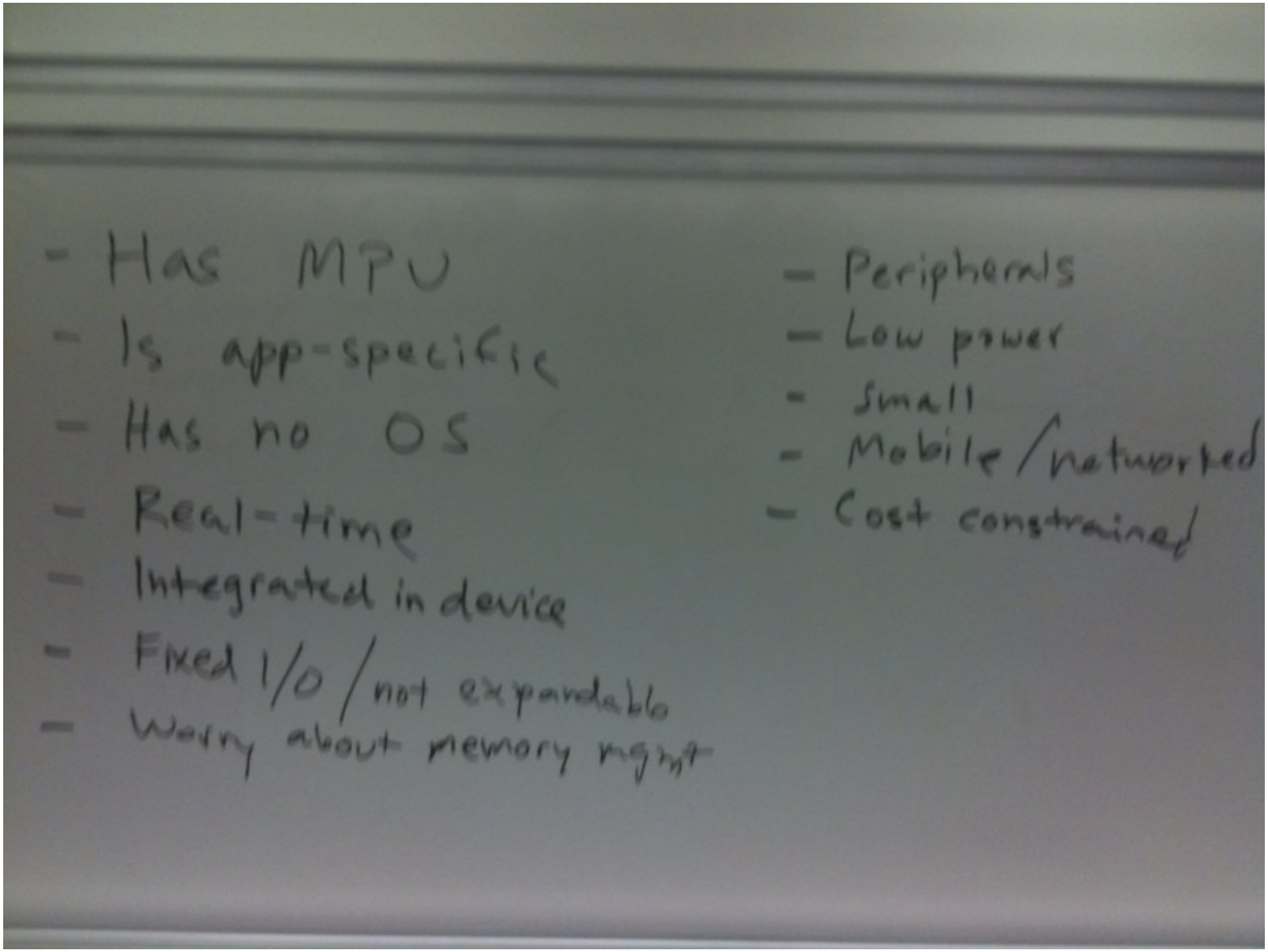


$1\ cm^3$
[IPSN'12]

$1\ mm^3$
[ISSCC'12]

Lecture 1: Introduction
September 2, 2014

What is an embedded system?



- Has MPU
- Is app-specific
- Has no OS
- Real-time
- Integrated in device
- Fixed I/O / not expandable
- Worry about memory mgmt

- Peripherals
- Low power
- Small
- Mobile / networked
- Cost constrained

Embedded, Everywhere

Embedded, Everywhere - Fitbit

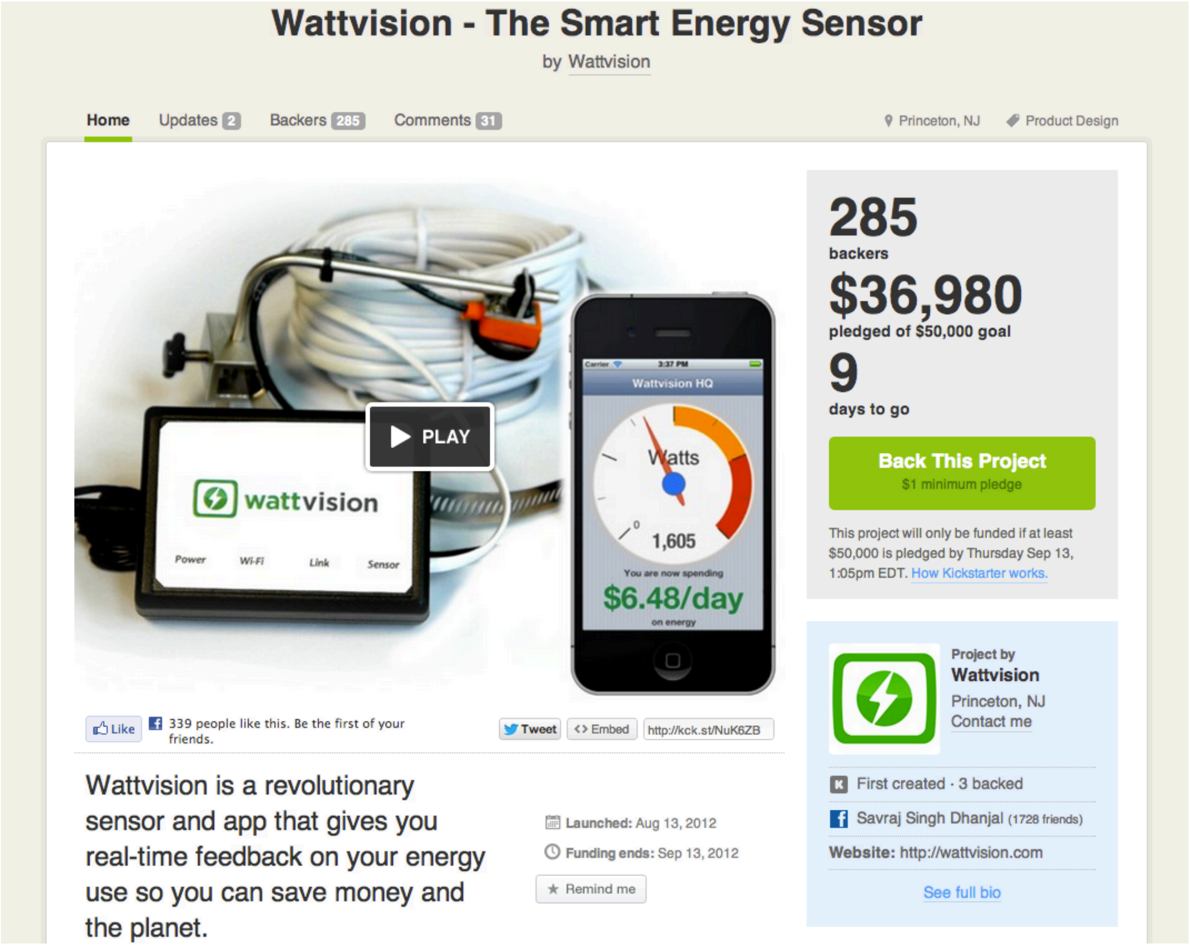
Calories

1800 calories burned

372 calories eaten

Intraday calories burned every 5 min eaten

Embedded, Everywhere - WattVision on Kickstarter



Wattvision - The Smart Energy Sensor

by Wattvision

285
backers

$36,980
pledged of $50,000 goal

9
days to go

This project will only be funded if at least $50,000 is pledged by Thursday Sep 13, 1:05pm EDT. How Kickstarter works.

339 people like this. Be the first of your friends.

http://kck.st/NuK6ZB

Wattvision is a revolutionary sensor and app that gives you real-time feedback on your energy use so you can save money and the planet.

Launched: Aug 13, 2012
Funding ends: Sep 13, 2012

Project by
Wattvision
Princeton, NJ
Contact me

First created · 3 backed
Savraj Singh Dhanjal (1728 friends)
Website: http://wattvision.com

See full bio

What is driving the embedded everywhere explosion?

Outline

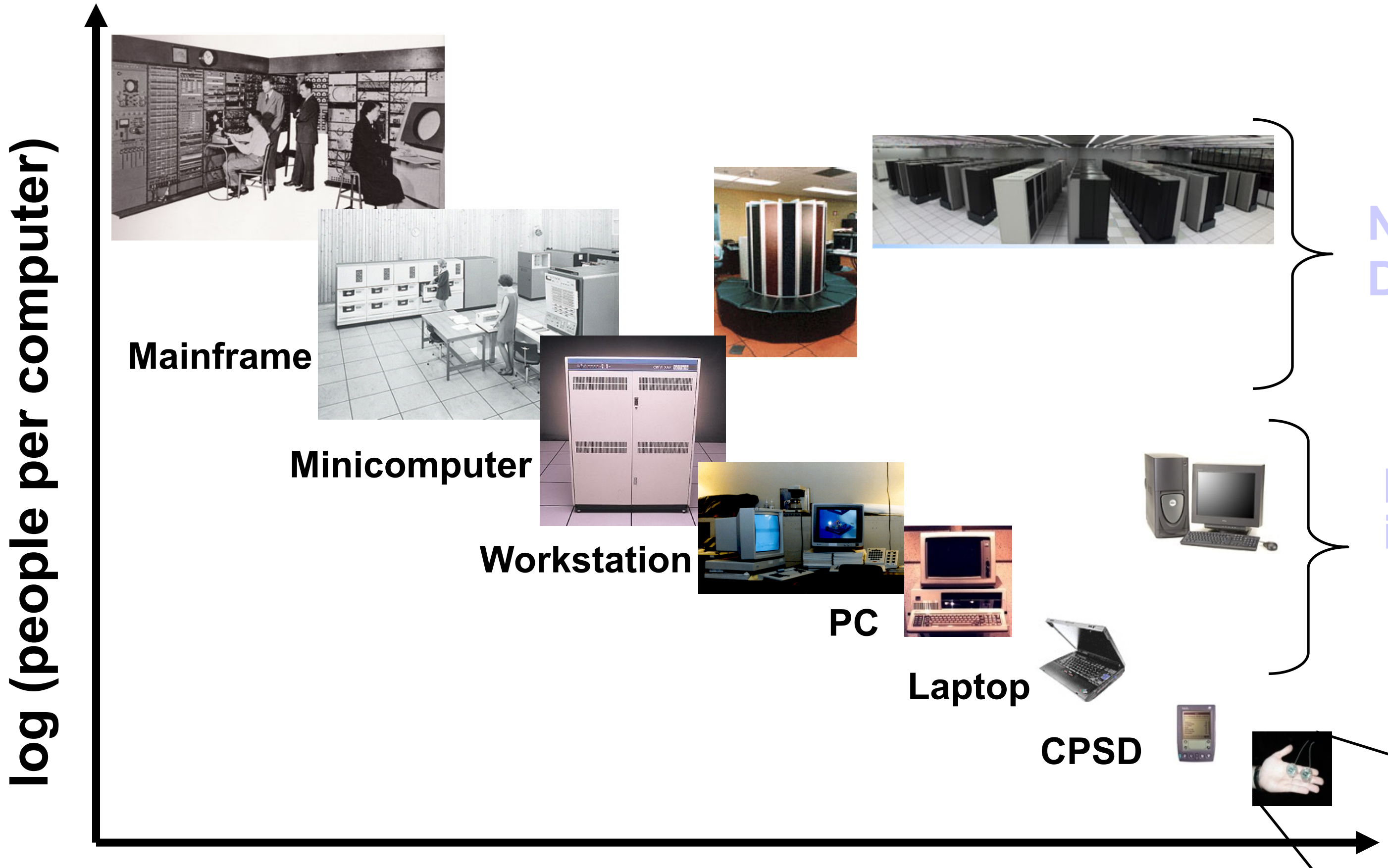
Technology Trends

Design Questions

Course Administrivia

Tools Overview/ISA Start

Bell's Law of Computer Classes:
A new computing class roughly every decade

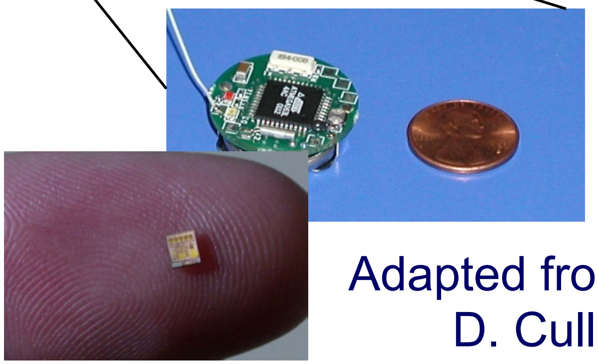


year

Number Crunching
Data Storage

productivity
interactive

streaming information to/from physical world



"Roughly every decade a new, lower priced computer class forms based on a new programming platform, network, and interface resulting in new usage and the establishment of a new industry."

Adapted from
D. Culler

Moore's Law (a statement about economics): IC transistor count doubles every 18-24 mo

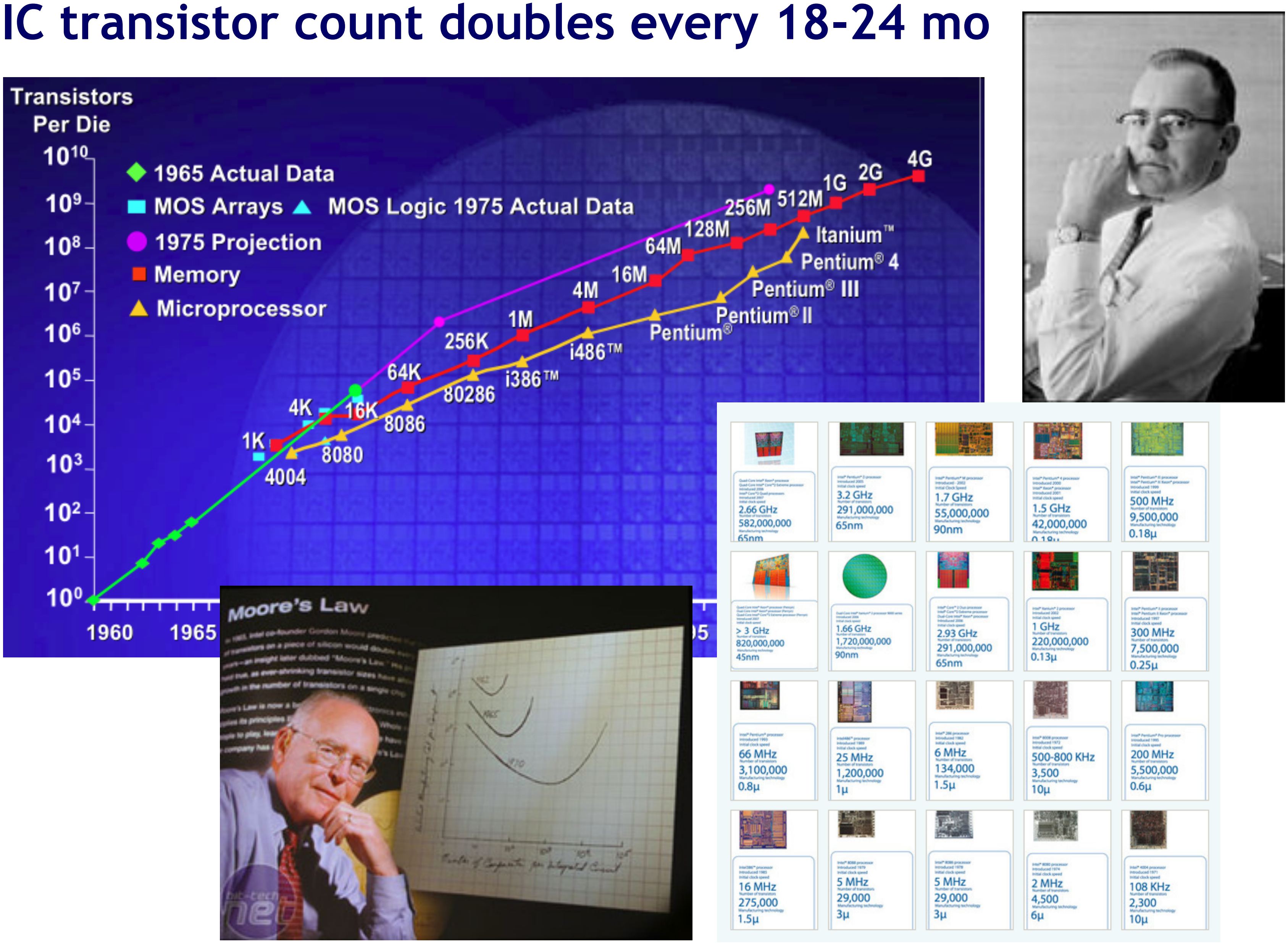
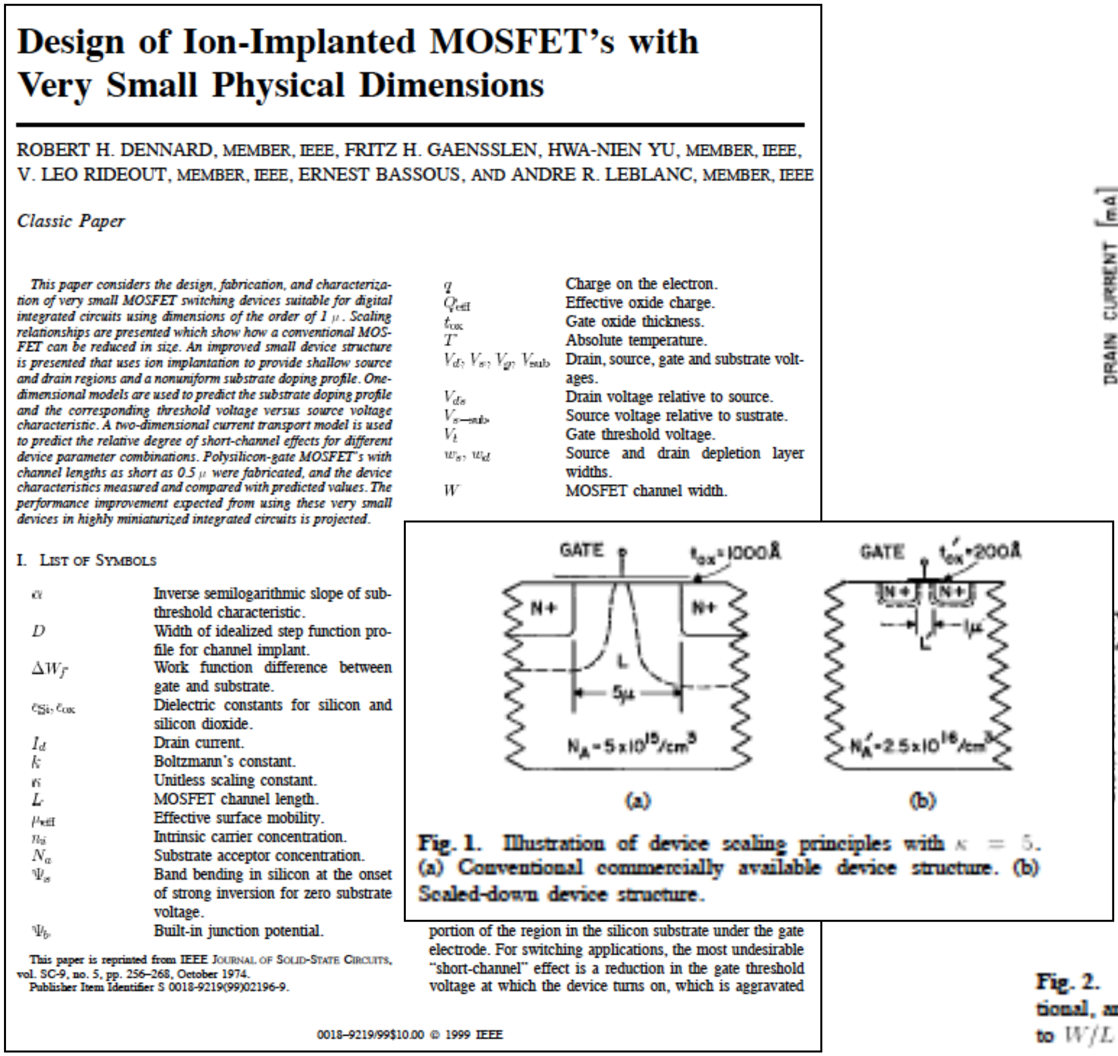


Photo Credit: Intel

Dennard Scaling made transistors fast and low-power: So everything got better!



Design of Ion-Implanted MOSFET's with Very Small Physical Dimensions

ROBERT H. DENNARD, MEMBER, IEEE, FRITZ H. GAENSSLEN, HWA-NIEN YU, MEMBER, IEEE, V. LEO RIDEOUT, MEMBER, IEEE, ERNEST BASSOUS, AND ANDRE R. LEBLANC, MEMBER, IEEE

Classic Paper

This paper considers the design, fabrication, and characterization of very small MOSFET switching devices suitable for digital integrated circuits using dimensions of the order of 1 μ. Scaling relationships are presented which show how a conventional MOSFET can be reduced in size. An improved small device structure is presented that uses ion implantation to provide shallow source and drain regions and a nonuniform substrate doping profile. One-dimensional models are used to predict the substrate doping profile and the corresponding threshold voltage versus source voltage characteristic. A two-dimensional current transport model is used to predict the relative degree of short-channel effects for different device parameter combinations. Polysilicon-gate MOSFET's with channel lengths as short as 0.5 μ were fabricated, and the device characteristics measured and compared with predicted values. The performance improvement expected from using these very small devices in highly miniaturized integrated circuits is projected.

I. List of Symbols

α	Inverse semilogarithmic slope of sub-threshold characteristic.
D	Width of idealized step function profile for channel implant.
ΔW_f	Work function difference between gate and substrate.
$\epsilon_{Si}, \epsilon_{ox}$	Dielectric constants for silicon and silicon dioxide.
I_d	Drain current.
k	Boltzmann's constant.
κ	Unitless scaling constant.
L	MOSFET channel length.
μ_{eff}	Effective surface mobility.
n_i	Intrinsic carrier concentration.
N_a	Substrate acceptor concentration.
Ψ_s	Band bending in silicon at the onset of strong inversion for zero substrate voltage.
Ψ_b	Built-in junction potential.
q	Charge on the electron.
Q_{eff}	Effective oxide charge.
t_{ox}	Gate oxide thickness.
T	Absolute temperature.
V_d, V_s, V_g, V_{sub}	Drain, source, gate and substrate voltages.
V_{ds}	Drain voltage relative to source.
V_{s-sub}	Source voltage relative to sustrate.
V_t	Gate threshold voltage.
w_s, w_d	Source and drain depletion layer widths.
W	MOSFET channel width.

Fig. 1. Illustration of device scaling principles with $\kappa = 5$. (a) Conventional commercially available device structure. (b) Scaled-down device structure.

portion of the region in the silicon substrate under the gate electrode. For switching applications, the most undesirable "short-channel" effect is a reduction in the gate threshold voltage at which the device turns on, which is aggravated

This paper is reprinted from IEEE Journal of Solid-State Circuits, vol. SC-9, no. 5, pp. 256–268, October 1974.
Publisher Item Identifier S 0018-9219(99)02196-9.

0018–9219/99$10.00 © 1999 IEEE

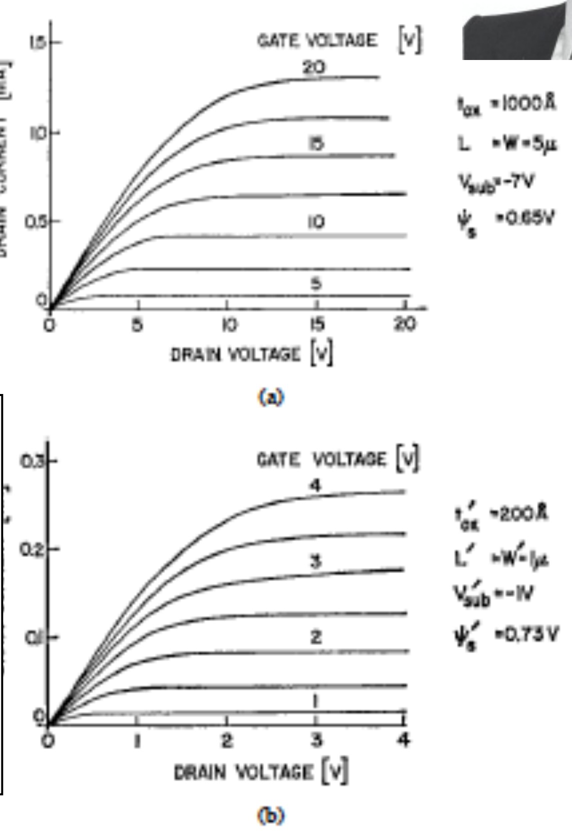
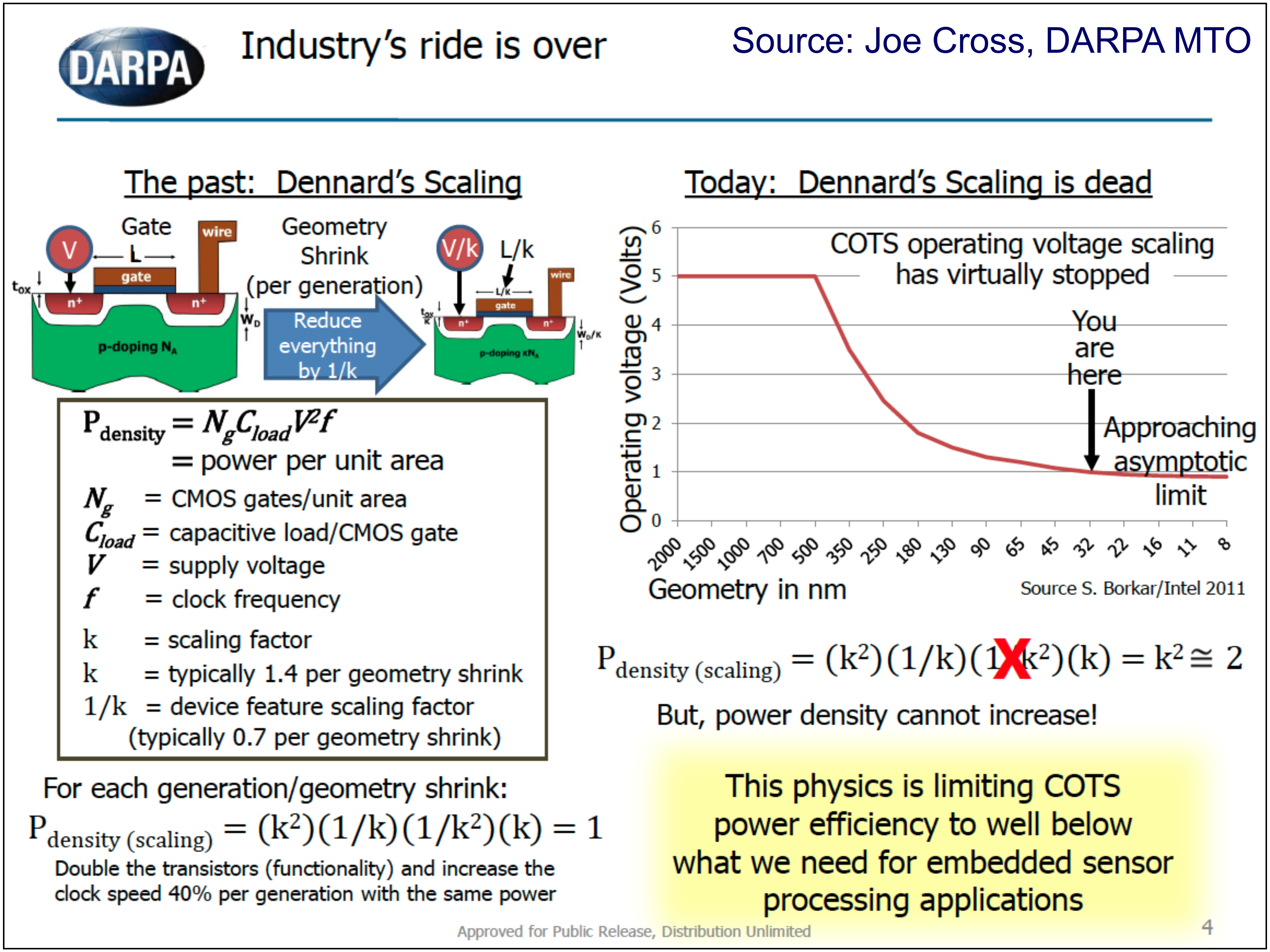


Fig. 2. Experimental drain voltage characteristics for (a) conventional, and (b) scaled-down structures shown in Fig. 1 normalized to $W/L = 1$.

Dennard Scaling…is Dead



Industry's ride is over

Source: Joe Cross, DARPA MTO

The past: Dennard's Scaling

$P_{density} = N_g C_{load} V^2 f$

= power per unit area

N_g = CMOS gates/unit area

C_{load} = capacitive load/CMOS gate

V = supply voltage

f = clock frequency

k = scaling factor

k = typically 1.4 per geometry shrink

1/k = device feature scaling factor (typically 0.7 per geometry shrink)

For each generation/geometry shrink:

$$P_{density\ (scaling)} = (k^2)(1/k)(1/k^2)(k) = 1$$

Double the transistors (functionality) and increase the clock speed 40% per generation with the same power

Today: Dennard's Scaling is dead

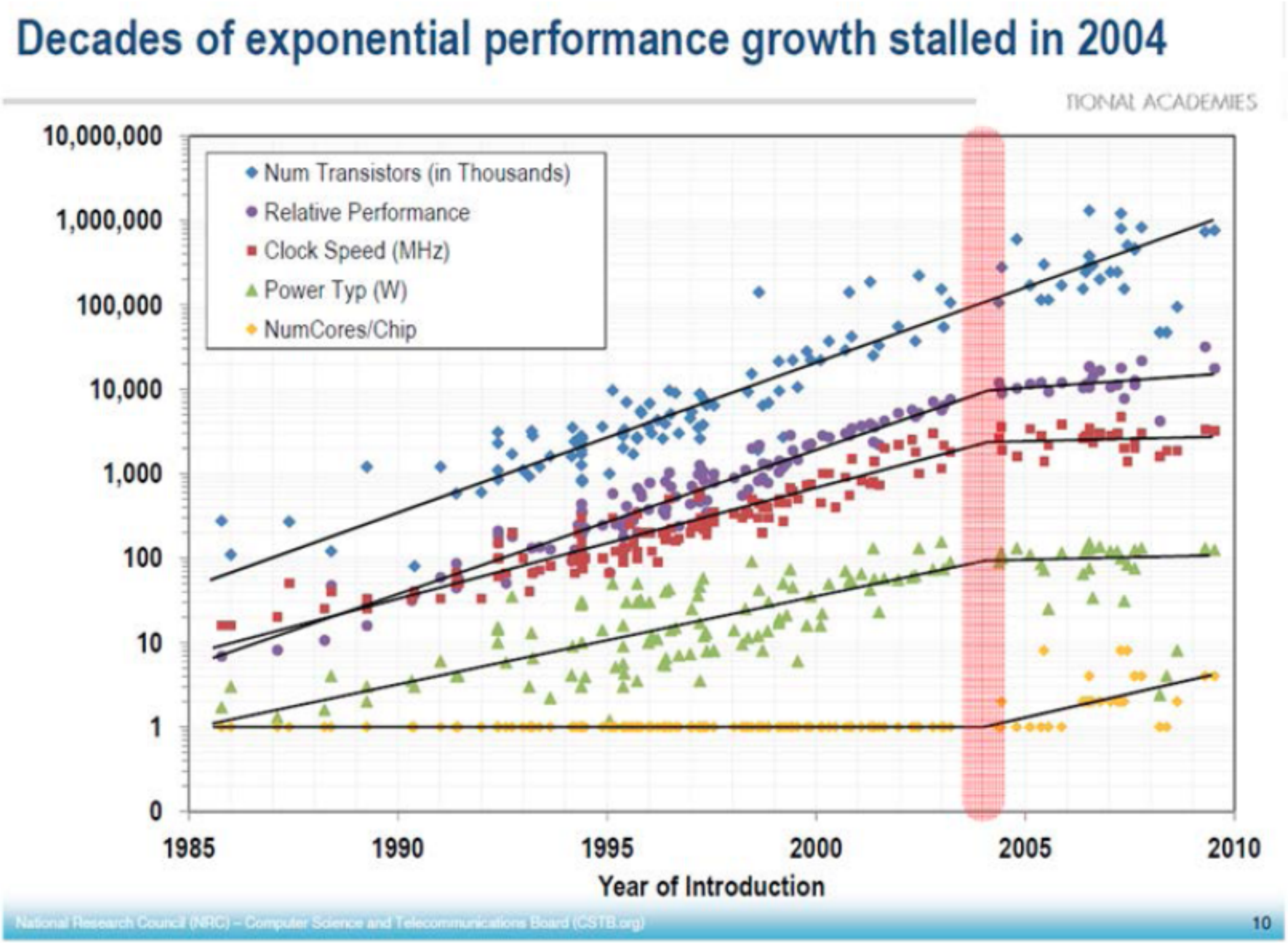
Source S. Borkar/Intel 2011

$$P_{density\ (scaling)} = (k^2)(1/k)(1\cancel{/}k^2)(k) = k^2 \cong 2$$

But, power density cannot increase!

This physics is limiting COTS power efficiency to well below what we need for embedded sensor processing applications

And the Party's Over

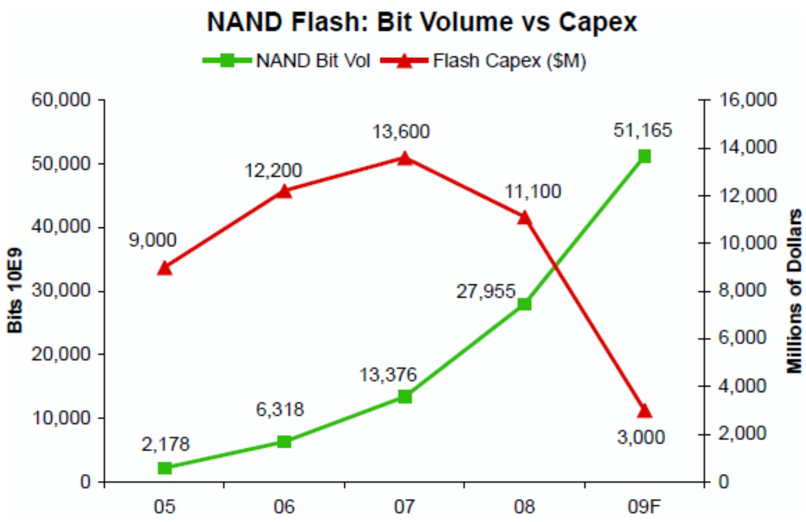


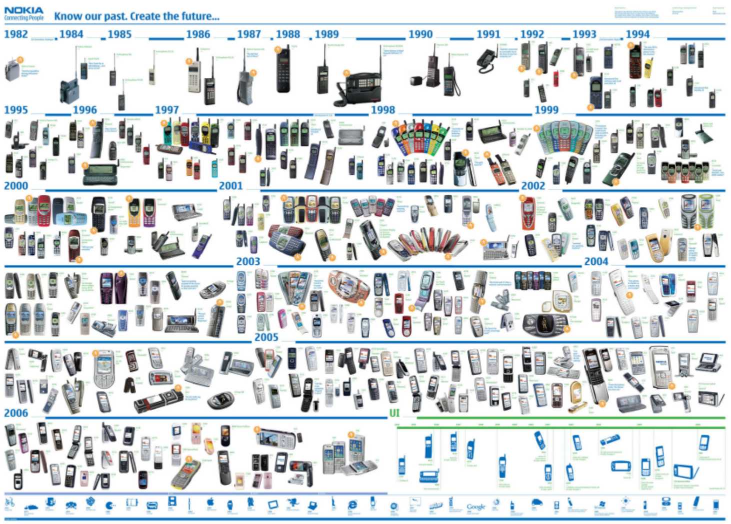
Decades of exponential performance growth stalled in 2004

National Research Council (NRC) – Computer Science and Telecommunications Board (CSTB.org) 10

Source: NRC, The Future of Computing Performance, Game Over or Next Level?

Flash memory scaling:
Rise of density & volumes; Fall (and rise) of prices





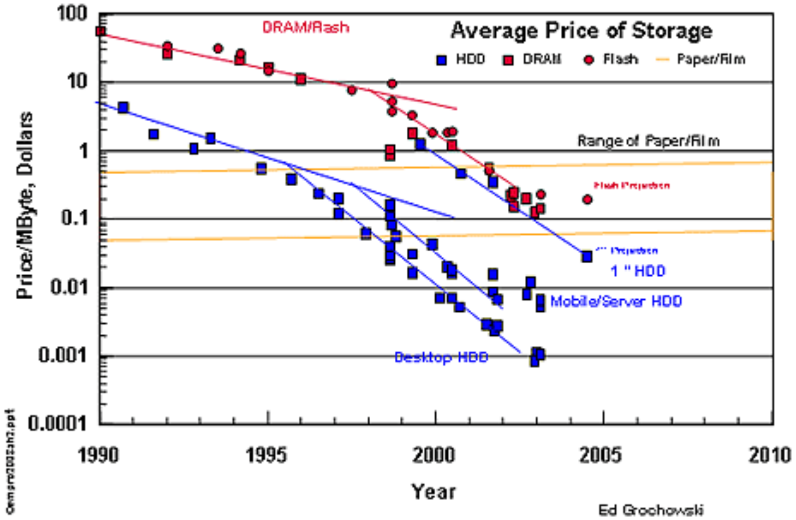
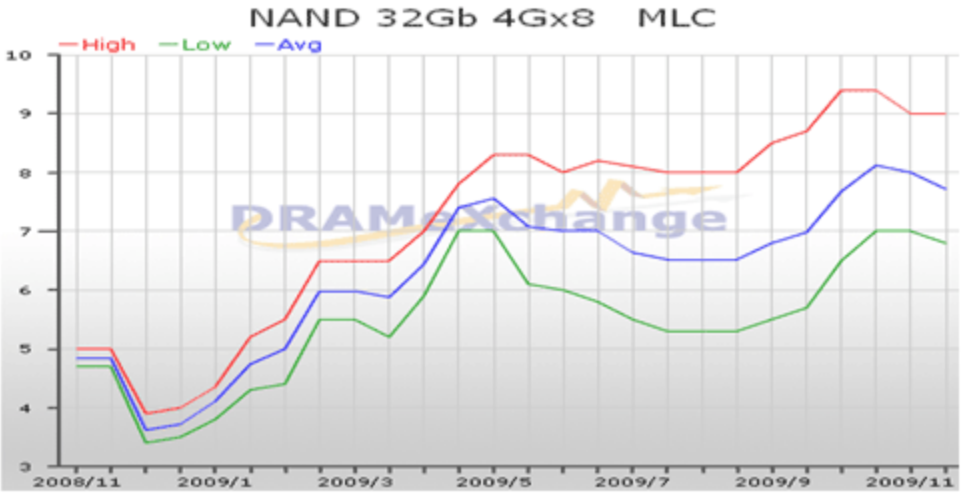
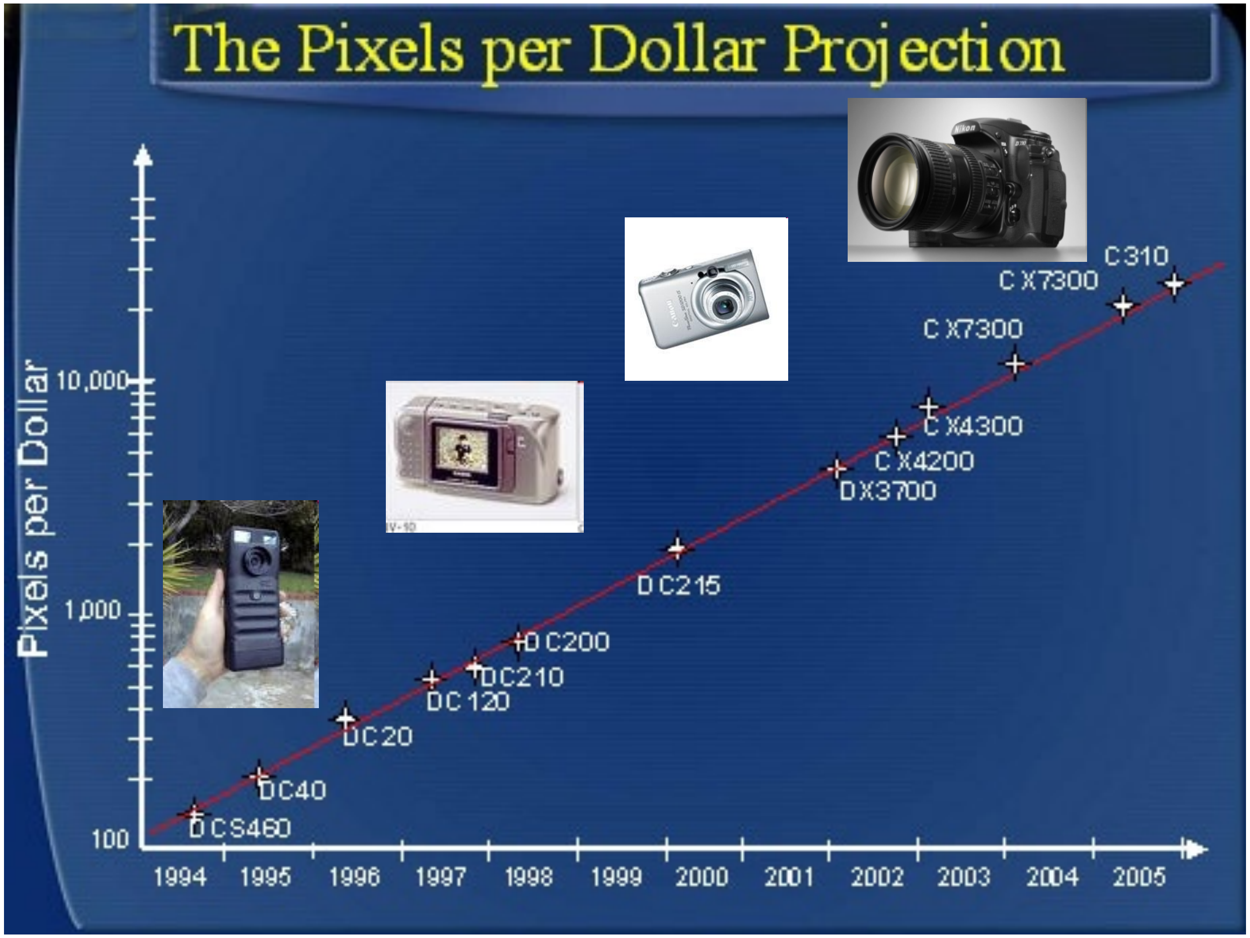


Figure-1 32Gb MLC NAND Flash contract price trend



Source: DRAMeXchange, Nov. 2009.

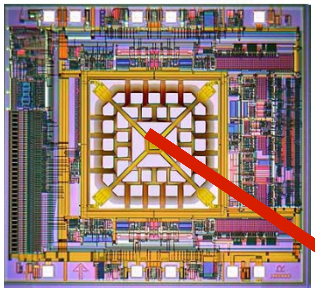
Hendy’s “Law”:
Pixels per dollar doubles annually



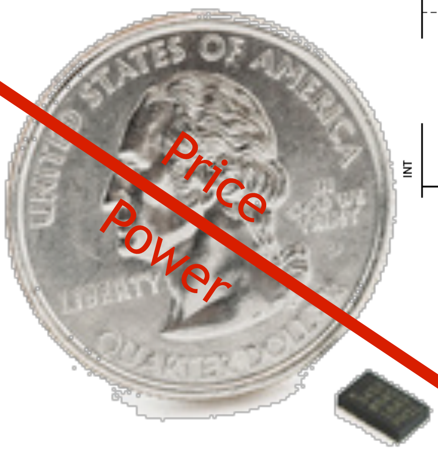
Credit: Barry Hendy/Wikipedia

MEMS Accelerometers: Rapidly falling price and power

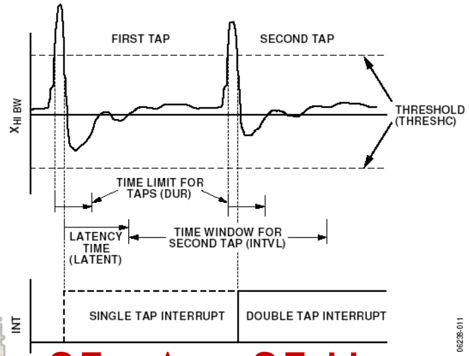
From Computer Desktop Encyclopedia™
Reproduced with permission.
© 2003 MEMSIC, Inc.



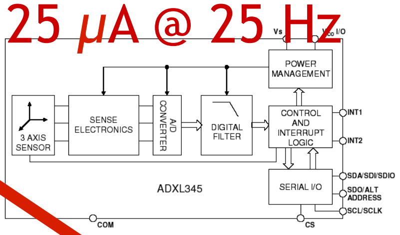
O(mA)

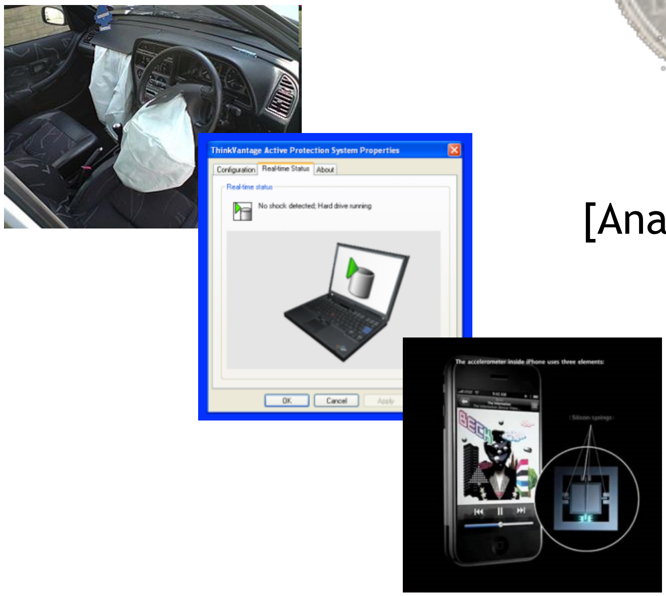


ADXL345
[Analog Devices, 2009]



25 µA @ 25 Hz

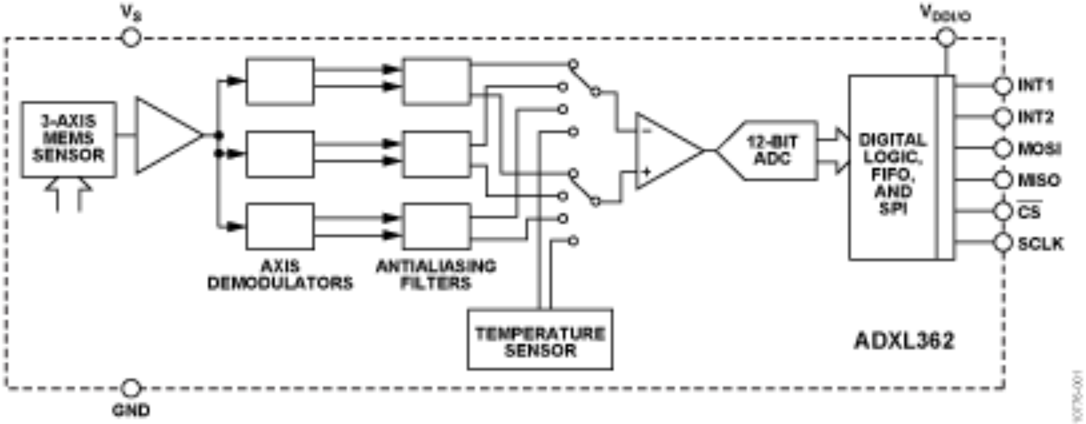




No shock detected; Hard drive running

10 µA @ 10 Hz @ 6 bits
[ST Microelectronics, annc. 2009]

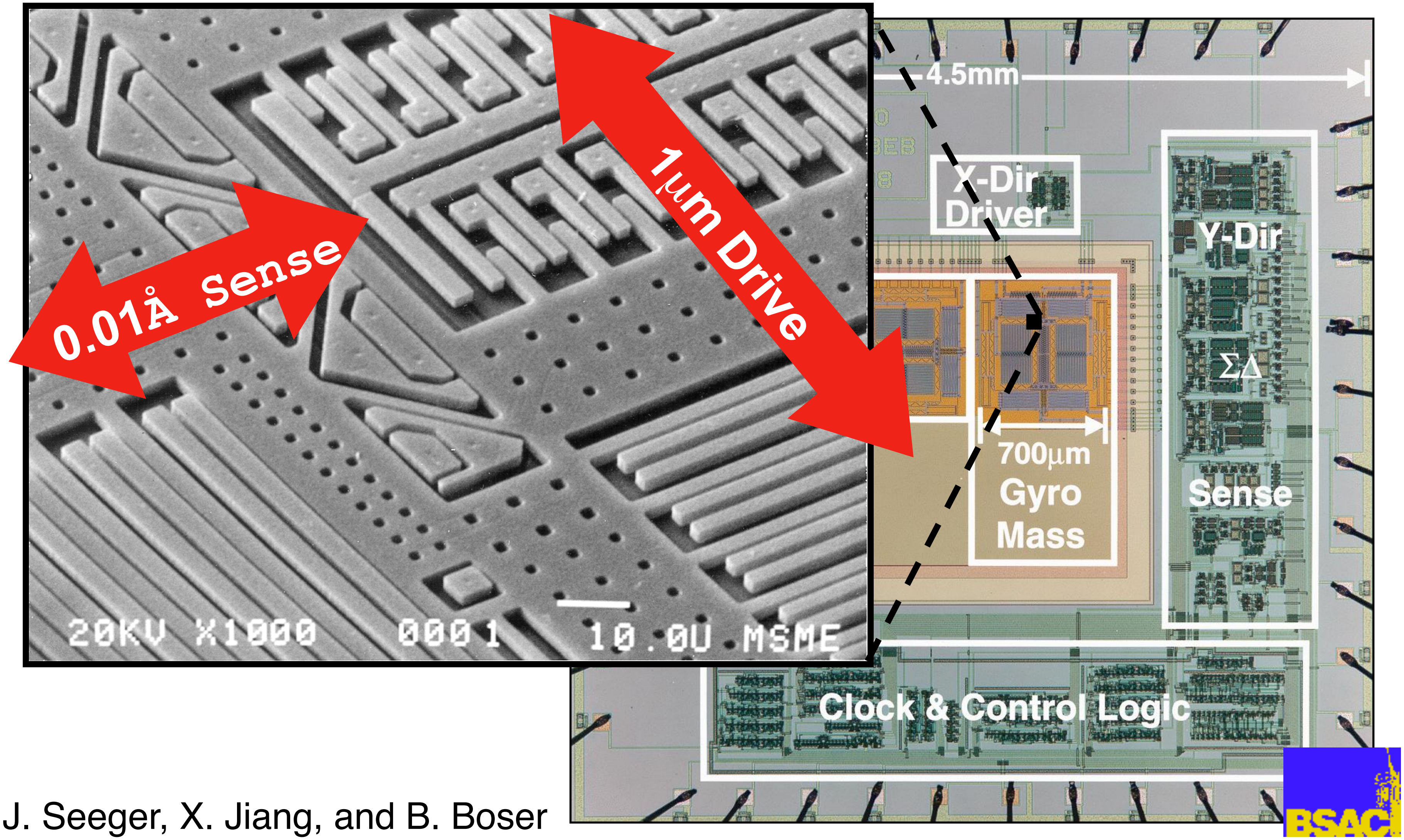
MEMS Accelerometer in 2012



1.8 μA @ 100 Hz @ 2V supply!

ADXL362
[Analog Devices, 2012]

MEMS Gyroscope Chip

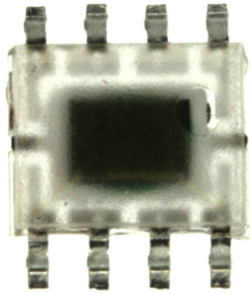


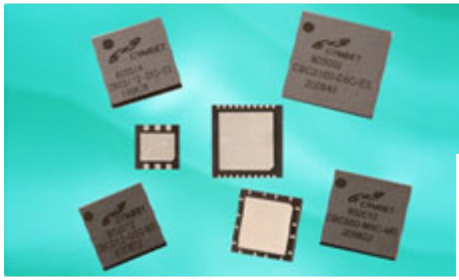
J. Seeger, X. Jiang, and B. Boser

Energy harvesting and storage: Small doesn't mean powerless…

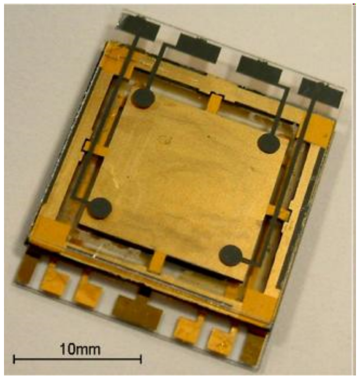

1st Annual Workshop on
October 22, 2009
MICRO POWER TECHNOLOGIES
Radisson Hotel, San Jose, CA


RF [Intel]

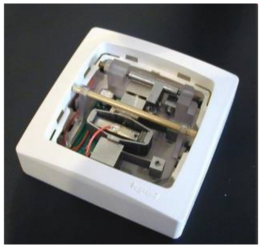

Clare Solar Cell

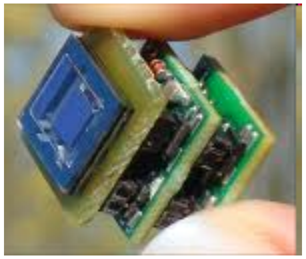


Thin-film batteries

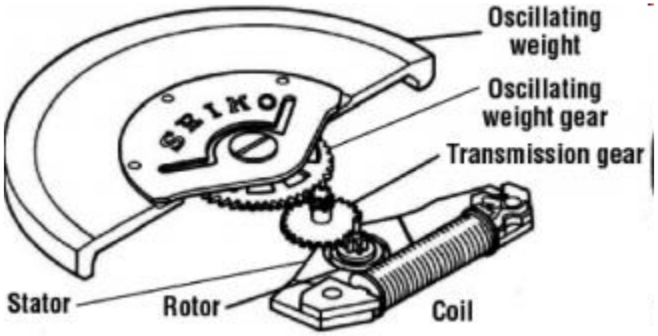


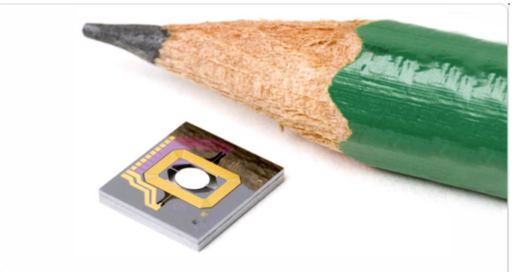
Electrostatic Energy Harvester [ICL]

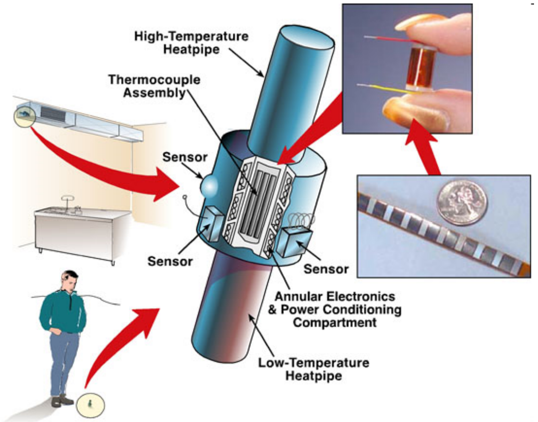

Shock Energy Harvesting CEDRAT Technologies



Piezoelectric [Holst/IMEC]

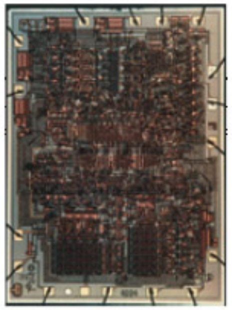






Thermoelectric Ambient Energy Harvester [PNNL]

Bell's Law, Take 2:
Corollary to the Laws of Scale



Intel® 4004 processor
Introduced 1971
Initial clock speed

108 KHz

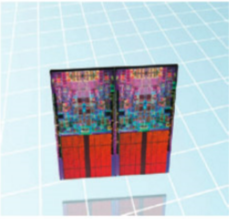
Number of transistors

2,300

Manufacturing technology

10μ

15x size decrease
40x transistors
55x smaller λ



Quad-Core Intel® Xeon® processor
Quad-Core Intel® Core™2 Extreme processor
Introduced 2006
Intel® Core™2 Quad processors
Introduced 2007
Initial clock speed

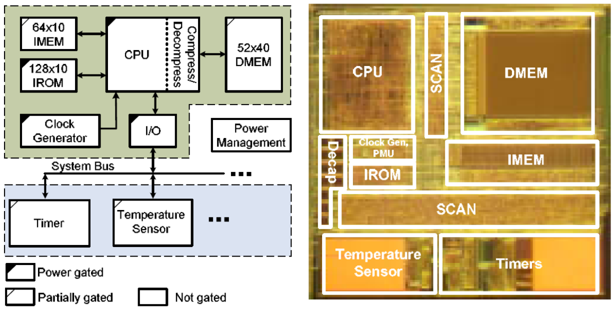
2.66 GHz

Number of transistors

582,000,000

Manufacturing technology

65nm



UMich Phoenix Processor
Introduced 2008
Initial clock speed

106 kHz @ 0.5V Vdd

Number of transistors

92,499

Manufacturing technology

0.18 μ

Photo credits: Intel, U. Michigan

Outline

Technology Trends

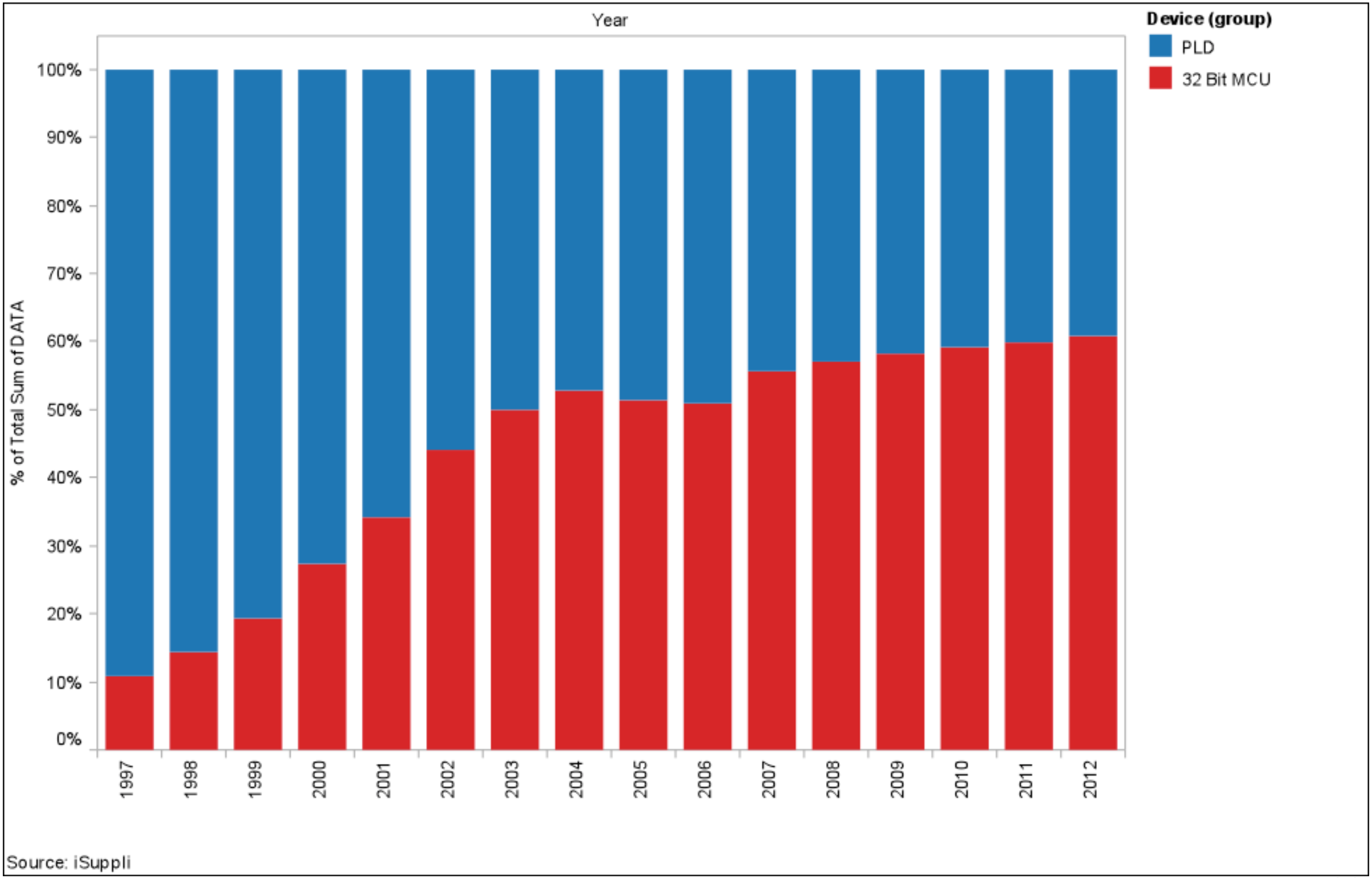
Design Questions

Course Administrivia

Tools Overview/ISA Start

Why study 32-bit MCUs and FPGAs?

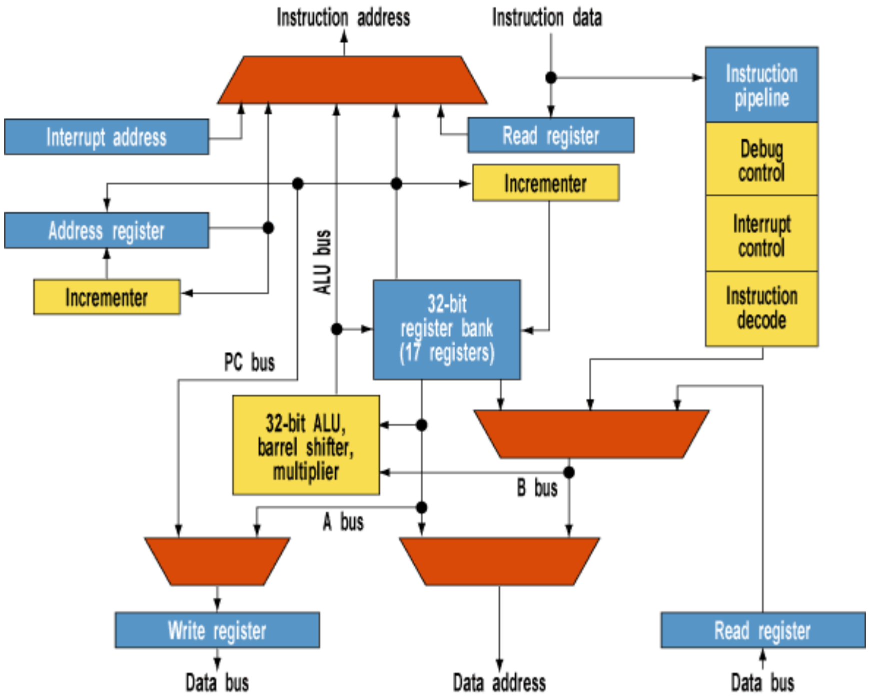
MCU-32 and PLDs are tied in embedded market share



Source: iSuppli

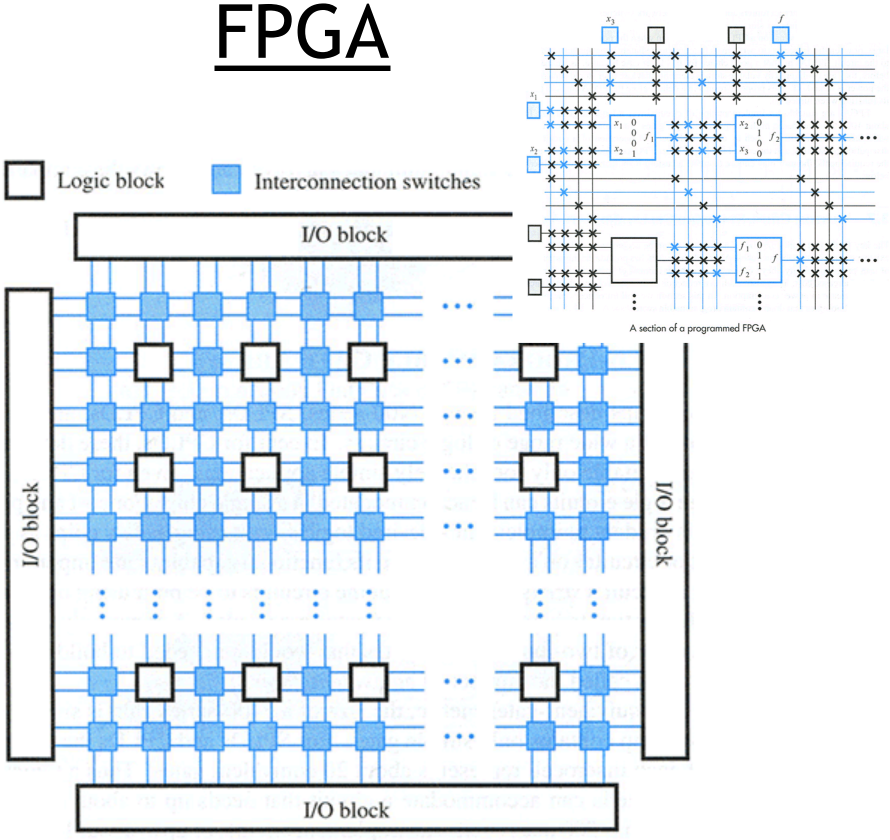
What distinguishes a Microprocessor from an FPGA?

MPU



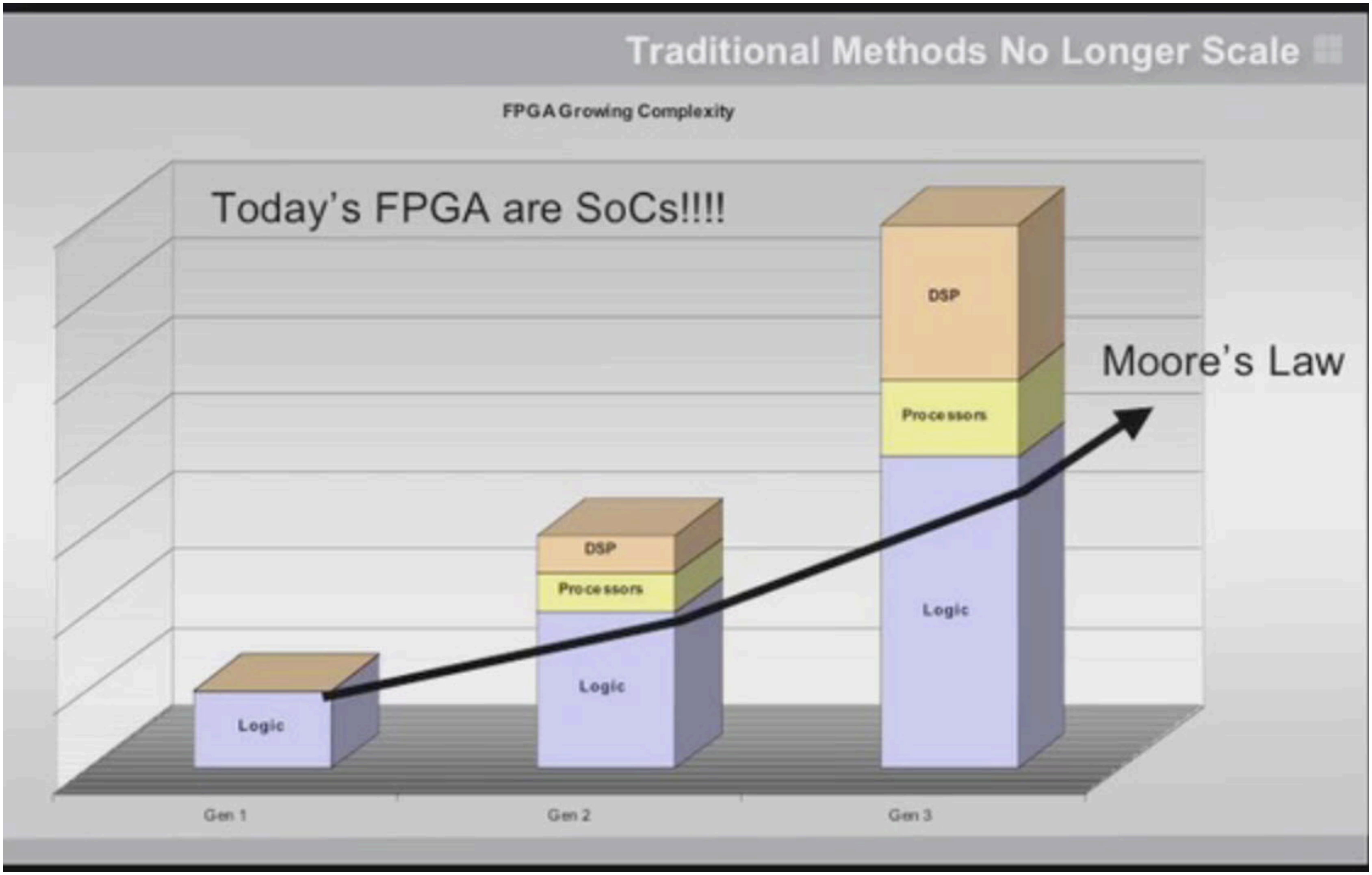
The Cortex M3's Thumbnail architecture looks like a conventional Arm processor. The differences are found in the Harvard architecture and the instruction decode that handles only Thumb and Thumb 2 instructions.

FPGA

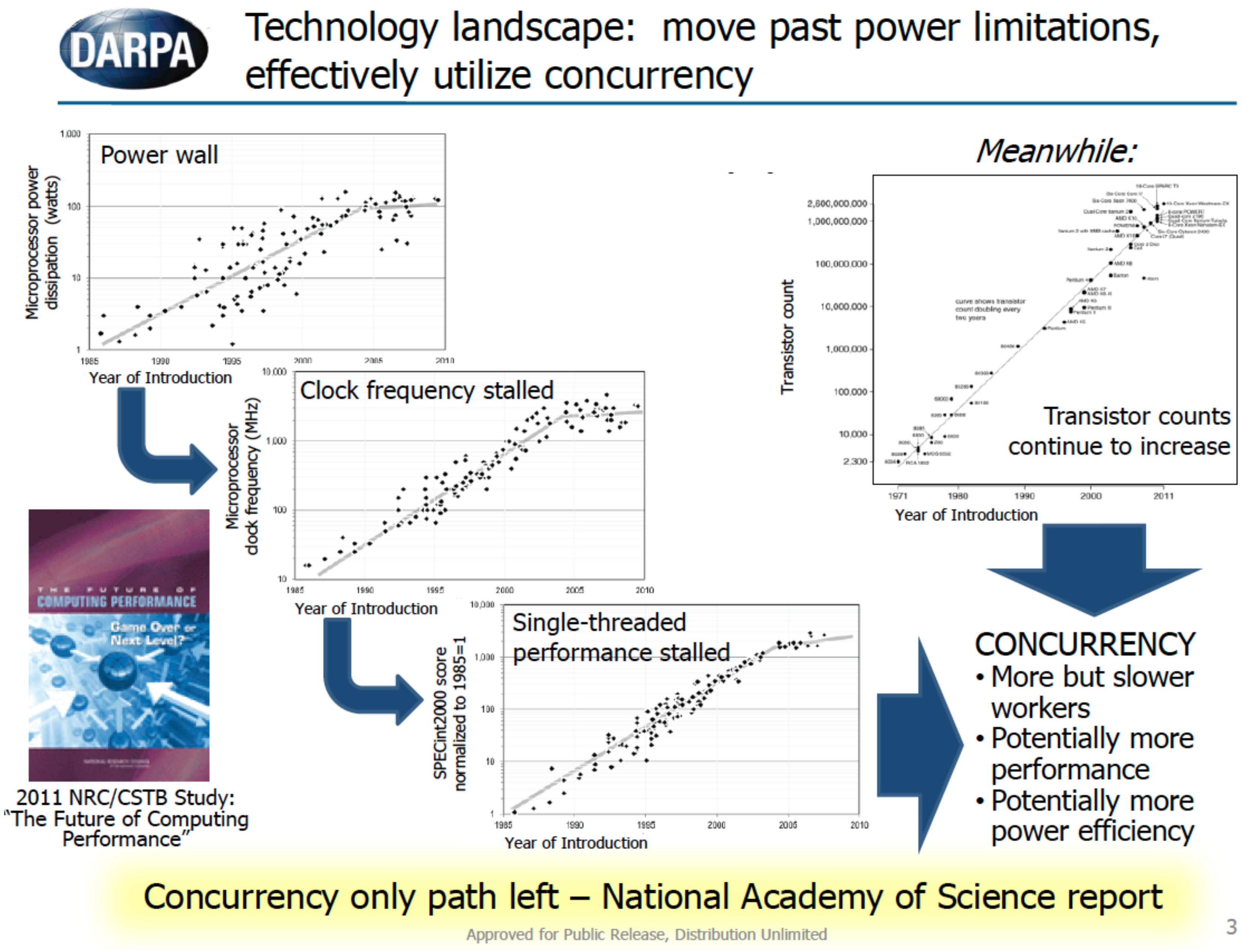


General structure of an FPGA

Modern FPGAs: best of both worlds!



Is the party really over?



Technology landscape: move past power limitations, effectively utilize concurrency

2011 NRC/CSTB Study: "The Future of Computing Performance"

CONCURRENCY
- More but slower workers
- Potentially more performance
- Potentially more power efficiency

Concurrency only path left – National Academy of Science report

Why study the ARM architecture (and the Cortex-M3 in particular)?

Lots of manufacturers ship ARM products

Actel®
POWER MATTERS

ATMEL®

CYPRESS
PERFORM

DUST NETWORKS

ember

NXP

ST

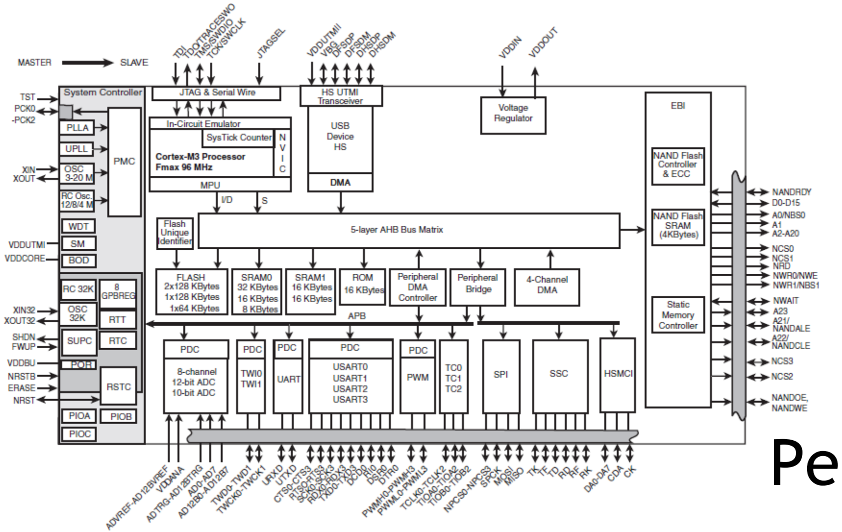
TEXAS INSTRUMENTS

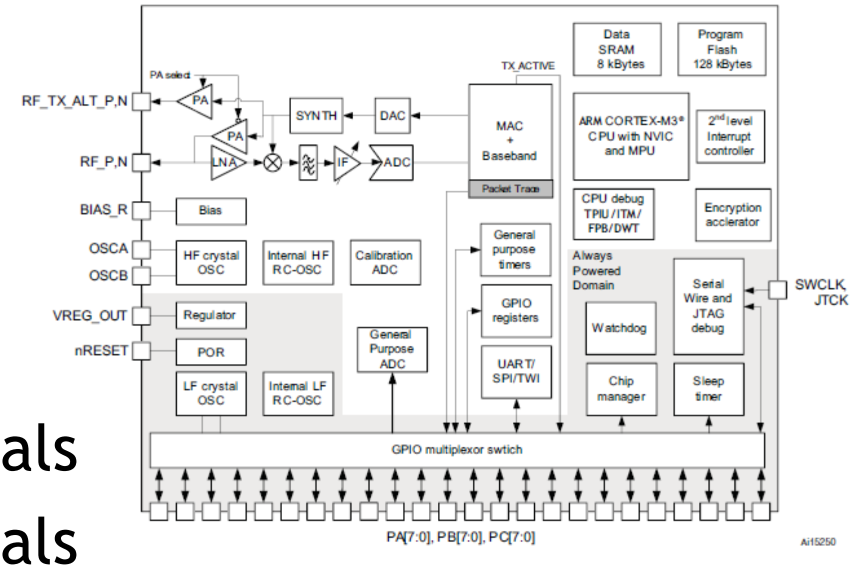
ARM is *the* big player

- ARM has a huge market share
  - As of 2011 ARM has chips in about 90% of the world's mobile handsets
  - As of 2010 ARM has chips in 95% of the smartphone market, 10% of the notebook market
    - Expected to hit 40% of the notebook market in 2015.
  - Heavy use in general embedded systems.
    - Cheap to use
      - ARM appears to get an average of 8¢ per device (averaged over cheap and expensive chips).
    - Flexible
      - Spin your own designs.

What differentiates these products from one another?

The difference is…

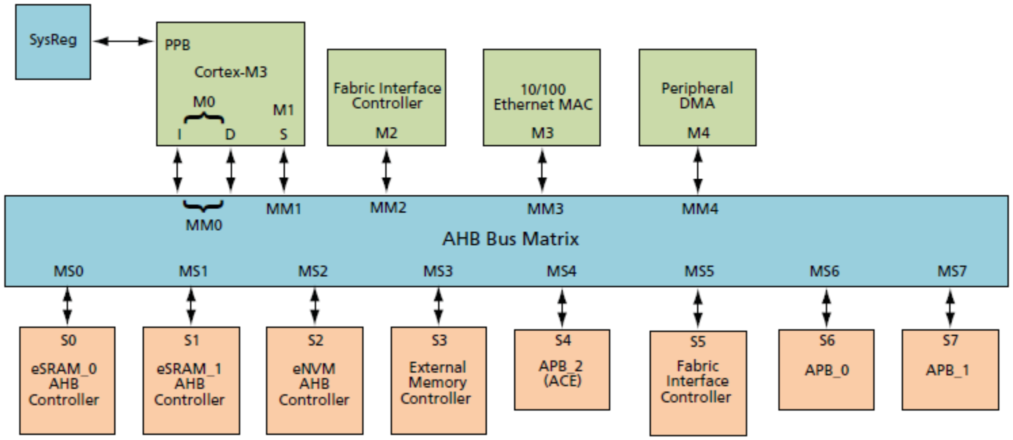




Peripherals

Peripherals

Peripherals



Outline

Technology Trends

Design Questions

Course Overview

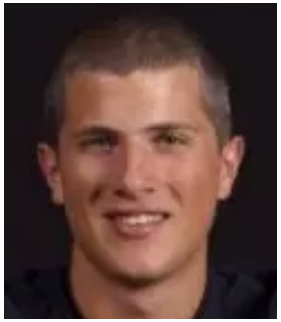
Tools Overview/ISA Start

F' 14 Instructional Staff
(see homepage for contact info, office hours)

Prabal
Dutta
Instructor

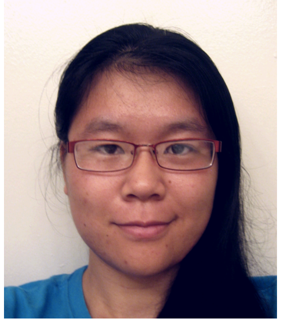
Matt
Smith
Lab Instructor

Chris
Fulara
IA



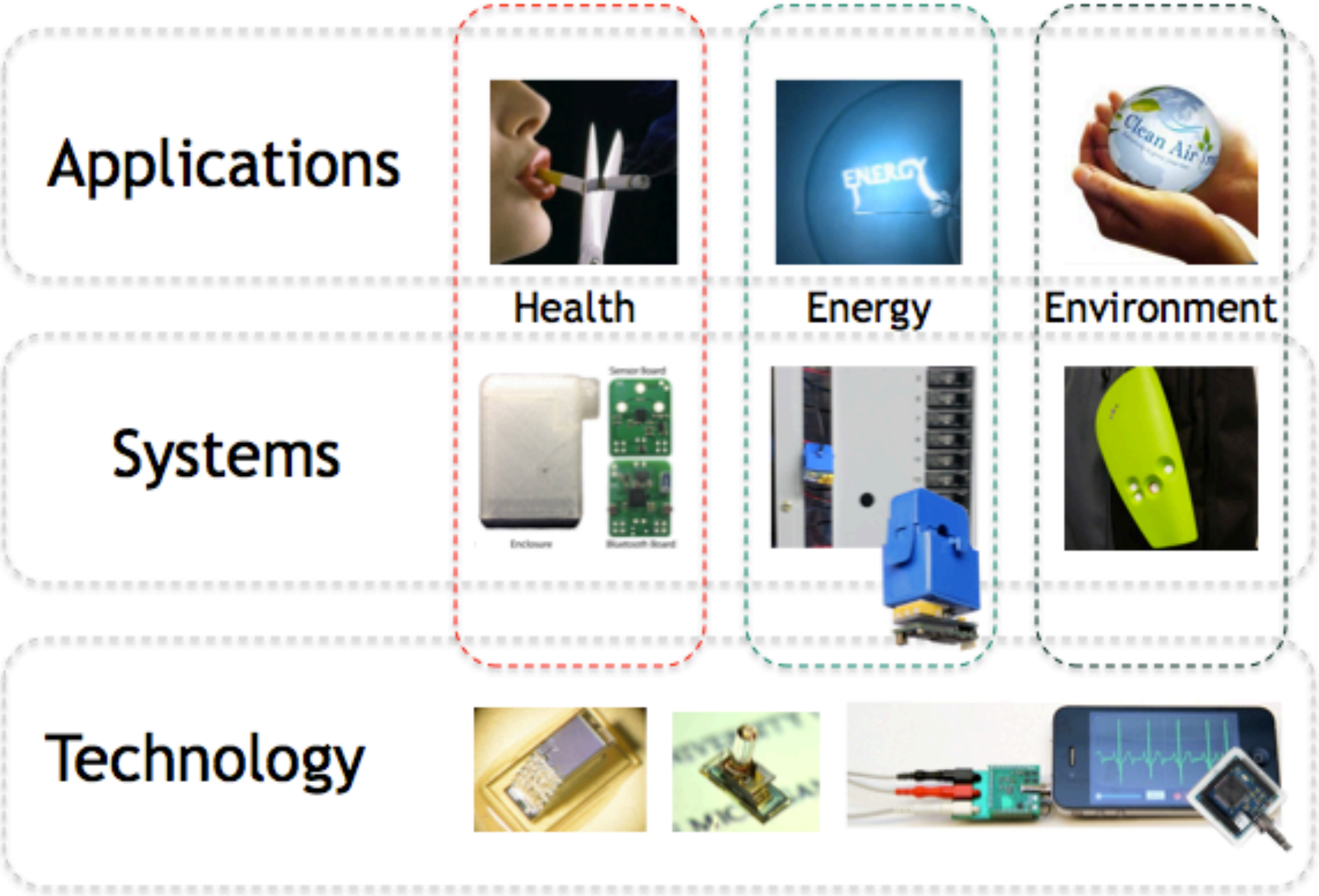
Chris
Heyer
IA

Ryan
Wooster
IA



Dili
Hu
Grader

My research interests



Applications

Systems

Technology

Health

Energy

Environment

Course goals

- *Learn to implement* embedded systems including hardware/software interfacing.
- *Learn to design* embedded systems and how to think about embedded software and hardware.
- *Design and build* non-trivial projects involving both hardware and software.

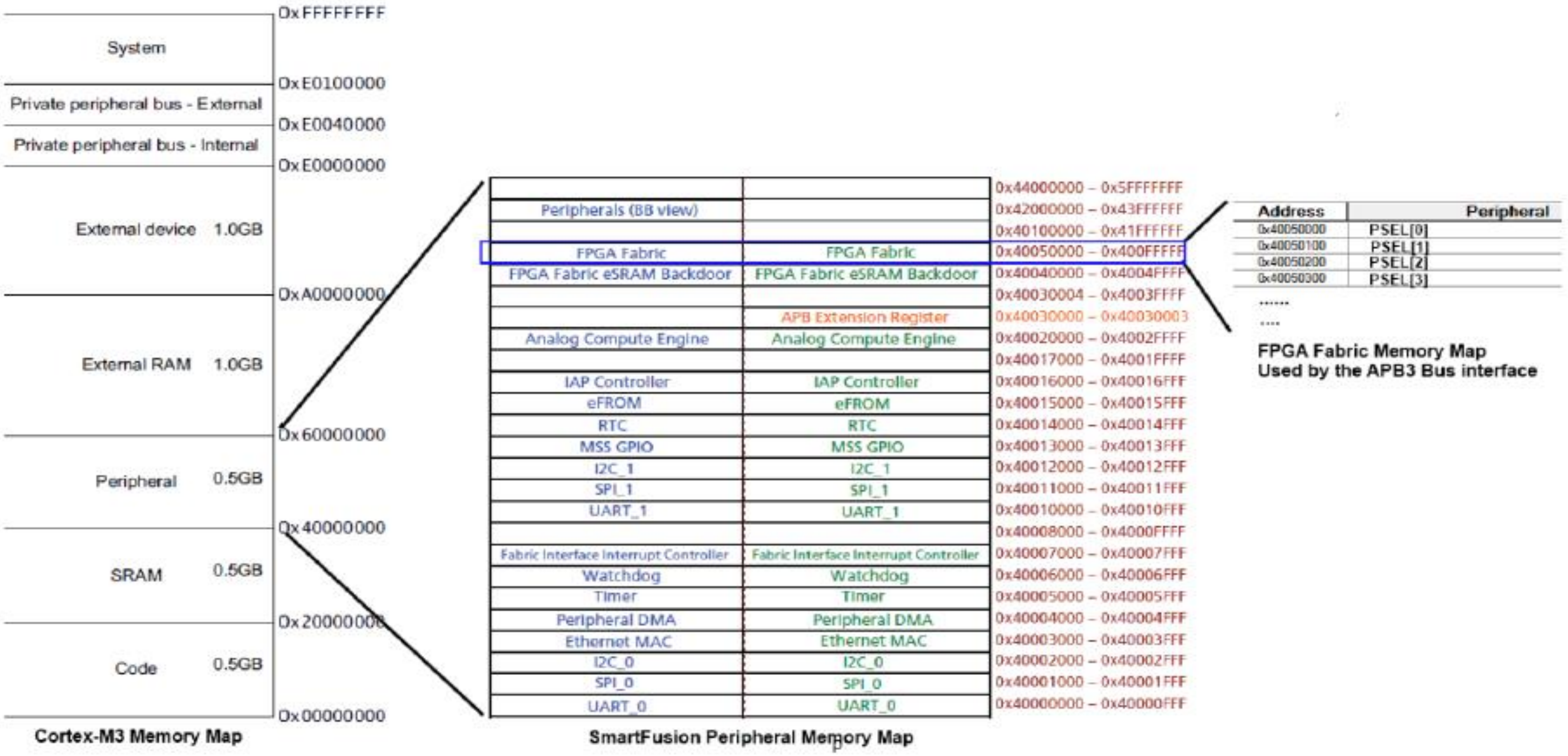
Prerequisites

- EECS 270: Introduction to Logic Design
  - Combinational and sequential logic design
  - Logic minimization, propagation delays, timing
- EECS 280: Programming and Intro Data Structures
  - C programming
  - Algorithms (e.g. sort) and data structures (e.g. lists)
- EECS 370: Introduction to Computer Organization
  - Basic computer architecture
  - CPU control/datapath, memory, I/O
  - Compiler, assembler

Topics

- Memory-mapped I/O
  - The idea of using memory addressed to talk to input and output devices.
    - Switches, LEDs, hard drives, keyboards, motors
- Interrupts
  - How to get the processor to become “event driven” and react to things as they happen.
- Working with analog signals
  - The real world isn’t digital!
- Common peripheral devices and interfaces
  - Serial buses, timers, etc.

Example: Memory-mapped I/O



Cortex-M3 Memory Map

		0x44000000 – 0x5FFFFFFF
Peripherals (BB view)		0x42000000 – 0x43FFFFFF
		0x40100000 – 0x41FFFFFF
FPGA Fabric	FPGA Fabric	0x40050000 – 0x400FFFFF
FPGA Fabric eSRAM Backdoor	FPGA Fabric eSRAM Backdoor	0x40040000 – 0x4004FFFF
		0x40030004 – 0x4003FFFF
	APB Extension Register	0x40030000 – 0x40030003
Analog Compute Engine	Analog Compute Engine	0x40020000 – 0x4002FFFF
		0x40017000 – 0x4001FFFF
IAP Controller	IAP Controller	0x40016000 – 0x40016FFF
eFROM	eFROM	0x40015000 – 0x40015FFF
RTC	RTC	0x40014000 – 0x40014FFF
MSS GPIO	MSS GPIO	0x40013000 – 0x40013FFF
I2C_1	I2C_1	0x40012000 – 0x40012FFF
SPI_1	SPI_1	0x40011000 – 0x40011FFF
UART_1	UART_1	0x40010000 – 0x40010FFF
		0x40008000 – 0x4000FFFF
Fabric Interface Interrupt Controller	Fabric Interface Interrupt Controller	0x40007000 – 0x40007FFF
Watchdog	Watchdog	0x40006000 – 0x40006FFF
Timer	Timer	0x40005000 – 0x40005FFF
Peripheral DMA	Peripheral DMA	0x40004000 – 0x40004FFF
Ethernet MAC	Ethernet MAC	0x40003000 – 0x40003FFF
I2C_0	I2C_0	0x40002000 – 0x40002FFF
SPI_0	SPI_0	0x40001000 – 0x40001FFF
UART_0	UART_0	0x40000000 – 0x40000FFF

SmartFusion Peripheral Memory Map

Address	Peripheral
0x40050000	PSEL[0]
0x40050100	PSEL[1]
0x40050200	PSEL[2]
0x40050300	PSEL[3]

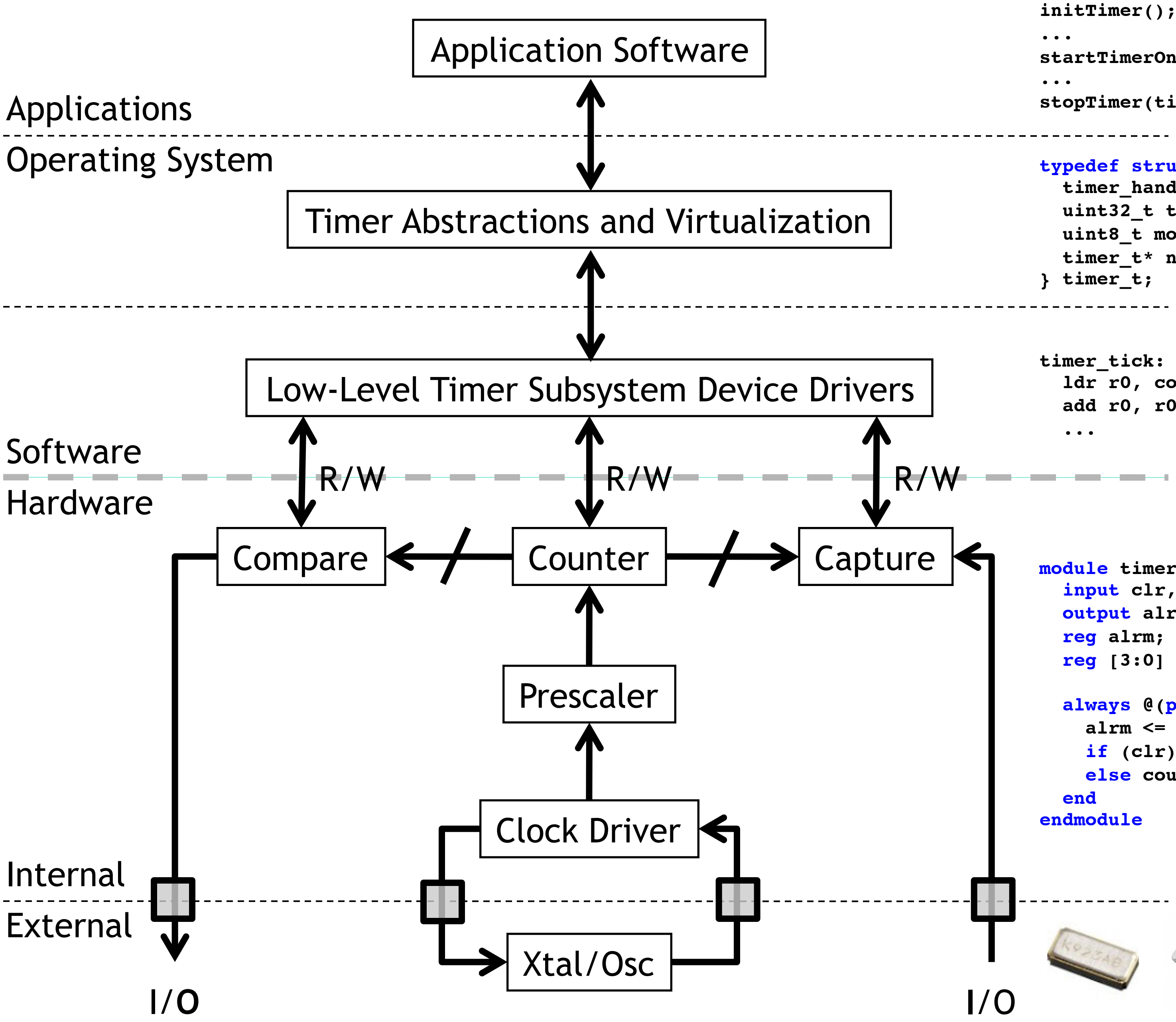
......

....

FPGA Fabric Memory Map
Used by the APB3 Bus interface

- This is *important*.
  - It means our software can tell the hardware what to do.
    - In lab 3 you'll design hardware on an FPGA which will control a motor.
      - But more importantly, that hardware will be designed so the software can tell the hardware exactly what to do with the motor. All by simply writing to certain memory locations!
  - In the same way, the software can read memory locations to access data from sensors etc…

Example: Anatomy of a timer system



```
...
timer_t timerX;
initTimer();
...
startTimerOneShot(timerX, 1024);
...
stopTimer(timerX);
```

```
typedef struct timer {
  timer_handler_t handler;
  uint32_t time;
  uint8_t mode;
  timer_t* next_timer;
} timer_t;
```

```
timer_tick:
  ldr r0, count;
  add r0, r0, #1
  ...
```

```
module timer(clr, ena, clk, alrm);
  input clr, ena, clk;
  output alrm;
  reg alrm;
  reg [3:0] count;

  always @(posedge clk) begin
    alrm <= 0;
    if (clr) count <= 0;
    else count <= count+1;
  end
endmodule
```

Grades

```
Item                 Weight
======               =========
Labs (7)             25%
Project              25%
Exams                35% (15% midterm; 20% final)
HW/Guest talks       10%
Oral presentation     5%
```

- Project & Exams tend to be the differentiators
- Class median is generally a B+

Time

- Assume you are going to spend a lot of time in this class.
  - 2-3 hours/week in lecture (we cancel a few classes during project time)
  - 8-12 hours/week working in lab
    - *Expect more during project time; some labs are a bit shorter.*
  - ~20 hours (total) working on homework
  - ~20 hours (total) studying for exams.
  - ~8 hour (total) on your oral presentation
- Averages out to about 15-20 hours/week pre-project and about 20 during the project…
  - This is more than we'd like, but we've chosen to go with state-of-the-art tools, and those generally have a steep learning curve.

Labs

- Start TODAY!
- 7 labs, 8 weeks, groups of 2
    1. FPGA + Hardware Tools
    2. MCU + Software Tools
    3. Memory + Memory-Mapped I/O
    4. Interrupts
    5. Timers and Counters
    6. Serial Bus Interfacing
    7. Data Converters (e.g. ADCs/DACs)

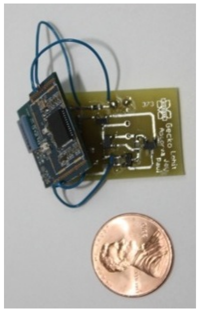
- Labs are very time consuming.
    - As noted, students estimated 8-12 hours per lab with one lab (which varied by group) taking longer.

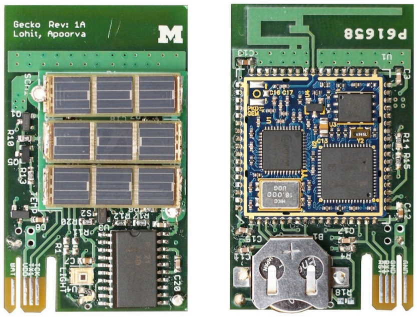
Open-Ended Project

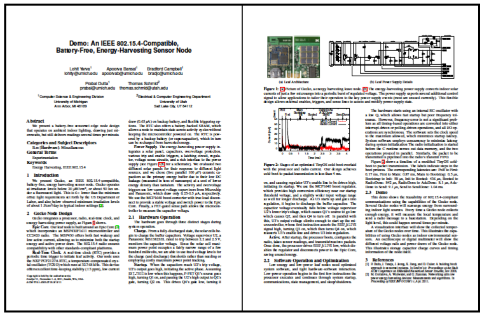
- Goal: learn how to build embedded systems
  - By building an embedded system
  - Work in teams of 2 to 4
  - You design your own project
- The major focus of the last third of the class.
  - Labs will be done and we will cancel some lectures and generally try to keep you focused
- Important to start early.
  - After all the effort in the labs, it's tempting to slack for a bit. The best projects are those that get going right away (or even earlier)
- Some project lead to undergraduate research

Sample projects from F’10 and their results

- Energy-harvesting sensors → Sensys demo, IPSN paper, TI project, Master’s thesis







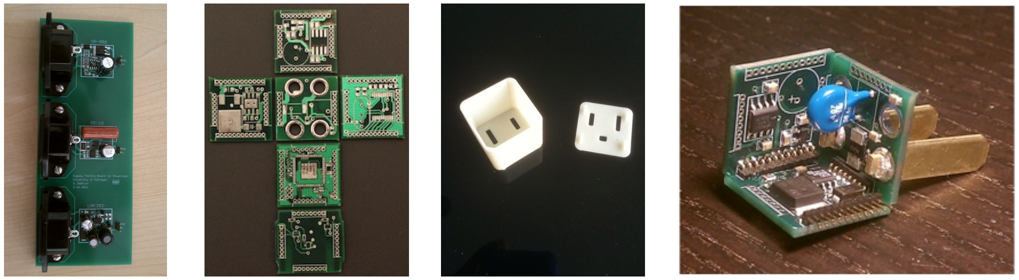
Demo: An IEEE 802.15.4-Compatible, Battery-Free, Energy-Harvesting Sensor Node

Lohit Yerva[1] Apoorva Bansal[1] Bradford Campbell[1]
lohity@umich.edu apoorvab@umich.edu bradjc@umich.edu
Prabal Dutta[1] Thomas Schmid[2]
prabal@umich.edu thomas.schmid@utah.edu

[1]Computer Science & Engineering Division
University of Michigan
Ann Arbor, MI 48109

[2]Electrical & Computer Engineering Department
University of Utah
Salt Lake City, UT 84112

- Wireless AC Power Meter → SURE, IPSN demo, NSF GRFP, SenSys paper, Grad School



Letters of recommendation for graduate school

- Grad school apps will require supporting letters
- Faculty write letters and read “coded” letters

- Strong letters give evidence of research ability
- Strong letters can really help your case

- Weak letters are vague and give class standing
- Weak letters are useless (or even worse)

- Want a strong letter?
  - Do well in this class
  - Pull off an impressive project
  - Continue class project as independent research in W’15

Homework

- Start TODAY!
- 4-5 assignments
  - A few “mini” assignments
    - Mainly to get you up to speed on lab topics
  - A few “standard” assignments
    - Hit material we can’t do in lab.
- Also a small part is for showing up to guest lectures
- And a tiny bit for doing completing evaluations

Midterm and Final Exams

- Midterm (Thu, Oct 16, 2014 from 10:30am-12:00pm)
  - Emphasize problem solving fundamentals

- Final (Tue, Dec 16, 2014 from 1:30-3:30pm)
  - Cumulative topics w/ experience of projects
  - Some small amount of material from presentations

Looking for me?

- Nominal Office Hours
    - Tuesdays: 1:30-3:00pm in 4773 BBB
    - Sometimes in lab sections
- Traveling next week so
    - Guest lectures on Tue 9/9 and Thu 9/11
    - No office hours next week

Outline

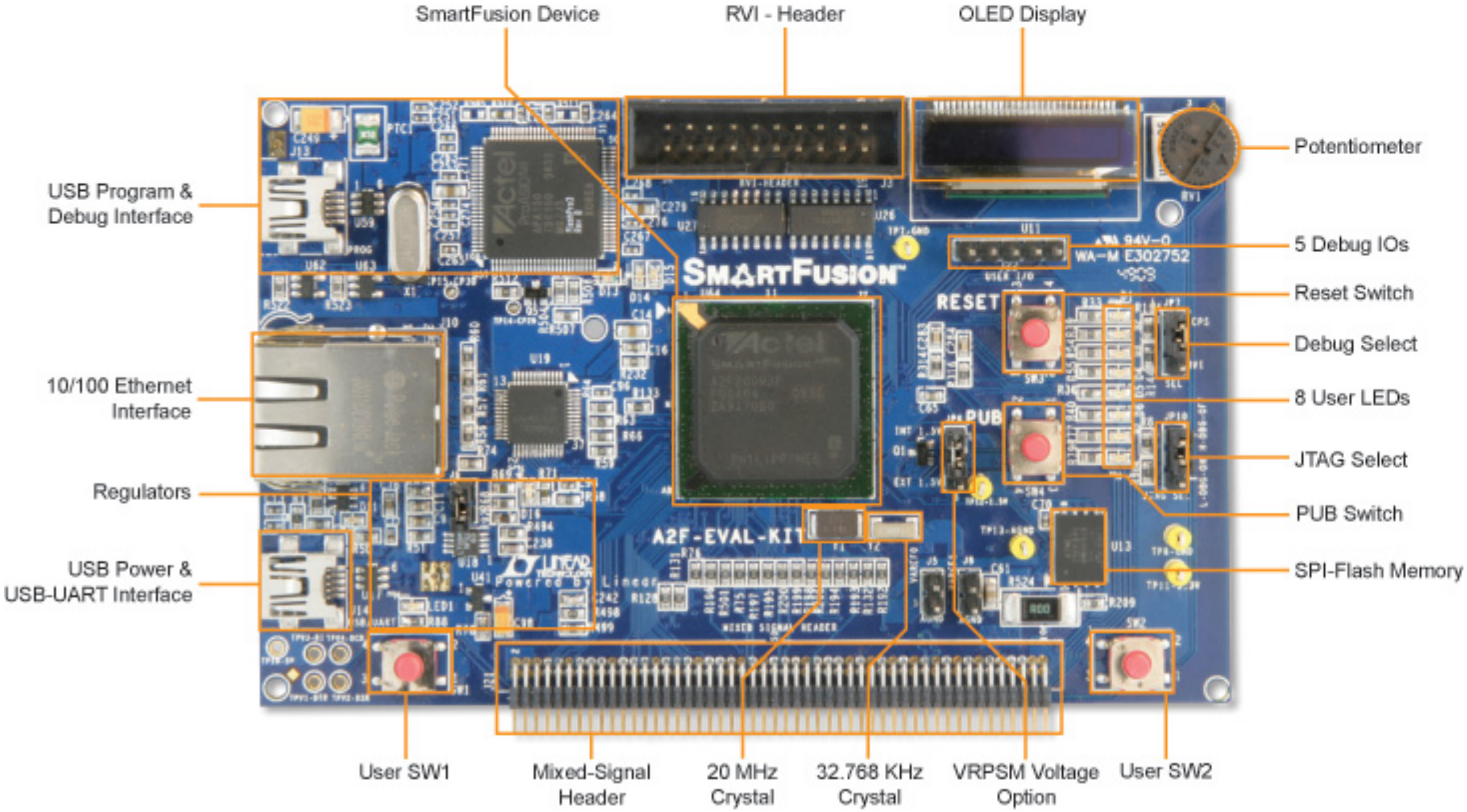
Technology Trends

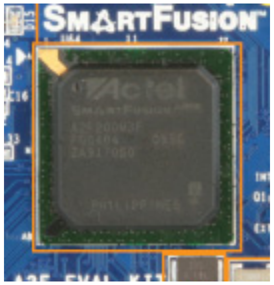
Design Questions

Course Overview

Tools Overview/ISA Start

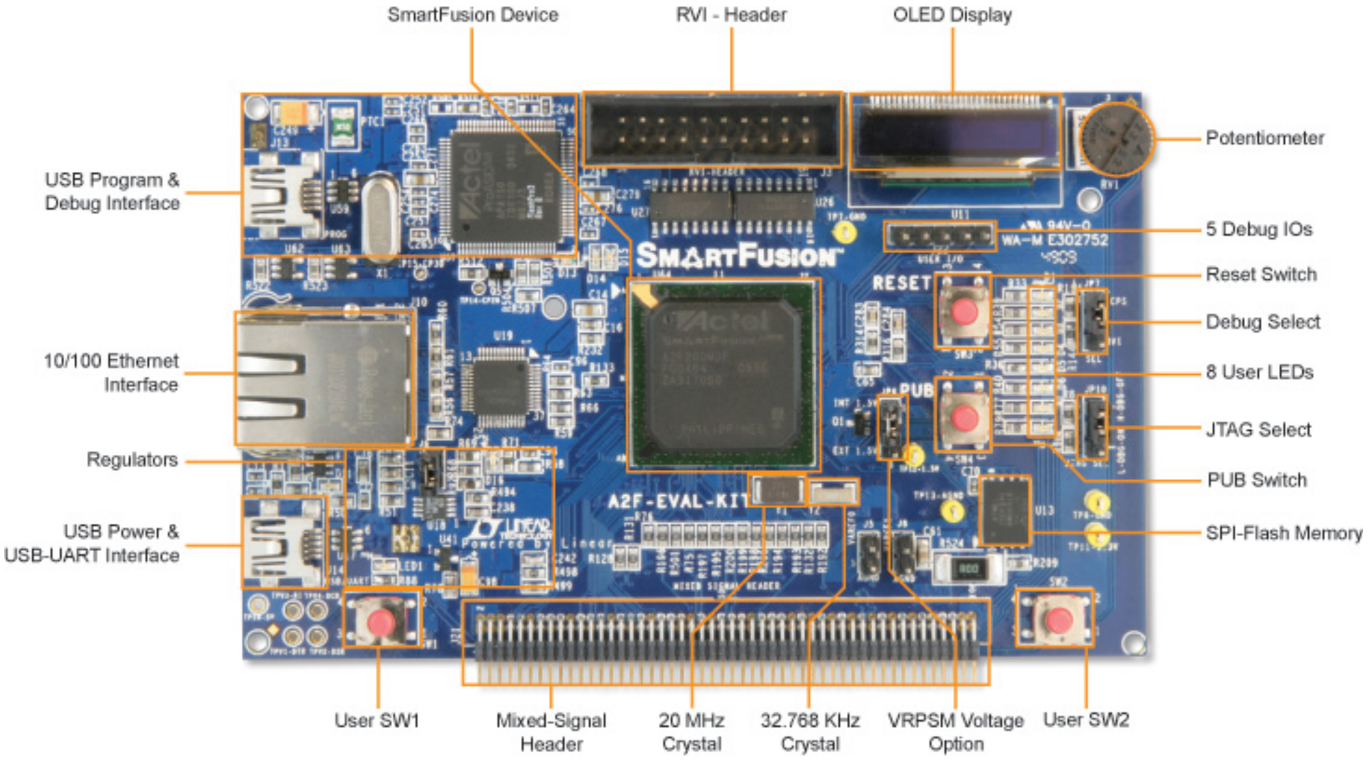
We are using Actel's SmartFusion Evaluation Kit



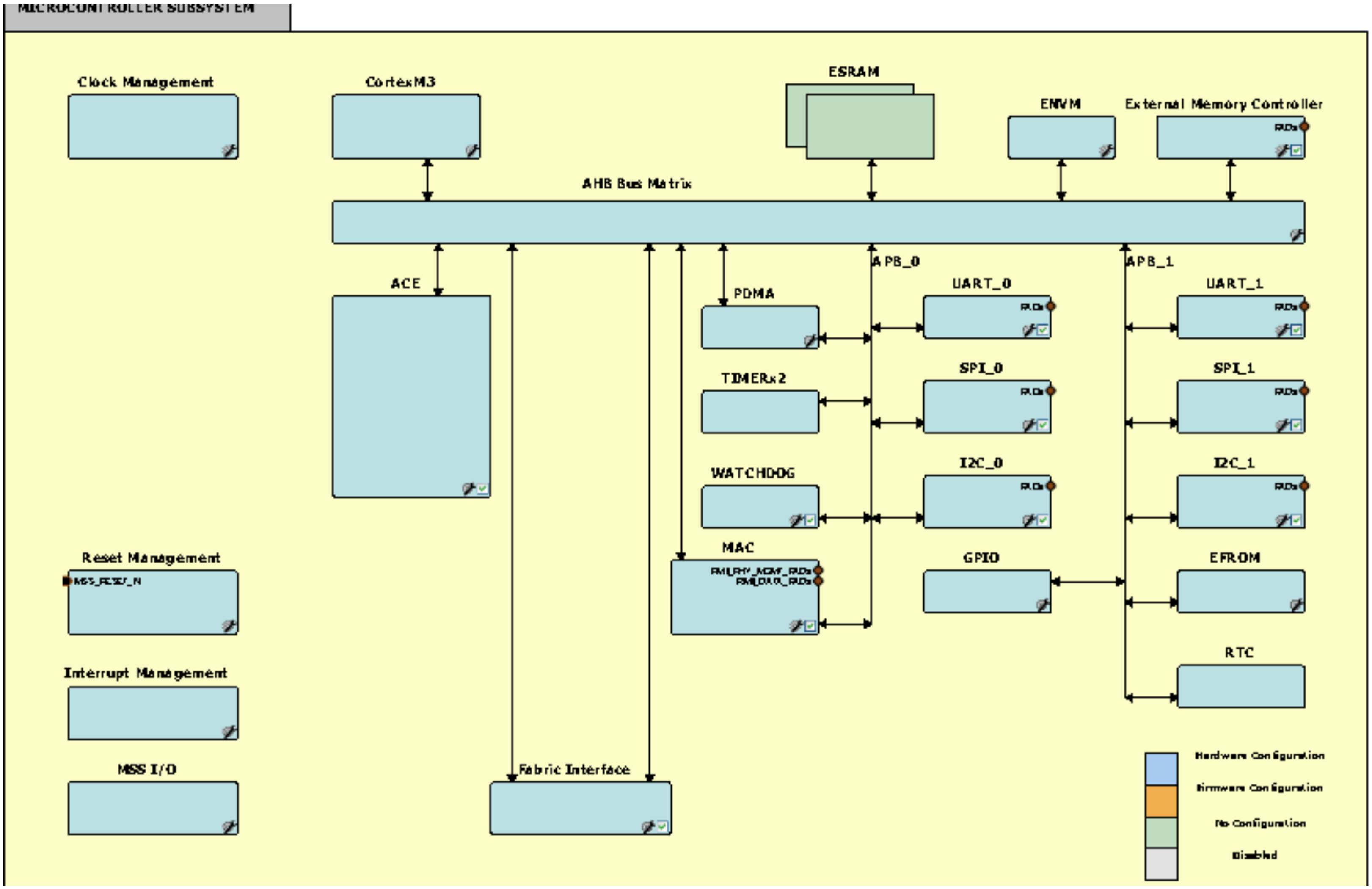


A2F200M3F-FGG484ES
  - 200,000 System FPGA gates, 256 KB flash memory, 64 KB SRAM, and additional distributed SRAM in the FPGA fabric and external memory controller
  - Peripherals include Ethernet, DMAs, I^{2}Cs, UARTs, timers, ADCs, DACs and additional analog resources

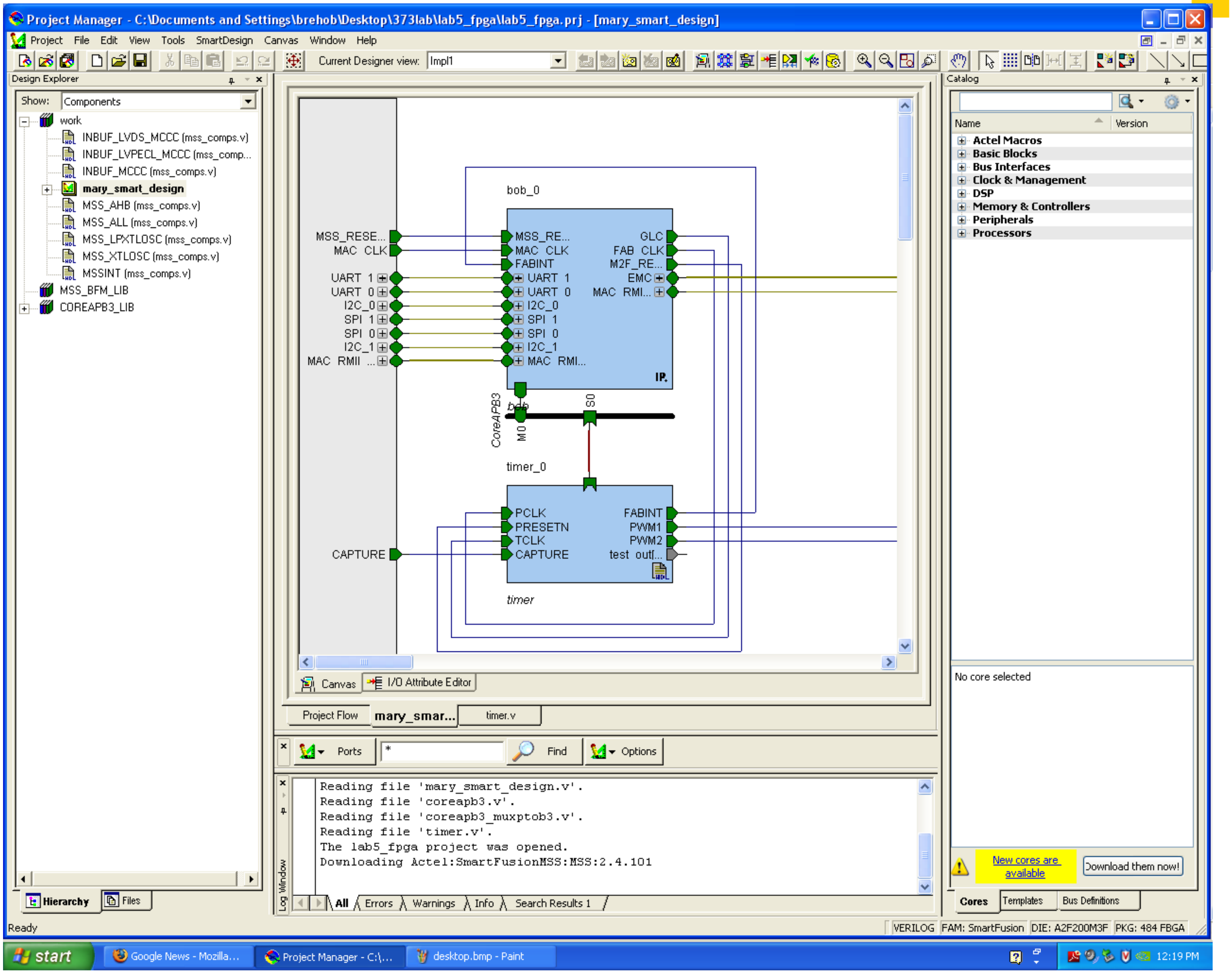
- USB connection for programming and debug from Actel's design tools
- USB to UART connection to UART_0 for HyperTerminal examples
- 10/100 Ethernet interface with on-chip MAC and external PHY
- Mixed-signal header for daughter card support



FPGA work

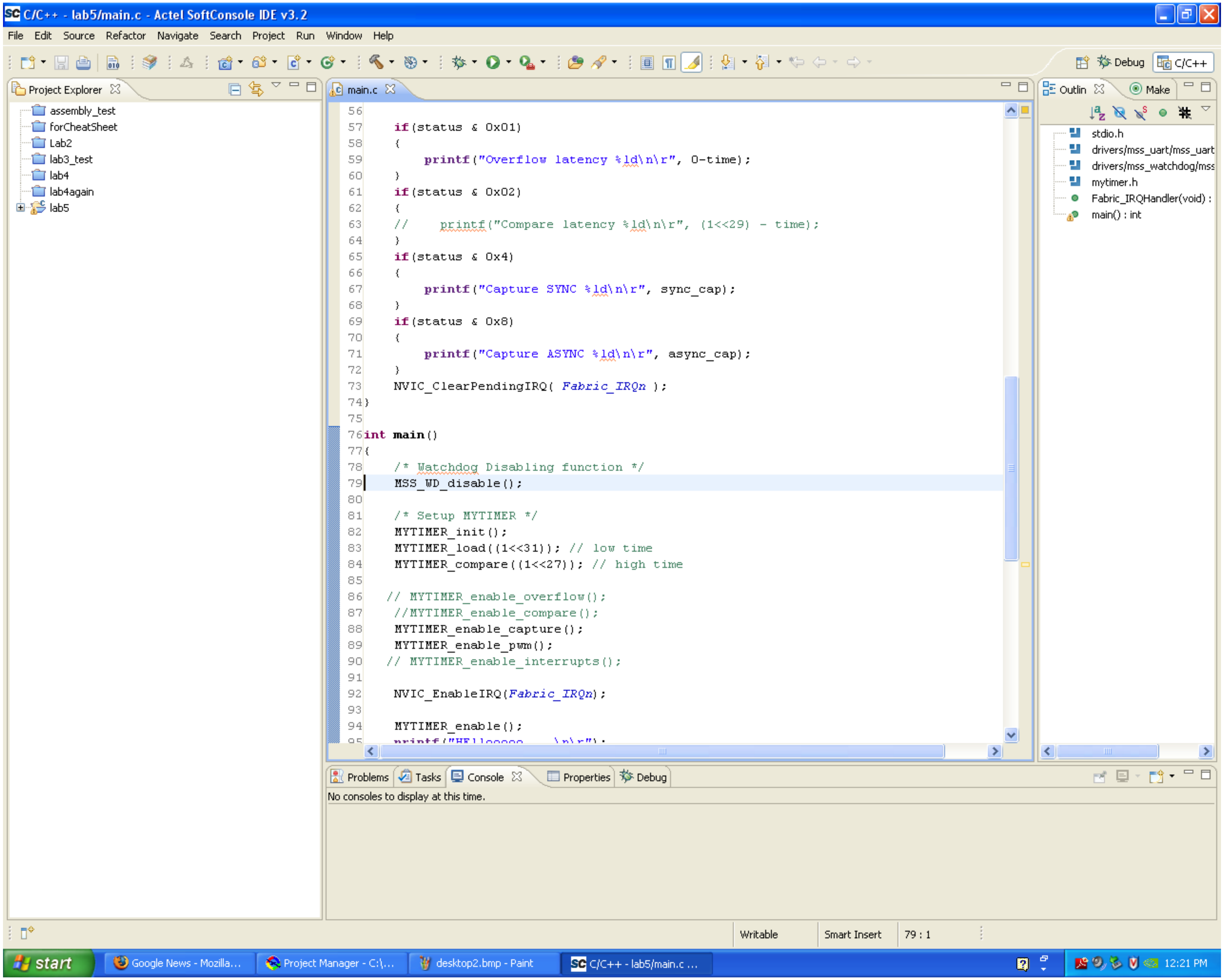

MICROCONTROLLER SUBSYSTEM
Clock Management
CortexM3
ESRAM
ENVM
External Memory Controller
PADs
AHB Bus Matrix
ACE
PDMA
APB_0
APB_1
UART_0
UART_1
TIMERx2
SPI_0
SPI_1
WATCHDOG
I2C_0
I2C_1
MAC
RMII_PHY_MGMT_PADs
RMII_DATA_PADs
GPIO
EFROM
Reset Management
MSS_RESET_N
RTC
Interrupt Management
MSS I/O
Fabric Interface
Hardware Configuration
Firmware Configuration
No Configuration
Disabled

“Smart Design” configurator



```
Reading file 'mary_smart_design.v'.
Reading file 'coreapb3.v'.
Reading file 'coreapb3_muxptob3.v'.
Reading file 'timer.v'.
The lab5_fpga project was opened.
Downloading Actel:SmartFusionMSS:MSS:2.4.101
```

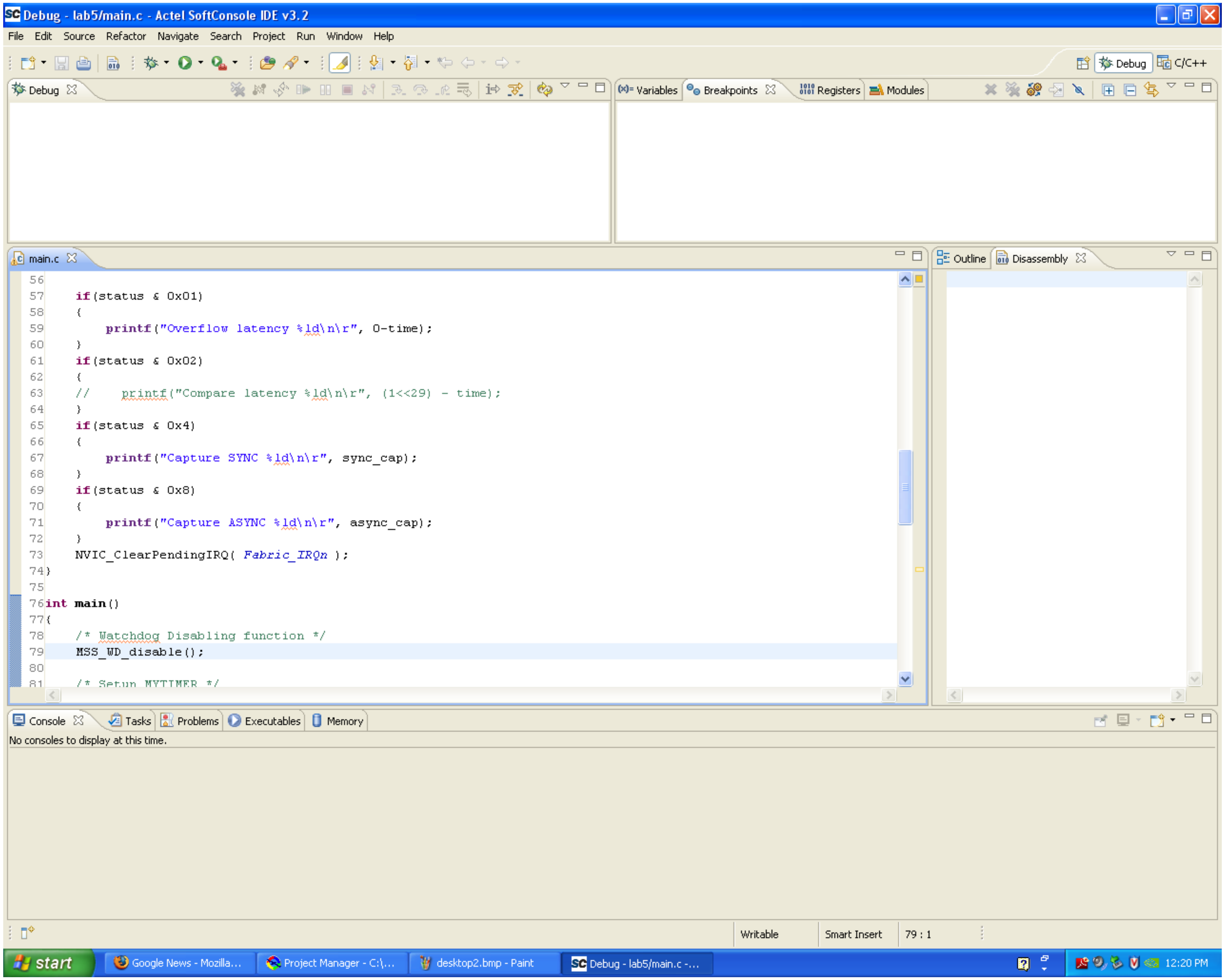
Eclipse-based “Actel SoftConsole IDE”



```c

    if(status & 0x01)
    {
        printf("Overflow latency %ld\n\r", 0-time);
    }
    if(status & 0x02)
    {
    //    printf("Compare latency %ld\n\r", (1<<29) - time);
    }
    if(status & 0x4)
    {
        printf("Capture SYNC %ld\n\r", sync_cap);
    }
    if(status & 0x8)
    {
        printf("Capture ASYNC %ld\n\r", async_cap);
    }
    NVIC_ClearPendingIRQ( Fabric_IRQn );
}

int main()
{
    /* Watchdog Disabling function */
    MSS_WD_disable();

    /* Setup MYTIMER */
    MYTIMER_init();
    MYTIMER_load((1<<31)); // low time
    MYTIMER_compare((1<<27)); // high time

   // MYTIMER_enable_overflow();
    //MYTIMER_enable_compare();
    MYTIMER_enable_capture();
    MYTIMER_enable_pwm();
   // MYTIMER_enable_interrupts();

    NVIC_EnableIRQ(Fabric_IRQn);

    MYTIMER_enable();
```

No consoles to display at this time.

Debugger is GDB-based. Includes command line. Works really quite well.



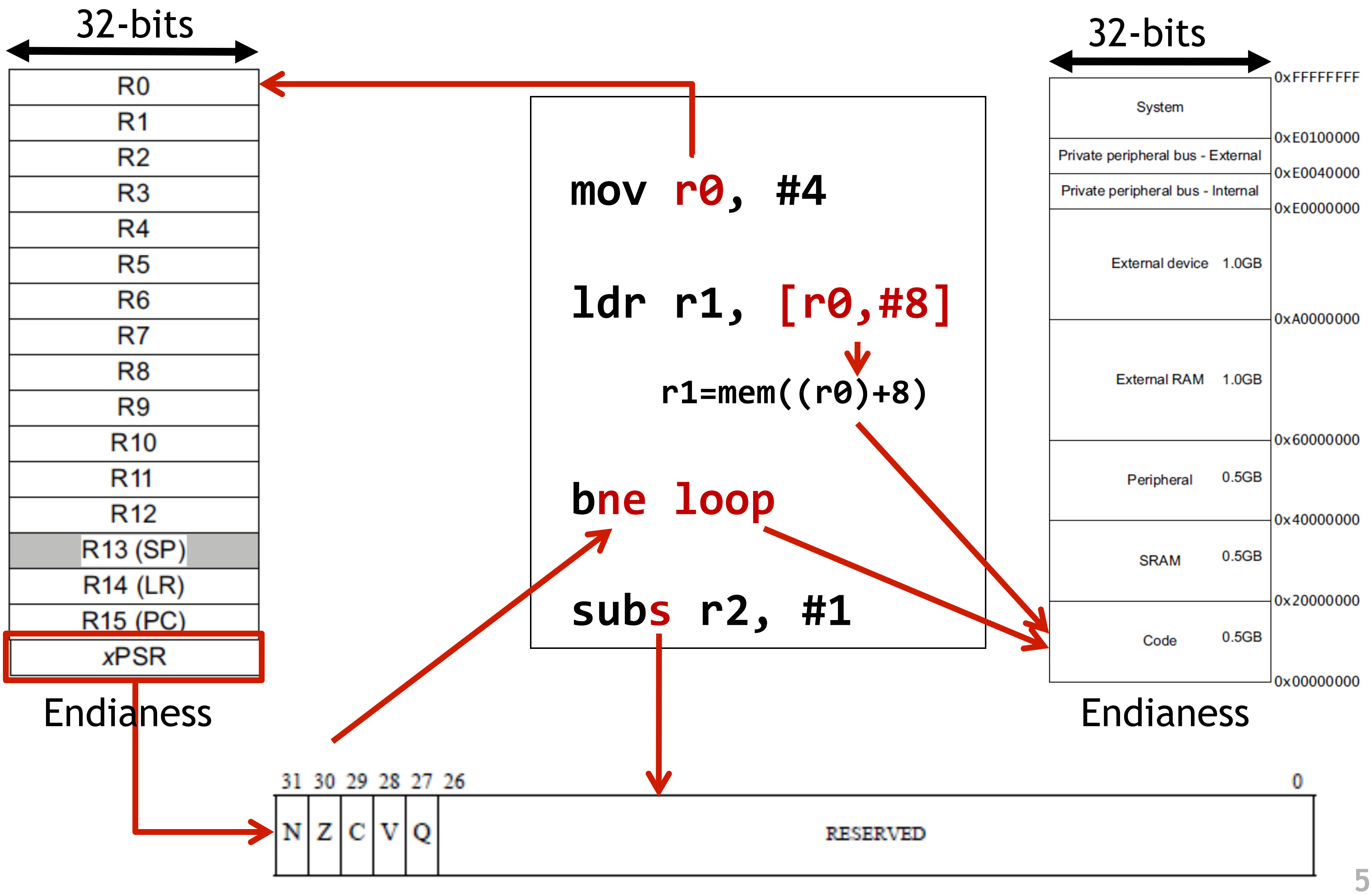
```c

    if(status & 0x01)
    {
        printf("Overflow latency %ld\n\r", 0-time);
    }
    if(status & 0x02)
    {
    //    printf("Compare latency %ld\n\r", (1<<29) - time);
    }
    if(status & 0x4)
    {
        printf("Capture SYNC %ld\n\r", sync_cap);
    }
    if(status & 0x8)
    {
        printf("Capture ASYNC %ld\n\r", async_cap);
    }
    NVIC_ClearPendingIRQ( Fabric_IRQn );
}

int main()
{
    /* Watchdog Disabling function */
    MSS_WD_disable();

    /* Setup MYTIMER */
```

```
No consoles to display at this time.
```

ARM ISA

Major elements of an Instruction Set Architecture

(registers, memory, word size, endianess, conditions, instructions, addressing modes)



32-bits

R0
R1
R2
R3
R4
R5
R6
R7
R8
R9
R10
R11
R12
R13 (SP)
R14 (LR)
R15 (PC)
xPSR

Endianess

```
mov r0, #4

ldr r1, [r0,#8]

     r1=mem((r0)+8)

bne loop

subs r2, #1
```

32-bits

Region	Size	Start address
System		0xE0100000 – 0xFFFFFFFF
Private peripheral bus - External		0xE0040000
Private peripheral bus - Internal		0xE0000000
External device	1.0GB	0xA0000000
External RAM	1.0GB	0x60000000
Peripheral	0.5GB	0x40000000
SRAM	0.5GB	0x20000000
Code	0.5GB	0x00000000

Endianess

31	30	29	28	27	26 … 0
N	Z	C	V	Q	RESERVED

Assembly example

```
data:
    .byte 0x12, 20, 0x20, -1
func:
        mov r0, #0
        mov r4, #0
        movw   r1, #:lower16:data
        movt   r1, #:upper16:data
top:    ldrb   r2, [r1],#1
        add r4, r4, r2
        add r0, r0, #1
        cmp r0, #4
        bne top
```

Questions?

Comments?

Discussion?