Intel® oneAPI VTune™ Profiler 2021.1.1 Gold

Elapsed Time: 0.053s

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

Clockticks:	59,040,000
Instructions Retired:	93,780,000
CPI Rate:	0.630
MUX Reliability:	0.915
Retiring:	36.6% 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:	0.0% of Pipeline Slots
Microcode Sequencer:	1.9% of Pipeline Slots
Assists:	0.0% of Pipeline Slots
Front-End Bound:	20.6% 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: 18.3% 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:	0.9% of Clockticks
Branch Resteers:	9.1% of Clockticks

> Issue: A significant fraction of cycles was stalled due to Branch Resteers. Branch Resteers estimate the

> Front-End delay in fetching operations from corrected path, following all sorts of mispredicted branches. For example, branchy code with lots of mispredictions might get categorized as Branch Resteers. Note the value of this node may overlap its siblings.

Mispredicts Resteers: 0.0% of Clockticks
Clears Resteers: 9.1% of Clockticks

> A significant fraction of cycles could be stalled due to Branch Resteers as a result of Machine Clears.

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: 2.3% of Pipeline Slots
Front-End Bandwidth MITE: 36.6% of Clockticks
Front-End Bandwidth DSB: 0.0% of Clockticks
(Info) DSB Coverage: 26.9%
Bad Speculation: 6.9% of Pipeline Slots
Branch Mispredict: 0.0% of Pipeline Slots
Machine Clears: 6.9% of Pipeline Slots
Back-End Bound: 36.0% 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: 15.2% of Pipeline Slots
- **L1 Bound:** 9.1% of Clockticks
  - **DTLB Overhead:** 0.5% of Clockticks
    - **Load STLB Hit:** 0.0% of Clockticks
    - **Load STLB Miss:** 0.5% 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:** 0.5% of Clockticks
  - **FB Full:** 0.0% of Clockticks
- **L2 Bound:** 0.0% of Clockticks
- **L3 Bound:** 9.1% 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:** 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:** 0.0% of Clockticks
    - **Store STLB Hit:** 0.0% of Clockticks
    - **Store STLB Hit:** 0.0% of Clockticks

Core Bound: 20.8% 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: 12.5% of Clockticks
Cycles of 0 Ports Utilized: 13.7% of Clockticks
Serializing Operations: 0.0% of Clockticks
Mixing Vectors: 0.0% of uOps
Cycles of 1 Port Utilized: 0.0% of Clockticks
Cycles of 2 Ports Utilized: 9.1% of Clockticks
Cycles of 3+ Ports Utilized: 22.9% of Clockticks
ALU Operation Utilization: 20.6% of Clockticks
Port 0: 18.3% of Clockticks
Port 1: 18.3% of Clockticks
Port 5: 18.3% of Clockticks
Port 6: 27.4% of Clockticks
Load Operation Utilization: 22.9% of Clockticks
Port 2: 27.4% of Clockticks
Port 3: 27.4% of Clockticks
Store Operation Utilization: 18.3% of Clockticks
Port 4: 18.3% of Clockticks
Port 7: 9.1% of Clockticks
Vector Capacity Usage (FPU): 0.0%
Average CPU Frequency: 1.235 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_lowdelay_main.cfg/CLASS_C/RaceHorses_416x240_30_QP_27_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: 12.6 MB

Collection start time: 22:20:33 18/04/2021 UTC

Collection stop time: 22:20:33 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: 0.039s

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

Clockticks: 58,500,000
Instructions Retired: 93,960,000
CPI Rate: 0.623
MUX Reliability: 0.903
Retiring: 46.2% of Pipeline Slots
Light Operations: 37.8% 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: 8.3% of Pipeline Slots
Microcode Sequencer: 1.7% of Pipeline Slots
Assists: 0.0% of Pipeline Slots
Front-End Bound: 20.8% 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: 9.2% of Pipeline Slots
ICache Misses: 0.0% of Clockticks
ITLB Overhead: 1.8% 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: 11.5% 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: 27.7% 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: 9.2% of Clockticks
(Info) DSB Coverage: 25.0%

> 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: 16.2% of Pipeline Slots

> A significant proportion of pipeline slots containing useful work are being cancelled. This can be caused by mispredicting branches or by machine clears. Note that this metric value may be highlighted due to Branch Resteers issue.

Branch Mispredict: 0.0% of Pipeline Slots
Machine Clears: 16.2% of Pipeline Slots

> Issue: A significant portion of execution time is spent handling machine clears.
>
> Tips: See the "Memory Disambiguation" section in the Intel 64 and IA-32 Architectures Optimization Reference Manual.

Back-End Bound: 16.9% of Pipeline Slots
- **Memory Bound:** 8.6% of Pipeline Slots
  - **L1 Bound:** 9.2% of Clockticks
    - **DTLB Overhead:** 2.8% of Clockticks
      - **Load STLB Hit:** 0.0% of Clockticks
      - **Load STLB Miss:** 2.8% 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:** 9.2% 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:** 18.5% 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:** 1.8% of Clockticks
      - **Store STLB Hit:** 0.0% of Clockticks
      - **Store STLB Hit:** 1.8% of Clockticks
- **Core Bound:** 8.3% of Pipeline Slots
  - **Divider:** 0.0% of Clockticks
  - **Port Utilization:** 17.8% of Clockticks
    - **Cycles of 0 Ports Utilized:** 9.2% of Clockticks
      - **Serializing Operations:** 0.0% of Clockticks
      - **Mixing Vectors:** 0.0% of uOps
    - **Cycles of 1 Port Utilized:** 9.2% of Clockticks
    - **Cycles of 2 Ports Utilized:** 9.2% of Clockticks
    - **Cycles of 3+ Ports Utilized:** 18.5% of Clockticks
      - **ALU Operation Utilization:** 25.4% of Clockticks
        - **Port 0:** 18.5% of Clockticks
        - **Port 1:** 18.5% of Clockticks
        - **Port 5:** 27.7% of Clockticks
        - **Port 6:** 36.9% of Clockticks
      - **Load Operation Utilization:** 13.8% of Clockticks
        - **Port 2:** 18.5% of Clockticks
        - **Port 3:** 27.7% of Clockticks
      - **Store Operation Utilization:** 27.7% of Clockticks
        - **Port 4:** 27.7% of Clockticks
        - **Port 7:** 9.2% of Clockticks
    - **Vector Capacity Usage (FPU):** 0.0%

Average CPU Frequency: 1.669 GHz

Total Thread Count: 1
Paused Time: 0s

Effective Physical Core Utilization: 22.6% (0.904 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: 11.3% (0.904 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_lowdelay_main.cfg/CLASS_C/RaceHorses_416x240_30_QP_27_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: 07:41:32 19/04/2021 UTC

Collection stop time: 07:41:32 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