

Linux on the Mainframe: When Sort Moves Next Door

*What an IFL does to the software bill, how the data gets across,
and what EBCDIC means once it gets there*

Bill Ahlbrandt

President and CEO, Ahlbrandt Software

Abstract

Every IBM Z machine that runs z/OS can also have Integrated Facility for Linux (IFL) processors, and IBM's LinuxONE servers have nothing else. Work that runs on an IFL doesn't count toward z/OS software charges, so Linux on the same machine looks like an obvious home for batch sort. It's close to the data, it's off the four-hour average and it's the same processor architecture. This paper walks through what an IFL is and how it's charged, how data gets from z/OS to Linux on the same machine and what IBM publishes about each path, and what happens when mainframe records land in an operating system that's big-endian like z/OS but uses ASCII. The short version: the processor economics are the easy part. What decides a pilot is the transfer, which still runs on z/OS, and proving that the Linux sort produces exactly the output the z/OS sort did.

What I found

- IBM says an IFL "does not increase charges for IBM Z software running on 'standard' processors, nor does it affect the MSU rating." Sort CPU moved to an IFL is out of the z/OS **four-hour rolling average** entirely.
- The work of getting the data there and back stays behind. Unload, transmission and reload run in z/OS, mostly on general-purpose processors, and often in the same batch window. **The transfer is the cost to measure.**
- Only IFLs in the **same machine** as z/OS get memory-speed paths to it, like HiperSockets and SMC-D. For moving data, a separate LinuxONE server is just another server on the network.
- Linux on IBM Z is big-endian like z/OS, so binary fields keep their byte order. But it's an ASCII system, and EBCDIC sorts lower case before upper case and letters before digits. **If you convert first, the output comes out in a different order.**
- A pilot can settle it in a few weeks. Measure how much sort adds to the peak window, what the transfer costs on z/OS and whether the output is byte-for-byte identical, all before you count any savings.

1. The machine next door

In the first paper in this series I argued that the software cost of a sort step on z/OS depends more on when it runs and which pricing model you're on than on how much CPU it uses.¹ Under sub-capacity pricing, IBM's Monthly License Charges follow the highest rolling four-hour average of MSUs used in the month.² Work outside that window costs nothing extra, and work inside it can be expensive. That paper compared four options: rescheduling, the on-chip sort accelerator, specialty engines and moving the work. This one looks hard at one of them, running the sort in Linux on IFL processors inside the same physical machine as z/OS.

This idea isn't new to me. In the 1980s I was a systems programmer at Acxiom, which ran seventeen IBM mainframes, and we built what we called the Attached Processor: jobs stayed on MVS, which did the disk and tape I/O, while the CPU-heavy work ran on a cheaper machine alongside.

On paper the option looks good for three reasons. The data doesn't leave the building, or even the machine. The hardware is the same, so the byte order of binary data doesn't change. And the processor running the sort is outside the z/OS software bill. All three are true. Each one also has a catch for sort.

2. What an IFL is, and how it's charged

An IFL is a physical processor core that IBM firmware sets up to run Linux, and the hypervisors that host Linux, and nothing else. IBM puts it plainly: the IFL is "a processor dedicated to run Linux workloads on IBM Z and IBM LinuxONE systems. It is an optional feature, attractively priced".³ Linux came to the mainframe at the end of 1999, when IBM published its patches for the 2.2 kernel, and IFLs were announced in 2000.⁴ Linux runs on IFLs in its own logical partition, or as guests of z/VM or KVM. The Linux distributions certified for the platform are Red Hat Enterprise Linux, SUSE Linux Enterprise Server and Ubuntu.^{3,5}

The commercial part is what matters here. IBM's product page says an IFL "does not increase charges for IBM Z software running on 'standard' processors, nor does it affect the MSU rating or the IBM Z systems model designation."⁵ Adding IFLs doesn't raise the capacity number z/OS software is rated on, and Linux work on them doesn't show up in the z/OS consumption data the four-hour peak comes from. Where other vendors price z/OS products off the same MSU numbers, the same goes for their charges.

The zIIP is a partial comparison. IBM "does not generally assess IBM software charges on zIIP capacity",⁶ but a zIIP only runs work that z/OS sends it, and whether a sort qualifies depends on the sort product. As the first paper noted, DFSORT doesn't use zIIPs except for Db2 utility work.¹ An IFL has no eligibility test. Whatever runs in Linux runs there in full.

An IFL isn't free

The IFL is off the z/OS bill. There are other bills. The processor itself is hardware you buy, with maintenance, and IBM doesn't publish the price. Software on Linux is licensed on its own terms, and on IBM Z that's usually per core, where an IFL counts as one core. IBM's Linux FAQ says z/VM and most IBM Linux middleware are "priced per processor (core)";³ IBM's processor value unit rules say each IFL "is equivalent to 1 core";⁷ Red Hat licenses Enterprise Linux on IBM Z by the cores actually used, which it calls sub-capacity entitlement,⁸ and says that for OpenShift on IBM Z "1 Integrated Facility for Linux (IFL) requires 1 OpenShift core subscription."⁹ A sort engine on Linux will have its own license too, usually on a similar basis. You have to set all of these against the z/OS savings.

3. LinuxONE: the same engine in a Linux-only machine

In August 2015 IBM launched LinuxONE, mainframes configured with nothing but IFLs, in two models: Emperor, based on the z13, and the entry-level Rockhopper.¹⁰ The line has followed each IBM Z generation since: Emperor II in 2017, with up to 170 cores;¹¹ LinuxONE III with the z15 in September 2019;¹² Emperor 4 in September 2022,¹³ with up to 200 cores;¹⁴ and Rockhopper 4 in May 2023.¹⁵

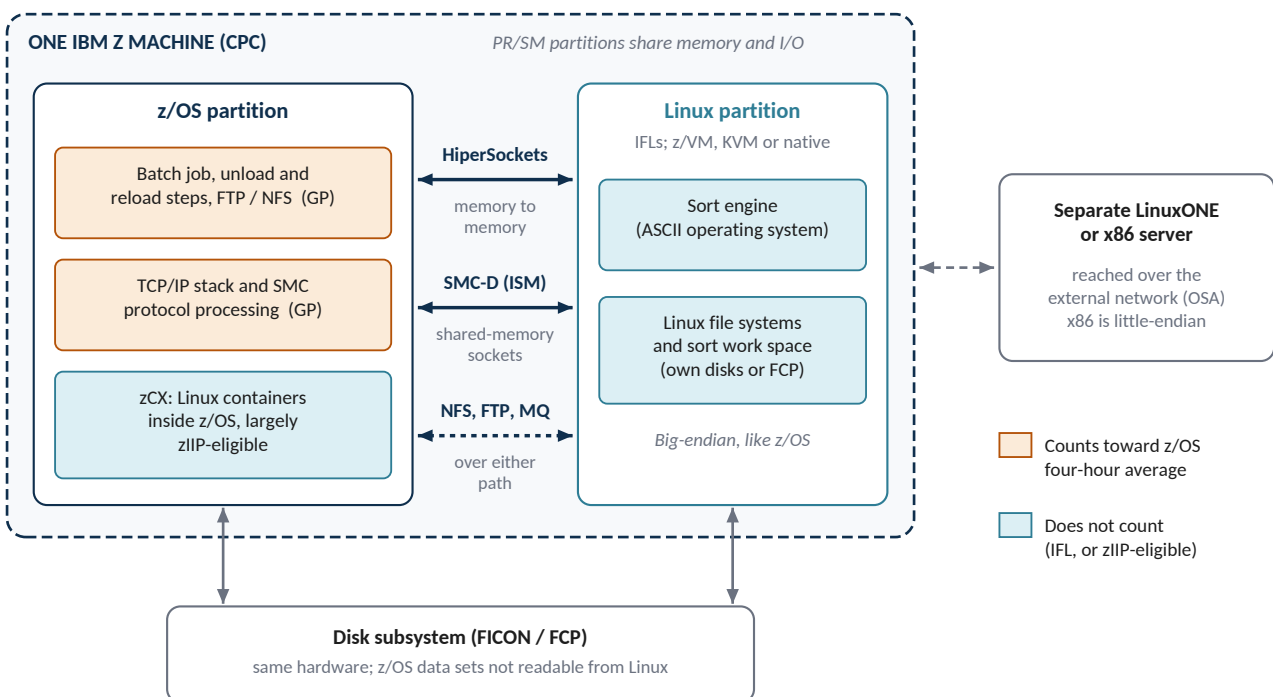
The current generation, LinuxONE Emperor 5, was announced on May 6, 2025, on the Telum II processor used in the z17.¹⁶ Its largest model, like the z17's, has up to 208 configurable cores, and IBM's IFL page cites up to 208 IFLs and 64 TB of memory.^{5,17} Rockhopper 5 and a rack-mountable LinuxONE 5 Express followed in July 2026.¹⁸ IBM sells the line on consolidation. For Emperor 5 it claims up to 44 percent lower five-year total cost of ownership against an x86 setup of 23 servers running a containerized transaction workload;¹⁹ for Emperor 4, 75 percent less energy for five systems replacing 192 x86 servers with 10,364 cores.¹³ Those are vendor numbers for specific configurations.

For sort there's a distinction the marketing doesn't make. A LinuxONE server has no z/OS on it. Its IFLs are the same engines as the IFLs in a z/OS machine, but data held under z/OS gets to them over the external network, exactly as it would get to an x86 server. The memory-speed paths in the next section only exist between partitions of **one** machine. If you want "sort next door" in the physical sense, you need IFLs in the z/OS machine itself. For this purpose a LinuxONE server is a well-built neighbor across the street.

4. How the data gets across

Partitions on the same machine share physical memory, managed by the PR/SM hypervisor, and IBM gives you two ways to use it for communication. **HiperSockets** is an internal TCP/IP network built in firmware, "with the communication path in system memory and the transfer of information between the virtual servers at memory speed", and it works between z/OS, Linux, z/VM and KVM images in any combination.²⁰ **SMC-D** (Shared Memory Communications – Direct Memory Access) goes further. It uses Internal Shared Memory to move data between TCP socket endpoints on the same machine while the application still sees ordinary sockets.²⁰ IBM supports it in both z/OS and Linux on IBM Z.²¹ Exhibit 1 shows the paths.

Exhibit 1. Data paths between z/OS and Linux on one IBM Z machine



Schematic, not to scale. HiperSockets and SMC-D only exist between partitions of the same machine. The z/OS end of every path runs in z/OS and is charged as z/OS work. Sources: IBM Z Connectivity Handbook; z/OS Container Extensions Guide.^{20, 23}

IBM has published comparisons of SMC-D with HiperSockets. For streaming workloads it reports up to 89 percent lower latency, up to nine times the throughput and up to 89 percent lower network-related CPU cost. For small request-and-response exchanges the gains are smaller, up to 48 percent lower latency and 47 percent lower CPU.²² Those were IBM lab measurements between z/OS systems on z13-generation hardware, and they only cover the network layer. Reading, converting and writing the data aren't in them. For z/OS to Linux on current hardware, measure it yourself.

Either way, something still has to carry the data: file transfer (FTP or a managed transfer product), the z/OS NFS server, which lets a Linux client mount z/OS data sets, a message queue, or a program that streams records through a socket straight into the sort. That last one saves landing the data twice. Shared disk isn't the shortcut it looks like. Linux on IBM Z uses the same FICON-attached disk subsystems as z/OS, but it has no access method for z/OS cataloged data sets, so it can't read them just because they sit on the same disks.

A third option runs Linux inside z/OS itself. IBM **z/OS Container Extensions** (zCX) runs Linux applications "as Docker containers on z/OS as part of a z/OS workload", using images built for IBM Z. Its virtual processors can be dispatched on zIIPs, and IBM says that with enough zIIPs "the majority of zCX instance processing can execute on zIIPs."²³ zCX gets to z/OS data over TCP/IP, including the z/OS NFS server,²³ so the data-movement questions are the same. The difference is that z/OS measures the Linux work, and zIIP rules decide whether it's charged.

5. Same byte order, different alphabet

Linux on IBM Z is big-endian, just like z/OS.⁴ A four-byte binary integer written by a COBOL program under z/OS has the same bytes in the same order when Linux on the same hardware reads it. A sort that compares binary keys, or adds them up in a SUM step, can do it without reversing them. On little-endian x86, any program that loads a mainframe binary field into a native integer has to swap the bytes first, and if that step is wrong, nothing tells you.

Character data is another story. z/OS stores text in EBCDIC. Linux on IBM Z "supports Unicode and ASCII just like any other Linux distribution—it is not an EBCDIC-based operating system."⁴ The two encodings use different codes for the same letters, and they also sort them differently. Sorting EBCDIC "puts lowercase letters before uppercase letters and letters before numbers, exactly the opposite of ASCII."²⁴ Take account keys like A123, a123 and 1234. They sort one way on z/OS and in the reverse order on Linux if the data has been converted to ASCII first. Downstream programs that match, merge or search that file may then fail, and they won't necessarily fail loudly.

So if you move sort to Linux, you have to pick one of three approaches.

- **Keep the data in EBCDIC** and use a sort engine that compares EBCDIC bytes, reads zoned and packed decimal signs the way the mainframe does, and writes EBCDIC output. Nothing downstream changes.
- **Convert to ASCII and sort in ASCII order**, and live with a different sequence. That's only safe when every program that reads the output moves too. It also takes field-by-field conversion: character fields get converted, but packed, binary and zoned fields have to be left alone, because a blanket code-page translation corrupts them.
- **Convert to ASCII but sort in EBCDIC sequence**, using an alternate collating sequence, the way DFSORT itself does with ALTSEQ.²⁵ That keeps the order, but the output is no longer in the encoding the rest of the batch stream expects.

Whichever you pick, the transport has to match it. FTP in text mode and NFS with text translation convert records on the way; binary transfer moves them untouched. One step transferred in the wrong mode will pass every syntax check and still give you the wrong answer. That's why the pilot in section 8 insists on comparing outputs byte for byte. Exhibit 2, in the next section, sums up the four places a sort step can run.

6. The economics: what leaves the four-hour average, and what stays

The rule of thumb from the first paper was that under the rolling four-hour average, taking work off z/OS saves the smaller of two numbers: the MSUs that work adds inside the monthly peak window, and the gap between that peak and the next-highest one.¹ Moving sort to an IFL is one way of taking it off, so the rule still holds. If online work sets the monthly peak, moving overnight sort to Linux saves general-purpose capacity but no Monthly License Charge. If batch sets the peak, the gap to the next peak caps the saving.

Exhibit 2. Where a sort step can run, and what goes with each choice

	z/OS on general-purpose processors	z/OS on zIIP	Linux on IFL, same machine	Linux on x86 or separate server
Counts toward z/OS four-hour average?	Yes	No, where the product uses zIIP ⁶	No ⁵	No
Other charges that come with the processor	Capacity-based ISV products	zIIP hardware	IFL hardware; per-core Linux and ISV licenses ^{3, 7}	Server, per-core licenses
Where the data is sorted	In place	In place	Across the partition boundary, in memory	Across the external network
Path from z/OS data	None needed	None needed	HiperSockets, SMC-D, NFS ²⁰	OSA network, file transfer
Operating system character set	EBCDIC	EBCDIC	ASCII / UTF-8 ⁴	ASCII / UTF-8
Byte order	Big-endian	Big-endian	Big-endian	Little-endian
Main question for sort	When does it run?	How much qualifies?	What does the transfer cost on z/OS?	Transfer cost and identical output

Sources as cited. Charges refer to IBM Monthly License Charge products under sub-capacity pricing.

What the IFL adds is a new line on the z/OS side: the transfer. The sort's own CPU goes away. The CPU to read the input, drive the network stack, convert (or not) and read back or reload the output stays, and it runs in the same batch window the sort did. Exhibit 3 applies this to the made-up month-end case from the first paper, with two assumed levels of transfer overhead.

Exhibit 3. Moving a month-end sort workload to IFLs (illustrative)

MSUs, four-hour averages in the monthly peak window	Before	After: transfer at 20% of sort CPU	After: transfer at 60% of sort CPU
Sort steps on z/OS general-purpose processors	280	0	0
Transfer out and back, on z/OS general-purpose processors	0	56	168
Batch four-hour peak	1,120	896	1,008
Online four-hour peak (unchanged)	1,000	1,000	1,000
Billable four-hour peak	1,120	1,000	1,008
Reduction in billable MSUs	–	120	112
Net MSUs taken out of consumption (consumption-pricing view)	–	224	112

Made-up figures, carrying on from panel B of Exhibit 2 in the first paper of this series.¹ The transfer overheads are assumptions picked to show sensitivity. I haven't measured them; the real number depends on the path, the data volume and whether conversion is done on z/OS.

Two things come out of this. Under the four-hour average, the transfer overhead hardly matters in this example, because the online peak was already capping the saving. It would matter a lot in a month where batch set the peak by a wider margin. Under Tailored Fit Pricing's consumption model, where every MSU counts,² the overhead matters in full: at 60 percent, half the apparent benefit goes on moving the data. It's worth engineering down. SMC-D cuts network CPU by IBM's measurements,²² streaming avoids landing the data twice, and sorting in EBCDIC takes conversion off z/OS.

The other side of the ledger is the IFL setup itself: processor capacity, which may already be installed and idle or may need to be bought; per-core Linux and middleware licenses; and the sort license on Linux. None of these shows up in SCRT reports, so they're easy to leave out. Don't.

7. What's been published

Sagicor Bank Jamaica runs its Temenos core banking system on LinuxONE. IBM's case study, published in April 2024, says its end-of-day close dropped from more than six hours to under three, and estimates yearly savings of about USD 1 million.²⁶ Citi was named at the Emperor 4 launch as hosting MongoDB on LinuxONE.¹³ IBM published both, and both are Linux applications, with no z/OS batch moved across a partition boundary. I haven't found a public, documented case of a z/OS shop moving DFSORT-style batch sort to IFLs in the same machine and reporting the result. So measure your own case. Don't lean on published comparisons, including the vendor consolidation numbers in section 3.

8. A pilot, step by step

A pilot has to answer three questions. Does the sort workload matter to the bill? What does moving it cost on z/OS? Is the output identical? Exhibit 4 lays out a sequence that answers them with data most shops already collect.

Exhibit 4. Pilot checklist

- 1 Find out what's at stake.** From SCRT reports and SMF 70 and 89, find the four-hour window that set each recent monthly bill, and whether batch or online set it. If online sets every peak and you're not on consumption pricing, the IFL case has to rest on capacity and batch-window relief. MLC won't move.^{1,2}
- 2 Pick the candidates.** Join DFSORT SMF type 16 records to type 30 step data²⁵ to find large sort, merge and copy steps inside the peak window. Start with sequential data and simple record formats. VSAM input, program-invoked sorts and exits need unloading or redesign, so leave them for a second phase.
- 3 Settle the character-set strategy** before anything moves: EBCDIC end to end, field-level conversion, or conversion with an EBCDIC collating sequence (section 5). Write it down for every field of every key.
- 4 Build the path and measure it.** Set up HiperSockets or SMC-D between the z/OS and Linux partitions, pick the transport, and record z/OS CPU for the transfer from SMF 30 and, for TCP/IP and FTP, SMF 119. This is the number that decides the net saving (Exhibit 3).
- 5 Prove the output is identical.** Run each pilot step both ways and compare the outputs byte for byte, including record counts, return codes and any messages the scheduler acts on. Include month-end and year-end data, empty files and maximum-length records.
- 6 Hook up operations and control.** Tie the Linux step into the z/OS scheduler, restart and recovery; map security IDs across the boundary; encrypt the path where policy says so, as zCX does with AT-TLS for NFS.²³
- 7 Count the other side.** IFL capacity, per-core Linux and middleware licenses, the Linux sort license and staff time. Compare that with the saving from step 1, less the transfer cost from step 4.

9. Bottom line

- **The charging rule is clear, and it's in your favor.** Work on an IFL doesn't raise z/OS software charges or the machine's MSU rating. Sort moved there is out of the four-hour average.
- **The peak still caps the saving.** The first paper's rule still holds: what matters is how much sort adds to the peak window and the gap to the next peak.
- **The transfer is the hidden cost.** It runs on z/OS, in the same window. Memory-speed paths only exist inside one machine. Measure them between z/OS and Linux yourself; IBM's figures are z/OS to z/OS.

- **Big-endian helps. ASCII is the catch.** Binary fields travel unchanged, but you have to keep EBCDIC collating order or decide on purpose to drop it, and you have to prove the output is identical.
- **Measure first.** A seven-step pilot, mostly built on SMF data you already have, will tell you whether the machine next door is the right home for your sort before you commit to anything.

About the author

Bill Ahlbrandt is President and CEO of Ahlbrandt Software, which he founded in 1993. Before that he was a systems programmer, Director of Information Security and Director of R&D at Acxiom. He has been working on mainframe sort and batch for more than thirty years, and his company's sort engine runs mainframe sort jobs on Linux and other platforms for banks, insurers and government agencies.

Full disclosure: my company sells sort software for Linux and other platforms besides the mainframe, so I have an interest in the options this paper describes. The paper names no product, and my advice is to measure before you move any workload.

References

1. B. Ahlbrandt, *What Sort Really Costs on the Mainframe*, Ahlbrandt Software white paper, September 2026.
2. IBM, *IBM Z Software Pricing Reference Guide*, ZSO01378-USEN; IBM Tailored Fit Pricing announcement materials, 14 May 2019.
3. IBM, *Linux on IBM Z: Frequently Asked Questions*, May 2025.
4. Wikipedia, "Linux on IBM Z" (history, byte order and character set), accessed September 2026. Secondary source.
5. IBM, Integrated Facility for Linux (IFL), ibm.com/products/integrated-facility-for-linux, 16 May 2024.
6. IBM, IBM z Integrated Information Processor (zIIP), ibm.com/products/z-integrated-information-processor.
7. IBM Passport Advantage, Processor Value Units (PVUs) licensing for customers, table of PVUs per core.
8. Red Hat, *Red Hat Enterprise Linux subscription guide*, 21 July 2026.
9. Red Hat, *Self-managed Red Hat OpenShift subscription guide*.
10. CIO, "IBM launches LinuxONE at LinuxCon, announces Open Mainframe Project", 17 August 2015.
11. Planet Mainframe, "LinuxONE Emperor II: the Linux mainframe", 30 January 2018.
12. J. Elliott, "IBM z15 and IBM LinuxONE III announced", Jim Elliott's Mainframe Blog, September 2019.
13. IBM Newsroom, "New IBM LinuxONE Servers Help Reduce Energy Consumption...", 13 September 2022, with footnotes. Vendor figures.
14. Mainline Information Systems, "IBM LinuxONE Emperor 4 announcement", September 2022.
15. IBM Newsroom, "IBM Furthers Flexibility, Sustainability and Security within the Data Center with New IBM z16 and LinuxONE 4 Single Frame and Rack Mount Options", 4 April 2023 (availability 17 May 2023).
16. IBM Newsroom, LinuxONE Emperor 5 announcement, 6 May 2025.
17. K. Stine, IBM, *IBM z17 and LinuxONE Emperor 5 Technical Overview*, IBM Z Tech Bytes / VM Workshop, 2025.
18. IBM, "Expanding the IBM LinuxONE 5 Family: Introducing LinuxONE Rockhopper 5 and LinuxONE 5 Express", 7 July 2026.
19. T. Tarquinio, IBM, "The next generation of IBM LinuxONE", IBM Community, 6 May 2025, with claim footnotes. Vendor figures.
20. IBM Redbooks, *IBM Z Connectivity Handbook*, SG24-5444, July 2026 edition.
21. IBM, *Accelerate Networking with Shared Memory Communications on IBM Z*, v2.0, 18 April 2023.
22. IBM Support, "On the IBM z13 System, Colocation Matters!", 22 May 2020. IBM laboratory measurements, z/OS to z/OS.
23. IBM, *z/OS 3.2 Container Extensions Guide*, last updated 22 June 2026.
24. Wikipedia, "EBCDIC" (collating sequence), accessed September 2026. Secondary source.
25. IBM, *z/OS DFSORT Application Programming Guide* (ALTSEQ; SMF type 16 records).
26. IBM, case study: Sagicor Bank Jamaica, 5 April 2024. Vendor-published.