Intel® oneAPI VTune™ Profiler 2021.1.1 Gold

Elapsed Time: 0.049s

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

Clockticks:	48,420,000
Instructions Retired:	76,860,000
CPI Rate:	0.630
MUX Reliability:	0.781
Retiring:	44.6% of Pipeline Slots
Light Operations:	39.7% 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:	4.9% of Pipeline Slots
Microcode Sequencer:	2.5% of Pipeline Slots
Assists:	0.0% of Pipeline Slots
Front-End Bound:	19.5% of Pipeline Slots
Front-End Latency:	11.2% of Pipeline Slots
ICache Misses:	11.2% of Clockticks
ITLB Overhead:	2.2% of Clockticks
Branch Resteers:	11.2% of Clockticks
Mispredicts Resteers:	0.0% of Clockticks
Clears Resteers:	11.2% 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:	8.4% of Pipeline Slots
Front-End Bandwidth MITE:	22.3% of Clockticks
Front-End Bandwidth DSB:	11.2% of Clockticks
(Info) DSB Coverage:	33.3%
Bad Speculation:	5.6% of Pipeline Slots
Branch Mispredict:	0.0% of Pipeline Slots
Machine Clears:	5.6% of Pipeline Slots
Back-End Bound:	30.3% 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:	0.0% of Pipeline Slots
L1 Bound:	0.0% of Clockticks
DTLB Overhead:	1.1% of Clockticks
Load STLB Hit:	0.0% of Clockticks
Load STLB Miss:	1.1% 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:	0.6% of Clockticks
FB Full:	0.0% of Clockticks
L2 Bound:	
L3 Bound:	11.2% of Clockticks
Contested Accesses:	
Data Sharing:	
L3 Latency:	
SQ Full:	0.0% of Clockticks
DRAM Bound:	
Memory Bandwidth:	0.0% of Clockticks
Memory Latency:	0.0% 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:	1.1% of Clockticks
Store STLB Hit:	0.0% of Clockticks
Store STLB Hit:	1.1% of Clockticks
Core Bound:	30.3% 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: 16.1% of Clockticks
Cycles of 0 Ports Utilized: 11.2% of Clockticks
Serializing Operations: 11.2% of Clockticks
Mixing Vectors: 0.0% of uOps
Cycles of 1 Port Utilized: 0.0% of Clockticks
Cycles of 2 Ports Utilized: 11.2% of Clockticks
Cycles of 3+ Ports Utilized: 16.7% of Clockticks
ALU Operation Utilization: 16.7% of Clockticks
Port 0: 11.2% of Clockticks
Port 1: 11.2% of Clockticks
Port 5: 22.3% of Clockticks
Port 6: 22.3% of Clockticks
Load Operation Utilization: 16.7% of Clockticks
Port 2: 22.3% of Clockticks
Port 3: 22.3% of Clockticks
Store Operation Utilization: 22.3% of Clockticks
Port 4: 22.3% of Clockticks
Port 7: 11.2% of Clockticks
Vector Capacity Usage (FPU): 0.0%
Average CPU Frequency: 1.112 GHz
Total Thread Count: 1
Paused Time: 0s

Effective Physical Core Utilization: 22.4% (0.897 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% (0.897 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_32_hm.bin"

User Name: root

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: 9.5 MB

Collection start time: 09:54:25 10/02/2021 UTC

Collection stop time: 09:54:25 10/02/2021 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