Parallelism with FPGAs in Computer Systems & Organization Course

Kevin A. Walsh and Nicholas Powers
College of the Holy Cross, Worcester, MA

Context

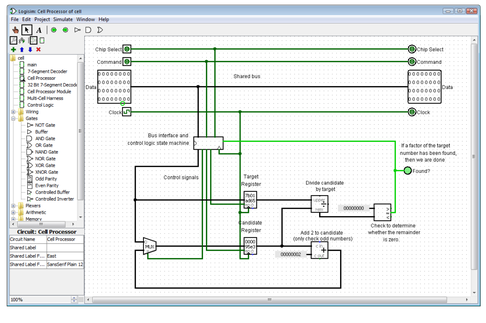
Introducing parallelism at a liberal arts college

- Holy Cross is a small, undergraduate-only college, with 2,891 students and 272 faculty.
- The Mathematics and Computer Science department comprises 13 mathematicians and 3 computer science faculty.
- The department typically graduates about 8 computer science majors and minors per year.

CSCI 226 – Computer Systems and Organization

- NSF/TCPP curriculum concepts were integrated into the Fall 2012 offering of CSCI 226.
- Participants included 10 students and 1 teaching assistant.
- Students typically take the course early in their studies, after a two-semester introductory sequence in programming and data structures.
- Topics include digital logic, data representation, computer organization, memory, datapaths and state machines, microcode, assembly language, assemblers, linkers, and loaders, program organization, caching, and parallel processing.

Software Simulation



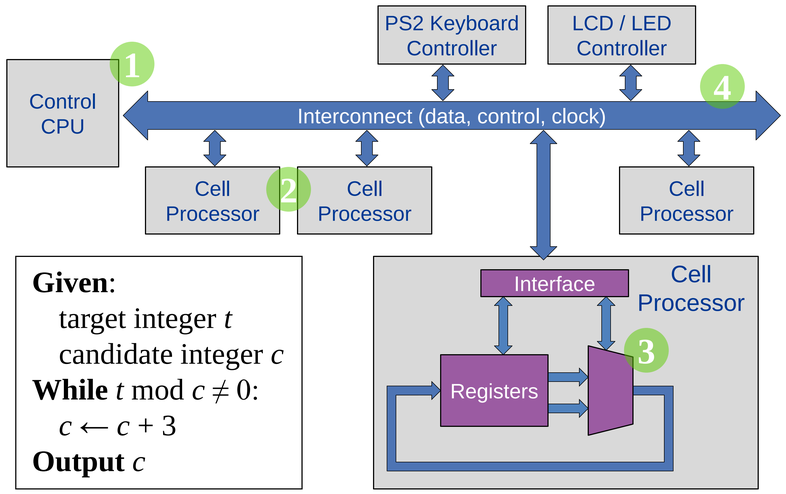
Initial design and debugging using *Logisim*

- Students performed schematic design entry and interactive testing & debugging using *Logisim*.
- *Logisim* is designed expressly for educational use.
- Tested with single- and multi-cell harnesses.

Student Project: Outline & Goals

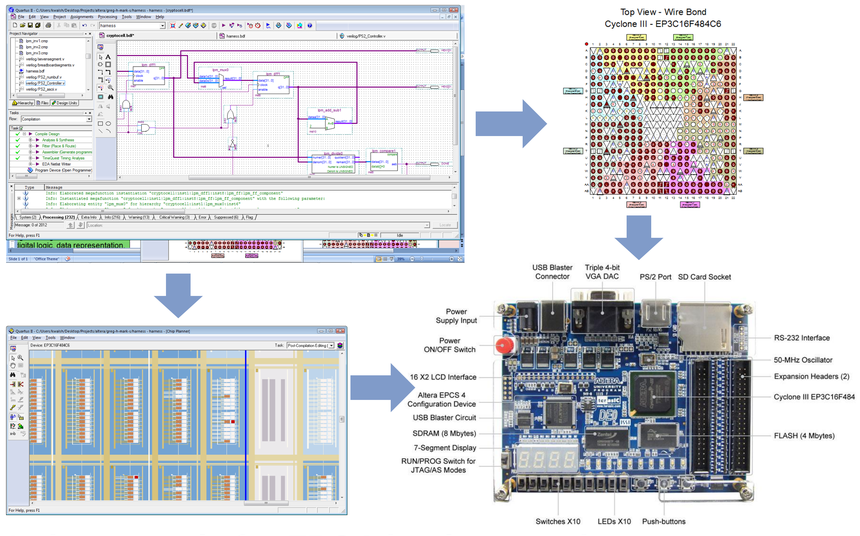
Design a cell processor to factor large integers.

- Design & test in software, then build with FPGAs.
- Connect multiple cells with a shared interconnect.
- Evaluate performance, cost, and efficiency of single-cell and multi-cell designs.



Given:
target integer t
candidate integer c
While $t \bmod c \neq 0$:
$c \leftarrow c + 3$
Output c

FPGA Instantiation



Industry tools for FPGA implementation

- Students manually transferred *Logisim* designs using *Altera Quartus* schematic editor.
- Designs were instantiated on an array of DE0 demo boards with a custom-built I/O harness.

Parallel Concepts

1 Task and Data-level Parallelism

- Prime factorization is trivially parallelizable, and can prompt a discussion of workload partitioning and load balancing strategies.

2 Processor-Level Parallelism

- Students first designed and tested a single-cell design, then investigated the use of multi-cell designs to execute threads concurrently.
- Heterogeneous design uses multiple worker cells and a single control processor.

3 Micro-architectural Parallelism

- Within a single cell, some datapath components perform concurrent operations.

4 Shared Bus Interconnect

- Physical implementation using an array of FPGA prototype boards motivates need for shared bus.
- Raises issues of bus mastering and arbitration, and the use of specialized I/O devices.

Lessons Learned

Provide early, hands-on exposure to parallelism

- Establish a familiar motivating example to be used during the remainder of the course.
- The project objective—prime factorization—is simple enough to be accessible to students in the first weeks of class; implementation requires only four major components: registers, adder, divider/comparator, and a simple state machine.

Hardware can illustrate parallelism

- Schematic design is intuitively concurrent; HDL syntax looks similar to sequential programming. *Risk*: students may associate parallelism with hardware and ignore parallel programming.
- Physical instantiation reinforces concurrency.

Use appropriate & accessible software tools

- Industry design tools are too complex for rapid prototyping and testing by novices.
- Make experimentation quick, interactive, and fun!