HPC.NRW

PROFILING WITH INTEL VTUNE

Analysis Types

August 9, 2023 | Dr. Martin Errenst

ANALYSIS TYPES

HPC.NRW

- Get an overview of different analysis types, for example
  - Hotspots analysis
  - Memory access
  - Threading
- Start with generic types
- Select specific types when necessary

SIMPLE EXAMPLE

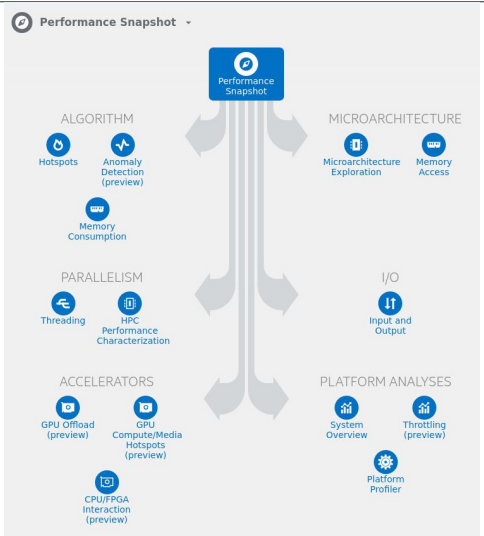
- Same as in introduction
- Three versions:
  1. Single process
  2. OMP SIMD vectorized
  3. OMP multi-threaded + SIMD
- Mostly useful to show features of analysis types

```
// Repetitions for larger workload
// #pragma omp parallel for
for(size_t j = 0; j < repetitions; j++){
   std::vector<float> v3(vsize), v4(vsize);

   // add and multiply random vectors
   //#pragma omp simd
   for(size_t i = 0; i < vsize; i++){
      v3[i] = v1[i] + v2[i];
      v4[i] = v1[i] * v2[i];
   }
}
```

AVAILABLE ANALYSIS TYPES

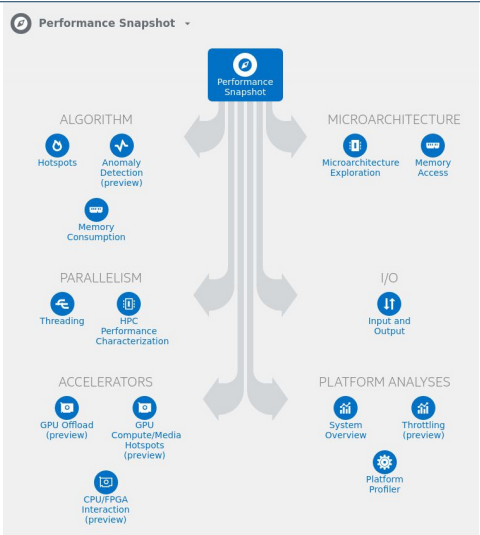
HPC.NRW



AVAILABLE ANALYSIS TYPES

HPC.NRW

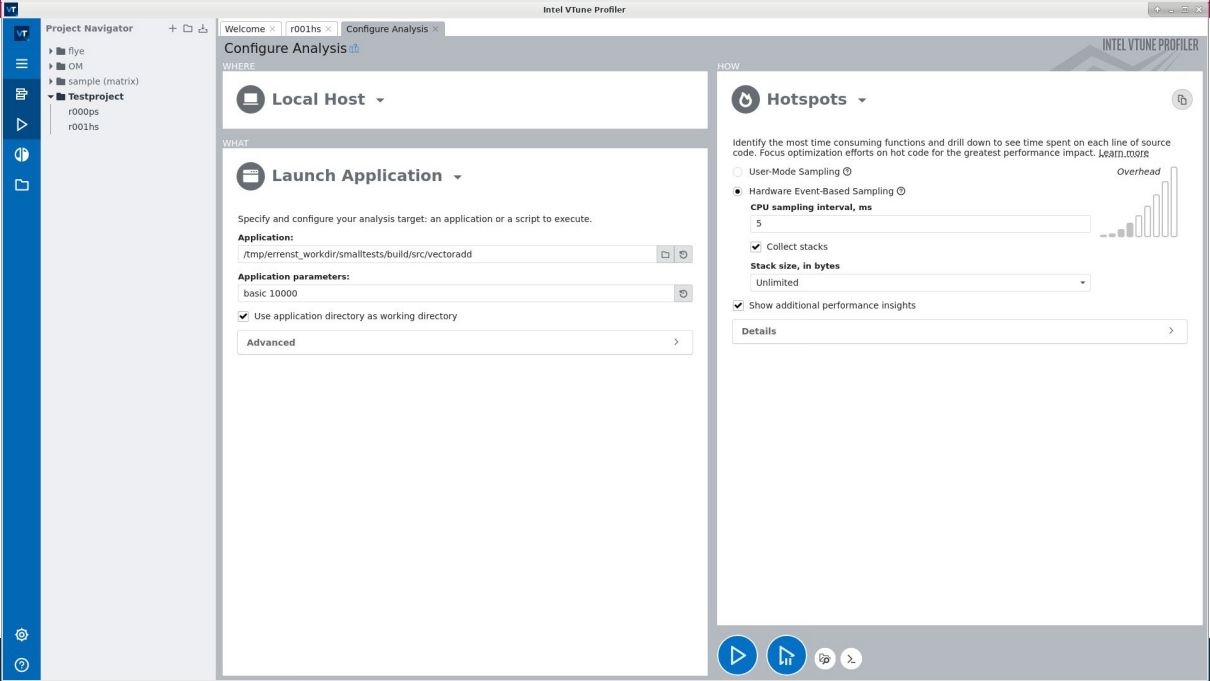
- Different emphasis on algorithm vs. platform
- Metrics and their presentation differ between analysis types



HOTSPOT

Find the worst offenders

- A “hotspot” is a code segment where the program spends most of its time
- User and hardware sampling mode available
- Consider collecting call stacks
- Different sampling intervals and overhead


Intel VTune Profiler
Project Navigator
flye
OM
sample (matrix)
Testproject
r000ps
r001hs
Welcome
r001hs
Configure Analysis
Configure Analysis
INTEL VTUNE PROFILER
WHERE
Local Host
WHAT
Launch Application
Specify and configure your analysis target: an application or a script to execute.
Application:
/tmp/errenst_workdir/smalltests/build/src/vectoradd
Application parameters:
basic 10000
Use application directory as working directory
Advanced
HOW
Hotspots
Identify the most time consuming functions and drill down to see time spent on each line of source code. Focus optimization efforts on hot code for the greatest performance impact. Learn more
User-Mode Sampling
Overhead
Hardware Event-Based Sampling
CPU sampling interval, ms
5
Collect stacks
Stack size, in bytes
Unlimited
Show additional performance insights
Details

Hotspots

Elapsed Time : 34.710s

CPU Time:	34.213s
Instructions Retired:	56,500,578,043
Microarchitecture Usage:	22.2% of Pipeline Slots
CPI Rate:	1.583
Total Thread Count:	1
Paused Time:	0s

Hotspots Insights

If you see significant hotspots in the Top Hotspots list, switch to the Bottom-up view for in-depth analysis per function. Otherwise, use the Caller/Callee or the Flame Graph view to track critical paths for these hotspots.

Explore Additional Insights

Parallelism: 1.1% (0.986 out of 88 logical CPUs)
Use Threading to explore more opportunities to increase parallelism in your application.

Microarchitecture Usage: 22.2%
Use Microarchitecture Exploration to explore how efficiently your application runs on the used hardware.

Top Hotspots

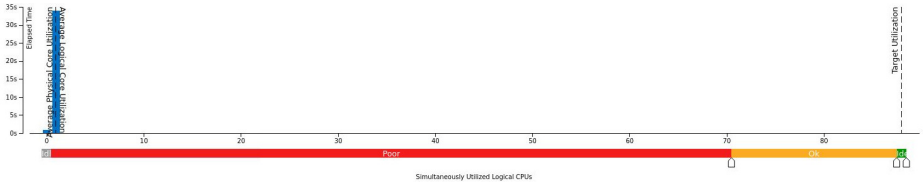
This section lists the most active functions in your application. Optimizing these hotspot functions typically results in improving overall application performance.

Function	Module	CPU Time	% of CPU Time
__memset_sse2	libc-2.17.so	13.480s	39.4%
doComputation	vectoradd	7.974s	23.3%
func@0xffffffff8178c3ad	vmlinux	4.288s	12.5%
brk	libc-2.17.so	3.114s	9.1%
func@0xffffffff8178c5c3	vmlinux	2.493s	7.3%
[Others]	N/A*	2.864s	8.4%

*N/A is applied to non-summable metrics.

Effective CPU Utilization Histogram

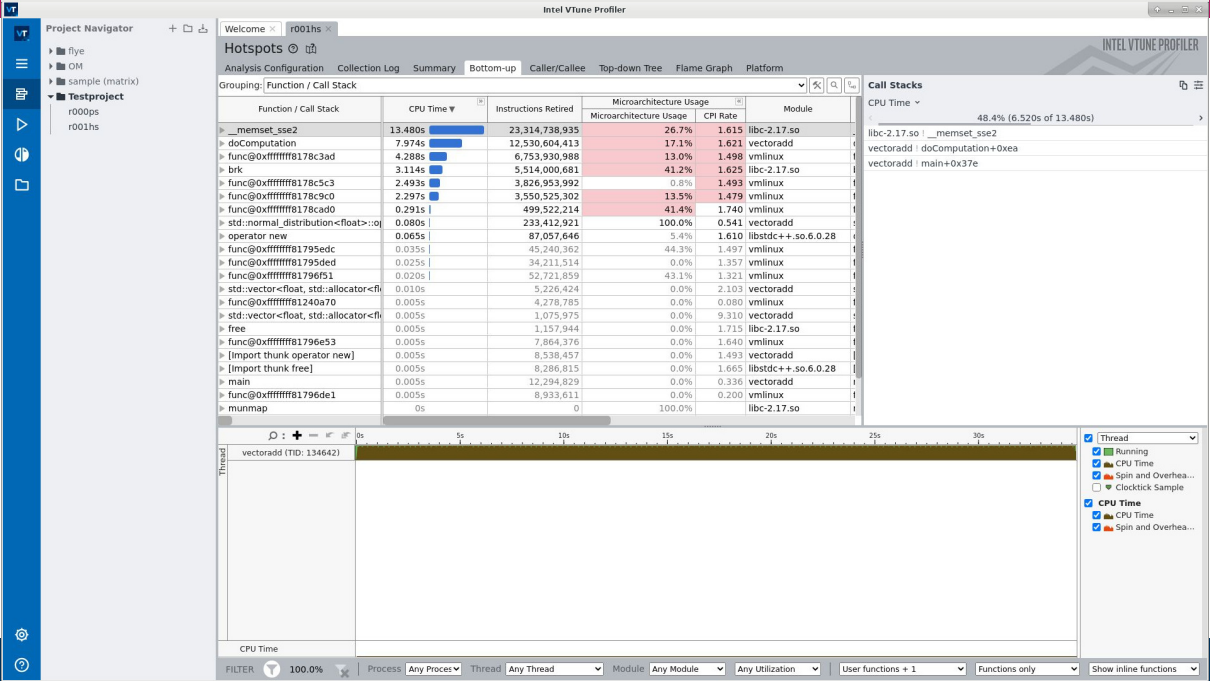
This histogram displays a percentage of the wall time the specific number of CPUs were running simultaneously. Spin and Overhead time adds to the Idle CPU utilization value.



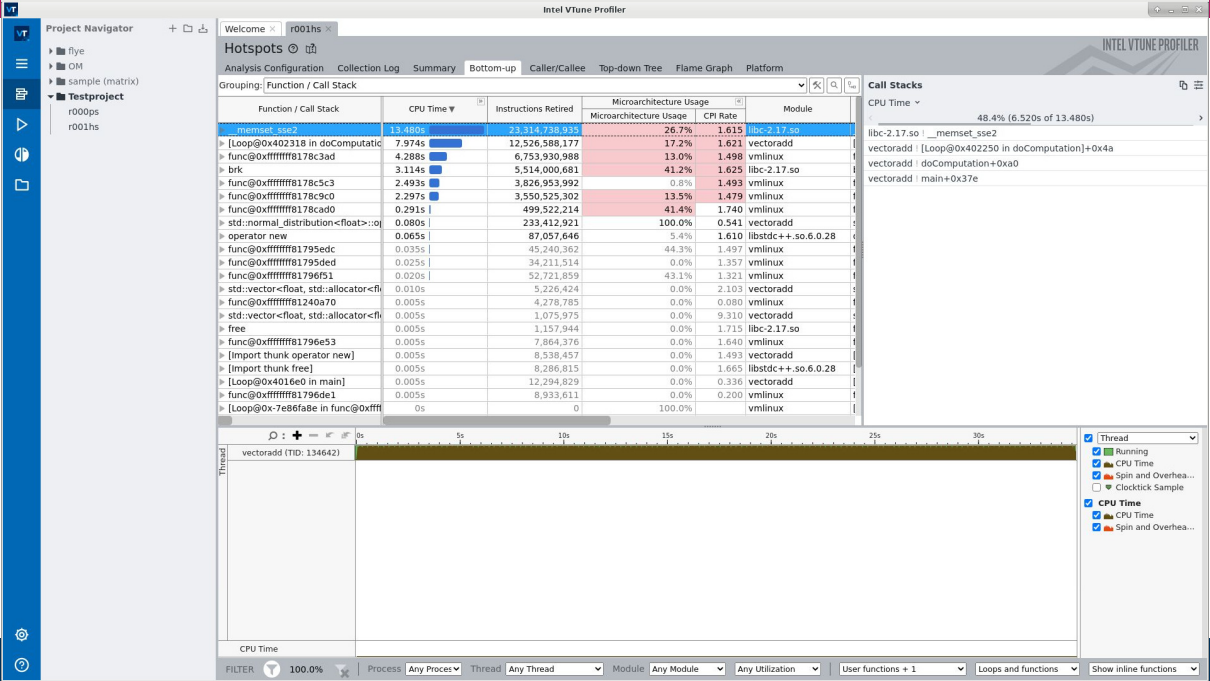
Collection and Platform Info

This section provides information about this collection, including result set size and collection platform data.

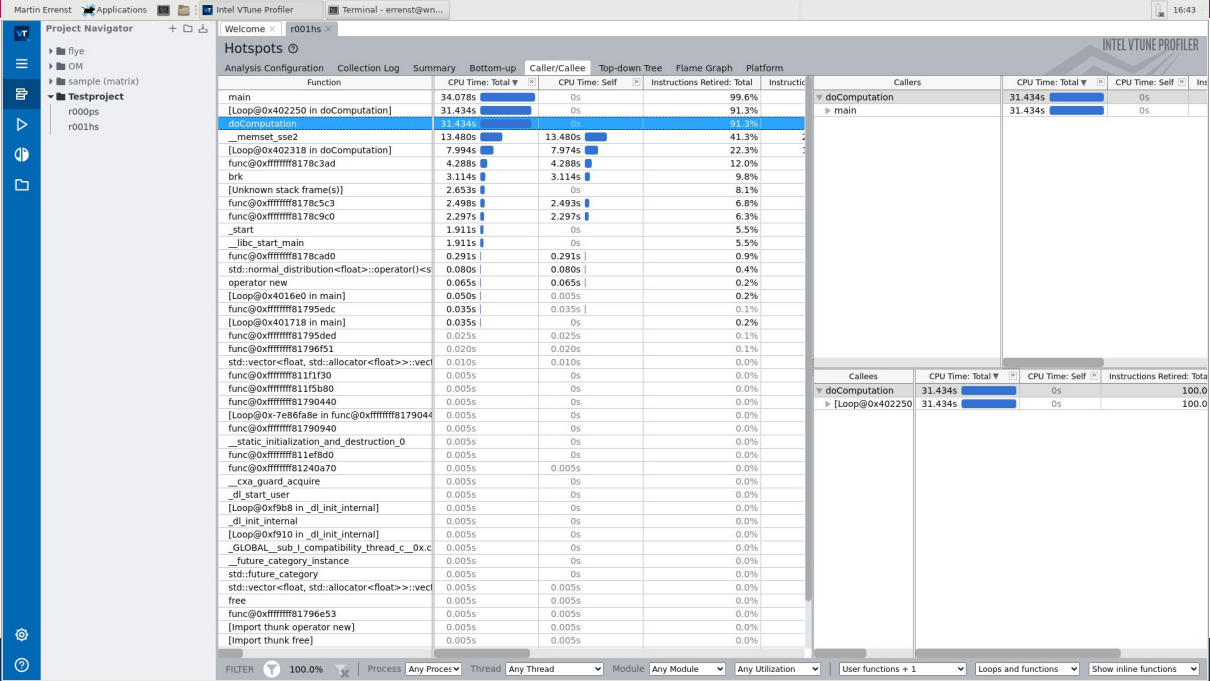
- Results show:
  - Elapsed time
  - Amount of instructions (retired and CPI)
  - Top 5 hotspots
  - Utilization of CPU features (microarchitecture) and parallelization
- Thread histogram only useful when working with threads
- Possible issues marked in red
  - Not always really a problem!


Intel VTune Profiler
Project Navigator
flye
OM
sample (matrix)
Testproject
r000ps
r001hs
Welcome
r001hs
Hotspots
INTEL VTUNE PROFILER
Analysis Configuration
Collection Log
Summary
Bottom-up
Caller/Callee
Top-down Tree
Flame Graph
Platform
Grouping: Function / Call Stack
Function / Call Stack
CPU Time
Instructions Retired
Microarchitecture Usage
Microarchitecture Usage
CPI Rate
Module
__memset_sse2 13.480s 23,314,738,935 26.7% 1.615 libc-2.17.so
doComputation 7.974s 12,530,604,413 17.1% 1.621 vectoradd
func@0xffffffff8178c3ad 4.288s 6,753,930,988 13.0% 1.498 vmlinux
brk 3.114s 5,514,000,681 41.2% 1.625 libc-2.17.so
func@0xffffffff8178c5c3 2.493s 3,826,953,992 0.8% 1.493 vmlinux
func@0xffffffff8178c9c0 2.297s 3,550,525,302 13.5% 1.479 vmlinux
func@0xffffffff8178cad0 0.291s 499,522,214 41.4% 1.740 vmlinux
std::normal_distribution<float>::op 0.080s 233,412,921 100.0% 0.541 vectoradd
operator new 0.065s 87,057,646 5.4% 1.610 libstdc++.so.6.0.28
func@0xffffffff81795edc 0.035s 45,240,362 44.3% 1.497 vmlinux
func@0xffffffff81795ded 0.025s 34,211,514 0.0% 1.357 vmlinux
func@0xffffffff81796f51 0.020s 52,721,859 43.1% 1.321 vmlinux
std::vector<float, std::allocator<fl 0.010s 5,226,424 0.0% 2.103 vectoradd
func@0xffffffff81240a70 0.005s 4,278,785 0.0% 0.080 vmlinux
std::vector<float, std::allocator<fl 0.005s 1,075,975 0.0% 9.310 vectoradd
free 0.005s 1,157,944 0.0% 1.715 libc-2.17.so
func@0xffffffff81796e53 0.005s 7,864,376 0.0% 1.640 vmlinux
[Import thunk operator new] 0.005s 8,538,457 0.0% 1.493 vectoradd
[Import thunk free] 0.005s 8,286,815 0.0% 1.665 libstdc++.so.6.0.28
main 0.005s 12,294,829 0.0% 0.336 vectoradd
func@0xffffffff81796de1 0.005s 8,933,611 0.0% 0.200 vmlinux
munmap 0s 0 100.0% libc-2.17.so
Call Stacks
CPU Time
48.4% (6.520s of 13.480s)
libc-2.17.so | __memset_sse2
vectoradd | doComputation+0xea
vectoradd | main+0x37e
0s
5s
10s
15s
20s
25s
30s
Thread
vectoradd (TID: 134642)
CPU Time
Thread
Running
CPU Time
Spin and Overhea...
Clocktick Sample
CPU Time
CPU Time
Spin and Overhea...
FILTER 100.0%
Process Any Proces
Thread Any Thread
Module Any Module
Any Utilization
User functions + 1
Functions only
Show inline functions

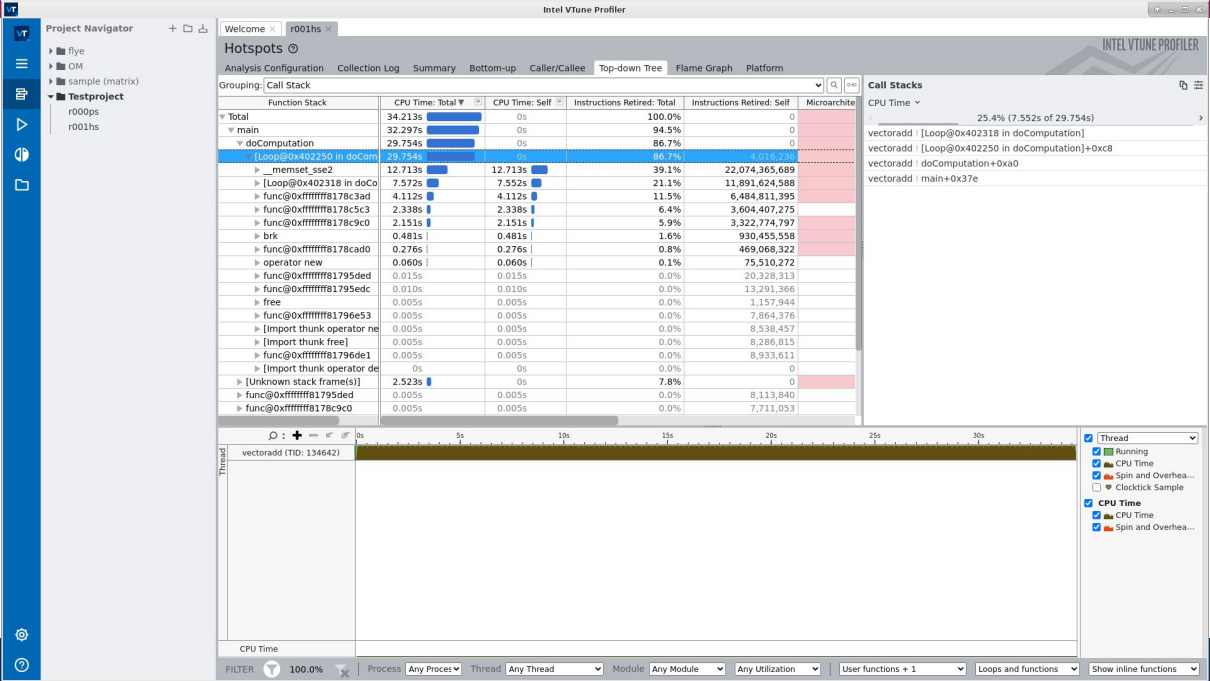
- List of functions with attributed measurements
- Largest hotspots at the top of the list
- Columns change per analysis type!
- Call stack for selected function on the right
- Timeline of thread utilization at bottom
- Useful feature on bottom right: show functions **and loops**


Intel VTune Profiler
Project Navigator
flye
OM
sample (matrix)
Testproject
r000ps
r001hs
Welcome
r001hs
Hotspots
INTEL VTUNE PROFILER
Analysis Configuration
Collection Log
Summary
Bottom-up
Caller/Callee
Top-down Tree
Flame Graph
Platform
Grouping: Function / Call Stack
Function / Call Stack
CPU Time
Instructions Retired
Microarchitecture Usage
Microarchitecture Usage
CPI Rate
Module
__memset_sse2 13.480s 23,314,738,935 26.7% 1.615 libc-2.17.so
[Loop@0x402318 in doComputatic 7.974s 12,526,588,177 17.2% 1.621 vectoradd
func@0xffffffff8178c3ad 4.288s 6,753,930,988 13.0% 1.498 vmlinux
brk 3.114s 5,514,000,681 41.2% 1.625 libc-2.17.so
func@0xffffffff8178c5c3 2.493s 3,826,953,992 0.8% 1.493 vmlinux
func@0xffffffff8178c9c0 2.297s 3,550,525,302 13.5% 1.479 vmlinux
func@0xffffffff8178cad0 0.291s 499,522,214 41.4% 1.740 vmlinux
std::normal_distribution<float>::op 0.080s 233,412,921 100.0% 0.541 vectoradd
operator new 0.065s 87,057,646 5.4% 1.610 libstdc++.so.6.0.28
func@0xffffffff81795edc 0.035s 45,240,362 44.3% 1.497 vmlinux
func@0xffffffff81795ded 0.025s 34,211,514 0.0% 1.357 vmlinux
func@0xffffffff81796f51 0.020s 52,721,859 43.1% 1.321 vmlinux
std::vector<float, std::allocator<fl 0.010s 5,226,424 0.0% 2.103 vectoradd
func@0xffffffff81240a70 0.005s 4,278,785 0.0% 0.080 vmlinux
std::vector<float, std::allocator<fl 0.005s 1,075,975 0.0% 9.310 vectoradd
free 0.005s 1,157,944 0.0% 1.715 libc-2.17.so
func@0xffffffff81796e53 0.005s 7,864,376 0.0% 1.640 vmlinux
[Import thunk operator new] 0.005s 8,538,457 0.0% 1.493 vectoradd
[Import thunk free] 0.005s 8,286,815 0.0% 1.665 libstdc++.so.6.0.28
[Loop@0x4016e0 in main] 0.005s 12,294,829 0.0% 0.336 vectoradd
func@0xffffffff81796de1 0.005s 8,933,611 0.0% 0.200 vmlinux
[Loop@0x-7e86fa8e in func@0xfff 0s 0 100.0% vmlinux
Call Stacks
CPU Time
48.4% (6.520s of 13.480s)
libc-2.17.so | __memset_sse2
vectoradd | [Loop@0x402250 in doComputation]+0x4a
vectoradd | doComputation+0xa0
vectoradd | main+0x37e
0s
5s
10s
15s
20s
25s
30s
Thread
vectoradd (TID: 134642)
CPU Time
Thread
Running
CPU Time
Spin and Overhea...
Clocktick Sample
CPU Time
CPU Time
Spin and Overhea...
FILTER
100.0%
Process
Any Proces
Thread
Any Thread
Module
Any Module
Any Utilization
User functions + 1
Loops and functions
Show inline functions

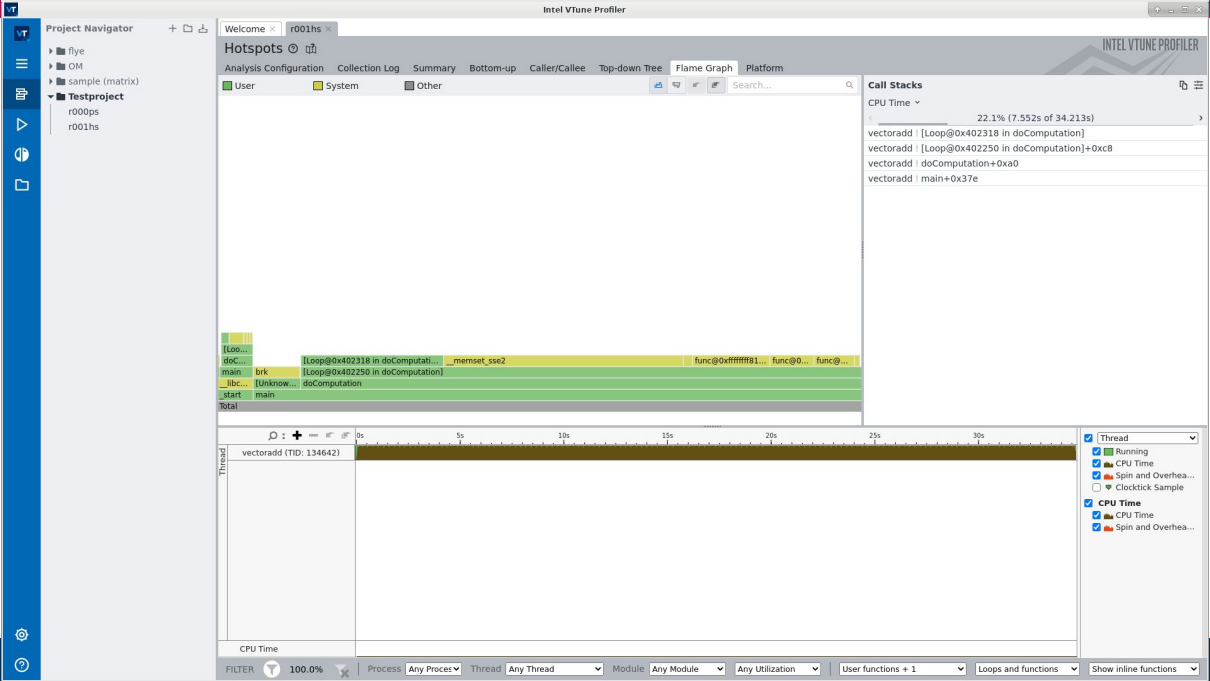
- Cryptic function names point to standard libraries (see “Module” column)
- Compiling with `-g` and possibly debug symbols can help
- Focus on your own functions first!
  - You probably don’t want to spend time on someone else libraries
  - Consider filter results by module (bottom filter bar)
- Indirect inefficiency: If library functions are your worst hotspots, check if there are better ways to use them in your code. Examples are reducing allocations, trying a different container type etc.


Martin Ernest
Applications
Intel VTune Profiler
Terminal - errenst@wn...
16:43
Project Navigator
flye
OM
sample (matrix)
Testproject
r000ps
r001hs
Welcome
r001hs
Hotspots
INTEL VTUNE PROFILER
Analysis Configuration
Collection Log
Summary
Bottom-up
Caller/Callee
Top-down Tree
Flame Graph
Platform
Function
CPU Time: Total
CPU Time: Self
Instructions Retired: Total
main 34.078s 0s 99.6%
[Loop@0x402250 in doComputation] 31.434s 0s 91.3%
doComputation 31.434s 0s 91.3%
__memset_sse2 13.480s 13.480s 41.3%
[Loop@0x402318 in doComputation] 7.994s 7.974s 22.3%
func@0xffffffff8178c3ad 4.288s 4.288s 12.0%
brk 3.114s 3.114s 9.8%
[Unknown stack frame(s)] 2.653s 0s 8.1%
func@0xffffffff8178c5c3 2.498s 2.493s 6.8%
func@0xffffffff8178c9c0 2.297s 2.297s 6.3%
_start 1.911s 0s 5.5%
__libc_start_main 1.911s 0s 5.5%
func@0xffffffff8178cad0 0.291s 0.291s 0.9%
std::normal_distribution<float>::operator()<s 0.080s 0.080s 0.4%
operator new 0.065s 0.065s 0.2%
[Loop@0x4016e0 in main] 0.050s 0.005s 0.2%
func@0xffffffff81795edc 0.035s 0.035s 0.1%
[Loop@0x401718 in main] 0.035s 0s 0.2%
func@0xffffffff81795ded 0.025s 0.025s 0.1%
func@0xffffffff81796f51 0.020s 0.020s 0.1%
std::vector<float, std::allocator<float>>::vect 0.010s 0.010s 0.0%
func@0xffffffff811f1f30 0.005s 0s 0.0%
func@0xffffffff811f5b80 0.005s 0s 0.0%
func@0xffffffff81790440 0.005s 0s 0.0%
[Loop@0x-7e86fa8e in func@0xffffffff8179044 0.005s 0s 0.0%
func@0xffffffff81790940 0.005s 0s 0.0%
__static_initialization_and_destruction_0 0.005s 0s 0.0%
func@0xffffffff811ef8d0 0.005s 0s 0.0%
func@0xffffffff81240a70 0.005s 0.005s 0.0%
__cxa_guard_acquire 0.005s 0s 0.0%
_dl_start_user 0.005s 0s 0.0%
[Loop@0xf9b8 in _dl_init_internal] 0.005s 0s 0.0%
_dl_init_internal 0.005s 0s 0.0%
[Loop@0xf910 in _dl_init_internal] 0.005s 0s 0.0%
_GLOBAL__sub_I_compatibility_thread_c__0x.c 0.005s 0s 0.0%
__future_category_instance 0.005s 0s 0.0%
std::future_category 0.005s 0s 0.0%
std::vector<float, std::allocator<float>>::vect 0.005s 0.005s 0.0%
free 0.005s 0.005s 0.0%
func@0xffffffff81796e53 0.005s 0.005s 0.0%
[Import thunk operator new] 0.005s 0.005s 0.0%
[Import thunk free] 0.005s 0.005s 0.0%
Callers
CPU Time: Total
CPU Time: Self
doComputation 31.434s 0s
main 31.434s 0s
Callees
CPU Time: Total
CPU Time: Self
Instructions Retired: Tota
doComputation 31.434s 0s 100.0
[Loop@0x402250 31.434s 0s 100.0
FILTER
100.0%
Process
Any Proces
Thread
Any Thread
Module
Any Module
Any Utilization
User functions + 1
Loops and functions
Show inline functions

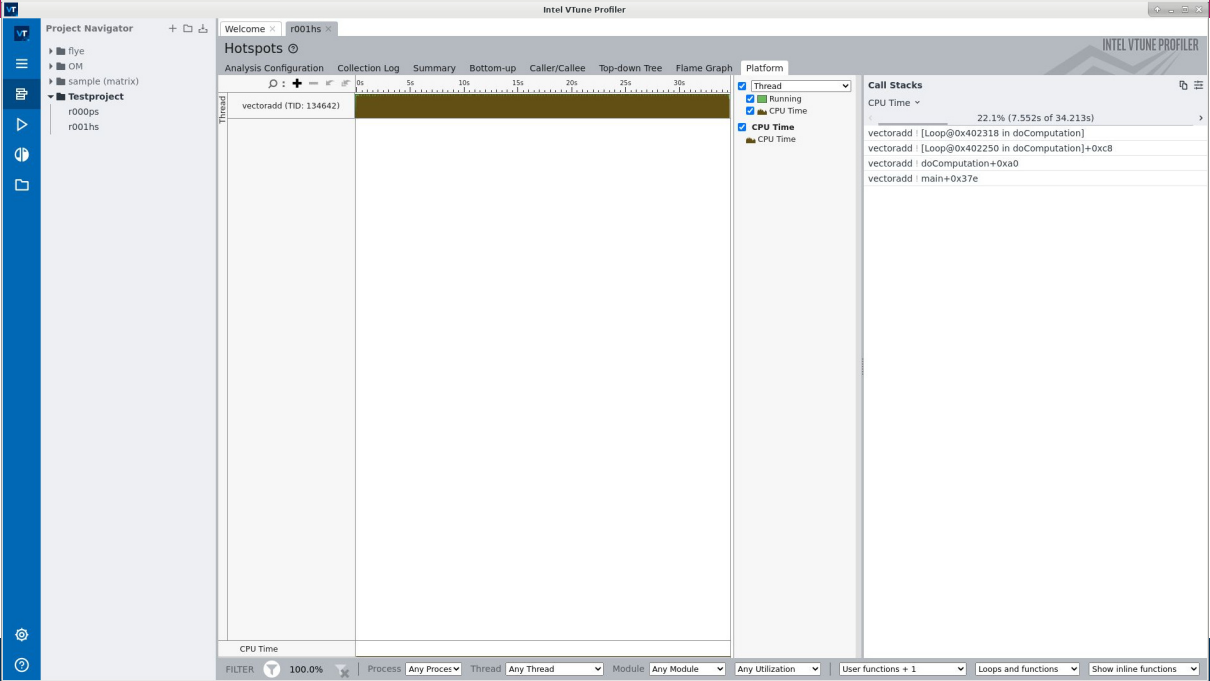
- For a selected function, show:
  - Callers: Where is this function called from?
  - Callees: What functions are called by the selected function?
- Can help with navigating through larger programs
- Hotspot might be caused by a single function in the Callee-stack
- Caller-stack can help with finding section of your program that cause the hotspot
  - If the program is configurable, maybe you can skip/remove/alter the section?


Intel VTune Profiler
Project Navigator
flye
OM
sample (matrix)
Testproject
r000ps
r001hs
Welcome
r001hs
Hotspots
INTEL VTUNE PROFILER
Analysis Configuration
Collection Log
Summary
Bottom-up
Caller/Callee
Top-down Tree
Flame Graph
Platform
Grouping: Call Stack
Function Stack
CPU Time: Total
CPU Time: Self
Instructions Retired: Total
Instructions Retired: Self
Microarchite
Total 34.213s 0s 100.0% 0
main 32.297s 0s 94.5% 0
doComputation 29.754s 0s 86.7% 0
[Loop@0x402250 in doCom 29.754s 0s 86.7% 4,036,236
__memset_sse2 12.713s 12.713s 39.1% 22,074,365,689
[Loop@0x402318 in doCo 7.572s 7.552s 21.1% 11,891,624,588
func@0xffffffff8178c3ad 4.112s 4.112s 11.5% 6,484,811,395
func@0xffffffff8178c5c3 2.338s 2.338s 6.4% 3,604,407,275
func@0xffffffff8178c9c0 2.151s 2.151s 5.9% 3,322,774,797
brk 0.481s 0.481s 1.6% 930,455,558
func@0xffffffff8178cad0 0.276s 0.276s 0.8% 469,068,322
operator new 0.060s 0.060s 0.1% 75,510,272
func@0xffffffff81795ded 0.015s 0.015s 0.0% 20,328,313
func@0xffffffff81795edc 0.010s 0.010s 0.0% 13,291,366
free 0.005s 0.005s 0.0% 1,157,944
func@0xffffffff81796e53 0.005s 0.005s 0.0% 7,864,376
[Import thunk operator ne 0.005s 0.005s 0.0% 8,538,457
[Import thunk free] 0.005s 0.005s 0.0% 8,286,815
func@0xffffffff81796de1 0.005s 0.005s 0.0% 8,933,611
[Import thunk operator de 0s 0s 0.0% 0
[Unknown stack frame(s)] 2.523s 0s 7.8% 0
func@0xffffffff81795ded 0.005s 0.005s 0.0% 8,113,840
func@0xffffffff8178c9c0 0.005s 0.005s 0.0% 7,711,053
Call Stacks
CPU Time
25.4% (7.552s of 29.754s)
vectoradd | [Loop@0x402318 in doComputation]
vectoradd | [Loop@0x402250 in doComputation]+0xc8
vectoradd | doComputation+0xa0
vectoradd | main+0x37e
0s
5s
10s
15s
20s
25s
30s
Thread
vectoradd (TID: 134642)
CPU Time
Thread
Running
CPU Time
Spin and Overhea...
Clocktick Sample
CPU Time
CPU Time
Spin and Overhea...
FILTER 100.0%
Process Any Proces
Thread Any Thread
Module Any Module
Any Utilization
User functions + 1
Loops and functions
Show inline functions

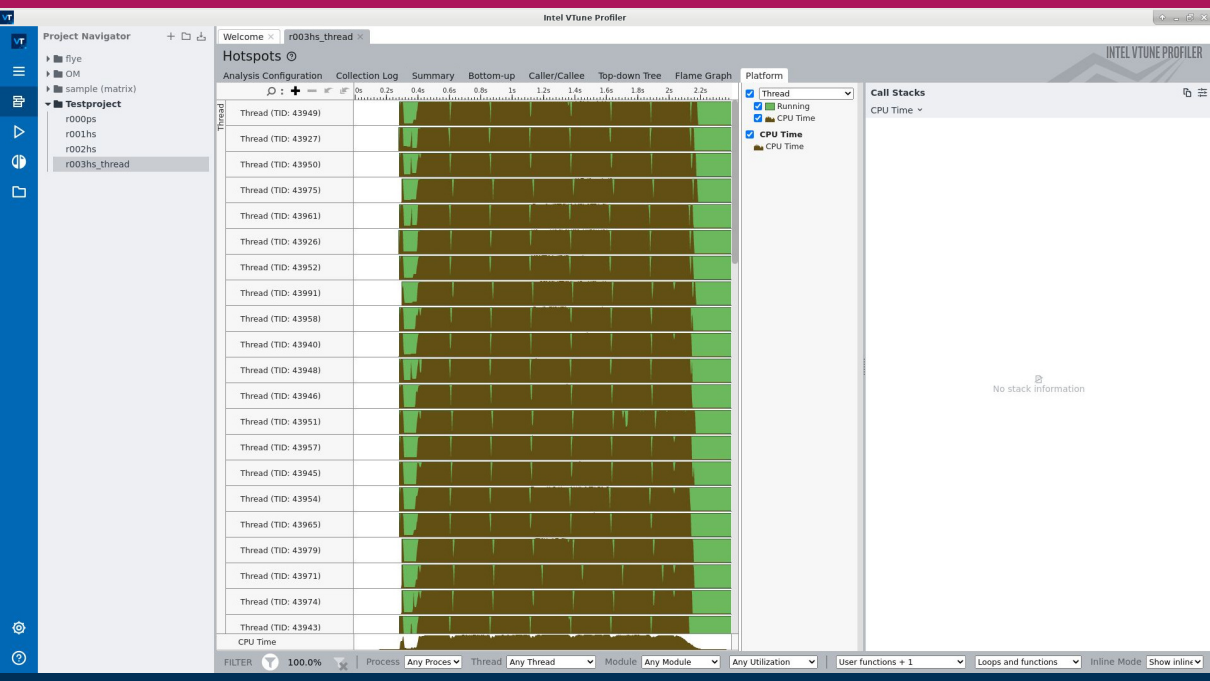
- Go through call stack from start of the program (`main`)
- Can help with navigation in larger programs
- Identify problematic sections of your program


Intel VTune Profiler
Project Navigator
flye
OM
sample (matrix)
Testproject
r000ps
r001hs
Welcome
r001hs
Hotspots
INTEL VTUNE PROFILER
Analysis Configuration
Collection Log
Summary
Bottom-up
Caller/Callee
Top-down Tree
Flame Graph
Platform
User
System
Other
Search...
Call Stacks
CPU Time
22.1% (7.552s of 34.213s)
vectoradd | [Loop@0x402318 in doComputation]
vectoradd | [Loop@0x402250 in doComputation]+0xc8
vectoradd | doComputation+0xa0
vectoradd | main+0x37e
[Loo...
doC...
[Loop@0x402318 in doComputati...
__memset_sse2
func@0xffffffff81...
func@0...
func@...
main
brk
[Loop@0x402250 in doComputation]
__libc...
[Unknow...
doComputation
_start
main
Total
0s
5s
10s
15s
20s
25s
30s
Thread
vectoradd (TID: 134642)
CPU Time
Thread
Running
CPU Time
Spin and Overhea...
Clocktick Sample
CPU Time
CPU Time
Spin and Overhea...
FILTER
100.0%
Process
Any Proces
Thread
Any Thread
Module
Any Module
Any Utilization
User functions + 1
Loops and functions
Show inline functions

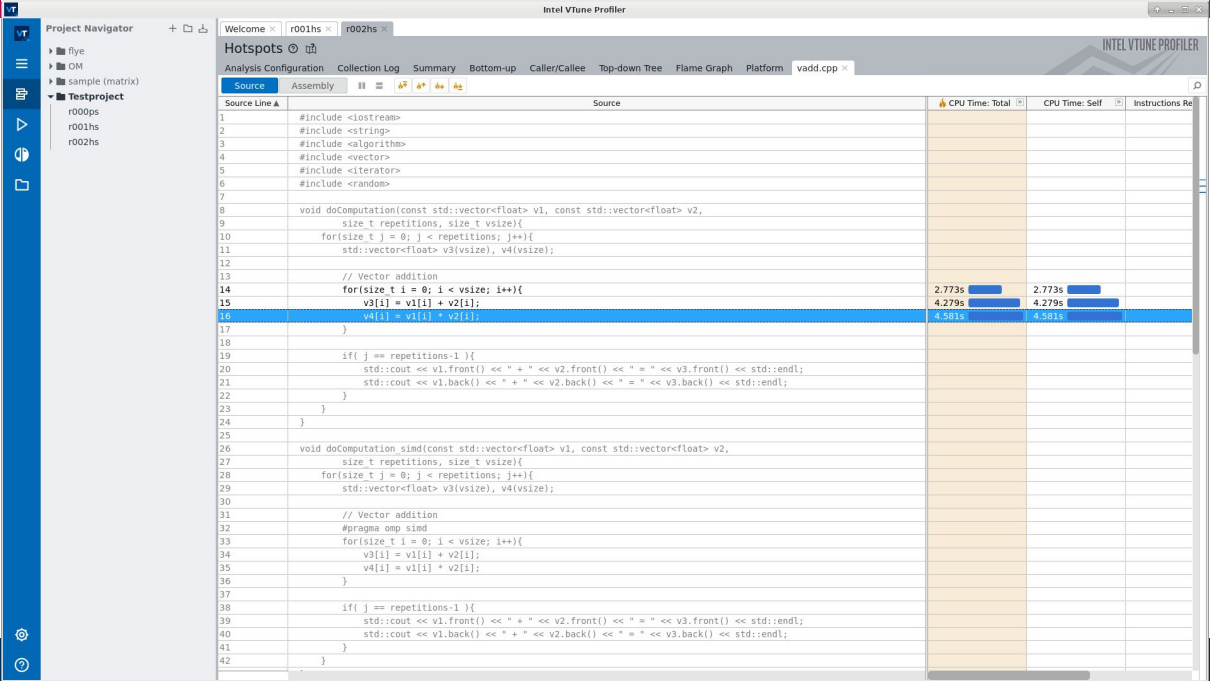
- Show (vertical) depth of call stacks in a timeline
- Hotspots are long on horizontal axis
- Depth and shape of call stacks might tell you something as well


Intel VTune Profiler
Project Navigator
flye
OM
sample (matrix)
Testproject
r000ps
r001hs
Welcome
r001hs
Hotspots
INTEL VTUNE PROFILER
Analysis Configuration
Collection Log
Summary
Bottom-up
Caller/Callee
Top-down Tree
Flame Graph
Platform
Thread
vectoradd (TID: 134642)
0s
5s
10s
15s
20s
25s
30s
Thread
Running
CPU Time
CPU Time
CPU Time
Call Stacks
CPU Time
22.1% (7.552s of 34.213s)
vectoradd | [Loop@0x402318 in doComputation]
vectoradd | [Loop@0x402250 in doComputation]+0xc8
vectoradd | doComputation+0xa0
vectoradd | main+0x37e
CPU Time
FILTER
100.0%
Process
Any Process
Thread
Any Thread
Module
Any Module
Any Utilization
User functions + 1
Loops and functions
Show inline functions

– Timeline of thread and other resource activity
– Depends on analysis type, e.g. others might list memory or disk I/O


Intel VTune Profiler
Project Navigator
flye
OM
sample (matrix)
Testproject
r000ps
r001hs
r002hs
r003hs_thread
Welcome
r003hs_thread
Hotspots
INTEL VTUNE PROFILER
Analysis Configuration
Collection Log
Summary
Bottom-up
Caller/Callee
Top-down Tree
Flame Graph
Platform
0s
0.2s
0.4s
0.6s
0.8s
1s
1.2s
1.4s
1.6s
1.8s
2s
2.2s
Thread
Thread (TID: 43949)
Thread (TID: 43927)
Thread (TID: 43950)
Thread (TID: 43975)
Thread (TID: 43961)
Thread (TID: 43926)
Thread (TID: 43952)
Thread (TID: 43991)
Thread (TID: 43958)
Thread (TID: 43940)
Thread (TID: 43948)
Thread (TID: 43946)
Thread (TID: 43951)
Thread (TID: 43957)
Thread (TID: 43945)
Thread (TID: 43954)
Thread (TID: 43965)
Thread (TID: 43979)
Thread (TID: 43971)
Thread (TID: 43974)
Thread (TID: 43943)
CPU Time
Thread
Running
CPU Time
CPU Time
CPU Time
Call Stacks
CPU Time
No stack information
FILTER
100.0%
Process
Any Proces
Thread
Any Thread
Module
Any Module
Any Utilization
User functions + 1
Loops and functions
Inline Mode
Show inline

– More interesting for multithreaded applications
– Can show synchronisation points or idle times


Project Navigator
flye
OM
sample (matrix)
Testproject
r000ps
r001hs
r002hs
Welcome
r001hs
r002hs
Hotspots
INTEL VTUNE PROFILER
Analysis Configuration
Collection Log
Summary
Bottom-up
Caller/Callee
Top-down Tree
Flame Graph
Platform
vadd.cpp
Source
Assembly
Source Line
Source
CPU Time: Total
CPU Time: Self
Instructions Re
1 #include <iostream>
2 #include <string>
3 #include <algorithm>
4 #include <vector>
5 #include <iterator>
6 #include <random>
7
8 void doComputation(const std::vector<float> v1, const std::vector<float> v2,
9 size_t repetitions, size_t vsize){
10 for(size_t j = 0; j < repetitions; j++){
11 std::vector<float> v3(vsize), v4(vsize);
12
13 // Vector addition
14 for(size_t i = 0; i < vsize; i++){
15 v3[i] = v1[i] + v2[i];
16 v4[i] = v1[i] * v2[i];
17 }
18
19 if(j == repetitions-1){
20 std::cout << v1.front() << " + " << v2.front() << " = " << v3.front() << std::endl;
21 std::cout << v1.back() << " + " << v2.back() << " = " << v3.back() << std::endl;
22 }
23 }
24 }
25
26 void doComputation_simd(const std::vector<float> v1, const std::vector<float> v2,
27 size_t repetitions, size_t vsize){
28 for(size_t j = 0; j < repetitions; j++){
29 std::vector<float> v3(vsize), v4(vsize);
30
31 // Vector addition
32 #pragma omp simd
33 for(size_t i = 0; i < vsize; i++){
34 v3[i] = v1[i] + v2[i];
35 v4[i] = v1[i] * v2[i];
36 }
37
38 if(j == repetitions-1){
39 std::cout << v1.front() << " + " << v2.front() << " = " << v3.front() << std::endl;
40 std::cout << v1.back() << " + " << v2.back() << " = " << v3.back() << std::endl;
41 }
42 }
2.773s
2.773s
4.279s
4.279s
4.581s
4.581s

- Source lines info works when compiled with `-g`
- Annotated assembly always available
- Metrics are sometimes attributed to wrong line
- Larger hotspot-lines have higher probability of having relevant measurements
- Very useful to find starting point of hotspot optimization

- Gathering general information about utilization CPU resources
- Is my application “Front-end” and “Back-end” bound?
- How often do I hit branch mis-predictions or have resteers?

Intel VTune Profiler

Project Navigator

- flye
- OM
- sample (matrix)
- **Testproject**
  - r000ps
  - r001hs
  - r002hs
  - r003hs_thread
  - r004ue

Welcome | r003hs_thread | r004ue

Microarchitecture Exploration

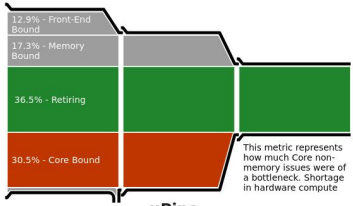
Microarchitecture Exploration

INTEL VTUNE PROFILER

Analysis Configuration | Collection Log | Summary | Bottom-up | Event Count | Platform

Elapsed Time: 31.951s

Clockticks:	115,993,500,000	
Instructions Retired:	139,303,500,000	
CPI Rate:	0.833	
MUX Reliability:	0.985	
Retiring:	36.5%	of Pipeline Slots
Front-End Bound:	12.9%	of Pipeline Slots
Front-End Latency:	9.1%	of Pipeline Slots
ICache Misses:	0.3%	of Clockticks
ITLB Overhead:	1.8%	of Clockticks
Branch Resteers:	8.8%	of Clockticks
DSB Switches:	0.3%	of Clockticks
Length Changing Prefixes:	0.0%	of Clockticks
MS Switches:	5.0%	of Clockticks
Front-End Bandwidth:	3.8%	of Pipeline Slots
Front-End Bandwidth MITE:	7.3%	of Pipeline Slots
Front-End Bandwidth DSB:	0.0%	of Pipeline Slots
(Info) DSB Coverage:	54.4%	
(Info) DSB Misses Cost:	3.9%	of Pipeline Slots
Bad Speculation:	2.8%	of Pipeline Slots
Branch Mispredict:	0.0%	of Pipeline Slots
Machine Clears:	2.8%	of Pipeline Slots
Back-End Bound:	47.8%	of Pipeline Slots
Memory Bound:	17.3%	of Pipeline Slots
Core Bound:	30.5%	of Pipeline Slots
Divider:	0.0%	of Clockticks
Port Utilization:	45.3%	of Clockticks
Cycles of 0 Ports Utilized:	19.2%	of Clockticks
Cycles of 1 Port Utilized:	6.9%	of Clockticks
Cycles of 2 Ports Utilized:	5.7%	of Clockticks
Cycles of 3+ Ports Utilized:	17.7%	of Clockticks
Vector Capacity Usage (FPU):	6.2%	
Average CPU Frequency:	3.7 GHz	
Total Thread Count:	1	
Paused Time:	0s	



µPipe

This diagram represents inefficiencies in CPU usage. Treat it as a pipe with an output flow equal to the "pipe efficiency" ratio: (Actual Instructions Retired)/(Maximum Possible Instruction Retired). If there are pipeline stalls decreasing the pipe efficiency, the pipe shape gets more narrow.

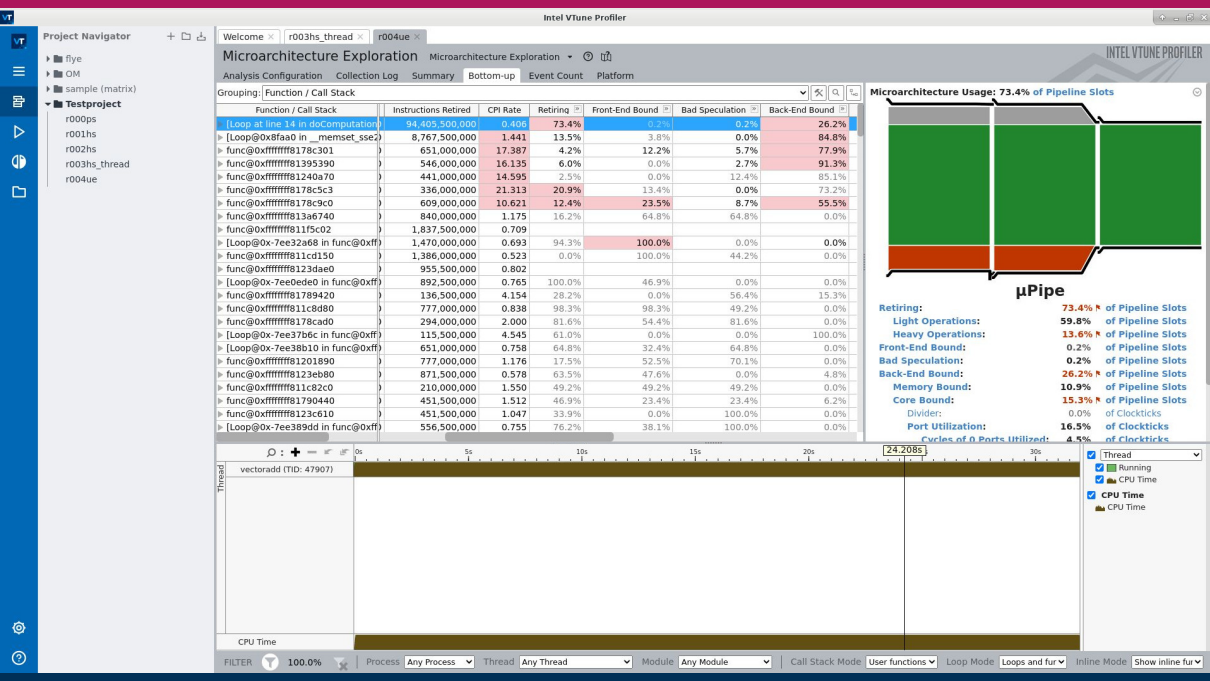
Effective Physical Core Utilization: 2.3% (0.993 out of 44)

Effective Logical Core Utilization: 1.1% (0.993 out of 88)

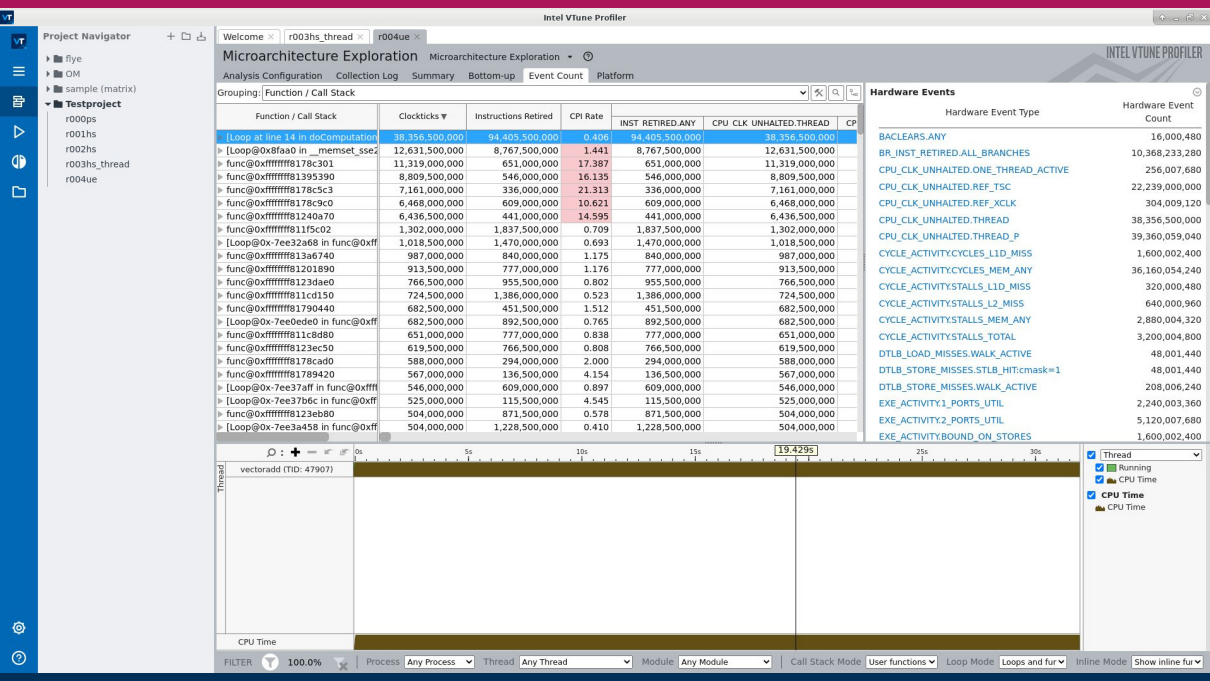
Effective CPU Utilization Histogram

This histogram displays a percentage of the wall time the specific number of CPUs were running simultaneously. Spin and Overhead time adds to the Idle CPU utilization value.

- Different summary overview
- Focus on high level metrics
  - Front end
  - Back end
  - Speculation
- Mouse-over provides very useful ”metric passport”
  - Explains what a metric is measuring
  - What is considered to be a good/bad value?


Intel VTune Profiler
Project Navigator
flye
OM
sample (matrix)
Testproject
r000ps
r001hs
r002hs
r003hs_thread
r004ue
Welcome
r003hs_thread
r004ue
Microarchitecture Exploration
Microarchitecture Exploration
INTEL VTUNE PROFILER
Analysis Configuration
Collection Log
Summary
Bottom-up
Event Count
Platform
Grouping: Function / Call Stack
Function / Call Stack
Instructions Retired
CPI Rate
Retiring
Front-End Bound
Bad Speculation
Back-End Bound
[Loop at line 14 in doComputation] 94,405,500,000 0.406 73.4% 0.2% 0.2% 26.2%
[Loop@0x8ffaa0 in __memset_sse2] 8,767,500,000 1.441 13.5% 3.8% 0.0% 84.8%
func@0xffffffff8178c301 651,000,000 17.387 4.2% 12.2% 5.7% 77.9%
func@0xffffffff81395390 546,000,000 16.135 6.0% 0.0% 2.7% 91.3%
func@0xffffffff81240a70 441,000,000 14.595 2.5% 0.0% 12.4% 85.1%
func@0xffffffff8178c5c3 336,000,000 21.313 20.9% 13.4% 0.0% 73.2%
func@0xffffffff8178c9c0 609,000,000 10.621 12.4% 23.5% 8.7% 55.5%
func@0xffffffff813a6740 840,000,000 1.175 16.2% 64.8% 64.8% 0.0%
func@0xffffffff811f5c02 1,837,500,000 0.709
[Loop@0x-7ee32a68 in func@0xff 1,470,000,000 0.693 94.3% 100.0% 0.0% 0.0%
func@0xffffffff811cd150 1,386,000,000 0.523 0.0% 100.0% 44.2% 0.0%
func@0xffffffff8123dae0 955,500,000 0.802
[Loop@0x-7ee0ede0 in func@0xff 892,500,000 0.765 100.0% 46.9% 0.0% 0.0%
func@0xffffffff81789420 136,500,000 4.154 28.2% 0.0% 56.4% 15.3%
func@0xffffffff811c8d80 777,000,000 0.838 98.3% 98.3% 49.2% 0.0%
func@0xffffffff8178cad0 294,000,000 2.000 81.6% 54.4% 81.6% 0.0%
[Loop@0x-7ee37b6c in func@0xff 115,500,000 4.545 61.0% 0.0% 0.0% 100.0%
[Loop@0x-7ee38b10 in func@0xff 651,000,000 0.758 64.8% 32.4% 64.8% 0.0%
func@0xffffffff81201890 777,000,000 1.176 17.5% 52.5% 70.1% 0.0%
func@0xffffffff8123eb80 871,500,000 0.578 63.5% 47.6% 0.0% 4.8%
func@0xffffffff811c82c0 210,000,000 1.550 49.2% 49.2% 49.2% 0.0%
func@0xffffffff81790440 451,500,000 1.512 46.9% 23.4% 23.4% 6.2%
func@0xffffffff8123c610 451,500,000 1.047 33.9% 0.0% 100.0% 0.0%
[Loop@0x-7ee389dd in func@0xff 556,500,000 0.755 76.2% 38.1% 100.0% 0.0%
Microarchitecture Usage: 73.4% of Pipeline Slots
µPipe
Retiring: 73.4% of Pipeline Slots
Light Operations: 59.8% of Pipeline Slots
Heavy Operations: 13.6% of Pipeline Slots
Front-End Bound: 0.2% of Pipeline Slots
Bad Speculation: 0.2% of Pipeline Slots
Back-End Bound: 26.2% of Pipeline Slots
Memory Bound: 10.9% of Pipeline Slots
Core Bound: 15.3% of Pipeline Slots
Divider: 0.0% of Clockticks
Port Utilization: 16.5% of Clockticks
Cycles of 0 Ports Utilized: 4.5% of Clockticks
0s
5s
10s
15s
20s
24.208s
30s
Thread
vectoradd (TID: 47907)
CPU Time
Thread
Running
CPU Time
CPU Time
CPU Time
FILTER
100.0%
Process
Any Process
Thread
Any Thread
Module
Any Module
Call Stack Mode
User functions
Loop Mode
Loops and fur
Inline Mode
Show inline fur

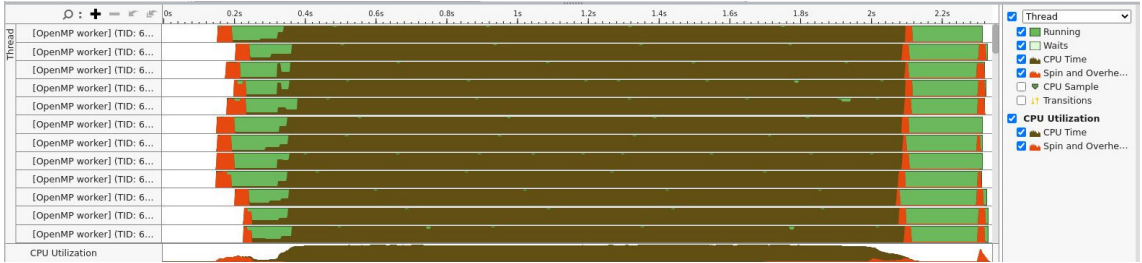
- Similar to hotspots results, but new columns
- Effect of function on high level metrics on the right


Intel VTune Profiler
Project Navigator
flye
OM
sample (matrix)
Testproject
r000ps
r001hs
r002hs
r003hs_thread
r004ue
Welcome
r003hs_thread
r004ue
Microarchitecture Exploration
Microarchitecture Exploration
INTEL VTUNE PROFILER
Analysis Configuration
Collection Log
Summary
Bottom-up
Event Count
Platform
Grouping: Function / Call Stack
Function / Call Stack
Clockticks
Instructions Retired
CPI Rate
INST_RETIRED.ANY
CPU_CLK_UNHALTED.THREAD
[Loop at line 14 in doComputation] 38,356,500,000 94,405,500,000 0.406 94,405,500,000 38,356,500,000
[Loop@0x8faa0 in __memset_sse2 12,631,500,000 8,767,500,000 1.441 8,767,500,000 12,631,500,000
func@0xffffffff8178c301 11,319,000,000 651,000,000 17.387 651,000,000 11,319,000,000
func@0xffffffff81395390 8,809,500,000 546,000,000 16.135 546,000,000 8,809,500,000
func@0xffffffff8178c5c3 7,161,000,000 336,000,000 21.313 336,000,000 7,161,000,000
func@0xffffffff8178c9c0 6,468,000,000 609,000,000 10.621 609,000,000 6,468,000,000
func@0xffffffff81240a70 6,436,500,000 441,000,000 14.595 441,000,000 6,436,500,000
func@0xffffffff811f5c02 1,302,000,000 1,837,500,000 0.709 1,837,500,000 1,302,000,000
[Loop@0x-7ee32a68 in func@0xff 1,018,500,000 1,470,000,000 0.693 1,470,000,000 1,018,500,000
func@0xffffffff813a6740 987,000,000 840,000,000 1.175 840,000,000 987,000,000
func@0xffffffff81201890 913,500,000 777,000,000 1.176 777,000,000 913,500,000
func@0xffffffff8123dae0 766,500,000 955,500,000 0.802 955,500,000 766,500,000
func@0xffffffff811cd150 724,500,000 1,386,000,000 0.523 1,386,000,000 724,500,000
func@0xffffffff81790440 682,500,000 451,500,000 1.512 451,500,000 682,500,000
[Loop@0x-7ee0ede0 in func@0xff 682,500,000 892,500,000 0.765 892,500,000 682,500,000
func@0xffffffff811c8d80 651,000,000 777,000,000 0.838 777,000,000 651,000,000
func@0xffffffff8123ec50 619,500,000 766,500,000 0.808 766,500,000 619,500,000
func@0xffffffff8178cad0 588,000,000 294,000,000 2.000 294,000,000 588,000,000
func@0xffffffff81789420 567,000,000 136,500,000 4.154 136,500,000 567,000,000
[Loop@0x-7ee37aff in func@0xfff 546,000,000 609,000,000 0.897 609,000,000 546,000,000
[Loop@0x-7ee37b6c in func@0xff 525,000,000 115,500,000 4.545 115,500,000 525,000,000
func@0xffffffff8123eb80 504,000,000 871,500,000 0.578 871,500,000 504,000,000
[Loop@0x-7ee3a458 in func@0xff 504,000,000 1,228,500,000 0.410 1,228,500,000 504,000,000
Hardware Events
Hardware Event Type
Hardware Event Count
BACLEARS.ANY 16,000,480
BR_INST_RETIRED.ALL_BRANCHES 10,368,233,280
CPU_CLK_UNHALTED.ONE_THREAD_ACTIVE 256,007,680
CPU_CLK_UNHALTED.REF_TSC 22,239,000,000
CPU_CLK_UNHALTED.REF_XCLK 304,009,120
CPU_CLK_UNHALTED.THREAD 38,356,500,000
CPU_CLK_UNHALTED.THREAD_P 39,360,059,040
CYCLE_ACTIVITY.CYCLES_L1D_MISS 1,600,002,400
CYCLE_ACTIVITY.CYCLES_MEM_ANY 36,160,054,240
CYCLE_ACTIVITY.STALLS_L1D_MISS 320,000,480
CYCLE_ACTIVITY.STALLS_L2_MISS 640,000,960
CYCLE_ACTIVITY.STALLS_MEM_ANY 2,880,004,320
CYCLE_ACTIVITY.STALLS_TOTAL 3,200,004,800
DTLB_LOAD_MISSES.WALK_ACTIVE 48,001,440
DTLB_STORE_MISSES.STLB_HIT:cmask=1 48,001,440
DTLB_STORE_MISSES.WALK_ACTIVE 208,006,240
EXE_ACTIVITY.1_PORTS_UTIL 2,240,003,360
EXE_ACTIVITY.2_PORTS_UTIL 5,120,007,680
EXE_ACTIVITY.BOUND_ON_STORES 1,600,002,400
0s
5s
10s
15s
19.429s
25s
30s
Thread
vectoradd (TID: 47907)
CPU Time
Thread
Running
CPU Time
CPU Time
CPU Time
FILTER
100.0%
Process
Any Process
Thread
Any Thread
Module
Any Module
Call Stack Mode
User functions
Loop Mode
Loops and fur
Inline Mode
Show inline fur

- Raw data of basic metrics
- Metrics in bottom-up view and summary are usually composites of these

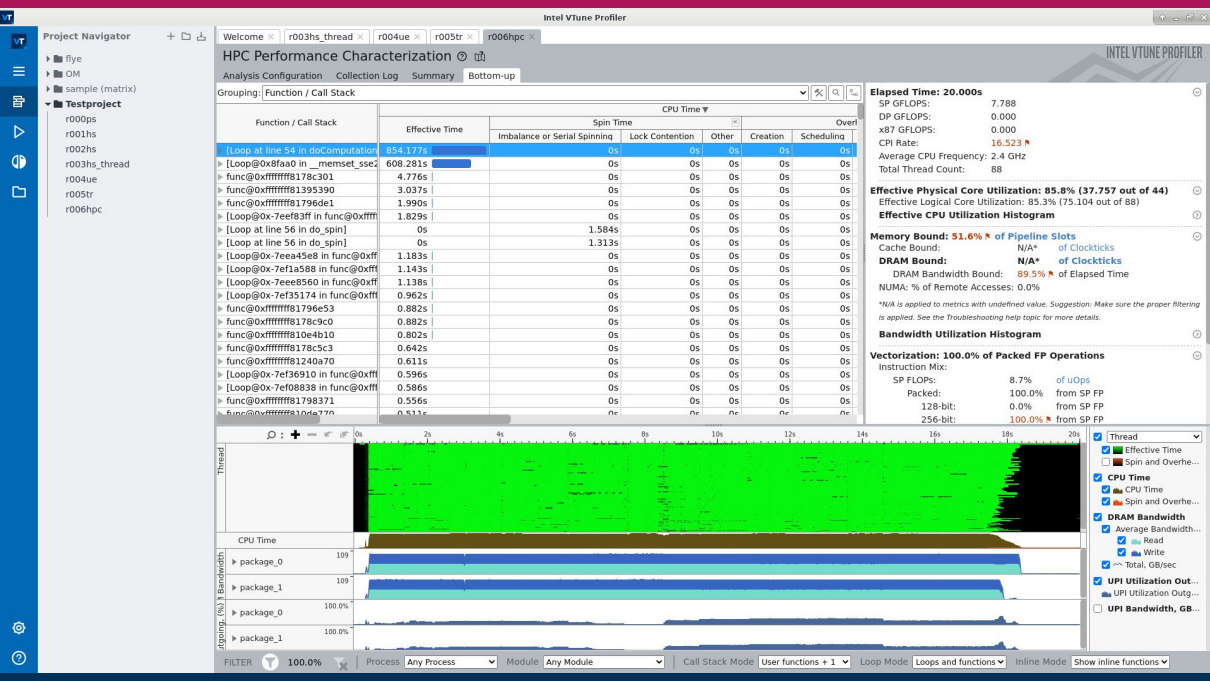
HPC.NRW

- Useful to investigate scaling problems in multithreaded applications
- Gather information on time spent in locks



- To measure allocation counts & sizes
- Identify code sections with large/many allocations
- DRAM bandwidth
- Access to remote DRAM in NUMA (multi-socket) systems
- Can result in very large profile result files

- Analyze aspects in HPC context
- CPU utilization, including OpenMP efficiency
- Memory and FPU utilization


Intel VTune Profiler
Project Navigator
flye
OM
sample (matrix)
Testproject
r000ps
r001hs
r002hs
r003hs_thread
r004ue
r005tr
r006hpc
Welcome
r003hs_thread
r004ue
r005tr
r006hpc
HPC Performance Characterization
INTEL VTUNE PROFILER
Analysis Configuration
Collection Log
Summary
Bottom-up
Grouping: Function / Call Stack
CPU Time
Function / Call Stack
Effective Time
Spin Time
Imbalance or Serial Spinning
Lock Contention
Other
Creation
Scheduling
[Loop at line 54 in doComputation] 854.177s 0s 0s 0s 0s 0s
[Loop@0x8faa0 in __memset_sse2 608.281s 0s 0s 0s 0s 0s
func@0xffffffff8178c301 4.776s 0s 0s 0s 0s 0s
func@0xffffffff81395390 3.037s 0s 0s 0s 0s 0s
func@0xffffffff81796de1 1.990s 0s 0s 0s 0s 0s
[Loop@0x-7eef83ff in func@0xffff 1.829s 0s 0s 0s 0s 0s
[Loop at line 56 in do_spin] 0s 1.584s 0s 0s 0s 0s
[Loop at line 56 in do_spin] 0s 1.313s 0s 0s 0s 0s
[Loop@0x-7eea45e8 in func@0xff 1.183s 0s 0s 0s 0s 0s
[Loop@0x-7ef1a588 in func@0xff 1.143s 0s 0s 0s 0s 0s
[Loop@0x-7eee8560 in func@0xff 1.138s 0s 0s 0s 0s 0s
[Loop@0x-7ef35174 in func@0xff 0.962s 0s 0s 0s 0s 0s
func@0xffffffff81796e53 0.882s 0s 0s 0s 0s 0s
func@0xffffffff8178c9c0 0.882s 0s 0s 0s 0s 0s
func@0xffffffff810e4b10 0.802s 0s 0s 0s 0s 0s
func@0xffffffff8178c5c3 0.642s 0s 0s 0s 0s 0s
func@0xffffffff81240a70 0.611s 0s 0s 0s 0s 0s
[Loop@0x-7ef36910 in func@0xff 0.596s 0s 0s 0s 0s 0s
[Loop@0x-7ef08838 in func@0xff 0.586s 0s 0s 0s 0s 0s
func@0xffffffff81798371 0.556s 0s 0s 0s 0s 0s
Elapsed Time: 20.000s
SP GFLOPS: 7.788
DP GFLOPS: 0.000
x87 GFLOPS: 0.000
CPI Rate: 16.523
Average CPU Frequency: 2.4 GHz
Total Thread Count: 88
Effective Physical Core Utilization: 85.8% (37.757 out of 44)
Effective Logical Core Utilization: 85.3% (75.104 out of 88)
Effective CPU Utilization Histogram
Memory Bound: 51.6% of Pipeline Slots
Cache Bound: N/A* of Clockticks
DRAM Bound: N/A* of Clockticks
DRAM Bandwidth Bound: 89.5% of Elapsed Time
NUMA: % of Remote Accesses: 0.0%
*N/A is applied to metrics with undefined value. Suggestion: Make sure the proper filtering is applied. See the Troubleshooting help topic for more details.
Bandwidth Utilization Histogram
Vectorization: 100.0% of Packed FP Operations
Instruction Mix:
SP FLOPs: 8.7% of uOps
Packed: 100.0% from SP FP
128-bit: 0.0% from SP FP
256-bit: 100.0% from SP FP
0s
2s
4s
6s
8s
10s
12s
14s
16s
18s
20s
Thread
CPU Time
Bandwidth (%)
package_0
package_1
Thread
Effective Time
Spin and Overhe...
CPU Time
CPU Time
Spin and Overhe...
DRAM Bandwidth
Average Bandwidth...
Read
Write
Total, GB/sec
UPI Utilization Out...
UPI Utilization Outg...
UPI Bandwidth, GB...
FILTER
100.0%
Process
Any Process
Module
Any Module
Call Stack Mode
User functions + 1
Loop Mode
Loops and functions
Inline Mode
Show inline functions

VARIOUS PLATFORM ANALYSIS TYPES

HPC.NRW

- Input and Output
- System Overview
- GPU offload / compute
- CPU/FPGA interaction

YOU HAVE EVERYTHING YOU NEED!

HPC.NRW

- This is everything you need to profile and analyze an application!
- Identifying and solving a particular problem is the hard part
  - Depends very much on the details
  - Scientific method: measure, form hypothesis, change it, measure again
  - Aim for relative improvements
- It is useful to look up metric definitions in reference material (tooltips, documentation)