Comparative Analysis of OpenCL vs. HDL with Image-Processing Kernels on Stratix-V FPGA

Kenneth Hill, Stefan Craciun, Alan George, Herman Lam
Center for High-Performance Reconfigurable Computing
University of Florida
Objective

- Compare two methodologies for FPGA development:
  - Traditional VHDL/Verilog
  - OpenCL
- Metrics:
  - Performance
  - Resource utilization
  - Programmer productivity
VHDL/Verilog Drawbacks

- Steep learning curves
- Large and complex codes
- Long compilation times
- Platform-specific code
OpenCL Potential Benefits

- Higher productivity
- Code portability
- Easier for software developers
Methodology

- Define accelerator for image processing
- Implement accelerator’s architecture using both VHDL and OpenCL
  - Measure time to write/debug the code
- Collect performance and resource utilization on Stratix V FPGA
Workloads

Edge detectors
Sobel
Canny

Feature extraction
SURF interest-point locations

- Apply 2D convolution on a sliding window
- Further details in the paper
Accelerator’s Architecture

- Accelerators’ engine:
  - Sliding windows
  - 2D convolution unit
- The same architecture will be implemented using VHDL and OpenCL
VHDL Design Flow

- Error-prone
- Time consuming
- Many feedback loops
OpenCL Design Flow

- More similar to software design cycle
- Fast modify-compile-simulate cycle
- Expensive synthesis delayed until the very end
OpenCL Optimization

### Sliding Window

```c
#pragma unroll
for (int i = MAX_VAL; i > 0; --i)
{
    rows[i] = rows[i - 1];
}
rows[0] = frame_in;
```

### 2D Convolution Engine

```c
int x = 0;
int y = 0;
#pragma unroll
for (int i = 0; i < 3; ++i)
{
    #pragma unroll
    for (int j = 0; j < 3; ++j)
    {
        x += rows(i, j) * Gx[i][j];
        y += rows(i, j) * Gy[i][j];
    }
}
```

- Loops automatically unrolled
  - All shifts and products are done in parallel
Experimental Setup

- Intel-Altera Stratix V FPGA (28nm)
  - Gidel
  - Bittware
  - Nallatech
- VGA (480p) and HD (720p) images
Performance Comparison

- 2%-10% slowdown in OpenCL
Resource Usage Comparison

~3x more LUTs

<table>
<thead>
<tr>
<th></th>
<th>VHDL</th>
<th>OpenCL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sobel</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Canny</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SURF</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Resource Usage Comparison

Up to 4.8x more DSPs
Resource Usage Comparison

Up to 2.5x more memory
Productivity Comparison

- OpenCL provides 6x improvement in productivity
- **OpenCL** implementation: *1 month*
  - Most of the time spent writing/debugging code (GDB)
- **VHDL** implementation: *6 months*
  - Most of the time spent in functional verification
Conclusions of the Paper

• For image processing acceleration on FPGAs, OpenCL provides:
  - Code portability
  - 6x higher productivity than VHDL
  - Between 2.5x and 4.8x more resource usage
  - Between 2% and 10% lower performance
Weaknesses

- Limited results, hard to make general claims about OpenCL vs VHDL
  - Just one app domain (image processing)
  - Just one FPGA (Stratix V)
- Oversells OpenCL’s code portability
  - Vendor extensions are required to achieve good performance (Altera channels)
  - Code does not work on Xilinx FPGAs
How is the Paper Related to PD?

- Paper implements accelerator for image processing
  - PD: accelerators are necessary due to power wall and dark silicon problem
- Paper evaluates accelerator on FPGA platform
  - PD: FPGAs provide a flexible and efficient platform for implementing hardware accelerators
How does it extend PD?

- Extensive discussion of High Level Synthesis (HLS)
  - We did not talk that much about HLS in PD
- The paper presents OpenCL as a promising tool
  - Programmer friendly entry point to FPGA platform