RiceNIC
Prototyping Network Interfaces

Jeffrey Shafer
Scott Rixner
RiceNIC Overview

Gigabit Ethernet Network Interface Card
RiceNIC Overview

- Reconfigurable and programmable
  - Can alter both FPGA design and program embedded processors
- Built on commercial development board
- Reference design is freely available
- Targeted at research and education applications
RiceNIC Target Applications

- Experimental research into network server architectures
  - Explore hardware/software interface between OS and NIC
  - Explore NIC architectures and desired functions
- Examples using RiceNIC
  - TCP offload (at Rice)
    - Run TCP stack on NIC
    - Moved connection management from host OS to NIC
  - Low Power Networking (at University of Florida)
    - Adaptive MAC varies performance versus energy efficiency
  - Virtual Machines (at Rice, EPFL and HP Labs)
    - Each virtual machine can communicate independently with NIC
Outline

- RiceNIC Introduction
- Competing Development Methods
- NIC Architecture and Implementation
- Applications
  - Education and Research
- Lessons Learned
- Conclusions
How to Build the RiceNIC Prototyping Tool?

- Buy a commercial software programmable NIC?
  - Straightforward to rewrite firmware
  - Used for previous research with mixed results
  - Insufficient performance / flexibility for prototyping
- Build new board from scratch?
  - Highly customizable
  - Expensive / time consuming
- Use commercial prototyping board?
  - Inexpensive / fast

We Chose a Commercial Board. How Did it Turn Out?
Avnet Development Board

- FPGA Development Board
- Virtex and Spartan FPGAs
- 2 PowerPC 405 processors
  - 300 MHz
  - Separate 16kB I and D-cache
- Serial Port

- 256 MB DDR Memory
- 2 MB SRAM Memory
- 56kB on-FPGA memory
- Gigabit Ethernet PHY
- 64-bit / 66 MHz PCI bus

June 9, 2007
RiceNIC - Prototyping Network Interfaces
System Architecture

- PCI Bridge
- Spartan FPGA
- PCI Bus (64b / 66 MHz)
- SRAM (2MB)
- PROM
- UART
- RS-232
- Ethernet PHY (10/100/1000)
- DDR SDRAM (256 MB)
- DDR Controller
-DMA Unit
- BRAM Block
- MAC Control
- PowerPC (300MHz)
- Cache
- Avnet Hardware
- Avnet Memory
- Custom Design

June 9, 2007

RiceNIC - Prototyping Network Interfaces
FPGA Utilization

- Implemented with Xilinx tools (ISE and EDK)
- Used ChipScope on-chip logic analyzer

<table>
<thead>
<tr>
<th>Component</th>
<th>Virtex FPGA</th>
<th>Spartan FPGA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Slice Flip Flops</td>
<td>9,089 / 27,392</td>
<td>33%</td>
</tr>
<tr>
<td>4 input LUTs (Logic)</td>
<td>11,811 / 27,392</td>
<td>43%</td>
</tr>
<tr>
<td>BRAMs</td>
<td>51 / 136</td>
<td>37%</td>
</tr>
<tr>
<td>Occupied Slices</td>
<td>9,164 / 13,696</td>
<td>66%</td>
</tr>
<tr>
<td>Global Clocks</td>
<td>10 / 16</td>
<td>62%</td>
</tr>
<tr>
<td>Digital Clock Managers</td>
<td>5 / 8</td>
<td>62%</td>
</tr>
</tbody>
</table>
Virtex FPGA Placement

- Ethernet MAC
- DDR Controller
- PowerPC Processors
- PLB & Control
- PCI DMA Engine
Debugging Support

- **Software Debugging**
  - Serial Console (RS-232 port)
    - Command line interface to PowerPC processors
    - Runtime debugging / configuration changes / bootstrapping
  - Firmware Profiler
    - Timer based statistical profiler (similar to Oprofile)
    - Exports results via serial console

- **Hardware Debugging**
  - Xilinx Chipscope on Virtex/Spartan FPGAs

Essential Features Not Found in Other NICs
Performance

- TCP streaming test (netperf)
- RiceNIC firmware using 1 processor
- High throughput for packets larger than 1000 bytes
- Spare capacity for researchers to prototype new systems
RiceNIC Application - Education

- Added Network Address Translation module to RiceNIC firmware
- NIC still runs at full Ethernet speeds
- Project time – 1 day
  - Feasible for a class project
- Could have implemented similar module on any commercial software programmable NIC, but...
  - Debugging of NAT module by non-experts greatly assisted by RiceNIC serial console which printed packet traces
RiceNIC Application – Low Power MAC

- Energy Efficient Internet Project †
  *(at USF and UF)*
- Replacing MAC core on RiceNIC FPGA with a custom low-power variant that supports adaptive link rates
  - This research could not be done on any software-programmable NIC!

† The Energy Efficient Internet Project: http://www.csee.usf.edu/~christen/energy/main.html
RiceNIC Application Networking in Virtual Machines †

- Each guest OS can talk directly to single shared NIC
  - Separate interfaces for concurrent guests
- Experimental research project with many modifications
  - FPGA provides isolated contexts in memory + event notification
  - Firmware provides packet multiplexing / demultiplexing
  - Xen / OS modifications
  - Used 1 PowerPC processor and 12MB of RAM

Prototyping (with RiceNIC) critical to project success

Could not use software programmable NIC
- Needed to change hardware architecture
- Software emulation too slow

RiceNIC prototype ran at full speed

Used RiceNIC to experimentally determine key architectural features
- Minimum on-NIC packet buffer size per virtual machine

Could not obtain equivalent results via simulation in a timely fashion
Lessons Learned

- Use a commercial development board
  - Avnet card perfect for RiceNIC
    - Contains large FPGA, PCI, GigE, and memory
  - Board arrived fully tested / documented
    - Implementation can begin immediately
  - Reduced up-front project expenses
    - For low volume RiceNIC, might never be cheaper to fabricate a custom board
Lessons Learned – Design Time

- **Design time**
  - Hardware (FPGA) design
    - 1 year / 1 graduate student
  - Software (firmware / driver) design
    - 1 month / 1 graduate student
    - Significant prior experience in writing NIC firmware

- **Development stages**
  - Learning Xilinx ISE / EDK tools – 1 month
    - Time spent would be comparable with competing tools
  - Design / Implementation – 9 months
  - Testing – 3 months
Lessons Learned – Testing

- PCI core required extensive iterative testing
  - Must work correctly on a wide variety of host computer systems with non-deterministic bus transfer and error timing characteristics
- Simulator can ignore these low-level issues…
  - … but working prototype must function perfectly at even the most detailed level
- Saving grace - Fully-functional prototype lends confidence to experimental research
Lessons Learned

- FPGAs well matched with requirements
  - Clock rates required for gigabit NIC are achievable with modern FPGAs
    - 66 MHz PCI interface, 100 MHz general logic, 300 MHz embedded processors
  - Saved $$ and time versus creating an ASIC
Lessons Learned

- How would you build a 10 Gigabit NIC?
  - Make same design choice to use a commercial prototyping board
  - Newer Avnet boards have larger FPGAs, PCI Express links, and 10G interfaces
    - Prototyping boards closely follow the commercial state-of-the-art
RiceNIC Usage

- RiceNIC is **free for research and education applications**
  - Platform includes the FPGA configuration (bitstream), VHDL source code, embedded PowerPC firmware, and Linux device driver
- Purchase Avnet Virtex-II Pro development board
- Additional requirements to modify FPGA designs
  - Xilinx development software (ISE, EDK, Chipscope)
  - New licenses for the Xilinx MAC and PCI cores
  - Can still modify firmware without any FPGA changes or additional licenses
Conclusions

- Research into network system demands experimental prototypes
  - Complex, asynchronous interactions among system components
  - Simulation insufficient for new architectures
  - Prototyping tools such as RiceNIC are critical for the development and understanding of future computer systems
- Development method was successful
- RiceNIC already used in several research projects at several institutions
- Would we use a commercial prototyping board to build the RiceNIC again? Yes!
Questions?

- RiceNIC website:

http://www.cs.rice.edu/CS/Architecture/ricenic/