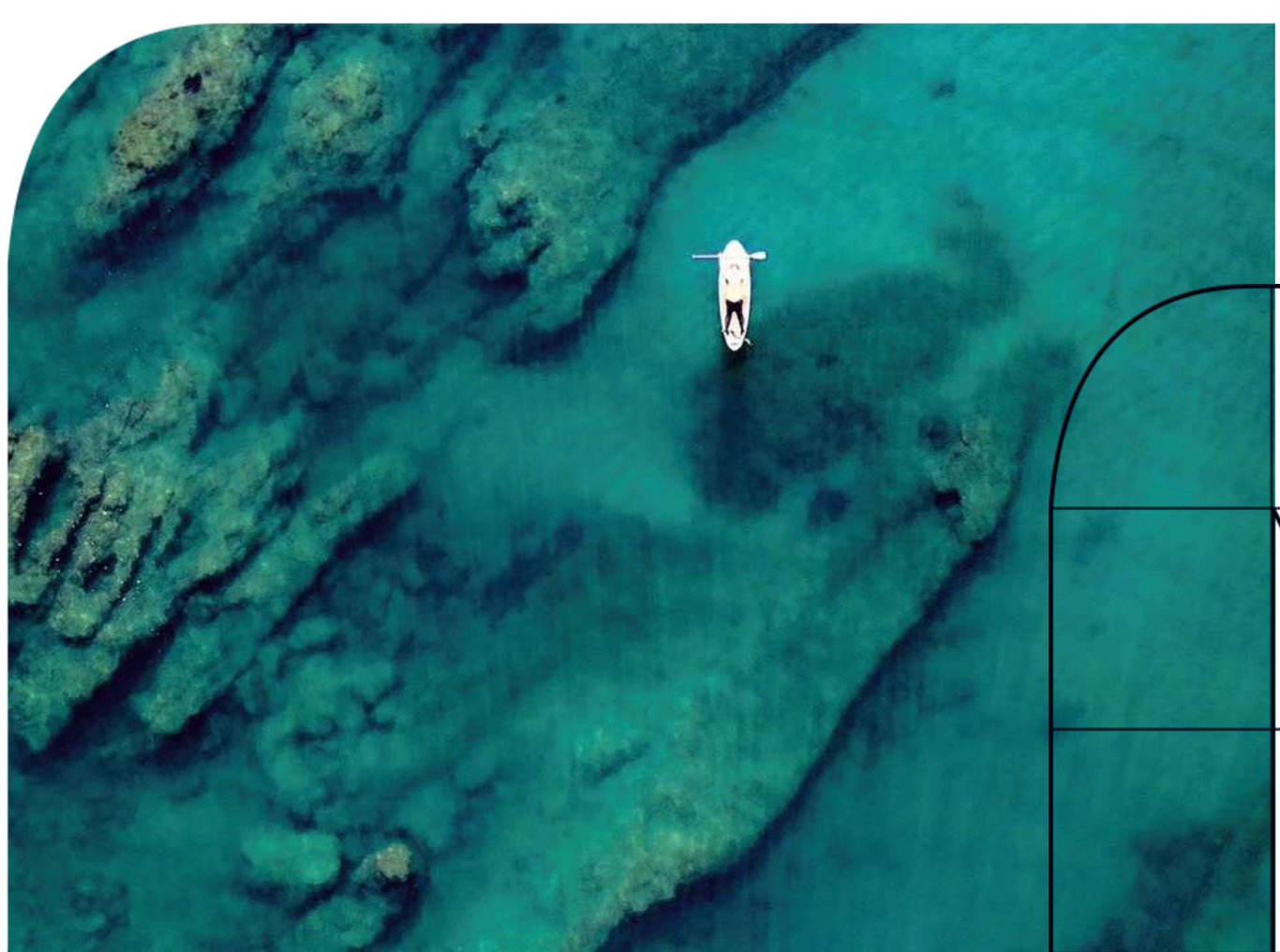
管理番号：HPERL001

Power Efficiency of Java on Arm-based server

Advantages of Arm-based servers in power consumption with Java application



Context

Sustainability has become a major priority for modern organizations, and they are increasingly transparent regarding their efforts and results to shareholders and stakeholders. In particular, a company's environmental impact is one of the major indicators, and activities toward decarbonization are accelerating in various places - "green" is now a key component of any decision making.

IT infrastructure and peripheral equipment continue to become increasingly important in companies' business operations across virtually any industry. Though the equipment may not directly emit carbon dioxide (CO_2), the electricity required to power them often does require CO_2 to be generated during its production. In order to reduce carbon emissions, it is of course necessary to be aware of the source of this electricity (renewable, fossil fuels, nuclear) but the most important thing is to "reduce power consumption". HPE's ProLiant RL300 with Ampere® Altra® family processors is designed exactly for that purpose.

Behind the Ampere Altra® CPUs is Arm64 (also known as Aarch64) architecture, which has long been adopted in fields where power saving is essential, such as smartphones and embedded devices. More recently, its application range has been expanding to compute infrastructure, even including the HPC (High Performance Computing) field*1. Experts tout the high power-performance ratio of Arm architecture, especially Ampere's cloud native processor which excels in various enterprise applications and benchmarks, including Java-based applications.

Since many customers use systems developed in Java at NTT DATA, moving Java applications to sustainable infrastructure would contribute not only to our company's pledge toward environmentally friendly business operations but also customers sustainability. Therefore, it was decided to verify the impact that the RL300 Arm-based server can have on NTT Data's goals toward energy savings in enterprise Java applications.

Modus Operandi

To verify the power savings of Arm server, the same application was run on it and x86 server (featuring an Intel processor with x86-64 architecture). The processors in both machines provide similar application performance (Table 1), so our aim was to characterize the power consumption of both systems. The x86 based system features 2 CPUs with 28 cores (56 threads with Hyper-Threading) each, whereas the Ampere-based system features 1 CPU with 128 cores (128 threads due to single-threaded).

The following things were done:

- SysBench CPU Benchmark
- Java Application (RESTful Web Service: ATRS *2)

Table 1. Benchmark score of processors on each servers

Server Name	CPU	Architecture	SPECrate2017 int_base
x86	Intel Xeon Gold 6330 x2 sockets	x86-64	359
Arm	Ampere M128-30 x1 sockets	Arm64	366

*1 https://www.arm.com/markets/computing-infrastructure
*2 https://github.com/Macchinetta/atrs

Power Saving Settings which are listed in Table 2 and Table 3 were used.

CPU benchmark in SysBench searches prime number in parallel, in this case look for prime numbers below 10,000. Server power consumption was measured while executing the search for 60 seconds. However, since the number of logical cores is different between Arm server and x86 server, two patterns were executed on both servers: when the number of threads was specified according to x86 server (112 threads) and when the number was specified according to Arm server (128 threads). They are manufactured by HPE, and they can be managed by HPE Integrated Lights-Out (iLO), featuring a Redfish-compliant API, the industry standard for server management. By calling Redfish API, various information on the server including power consumption can be obtained. It can be determined by looking at the *PowerConsumedWatts* value of the *PowerControl* item included in the response of *PowerResource* Information. Therefore, no special equipment was needed to validate system-level power consumption.

Table 2. Power Saving Settings on x86 server

Option	Setting
Power Regulator	Dynamic Power Savings Mode
Minimum Processor Idle Power Core C-State	C6 State
Minimum Processor Idle Power Package C-State	Package C6 (non-retention) State
Intel® Turbo Boost Technology	Enabled
Energy Performance Preference	Disabled
Energy Performance Bias	Balanced Performance
Collaborative Power Control	Enabled
Intel DMI Link Frequency	Auto
NUMA Group Size Optimization	Flat
Intel Performance Monitoring Support	Disabled
Uncore Frequency Scaling	Auto
Sub-NUMA Clustering	Disabled
Energy Efficient Turbo	Enabled
Local/Remote Threshold	Auto
LLC Dead Line Allocation	Enabled
State A to S	Disabled

Table 3. Power Saving Settings on Arm-based server

Option	Setting
APEI Suppport	Enabled
CPPC Support	Enabled
LPI Support	Enabled

Results

SysBench CPU benchmark

Figure 1 shows a comparison of processing performance in prime number search, measured in events per second. Arm server performed roughly three times more calculations than x86 server. Figure 2 shows the power consumption of each test pattern, but regardless of the number of threads, Arm server consumes less power. In the case of 128 threads (matching the number of logical cores of Arm server), the power consumption of Arm server was 350 W, while x86 server consumed 490 W – which is 140W or 40% higher. More importantly, Arm server can generate nearly 4.8 times the performance per watt as the competitive x86 server (Arm server processed 456806 events per second and used 350 W. 456806 / 350 = 1305 events/s per watt. x86 server processed 130842 events per second and used 490 W. 130842 / 490 = 267 events/s per watt. 1305 / 267 ≒ 4.8). Thus, NTT DATA can confirm the initial hypothesis that Arm servers can deliver significantly higher processing performance while minimizing power consumption.

Java Application

Next, the power consumption of the chassis was measured running the same Java application and the same amount of load applied over the same period of time. This application assumes a typical web application for enterprises, retrieves data from a relational database based on the parameters included in the request, and returns it as a response in JSON format. The operating system used was Rocky Linux 8.7 and Java version was 19.0.2. No option, including the number of threads, is specified to tune the Java Virtual Machine (JVM) when the application is executed. It means JVM would compute number of threads from number of logical processors. The application was loaded from another server by running the load testing tool Gatling. The load was applied for 15 minutes, where the number of users gradually increased for the first 5 minutes until the maximum was reached. After 5 minutes, and the load was continued to be applied with the same number of users for 10 minutes.

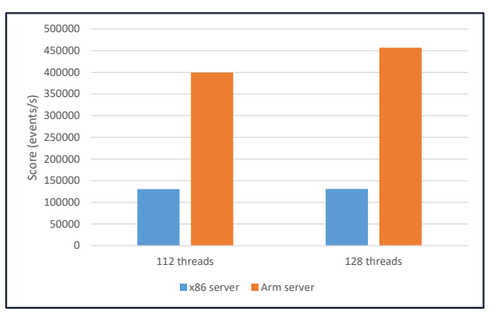


Figure 1. SysBench measurement results

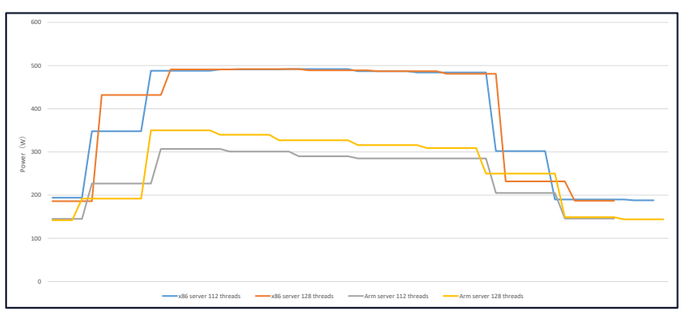


Figure 2. Power consumption during SysBench

During that time, Redfish API was called every second and the power consumption was recorded. Since Java is an architecture running programs in JVM, as long as the corresponding JDK (Java SE Development Kit) is used, the application can run in any environment without any change or recompiling. JDK has long supported not only x86-64 but also 64-bit Arm processors (AArch64). It was possible to run the application on Arm without additional work. The measurement results are shown in figure 3.

Looking at the period of stable power consumption in the graph, x86 server consumed 380 W, while Arm server only consumed 240 W. Arm server consumed 140W less power, similar to the CPU benchmark. Since both Arm server and x86 server are primarily running only verification Java applications, it can be assumed that running on Arm server consumes about 37% less power in the application. The results proved the high power-performance ratio of the Arm architecture. What could be more impressive is that from idle (145W) to full load (240W), the Ampere CPU only added 95W whereas x86 changed from 195W to 380W (185W) so to do the same work, Ampere consumes half at the power.

Considering that almost all web applications run continuously day and night, the power savings validated during our tests become even more impactful. Since the power consumption of the executing hardware is one of the major factors in the power consumption of Web applications, we believe that it is a fruitful step in promoting power saving in the future to find that such a large power saving effect can be obtained simply by changing the server to Arm server. This is especially beneficial for Java-based applications because they can be moved to different servers without any changes or recompilation.

NTT DATA tested each server by specifying various Java runtime options, all of which were run during this study without showing significant differences in power consumption. The reason for this is that if Java is sufficiently optimized, there will be no significant difference in terms of hardware usage, and therefore there will be virtually no effect on power consumption. Similar results can be achieved not only for Java applications but also for applications written in other programming languages, though exact performance and efficiency improvements are highly dependent on the implementation maturity of software and tools (including runtimes and compilers) with respect to Arm64. In recent years, Java has been actively supported and improved with Arm64 in OpenJDK, which is an open-source implementation, so the full magnitude of the power saving effect of Arm server can be achieved without worry. In the case of other languages and software, the migration process and results can vary.

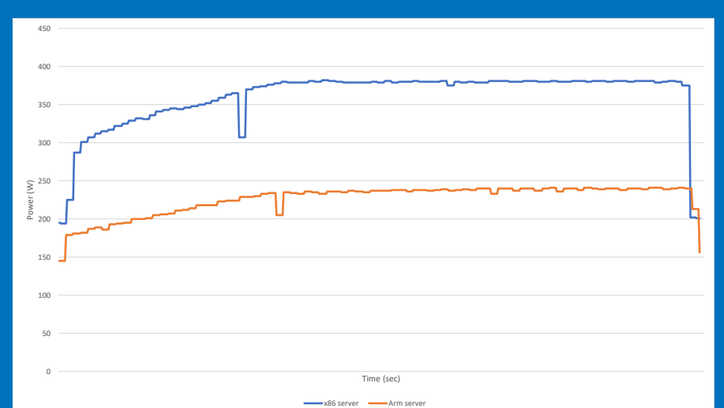


Figure 3. Power consumption during Java application

Conclusion

Verification – impact on CO_2 emission reduction

The potential CO_2 emission reduction over 3 years by replacing x86 servers by Arm-based servers was estimated.

Assumptions

- One rack can hold 21 x86 servers. We assume those 21 x86 servers replace to Arm servers.
- Servers operate 24 hours a day, 365 days a year.
- Electricity charges and CO_2 emission factors are as follows.
  - Electricity charge (JPY/kWh) :
    Referred by TEPCO's rate averaging the difference from seasons and time zone of a day (See Reference #2)
  - CO_2 Emission Factor (t-CO_2/kWh):
    Referred by CO_2 Emission coefficient which TEPCO provides on their web (See Reference #3).

Results

Replacing x86 servers with Arm servers can reduce annual CO_2 emissions by about 10t-CO_2 and reduce CO_2 emissions by about 30t-CO_2 after 3 years of use.

CO_2 Emission per Rack (kg-CO_2/Rack)			
Year	x86 Server	Arm Server	Savings
1yr	26,350	16,601	9,750
2yr	52,700	33,201	19,499
3yr	79,050	49,802	29,249

Conclusion

The word "green application" first comes to mind about power saving by improving processing efficiency in terms of software such as algorithms. However, the program is ultimately executed on hardware. So both software and hardware should be considered to avoid local power saving. Arm server will make a solid contribution to hardware power savings. From the results of NTT DATA's verification, the use cases for Arm server can be expanded not only to HPC but also to the enterprise domain. Ampere Altra® and Altra® Max processors used in the RL300 are specifically designed to improve performance and efficiency for cloud workloads, but given the findings, the RL300 will be a critical building block to improve performance and reduce carbon footprint of IaaS, PaaS, and SaaS offerings. In terms of combining performance and power saving, Arm servers are to be used in a wider range of areas.

References

1. Introduction of HPE ProLiant RL300 Gen11
2. UPL in TEPCO (Japanese)
3. Carbon Intensity in TEPCO (Japanese)