Compute Node Design for DAQ and Trigger Subsystem in Giessen

Justus-Liebig University in Giessen
Outline

• Design goals

• Current work in Giessen
  – Hardware
  – Software

• Future work
Design Goals

For compute networks and nodes in PANDA, they are expected to have:

• High computing and communication capabilities to process large amount of data.
• Highly universal, configurable and scalable platform for multiple applications.
• Convenience to be modified and upgraded in-field.
Present Work in Giessen

- Prototype design
- Based on ML403 commercial board from Xilinx.
ML403 Evaluation Board

- **Xilinx Devices:**
  - Virtex 4 XC4VFX12-FF668-10 FPGA

- **Memory:**
  - 64 MB DDR SDRAM
  - 8 Mb SRAM
  - 8 MB Linear Flash
  - 32 Mb Platform Flash
  - System ACE CF card
  - 4 Kb IIC EEPROM

- **Interfaces:**
  - 10/100/1000 RJ-45 Ethernet Port
  - RS-232 Serial Port
  - JTAG
  - 3 USB Ports
  - 2 PS/2 Connectors (Keyboard/Mouse)
  - 2 Audio (Microphone/Head Phone)
  - General Purpose I/O: Buttens and LEDs
  - ......

- **Display:**
  - 16 x 2 Character LCD
  - DB 15 VGA Display

- **Clocks:**
  - 100 MHz Oscillator
  - 2 Clock Sockets
# Virtex 4 FX FPGA

<table>
<thead>
<tr>
<th>Device</th>
<th>Configurable Logic Blocks (CLBs)(1)</th>
<th>Block RAM</th>
<th>PowerPC Processor Blocks</th>
<th>Ethernet MACs</th>
<th>RocketIO Transceiver Blocks</th>
<th>Total I/O Banks</th>
<th>Max User I/O</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Array(3) Row x Col</td>
<td>Logic Cells</td>
<td>Slices</td>
<td>Max Distributed RAM (Kb)</td>
<td>18 Kb Blocks</td>
<td>Max Block RAM (Kb)</td>
<td>DCMs</td>
</tr>
<tr>
<td>XC4VXS25</td>
<td>64 x 40</td>
<td>23,040</td>
<td>10,240</td>
<td>160</td>
<td>128</td>
<td>128</td>
<td>2,304</td>
</tr>
<tr>
<td>XC4VXS35</td>
<td>96 x 40</td>
<td>34,560</td>
<td>15,360</td>
<td>240</td>
<td>192</td>
<td>192</td>
<td>3,456</td>
</tr>
<tr>
<td>XC4VXS55</td>
<td>128 x 48</td>
<td>55,296</td>
<td>24,576</td>
<td>384</td>
<td>512</td>
<td>320</td>
<td>5,760</td>
</tr>
<tr>
<td>XC4VFX12</td>
<td>64 x 24</td>
<td>12,312</td>
<td>5,472</td>
<td>86</td>
<td>32</td>
<td>36</td>
<td>648</td>
</tr>
<tr>
<td>XC4VFX20</td>
<td>64 x 36</td>
<td>19,224</td>
<td>8,544</td>
<td>134</td>
<td>32</td>
<td>68</td>
<td>1,224</td>
</tr>
<tr>
<td>XC4VFX40</td>
<td>96 x 52</td>
<td>41,904</td>
<td>18,642</td>
<td>291</td>
<td>48</td>
<td>144</td>
<td>2,592</td>
</tr>
<tr>
<td>XC4VFX60</td>
<td>128 x 52</td>
<td>56,880</td>
<td>25,280</td>
<td>395</td>
<td>128</td>
<td>232</td>
<td>4,176</td>
</tr>
<tr>
<td>XC4VFX100</td>
<td>160 x 68</td>
<td>94,896</td>
<td>42,176</td>
<td>659</td>
<td>160</td>
<td>376</td>
<td>6,768</td>
</tr>
<tr>
<td>XC4VFX140</td>
<td>192 x 84</td>
<td>142,128</td>
<td>63,168</td>
<td>987</td>
<td>192</td>
<td>552</td>
<td>9,936</td>
</tr>
</tbody>
</table>
Hardware Platform
Hardware Platform

- Most components are inside the FPGA, such as Memory Controller, Tri-mode Ethernet, and Custom Logic.
- The programmable capability of FPGA makes it easy to modify and scale the system.
- Via custom logic design, the hardware platform could be universal for multiple applications.
- Custom logic could be designed with HDL (VHDL or Verilog) for different purposes, for instance, event-select algorithm.
- Design software:
  - EDK 8.1: high level embedded systems design.
  - ISE 8.1: logic design.
  - Modelsim 6.0: simulation.
Design in EDK
Design in EDK
Tri-mode Ethernet

- 2 Tri-mode Ethernet MACs in V4FX12.
- 10/100/1000 Mbps modes. (1000 Mbps is most interesting for us.)
- Gbit Ethernet provides guaranteed communication performance for interconnected networks.
Flash Memory

- FPGA configuration data and operating system image are both stored in flash memory chips.
- The hardware platform could be updated by this method: 1. power on and boot the system. 2. update the configuration data in flash. 3. reboot the system. (talk about it later in detail)
- Also the operating system image.
Software

• An open source Linux 2.6 runs on PPC 405.
• Totally free and well maintained by many people around the world.
• Many device drivers and other resources from Internet.
• Powerful network support, suitable for our case.
Software

With the OS support,

- the hardware and software platform could be universal for multiple uses via different application programs.
- physicists could write their own application programs with high level programming languages, such as C and C++, without the knowledge of circuit design.
- applications could be more portable among different architectures.
Kernel Compilation

• Getting the kernel sources
  – www.kernel.org
  – www.penguinppc.org

• Configuring the kernel
  – ARCH := ppc
  – Select the features which are to be included in the kernel.

• Building a PowerPC cross-compile toolchain
  – Cross-compiler: powerpc-405-linux-gnu-gcc

• Then, compiling...
Network File System

• Generating the root file system
  – busybox

• Mounting the root file system via NFS
  – “Diskless” booting and the on-board flash memory space could be saved.
  – The file system could be easily changed through PC operations, without frequently copying files into CF card or flash chips.
  – Convenient especially for in-field file modification.
System Upgrading

• The OS kernel image and the hardware configuration data are both stored in the flash chips on board.
• After the booting of Linux, the flash memory could be addressed via its device driver. Then the configuration data and kernel image could be upgraded in Linux.
• Reboot the system and hence the updated hardware and OS begin working.
• Operated remotely and need no download cable.
Current Progress

• Now Linux 2.6 is running on the ML403 platform, with Gbit Enet data transfer, NFS, reserved memory for buffering incoming data, and many other features supported.

• Host (PC) and client (board) transceiving application programs have been realized to transfer packets along the dataflow with UDP protocol successfully.
Future work

• Measuring the detailed latency and throughput parameters of the network
• Well-designed transceiving programs for Enet data transfer
• Custom logic design for specific algorithm
• Many things to do...
Thank You!