Design for Higher Performance and Lower Power with OpenCL on Altera FPGAs
Agenda

- What is an FPGA?
- Why FPGAs?
- OpenCL™ overview and programming model
- Benefits of OpenCL on Altera® FPGAs
- Next Steps
What is an FPGA?
What is an FPGA?

- Field Programmable Gate Array (FPGA)
- LABs (Logic Array Blocks) arranged in an array
- Programmable interconnect creates custom functionality regions in the array
  - Arbitrary complexity control instructions
  - Arbitrary width data instructions
  - Arbitrary size memory
- Processor units customized to your application
High Level Architecture Difference between FPGA & GPU
High Level Architecture Difference between FPGA & GPU
High Level Architecture Difference between FPGA & GPU
High Level Architecture Difference between FPGA & GPU

Host

(Compute) Device

Compute Unit

Processing Element
High Level Architecture Difference between FPGA & GPU
FPGAs – The Ultimate Parallel Processor

- **Few parallel tasks**
  - Minimal latency (branch prediction)
  - Complex decision making (control compute intensive)

- **Few parallel tasks**
  - Low latency
  - Simple decision making (data path compute intensive)

- **Few SPMD engines**
  - Large latency
  - Extreme floating point instruction parallelization (performance)
  - Simple decision making (array compute intensive)

- **Parallel SPMD engines**
  - Low latency (large local memory)
  - Instruction pipelining and task parallelization
  - Moderate decision making
  - Reasonable floating point
  - Integer arithmetic with masking operations (parallel task compute intensive)
Why Choose FPGAs?
Maximize Throughput
Minimize Latency

Quick Data Access
- Avoid transfer/copy
- Work in local memory instead of shared memory
- Coalesce access

More Operations Per Second
- Pipelining
  - Instructions
  - Processes
- Loop unrolling
- Duplication (SPMD)
- Multi-threading (SMT)
Why should I choose an FPGA?

- **Flexible custom architecture**
  - Configurable local memory size
  - High throughput data parallelization with custom pipeline engines
  - High performance task parallelization with dedicated processors
  - Huge power savings with task specific hardware

- **Parallel architecture**
  - *Dynamic Pipeline Parallelism*
  - Attempt to create a deeply pipelined representation of a kernel
  - On each clock cycle, we attempt to send in input data for a new thread
  - Method of mapping coarse grained thread parallelism to fine-grained FPGA parallelism
Why should I choose an FPGA?

- High-end GPU cards may exceed 250W power requirements

- Power and cooling has become a top concern among HPC data centers
  - Energy prices have been hovering at near historic levels
  - Processor based design has increasingly come up against the power wall
    - More challenging to obtain higher single-core performance while maintaining reasonable power
  - Companies are increasingly sensitive about reducing their carbon footprint
    - The “Green Movement”

Source: IDC, 2010
Why should I choose an FPGA?

- **Faster I/O Handling**
  - Advanced programmable logic blocks connect directly to row or column interconnect
  - Control available I/O features

- **High Speed Data Throughput**
  - Transceivers
  - Stratix V FPGAs support transceivers with data rates up to 14.1 Gbps and 28.05 Gbps
  - Support for [40G and 100G Ethernet](#)

- **OpenCL Channels**
  - Enabling data passing between
    - I/O → Kernel
    - Kernel → Kernel (bypasses the need to access global I/O)
    - Kernel → I/O
OpenCL Overview and Programming Model
OpenCL Overview

- **OpenCL is a software programming model**
  - Uses Standard C language
  - Uses OpenCL C extensions (adds parallelism to C)
  - Includes API (open standard for different devices)

- **Provides increased performance**
  - CPU offload
  - Performance via hardware acceleration

- **Portable, royalty free open standard**
  - Managed by large industry-wide consortium
  - 12 promoters, 79 contributors
Altera SDK for OpenCL

OpenCL
Host Program + Kernels

Standard C Compiler
Executable File

SDK for OpenCL

Binary Programming File

x86
SoC Solution

- ARM Cortex-A9 Host processor and FPGA accelerator in one package
  - Lower cost
  - Power efficient
  - Real-time system acceleration

Notes:
(1) Integrated direct memory access (DMA)
(2) Integrated ECC
OpenCL Programming Model

Host Program

```c
main() {
    read_data(...);
    manipulate(...);
    clEnqueueWriteBuffer(...);
    clEnqueueNDRange(..., sum, ...);
    clEnqueueReadBuffer(...);
    display_result(...);
}
```

Kernel Program

```c
__kernel void
sum(__global float *a,
    __global float *b,
    __global float *y)
{
    int gid = get_global_id(0);
    y[gid] = a[gid] + b[gid];
}
```
OpenCL Host Program

- Pure software written in standard ‘C’
- Communicates with the accelerator device via a set of library routines
  - Abstracts away host processor to hardware accelerator communication

```c
main() {
    read_data_from_file( ... );
    maninpulate_data( ... );
    clEnqueueWriteBuffer( ... );
    clEnqueueTask( ..., my_kernel, ...);
    clEnqueueReadBuffer( ... );
    display_result_to_user( ... );
}
```
OpenCL Kernels

- **Data-parallel function**
  - Defines many parallel threads of execution
  - Specifies each thread by a "get_global_id" identifier
  - Contains keyword extensions to specify parallelism and memory hierarchy

- **Executed by compute object**
  - CPU
  - GPU
  - Accelerator
  - FPGA

```c
__kernel void sum(__global const float *a, __global const float *b, __global float *answer)
{
    int xid = get_global_id(0);
    answer[xid] = a[xid] + b[xid];
}
```

```c
float *a = 0 1 2 3 4 5 6 7
float *b = 7 6 5 4 3 2 1 0
float *answer = 7 7 7 7 7 7 7 7
```
FPGA Architecture for OpenCL

Kernel System
OpenCL Memory Hierarchy

- **Hierarchical Memory Model**
  - **Constant** → used to hold lookup table data that is unchanging during the run of a program
  - **Local Memory** → Scratchpad space where threads can share information / intermediate results
  - **Global** → Off chip DDR memory
What is the Altera SDK for OpenCL?

The Altera SDK for OpenCL provides the following logical components:

- **Compiler** - Translates your OpenCL C device code into a hardware configuration file that can be loaded onto an Altera FPGA.

- **Loader** - Loads the FPGA hardware configuration file produced by the compiler onto the FPGA board.

- **Host runtime environment** - Provides the OpenCL host platform API and runtime API for your OpenCL host application. The host runtime environment consists of the following libraries:
  - static library that provides OpenCL host APIs
  - dynamic library that provides the low-level interface to the FPGA via the PCIe bus

- **Design examples**
  - FFT, vector add, matrix mult, Sobel filter, JPEG decoder, Black-Scholes…
  - [http://www.altera.com/support/examples/opencl/opencl.html](http://www.altera.com/support/examples/opencl/opencl.html)
```c
__kernel void sum(__global const float *a,
                __global const float *b,
                __global float *answer)
{
    int xid = get_global_id(0);
    answer[xid] = a[xid] + b[xid];
}
```
Vector Add Example

First Cycle

Second Cycle

Third Cycle
channels int DataChannel;

kernel producer(...) {
    write_channel_altera(DataChannel, value);
}

kernel consumer(...) {
    value = read_channel_altera(DataChannel);
}
Benefits of OpenCL on Altera FPGAs
Benefits of OpenCL on FPGAs

- **Higher performance solution**
  - Increase your performance by offloading performance-intensive functions from the host processor in an FPGA.

- **Power-efficient solution**
  - Achieve significantly lower power with high performance compared to other hardware alternatives. With the FPGA’s fine-grain architecture, the Altera SDK for OpenCL generates only the logic you need to deliver with as low as 1/5 of the power of hardware alternatives.

- **Programmer-friendly solution to target FPGAs**
  - As a software programmer, you can now target FPGAs using OpenCL without learning a hardware description language (HDL).
Higher Performance: AES Encryption

- **Encryption/decryption**
  - 256bit key
  - Counter (CTR) method

- **Advantage FPGA**
  - Integer arithmetic
  - Coarse grain bit operations
  - Complex decision making

- **Results**

<table>
<thead>
<tr>
<th>Platform</th>
<th>Power (W)</th>
<th>Performance (GB/s)</th>
<th>Efficiency (GB/s/W)</th>
</tr>
</thead>
<tbody>
<tr>
<td>E5503 Xeon Processor (single core)</td>
<td>est 80</td>
<td>0.01</td>
<td>1.25e-4</td>
</tr>
<tr>
<td>AMD Radeon HD 7970</td>
<td>est 100</td>
<td>0.33</td>
<td>3.30e-3</td>
</tr>
<tr>
<td>PCie385 A7 Accelerator</td>
<td>25</td>
<td>5.20</td>
<td>2.08e-1</td>
</tr>
</tbody>
</table>

© 2013 Altera Corporation—Public
Higher Performance/Watt: Multi-Asset Barrier Option Pricing

- **Monte-Carlo simulation**
  - No closed form solution possible
  - High quality random number generator required
  - Billions of simulations required

- **Used GPU vendors example code**

- **Advantage FPGA**
  - Complex Control Flow

- **Optimizations**
  - Channels, loop pipelining

- **Results**

<table>
<thead>
<tr>
<th>Platform</th>
<th>Power (W)</th>
<th>Performance (Bsims/s)</th>
<th>Efficiency (Msims/s/W)</th>
</tr>
</thead>
<tbody>
<tr>
<td>W3690 Xeon Processor</td>
<td>130</td>
<td>.032</td>
<td>0.0025</td>
</tr>
<tr>
<td>nVidia Kepler20</td>
<td>212</td>
<td>10.1</td>
<td>48</td>
</tr>
<tr>
<td>Bittware S5-PCIe-HQ</td>
<td>45</td>
<td>12.0</td>
<td>266</td>
</tr>
</tbody>
</table>
Higher Performance/Watt: Document Filtering

- **Unstructured data analytics**
  - Bloom Filter

- **Advantage FPGA**
  - Integer Arithmetic
  - Flexible Memory Configuration

---

### Results

<table>
<thead>
<tr>
<th>Platform</th>
<th>Power (W)</th>
<th>Performance (MTs)</th>
<th>Efficiency (MTs/W)</th>
</tr>
</thead>
<tbody>
<tr>
<td>W3690 Xeon Processor</td>
<td>130</td>
<td>2070</td>
<td>15.92</td>
</tr>
<tr>
<td>nVidia Tesla C2075</td>
<td>215</td>
<td>3240</td>
<td>15.07</td>
</tr>
<tr>
<td>PCIe385 A7 Accelerator</td>
<td>25</td>
<td>3602</td>
<td>144.08</td>
</tr>
</tbody>
</table>

© 2013 Altera Corporation—Public
Faster Time-to-Market – HDR Video Processing

- goHDR developed a new video camera requiring intense video processing
  - Proprietary video codec algorithms
  - Captures frames with different exposure levels → clearer picture

- OpenCL enabled code implementation in an FPGA in less than 1 week
  - Port C-code to OpenCL to FPGA implementation
  - C to HDL typically requires 3–6 months

Save Months of Development
Next Steps
How Do I Get Started?

- **Buy a board from one of Altera’s Preferred Board Partners**
  - [www.altera.com/OpenCL_Boards](http://www.altera.com/OpenCL_Boards)
  - Includes Quartus II Development Kit Edition Software (one year license)
  - Includes an Altera SDK for OpenCL License (one year, perpetual license)
  - Or – make your own (custom boards support feature in Altera in OpenCL SDK)

- **Download the Altera SDK for OpenCL**
  - Purchase a 1 year perpetual license for $995
  - Requires Quartus II v 13.0+ Subscription Edition

- **Operating System Support**
  - Microsoft 64-bit Windows 7
  - Red Hat Enterprise 64-bit Linux (RHEL) 5.6

- **Memory requirements**
  - Computer equipped with at least 16 GB RAM
## OpenCL Training Resources

<table>
<thead>
<tr>
<th>Instructor-led training</th>
<th>OpenCL for Altera FPGAs Training (four days) by Acceleware – available now! Learn how to write and optimize OpenCL applications for Altera FPGAs. The training includes innovative hands-on exercises and a series of progressive lectures. Small class sizes maximize learning and ensure a personal educational experience.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Local workshop or training class</td>
<td>Parallel Computing with OpenCL Workshop (one day) Get an overview of the OpenCL standard and the OpenCL for FPGA design flow. Workshop includes hands-on exercises.</td>
</tr>
<tr>
<td>Free online classes</td>
<td>Introduction to Parallel Computing with OpenCL (30 minutes) Get an overview of the OpenCL standard and the advantages of using Altera’s OpenCL solution. Writing OpenCL Programs for Altera FPGAs (1 hour) Understand the basics of the OpenCL standard and learn to write simple programs. Running OpenCL on Altera FPGAs (30 minutes) Get to know the Altera SDK for OpenCL and learn to compile and run OpenCL programs on Altera FPGAs. Basics of Programmable Logic (1.5 hours) Get a basic introduction to programmable logic devices, focusing on FPGAs.</td>
</tr>
</tbody>
</table>
Additional Resources

- **Documentation**
  - *The OpenCL Specification Version 1.0* (PDF)
  - *Altera SDK for OpenCL Getting Started Guide* (PDF)
  - *Altera SDK for OpenCL CVSoC Getting Started Guide* (PDF)
  - *Altera SDK for OpenCL Programming Guide* (PDF)
  - *Altera SDK for OpenCL Best Practices Guide* (PDF)
  - All documentation is available on [www.altera.com/opencl](http://www.altera.com/opencl)

- **Forum**
  - Search for OpenCL on the left navigation tab
Thank You