L0: INTRODUCTION

18-545: ADVANCED DIGITAL DESIGN PROJECT
FALL 2011
BILL NACE
18-545: Advanced Digital Design Project

- Digital system capstone design project
  - Spend the entire semester working on a single project
  - Work in teams of 3 people

- Course is designed to teach you
  - How to design, implement and debug a real working system
  - How to plan, manage and execute a sizable project
  - How to work effectively in a team
Project Description

☐ Design a video game
  ☐ Must output to a video display
  ☐ Must have sound effects
  ☐ Must take user input (keyboard, mouse, wiimote, etc)
  ☐ Must support multiple simultaneous players
  ☐ Must have scoring mechanisms or victory conditions
  ☐ Must not have been written by you
    ☐ ... this is not a game design class
Project Forms

- Stand-alone video game (Battlezone, Quake, etc)
  - Most common path
- Classic console / arcade game platform (Super NES)
  - Games come from cartridges or ROMs
- Hardware accelerator for an existing game or graphics library
  - (OpenGL accelerator for Quake)
- Other: Research possibilities
Design Platform: Choice #1

- Virtex 5-LX Evaluation Board
  - Xilinx XC5VLX110T (110K Logic Cells)
    - 6+MB RAM, 680 pins
  - The “OpenSPARC” Board
    - UltraSPARC T1 (soft core)
  - MicroBlaze can be instantiated
  - Or, plenty of space for generic hardware
Design Platform: Choice #2

- Virtex 5-FX Board
  - Xilinx XC5VFX70T Device (70K Logic Cells)
    - 6+MB RAM, 640 pins
    - 2 PowerPC 440 (hard cores)
  - FX device has “FSL”
    - Fast Simplex Link is low latency path to your hardware device
Your Task
Today

- Lecture
  - Course introduction
  - Logistics
  - Project description
- Handouts
  - Syllabus
- Team Assignments
Course Logistics

☐ Class times
  ☐ Lectures: MW (mostly just Monday) 10:30 - 12:20, DH 1217
  ☐ Labs: 24-hour access, HH 1307
    ☐ Mandatory Labs on most Wednesdays
      ☐ **Mandatory** means mandatory -- be physically present

☐ Blackboard will only be used for “Assignment” turn-in

☐ [www.ece.cmu.edu/~ece545](http://www.ece.cmu.edu/~ece545) for information repository
  ☐ Syllabus, schedule, FPGA resources, Past Project reports
Teaching Staff

Bill Nace
- wnace@cmu.edu
- Hamershlag Hall D-207
- Office Hours: Wed 2:00 - 4:00

Shannon Lown
Course Management Assistant
- shannonh@ece.cmu.edu
- Hamershlag Hall D-level (Course Hub)
- M-F 9:00 - 4:00
Teaching Assistants

☐ Lincoln Roop
  ☐ lroop@andrew.cmu.edu
  ☐ Office hours: TBA
Pre-Requisites

☐ Official

☐ What I really care about

☐ Have a good grasp of computer architecture

☐ Know how to code synthesizable Verilog

☐ Know how to code C/C++

☐ Know your way around a *NIX OS

☐ Are a good teammate and a decent human being
Pre-Requisites (2)

☐ What would also be good
  ☐ Know something about computer graphics
  ☐ Know something about FPGA design

☐ You are supposed to walk into a capstone with all the technical skills necessary to do the project
  ☐ And the ability to find / decode / learn any details necessary
Lab Assignments

- **Purpose**: familiarize yourself with the Xilinx boards / tools
  - **Alternate purpose**: Forcing function for early start, learn historically neglected skills

- **Very open-ended**
  - **Goal**: you spend some time just “messing-around” in lab

- **Front-loaded**
  - **3 labs will be completed in first several weeks**

- **Work in project teams, one report per team**
Lab Room

- Hamershlag Hall 1307
- Key swipe access 24-hours
- Each team will be assigned a lab bench
  - Xilinx Development board
  - Linux computer with Xilinx software
  - 2nd monitor, mouse, keyboard for use with the FPGA board
- Other lab equipment
  - Logic analyzers, multi-meters, tools, wire, solder, etc
  - CF card reader
Course Texts

- *The Pentium Chronicles* by Robert Colwell
  - Learn about the 4 phases of product development

- *Debugging* by David J. Agans
  - Make you think about your debugging process

- I have lending library as well
  - Graphics, game/GPU programming, etc
Project Logistics

☐ Project Milestones
  ☐ Project idea statement
  ☐ Proposal
  ☐ Design Review
  ☐ Final Presentation / in-lab demo
  ☐ Public Demo!
Project Startup

- Start thinking about what you want to build **NOW**
  - Brainstorm with your team
  - Research what has been done by others
    - Project reports are on website
  - Iterate with instructor / TAs to make the project feasible
    - World of Warcraft ➔ too hard
    - Pong ➔ too easy
    - Networked VR Pong ➔ now we’re talking ...
Project Teams

- You are responsible for organizing yourselves into teams of 3
  - I reserve the right to meddle in that process

- Choose your teammates wisely
  - People with a diverse set of skills
  - People who have the same goals / standards / expectations
  - People you can work with
  - People who will actually stay in the class

- Form teams TODAY
Weekly Progress Reports

☐ Each week, each person must submit a progress report
  ☐ Use Blackboard
  ☐ Due every Monday at 9:30 am

☐ Content
  ☐ What you’ve accomplished in the past week
  ☐ What you plan to accomplish next week
  ☐ Are you on schedule?
Grading

- **Mid-Semester:**
  - Participation (status reports) 10%, Labs 40%, Project 50%

- **Final:** Participation 10%, Labs 10%, Project 80%
  - not submitting status reports will hurt your project grade

- **Late Policy**
  - Late work is not accepted
  - i.e. you’d better have a project to demo
How to get an A

☐ Your project:
  ☐ Works great. Perhaps you have a bug or two, but you have good work-arounds in place
  ☐ Is challenging:
    ☐ Uses different input / output devices
    ☐ Integrated with FPGA hardware that you’ve designed

☐ You:
  ☐ Were a good team member: Contributed. Did lots of work
  ☐ Were a good student: Participated in discussions. On-time
How to get a B

☐ Your Project:
  ☐ Mostly works
  ☐ Is a bunch of code you grabbed from the net, ported to PowerPC and didn’t have to do much else
  ☐ Uses keyboard / mouse. Nothing else.

☐ You:
  ☐ Undistinguished member of the team
  ☐ Showed up for class, but didn’t do much else
  ☐ Blew off a “Mandatory Lab”
Academic Honesty

☐ All work you submit for this class must be your own

☐ Given the nature of this class, there may be instances where you will use code/designs from other people

☐ Must be clearly and explicitly noted

☐ Must have proper and complete citation (i.e. based on your citation, we should be able to quickly and easily find it)
Project Overview

☐ Video game

☐ Must output to a video display
☐ Must have sound effects
☐ Must take user input
☐ Must support multiple simultaneous players
☐ Must have scoring mechanisms or victory conditions

☐ You may NOT design a game from scratch
☐ This is not a course in game design
Many different ways to accomplish

Stand-alone game

Classic Console

Hardware accelerator

... and many others I haven’t thought of ...
Let’s Discuss

- The basics
  - Graphics
  - Sound
  - User input
  - Multiple players

- Project types
  - Stand alone games
  - Classic consoles
  - Hardware Accelerators
  - Helpful links
Graphics: Bit Mapped

- Frame buffer
  - Block of memory that represents the screen
  - Each pixel is represented as a number or set of numbers

- Very general mechanism
  - Can be used for 2D or 3D graphics

- Requires a fair chunk of memory
Graphics: Vector

- Directly control electron gun to draw lines on the screen
- Low storage requirements
- Any line is just a pair of vertices
  - Plus attributes like color, dashes
- Really good when the amount of memory needed for a high-resolution frame buffer was unthinkable (multiple boards with multiple memory chips)
- Simple to transform vector graphics to bitmapped
Graphics: 2D and 3D Techniques

- 2D – sprites
  - Dedicated sprite-handling hardware, with limits on size / number

- 3D – rasterization
  - Scan through all objects in viewing frustrum
  - Check to see if object is in front of closest object (Using Z-buffer)
  - If so, alter pixel accordingly

- 3D – ray tracing
  - SaarCor – http://www.saarcor.de/
  - OpenRT – http://www.openrt.de/
Graphics: 3D Stereoscopic

- Send a slightly different image to each eye
  - Impression of true 3D images
- Red/green tinted lenses
- LCD shutter glasses
- VR helmet
- Dual LCD with restricted viewing angles
- True 3D displays
Sound

☐ Background music

☐ Sound effects

☐ Main game component
   ☐ Dance Dance Revolution, Taiko Drum Master
   ☐ Guitar Hero, Rock Band
User Input: Interfaces

- 5 buttons
- 4 DIP switches
- Two PS/2 ports
- FPGA pins brought to connectors
User Input: Peripherals
Multiple Simultaneous Players

- **Style**
  - Co-operative
    - Players vs. environment
  - Adversarial
    - Player vs. player

- **Display**
  - Same screen
  - Split screen
  - Multiple display
    - One board w/ 2 output
    - Multiple networked boards
Stand-alone Games

- Look for arcade game built from around 1980 - 1985
- About the right level of complexity
- XUP board can do more advanced games

Examples:
- Pacman (1980)
- Missile Command (1980)
- Battlezone (1980)
- Xevious (1982)

PS: I SPENT MOST OF MY HIGH SCHOOL QUARTERS ON "DEFENDER." ... JUST SAYIN’
Stand-alone Games (2)

- Enormous number of online resources
  - Service manuals
    - Often include board schematics

- Multiple Arcade Machine Emulator (MAME)
  - Faithful software emulator of many hardware boards
    - Many arcade games used the same or similar hardware
  - Game ROMs are sometimes in a legal gray zone
  - Can migrate from software to hardware
  - http://www.mame.net/
Stand-alone Games (3)

- Open-source software implementations of the game
- Other hardware implementations of the game
  - Classes at other universities
  - FPGA/arcade enthusiasts
Classic Consoles

- Console game systems lagged arcade games by ~5 years
  - Not really true any more
- Look for consoles from around 1985 – 1990
  - Atari 2600 (1977)
  - NES (1985)
  - Sega Genesis (1989)
- Don’t forget about handhelds
  - Gameboy (1989)
  - GBA (2001)
Enormous number of online resources

Some consoles have been extremely well documented
- NES – http://nesdev.parodius.com/

Console emulators
- Multiple Emulator Super System (MESS)
- Essentially MAME for consoles
- Faithful low-level emulation of console hardware
- Can emulate 250+ consoles
- http://www.mess.org/
Classic Consoles (3)

- Open-source software implementations of the console
- Other hardware implementations of the console
  - Classes at other universities
  - FPGA/arcade enthusiasts
Hardware Accelerators

☐ Bulk of the game runs in software on CPU
  ☐ Embedded PowerPC cores running Linux
  ☐ Workstation CPU running Linux or Windows XP

☐ Offload compute intensive task to custom hardware in FPGA
  ☐ Graphics routines
  ☐ Floating point math
    ☐ Embedded PowerPC cores do not have FPUs
  ☐ Game world physics calculations
Hardware Accelerator: Example

- OpenGL graphics processor
  - Run OpenGL application on embedded PowerPC under Linux
  - Use open-source OpenGL Mesa library Direct Rendering Interface
to offload OpenGL operations to hardware
  - Implement graphic processor in FPGA
- At least one known project that tried something similar
  - http://www.cs.unc.edu/~lastra/comp290/
n=Resources.Hardware
“Research” Projects

- Prof Red Wittaker (RI): Lunar X-prize
  - Mission to the moon: launch, land, rove, broadcast
  - Mission-critical algorithms require high-frame rates on rad-hardened computing infrastructure
  - Project 1: Optimize video compression for low-B/W transmission
    - Algorithms exist. Need parallel compressors to encode from multiple cameras. Perhaps modify for 3D laser imagery
  - Project 2: Implement laser and visual servo control routines
    - Feature extraction algs exist (star-tracking, SURF feature detect). Show high-frame rate servoing and obstacle avoidance
Helpful Links: Classes

- Columbia CS 4840

- Stanford CS 248

- UNC COMP 290
  - http://www.cs.unc.edu/~lastra/comp290/

- MIT 6.111
Helpful Links: FPGA Systems

- FPGA Arcade

- FPGA Games

- FPGA CPU
  - [http://www.fpgacpu.org](http://www.fpgacpu.org)

- Booting linux on Xilinx FPGAs
  - [http://splish.ee.byu.edu/projects/LinuxFPGA/configuring.htm](http://splish.ee.byu.edu/projects/LinuxFPGA/configuring.htm)
Helpful Links: OpenGL

- OpenGL
  - http://www.opengl.org/

- Mesa open-source OpenGL library
  - http://www.mesa3d.org/

- Mesa Direct Rendering Interface
  - http://dri.freedesktop.org/wiki/FrontPage

- Graphics pipeline tutorial
  - http://www.extremetech.com/article2/0,1697,9722,00.asp

- Linux games
  - http://www.linuxgames.com/
Reminders

☐ Find teams right now

☐ Reading assignment: next Wednesday

☐ Lab 1: next Wednesday