FPGA HPC – The road beyond processors

Jeff Mason
Xilinx Research Labs
mason@xilinx.com
Outline

• FPGA High Performance Computing (FHPC)
• Xilinx FPGAs
• What can be done today
• Why not more (Double Precision FP)
• The first stepping stone, ease of use
• Future stones, ESL
• Partial Reconfiguration
• Platforms of tomorrow
Pentium Clock Speed Over Time

- 3 GHz introduced in 2002
- Fastest processor today is 3.8GHz
- No increased frequency at 90nm
- Moore’s law no longer improving clock speed
The power consumption of today’s advanced computing systems is rapidly becoming the limiting factor with respect to improved/increased computational ability.

- 80% increase in power density/generation
- Voltage scales by ~0.8
- 225% increase in current consumption/unit area!

Source: Shekhar Borkar, Intel
Xilinx Virtex5 for HPC

- 6 input LUTs
- 550 MHz DSP48E with 25x18 multipliers for better single precision floating point and DSP acceleration.
- 11.6 Mbits of flexible embedded Block RAM
- 100Mbps–3.2 Gbps transceivers for fast chip-to-chip communication
## Choices

<table>
<thead>
<tr>
<th></th>
<th>Intel Woodcrest</th>
<th>Xilinx V5LX330</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>GOPS</strong></td>
<td>• 24G 64-bit Op/s</td>
<td>• 2.59T 64bit Op/s</td>
</tr>
<tr>
<td><strong>FLOPS</strong></td>
<td>• 48 GFlop/s for 2 chip system</td>
<td>• 60 GFlop/s</td>
</tr>
</tbody>
</table>
| **BW external memory** | • CPU into MCH  
|                      | 2x 10.6 Gbyte/s                                         | • Select IO  
|                      | • Bensley MCH  
|                      | 38.4 GByte/s for 6channels                             | 56 GByte/s                                          |
| **BW internal memory** | • L1 Cache BW  
|                      | 16 Gbyte/s                                              | • BRAM BW  
|                      | • Register File BW  
|                      | 384 Gbyte/s                                             | 1.8 TByte/s                                         |
|                      |                                                          | • FF BW  
|                      |                                                          | 10.4 TByte/s                                        |
| **Power**            | • 80W-130W                                               | • 15W                                               |

### Microprocessor:
- All functions are hardened in silicon and you pay for them whether you use them or not
- Can’t use that silicon for something else
- Small number of parallel operations; often idle

### FPGA:
- Most of logic is configurable
- Processing power doesn’t go to waste – the same silicon can be used for many different functions
- Different operations map to different silicon allows massive pipelining; lots of parallelism
### FPGA Acceleration Examples – today!

<table>
<thead>
<tr>
<th>Applications</th>
<th>HW (FPGA)</th>
<th>SW Only</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hough &amp; inverse Hough processing</td>
<td>2 seconds of processing time @20Mhz</td>
<td>12 minutes processing time</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Pentium 4 - 3Ghz</td>
</tr>
<tr>
<td></td>
<td><em>370x faster</em></td>
<td></td>
</tr>
<tr>
<td>AES 1MB data processing/cryptography rate Encryption Decryption</td>
<td>424 ms/19.7 MB/s</td>
<td>5,558 ms / 1.51 MB/s</td>
</tr>
</tbody>
</table>
|                                       |                                               | 424 ms / 19.7 MB/s                           | *
|                                       |                                               | 5,562 ms / 1.51 MB/s                         | 13x faster |
| Smith-Waterman ssearch34 from FASTA   | 100 sec FPGA processing                       | 6461 sec processing time                     |
|                                       | *64x faster*                                  | Opteron                                      |
| Multi-dimensional hypercube search    | 1.06 Sec FPGA@140Mhz Virtex II                | 119.5 Sec                                   |
|                                       |                                               | Opteron - 2.2 Ghz                            | 113x faster |
| Monte-Carlo Analysis                  | 10 sec of Processing @200 Mhz                 | 100 sec processing time                      |
| 64,000 paths                          | FPGA system                                   | Opteron - 2.4 Ghz                            | 10x faster |
|                                       |                                               |                                              |
| BJM Financial Analysis                | 242 sec of Processing @61 Mhz                 | 6300 sec processing time                     |
| 5 million paths                       | FPGA system                                   | Pentium 4 – 1.5 Ghz                          | 26x faster |
|                                       |                                               |                                              |
| Black-Scholes                         | 18 msec FPGA@110Mhz Virtex-4                  | 3.7 Sec 1M iterations                        |
|                                       |                                               | Opteron - 2.2 Ghz                            | *203x faster* |

- **Celoxica provided data**
Why isn’t FHPC everywhere?

• No one has yet solved the programming model problem to the HPC programmers satisfaction
  – HPC programmers will NOT program in HDL or other hardware-centric languages
    • More than a tool issue – HDL requires a different way of thinking about how to design
    • Not interested in how to program but rather in the results of programming
  – Predefined libraries don’t solve the problem
    • Limits opportunity to places where the supplier has enough application expertise to develop IP
    • Customers want to add their own “secret sauce” to the IP
More reasons why isn’t FHPC everywhere?

• The hardware is not standard
  – Each new FHPC based system requires recompilation for memory access, processor access and more

• FPGAs are viewed as co-processors
  – Every system requires an INTEL/AMD processor or it is not considered.
  – One processor for 100s (1000s) of FPGAs is not even on the radar screen

• The complete solution does not exist
  – Software/Hardware partitioning is still an art, not a single button click of science

• FHPC is similar to eating Brussels sprouts – it is good for you but it has always been less than enjoyable to chock it down in the past
Double Precision Floating point

• When will Xilinx release hard DPFP blocks:
  – Probably never. One user’s IP is another's wasted silicon.
  – No customer will ever be happy with mix of hard IP on a given device, it is always a compromise for all customers
• When will Xilinx release a chip more suited to HPC with large amounts of DSP blocks:
  – As soon as we can make a business argument that it will sell.
  – If you have $2M extra dollars for this endeavor, see me after the talk.
• Xilinx makes programmable logic, not hardened IP
• How fast/big is the current DP Floating point really?
# FPGA Double Precision resources

<table>
<thead>
<tr>
<th></th>
<th>Opteron dual-core</th>
<th>Opteron quad-core</th>
<th>Virtex4 LX200</th>
<th>Virtex5 LX330</th>
<th>Virtex5 Theoretical chip</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2.8 GHz 95 watts</td>
<td>2.3 GHz 95 watts</td>
<td>185 MHz 26 watts</td>
<td>237 MHz 26 watts</td>
<td>356 MHz 26 watts</td>
</tr>
<tr>
<td>mult/add</td>
<td>11.2</td>
<td>36.8</td>
<td>15.9(^A)</td>
<td>28.0(^B)</td>
<td>59.1(^C)</td>
</tr>
<tr>
<td>all mult</td>
<td>5.6</td>
<td>18.4</td>
<td>12.0</td>
<td>20.8</td>
<td>56.5</td>
</tr>
<tr>
<td>all add</td>
<td>5.6</td>
<td>18.4</td>
<td>23.9</td>
<td>55.3</td>
<td>45.5</td>
</tr>
</tbody>
</table>

A) 43 full mults plus 43 adds @ 185 MHz
   \(6 \times 16 = 96\) DSP
   \((6 \times 387) + (37 \times 1229) + (43 \times 637) = 75,186 < 75,588\)

B) 61 full mults plus 61 adds @ 237 MHz
   \(19 \times 10 = 190\) DSP\(\text{e} < 192\)
   \((19 \times 131) + (42 \times 615) + (61 \times 265) = 44,484 < 45,090\)

C) 69 max mults plus 69 full adds @ 356 MHz
   \((69 \times 10) + (69 \times 3) = 897\) DSP\(\text{e} < 960\)
   \((69 \times 131) + (69 \times 244) = 25,875 < 25,890\)

D) This does not YET exist – talk to me to help make it happen!

*courtesy of Dave Strenski, Cray Research*
First step to FHPC

• Make it easy to get hardware into users hands
• Make it easy for the users to use this hardware and get ‘enough of a performance’ increase to be useful
• Provide a decrease in power consumption
• Do not try to be the fastest thing around when being as fast with less power is sufficient
• Do something now, not in 5 years
Ease of use

• HPC programmer must be able to access FHPC as easily as porting to new processor
  – FPHC tools must place ease of use as paramount to performance. A 2 times speed up with no effort is MUCH more appealing than a 20 times speed up with work

• Compiling High Level languages into Massively Parallel Systems – CHiMPS
  – Xilinx research project to do just this
  – It works now!
Future steps are much easier

- Much greater performance gains are available with other Xilinx or vendor tools at the expense of code rewrite
- Much easier to expand number of nodes in a cluster if user already has a cluster than it is to buy a new cluster
- Xilinx Electronic Systems Level (ESL) partners already provide many of these tools that get significant performance increases
Partial Reconfiguration

• Not just a research idea looking for a solution
• Required to keep a single chip ‘active’ talking to buses/memories while a new program is loaded into it.
• Alternative is to use separate chip to talk to bus/memory at cost of memory latency
• Only Xilinx currently provides this functionality!
Limitations of current platforms

- Currently no standardized FPHC platform
- FPGAs are second class citizens relegated to the whims of a processor
- Typically many-to-one ratio of processors to FPGA
- Memory accesses by FPGA are slow (or non-existent) compared to accesses by processors
- Memory coherency between FPGA and processor is missing though it exists between multi-core processors
Today's platforms for FHPC

- Some connect to system with PCI/USB bus
- Co-processor models replace unused processor and connect to main processor with Hyperlink or FSB
- Some are their own standalone system
Platform of tomorrow

- Standardized so it can be targeted by many vendors
- Memory access consistency between processors and FPGAs, including coherency
- FPGAs are not co-processors!
  - Many to one ratio of FPGAs to processors
  - Node might have 1 processor and many FPGAs – or even no processor and many FPGAs
- HPC tools of today work seamlessly (MPI)
- New HPC tools that take advantage of multi-FPGA systems without overhead necessary in processor world can now be developed
Tomorrows platform for FHPC
Non processor computation

- Imagine the following
  1. Blade in a cluster that has only FPGAs. CPU blades also exist and are used for things that CPUs are good at (file IO, internet, …)
  2. All blades in a cluster have only FPGAs. If processor functionality is needed, then embedded soft processor is configured and used in FPGA
  3. Super-computer with thousands of FPGAs for each processor
Thank you

• The concept is interesting and well-formed, but in order to earn better than a "C," the idea must be feasible.