6.375 Final Project

Guidelines

- Ideally – groups of two to three
- Goal: A complex digital design – functional on FPGAs, useful, substantial
- Ideas related to your research domain welcome
- Past projects online –
  - http://csg.csail.mit.edu/6.375/6_375_2008_www/projects.html
  - http://csg.csail.mit.edu/6.375/6_375_2007_www/projects.html
  - http://csg.csail.mit.edu/6.375/6_375_2006_www/projects.html
  - http://csg.csail.mit.edu/6.884/projects.html

Project Schedule

- Weekly project meetings and deadlines
- Intermediate reports due according to following schedule:

Date	Task	Report
Mar 16		Preliminary proposal
April 01	High-level Design & Arch	Final proposal with diagram
Apr 08	Detailed Microarchitecture	Testing plan
Apr 22	Routable Implementation	Synthesis report (SR)
Apr 29	Simulation Demonstration	Perf expectations, tuning & SR
May 06	FPGA Demonstration	Design exploration plan & SR
May 13	Project Presentation	
May 14		Project Report

Project ideas

- Floating point Unit & other processor accelerators
  - Asif Khan, Murali Vijayaraghavan
- SHA-3: Hash algorithm evaluation
  - Elliott Fleming
- Channel Estimation
  - Alfred Man Cheuk Ng
- HAsim: Cycle-accurate performance modeling
  - Michael Pellauer
- DRAM controller
  - Jae Lee

Status of the PowerPC Project

- Single-threaded PowerPC pipeline optimized for FPGA implementation
- It can execute sample code at 125MHz
- It has a FIFO memory interface that connects it to the DDR2 memory controller
- It also has a PLB slave interface for gathering statistics

Floating Point Unit

- PowerPC FPU coded in Bluespec for FPGA implementation
- Simple FIFO input, output interface – <operation type, source operand values> provided at input, <destination operand values> expected at output after a specified number of cycles
- Operations are outlined in the PowerPC ISA Book 1
- Primary goals are minimizing FPGA resource utilization and maximizing clock frequency
- Parameterization is also desired

Reference Material

- **PowerPC User Instruction Set Architecture Book I Version 2.01**
  provides all the details of the floating point processor, available upon request
- **http://grouper.ieee.org/groups/754/**
  links to related papers, publically available software implementations and test rigs
- **http://www.opencores.com/projects.cgi/web/fpu100/overview**
  open-source VHDL 32-bit FPU implementation
- **Floating Point Trailblaze**
  Verilog 64-bit FPU implementation from UT Austin's TRIPS project, available upon request

Crypto-engine

Symmetric-key cryptography

- Block ciphers
- Encryption:

  f(key, msg) -> c
- Decryption:

  f^{-1}(key, c) -> msg
- Standard used currently is AES(128, 192, 256 bits)
- Reference: http://www.csrc.nist.gov/publications/fips/fips197/fips-197.pdf

Public-key cryptography

- Encryption:
  f(public key, msg) -> c
- Decryption:
  g(private key, c) -> msg
- Example: RSA (1024 bits or more)
- Reference
  http://people.csail.mit.edu/rivest/Rsapaper.pdf

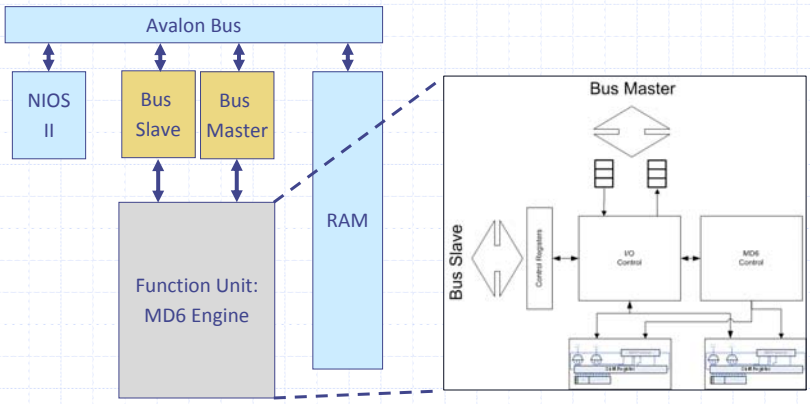
Crypto-engine Interfaces

- AES:
  method Action setKey(key)
  method c encrypt(msg, size)
  method msg decrypt(c, size)
- RSA:
  method Action setPrivateKey(private key)
  method Action setPublicKey(public key)
  method c encrypt(msg, size)
  method msg decrypt(c, size)
- Or similar interfaces if you want to implement different crypto algorithms (including hashing)

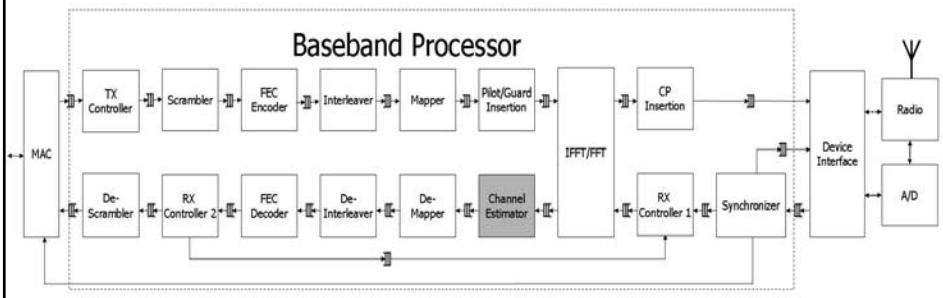
SHA-3 Projects

- SHA-3 – new standard hash functions
  - Influence the international standard process
- >40 propsals, few HW implementations
- Many based on existing crypto/hash functions – pick a few
- Some more exotic – pick one
  - FFT
  - Matrix multiplication

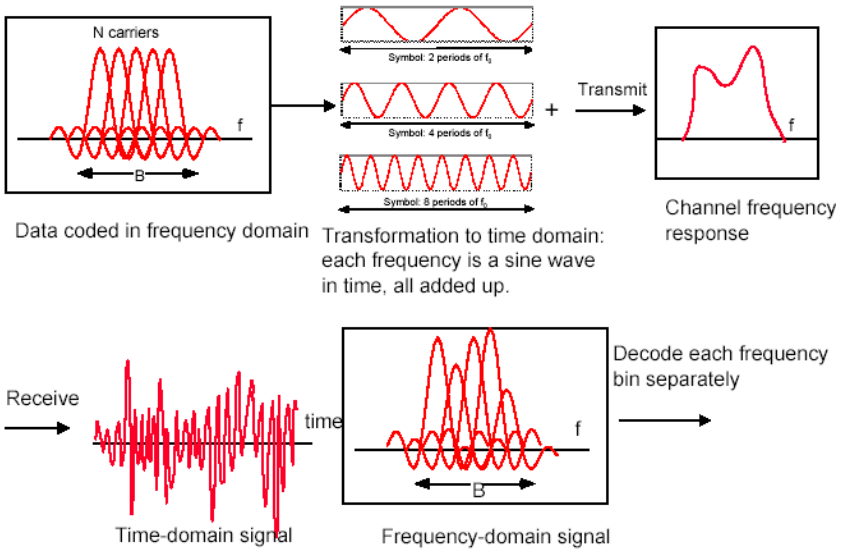
General Architecture



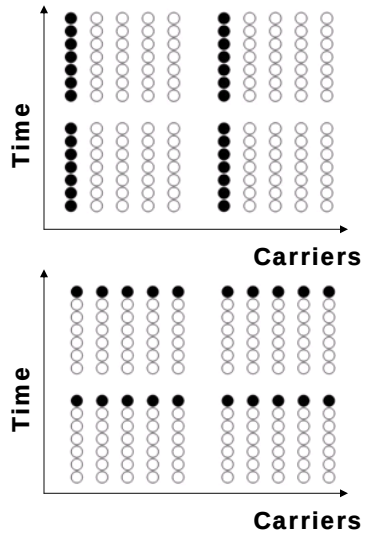
6.375 Project Idea: Channel Estimator



Impact of Channel Frequency Response



Pilot for Channel Estimation



- Comb Type:
  - Part of the sub-carriers are always reserved as pilot for each symbol
- Block Type:
  - All sub-carriers is used as pilot in a specific period
- 802.11a/g use both

Possible Parameterizations

- No. subcarriers
- Values and locations of pilots
- Locations of guard bands (unused subcarriers)

All the above parameterizations are preferred to be dynamic if possible!

References

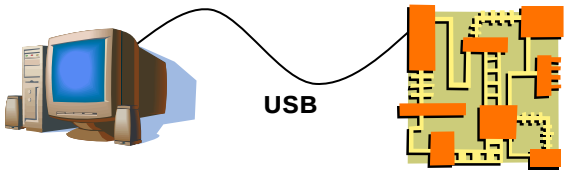
[1] *www.freescale.com/files/dsp/doc/app_note/AN3059.pdf*
[2] *www.winlab.rutgers.edu/~spasojev/courses/projects/CE_OFDM.ppt*

Rapid Processor Simulation Using HAsim

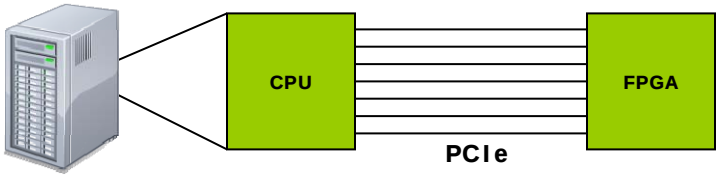
Michael Pellauer
pellauer@csail.mit.edu

FPGAs: Stand-Alone vs Accelerator

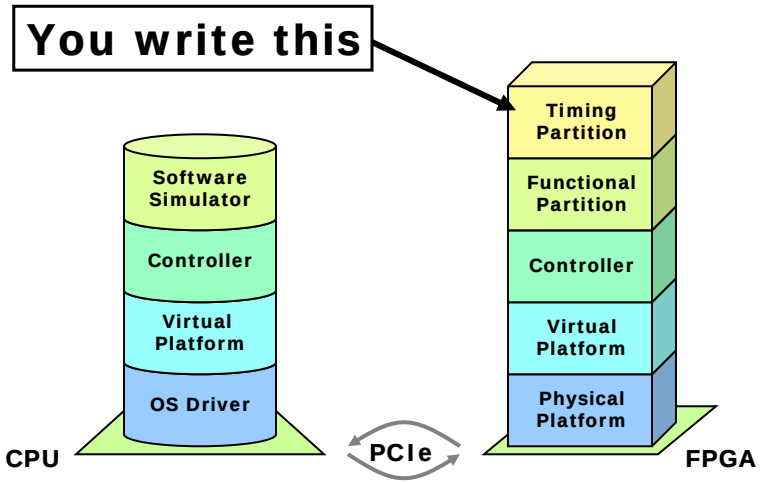
- Stand-alone FPGA (like 6.375):



- Accelerator FPGA (like GPU):



The HAsim Simulator



Project Ideas

- **Existing Timing Models**
  - Unpipelined (Multicore)
  - 5-stage inorder pipeline (Multicore)
  - Out-of-order superscalar
- **Proposed idea: SMT Processors**
  - Symmetric Multi-Threaded
  - Each core would have to track model state for multiple threads
  - Perform architectural exploration using SPEC benchmarks


Any Questions?
pellauer@csail.mit.edu
Naragenſet
RHODE ISLAND
Cape Cod

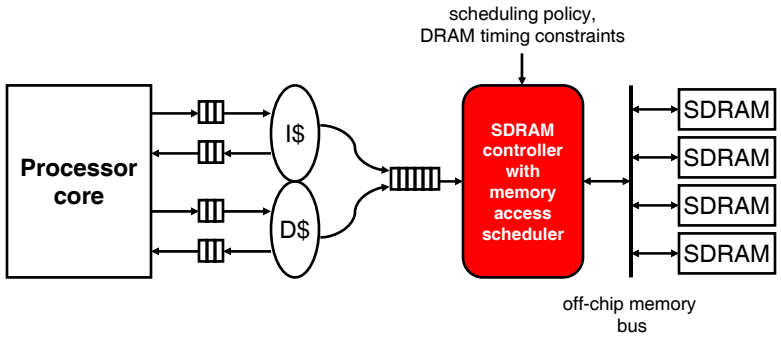
Memory Access Scheduling

Jae Lee

6.375 Complex Digital Systems

March 2, 2009

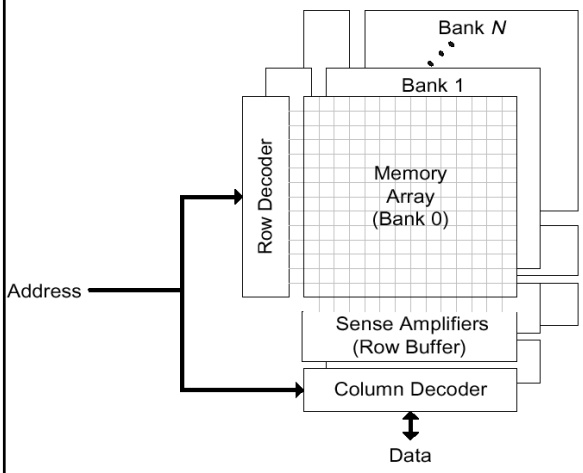
Memory Access Scheduling



- Memory access scheduling: reorder memory accesses to maximize DRAM throughput and reduce latency by exploiting multiple banks and row buffers.

** S. Rixner, et. al, Memory Access Scheduling, In ISCA '00.*

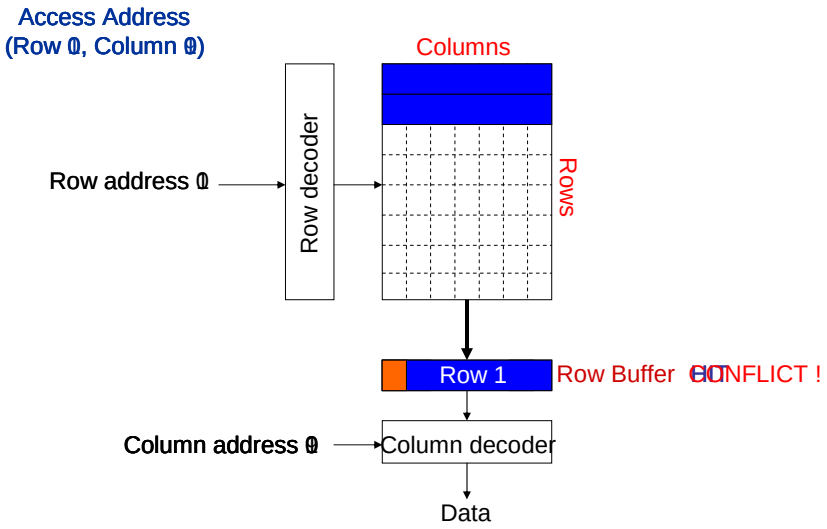
3D Structure of DRAM Chip



- Memory data location: (bank,row,column)
- Three steps in accessing the memory data:
  - bank precharge
  - row access (row activation)
  - column access

** The figure taken from: S. Rixner, et. al, Memory Access Scheduling, in ISCA '00.*

DRAM Bank Operation*



A row-conflict memory access takes significantly longer than a row-hit access.

** This slide was taken from a presentation of Prof. Onur Mutlu at CMU [MICRO '07].*

What Should Be Done?

- Implement SDRAM controller with memory accessing scheduler
  - May be able to reuse a baseline SDRAM controller from the FPGA vendor
- Commonly used scheduling policy (FR-FCFS)*
  (1) Row-hit (column) first: Service row-hit memory accesses first
  (2) Oldest-first: Then service older accesses first
- Evaluation strategy: an idea
  - Functional verification: FPGA
  - Performance evaluation: cycle-accurate model

** S. Rixner, et. al, Memory Access Scheduling, In ISCA '00.*