Threads Efficiency Analysis of Selected Operating Systems

Josef Horalek and Vladimir Sobeslav(✉)

Faculty of Informatics and Management, University of Hradec Kralove, Hradec Kralove, Czech Republic
{josef.horalek,vladimir.sobeslav}@uhk.cz

Abstract. The aim of the article is to present results of the testing focused on efficiency of CPU performance and its threads in single-threading and multi-threading modes in various versions of operating systems from the family of Microsoft Windows. The main task was to verify whether the chosen operating system version affects the efficiency of using threads by the operating system, with the emphasis of their upgrade in technological and industrial systems.

Keywords: 7-Zip · WinRAR · Cinebench · AIDA64 · Windows Single-threading · Multithreading

1 Introduction

During development of operating systems from the family of Microsoft Windows, older versions are continuously becoming obsolescent and are being replaced by the newer ones. This life cycle is currently affecting one of the most widespread versions of the operating system, Microsoft Windows 7, whose official support was announced to be terminated on 14 January, 2020. Therefore, this version must be replaced by its successors, Windows 8.1 or Windows 10. In terms of common usage of the operating system, this seems to be rather a trivial concern, however, in the area of technological and industrial use of the systems, this step is quite elaborate as it must be weighed from several viewpoints. The first one is considering the life cycle of operating systems in relation to the applications used, as in the area of industrial and technological use, it is paramount to ensure the maximum functionality of the specialized systems for supervisory control, such as SCADA systems. It is also necessary to take into account possible life expectancy and duration of the official support of the operating system itself, in order to moderate the frequency of operating system upgrades; these can cause various compatibility issues, which often cannot be predicted and can disrupt or even stop the technological systems from functioning. Last but not least, it is necessary to take into account the operating system's performance in relation to the specialized industrial supervisory control systems. With regard to the architecture of such control systems frequently programmed as single-threading applications and still in use in many industrial areas in newer installations, systems with modern multithreading architecture are employed.

P. C. Vinh and A. Rakib (Eds.): ICCASA 2020/ICTCC 2020, LNICST 343, pp. 277–287, 2021.
https://doi.org/10.1007/978-3-030-67101-3_22

With regard to the aforementioned state-of-the-art, a survey was realized, with its results being presented in this article. The survey was comprised of several testing sequences aimed at the efficiency of thread usage across the most used versions of the operating systems from the family of Microsoft Windows.

The aim of the tests realized using two different hardware configurations of the systems was to determine whether newer Microsoft Windows operating systems manage the computations more efficiently in single-threading mode, or multithreading mode. It also aimed to determine which version of the operating system would be more suitable to replace current commonly used Microsoft Windows 7 as the interface between hardware and technological and industrial systems such as SCADA.

In the area of analysis and testing of Microsoft Windows operating systems, many researches and surveys have been conducted and published. The one that is closest to this subject matter are tests focused on the usage of main memory [1] and [2]. The closest to the area we have surveyed is the article "A Survey of Main Memory Acquisition and Analysis Techniques for the Windows Operating System" [3, 4]. However, thread usage research is not getting much attention despite being relevant and affecting efficiency and maintaining the consistency not only in the are of technological systems, but also in regular use of information systems.

1.1 Introduction to Threads Principles

As the aim of the article is to present the results of the testing aimed at the CPU performance efficiency and its thread in single-threading and multithreading modes while employing various versions of the operating systems from the family of Microsoft Windows, it is necessary to briefly introduce the basic features of threads. A thread (often called lightweight process – LWP) is an elementary unit of CPU usage containing program pointer, register, and stack [5, 6]. Data part of the processor and allocated OS resources (together forming a task) are shared by all the threads of a process. Regular or heavyweight process is a task with a single thread.

A task does not do anything if no thread is assigned to it, and any single thread can be assigned only to exactly one task. Therefore, switching between threads leads to lowering the CPU workload as compared to more complicated switching of context in complex processes. Switching between threads represents alignment of a register set, and it is not necessary to make changes in memory [7]. Some systems implement user level of threads in user libraries instead of implementation of threads via system calls. The threads implemented in such a way do not, therefore, require help of the OS and do not cause an interrupt. Switching between user threads is not dependent on the OS and is, therefore, very fast [8, 9]. The use of thread interrupting and switching leads to a relevant solution for the server to efficiently handle multiple queries. User level of thread switching, however, also had its disadvantages. If the OS kernel is a single-thread, every kernel calling by a process results in stopping the whole task until the kernel response. On the other hand, in multithread and multiprocessor system, every process has its program pointer, stack, and address space. This organization method is suitable if the processes ran by individual programs in the system are mutually independent. To state an example, single-processor OS at file server is very often stalled by waiting for drive access. The server's performance would be improved if another process could run while

the first one is stalled. However, if other processes want to access the drive or use the same address space (which is not uncommon in file servers), it is impossible to allocate CPU to any processes and the CPU remains unused. If the aforementioned example used multithread architecture or if one task thread was blocked, it would be possible to allocate the CPU to another thread as it is in the same address space. Cooperation of the threads that belong to the same task leads to easier access and higher system performance. As opposed to a process, threads are dependent on each other as all the thread have access to any task address and threads can read or write into any stack of that task's thread. No protection of individual threads is in effect. As opposed to the processes that are ran by different users and can, therefore, behave unfriendly to each other, threads are programmed to help each other, and so any protection of the memory from other threads is unnecessary. In threads, it must be taken into account that if a problem between producer and receiver of the thread requires a shared buffer (as both originate in the same task), to switch between them, not much direction is needed. What is more, in multiprocessor systems, every thread can run on a different processor and the performance is thus maximized. All in all, it should be mentioned that threads on user level do not use kernel and switching between them is, therefore, faster than with the threads supported by the kernel. Any kernel calls stall the whole process and as the kernel plans only processes, it has no information about existence of the threads and the waiting process is not eligible for allocation of CPU [10].

2 Testing Methodology

For the practical testing of thread use efficiency in Microsoft Windows, two PC configurations were chosen (Table 1.). These configurations represent standard laboratory equipment in laboratory of operating systems and computer networks at FIM of UHK. Tested operating systems on the hardware configurations were always installed with up-to-date updates available at the time of the testing. Afterwards, testing software was installed. Every measuring was repeated 30 times. For the testing purposes, 64-bit editions of Windows 7 Professional, Windows 8.1 Professional, and Windows 10 Professional were used.

Table 1. Configuration of tested PC

Components	Configuration 1	Configuration 2
CPU	DualCore Intel Core i3-6100, 3,7 GHz (2 cores, 4 treads)	QuadCore Intel Core i7-3770 K, 3,9G (4 cores, 8 treads)
Motherboards	Asus B150I Pro Gaming/Aura	Asus Maximus V Extreme
Chip set	Intel Sunrise Point B150, Intel Skylake-S	Intel Panther Point Z77, Intel Ivy Bridge
RAM	8121 MB (DDR4 SDRAM)	32708 MB (DDR3 SDRAM)
BIOS	AMI (12/22/2017)	AMI (08/19/2013)
Graphics adapter	GeForce GTX 1050 Ti (4 GB)	GeForce GTX 960 (4 GB)
IDE controller	SATA AHCI	SATA AHCI
Disk drive	Kingston SA400S37120G (120 GB, SATA-III)	Intel SSDSC2CW180A3 (180 GB, SATA-III)

2.1 Presentation of Conducted Tests

For the testing of thread usage efficiency in Microsoft Windows operating systems, third party applications that can be used for testing a tasking in single-threading and multithreading modes, were used. Based upon an in-depth analysis of available utilities, the following utilities were chosen and the tests below were executed.

7-Zip 18.05 [11] is a free open-source software. Most of the code is under the GNU LGPL license. Some parts of the code are under the BSD 3-clause License. For the testing, internal Performance benchmark that tests compression and decompression speeds in kB/s, was used. The testing results showcase average value of compression/decompression in MIPS (Million Instructions per Second). Higher MIPS means better results. The tests were run using single (Single-Threading) and all available CPU threads (Multithreading). The testing was repeated thirty times, using 128 MB dictionary size.

WinRAR 5.60 [12] contains a built-in benchmark whose functionality is very similar to those of **7-Zip 18.05**. However, speed is measured in kB/s. The tests were run using single (Single-Threading) and all available CPU threads (Multithreading). The testing was repeated thirty times, using 128 MB dictionary size.

CPU-Z 1.85.0 [13] is a freeware that gathers information on some of the main system devices. For our testing, a benchmark to test single-threading and multithreading was used. The testing results are measured in points, with more points meaning better results. The testing was repeated thirty times.

Cinebench R15.038_RC184115 [14] uses graphical rendering for testing the CPU in single-threading and multithreading modes. The testing results are measured in points, with more points meaning better results. The testing was repeated thirty times.

AIDA64 CPU Tests [15] use all available threads. Simultaneous multithreading and Hyper-threading were enabled. For the testing, CPU Queen module, which is an integer benchmark specialized in processor prediction and penalties of the CPU, was used. It finds the solutions for the classic "Queens problem" on a 10 by 10 sized chessboard [16]. CPU Zlib measures the performance of processor subsystem and memory combined using the public library Zlib for compression. Furthermore, data encryption speed using CPU AES was tested. The last test was performed using AES CPU Hash, which measured the CPU performance using SHA1 algorithm.

3 Testing Results and Discussion

The gathered data was processed using standard descriptive statistical methods. Arithmetic mean, Variance, and Standard deviation were determined so the gathered data could be relevantly evaluated.

3.1 7-Zip 18.05 Test

The testing results infer that all the tested operating system versions can distribute the workload on individual threads, which is seen while comparing the MIPS results at two QuadCore Intel Core i7-3770K CPUs. Individual measurements have low Variance

with maximum deviation of 1.4% from the average, and therefore, the gathered data show high congruity. For the data measured in single-threading, Windows 7 PRO had the best results for both configurations, although deviation among other versions of Microsoft Windows operating system was still below 1.4%, which is negligible for pragmatic evaluation. In multithreading mode, for Configuration 1, the best results were achieved by Windows 10 PRO operating system, and for Configuration 2, Windows 7 PRO. Respective deviations were below 1.2%, which is negligible. Full results are showcased in Table 2.

Table 2. Overall performance (MIPS) for 7-Zip 18.05

Single-threading			
Configuration 1	Arithmetic mean (MIPS)	Variance σ^2	Standard deviation σ
Windows 7 PRO	5463,5	162,6	12,8
Windows 8.1 PRO	5407,2	228,4	15,1
Windows 10 PRO	4740,7	820,5	28,6
Configuration 2	Arithmetic mean (MIPS)	Variance σ	Standard deviation σ
Windows 7 PRO	5386,4	76,6	8,8
Windows 8.1 PRO	5369,7	451,2	21,2
Windows 10 PRO	5327,0	102,8	10,1
Multithreading			
Configuration 1	Arithmetic mean (MIPS)	Variance σ^2	Standard deviation σ
Windows 7 PRO	14552,3	1510,6	38,9
Windows 8.1 PRO	14422,6	732,2	27,1
Windows 10 PRO	14648,9	369,0	19,2
Configuration 2	Arithmetic mean (MIPS)	Variance σ^2	Standard deviation σ
Windows 7 PRO	25896,6	4585,0	67,7
Windows 8.1 PRO	25661,6	8299,0	91,1
Windows 10 PRO	25669,2	3395,8	184,3

3.2 Test WinRAR 5.60

The results of this test are similar to the first one, except for that while using multi-threading in Configuration 1, Windows 8.1 had the best results. Maximum deviation of individual operating system versions was 7%, which was measured in Configuration 1 multithreading. The data alone is the measured with minimum variance, which is below 1%, therefore is considered as measurement error. Full results are showcased in Table 3.

Table 3. Overall performance (kB/s) for WinRAR 5.60

Single-threading			
Configuration 1	Arithmetic mean (kB/s)	Variance σ^2	Standard deviation σ
Windows 7 PRO	1523,1	2,1	1,5
Windows 8.1 PRO	1517,4	1,3	1,2
Windows 10 PRO	1515,4	1,8	1,4
Configuration 2	Arithmetic mean (kB/s)	Variance σ^2	Standard deviation σ
Windows 7 PRO	1557,8	5,5	2,3
Windows 8.1 PRO	1556,0	11,5	3,4
Windows 10 PRO	1515,0	23,0	4,8
Multithreading			
Configuration 1	Arithmetic mean (kB/s)	Variance σ^2	Standard deviation σ
Windows 7 PRO	4585,8	454,0	21,3
Windows 8.1 PRO	4971,4	33,0	5,7
Windows 10 PRO	4943,8	42,5	6,5
Configuration 2	Arithmetic mean (kB/s)	Variance σ^2	Standard deviation σ
Windows 7 PRO	9548,0	76,4	8,7
Windows 8.1 PRO	9524,3	28,3	5,3
Windows 10 PRO	9428,0	68,6	8,3

3.3 Test CPU-Z 1.85.0 64-Bit

The results while using CPU-Z are different from the others, especially considering their accuracy while using Configuration 1 while measuring in single-threading mode, where the deviations reached value of up to 6%. However, with Configuration 2, the value was below 1% in both single-threading and multithreading. Therefore, it can be inferred that the performance of DualCore Intel Core i3-6100 in single-threading mode is greatly dependent on the type of the calculations used in the given test. Overall, the best values were achieved by Windows 7 and Windows 8.1. Full results are showcased in Table 4.

3.4 Test Cinebench R15.038_RC184115

The results of the tests while using Cinebench R15.038_RC184115 show generally the most stable results not only among individual operating system versions, but also in both single-threading and multithreading modes. For both configurations and modes, the best results were achieved by Windows 7, although compared to the other versions of the operating system, results were below 1%, which is a negligible deviation and can be considered as a measurement error. Full results are showcased in Table 5.

Table 4. CPU-Z 1.85.0 64-bit

Single-threading			
Configuration 1	Arithmetic mean (points)	Variance σ^2	Standard deviation σ
Windows 7 PRO	355,4	141,8	11,9
Windows 8.1 PRO	386,1	194,6	14,0
Windows 10 PRO	357,4	99,0	9,9
Configuration 2	Arithmetic mean (points)	Variance σ^2	Standard deviation σ
Windows 7 PRO	403,0	0,2	0,5
Windows 8.1 PRO	402,9	0,4	0,6
Windows 10 PRO	383,8	3,3	1,8
Multithreading			
Configuration 1	Arithmetic mean (points)	Variance σ^2	Standard deviation σ
Windows 7 PRO	1104,8	0,4	0,6
Windows 8.1 PRO	1102,5	2,3	1,5
Windows 10 PRO	1049,4	0,9	1,0
Configuration 2	Arithmetic mean (points)	Variance σ^2	Standard deviation σ
Windows 7 PRO	1968,8	127,9	11,3
Windows 8.1 PRO	1974,9	0,2	0,4
Windows 10 PRO	1814,3	73,1	8,6

Table 5. Cinebench R15.038_RC184115

Single-Threading			
Configuration 1	Arithmetic mean (points)	Variance σ^2	Standard deviation σ
Windows 7 PRO	160,0	0,0	0,0
Windows 8.1 PRO	158,6	0,2	0,5
Windows 10 PRO	157,3	3,3	1,8
Configuration 2	Arithmetic mean (points)	Variance σ^2	Standard deviation σ
Windows 7 PRO	145,0	0,0	0,0
Windows 8.1 PRO	144,4	0,2	0,5
Windows 10 PRO	143,5	0,3	0,5
Multithreading			
Configuration 1	Arithmetic mean (points)	Variance σ^2	Standard deviation σ
Windows 7 PRO	407,2	1,5	1,2
Windows 8.1 PRO	405,4	0,4	0,7
Windows 10 PRO	394,2	5,1	2,2
Configuration 2	Arithmetic mean (points)	Variance σ^2	Standard deviation σ
Windows 7 PRO	719,6	2,4	1,6
Windows 8.1 PRO	718,9	2,3	1,5
Windows 10 PRO	711,6	9,4	3,1

3.5 Test AIDA64 CPU Tests

The last performed tests were aimed at comparison of the CPUs working in multi-threading mode. To perform these tests, AIDA64 CPU Tests utility was used along with CPU Queen module, which is a benchmark specialized in processor prediction and penalties of the CPU. It finds the solutions for the classic "Queens problem" on a 10 by 10 sized chessboard [16]. CPU Zlib measures the performance of processor subsystem and memory combined using the public library Zlib for compression. Furthermore, data encryption speed using CPU AES was tested. The last test was performed using AES CPU Hash, which measured the CPU performance using SHA1 algorithm.

The results showcased in Tables 6 and 7 show that performance and work efficiency are almost unaffected by the chosen operating system and that the individual results vary by 1% at maximum. Throughout their iterations, the individual tests show minimum variance and standard deviation and point at the performance of individual computing system configurations being stable and not showing any significant deviations.

Table 6. AIDA64 CPU tests multithreading – configuration 1

Configuration 1	Test	Arithmetic mean	Variance σ^2	Standard deviation σ
Windows 7 PRO	CPU Queen (points)	25 258,9	225,73	15,02
	CPU Zlib (MB/s)	166,0	0,37	0,60
	CPU AES (MB/s)	8 420,1	0,45	0,67
	CPU Hash (MB/s)	2 128,0	45,25	6,73
Windows 8.1 PRO	CPU Queen (points)	25 218,4	133,43	11,55
	CPU Zlib (MB/s)	167,8	0,35	0,59
	CPU AES (MB/s)	8 419,7	0,23	0,48
	CPU Hash (MB/s)	2 117,2	1,16	1,08
Windows 10 PRO	CPU Queen (points)	25 163,5	295,65	54,32
	CPU Zlib (MB/s)	167,8	0,28	0,53
	CPU AES (MB/s)	8 414,9	0,83	0,91
	CPU Hash (MB/s)	2 116,6	0,25	0,50

Table 7. AIDA64 CPU tests multithreading – configuration 2

Configuration 2	Test	Arithmetic mean	Variance σ^2	Standard deviation σ
Windows 7 PRO	CPU Queen (points)	49 461,5	75,75	8,70
	CPU Zlib (MB/s)	331,9	4,03	2,01
	CPU AES (MB/s)	15 283,7	5,51	2,35
	CPU Hash (MB/s)	3 167,9	10,83	3,29
Windows 8.1 PRO	CPU Queen (points)	49 469,7	95,43	9,77
	CPU Zlib (MB/s)	332,5	0,98	0,99
	CPU AES (MB/s)	15 279,2	6,06	2,46
	CPU Hash (MB/s)	3 163,0	0,50	0,71
Windows 10 PRO	CPU Queen (points)	49 393,2	63,33	25,25
	CPU Zlib (MB/s)	335,4	2,89	1,70
	CPU AES (MB/s)	15 245,4	65,34	8,08
	CPU Hash (MB/s)	3 157,6	0,25	0,50

4 Conclusion

The aim of the article was to find the answer as to which version of Microsoft Windows operating system would be the most suitable to replace the obsolescent, yet widespread, Windows 7. The analysis and the testing were aimed at performance of Windows 8.1 and Windows 10 in processing and using CPU threads from the viewpoint of the used operating system version. For the purposes of the analysis, five different utilities were used. The utilities served to survey the operation execution in single-threading and multithreading modes. Individual tests were evaluated using chosen descriptive statistics methods in order to eliminate random and systematic measurement errors. From the gathered data it is unequivocally clear that all the three Microsoft Windows versions had shown minimum deviations, and it can be, therefore, stated that thread usage efficiency in both single-threading and multithreading modes is same in all the tested systems, with maximum deviation being below 3% with the exception of the CPU-Z test, where deviation reaching up to 6% was measured while using Configuration 1 in single-threading mode.

It can be, therefore, ascertained that currently used Windows 7 systems can be substituted by Windows 8.1 or Windows 10 as their efficiency of thread usage by the CPU remains unchanged. As for the employment of the operating systems in

technological and industrial systems, taking into account the aforementioned long-term life cycle of system replacement, it is more suitable to employ Windows 10 Pro as its life cycle is fully open and without any announced terms of the official support termination.

Acknowledgment. This work and the contribution were supported by a Specic Research Project, Faculty of Informatics and Management, University of Hradec Kralove, Czech Republic. We would like to thank Mr. J. Spacek, a graduate of Faculty of management and informatics, University of Hradec Kralove, for the practical verification of the proposed solutions and close cooperation in the solution.

References

1. Okolica, J., Peterson, G.L.: Windows operating systems agnostic memory analysis. Digit. Invest. **7**, S48–S56 (2010). https://doi.org/10.1016/j.diin.2010.05.007. ISSN 17422876. Accessed 27 Feb 2019
2. Stüttgen, J., Cohen, M.: Anti-forensic resilient memory acquisition. Digit. Invest. **10**, S105–S115 (2013). https://doi.org/10.1016/j.diin.2013.06.012. ISSN 17422876. Accessed 27 Feb 2019
3. Vömel, S., Freiling, F.C.: A survey of main memory acquisition and analysis techniques for the windows operating system. Digital Investigation **8**(1), 3–22 (2011). https://doi.org/10.1016/j.diin.2011.06.002. ISSN 17422876. Accessed 27 Feb 2019
4. Skaletsky, A., et al.: Dynamic program analysis of Microsoft Windows applications. In: 2010 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS), White Plains, NY, pp. 2–12 (2010). https://doi.org/10.1109/ISPASS.2010.5452079
5. Eich, M., Vögele, T.,: Design and control of a lightweight magnetic climbing robot for vessel inspection. In: 2011 19th Mediterranean Conference on Control and Automation (MED), Corfu, pp. 1200–1205 (2011). https://doi.org/10.1109/med.2011.5983075
6. Kwon, H., Kim, T., Yu, S.J., Kim, H.K.: Self-similarity based lightweight intrusion detection method for cloud computing. In: Nguyen, N.T., Kim, C.-G., Janiak, A. (eds.) ACIIDS 2011. LNCS (LNAI), vol. 6592, pp. 353–362. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-20042-7_36
7. Anders Larsson; AliView: a fast and lightweight alignment viewer and editor for large datasets. Bioinformatics **30**(22), 3276–3278 (2014). https://doi.org/10.1093/bioinformatics/btu531
8. Tanchaisawat, T., Bergado, D.T., Voottipruex, P.: Numerical simulation and sensitivity analyses of full-scale test embankment with reinforced lightweight geomaterials on soft Bangkok clay. Geotext. Geomembr. **26**(6), 498–511 (2008). https://doi.org/10.1016/j.geotexmem. ISSN 0266-1144
9. Ambrogio, G., Filice, L., Gagliardi, F.: Formability of lightweight alloys by hot incremental sheet forming. Mater. Des. **34**, 501–508 (2012). https://doi.org/10.1016/j.matdes.2011.08.024. ISSN 0261-3069
10. Cheng, Y., Fu, X., Du, X., Luo, B., Guizani, M.: A lightweight live memory forensic approach based on hardware virtualization. Inf. Sci. **379**, 23–41 (2017). https://doi.org/10.1016/j.ins.2016.07.019. ISSN 0020-0255
11. 7-Zip. 7-Zip (2018). Dostupné z. https://www.7-zip.org/. Accessed 25 Feb 2019

12. RARLAB. RARLAB products (2019). Dostupné z. https://www.rarlab.com/. Accessed 25 Feb 2019
13. CPU-Z: System information software. cpuid (2019). Dostupné z. https://www.cpuid.com/softwares/cpu-z.html. Accessed 25 Feb 2019
14. CINEBENCH [online]. MAXON Compute (2019). Dostupné z. https://www.maxon.net/en/products/cinebench/. Accessed 25 Feb 2019
15. AIDA64 User manual. ABSEIRA (2019). Dostupné z. https://www.aida64.co.uk/user-manual/aida64-user-manual. Accessed 25 Feb 2019
16. Queens Problem: Wolfram Math World. Wolfram Research (2019). Dostupné z. http://mathworld.wolfram.com/QueensProblem.html. Accessed 25 Feb 2019
17. Nurmuhumatovich, J.A., Mikusova, M.: Testing trajectory of road trains with program complexes. Arch. Automot. Eng. – Arch. Motoryzacji **83**(1), 103–112 (2019). https://doi.org/10.14669/AM.VOL83.ART7