DREAMER

PALMER DABBELT, RICHARD LIN, ALBERT MAGYAR



Fast reconfigurable logic for emulation

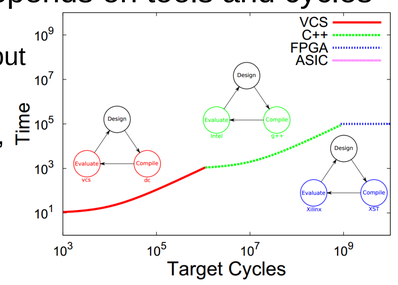
Background

The Design-Test-Execute Loop

- Design – test – execute: current iterative method for hardware design

Iron Law of Emulation

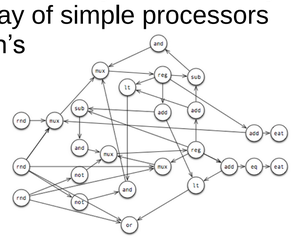
- Time to simulate a design depends on tools and cycles
  - $T = t_{design} + t_{compile} + N t_{cycle}$
  - Software: fast compile times, but slow emulation for complex designs or long simulations
  - FPGA: high compile overhead, suitable for complex designs
  - ASIC: if you have lots of time and money to burn…



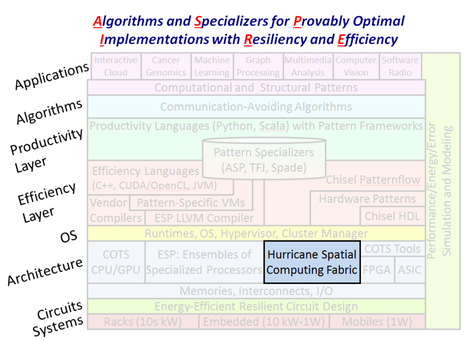
- Practically, FPGA emulation is the fastest tool for emulation … but still isn't ideal:
  - Inefficient support for word-wide operations (natively bit-wide)
  - Logic synthesis and mapping takes a long time
  - … and must be done after *any* change in the design
  - … even if all you want to do is add a logic probe

Overview

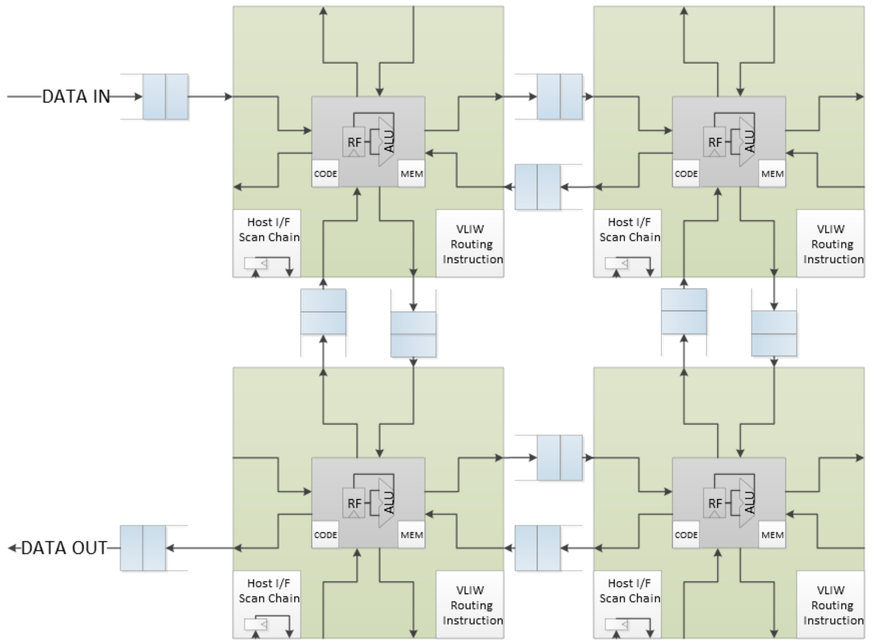
- **DREAMER**: a new reconfigurable logic platform
  - New point in the space between programmability and efficiency
  - Instead of array of bits, use parallel array of simple processors
    - Each tile executes a portion of design's **hardware graph** (right)
  - Supporting tools to map RTL to array
    - Compiles in seconds, not hours
  - Integrated debugging support
    - Probe state without recompiling
  - Overall, **faster design iteration time**



In the big picture:



Architecture



The Array

- Massively parallel array of simple compute tiles
  - Cores connected in a nearest-neighbor network
  - Each connection is a 2-element buffer (reduce deadlocks)
  - Ready-valid interlocked ports (avoid explicit no-ops)
  - Additionally, all tiles' host interfaces connected in a scan chain (for debugging and control)
  - Edge ports can be mapped to external data
- Statically scheduled computation and communication
  - Simple hardware architecture, push complexity to tools
  - All inter-tile data movement explicit (one tile at a time)

The Tile

- Simple compute cores
  - A mini computer – datapath, registers, memory, and code
  - Reduced ISA specialized for simulation, containing:
    - your standard arithmetic / bitwise-logical operations
    - emulation-specific instructions like mux
  - No branching – simple emulation loop
- Dedicated host interface
  - Process host command packets in-place, then shift response to next tile in chain
  - Arbitrary peek/poke access into all state elements
  - Control tile execution

Dataflow Machine

- Generalizing from emulation, array can be used as a general dataflow style machine
  - Can even be programmed in Chisel
  - … or parallel assembly, if you're feeling adventurous
  - Edge ports can be connected to any ready-valid interface:
    - GPIO, DRAM, or coprocessor data movement
- Possible applications
  - General-purpose reconfigurable logic coprocessor
  - DSP filtering, or really anything dataflow-parallel

Hardware Implementation

FPGA

(Zedboard with Xilinx Zynq)

See our pretty demos!

- 16 tiles in a 4 x 4 array
- Clock: 40 MHz

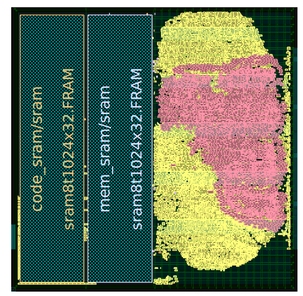
ASIC

(TSMC 45nm process)

- 0.9 ns clock (1.1 GHz)
- 63,442 µm² per tile (1,576 tiles in 100mm², 4,728 tiles in 300mm²)

Specs:

1024 x 32 data memory
1024 x 32 code memory
32 x 32 register file



Future Work

- FPGA area optimization
- Pipelining for faster clock speeds and realistic SRAM data memory
- Automatic instruction set generation and specialization
- DSP optimization
- Efficiently handling fine-grained (bit-width) operations