Intel® VTune Profiler and Intel® Advisor Overview

Kevin O’Leary
Agenda

1. Intel® VTune Profiler overview
2. Intel® VTune Profiler for GPU
3. Offload Modeling with Intel® Advisor
4. GPU Roofline with Intel® Advisor
Using Intel® Analyzers to increase performance

Use Offload Advisor to find kernels to offload

Optimize your kernels
With Advisor and VTune

Intel® Advisor – Offload Advisor

Choose

- Kernel Style: Data Parallel C++
- Directive Style: OpenMP

Intel® Advisor – GPU Roofline and Flow Graph analyzer

Optimized Code

Existing CUDA code
New Code
Existing C++
Existing Fortran or C
Existing OpenCL™ Applications

Intel® DPC++ Compatibility Tool
Intel Analysis Tools for GPU Compute

**Intel® Advisor**

- **Offload Advisor**
  - Identify high-impact opportunities to offload
  - Detect bottlenecks and key bounding factors
  - Get your code ready even before you have the hardware by modeling performance, headroom and bottlenecks

- **Roofline Analysis**
  - See performance headroom against hardware limitations
  - Determine performance optimization strategy by identifying bottlenecks and which optimizations will payoff the most
  - Visualize optimization progress

- **Flow Graph Analyzer**
  - Visualize your CPU/GPU code and get recommendations for the CPU device

**Intel® VTune™ Profiler**

- **Offload Performance Tuning**
  - Explore code execution on various CPU and GPU cores on your platform
  - Correlate CPU and GPU activity
  - Identify whether your application is GPU or CPU bound

- **HPC Performance Characterization**
  - Identify whether the OpenMP application offloads work to GPU effectively

- **GPU Compute/Media Hotspots**
  - Analyze the most time-consuming GPU kernels, characterize GPU usage based on GPU hardware metrics
  - GPU code performance at the source-line level and kernel assembly level
Intel® VTune™ Profiler
Intel® VTune™ Profiler GUI: quick overview

- GUI provides 3 panes to configure the analysis:
  - **WHERE** is used to specify an analysis system
  - **WHAT** is used to specify an analysis target
  - **HOW** is used to select an analysis type

VTune Profiler documentation: [WHERE: Analysis system](#), [WHAT: Analysis Target](#) and [HOW: Analysis Types](#)
VTune CLI: quick overview

- CLI has its own help with several levels:
  - `vtune -help`
  - `vtune -help collect`
  - `vtune -help collect gpu-offload`
- Run collection:
  - `vtune -collect <analysis_type> <target>`
- Generate a report:
  - `vtune -report <report_name> -r <result_dir>`

VTune Profiler documentation: [Command Line Interface](#)
GPU offload

- Helps to identify whether the application offloads work to GPU effectively.
- Can be used to profile OpenCL, Level0 and Intel Media SDK based applications or DPC++ and OpenMP applications that offload work on Intel GPU.

VTune Profiler documentation: GPU Offload Analysis
Optimize your GPU usage using Intel® VTune Profiler

GPU offload

This analysis enables you to:

- Identify how effectively your application uses DPC++ or OpenCL kernels.
- Explore GPU usage and analyze a software queue for GPU engines at each moment of time.

**Elapased Time**: 2.017s

**GPU Usage**: 47.8%

Use this section to understand whether the GPU was utilized properly and which of the engines were utilized. Identify the amount of gaps in the GPU utilization that potentially could be loaded with some work. This metric is calculated for the engines that had at least one piece of work scheduled to them.

<table>
<thead>
<tr>
<th>GPU Engine / Packet Type</th>
<th>GPU Time</th>
<th>GPU Utilization</th>
</tr>
</thead>
<tbody>
<tr>
<td>Render and GPPGU</td>
<td>0.964s</td>
<td>47.8%</td>
</tr>
<tr>
<td>Unknown</td>
<td>0.964s</td>
<td>47.8%</td>
</tr>
</tbody>
</table>

*NAI is applied to non-summable metrics.

- **Packet Queue Depth Histogram**
- **Packet Duration Histogram**

**Hottest GPU Computing Tasks**

This section lists the most active computing tasks running on the GPU, sorted by the Total Time. Focus on the computing tasks flagged as performance-critical.

<table>
<thead>
<tr>
<th>Computing Task</th>
<th>Total Time</th>
<th>Total Compute Time</th>
<th>Total Transfer Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Matrix&lt;float&gt;</td>
<td>3.980s</td>
<td>0.961s</td>
<td>3.019s</td>
</tr>
<tr>
<td>clEnqueueReadBufferRect</td>
<td>0.000s</td>
<td>0.000s</td>
<td>0.000s</td>
</tr>
</tbody>
</table>

*NAI is applied to non-summable metrics.
Optimize your GPU usage using Intel® VTune™ Profiler

GPU offload

Use the GPU offload features Intel® VTune™ Profiler to see how effectively we are using our GPU.

VTune Profiler shows a synchronized time line between the CPU and GPU. GPU offload does indicate that our GPU execution units are stalling as indicated by the dark read bar in our timeline.
HPC Performance Characterization

- Helps to identify whether the OpenMP application offloads work to GPU effectively.

- The main difference with GPU Offload analysis is that the data is collected through OMPT interface.

VTune Profiler documentation: HPC Performance Characterization Analysis
GPU Compute/Media Hotspots

- Allows to analyze the most time-consuming GPU kernels, characterize GPU usage based on GPU hardware metrics, identify performance issues caused by memory latency or inefficient kernel algorithms, and analyze GPU instruction frequency per certain instruction types.

VTune Profiler documentation: [GPU Compute/Media Hotspots Analysis](#)
Run VTune Profiler GPU Hotspots to try to identify the source of our low GPU utilization and stalls. Click on the graphics tab in GPU Hotspots and you can see a high-level diagram of your architecture.
Intel® VTune™ Profiler for Intel GPUs – Timelines for Correlation

Identify too much or too little kernel activity

Correlate GPU activity with kernels and threads
Intel® VTune™ Profiler for DPC++ Code

Performance metrics at the DPC++ statement level

GPU assembly available for compute kernels
Intel® Advisor
Rich Set of Capabilities for High Performance Code Design
Intel® Advisor

**Offload Advisor**
Design offload strategy and model performance on GPU.

**Roofline Analysis**
Optimize your application for memory and compute.

**Vectorization Optimization**
Enable more vector parallelism and improve its efficiency.

**Thread Prototyping**
Model, tune, and test multiple threading designs.

**Build Heterogeneous Algorithms**
Create and analyze data flow and dependency computation graphs.
Offload Modeling
With Intel® Advisor
Intel® Advisor - Offload Modeling

- Run on CPU or GPU – Predict for GPU
- Helps to define which sections of the code should run on given accelerator
- Provides performance projection on accelerators

Offloading is LLC BW Bound

Not Profitable for offloading

Recommended for offloading

No data available
Intel® Advisor - Offload Modeling
Find code that can be profitably offloaded

Loop on GPU is 3.3x faster than on CPU

The whole app is 3.3x faster

Loop takes 100% of the whole app execution time
In-Depth Analysis of Top Offload Regions

Provides a detailed description of modeling for each loop

- Timings (total time, time on the accelerator, speedup)
- Offload metrics (offload tax data transfers)
- Memory traffic (DRAM, L3, L2, L1), trip count

Highlight which part of the code should be run on the accelerator

Loop at multiply.c:53 is recommended for offloading
- LLC BW bound
- Estimated to run on GPU in 6.997s
- Transfers 101MB of data
In-Depth Analysis of Top Offload Regions

Loop metrics are matched with Source and Call Tree

<table>
<thead>
<tr>
<th>Source</th>
<th>Top-Down</th>
<th>Recommendations</th>
<th>Measured Time</th>
<th>Basic Estimated Metrics</th>
<th>Estimated Bounded By</th>
<th>Estimated Data Transfer</th>
</tr>
</thead>
<tbody>
<tr>
<td>▼ Total</td>
<td></td>
<td></td>
<td>23.28s</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>▼ func@0x4b2e57/59</td>
<td></td>
<td></td>
<td>23.27s</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>▼ func@0x4b2e8775</td>
<td></td>
<td></td>
<td>23.27s</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>▼ BaseThreadInitThunk</td>
<td></td>
<td></td>
<td>23.27s</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>▼ ThreadFunction</td>
<td></td>
<td></td>
<td>23.27s</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>▼ multiply1</td>
<td></td>
<td></td>
<td>23.27s</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>▼ [loop in multiply1 at multiply.c:53]</td>
<td></td>
<td></td>
<td>23.27s</td>
<td>3.326x</td>
<td>6...</td>
<td>[Offloaded]</td>
</tr>
<tr>
<td>▼ _socl_common_main_seh</td>
<td></td>
<td></td>
<td>98.5ms</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Loop metrics include time, speed-up, and offload summary. The table shows various regions, their metrics, and estimated data transfer with reuse.
GPU Roofline
With Intel® Advisor
Where are the bottlenecks?

How much performance is being left on the table?

Which bottlenecks can be addressed, and which should be addressed?

What's the most likely cause?

What are the next steps?
Focus optimization effort where it makes the most difference

- Large, red loops have the most impact
- Loops far from the upper roofs have more room to improve

Additional roofs can be plotted for specific computation types or cache levels
Find Effective Optimization Strategies
Intel® Advisor - GPU Roofline

- GPU Roofline Performance
- Highlights poor performing loops
- Shows likely causes of bottlenecks
- Suggests next optimization steps
- Shows performance ‘headroom’ for each loop
- Which can be improved
- Which are worth improving
Find Effective Optimization Strategies
Intel® Advisor - GPU Roofline

Configure levels to display

Shows performance headroom for each loop

Likely bottlenecks

Suggests optimization next steps
Customize to Display Only Desired Roofs

Click on the top-right corner and remove unused roofs.
Switch between report tabs
Customizable GPU Roofline chart
GPU performance of compute tasks

Summary metrics
Create a snapshot

New Details, GPU Source, and GPU Assembly info
Summary

- You can use the Advisor and VTune GUI & CLI to run the collection and to generate the reports.

- Advisor and VTune both provide several analysis types to profile GPU workload.
Legal Disclaimer & Optimization Notice

- Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. For more complete information visit www.intel.com/benchmarks.

- INFORMATION IN THIS DOCUMENT IS PROVIDED “AS IS”. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO THIS INFORMATION INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

- Copyright © 2018, Intel Corporation. All rights reserved. Intel, Pentium, Xeon, Xeon Phi, Core, VTune, Cilk, and the Intel logo are trademarks of Intel Corporation in the U.S. and other countries.

**Optimization Notice**

Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice.

Notice revision #20110804