Intel® oneAPI VTune™ Profiler 2021.1.1 Gold

Elapsed Time: 0.047s

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

Clockticks:	43,020,000
Instructions Retired:	66,960,000
CPI Rate:	0.642
MUX Reliability:	0.996
Retiring:	46.6% of Pipeline Slots
Light Operations:	44.5% 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:	2.2% of Pipeline Slots
Microcode Sequencer:	3.1% of Pipeline Slots
Assists:	0.0% of Pipeline Slots
Front-End Bound:	25.1% 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:	0.0% of Clockticks
ITLB Overhead:	0.0% 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
Front-End Bandwidth:	10.8% 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: 43.0% 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: 14.3% of Clockticks
(Info) DSB Coverage: 33.3%

> 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: 7.2% of Pipeline Slots
Branch Mispredict: 0.0% of Pipeline Slots
Machine Clears: 7.2% of Pipeline Slots
Back-End Bound: 21.1% 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:	8.6% of Pipeline Slots
L1 Bound:	12.6% of Clockticks
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
Split Loads:	0.0% of Clockticks
4K Aliasing:	0.0% of Clockticks
FB Full:	0.0% of Clockticks
L2 Bound:	0.0% of Clockticks
L3 Bound:	0.0% 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:	0.0% of Clockticks
Memory Latency:	12.6% of Clockticks
Store Bound:	0.0% of Clockticks
Store Latency:	0.0% of Clockticks
False Sharing:	0.0% of Clockticks
Split Stores:	0.0% of Clockticks
DTLB Store Overhead:	1.4% of Clockticks
Store STLB Hit:	0.0% of Clockticks
Store STLB Hit:	1.4% of Clockticks
Core Bound:	12.5% 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: 18.4% of Clockticks
Cycles of 0 Ports Utilized: 14.3% of Clockticks
Serializing Operations: 0.0% of Clockticks
Mixing Vectors: 0.0% of uOps
Cycles of 1 Port Utilized: 7.2% of Clockticks
Cycles of 2 Ports Utilized: 7.2% of Clockticks
Cycles of 3+ Ports Utilized: 14.3% of Clockticks
ALU Operation Utilization: 21.5% of Clockticks
Port 0: 14.3% of Clockticks
Port 1: 14.3% of Clockticks
Port 5: 28.7% of Clockticks
Port 6: 28.7% of Clockticks
Load Operation Utilization: 14.3% of Clockticks
Port 2: 28.7% of Clockticks
Port 3: 28.7% of Clockticks
Store Operation Utilization: 28.7% of Clockticks
Port 4: 28.7% of Clockticks
Port 7: 0.0% of Clockticks
Vector Capacity Usage (FPU): 0.0%
Average CPU Frequency: 1.035 GHz
Total Thread Count: 1
Paused Time: 0s

Effective Physical Core Utilization: 19.5% (0.781 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: 11.2% (0.892 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/HM/decoder/TAppDecoderStatic "-b" "./bin/HM/encoder_randomaccess_main.cfg/CLASS_C/RaceHorses_416x240_30_QP_37_HM.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: 12.5 MB

Collection start time: 22:26:58 18/04/2021 UTC

Collection stop time: 22:26:58 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: 0.042s

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

Clockticks:	42,300,000
Instructions Retired:	66,960,000
CPI Rate:	0.632
MUX Reliability:	0.894
Retiring:	49.2% of Pipeline Slots

> A high fraction of pipeline slots was utilized by useful work. While the goal is to make this metric value as big as possible, a high Retiring value for non-vectorized code could prompt you to consider code vectorization. Vectorization enables doing more computations without significantly increasing the number of instructions, thus improving the performance. Note that this metric value may be highlighted due to Microcode Sequencer (MS) issue, so the performance can be improved by avoiding using the MS.

Light Operations:	33.9% 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:	15.3% of Pipeline Slots

> CPU retired heavy-weight operations (instructions that required 2+ uops) in a significant fraction of cycles.

Microcode Sequencer: 2.7% of Pipeline Slots

Assists: 0.0% of Pipeline Slots

Front-End Bound: 16.4% of Pipeline Slots

Front-End Latency: 10.9% of Pipeline Slots

ICache Misses: 12.8% of Clockticks

ITLB Overhead: 1.3% 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

Front-End Bandwidth: 5.5% of Pipeline Slots

Front-End Bandwidth MITE: 32.8% of Clockticks

Front-End Bandwidth DSB: 0.0% of Clockticks

(Info) DSB Coverage: 33.3%

Bad Speculation: 0.0% of Pipeline Slots

Branch Mispredict: 0.0% of Pipeline Slots

Machine Clears: 0.0% of Pipeline Slots

Back-End Bound: 34.3% 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:	13.8% of Pipeline Slots
L1 Bound:	0.0% of Clockticks
DTLB Overhead:	5.1% of Clockticks
Load STLB Hit:	0.0% of Clockticks
Load STLB Miss:	5.1% 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:	0.0% of Clockticks
L3 Bound:	12.8% 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:	0.0% of Clockticks
Memory Latency:	12.8% of Clockticks
Store Bound:	0.0% of Clockticks
Store Latency:	0.0% of Clockticks
False Sharing:	0.0% of Clockticks
Split Stores:	0.0% of Clockticks
DTLB Store Overhead:	0.5% of Clockticks
Store STLB Hit:	0.0% of Clockticks
Store STLB Hit:	0.5% of Clockticks
Core Bound:	20.6% 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: 19.1% of Clockticks
Cycles of 0 Ports Utilized: 10.9% of Clockticks
Serializing Operations: 0.0% of Clockticks
Mixing Vectors: 0.0% of uOps
Cycles of 1 Port Utilized: 10.9% of Clockticks
Cycles of 2 Ports Utilized: 5.5% of Clockticks
Cycles of 3+ Ports Utilized: 16.4% of Clockticks
ALU Operation Utilization: 21.9% of Clockticks
Port 0: 21.9% of Clockticks
Port 1: 21.9% of Clockticks
Port 5: 21.9% of Clockticks
Port 6: 21.9% of Clockticks
Load Operation Utilization: 21.9% of Clockticks
Port 2: 21.9% of Clockticks
Port 3: 21.9% of Clockticks
Store Operation Utilization: 10.9% of Clockticks
Port 4: 10.9% of Clockticks
Port 7: 10.9% of Clockticks
Vector Capacity Usage (FPU): 0.0%
Average CPU Frequency: 1.139 GHz
Total Thread Count: 1
Paused Time: 0s

Effective Physical Core Utilization: 26.1% (1.044 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: 11.2% (0.895 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/HM/decoder/TAppDecoderStatic "-b" "./bin/HM/encoder_randomaccess_main.cfg/CLASS_C/RaceHorses_416x240_30_QP_37_HM.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: 12.3 MB

Collection start time: 07:47:24 19/04/2021 UTC

Collection stop time: 07:47:24 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