Simultaneous MultiThreading (SMT)

- The z13, aligned with multithreading industry directions, can process up two simultaneous threads in a single core.
- This capability is known in z Systems as simultaneous multithreading (SMT) and offers full z/Architecture efficiency for each thread.

**NOTE:** SMT is supported only by zilp and IFL specialty engines on z13.

- An operating system or SMP support can be configured to dispatch work to a thread on a zilp (for eligible workloads in z/OS) or an IFL (for 2xV on behalf of Linux on z System guest or native Linux on z Systems) core in single thread or SMT mode.
- The use of SMT provides a better and more efficient use of the processor resources and address memory latency, resulting in overall throughput gains.

**NOTE:** SMT is designed to deliver better overall throughput for many workloads, but not all workloads, where each thread is processing single-thread workloads.

- Each software Operating System / Hypervisor has the ability to intelligently drive SMT in a way that is best for its unique requirements.
- z/VMM management optimizes throughput by spreading a workload over the available cores until it demands the additional SMT capacity.

**How SMT benefits performance**

- When a program accesses a memory location that is not in the cache, it is called a cache miss.
- Because a core processor then must wait for the data to be fetched from the next cache level, or from main memory before it can continue to execute, cache misses directly influence the performance and capacity of the core to execute instructions.
- With simultaneous multithreading exploitation, when one thread in the core is waiting, for example, for data to be fetched from the next cache levels or from main memory, the second thread in the core can utilize the shared resources rather than remain idle.

**z/OS V2.2 is planned to support the use of SMT for zilp processors.**

- To support this new function, z/OS V2.2 is planned to provide:
  - IPL-time controls to enable SMT and set the SMT mode
  - Post- IPL controls to dynamically switch the SMT mode
  - SMF Type 30 record fields with a normalized value for CPU time spent on a processor running in SMT mode
  - SMF Type 70 records with new SMT-related fields
  - Parallel SystemgetX (XES) use of SMT mode for workloads using zilps to help improve physical processor utilization for synchronous requests
  - Hardware Instrumentation Services (HIS) updates to provide measurement data in SMT mode
  - New SMT-related metrics to help you with capacity planning and performance analysis.

**To underpin SMT, the z13 has a double symmetric instruction pipeline width and full architectural state per thread.**

- z/OS, V2.2 expands the CPU address changes and the 16-bit CPU ID consists of 15-bit core ID and 1-bit thread ID.
- For example, the CPU ID 6 (‘0000000000001101’) means core 3 thread 0 and the CPU ID 7 (‘0000000000000111’) means core 3 thread 1. For CPUs, only thread 0 is used in each core.
- Any given pipeline stage Simultaneous multithreading technology allows instructions from more than one thread to run in any given pipeline stage at a time.
- Each core has its own unique state information, such as PSW, and registers.
- The simultaneous threads cannot necessarily run instructions instantly and must at times compete to use certain hardware resources that are shared between the threads.
- In some cases, threads can share hardware resources that are not experiencing competition.
- The use of SMT provides more efficient use of the processors’ resources and helps address memory latency, resulting in overall throughput gains.
- The active thread shares core resources in space, such as data and instruction caches, TLBs, branch history table, and in some cases, the same instruction and address translators.
- Although SMT increases the processing capacity, the performance in some cases might be superior if you use a single thread.
- Enhanced hardware monitoring support measures through CPUMF, thread usage, and capacity.
- For workloads that need maximum performance, the partition’s SMT mode can be turned on.
- The SMT exploitation is functionally transparent to middleware and applications where no changes are required to run in an SMT-enabled partition.

**To activate SMT function on z/OS**, define the PROCVIEW CORE option in LOADxx. If you do not use the SMT-2 function, define PROCVIEW CPU. The default parameter of PROCVIEW is CPU.

**NOTE:** You can now display the activity for an Logical Partition (LPAR) by processor type and the Monitors Dashboard is enhanced with simplified multithreading (SMT) usage (ex: see bottom, right diagram).

**Which approach is designed for the highest volume of traffic? Which road is faster?**

Up to 2 simultaneous processors or up to 128 processors per LPAR on z13 processors when running in SMT mode.

**The SMT is functionally transparent to middleware and applications where certain changes are required to run in a SMT partition.**

Application servicing with SSL could see up to x2 improvement in throughput for z/OS, Java Technology Edition, Version 8 on IBM z13 with SMT vs. Java 7 on zEC12.

When the SMT facility is enabled, the hardware threads within a core share certain hardware resources such as execution units and caches. When the SMT is not enabled, a core executes a single hardware thread.

**Single-instruction, multiple-data (SIMD): Vector processing unit (one instruction is applied to a vector of data)**

- z13 introduces architectural extensions with instructions that are designed to allow reduced processor quiesce effects, reduced cache misses, reduced pipeline disruption, and increased parallelism with instructions that process several operands in a single instruction, thus 8 x Single Instruction Multiple Data (SIMD).
- The z13 super scalar processor has 32 vector registers and an instruction set architecture that includes a subset of 130 new instructions, known as SIMD, added to improve the efficiency of complex mathematical models and vector processing.
- The new SIMD instructions allow a larger number of operations to be processed in a single instruction.
- The SIMD instructions use the superscalar core to process operands in parallel.

**z/OS V2.1 is planned to provide support for the new vector extension facility (SIMD) instructions available on IBM z13 server**

- The new SIMD instructions are available for z/OS V2.1 with PTFs intended to help enable high-performance analytics processing, and is planned to be exploited by z/OS XML System Services, IBM 31-bit SDK for z/OS, Java Technology Edition, Version 8 (5655-DGG), IBM 64-bit SDK for z/OS, Java Technology Edition, Version 8 (5655-DGH); Enterprise PL/I for z/OS & z/VM (5655-W67), and Enterprise COBOL for z/OS (5655-W07).
- IBM intends to exploit the 64-bit SDK for z/OS, Java Technology Edition, Version 8 in IBM WebSphere® Liberty Profile for z/OS, and in the full profile of WebSphere Application Server for z/OS, which is also expected to benefit from SIMD acceleration.

**NOTE:** If SIMD is enabled, this feature will be fully used by z/OS V2R1 where several compilers have built-in functions for SIMD.

- SIMD uses the enhanced super scalar z13 core to process a large number of operands (vector) through a single instruction, allowing the development of SIMD optimized and optimized coded to improve efficiency of complex mathematical models and vector processing.

- SIMD - MASS and ATLAS libraries for z/OS (and Linux on z Systems):
  - Allow construction of rich, complex analytics models using BSIV/SWIV Analytics products that exploit SIMD to provide better accuracy of insight
  - Analyze workloads to be ported from Power and x86 with ease that can accelerate analytics to provide speed business insight
  - Increase programmer productivity of ISV and customer analytics workload development leading to rapid business insight generation for advantage

- SIMD instructions allow a larger number of operands to be processed with a single instruction.
- The SIMD instructions uses the superscalar core to process operands in parallel, enabling more parallelism.
- The set of SIMD instructions is a type of data parallel computing architecture that can decrease amount of code and accelerate code with integer, string, and floating point data types.
- SIMD provides the next phase of enhancements of 2 systems analytics capacity.
- The new SIMD instructions are a type of data parallel computing architecture that can decrease the amount of code as well as accelerate code that handles integer, string, character, and floating point data types.
- The SIMD instructions improve performance of complex mathematical models and permit integration of business transactions and analytic workloads on 2 Systems
- The 32 new vector registers have 128 bits.
- The SIMD instructions include string operations, vector floating point, and vector floating point joint operations.
- Each register contains multiple data elements of a fixed size. The instructions code specifies which data format to use and whether the size of the elements is byte (sixteen 8-bit operands), halfword (eight 16-bit operands), word (32-bit operands), doubleword (two 64-bit operands), or quadword (one 128-bit operand).
- The collection of elements in a register is called a vector.

**NOTE:** A single instruction operates on all of the elements in the register. Instructions have a non-destructive operand encoding that permits the addition of the register vector A and register vector B, and stores the result in the register vector A (A = A + B).

- ID MnCPU command with PROCVIEW CPU