

## Optimising for Windows on Arm

A Deep Dive into WindowsPerf's Newest GUI Tools

#### Table of content

- WindowsPerf
- The Graphical User interface (GUI)
- WindowsPerf GUI: WPA plugin
- WindowsPerf GUI: The Visual Studio extension
- The Demo
- Our roadmap
- References





## WindowsPerf

The one stop shop for profiling on Windows on arm

#### WindowsPerf

- WindowsPerf is a (Linux perf inspired) Windows on Arm performance profiling tool. Profiling is based on ARM64 PMU (Performance Monitor Unit) and its hardware counters.
- **counting model,** for obtaining aggregate counts of occurrences of special events, and
- **sampling model,** for determining the frequencies of event occurrences produced by program locations at the function, basic block, and/or instruction levels.



#### WindowsPerf architecture

WindowsPerf is made up of two main blocks.

- WindowsPerf Kernel Driver: wraps access to the arm hardware for the user space.
- WindowsPerf User space (CLI): A command line tool inspired by Linux perf that output data in text, csv and JSON format.







# The Graphical User interface (GUI)

#### WindowsPerf GUI the ecosystem

• WindowsPerf GUI is a essentially wrapper around the WindowsPerf CLI that helps users interact with it in a more approachable way.

- The GUI is split into two projects
  - WindowsPerf GUI Visual Studio 2022 extension.
  - WindowsPerf Windows Performance Analyzer plugin.



Linaro Connect

#### Why WindowsPerf GUI (WPA plugin)

- The WPA Plugin extends the **Windows Performance Analyzer** to read and visualize data from WindowsPerf *counting* JSON output.
- This integration provides a rich way to analyze counting and timeline data in graphs using already existing infrastructure and familiar environment.



🔊 Linaro Connect

#### Why WindowsPerf GUI (Visual Studio extension)

- The WindowsPerf GUI makes performance profiling more accessible by integrating directly into Visual Studio.
- It allows developers to execute and analyze performance directly within their development environment, reducing context switching and simplifying the process.

bug 🔹 🗛 🗚 🔹 🕨 Local Windows Debugger + 🕨 🌀 - 🐻 👼 🕫 🌭 👘	।'ख'ख  <b>Д</b> Q J Q <sub>2</sub>
x Hac e x main.c ⊡ WindowsPerfSample - 44 {	mpling Settings
c)     45     df_sum = factor1;       46     df_sum *= factor2 * factor2;       47     gf_sum *= (factor2 * factor2;       48     df_sum *= (factor2 * factor2;       49     j       51     #pragma optimize(**, off)       52     #define SIMO_LOOP_LINIT 10000       53     [g=dcLspec(noilline) void simd_hot(unsig)       54     [int22x2_t* aptr = (int32x2_t*);       55     [int22x2_t* optr = (int32x2_t*);       56     [int22x2_t* optr = (int32x2_t*);       57     [int2x2_t* optr = (int32x2_t*);       58     [int2x2_t* optr = (int32x2_t*);       59     [dit13]       59     [dit13]       59     [dit13]       50     [dit13]       57     [dit13]       58     [dit13]       59     [dit13]       59     [dit13]       50     [dit13]       50     [dit13]       51     [dit13]       52     [dit13]       53     [dit13]       54     [dit13]       55     [dit13]       56     [dit13]       57     [dit13]       58     [dit13]       59     [dit13]       59     [dit13]       50     [dit13] <th>Payload From File File path Select File Project Target (Debug   ARM64) Extra arguments : extra args Parameters CPU Core : CPU Core : CPU Core N° 0 Timeout (c) : 30</th>	Payload From File File path Select File Project Target (Debug   ARM64) Extra arguments : extra args Parameters CPU Core : CPU Core : CPU Core N° 0 Timeout (c) : 30
Name V  Secure at 247 PM  inclusor  inclusor	IndowsPerf command preview : perf rccord = kl_spec0x4000000 - c 0 - timeout 30 - annotatedisassemble vjson *C\Users\nader\source\vepos\windowsperfiample\ARM64\Debug XindowsPerfSample.exe* Seve





## WindowsPerf GUI: WPA plugin

#### Windows Performance Analyzer (WPA)

- Included in the Windows Assessment and Deployment Kit (Windows ADK), Windows Performance Analyzer (WPA) is a tool that creates graphs and data tables of Event Tracing for Windows (ETW) events.
- WPA is an extensible tool thanks to its open source SDK, giving us the possibility to feed it data coming out of WindowsPerf.



Linaro Connect

#### WindowsPerf GUI: WPA plugin

- WindowsPerf WPA plugin is an open source project that enables WPA to read JSON files generated by WindowsPerf CLI.
- The plugin was made in accordance with the data structure of WindowsPerf v3.3.3 onwards.
- The main focus of our plugin is the counting model with all of its features.
- Two types of tables are available in our plugin:
  - The counting table: showing data coming out of a single count in multiple views (grouped by event, grouped by CPU core...)
  - The timeline table: showing tables and graphs coming out of our timeline feature plotted over a time axis. (Compatible with on versions of WindowsPerf v3.3.0 or more)



#### WPA plugin how it works

- The plugin can be downloaded from our release page then two possible methods could set it up:
  - Move the *dll* into the *CustomDataSources* folder of Windows Performance Toolkit
  - Open WPA with the flag –addsearchdir and the dll path
- After that a count or a timeline can be ran using WindowsPerf CLI with the flags –json –output out.json to get the results in the correct format.
- Finally the file can then be opened using WPA and the data browsed using all of its features.





## WindowsPerf GUI: The Visual Studio extension

#### How it works

- The Visual Studio extension currently takes the sampling model as its focal point.
- It is a wrapper around WindowsPerf CLI that lets users configure the command to be passed onto the CLI using a graphical interface and then captures the output of said command at the end of the sampling period and displays it in a comprehensive manner from within the comfort of the IDF.





#### Available features: Settings

- A set of settings that are integrated within the Visual Studio settings UI are included to offer user a maximum customizability of the extension.
- A verification of available events and metrics, currently installed version and driver's version are available on the settings page.
- A color resolution selector is available on the a secondary settings page.

Options											
Search Options (Ctrl+E)	ρ	The path	for the wperf	.exe file	wperf.ex	(e					
Tabs and Windows			Validate								
Task List											
Terminal		wperf			3.3.3 (7d	12f9db)					
Trust Settings		wperf-dri	iver		3.3.3 (7d	12f9db)					
Projects and Solutions		Unit of our	- Statella anna A								
Source Control		List of av	allable events								
Work Items		Event N	ame	Event F	Raw Index	Event Description					Event
Text Editor		ase_fp_a	ddsub_spec	0x8031		ase floating-point add & subtract	operation	s spec	ulatively exec	uted	[core
Debugging		ase_fp_c	vt_spec	0x8039		ase floating-point convert operati	ions specu	lativel	y executed		[core
Performance Tools		ase_fp_d	liv_spec	0x8021		ase floating-point divide operatio	ns specula	tively	executed		[core
CMake		ase_fp_dot_spec 0		0x80f1		advanced simd floating-point dot-product operations speculatively executed [co					
Cross Platform		ase_fp_dp_spec 0x8		0x801d		on operations speculatively executed			[core ]		
Database Tools		ase_fp_fma_spec 0:		0x8029	3029 ase floating-point fma operations		speculati	vely ex	ecuted		[core
F# Tools		ase_fp_hp_spec 0x8		0x8015		ase floating-point half-precision operations speculatively executed					[core
IntelliCode		ase_fp_n	e_fp_mmla_spec 0x80f5			advanced simd floating-point matrix multiply operations speculatively executed [					
Live Share		ase_fp_n	nul_spec	0x802d		ase floating-point multiply operations speculatively executed [co					[core
Node.js Tools		ase fp p	reduce spec	0x803d		ase floating-point pairwise add st	ep operati	ons sp	eculativelv ex	ecuted	[core]
NuGet Package Manager		List of av	ailable metric	s:							
Test		Metric	Events				Formula	Unit	Description		
Test Adapter for Google Test		deasha	(Ital analyse It	d an also		and a 12 d and a seful instructional	I	Unit.	Description		
Text Templating		deache	(ITA_AIL_ITA_A	u_cache	_renn,izu_	cache,izd_cache_renn,inst_renred}					
vcpkg Package Manager		icacho	(III cocho III	io_renii,i	efill Di co						
Web Performance Test Tools		imiv	(inct coos dr		eriii,121_ca	o spos (d spos)					
Vindows Forms Designer		ittik	(Inst_spec,up	spec,vi	P_spec,as	e_spec,id_spec/					
▲ WindowsPerf		TUD	(m_ub,m_u	_rem,iz	_00,121_00	_rem,inst_remedy					
Sampling Manager							I	I	1	1	
WindowsPerf Path											
Xamarin											
XAML Designer											

Linaro Connect

0

#### Available features: Host Data

• A Dialog window is available that shows the users the output of the command wperf test which shows details about the hardware that is being used ( such as number of available performance counters, current version of the CPU...)

WindowsPerf test results		X
Test Name	Test Result	
request.ioctl_events [EVT_CORE]	: False	4
request.ioctl_events [EVT_DSU]	: False	
request.ioctl_events [EVT_DMC_CLK/EVT_DMC_CLKDIV2]	: False	
pmu_device.vendor_name	: Arm Limited	
pmu_device.product_name	: armv8-a	
pmu_device.product_name(extended)	:	
pmu_device.product []	: armv8-a,armv9-a,neoverse-n1,neoverse-n2,neoverse-n	2-r0p0,neoverse-n2-r0p1,nec
pmu_device.m_product_alias	: (neoverse-n2-r0p0:neoverse-n2),(neoverse-n2-r0p1:neo	verse-n2)
pmu_device.events_query(events) [EVT_CORE]	: 463	
pmu_device.events_query(events) [EVT_DSU]	: 0	
pmu_device.events_query(events) [EVT_DMC_CLK]	: 0	
pmu_device.events_query(events) [EVT_DMC_CLKDIV2]	: 0	
pmu_device.sampling.INTERVAL_DEFAULT	: 0x4000000	
pmu_device.version_name	: FEAT_PMUv3p1	
PMU_CTL_QUERY_HW_CFG [arch_id]	: 0x000f	
PMU_CTL_QUERY_HW_CFG [core_num]	: 0x0008	
PMU_CTL_QUERY_HW_CFG [fpc_num]	: 0x0001	
PMU_CTL_QUERY_HW_CFG [gpc_num]	: 0x0005	
PMU_CTL_QUERY_HW_CFG [total_gpc_num]	: 0x0005	
PMU_CTL_QUERY_HW_CFG [part_id]	: 0x0d4c	
PMU_CTL_QUERY_HW_CFG [pmu_ver]	: 0x0004	
PMU_CTL_QUERY_HW_CFG [rev_id]	: 0x0000	
PMU_CTL_QUERY_HW_CFG [variant_id]	: 0x0000	
PMU_CTL_QUERY_HW_CFG [vendor_id]	: 0x0041	
PMU_CTL_QUERY_HW_CFG [midr_value]	: 0x00000000000410fd4c0	
PMU_CTL_QUERY_HW_CFG [id_aa64dfr0_value]	: 0x0000000000010305408	
gpc_nums[EVT_CORE]	: 5	
gpc_nums[EVT_DSU]	: 0	
gpc_nums[EVT_DMC_CLK]	: 0	
gpc_nums[EVT_DMC_CLKDIV2]	: 0	
fpc_nums[EVT_CORE]	: 1	
fpc_nums[EVT_DSU]	: 0	
fpc_nums[EVT_DMC_CLK]	: 0	
fpc_nums[EVT_DMC_CLKDIV2]	: 0	
Carl Susseint CODD Calm		



Close

#### Available features: Sampling Settings

- 1. Payload selection where the user can either manually select the target payload for the sampling run or choose the project target which automatically detects the current build target and spins up the most recently built process.
- CPU Core selection that automatically detects the number of available CPUs on your machine and pin the process to that core.
- 3. An optional timeout field that sets the length of the sampling run.

ampling Settings			
Payload	1	Raw Event :	
O Payload From File		Raw Event :	
File path		rF123:80000	+
	Select File	vfp_spec   operation specula 🗸	Select Event frequency 🗸 + 🗕
Project Target (Debug   ARM64)		Event	Frequency
Extra arguments :		ld spos	0~400000
extra args		inst spec	0x4000000
Darametere		vfp_spec	0x4000000
Parameters		ase_spec	0x4000000
CPU Core :	2		
CPU Core N° 0	~		
Timeout (s) :			
30	3		
VindowsPerf command preview :			

wperf record -e ld\_spec:0x4000000,inst\_spec:0x4000000,vfp\_spec:0x4000000,ase\_spec:0x4000000 -c 0 --timeout 30 --annotate --disassemble -v --json -- "C \Users\nader\source\vepos\windowsperfsample\ARM64\Debug\WindowsPerfSample.exe"

🔊 Linaro Connect

Save

### Available features: Sampling Settings

- 1. Event selection: The user can select as many events as allowed by their hardware and for each event optionally select a frequency
- 2. Due to the sheer number of available events an autocomplete feature was added to the event field.
- **3.** If none of the available events are what the user is looking for they can manually input an event using its raw index.
- 4. The culmination of these settings is shown in the command preview at all times. This command is the one that the extension will run in the background.
- 5. Once the settings are saved the sampling can be started from the Sampling Explorer and the settings will persist across the session.

Sampling Settings			
- Pavload		Raw Event :	
O Payload From File			
		Raw Event :	3
File path		rF123:80000	-0
	Select File	cache 🗸 Select Event frequency 🗸	+1-
Project Target (Debug   ARM64)		IId_cache   level 1 data cache access	
Extra arguments :		I1d_cache_allocate   level 1 data cache allocation without refill	2
		I1d_cache_hwprf   level 1 data cache hardware prefetch	
exua arys		l1d_cache_inval   level 1 data cache invalidate	
Parameters		<pre>I1d_cache_Imiss_rd   level 1 data cache long-latency read miss</pre>	
		I1d_cache_miss   level 1 data or unified cache demand access miss	
CPU Core :		IId_cache_prfm   level 1 data or unified cache preload or prefetch	
CPU Core N° 0	~	IId_cache_rd   level I data cache access, read	
Timeout (s) :		Ind_cache_refill_inner_l.level1_data_cache_refill_inner	
30		11d_cache_refill_outer Llevel 1 data cache refill_outer	
		11d_cache_refill_prfm_level 1 data or unified cache refill_preload o	or prefetc
		I1d cache refill rd   level 1 data cache refill, read	
		l1d_cache_refill_wr   level 1 data cache refill, write	
		I1d_cache_rw   level 1 data or unified cache demand access	
		l1d_cache_wb   level 1 data cache write-back	
WindowsPerf command preview :	,	Itd_cache_wb_clean   level 1 data cache write back, cleaning and c	coherenc
windowsreit command preview.	20	<pre>I1d_cache_wb_victim   level 1 data cache write-back, victim</pre>	
WindowsPerfSample.exe"	BU annotate disassemble	V 11d eacha ur Llaval 1 data cacha accors urita	4
			Save



#### Available features: Sampling Explorer

- The sampling explorer is the starting point for any profiling session.
- The UI (User Interface) is split in two sections: •
  - A tree structure present the sampling output starting at the event cascading down to the line number Ο thanks to the annotate feature of WindowsPerf.
  - A further detailed section where whenever an element is selected on the tree further details are given Ο on that section.

ampling Explorer				•	ų х
▶ ■ 準 參					
Name	Line Number	Hits	Overhead	Name: VCRUNTIME140D.dll	4
+ Executed at 2:47 PM		3782	100 %	File Path: C:\Windows\SYSTEM32\VCRUNTIME140D.dll	
+ Executed at 2:52 PM		3633	100 %	Name: WindowsPerfSample.exe	
+ Executed at 2:56 PM		3633	100 %	File Path: C:\Users\nader\source\repos\windowsperfsample\ARM64\Debug\WindowsPerfSample.exe	
+ Executed at 3:02 PM		3603	100 %	Name: otdil dil	
+ Executed at 3:03 PM		3834	100 %	File Bath: C/Windows/SVSTEM22) at dll dll	
+ Executed at 3:06 PM		3634	100 %	File Fath: C:\windows\5Y51EM52\httdi.dli	
- Executed at 3:15 PM		3753	100 %	Name: ucrtbased.dll	
+ inst_spec		3263	86.94 %	File Path: C:\Windows\SYSTEM32\ucrtbased.dll	
- Id_spec		448	11.94 %		
- simd_hot		448	100 %	Event List:	
C:\Users\nader\source\repos\windowsperfsample\lib.c	75	411	91.74 %	Name: ase spec	
C:\Users\nader\source\repos\windowsperfsample\lib.c	74	29	6.47 %	Select Event frequency 67108864	
C:\Users\nader\source\repos\windowsperfsample\lib.c	76	8	1.79 %	News lock requires 07100007	
				Name: inst_spec	
				Select Event frequency 67108864	
				Name: Id_spec	
				Select Event frequency 67108864	
				🐼 Linaro Connect 🛛 Madrid 2024	

#### Available features: Sampling Explorer

- The user can have a zoomed in view of his code on the right side section giving them more context thanks to the disassembly feature that was introduced to WindowsPerf.
- The user can click on the link to jump to that line of code in the editor.
- The user can run multiple sampling events in sequence and visualize the impact of their changes on the performance counters.

Sampling Explorer							Ψ×
Name	Line Number	Hits	Overhead	16428	mov	x10, #0x10	
Executed at 2:47 PM		3782	100 %	1642c	ասլ	x10, x9, x10	
Executed at 2:52 PM		3633	100 %	10420	ind 2		
ase_spec		17	0.47 %	16430	ldr	x9, [x27, #0x20]	
inst_spec		3211	88.38 %	16424	add	×0 ×0 ×10	
Id_spec		366	10.07 %	10454	auu	,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,	
sime_not	50	500 227	99.73 %	16438	ldr	q16, [x9]	
C:\User\pader\source\teps\windowsperfsample\lib.c	J9 58	2/	92.55 %				
C-Ulsershader(source)repos/windowsperfsample/lib.c	60	4	1.1 %	1043C			
CheckForDebuggerJustMvCode			0.27 %	16440	add	v16.4s, v16.4s, v17.4s	
Executed at 2:56 PM		3633	100 %			0 [ 07 #0 20]	
ase_spec			0.19 %	16444	lar	w8, [x27, #0x30]	
inst_spec		3138	86.37 %	16448	sxtw	x8, w8	
ld_spec		455	12.52 %				
- simd_hot		455	100 %	1644c	mov	x9, #0x10	
<u>C:\Users\nader\source\repos\windowsperfsample\lib.c</u>	59	405	89.01 %	16450	ուլ	<b>x0. x8. x0</b>	
C:\Users\nader\source\repos\windowsperfsample\lb.c	58	43	9.45 %	10150	inter 1		
C:\Users\nader\source\repos\windowspertsample\lib.c	60		1.54 %	16454	ldr	x8, [x27, #0x18]	

Linaro Connect

#### Available features: Sampling Explorer

- When clicking on a sample all the lines that triggered hits will be highlighted forming a heatmap of the hotspots in the code base.
- The user can filter the highlighted lines by clicking on a specific event.
- This feature can be useful if the user wants to focus on a specific event.

```
double factor1 = 13.45;
     double factor2 = 13.45;
     df_sum /= factor1; // 9.09% with 1 hit (vfp_spec:100000)
     df_sum += factor2 * factor2; // 36.36% with 4 hits (vfp_spec:100000)
     df_sum *= (factor1 * 3.45 + factor2 / 6.78); // 36.36% with 4 hits (vfp_spec:100000)
     if (cond)
         return df_sum; // 18.18% with 2 hits (vfp_spec:100000)
     for (int i = 0; i < 32; i++)
         df_sum /= factor1;
         df_sum += factor2 * factor2;
         df_sum *= (factor1 * 3.45 + factor2 / 6.78);
     return df_sum;
 #define SIMD_LOOP_LIMIT 10000
∃void simd_hot(unsigned int * __restrict a, unsigned int * __restrict b, unsigned int *
     for (int i = 0; i < SIMD_LOOP_LIMIT; i++) // 100% with 1 hit (vfp_spec:100)
         a[i] = b[i] + c[i]; // 89.04% with 203 hits (ld_spec:67108864)
```





## The Demo!

0.json 🍳 👘				m	1) stat anen() is an	n =		
nader > {-} s	ዾ Windows PowerShell 🛛 🗡						- 0	×
"timeline"			• • •	· • · ·	10	10		
	PS C:\Users\hader>	wper+ stat -m 1	.mix -c ⊍ -t -:	1 🛛timeo	ut 10 -n 1	10jsonoutput	stat-core0.json	
"dmc":								
"Time_€								
"Mult								
"Kerr								
						"Performance_counter": [		
"F	Performance_counter": [					l "counter value": 6227	40501	
						"event name": "cvcle"		
	"event name": "cvcle"					"event_idx": "fixed",		
	"event_idx": "fixed".					"event_note": "e"		
	"event note": "e"							
						"counter_value": 4318	79587,	
	"counter_value": 64263770943,					"event_name": "inst_s	pec",	
	"event_name": "inst_spec",					"event_lux : 0x1D ,	L	
	"event_idx": "0x1b",					}.	î la	
	event_note : g0,1m1x							
							27388,	
	"counter value": 8251979479.					"event_name": "dp_spe	c",	
	"event name": "dp spec",					"event_idx": "0x73",		
	"event_idx": "0x73",					"event_note": "g0,imi	¢"	
	<pre>"event_note": "g0,imix"</pre>							
						t "counton voluo": 2284	573	
						"event name": "vfn sn	er"	
	"counter_value": 25614506,					"event idx": "0x75".		
	"event_name : vrp_spec ;					"event_note": "g0,imi	x"	
	"event_note": "g0.imix"							
	},							
						"counter_value": 4972	56,	
	"counter_value": 15900400264,					"event_name": "ase_sp	ec",	
	"event_name": "ase_spec",					"event_idx": "0x/4",		
	"event_idx": "0x74",					event_note : "g0,1m1		
	"event_note": "g0,imix"							
						"counter value": 1003	28893.	
	1 "counter value": 22084525922					"event_name": "ld spe	c",	
					47	"event_idx": "0x70".		







ැ ម័ទ

#### Sampling Settings

Payload		Raw Event :		
O Payload From File		Raw Event :		
File path		rF123:80000		] +
	Select File	vfp_spec   operation specula 🗸	Select Event frequency 🗸	) + -
<ul> <li>Project Target (Debug   ARM64)</li> <li>Extra arguments :</li> </ul>		Event	Frequency	
extra args		ld_spec	0x4000000	
		inst_spec	0x4000000	
Parameters		vfp_spec	0x4000000	
CPU Core :		ase_spec	0x400000	
CPU Core N° 0	~			
Timeout (s) :				
30				

WindowsPerf command preview :

wperf record -e ld\_spec:0x4000000,inst\_spec:0x4000000,vfp\_spec:0x4000000,ase\_spec:0x4000000 -c 0 --timeout 30 --annotate --disassemble -v --json -- "C: \Users\nader\source\repos\windowsperfsample\ARM64\Debug\WindowsPerfSample.exe"

Save

Х





ebug 🔹 ARM64 🔹 🕨 Local Windows Debugger 🔹 🖉 🗧 📮	🜞   ሬ 얘   Έ 결   🛛 ର 지 බ 🍦		<u>ن</u> <u>ش</u>
X lib.c + X main.c			- ¢
The WindowsPerfSample	+ (Global Scope)	🗕 🗘 si	md_hot(unsigned int *restrict a, unsigned int *restrict b, unsigned int *restrict c) $\rightarrow \frac{+}{+}$
44       {         45       df_sum /= factor1;         46       df_sum += factor2 * factor2         47       df_sum *= (factor1 * 3.45 *         48       }         49       return df_sum;         50       #pragma optimize("", off)         51       #pragma optimize("", off)         52       #define SIMD_LOOP_LIMIT 10000         53       =declspec(noinline) void simd_hot(         54       int32x2_t* aptr = (int32x2_t*)e         56       int32x2_t* cptr = (int32x2_t*)e         57       int32x2_t* cptr = (int32x2_t*)e         58       for (int i = 0; i < SIMD_LOOP_L	<pre>factor2 / 6.78); unsigned int*restrict a, unsigned ; ; ; IMIT / 2; i++) { // 8.45% (0.03% with 1 h cptr[i]); // 88.77% (0.44% with 16 hits ( cc:67108864), 0.11% with 4 hits (Id_spec:67108 </pre>	<pre>int*restrict b, unsigned its (ase_spec:67108864), 7.76% with 2 se_spec:67108864), 79.05% with 2872 1 64))</pre>	<pre>int*restrict c)  82 hits (inst_spec:67108864), 0.66% with 24 hits (ld_spec:67108864))  82 hits (inst_spec:67108864), 9.28% with 337 hits (ld_spec:67108864))  * In:59 Ch:21 SPC CRUE * I × I</pre>
▶ ■ ≦ ₩			
Name	Line Number	Hits Overhead 1642c mov	x10, #0x8
Executed at 2:47 PM		3782 100 % 16430 mul	x10, x9, x10
ld_spec		297 7.85 % 16434 ldr	x9, [x27, #0x20]
simd_hot C\Users\nader\source\renos\windowsnerfsamnle\lib.c	54	297 100 % 274 92 26 % 16438 add	x9, x9, x10
<u>C:\Users\nader\source\repos\windowsperfsample\lib.c</u>	55	23 7.74% 1643c ldr	d16. [x9]
Executed at 2:52 PM ase spec		3633 100 % 16440 1dp	d17. [v8]
inst_spec		3211 88.38 %	v16.25 v16.25 v17.25
simd_hot		365 99.73 % 16448 Idm	VI0123, VI0123, VI7123
<u>C:\Users\nader\source\repos\windowsperfsample\lib.c</u> C\Users\nader\source\repos\windowsperfsample\lib.c	59	337 92.33 %	
C:\Users\nader\source\repos\windowsperfsample\lib.c	60	4 1.1%	
CheckForDebuggerJustMyCode		0.27% 16450 mov	x9, #0x8
		16454 mul	x9, x8, x9
		16458 ldr	x8, [x27, #0x18]





## Our roadmap

#### WindowsPerf GUI Roadmap

- Deploying our Visual Studio extension to the Visual Studio marketplace.
- Packaging to an installer: one stop shop of WindowsPerf + VSextension + WPA-plugin.
- Adding the counting model to the Visual Studio extension.
- Linking the Visual Studio extension to the WPA plugin to further inspect the counting results.
- Maintaining a one on one feature completeness with the WindowsPerf CLI.



#### References

- Blog Posts
  - Introducing 1.0.0-beta release of WindowsPerf Visual Studio extension
  - Introducing the WindowsPerf GUI: the Visual Studio 2022 extension
  - Announcing WindowsPerf: Open-source performance analysis tool for Windows on Arm
  - WindowsPerf Release 2.4.0
  - WindowsPerf Release 2.5.1
  - WindowsPerf Release 3.0.0
  - WindowsPerf Release 3.3.0
- External Documentation
  - Perf for Windows on Arm (WindowsPerf) @ Arm Learning path
  - Get started with WindowsPerf @ Arm Learning Path
  - Sampling CPython with WindowsPerf @ Arm Learning Path



#### **Additional Resources**

- Arm Telemetry Solution @ Arm Developer
- Topdown-tool Install Guide @ Arm Learning Path
- Arm CPU Telemetry Solution Topdown Methodology Specification @ Arm Developer





## Thank you