Intel® oneAPI VTune™ Profiler 2021.1.1 Gold

Recommendations:

Increase execution time:

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

Hotspots: Start with Hotspots analysis to understand the efficiency of your algorithm.

> Use Hotspots analysis to identify the most time consuming functions. Drill down to see the time spent on every line of code.

Microarchitecture Exploration: There is low microarchitecture usage (3.6%) of available hardware resources. of Pipeline Slots

> Run Microarchitecture Exploration analysis to analyze CPU microarchitecture bottlenecks that can affect application performance.

Memory Access: The Memory Bound metric is high (43.3%). A significant fraction of execution pipeline slots could be stalled due to demand memory load and stores. of Pipeline Slots

> Use Memory Access analysis to measure metrics that can identify memory access issues.

Threading: There is poor utilization of logical CPU cores (56.8%) in your application.

> Use Threading to explore more opportunities to increase parallelism in your application.

Elapsed Time: 0.025s

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

CPU:

IPC: 0.066

> The IPC may be too low. 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 low IPC.

SP GFLOPS: 0.000

DP GFLOPS:	0.000
x87 GFLOPS:	0.001
Average CPU Frequency:	1.863 GHz

GPU:

Time:	0.0% (0s) of Elapsed time

> GPU utilization is low. Consider offloading more work to the GPU to increase overall application performance.

IPC Rate:	1.000

Effective Logical Core Utilization: 56.8% (4.542 out of 8)

> The metric value is low, which may signal a poor utilization of logical CPU cores while the utilization of physical cores is acceptable. Consider using logical cores, which in some cases can improve processor throughput and overall performance of multi-threaded applications.

Effective Physical Core Utilization: 100.0% (4.000 out of 4)

Microarchitecture Usage: 3.6% of Pipeline Slots

> You code efficiency on this platform is too low.
>
> Possible cause: memory stalls, instruction starvation, branch misprediction or long latency instructions.
>
> Next steps: Run Microarchitecture Exploration analysis to identify the cause of the low microarchitecture usage efficiency.

Retiring:	3.6% of Pipeline Slots
Front-End Bound:	10.7% of Pipeline Slots
Back-End Bound:	83.5% 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: 43.3% 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.

Core Bound: 40.2% 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).

Bad Speculation: 2.3% of Pipeline Slots

Memory Bound: 43.3% 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: 16.3% 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.

L2 Bound:	0.4% of Clockticks
L3 Bound:	1.4% of Clockticks
DRAM Bound:	7.7% of Clockticks
DRAM Bandwidth Bound:	0.0% of Elapsed Time
Store Bound:	0.7% of Clockticks

Vectorization:	0.1% of Packed FP Operations

> A significant fraction of floating point arithmetic instructions are scalar. Use Intel Advisor to see possible reasons why the code was not vectorized.

Instruction Mix:	
SP FLOPs:	0.0% of uOps
Packed:	100.0% from SP FP
128-bit:	100.0% from SP FP

> A significant fraction of floating point arithmetic vector instructions is executed with a partial vector load. Make sure you compile the code with the latest instruction set or use Intel Advisor for vectorization help.

256-bit:	0.0% from SP FP
Scalar:	0.0% from SP FP
DP FLOPs:	0.0% of uOps
Packed:	1.6% from DP FP
128-bit:	1.6% from DP FP
256-bit:	0.0% from DP FP
Scalar:	98.4% from DP FP

> A significant fraction of floating point arithmetic instructions are scalar. Use Intel Advisor to see possible reasons why the code was not vectorized.

x87 FLOPs:	0.1% of uOps
Non-FP:	99.9% of uOps
FP Arith/Mem Rd Instr. Ratio:	0.010

> The metric value is low. This can be a result of unaligned access to data for vector operations. Use Intel Advisor to find possible data access inefficiencies for vector operations.

FP Arith/Mem Wr Instr. Ratio: 0.016

> The metric value is low. This can be a result of unaligned access to data for vector operations. Use Intel Advisor to find possible data access inefficiencies for vector operations.

GPU Active Time: 0.0%

> GPU utilization is low. Consider offloading more work to the GPU to increase overall application performance.

Collection and Platform Info:

Application Command Line: ./codecs/hm/decoder/TAppDecoderStatic "-b" "./bin/hm/encoder_lowdelay_main.cfg/CLASS_C/RaceHorses_416x240_30_QP_32_hm.bin"

Operating System: 5.4.0-65-generic DISTRIB_ID=Ubuntu DISTRIB_RELEASE=18.04 DISTRIB_CODENAME=bionic DISTRIB_DESCRIPTION="Ubuntu 18.04.5 LTS"

Computer Name:	eimon
Result Size:	3.7 MB
Collection start time:	09:40:33 10/02/2021 UTC
Collection stop time:	09:40:33 10/02/2021 UTC
Collector Type:	Event-based sampling driver,Event-based counting driver

CPU:

Name:	Intel(R) Processor code named Kabylake ULX
Frequency:	1.992 GHz
Logical CPU Count:	8
Max DRAM Single-Package Bandwidth:	10.000 GB/s

Cache Allocation Technology:

Level 2 capability:	not detected
Level 3 capability:	not detected

GPU:

Name:

Display controller: Intel Corporation Device 22807

Vendor:	Intel Corporation
EU Count:	24
Max EU Thread Count:	7
Max Core Frequency:	1.150 GHz