Intel® oneAPI VTune™ Profiler 2021.1.1 Gold

Elapsed Time: 0.051s

> Application execution time is too short. Metrics data may be unreliable. Consider reducing the sampling interval or increasing your application execution time.

Clockticks:	186,840,000
Instructions Retired:	170,820,000
CPI Rate:	1.094

> The CPI may be too high. This could be caused by issues such as memory stalls, instruction starvation, branch misprediction or long latency instructions. Explore the other hardware-related metrics to identify what is causing high CPI.

MUX Reliability:	0.824
Retiring:	33.1% of Pipeline Slots
Light Operations:	27.4% of Pipeline Slots
FP Arithmetic:	0.0% of uOps
FP x87:	0.0% of uOps
FP Scalar:	0.0% of uOps
FP Vector:	0.0% of uOps
Other:	100.0% of uOps
Heavy Operations:	5.7% of Pipeline Slots
Microcode Sequencer:	3.8% of Pipeline Slots
Assists:	0.0% of Pipeline Slots
Front-End Bound:	26.8% of Pipeline Slots

> Issue: A significant portion of Pipeline Slots is remaining empty due to issues in the Front-End.
>
> Tips: Make sure the code working size is not too large, the code layout does not require too many memory accesses per cycle to get enough instructions for filling four pipeline slots, or check for microcode assists.

Front-End Latency: 14.3% of Pipeline Slots
- **ICache Misses:** 2.9% of Clockticks
- **ITLB Overhead:** 1.2% of Clockticks
- **Branch Resteers:** 5.5% of Clockticks
  - **Mispredicts Resteers:** 0.0% of Clockticks
  - **Clears Resteers:** 2.9% of Clockticks
  - **Unknown Branches:** 2.6% of Clockticks
- **DSB Switches:** 0.0% of Clockticks
- **Length Changing Prefixes:** 0.0% of Clockticks
- **MS Switches:** 0.0% of Clockticks

Front-End Bandwidth: 12.5% of Pipeline Slots

> This metric represents a fraction of slots during which CPU was stalled due to front-end bandwidth issues, such as inefficiencies in the instruction decoders or code restrictions for caching in the DSB (decoded uOps cache). In such cases, the front-end typically delivers a non-optimal amount of uOps to the back-end.

Front-End Bandwidth MITE: 25.0% of Clockticks

> This metric represents a fraction of cycles during which CPU was stalled due to the MITE fetch pipeline issues, such as inefficiencies in the instruction decoders.

Front-End Bandwidth DSB: 0.0% of Clockticks

(Info) DSB Coverage: 28.6%

> Issue: A significant fraction of uOps was not delivered by the DSB (known as Decoded ICache or uOp Cache). This may happen if a hot code region is too large to fit into the DSB.
>
> Tips: Consider changing the code layout (for example, via profile-guided optimization) to help your hot regions fit into the DSB.
>
> See the "Optimization for Decoded ICache" section in the Intel 64 and IA-32 Architectures Optimization Reference Manual.

Bad Speculation: 5.4% of Pipeline Slots
- **Branch Mispredict:** 0.0% of Pipeline Slots
- **Machine Clears:** 5.4% of Pipeline Slots

Back-End Bound: 34.7% of Pipeline Slots

A significant portion of pipeline slots are remaining empty. When operations take too long in the back-end, they introduce bubbles in the pipeline that ultimately cause fewer pipeline slots containing useful work to be retired per cycle than the machine is capable to support. This opportunity cost results in slower execution. Long-latency operations like divides and memory operations can cause this, as can too many operations being directed to a single execution port (for example, more multiply operations arriving in the back-end per cycle than the execution unit can support).

Memory Bound:	13.7% of Pipeline Slots
L1 Bound:	8.7% of Clockticks
DTLB Overhead:	0.7% of Clockticks
Load STLB Hit:	0.0% of Clockticks
Load STLB Miss:	0.7% of Clockticks
Loads Blocked by Store Forwarding:	0.0% of Clockticks
Lock Latency:	0.0% of Clockticks
Split Loads:	0.0% of Clockticks
4K Aliasing:	0.0% of Clockticks
FB Full:	0.0% of Clockticks
L2 Bound:	2.9% of Clockticks
L3 Bound:	2.9% of Clockticks
Contested Accesses:	0.0% of Clockticks
Data Sharing:	0.0% of Clockticks
L3 Latency:	0.0% of Clockticks
SQ Full:	0.0% of Clockticks
DRAM Bound:	0.0% of Clockticks
Memory Bandwidth:	5.8% of Clockticks
Memory Latency:	14.5% of Clockticks
Store Bound:	0.0% of Clockticks
Store Latency:	11.6% of Clockticks
False Sharing:	0.0% of Clockticks
Split Stores:	0.0% of Clockticks
DTLB Store Overhead:	0.7% of Clockticks
Store STLB Hit:	0.0% of Clockticks
Store STLB Hit:	0.7% of Clockticks
Core Bound:	21.0% of Pipeline Slots

This metric represents how much Core non-memory issues were of a bottleneck. Shortage in hardware compute resources, or dependencies software's instructions are both categorized under Core Bound. Hence it may indicate the machine ran out of an OOO resources, certain execution units are overloaded or dependencies in program's data- or instruction- flow are limiting the performance (e.g. FP-chained long-latency arithmetic operations).

Divider: 0.0% of Clockticks
Port Utilization: 22.1% of Clockticks

> Issue: A significant fraction of cycles was stalled due to Core non-divider-related issues.
>
> Tips: Use vectorization to reduce pressure on the execution ports as multiple elements are calculated with same uOp.

Cycles of 0 Ports Utilized: 23.3% of Clockticks

> CPU executed no uOps on any execution port during a significant fraction of cycles. Long-latency instructions like divides may contribute to this issue. Check the Assembly view and Appendix C in the Optimization Guide to identify instructions with 5 or more cycles latency.

Serializing Operations: 20.2% of Clockticks

> A significant fraction of cycles was spent handling serializing operations. Instructions like CPUID, WRMSR, or LFENCE serialize the out-of-order execution, which may limit performance.

Mixing Vectors: 0.0% of uOps
Cycles of 1 Port Utilized: 10.7% of Clockticks
Cycles of 2 Ports Utilized: 8.9% of Clockticks
Cycles of 3+ Ports Utilized: 14.3% of Clockticks
ALU Operation Utilization: 12.5% of Clockticks
Port 0: 7.2% of Clockticks
Port 1: 3.6% of Clockticks
Port 5: 10.7% of Clockticks
Port 6: 28.6% of Clockticks
Load Operation Utilization: 5.4% of Clockticks
Port 2: 7.2% of Clockticks
Port 3: 10.7% of Clockticks
Store Operation Utilization: 7.2% of Clockticks
Port 4: 7.2% of Clockticks
Port 7: 0.0% of Clockticks
Vector Capacity Usage (FPU): 0.0%
Average CPU Frequency: 1.282 GHz
Total Thread Count: 9
Paused Time: 0s

Effective Physical Core Utilization: 57.1% (2.285 out of 4)

> The metric value is low, which may signal a poor physical CPU cores utilization caused by:
> - load imbalance
> - threading runtime overhead
> - contended synchronization
> - thread/process underutilization
> - incorrect affinity that utilizes logical cores instead of physical cores
>
> Explore sub-metrics to estimate the efficiency of MPI and OpenMP parallelism or run the Locks and Waits analysis to identify parallel bottlenecks for other parallel runtimes.

Effective Logical Core Utilization: 35.4% (2.829 out of 8)

> The metric value is low, which may signal a poor logical CPU cores utilization. Consider improving physical core utilization as the first step and then look at opportunities to utilize logical cores, which in some cases can improve processor throughput and overall performance of multi-threaded applications.

Collection and Platform Info:

Application Command Line: ./codecs/HHI-VVC/decoder/vvdecapp "-b" "./bin/HHI-VVC/randomaccess_fast.cfg/CLASS_C/RaceHorses_416x240_30_QP_22_HHI-VVC.bin"

User Name: root

Operating System: 5.4.0-72-generic DISTRIB_ID=Ubuntu DISTRIB_RELEASE=18.04 DISTRIB_CODENAME=bionic DISTRIB_DESCRIPTION="Ubuntu 18.04.5 LTS"

Computer Name: eimon

Result Size: 14.9 MB

Collection start time: 22:28:00 18/04/2021 UTC

Collection stop time: 22:28:00 18/04/2021 UTC

Collector Type: Event-based sampling driver

CPU:

Name: Intel(R) Processor code named Kabylake ULX

Frequency: 1.992 GHz

Logical CPU Count: 8

Cache Allocation Technology:

Level 2 capability: not detected

Level 3 capability: not detected

Intel® oneAPI VTune™ Profiler 2021.1.1 Gold

Elapsed Time: 0.032s

> Application execution time is too short. Metrics data may be unreliable. Consider reducing the sampling interval or increasing your application execution time.

Clockticks: 182,700,000
Instructions Retired: 171,900,000
CPI Rate: 1.063

> The CPI may be too high. This could be caused by issues such as memory stalls, instruction starvation, branch misprediction or long latency instructions. Explore the other hardware-related metrics to identify what is causing high CPI.

MUX Reliability: 0.967
Retiring: 35.5% of Pipeline Slots
- **Light Operations:** 31.4% of Pipeline Slots
  - **FP Arithmetic:** 0.0% of uOps
    - **FP x87:** 0.0% of uOps
    - **FP Scalar:** 0.0% of uOps
    - **FP Vector:** 0.0% of uOps
  - **Other:** 100.0% of uOps
- **Heavy Operations:** 4.1% of Pipeline Slots
  - **Microcode Sequencer:** 2.7% of Pipeline Slots
    - **Assists:** 0.0% of Pipeline Slots

Front-End Bound: 26.6% of Pipeline Slots

> Issue: A significant portion of Pipeline Slots is remaining empty due to issues in the Front-End.
>
> Tips: Make sure the code working size is not too large, the code layout does not require too many memory accesses per cycle to get enough instructions for filling four pipeline slots, or check for microcode assists.

Front-End Latency: 23.6% of Pipeline Slots

> This metric represents a fraction of slots during which CPU was stalled due to front-end latency issues, such as instruction-cache misses, ITLB misses or fetch stalls after a branch misprediction. In such cases, the front-end delivers no uOps.

ICache Misses: 0.0% of Clockticks
ITLB Overhead: 2.1% of Clockticks
Branch Resteers: 0.0% of Clockticks
Mispredicts Resteers: 0.0% of Clockticks
Clears Resteers: 0.0% of Clockticks
Unknown Branches: 0.0% of Clockticks
DSB Switches: 0.0% of Clockticks
Length Changing Prefixes: 0.0% of Clockticks
MS Switches: 0.0% of Clockticks

> Issue: A significant fraction of cycles was stalled due to switches of uOp delivery to the Microcode Sequencer (MS). Commonly used instructions are optimized for delivery by the DSB or MITE pipelines. Certain operations cannot be handled natively by the execution pipeline, and must be performed by microcode (small programs injected into the execution stream). Switching to the MS too often can negatively impact performance. The MS is designated to deliver long uOp flows required by CISC instructions like CPUID, or uncommon conditions like Floating Point Assists when dealing with Denormals. Note that this metric value may be highlighted due to Microcode Sequencer issue.

Front-End Bandwidth: 3.0% of Pipeline Slots
Front-End Bandwidth MITE: 35.5% of Clockticks
Front-End Bandwidth DSB: 0.0% of Clockticks
(Info) DSB Coverage: 32.6%
Bad Speculation: 3.9% of Pipeline Slots
Branch Mispredict: 0.0% of Pipeline Slots
Machine Clears: 3.9% of Pipeline Slots
Back-End Bound: 34.0% of Pipeline Slots

> A significant portion of pipeline slots are remaining empty. When operations take too long in the back-end, they introduce bubbles in the pipeline that ultimately cause fewer pipeline slots containing useful work to be retired per cycle than the machine is capable to support. This opportunity cost results in slower execution. Long-latency operations like divides and memory operations can cause this, as can too many operations being directed to a single execution port (for example, more multiply operations arriving in the back-end per cycle than the execution unit can support).

Memory Bound: 21.0% of Pipeline Slots

The metric value is high. This can indicate that the significant fraction of execution pipeline slots could be stalled due to demand memory load and stores. Use Memory Access analysis to have the metric breakdown by memory hierarchy, memory bandwidth information, correlation by memory objects.

L1 Bound: 11.8% of Clockticks

This metric shows how often machine was stalled without missing the L1 data cache. The L1 cache typically has the shortest latency. However, in certain cases like loads blocked on older stores, a load might suffer a high latency even though it is being satisfied by the L1. Note that this metric value may be highlighted due to DTLB Overhead or Cycles of 1 Port Utilized issues.

DTLB Overhead: 1.3% of Clockticks
Load STLB Hit: 0.0% of Clockticks
Load STLB Miss: 1.3% of Clockticks
Loads Blocked by Store Forwarding: 0.0% of Clockticks
Lock Latency: 0.0% of Clockticks

A significant fraction of CPU cycles spent handling cache misses due to lock operations. Due to the microarchitecture handling of locks, they are classified as L1 Bound regardless of what memory source satisfied them. Note that this metric value may be highlighted due to Store Latency issue.

Split Loads: 0.0% of Clockticks
4K Aliasing: 0.0% of Clockticks
FB Full: 0.0% of Clockticks

This metric does a rough estimation of how often L1D Fill Buffer unavailability limited additional L1D miss memory access requests to proceed. The higher the metric value, the deeper the memory hierarchy level the misses are satisfied from. Often it hints on approaching bandwidth limits (to L2 cache, L3 cache or external memory). Avoid adding software prefetches if indeed memory BW limited.

L2 Bound:
L3 Bound: 5.9% of Clockticks

> This metric shows how often CPU was stalled on L3 cache, or contended with a sibling Core. Avoiding cache misses (L2 misses/L3 hits) improves the latency and increases performance.

Contested Accesses:
Data Sharing:
L3 Latency:
SQ Full: 0.0% of Clockticks
DRAM Bound:
Memory Bandwidth: 0.0% of Clockticks

Memory Latency: 5.9% of Clockticks

Store Bound: 0.0% of Clockticks
Store Latency: 11.8% of Clockticks
False Sharing: 0.0% of Clockticks
Split Stores: 0.0% of Clockticks
DTLB Store Overhead: 1.0% of Clockticks
Store STLB Hit: 0.0% of Clockticks
Store STLB Hit: 1.0% of Clockticks
Core Bound: 13.0% of Pipeline Slots

> This metric represents how much Core non-memory issues were of a bottleneck. Shortage in hardware compute resources, or dependencies software's instructions are both categorized under Core Bound. Hence it may indicate the machine ran out of an OOO resources, certain execution units are overloaded or dependencies in program's data- or instruction- flow are limiting the performance (e.g. FP-chained long-latency arithmetic operations).

Divider: 0.0% of Clockticks
Port Utilization: 11.0% of Clockticks
Cycles of 0 Ports Utilized: 9.9% of Clockticks
Serializing Operations: 14.8% of Clockticks
Mixing Vectors: 0.0% of uOps
Cycles of 1 Port Utilized: 5.9% of Clockticks
Cycles of 2 Ports Utilized: 15.8% of Clockticks
Cycles of 3+ Ports Utilized: 15.8% of Clockticks
ALU Operation Utilization: 10.8% of Clockticks
Port 0: 3.9% of Clockticks
Port 1: 3.9% of Clockticks
Port 5: 7.9% of Clockticks
Port 6: 27.6% of Clockticks
Load Operation Utilization: 2.0% of Clockticks
Port 2: 3.9% of Clockticks
Port 3: 7.9% of Clockticks
Store Operation Utilization: 7.9% of Clockticks
Port 4: 7.9% of Clockticks
Port 7: 0.0% of Clockticks
Vector Capacity Usage (FPU): 0.0%
Average CPU Frequency: 2.008 GHz
Total Thread Count: 9
Paused Time: 0s

Effective Physical Core Utilization: 51.6% (2.064 out of 4)

> The metric value is low, which may signal a poor physical CPU cores utilization caused by:
> - load imbalance
> - threading runtime overhead
> - contended synchronization
> - thread/process underutilization
> - incorrect affinity that utilizes logical cores instead of physical cores
>
> Explore sub-metrics to estimate the efficiency of MPI and OpenMP parallelism or run the Locks and Waits analysis to identify parallel bottlenecks for other parallel runtimes.

Effective Logical Core Utilization: 34.4% (2.752 out of 8)

> The metric value is low, which may signal a poor logical CPU cores utilization. Consider improving physical core utilization as the first step and then look at opportunities to utilize logical cores, which in some cases can improve processor throughput and overall performance of multi-threaded applications.

Collection and Platform Info:

Application Command Line: ./codecs/HHI-VVC/decoder/vvdecapp "-b" "./bin/HHI-VVC/randomaccess_fast.cfg/CLASS_C/RaceHorses_416x240_30_QP_22_HHI-VVC.bin"

User Name: root

Operating System: 5.4.0-72-generic DISTRIB_ID=Ubuntu DISTRIB_RELEASE=18.04 DISTRIB_CODENAME=bionic DISTRIB_DESCRIPTION="Ubuntu 18.04.5 LTS"

Computer Name: eimon

Result Size: 14.2 MB

Collection start time: 07:48:25 19/04/2021 UTC

Collection stop time: 07:48:25 19/04/2021 UTC

Collector Type: Event-based sampling driver

CPU:

Name: Intel(R) Processor code named Kabylake ULX

Frequency: 1.992 GHz

Logical CPU Count: 8

Cache Allocation Technology:

Level 2 capability: not detected

Level 3 capability: not detected