Three Steps To Accelerator Success
SAAHPC 2009

Allan Cantle, President & Founder
a.cantle@nallatech.com
July 2009
www.nallatech.com
Agenda

» Nallatech Overview
» A Common perspective on Accelerators
» Three Steps to Accelerator Success
  1. Assess how your application can be accelerated
  2. Avoid unnecessary Host to Accelerator Data Movement
  3. Select the Appropriate Accelerator & Implement Port
» FSB Accelerator Update
» Summary
Nallatech – The FPGA Acceleration Specialist

FPGA Acceleration for over 15 Years

PCle based acceleration
Many I/O & Compute Options

Intel Xeon Socket Filler (FSB & QPI)
FPGA Compute & I/O modules

Ruggedised Defence Computing VME/ VXS & XMC

Miniaturised Near Sensor Processing
High Level Tools For System & Software Engineers

- HDL User Application Code
- ANSI-C User Application Code
- Simulation
- Optimize code to extract maximum parallelism
- C-to-FPGA Compilation
- Application Builder: Application Integration Tool
- Xilinx ISE Implementation Tools
- Design Flow

- Choice of multiple C-based tool flows
- Automatic FPGA bitstream generation
- Supported by the Intel Quick Assist Acceleration Abstraction Layer

- Runtime
  - Intel Quick Assist AAL
  - FPGA API

Commercial In Confidence. Copyright ©2007-8, Nallatech.
A Common perspective on Accelerators

» Accelerators are performant for a subset of Functions
  » Maybe 10x to 100x+ Acceleration for a specific function

» Have the Functions readily available in well understood Library format for the Accelerator. E.g. MKL

» Rebuild your Application to use the Accelerator Libraries instead of the normal processor Libraries

» Hey Presto : You have your new Application running faster with the onboard Accelerator
Three Steps to Accelerator Success
1) Assess how your application can be accelerated
1) Assess how your application can be accelerated

Sequential Acceleration

Application Acceleration = 2.7x
1) Assess how your application can be accelerated

**Parallel Acceleration**

- Application Acceleration = 3.3x
- Application Acceleration = 2.9x

**Code Segment 4 Acceleration**
1) Assess how your application can be accelerated

Parallel Acceleration

<table>
<thead>
<tr>
<th>Time (Secs)</th>
<th>Code Segments 4+2 Acceleration</th>
</tr>
</thead>
<tbody>
<tr>
<td>1x</td>
<td>CS(1+3)</td>
</tr>
<tr>
<td>2x</td>
<td>CS(1+3)</td>
</tr>
<tr>
<td>5x</td>
<td>CS(1+3+5:7)</td>
</tr>
<tr>
<td>10x</td>
<td>CS(1+3+5:7)</td>
</tr>
</tbody>
</table>

Application Acceleration = 5x
2) Avoid unnecessary Host to Accelerator Data Movement

Peak Bandwidths

- **QPI** = 12.5+12.5 GB/s
- **FSB** = 8.5 GB/s
- **PCIex8** = 2+2 GB/S

Application

Xeon Host Processor

Function(s)

Accelerator
2) Avoid unnecessary Host to Accelerator Data Movement

<table>
<thead>
<tr>
<th>FPGA Function I/O Bandwidth requirements &amp; Performance</th>
<th>Single Function</th>
<th>Multiple Functions that will fit into a V5SX240</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Required I/O Bandwidth GBytes/s</td>
<td>Performance G(Fl)ops</td>
</tr>
<tr>
<td>NTBody gravitational (Bandwidth ~ 0)</td>
<td>0</td>
<td>7.5</td>
</tr>
<tr>
<td>Random number (Sobol pipelined)</td>
<td>0.8</td>
<td>614.4</td>
</tr>
<tr>
<td>Convolution (floating point, Image processing, 0 Frame latency 11x11 kernel)</td>
<td>2.4</td>
<td>72.3</td>
</tr>
<tr>
<td>Complex FFT (pipelined, single precision, 1024 points)</td>
<td>4.8</td>
<td>30</td>
</tr>
<tr>
<td>Black Scholes (single precision)</td>
<td>6.4</td>
<td>14</td>
</tr>
<tr>
<td>Convolution (floating point, Image processing, 1 Frame latency 11x11 kernel)</td>
<td>2.4</td>
<td>6.3</td>
</tr>
<tr>
<td>Pre-Stack-Time-Migration</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>Encryption AES (pipelined)</td>
<td>22.4</td>
<td>NA</td>
</tr>
</tbody>
</table>
2) Avoid unnecessary Host to Accelerator Data Movement

<table>
<thead>
<tr>
<th>FPGA Function I/O Bandwidth requirements &amp; Performance</th>
<th>Single Function</th>
<th>Multiple Functions that will fit into a V5SX240</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Required I/O Bandwidth GBytes/s</td>
<td>Performance G(FL)ops</td>
</tr>
<tr>
<td>NTBody gravitational (Bandwidth ~ 0)</td>
<td>0</td>
<td>7.5</td>
</tr>
<tr>
<td>Random number (Sobol pipelined)</td>
<td>0.8</td>
<td>614.4</td>
</tr>
<tr>
<td>Convolution (floating point, Image processing, 0 Frame latency 11x11 kernel)</td>
<td>2.4</td>
<td>72.3</td>
</tr>
<tr>
<td>Complex FFT (pipelined, single precision, 1024 points)</td>
<td>4.8</td>
<td>30</td>
</tr>
<tr>
<td>Black Scholes (single precision)</td>
<td>6.4</td>
<td>14</td>
</tr>
<tr>
<td>Convolution (floating point, Image processing, 1 Frame latency 11x11 kernel)</td>
<td>2.4</td>
<td>6.3</td>
</tr>
<tr>
<td>Pre-Stack-Time-Migration</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>Encryption AES (pipelined)</td>
<td>22.4</td>
<td>NA</td>
</tr>
</tbody>
</table>

- = I/O BW within PCIeX8 & FSB Peak Bandwidth Capability
- = I/O BW within QPI Peak Bandwidth Capability
- = I/O BW not achievable (Acceleration is Data Bound)
2) Avoid unnecessary Host to Accelerator Data Movement

» Serious Issue with Accelerators being Data Bound
  » Even with QPI Bandwidths
  » Exacerbated by partitioning at the Library Function Interface
  » Actual Host to Accelerator Bandwidths will be less than peak

» Solution
  » Partition the accelerator at the point of least Data Movement in the application whilst encapsulating the Acceleratable Function
3) Select the Appropriate Accelerator
3) Select the Appropriate Accelerator

**Processor**
- Multi Core
- 2 – 8 cores
- 40+ GF Peak DPFO
- 100+ GF Peak SPFP
- ~130W
- Easy to Program

**Typical Processor based Accelerator**
- Many Core
- 8 – 100+ cores
- 100+ GF Peak DPFP
- 1+ TF Peak SPFP
- Pipe/Ring &/or Mesh
- 25W – 300W+
- Difficult to Program

**FPGA Accelerator**
- Many Many Core
- 1000+ cores
- 30+ GF DPFP
- 100+ GF Peak SPFP
- 1-10 TOP Bit/Integer
- ~25W
- Very Difficult to Program
Intel Xeon Accelerator Modules

Intel Front Side Bus FPGA Accelerators
The Industry’s only Xilinx Virtex-5 FSB Accelerator Module

- 64-bit 1066MHz FSB interface
- 8GB/s peak bandwidth
- 105ns host latency
- 256GB direct system memory access
- Intel MP platform compatible

- Modular product – optimization for different applications
- Xilinx Virtex-5 FPGA technology
- Supported by Intel QuickAssist AAL
- C → FPGA compiler support
Integrated Development Platform
Unlock socket by lifting lever
Building the accelerator stack - 2

- Insert pin field
Building the accelerator stack - 3

- **FSB-BASE** module plugs directly into Intel Xeon socket
- Deals with low level FSB interface
- Often referred to as the “Bridge” from the host to the user logic
Building the accelerator stack - 4

- Heatsink fitted to FSB interface FPGA
ISI high density connectors provide LVDS links to upper module(s)
Fits onto connectors of FSB-BASE module
Building the accelerator stack - 7

- FSB-COMPUTE module #1 mates with connector
- Heatsinks applied to user FPGAs
Another ISI high density connector mates with FSB-COMPUTE #1 providing LVDS links to another upper module.
Building the accelerator stack - 9

- FSB-COMPUTE #2 mates with the connector
- Heatsinks applied to user FPGAs
Building the accelerator stack - 10

- The final ISI high density connector provides another LVDS link
The FSB-EXPANSION module mates with the connector, completing the stack of 5 Xilinx user FPGAs + FSB interface FPGA.
Building the accelerator stack - 12

- Heatsinks are fitted to user FPGA of FSB-Expansion module
Building the accelerator stack - 13

- The complete stack...
Stack Level Functional Block Diagram

64 LVDS pairs @ 800MHz = 6.4GB/sec
Latency = 20ns

128 LVDS pairs @ 800MHz = 12.8GB/sec
Latency = 20ns

64-bit/1,066MHz Front Side Bus
Chassis Level Architecture

5 SX240’s per stack,
3 Stacks/Chassis
= 15 FPGAs / Chassis

Direct FPGA Stack to FPGA Stack Communications

Off Module I/O
Summary

» Think Carefully about how to utilise an accelerator
  » Can save a lot of wasted porting effort
  » A short Feasibility study can pay big dividends

» Fast Functions tend to need fast data feeds
  » Check your Host/Accelerator Partitioning point

» FPGA’s are a good all round accelerator
  » They excel for certain functions but are also competent at most computation.
  » Mature Commercially Stable Accelerator
  » They are very power efficient processors