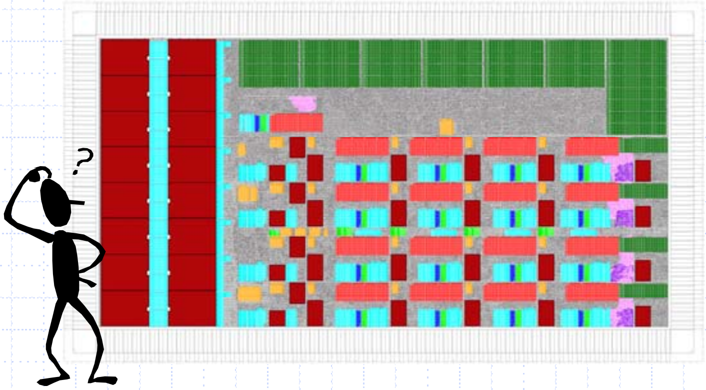
6.375: Complex Digital Systems



Lecturer:	Arvind
TA:	Richard S. Uhler
Administration:	Sally Lee

Why take 6.375

- Something new and exciting as well as useful
- Fun: Design systems that you never thought you could design in a course
  - made possible by large FPGAs and Bluespec

You will also discover that is possible to design complex digital systems with little knowledge of circuits

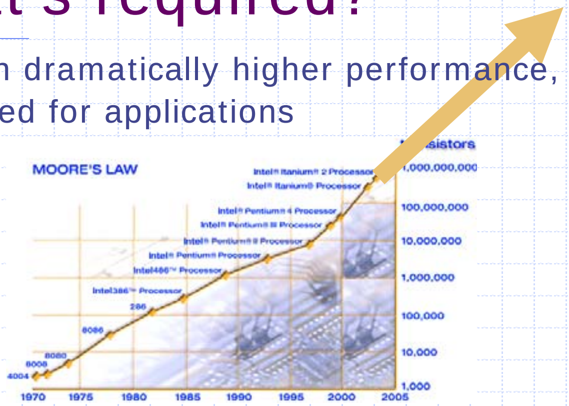
New, exciting and useful …

Wide Variety of Products Rely on ASICs

ASIC = Application-Specific Integrated Circuit

What's required?

ICs with dramatically higher performance, optimized for applications



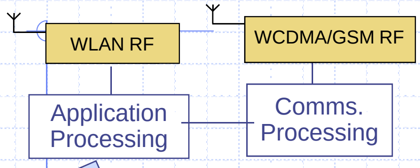
Source: http://www.intel.com/technology/silicon/mooreslaw/index.htm

and at a

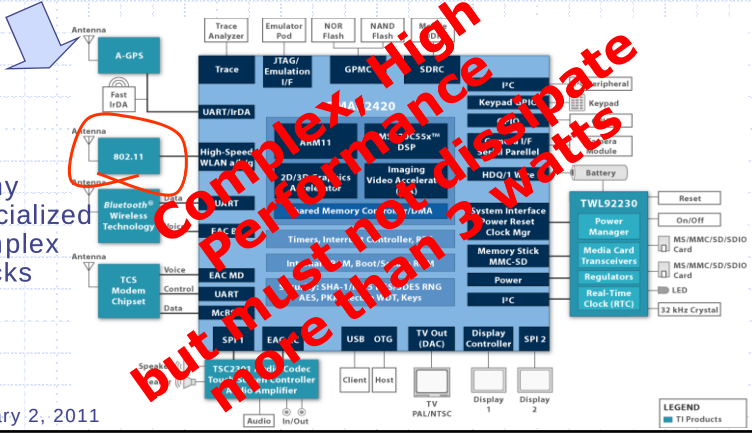
◆ size and power to deliver mobility

◆ cost to address mass consumer markets

Current Cellphone Architecture



Two chips, each with an ARM general-purpose processor (GPP) and a DSP (TI OMAP 2420)



Complex, High Performance

but must not dissipate more than 3 watts

Many specialized complex blocks

Server microprocessors also need specialized blocks

- compression/decompression
- encryption/decryption
- intrusion detection and other security related solutions
- Dealing with spam
- Self diagnosing errors and masking them
- …

Real power saving implies specialized hardware

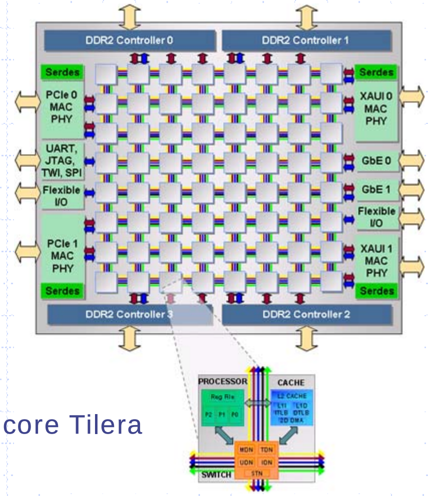
- H.264 video decoder implementations in software vs. hardware
  - the power/energy savings could be 100 to 1000 fold

but our mind set is that hardware design is:

- Difficult, risky
  - Increases time-to-ma
- Inflexible, brittle, er
  - Difficult to deal with changing standards, …

New design flows and tools can change this mind set

Will multicores reduce the need for new hardware?

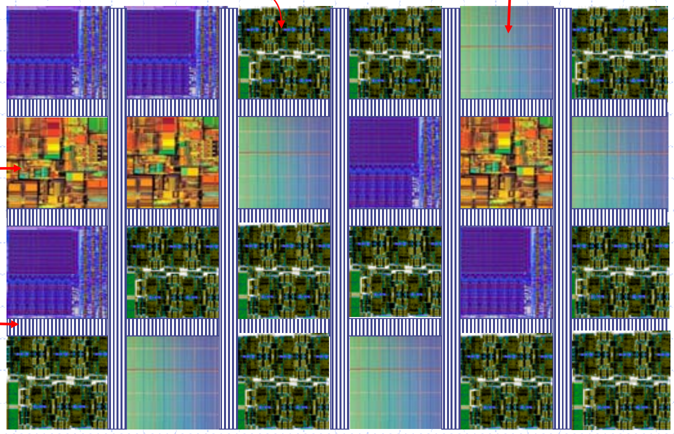


64-core Tilera

Unlikely –
because of
power and
performance

SoC & Multicore Convergence:

more application specific blocks



Application-specific processing units

On-chip memory banks

General-purpose processors

Structured on-chip networks

To reduce the design cost of SoCs we need …

- Extreme IP reuse
  - Multiple instantiations of a block for different performance and application requirements
  - Packaging of IP so that the blocks can be assembled easily to build a large system (black box model)
- Architectural exploration to understand cost, power and performance tradeoffs
- Full system simulations for validation and verification

"Intellectual Property"

Hardware design today is like programming was in the fifties, i.e., before the invention of high-level languages

Programmers had to know many detail of their computer

IBM 650
(1954)

IBM 650 Instruction: 60 1234 1009

Can you program a computer without knowing , for example, how many registers it has?

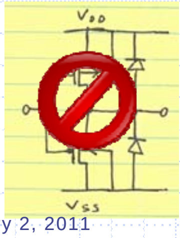
distribution, put it also into the *upper accumulator*, set *lower accumulator* to zero; and then go to location 1009 for the next instruction.”

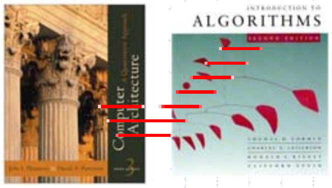
1950s
reaction

Fortran changed this mind set (1956)

For designing complex SoCs deep circuits knowledge is secondary

Using modern high-level hardware synthesis tools like Bluespec requires computer science training in programming and architecture rather than circuit design



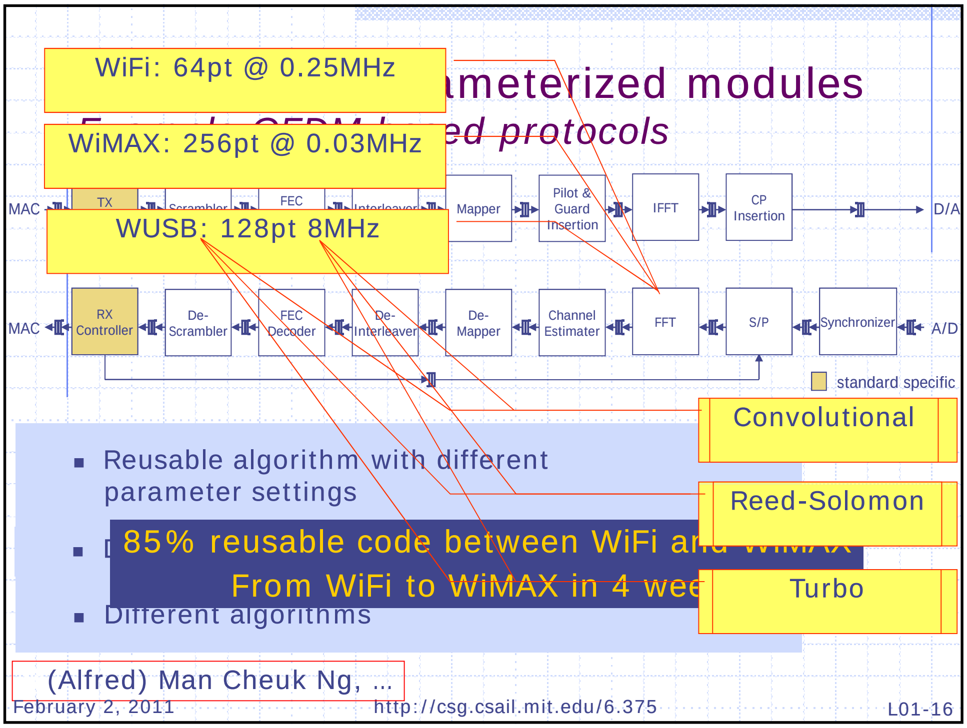


Bluespec A new way of expressing behavior

- A formal method of composing modules with parallel interfaces (ports)
  - Compiler manages muxing of ports and associated control
- Powerful and *zero-cost* parameterization of modules
  - Encapsulation of C and Verilog codes using Bluespec wrappers
  - Helps Transaction Level modeling

➔ *Smaller, simpler, clearer, more correct code*

➔ *not just simulation, synthesis as well*



ameterized modules

ed protocols

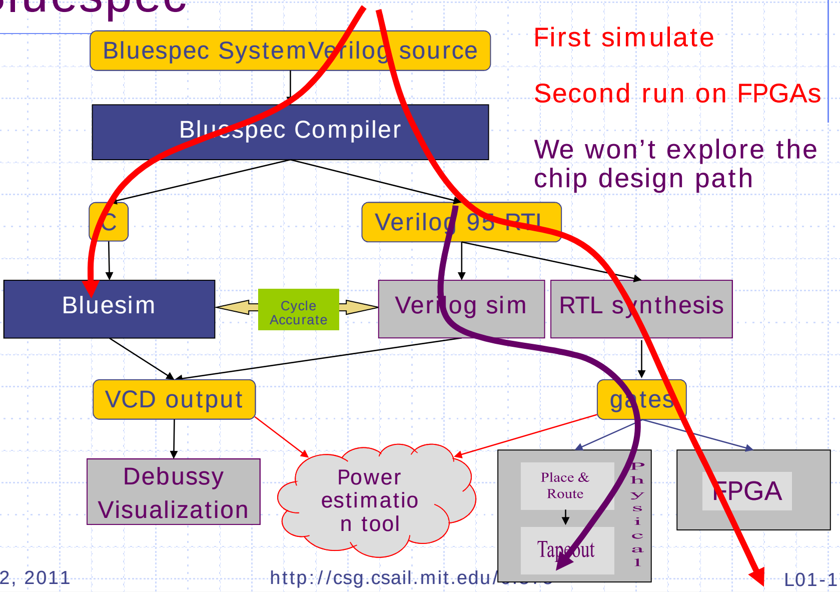
- Reusable algorithm with different parameter settings
- D
- Different algorithms

85% reusable code between WiFi an

From WiFi to WiMAX in 4 wee

(Alfred) Man Cheuk Ng, ...

High-level Synthesis from Bluespec



First simulate

Second run on FPGAs

We won't explore the chip design path

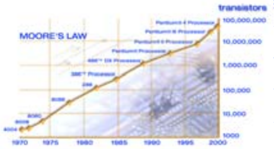
FPGAs: a new opportunity

Chip Design Styles

- Custom and Semi-Custom
  - Hand-drawn transistors (+ some standard cells)
  - High volume, best possible performance: used for most advanced microprocessors
- Standard-Cell-Based ASICs
  - High volume, moderate performance: Graphics chips, network chips, cell-phone chips
- Field-Programmable Gate Arrays
  - Prototyping
  - Low volume, low-moderate performance applications

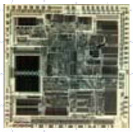
Different design styles have vastly different costs

Exponential growth: Moore's Law



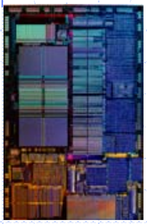
Intel 8080A, 1974
3Mhz, 6K transistors, 6u

Intel 8086, 1978, 33mm^2
10Mhz, 29K transistors, 3u

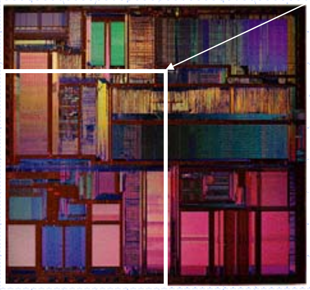


Intel 80286, 1982, 47mm^2
12.5Mhz, 134K transistors, 1.5u

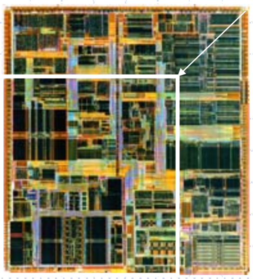
Intel 386DX, 1985, 43mm^2
33Mhz, 275K transistors, 1u



Intel 486, 1989, 81mm^2
50Mhz, 1.2M transistors, .8u



Intel Pentium, 1993/1994/1996, 295/147/90mm^2
66Mhz, 3.1M transistors, .8u/.6u/.35u



Intel Pentium II, 1997, 203mm^2/104mm^2
300/333Mhz, 7.5M transistors, .35u/.25u

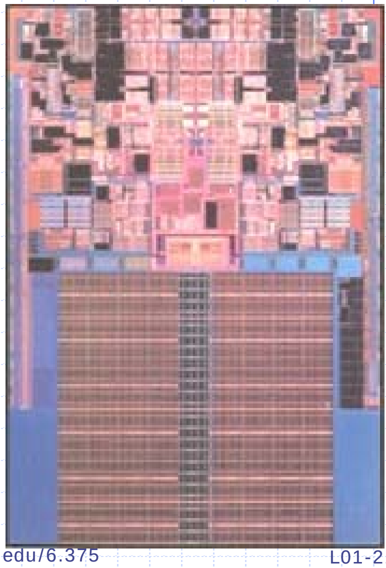
Shown with approximate relative sizes

http://www.intel.com/intel/intelis/museum/exhibit/hist_micro/hof/hof_main.htm

Intel Penryn (2007)

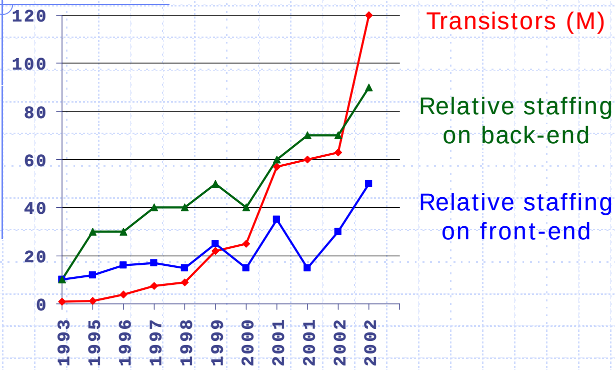
- Dual core
- Quad-issue out-of-order superscalar processors
- 6MB shared L2 cache
- 45nm technology
  - Metal gate transistors
  - High-K gate dielectric
- 410 Million transistors
- 3+? GHz clock frequency

Could fit over 500 486 processors on same size die.



But Design Effort is Growing

Nvidia Graphics Processing Units



9x growth in back-end staff

5x growth in front-end staff

- Front-end is designing the logic (RTL)
- Back-end is fitting all the gates and wires on the chip; meeting timing specifications; wiring up power, ground, and clock

Design Cost Impacts Chip Cost

An Altera study

- Non-Recurring Engineering (NRE) costs for a 90nm ASIC is ~ $30M
  - 59% chip design (architecture, logic & I/O design, product & test engineering)
  - 30% software and applications development
  - 11% prototyping (masks, wafers, boards)
- If we sell 100,000 units, NRE costs add

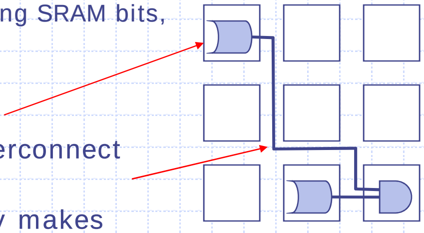
$30M/100K = $300 per chip!

Hand-crafted IBM-Sony-Toshiba Cell microprocessor achieves 4GHz in 90nm, but at the development cost of >$400M

Alternative: Use FPGAs

Field-Programmable Gate Arrays (FPGAs)

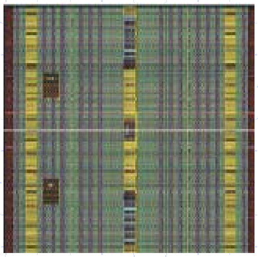
- Arrays mass-produced but programmed by customer after fabrication
  - Can be programmed by loading SRAM bits, or loading FLASH memory
- Each cell in array contains a programmable logic function
- Array has programmable interconnect between logic functions
- Overhead of programmability makes arrays expensive and slow as compared to ASICs
- However, much cheaper than an ASIC for small volumes because NRE costs do not include chip development costs (only include programming)



FPGA Pros and Cons

Advantages

- Dramatically reduce the cost of errors
- Little physical design work
- Remove the reticle costs from each design



Disadvantages (as compared to an ASIC)

[Kuon & Rose, FPGA2006]

- Switching power around ~12X worse
- Performance up 3-4X worse
- Area 20-40X greater

Still requires tremendous design effort at RTL level

The new opportunity

- “Big” FPGAs have become widely available
  - A multicore can be emulated on one FPGA
  - but the programming model is RTL and not too many people design hardware
- Enable the use of FPGAs via Bluespec

Fun: Design systems that you never thought you would design in a course

Some Bluespec/FPGA projects at MIT

- Video decoder – H.264
- AirBlue – A new platform to experiment with cross-layer wireless protocols
- Cycle-accurate performance models
    - Intel's Hasim
    - IBM's PowerPC
- Hardware software co-generation

H.264 Video Decoder

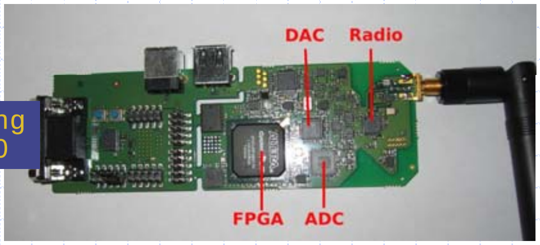
Chun-Chieh Lin, K Elliott Fleming [MEMOCODE 2008]

- Used everywhere - cell phones, DVDs, HD-DVDs
- Initial Design
  - Eight man-months
  - 8K lines of Bluespec
    - in contrast to 80K lines of C standard
  - Decoded 720p@32FPS
- Major architectural explorations over 3 months
  - High performance designs (4.2 mm sq in 180nm)
    - 720p@75FPS, 1080p@65FPS,
  - Low cost designs
    - QCIF@15FPS (2.2mm sq), 720p@30FPS (2.4mm sq)

Can be refined further to run 1080p@75FPS on FPGAs

AirBlue: A platform for Cross-Layer Wireless Protocol development

Now building AirBlue2.0



Fits in Nokia N95 phones

- *Cross-layer protocols* (i.e., jointly optimizing PHY and MAC layers) are the hottest area of research in wireless
- Several cross-layer experiments (e.g., SoftPhy) have already been conducted on full-speed 802.11a/g implementation

With Prof Hari Balakrishanan

Each new protocol required less than 100 lines of code

IBM: PowerPC Prototype

K. Ekanadham, Jessica Tseng (IBM)
Asif Khan, M. Vijayaraghavan (MIT)

- Goal: Implement a multithreaded, multicore, in-order PowerPC on an FPGA platform and boot Linux on it in 12 months
- Team:
  - 2(IBM) + 2(MIT) + Linux and FPGA help

The team accomplished the goal (Nov 2008)
- Bluespec PowerPC boots Linux on FPGAs in 10min;
- 100M instructions to reach “Hello World”;
- 15K lines of Bluespec generated 90K lines of Verilog

IBM synthesized the generated Verilog using their tools in 40nm library
– ran at 500MHz on the first try!

Phase II: IBM/MIT Collaboration

March 2009 –

- Goal: Produce a *cycle-accurate* and *highly parameterized model* of multithreaded, multicore PowerPC to run on FPGAs
  - demonstrate 1000X speedup and flexibility by running the models on FPGAs
- Use cheaper and widely available FPGA boards
  - Xilinx 110 as opposed to 330
- Target open source distribution

The model is currently able to boot 32-bit Linux on FPGAs and runs at 4.4 MIPS

The Course Philosophy

- Effective abstractions to reduce design effort
  - High-level design language rather than logic gates
  - Control specified with Guarded Atomic Actions rather than with finite state machines
  - Guarded module interfaces automatically ensure correctness of composition of existing modules
- Design discipline to avoid bad design points
  - Decoupled units rather than tightly coupled state machines
- Design space exploration to find good designs
  - Architecture choice has largest impact on solution quality

We learn by doing actual designs

The course has no text book but …

- Lecture slides (with animation)
  - Make sure you sure you understand the lectures before exploring other materials
  - http://csg.csail.mit.edu/6.375/handouts.html
- Small Example suite (from Bluespec Inc)
  - A series of small examples (currently over 70), focusing on one topic at a time. Good entry for learning the language by yourself
  - http://sites.google.com/a/bluespec.com/learning-bluespec/Home/Small-Examples
  - bluespec.com → Resources → Wiki → Small Examples
- Bluespec System Verilog Reference manual
  - It is a reference, not a tutorial
  - http://www.bluespec.com/forum/download.php?id=96
  - bluespec.com → Resources → Wiki → BSV Documentation → Reference Manual
- Bluespec System Verilog Users guide
  - How to use all the tools for developing BSV programs
  - http://www.bluespec.com/forum/download.php?id=107
  - bluespec.com → Resources → Wiki → BSV Documentation → User Guide