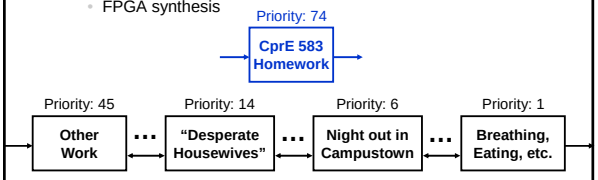
CprE / ComS 583 Reconfigurable Computing

Prof. Joseph Zambreno
Department of Electrical and Computer Engineering
Iowa State University

Lecture #11 – Logic Emulation Technology

Quick Points

- Project proposals due Sunday, September 30 (submit via WebCT)
- HW #3 out today
  - Due Tuesday, October 9
    - Systolic computing structures
    - Systolic mapping
    - Logic partitioning
    - FPGA synthesis



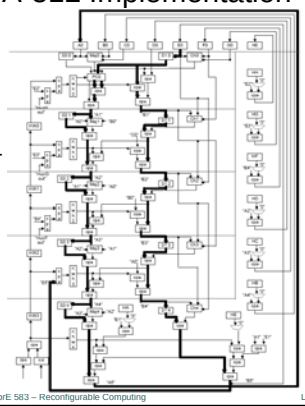
Recap – Introduction to Cryptography

- *Encryption* is the process of encoding a message such that its meaning is not obvious
- *Decryption* is the reverse process, i.e., transforming an encrypted message to its original form

- We denote plaintext by P and ciphertext by C
- C = E(P), P = D(C) and P = D(E(P)), where E() is the encryption function (algorithm) and D() the decryption function

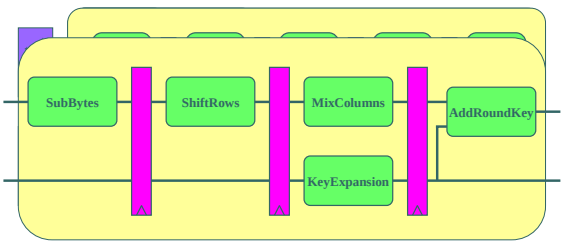
Recap – SHA-512 Implementation

- Partial unrolling (5 rounds), pipelining
- 1 Gbps on Virtex-E FPGAs
- See [LieGre04A] for details



Recap – AES-128E Optimization

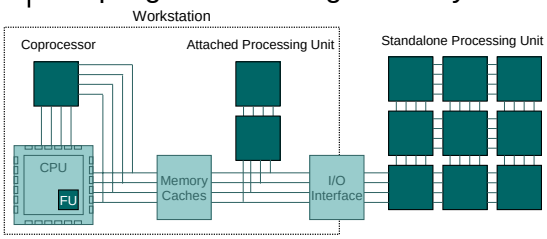
Design	f_{clk} (MHz)	N_{slice}	N_{bram}	$Tput$ (Gbps)	Lat (ns)	Eff ($\frac{Mbps}{slice}$)
UF10-PP3D	**184.16**	16938	0	**23.57**	162.9	1.39



Outline

- Recap
- Multi-FPGA Systems
  - Network topologies
  - System software
  - Theoretical Limits
  - Example Systems
- Application – Logic Emulation

Coupling in a Reconfigurable System



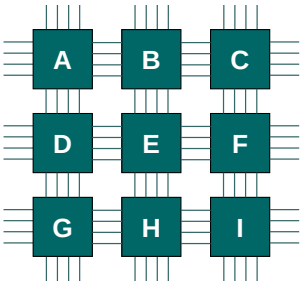
- Many places to put reconfigurable computing components
- Most implementations involve multiple discrete devices
- How should these devices be connected together?

Modern Multi-FPGA Systems

- Large logic capacity
  - All projects end up pushing capacity limits
- Large amount of on-board RAM
  - High speed and high density
  - To support genome, vision and pharmacological apps
- High speed FPGA-FPGA connections
  - To make multiple FPGAs more like one big FPGA
  - Inter-chip connectivity an issue
- Parallel computers in the traditional sense
  - Suitable for spatially parallel applications
  - Transmogrifier-4, BEE2

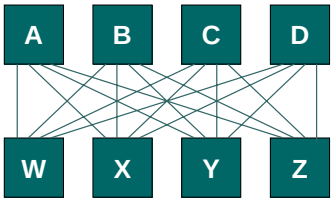
Mesh Topology

- Chips are connected in a nearest-neighbor pattern
- Simplicity is key
- Linear array is essentially a 1-dimensional mesh

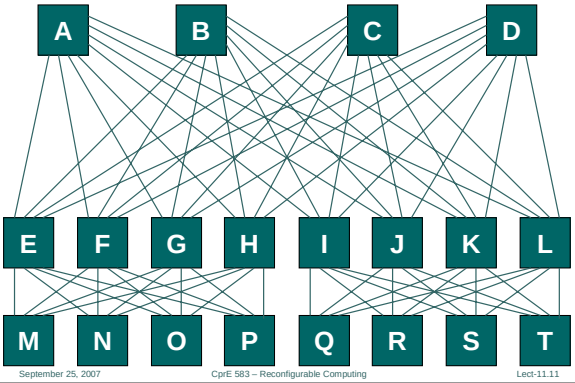


Crossbar Topology

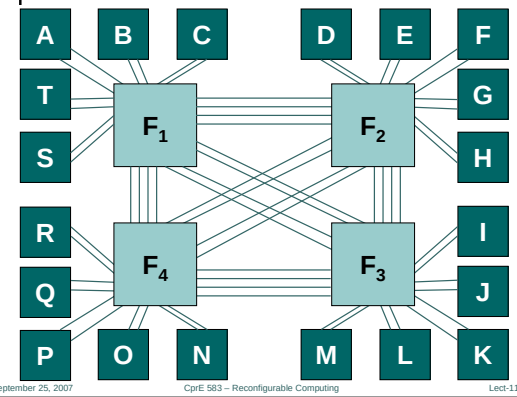
- Devices A-D are routing only
- Gives predictable performance
- Potential waste of resources for near-neighbor connections



Crossbar Hierarchy



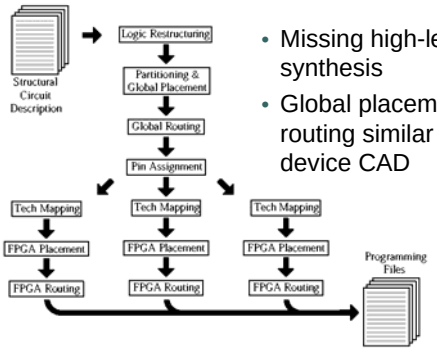
Other Two-Level Schemes



Thought Exercise

- Consider the linear array, mesh, crossbar, hierarchy, and other two-level topologies
- In groups of 2, analyze the average distance needed to communicate given a random placement of functions to FPGAs
  - Can this be represented as a function of N?
- Assume finite number of pins per device
- Best topology wins a prize

Multi-FPGA Synthesis



- Missing high-level synthesis
- Global placement and routing similar to intra-device CAD

Bipartitioning

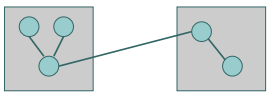
- Perhaps biggest problem in multi-FPGA design is partitioning
  - NP-complete for general graphs
  - Many heuristics/attacks
- Partitioner must deal with logic and pin constraints
- Better to recursively bipartition circuit

KL FM Partitioning Heuristic

- **KLFM** – **F**iduccia-**M**attheyses (**K**ernighan-**L**in refinement)
- Greedy, iterative
  - Pick cell that decreases cut and move it
  - Repeat
- Small amount of
  - Look past moves that make locally worse
  - Randomization

KL FM Algorithm

- Randomly partition into two halves
- Repeat until no updates
  - Start with all cells free
  - Repeat until no cells free
    - Move cell with largest gain (balance allows)
    - Update costs of neighbors
    - Lock cell in place (record current cost)
  - Pick least cost point in previous sequence and use as next starting position
- Repeat for different random starting points



Problems with Meshes

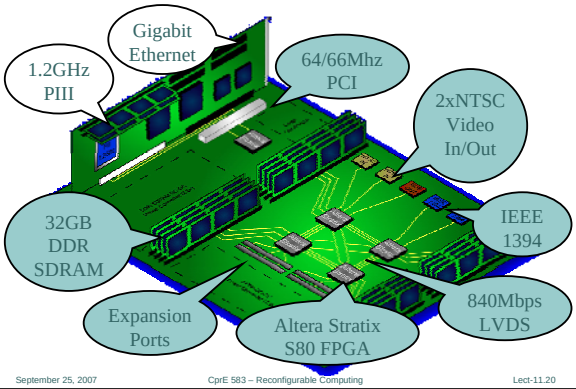
- **Rent's Rule** for the number of wires leaving a partition: $P = KG^B$
- Perimeter grows as $G^{0.5}$ but unfortunately most circuits grow at G^B where $B > 0.5$
- Effectively devices highly pin limited
- What does this mean for meshes?

Not Limited	Gate Limited
– unused FPGA pins – unused FPGA gates	– some unused pins – no unused gates
Pin Limited	**Balanced**
– no unused pins – some unused gates	– no unused pins – no unused gates

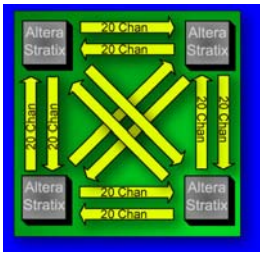
Multi-FPGA Systems

- Transmogrifier-4 (University of Toronto)
- Four Altera Stratix EP1S80F1508C6 FPGAs, each with:
  - 79,040 LUTs
  - 7.4Mb internal block RAM
  - 176 9x9 MACs (4 9x9's can become 1 36x36)
  - 1508 pin flip chips
- Total TM-4 Capacity:
  - 316,160 Luts
  - 29.6Mb internal block RAM
  - 704 9x9 MACs

Transmogrifier-4



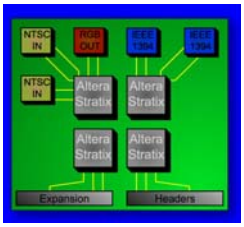
TM-4 FPGA Interconnects



- Differential LVDS
- Run up to 840 Mbps
- Configurable as low speed single ended
- 20 transmit and 20 receive channels between each pair of FPGAs

240 Channels ~ 840 Mbps / Channel ~ 200 Gbps Bandwidth

TM-4 Peripherals



- Video I/O support
  - 2 x NTSC to RGB decoders
  - 1 x RGB video DAC
  - 2 x IEEE-1394 (firewire)
    - 2 x 400Mbps ports per bus
    - Hard link layer
- Expansion headers
  - High-speed connectors

2 NTSC Video In ~ RGB Out ~ 2 400Mbps IEEE-1394

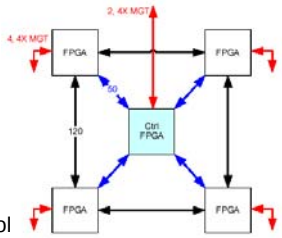
TM-4 Software Support

- Virtual "ports" package
  - Transparent connectivity to host software
- Inter-FPGA router
- Remote access utilities
  - User access manager
  - Remote network TM-4 interface API
- Debugging support
  - On-FPGA logic analyzer support
  - Device simulation models

Handshake Flow Control ~ Burst Modes ~ Interrupt

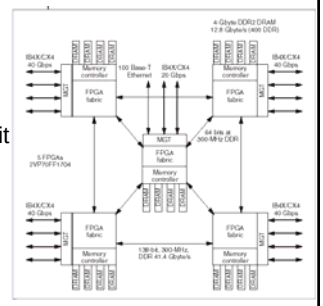
Berkeley Emulation Engine (BEE2)

- Five Virtex-2 Pro XC2VP70 FPGAs, each with:
  - 74,448 LUTs
  - 5.9Mb internal block RAM
  - 328 9x9 MACs
- Four processing elements and one control element
  - 120 bit 200 MHz DDR
  - 48 Gbps link
- Star connection from control node to computing nodes
  - 50 bit 200 MHz DDR
  - 20 Gbps link



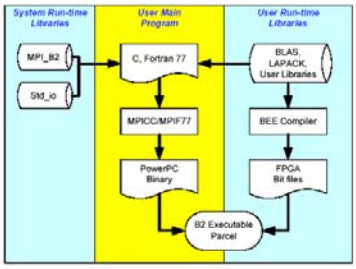
BEE2 Details

- Up to 8 boards in a card cage
- Off-board communication takes place with multi-gigabit transceiver (MGT)
- Lots of off chip DDR DRAM
- Scalable



BEE2 Programming Environment

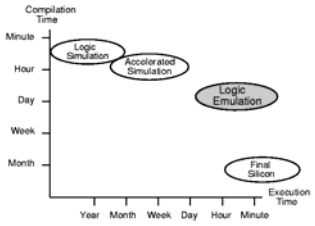
- Dataflow computing style
- Integration with processor programming environment



Logic Emulation

- Custom ASIC circuits – $$$
- ASIC designers want to ensure that the circuit is correct before final stages of design
- Software simulation?
- Logic emulation – circuit is mapped onto a multi-FPGA system
  - Several orders of magnitude faster than software simulation
- The original “killer app” for FPGAs

Logic Emulation (cont.)

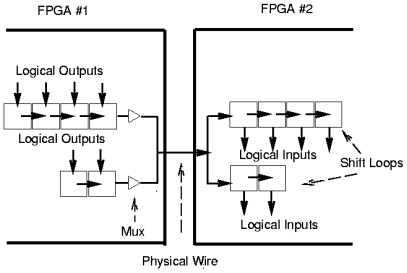


- Emulation takes a sizable amount of resources
- Compilation time can be large due to FPGA compiles

Example System: Virtual Wires

- Goal is to take an ASIC design and map it to multi-FPGA hardware
- Can replace new chip in target system to allow for software development
- Important issues include
  - How is system interfaced to workstation
  - What is interface to target system
  - How can memory be emulated
  - Logic analysis / debugging

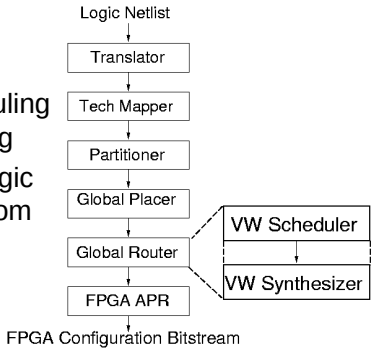
Virtual Wires



- Overcome pin limitations by multiplexing pins and signals
- Schedule when communication will take place

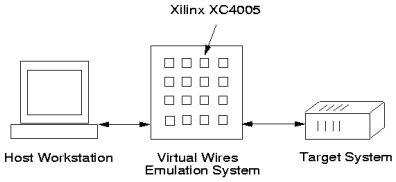
Virtual Wires Software Flow

- Global router enhanced to include scheduling and embedding
- Multiplexing logic synthesized from FPGA logic



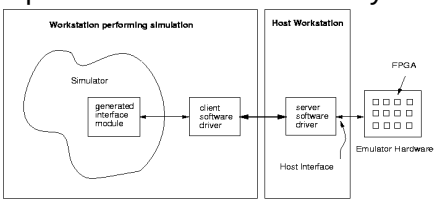
Emulation System Configuration

- Pod interface to target system
- Serial or Sbus interface to host workstation
- (not shown) Physical connection to logic analyzer also a possibility
- Target system must be slowed down to accommodate emulation



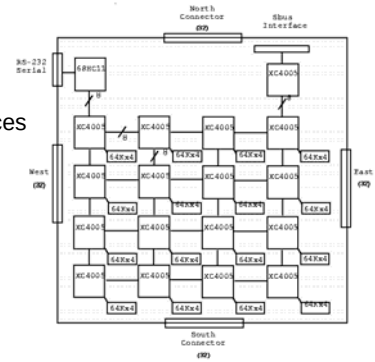
Simulation Acceleration

- FPGA system takes the place of one portion of simulated design
- Inputs transported to FPGA system
- Outputs returned from FPGA system

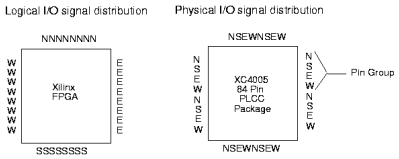


Virtual Wires Emulation Board

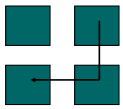
- Pod connectors located along perimeter
- Two host interfaces
- Near-neighbor communication



Device Pin Layout

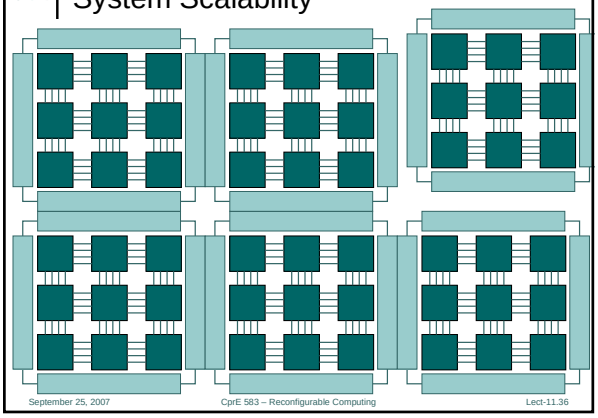


- Many nets may pass through an intermediate FPGA in traversing source to destination



- Physical assignment of IO to pins important to allow device routability at the expense of board routability

System Scalability



Summary

- Most FPGA systems require multiple devices
  - System software involves many steps
  - Bipartitioning has been the subject of much research
- Topologies affect performance and use
  - An active area of research as “devices” migrate inside the chip
- One common use of multi-FPGA systems is logic emulation
  - An example system (virtual wires) uses a near-neighbor mesh with several external interfaces.
  - Virtual wires overcome pin limitations by intelligently multiplexing I/O signals
  - www.mentor.com/products/fv/emulation/vstation_pro
  - www.synplicity.com/products/haps