# z/VM and the IBM z13

## Session 16453

*John Franciscovich*

*IBM:  z/VM Development, Endicott, NY*

#SHAREorg

SHARE is an independent volunteer-run information technology association that provides **education**, professional **networking** and industry **influence**.

SHARE in Seattle 2015

# Acknowledgements


- Bill Bitner
- Emily Hugenbruch
- Xenia Tkatschow
- Brian Wade
- Mark Lorenc
- Kevin Adams
- Romney White
- Angelo Macchiano
- Sue Farrell

- … and anyone else who contributed to this presentation that I may have omitted

# Topics

- Overview of z/VM support for the IBM z13™

- z/VM Enhancements to exploit z13 features
  - Simultaneous Multithreading (SMT)
  - Increased Processor Scalability
  - Multi-Vswitch Link Aggregation
  - Crypto Enhancements

# z/VM Support for the IBM z13

# Expanding the Horizon of Virtualization

- Release for Announcement – The IBM z13™
  - January 14, 2015
  - Announcement Link

- z/VM Compatibility Support
  - PTFs available February 13, 2015
  - Also includes Crypto enhanced domain support
  - z/VM 6.2 and z/VM 6.3
  - No z/VM 5.4 support
  - Refer to bucket for full list

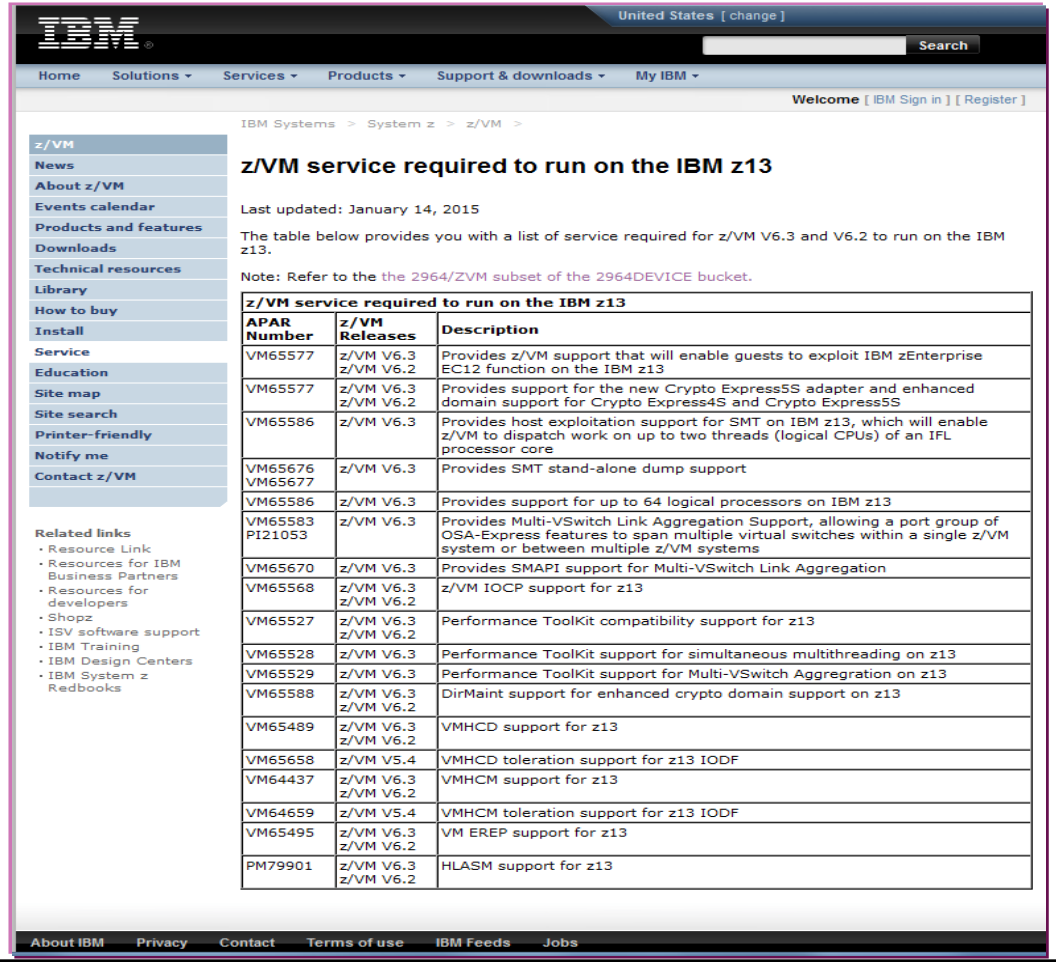
- Enhancements and Exploitation Support only on z/VM 6.3
  - IBM z13 Simultaneous Multithreading
  - Increased Processor Scalability
  - Multi-Vswitch Link Aggregation Support (Link Aggregation with Shared OSAs)

# z/VM Support for IBM z13

- **Updates for z/VM 6.2 and 6.3**
  - Many components affected
- No z/VM 5.4 Support
- No z/VM 6.1 Support even if you have extended support contract.
- **PSP Bucket**
  - Upgrade **2964DEVICE**
  - Subset **2964/ZVM**
- **If running Linux, please also check for required updates prior to migration.**

# z/VM Service Required for the IBM z13

http://www.vm.ibm.com/service/vmreqz13.html



IBM

United States [ change ]

Search

Home | Solutions | Services | Products | Support & downloads | My IBM

Welcome [ IBM Sign in ] [ Register ]

IBM Systems > System z > z/VM >

- z/VM
- News
- About z/VM
- Events calendar
- Products and features
- Downloads
- Technical resources
- Library
- How to buy
- Install
- Service
- Education
- Site map
- Site search
- Printer-friendly
- Notify me
- Contact z/VM

Related links
- Resource Link
- Resources for IBM Business Partners
- Resources for developers
- Shopz
- ISV software support
- IBM Training
- IBM Design Centers
- IBM System z Redbooks

## z/VM service required to run on the IBM z13

Last updated: January 14, 2015

The table below provides you with a list of service required for z/VM V6.3 and V6.2 to run on the IBM z13.

Note: Refer to the the 2964/ZVM subset of the 2964DEVICE bucket.

**z/VM service required to run on the IBM z13**

| APAR Number | z/VM Releases | Description |
|---|---|---|
| VM65577 | z/VM V6.3<br>z/VM V6.2 | Provides z/VM support that will enable guests to exploit IBM zEnterprise EC12 function on the IBM z13 |
| VM65577 | z/VM V6.3<br>z/VM V6.2 | Provides support for the new Crypto Express5S adapter and enhanced domain support for Crypto Express4S and Crypto Express5S |
| VM65586 | z/VM V6.3 | Provides host exploitation support for SMT on IBM z13, which will enable z/VM to dispatch work on up to two threads (logical CPUs) of an IFL processor core |
| VM65676<br>VM65677 | z/VM V6.3 | Provides SMT stand-alone dump support |
| VM65586 | z/VM V6.3 | Provides support for up to 64 logical processors on IBM z13 |
| VM65583<br>PI21053 | z/VM V6.3 | Provides Multi-VSwitch Link Aggregation Support, allowing a port group of OSA-Express features to span multiple virtual switches within a single z/VM system or between multiple z/VM systems |
| VM65670 | z/VM V6.3 | Provides SMAPI support for Multi-VSwitch Link Aggregation |
| VM65568 | z/VM V6.3<br>z/VM V6.2 | z/VM IOCP support for z13 |
| VM65527 | z/VM V6.3<br>z/VM V6.2 | Performance ToolKit compatibility support for z13 |
| VM65528 | z/VM V6.3 | Performance ToolKit support for simultaneous multithreading on z13 |
| VM65529 | z/VM V6.3 | Performance ToolKit support for Multi-VSwitch Aggregration on z13 |
| VM65588 | z/VM V6.3<br>z/VM V6.2 | DirMaint support for enhanced crypto domain support on z13 |
| VM65489 | z/VM V6.3<br>z/VM V6.2 | VMHCD support for z13 |
| VM65658 | z/VM V5.4 | VMHCD toleration support for z13 IODF |
| VM64437 | z/VM V6.3<br>z/VM V6.2 | VMHCM support for z13 |
| VM64659 | z/VM V5.4 | VMHCM toleration support for z13 IODF |
| VM65495 | z/VM V6.3<br>z/VM V6.2 | VM EREP support for z13 |
| PM79901 | z/VM V6.3<br>z/VM V6.2 | HLASM support for z13 |

About IBM | Privacy | Contact | Terms of use | IBM Feeds | Jobs

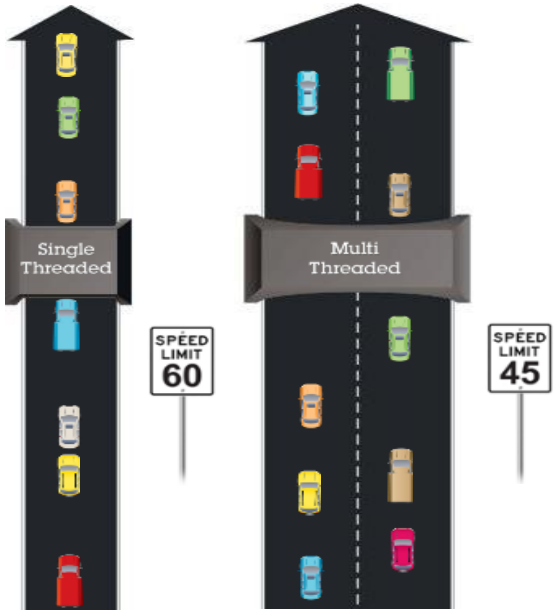
# z/VM z13 Compatibility

- Guest support for the following new hardware facilities:
  - Load/Store-On-Condition Facility 2
  - Load-and-Zero-Rightmost-Byte Facility
  - Decimal-Floating-Point Packed Conversion Facility
  - PCIe: Extended-I/O Address-Translation Facility guest exploitation
- Integer or Binary Floating Point for Capability Numbers
  - Accounting / Monitor will now report integer and binary floating point numbers for capability values
  - Q CAPABILITY will now report decimal numbers for capability values.
- Removal of guest zAAP support
- Toleration of
  - STP Hardware-Based TOD-Clock Steering
  - SMT feature on machine;  StandAlone dump compatibility
  - z/Architecture Vector Registers; mask from guests
- Dynamic I/O support for HiperSockets VCHID and CS5 Coupling

# Simultaneous multithreading (SMT)

# Why Simultaneous Multithreading (SMT)?

- Increase processing efficiency and overall throughput for many workloads
  - Amount of benefit for different workloads **will** vary
- Helps address memory latency
  - Second thread can run on processor while first thread resolves a cache miss




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

**Illustrative numbers only*

# Cores, Threads, and Logical Processors (CPUs)

- Core
  - When multithreading is enabled (IFL only)
    - Contains multiple threads, each the equivalent of a logical processor (CPU)
    - Each core has same number of threads
  - When multithreading is not installed or not enabled
    - Equivalent of a single logical processor

- Thread
  - When multithreading is enabled
    - Synonymous with logical processor that is a member of a core
    - Each thread within a core has same processor type and polarization
    - Threads within a core share some hardware resources
      - Execution of one thread in a core can affect performance of other threads in the same core
  - When multithreading is not installed or not enabled
    - Each core runs a single thread (logical processor)

- Logical Processor (CPU)
  - A thread in a core
  - All of the architected resources available to programs

# What does it really mean to have two threads share the same core?

## Threads share:

- Cache
- Execution units
- Dispatch vector
- Translation Lookaside Buffer (TLB)

## Each thread has its own:

- Program Status Word (PSW)
- Registers
- Timing Facilities
- Translation Lookaside Buffer entries

# Two Threads – Example 1

**Thread 0**

```
L     R3,FIELDA
L     R6,FIELDC
```

**Thread 1**

```
LLGC  R5,FIELDB
LGR   R3,R5
LGHI  R0,1
SLLG  R3,R0,0(R3)
```

Let's say:

- FIELDA is in the L3 cache
- FIELDB is in the L1 cache
- FIELDC is in L4 cache

*Note that this is a contrived example, not necessarily representative of the real amount of time these instructions take.

# Two Threads – Example 1

| *Thread 0* | *Thread 1* |
|---|---|
| Resolving FIELDA | Resolving FIELDB |
| | LLGC R5,FIELDB |
| L R3,FIELDA | |
| Resolving FIELDC | LGR R3,R5 |
| | LGR R3,R5 |
| | LGHI R0,1 |
| | SLLG R3,R0,0(R3) |
| L R6,FIELDC | |

This lets thread 1 run and get work done while thread 0 is waiting for its memory references to be resolved, thus using the core more efficiently.

Because each thread keeps its own registers, their instructions can be interleaved.

# Two Threads – Example 2

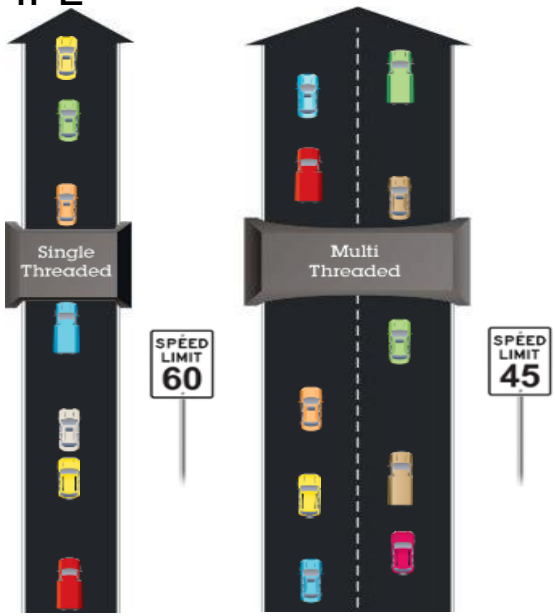
| *Thread 0* | *Thread 1* |
|---|---|
| | AR R0,R1 |
| AR R3,R4 | |
| | AR R3,R5 |
| | AR R3,R5 |
| | AR R0,R1 |
| AR R3,R1 | |
| AR R9,R4 | |
| | AR R2,R1 |

Of course this doesn't work well all the time - what if both threads just have instructions with no memory references?

In this case, each thread will run noticeably slower than it would if it had its own core.

# SMT on z/VM

- Objective is to improve capacity, not performance.
- Allows z/VM to dispatch work on up to two threads of a z13 IFL
  - Up to 32 IFLs (cores) supported
- VM65586 for z/VM 6.3 only
  - PTFs planned to be available March 13, 2015
- Transparent to virtual machine
  - Guest does not need to be SMT aware
  - SMT is not virtualized to the guest
- z13 SMT support limited to IFLs and zIIPs
  - z/VM support is only for IFLs
- SMT is disabled by default
  - Requires a System Configuration setting and re-IPL
  - When enabled, applies to the entire system
- Potential to increase the overall capacity of the system
  - Workload dependent

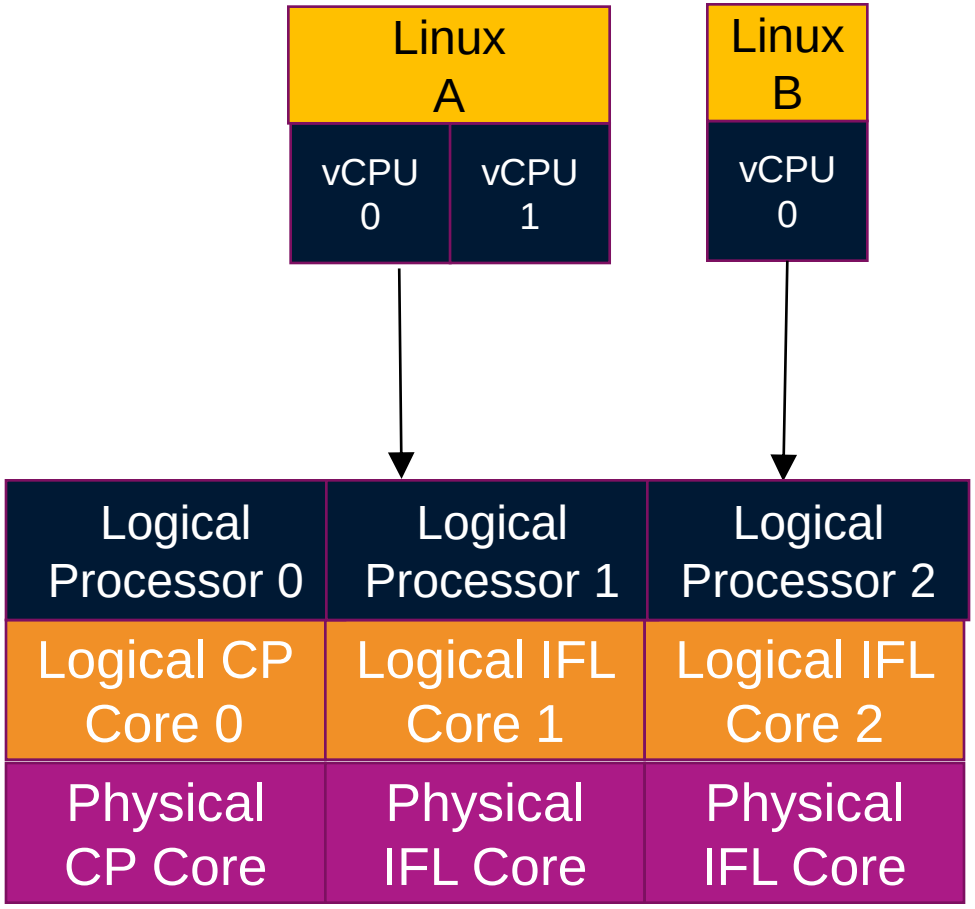



*Which approach is designed for the higher volume of traffic? Which road is faster?*

**Illustrative numbers only*

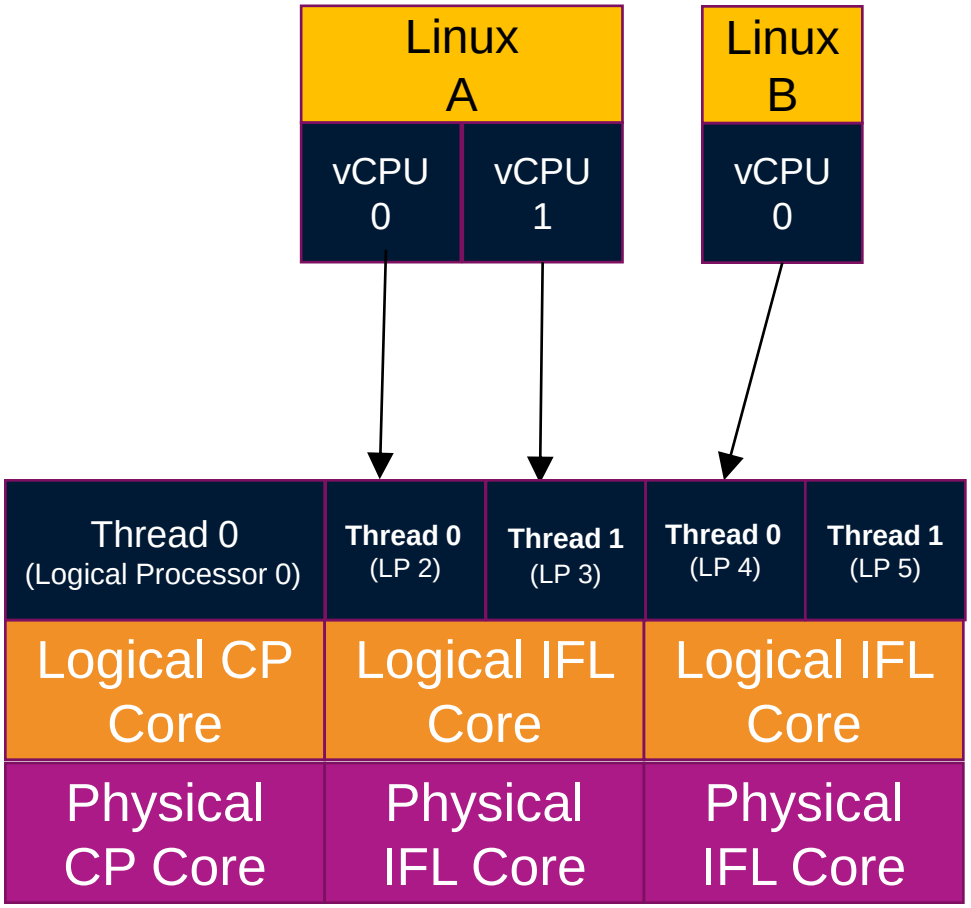
# Dispatching: SMT Disabled

- Physical IFLs and CPs are single-threaded
- z/VM creates a logical processor (CPU) for each associated logical IFL or CP
- The virtual processors of guests are dispatched on individual logical processors (CPUs)



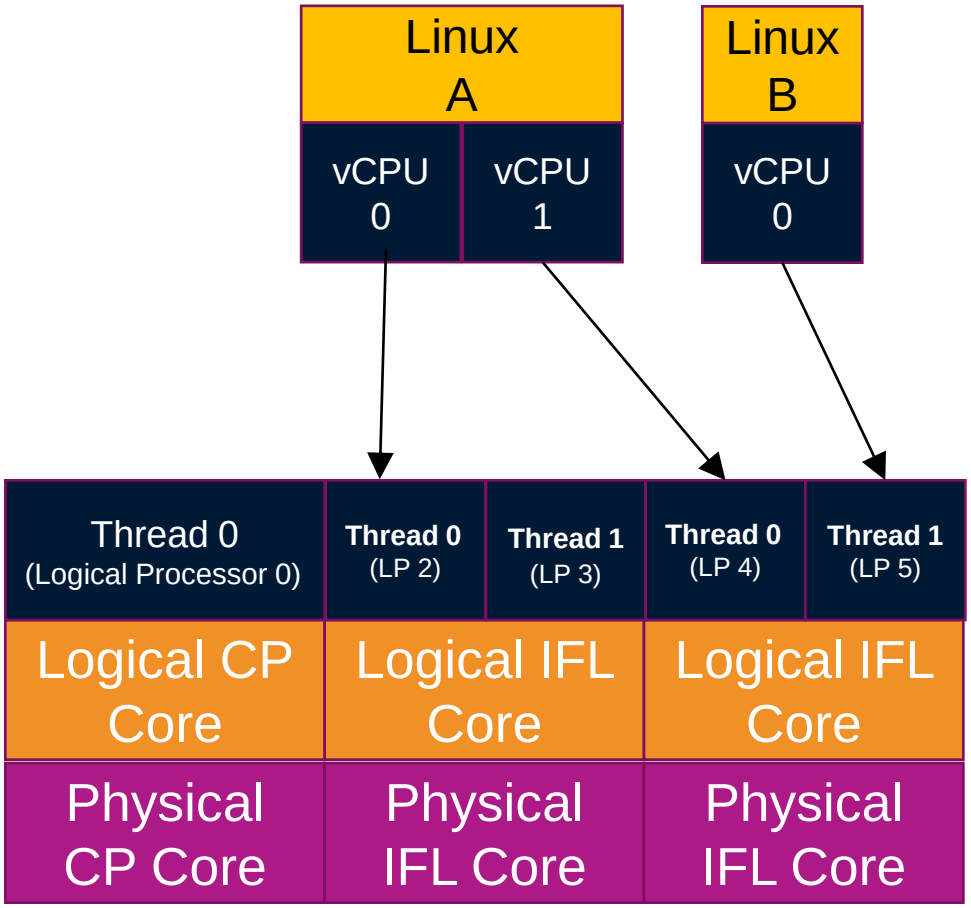
# Dispatching: SMT Enabled

- Physical IFLs (or Cores) with SMT allow up to two threads to be used.
- Logical IFLs are presented to z/VM as in the past.
- z/VM creates a logical processor (CPU) associated with each thread for it to use.
- The virtual CPUs of guests can then be dispatched on different threads intelligently, based on topology information.
- In a mixed-engine environment, general purpose processors can not do threading, but a second logical processor address is consumed (LP 1 in example)



# Dispatching: SMT Enabled

- Physical IFLs (or Cores) with SMT allow up to two threads to be used.
- Logical IFLs are presented to z/VM as in the past.
- z/VM creates a logical processor (CPU) associated with each thread for it to use.
- The virtual CPUs of guests can then be dispatched on different threads intelligently, based on topology information.
- In a mixed-engine environment, general purpose processors can not do threading, but a second logical processor address is consumed (LP 1 in example)



# SMT in Mixed Engine Environments

- z/VM only supports multithreading on IFL cores

- A VM mode partition will have both
  - Multithreaded IFL cores
  - Singlethreaded cores (e.g., CPs, zIIPs)

- Guests in a VM mode partition must have virtual IFLs defined in order to use IFL cores
  - Virtual IFLs are only used when you **SET VCONFIG MODE LINUX** or **VM.**
  - If you **SET VCONFIG MODE LINUX** you have to choose all virtual IFLs
  - Resetting your VCONFIG settings or redefining the type of CPUs will cause a SYSTEM RESET that will kill your guest operating system.

# How do I enable SMT on my z/VM system?

- Add the **MULTITHreading ENAble** statement to your SYSTEM CONFIG

- The system must be in *vertical polarization mode* (this is the default)
  - Make sure you ***don't*** have an **SRM POLARIZATION HORIZONTAL** statement in your SYSTEM CONFIG.

- The system must be using the *reshuffle dispatcher method* (this is the default)
  - Make sure you ***don't*** have an **SRM DSPWDMethod REBALANCE** statement in your SYSTEM CONFIG.

- Re-IPL your system!

# Enabling SMT – New MULTITHreading statement

- **MULTITHreading** configuration statement allows you to specify either
  - maximum number of threads for all core types
  - different number of threads for each type
    - z/VM only supports IFL cores for multithreading.

- **CPSYNTAX** has been updated to verify:
  - Are there multiple **MULTITHreading** statements?
  - Is the maximum activated thread value less than the number of threads specified for any type?
  - Is **MULTITHreading ENABLE** specified with any incompatible **SRM** statements?

# I believe I enabled SMT, but how do I know it's on?

- New command – **Query MULTITHread (MT)**

- Compares what you requested in the SYSTEM CONFIG statement to what was actually available to be activated, given the hardware and software levels.

```
query multithread
Multithreading is enabled.
                Requested     Activated
                Threads       Threads
MAX_THREADS       MAX             2
CP core             2             1
IFL core            2             2
ICF core            2             1
zIIP core           2             1
Ready; T=0.01/0.01  11:51:29
```

# QUERY PROCessors with SMT

- Shows which core each thread/processor is on:

```
query processors
PROCESSOR 00 MASTER CP    CORE 0000
PROCESSOR 02 ALTERNATE CP    CORE 0001
PROCESSOR 04 ALTERNATE IFL   CORE 0002
PROCESSOR 05 ALTERNATE IFL   CORE 0002
PROCESSOR 06 PARKED IFL   CORE 0003
PROCESSOR 07 PARKED IFL   CORE 0003
PROCESSOR 08 ALTERNATE IFL   CORE 0004
PROCESSOR 09 ALTERNATE IFL   CORE 0004
PROCESSOR 0A ALTERNATE IFL   CORE 0005
PROCESSOR 0B ALTERNATE IFL   CORE 0005
PROCESSOR 0C ALTERNATE IFL   CORE 0006
PROCESSOR 0D ALTERNATE IFL   CORE 0006
PROCESSOR 0E PARKED IFL   CORE 0007
PROCESSOR 0F PARKED IFL   CORE 0007
PROCESSOR 10 ALTERNATE IFL   CORE 0008
PROCESSOR 11 ALTERNATE IFL   CORE 0008
PROCESSOR 12 ALTERNATE IFL   CORE 0009
PROCESSOR 13 ALTERNATE IFL   CORE 0009
PROCESSOR 14 ALTERNATE ZIIP CORE 000A
PROCESSOR 16 ALTERNATE ZIIP CORE 000B
Ready; T=0.01/0.01 11:55:52
```

# Vary On and Off

- When SMT is enabled
  - Use **VARY CORE** to vary off or on an entire core
    - Multithread or single thread cores
    - Cannot vary a single thread of a core.
  - **VARY PROCESSOR** isn't allowed

- When SMT is not installed or not enabled
  - **VARY CORE** is the same as **VARY PROCESSOR**

```
vary off processor a
HCPCPS1321E VARY PROCESSOR is not valid because multithreading is enabled.
Ready(01321);
vary off core 5
Command accepted
Ready;
Core 0005 offline Proc 000A-000B
vary on core 5
Command accepted
Core 0005 online Proc 000A-000B
Ready;
```

# SMT is enabled - how do I see what's going on with my cores?

- **Indicate Load** will still show information by processor, which means by individual thread on multithreaded cores.
  - Can be confusing because each thread won't always be able to use 100% of the core
- A new command, **INDicate MULTITHread (MT)** will show you the per type information, giving you an idea of how much capacity you have left for each type. The utilization shown is an average of the utilization of the cores of that type.

```
indicate multith
Multithreading is enabled.
Statistics from the interval 12:00:53 - 12:01:23
Core Type CP   Busy    8%   TD  1.00 of  1   Prod 100%   Util   8%
  CF  100%   MaxCF  100%
Core Type IFL  Busy    1%   TD  1.50 of  2   Prod  90%   Util   1%
  CF  113%   MaxCF  125%
Core Type ZIIP Busy    0%   TD  1.00 of  1   Prod 100%   Util   0%
  CF  100%   MaxCF  100%
Ready;
```

# What are all those other numbers on Indicate MT?

- Busy time - how often the core was executing instructions during the interval.
- Thread density - how often the core was able to run both threads at once, while the core was in use at all.
- Productivity - how often the core was completely busy on both threads.
- Utilization - how much of the maximum core capacity was used.
- Capacity factor - a way of looking at the amount of work the multithreaded core was able to accomplish compared to the amount of work a single threaded core could accomplish.
- Maximum Capacity factor - how much work could've been accomplished at the current rate, if the core had been kept busier.

```
indicate multith
Multithreading is enabled.
Statistics from the interval 12:00:53 - 12:01:23
Core Type CP   Busy    8%   TD  1.00 of  1   Prod 100%   Util   8%
  CF  100%   MaxCF  100%
Core Type IFL  Busy    1%   TD  1.50 of  2   Prod  90%   Util   1%
  CF  113%   MaxCF  125%
Core Type ZIIP Busy    0%   TD  1.00 of  1   Prod 100%   Util   0%
  CF  100%   MaxCF  100%
Ready;
```

# Processor Time Reporting

- **Raw time** (the old way, but with new implications)
  - Time each virtual CPU is dispatched on a thread
    - Includes time thread is not doing work
  - The only time measurement when SMT is disabled
  - Used to compute dispatcher time slice and scheduler priority

- **MT-1 equivalent time** (new)
  - Used when SMT is enabled
  - Approximates what the raw time would be if SMT were disabled
    - Adjusts CPU time for each thread

# Processor Time Reporting

| | Raw Time | MT-1 Equivalent time |
|---|---|---|
| INDICATE USER | | x |
| QUERY TIME | | x |
| LOGOFF | | x |
| TYPE 1 Accounting record | | x |
| TYPE F Accounting record | x | |
| Diag x'0c' | x | |
| Diag x'70' | x | |
| Diag x'270' | x | |
| Diag x'2FC' | x | |
| Monitor Records | x | x |

Note: "CONNECT" time displayed by commands represents wall-clock time and is not changed

# SMT - CPU Pooling Implications

- With SMT enabled
  - **CAPACITY** limit for CPU pools is defined by number of IFLs (cores) …. but limit is enforced using thread utilization
  - In some cases, guests in a CPU pool will not be able to execute the same amount of work as before SMT with the same capacity limit
  - Capacity limits for CPU pools might need to be increased

# Prorated Core Time (availability TBD)

- Prorated core time will divide the time a core is dispatched evenly among the threads dispatched in that interval
  - CPU pool capacity consumed as if by cores
  - Suitable for core-based software licensing

- When SMT is enabled, prorated core time will be calculated for users who are
  - In a CPU pool limited by the **CAPACITY** option
  - Limited by the **SET SHARE LIMITHARD** command

  (currently raw time is used; raw time will continue to be used when SMT is disabled)

- **QUERY CPUPOOL** will show capacity in cores instead of CPUs

- Prorated core time will be reported in monitor records and the new Type F accounting record.

- Watch for APAR VM65680

# Live Guest Relocation Implications

- Guests can be relocated between SMT enabled and SMT disabled z/VM systems because SMT is transparent to guests
  - Capacity will be affected
    - Might require adjustment to the number of virtual CPUs
  - Because of differences in CPU time calculation they may see their CPU time advance at different rates.

# SMT Performance Summary

- Performance varies from workload to workload
- Workload throughput and response time is the best way to determine whether SMT is providing value
- Throughput for measured workloads varied from 0.64x to 1.34x
- BEST results were observed in applications that consist of highly parallel activity
- WORST results were observed in applications that consist of highly serial activity
- Adjusting the workload when SMT is enabled may be necessary to overcome serial activity
  - For example, adding virtual CPUs to a guest
- More to come in updated Performance Report

# Increased CPU Scalability

# Increased CPU Scalability

- Various improvements to allow z/VM systems to be larger in terms of processors and more efficient, improving the n-way curve

- APAR VM65586 for z/VM 6.3 **only**
  - PTFs planned to be available March 13, 2015

- For z13
  - With SMT disabled, increases logical processors supported from 32 to 64
  - With SMT enabled, the limit is 32 IFLs (64 threads)

- For processors prior to z13
  - Limit remains at 32
  - May still benefit from improved n-way curves

# Areas Improved to Increase CPU Scalability

- Improvements were made to the following areas to improve efficiency and reduce contention
  - Scheduler lock
  - Vswitch data transfer buffers
  - Serialization and processing of VDisk I/Os
  - Memory Management

- Some areas needing improvement were known – others required thorough investigation and experimentation

- All tested workloads now show acceptable scaling up to a 64-way LPAR
  - Benefits are workload dependent

# CPU Scalability: Scheduler Lock

- Reduced contention for scheduler lock
  - Known to cause large overhead for many workloads
- Eliminated unnecessary exclusive holds of scheduler lock
  - Only share needed
  - Not needed at all
- Improved efficiency of processes that use the scheduler lock
  - Work stacking improvements
  - Test-idle changes

# CPU Scalability: Vswitch Data Transfer Buffers

- Frequently obtained in systems with high Vswitch activity
  - Caused spin lock overhead
- Added processor-local queues to existing global queues
- Modified structure of global queues
- Improved efficiency
  - Dequeuing batches of buffers
  - Moving batches of buffers from global queues to local queues
- Minimized lock hold time

# CPU Scalability: VDISK I/O

- Reduced attaches and detaches of VDISK address spaces
  - Previously attached and detached after each CCW was processed
  - Now only done once per channel program

- Reduced serialization contention when updating CP-use access list (used to read and write VDISK data)
  - Removed lock usage for some list processing
  - Reduced likelihood of concurrent tasks using the same section of the list
    - Starting index of search for available list entries is derived from processor address

# CPU Scalability: Memory Management

- Improved management of frame lists
  - Local frame available list
  - Processed frame list
- Improved page table serialization
  - Mainly affects memory allocation and de-allocation
- Most benefit seen during system initialization and when memory is very constrained

# CPU Scalability: Processor Parking

- Base z/VM 6.3.0
  - T/V ratio used as indicator of MP overhead
    - Used in processor parking decision algorithm

- z/VM 6.3.0 with CPU scalability enhancements
  - CP overhead due to MP is reduced
  - T/V ratio is a less accurate indicator of MP overhead
    - Removed from parking decision

- We no longer park entitled engines (vertical highs or mediums)

# Multi-Vswitch Link Aggregation

# Overview

- Makes it possible to do Link Aggregation with VSwitches without the requirement for dedicated OSAs
- Allows a port group of OSA-Express features to span VSwitches within a single or multiple z/VM systems in same CEC
  - Cannot be shared with non-z/VM logical partitions or z/VM systems without support
- Only available on z13
  - Requires OSA enhancements introduced with the z13
- Allows better consolidation and availability while improving TCO
- APARs VM65583 and PI21053 for z/VM 6.3 **only**
  - PTFs planned to be available June 26, 2015

# New Attributes

- **Inter-VSwitch Link (IVL)network**
  - Provides management and data communication among multiple z/VM systems

- **IVL virtual switch**
  - One per z/VM system
  - Communicates with other systems in the same IVL Domain
  - Accommodates all inter-LPAR control traffic between systems

- **IVL domain**
  - Group of up to 16 z/VM systems
    - Each system has an IVL virtual switch defined with UPLINK port configured and connected to same external LAN segment
  - Each IVL domain is assigned a separate, reserved multicast MAC address
  - Each z/VM system can belong to one IVL domain
  - Defined with configuration of IVL virtual switch

**DEFINE VSWITCH SUE TYPE IVL DOMAIN A UPLINK RDEV 0300 0200**

# New Attributes…

- **Global virtual switch**
  - Collection of virtual switches that share the same networking characteristics
    - Same name
    - Reside on systems that are part of the same IVL domain
  - In order to create a global virtual switch "member"
    - The system's IVL virtual switch must be defined and its UPLINK port connected
    - The virtual switch's attributes must match any existing virtual switches with the same name in the IVL domain
  - Created when first member is defined, deleted when last member is detached

**DEFINE VSWITCH RICK TYPE QDIO GLOBAL ETHERNET UPLINK GROUP MARY**

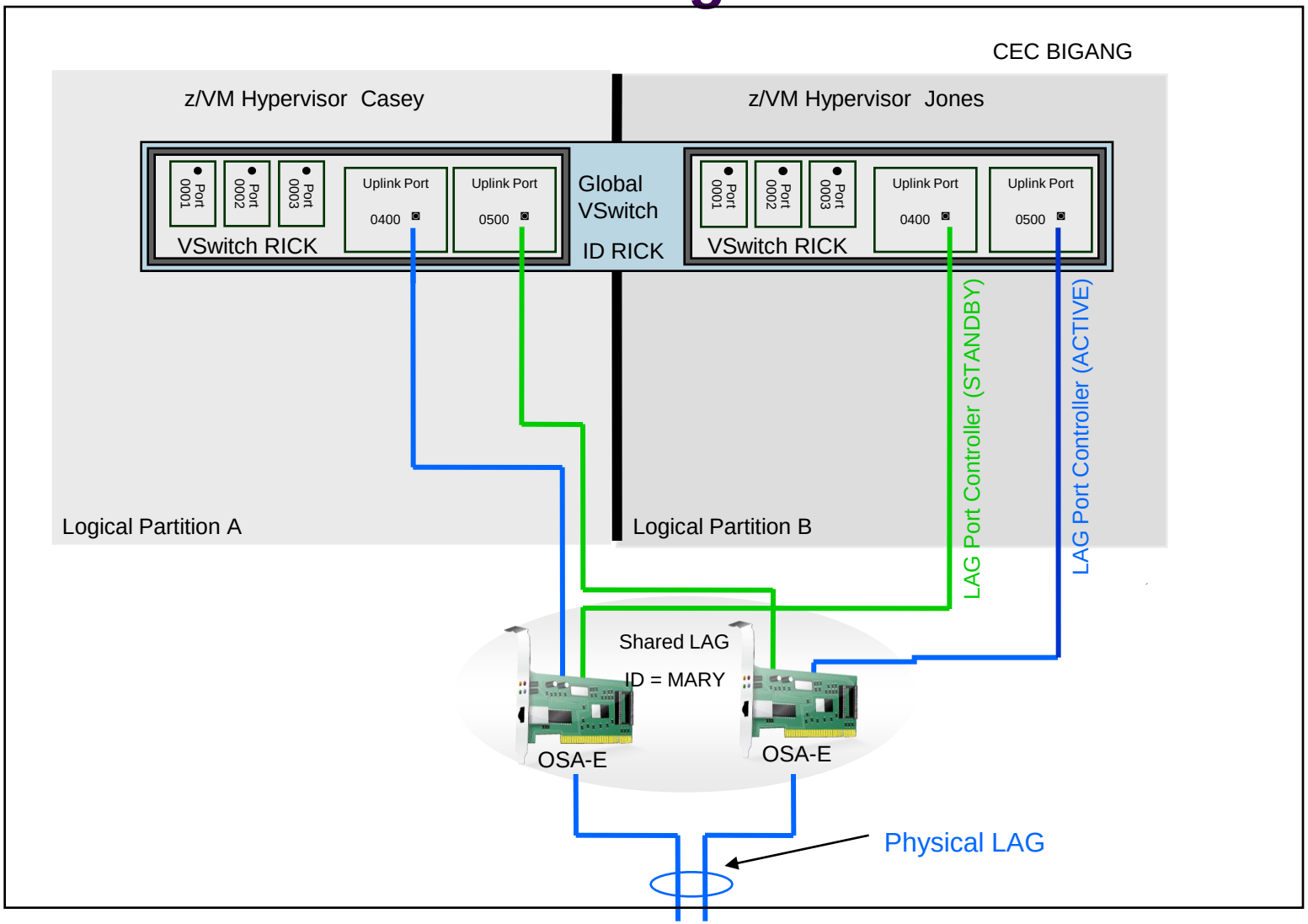
# New Attributes…

- **Shared port group**
  - Set of OSA-Express features configured to be shared by one or more global virtual switches and global virtual switch members
    - Can only be used with global virtual switches and members
  - Provides connectivity across multiple virtual switches and multiple z/VM systems
  - Propagated to all systems in the same IVL domain
  - Single point of control for all virtual switches using the shared port group

**SET PORT GROUP MARY LACP ACTIVE SHARED**

**SET PORT GROUP MARY JOIN 400 500**

# Multi-VSwitch LAG Configuration



# Crypto Enhancements

# Crypto Express5S

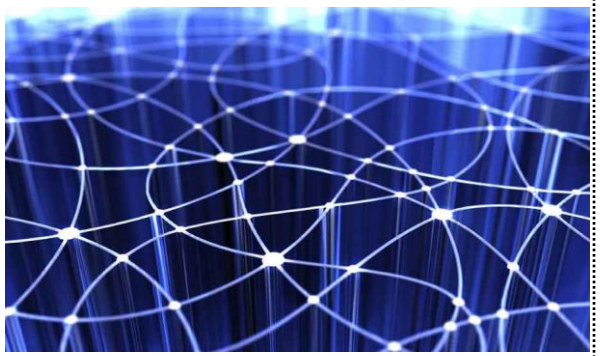
- Available on IBM z13
- Can be configured one of 3 ways
  - Shared or dedicated access
    1. IBM Common Cryptographic Architecture (CCA) coprocessor
    2. Accelerator
  - Dedicated access only
    3. IBM Enterprise Public Key Cryptographic Standards (PKCS) #11 (EP11) coprocessor
- Enhanced domain support
  - z13 supports up to 16 APs and up to 85 domains per AP
  - z/VM supports architected increases in
    - Maximum number of crypto features (APs) from 64 to 256
    - Maximum number of domains per AP from 16 to 256

# Crypto Express5S

- Supported for z/Architecture guests
  - Authorized in directory (**CRYPTO** statement)

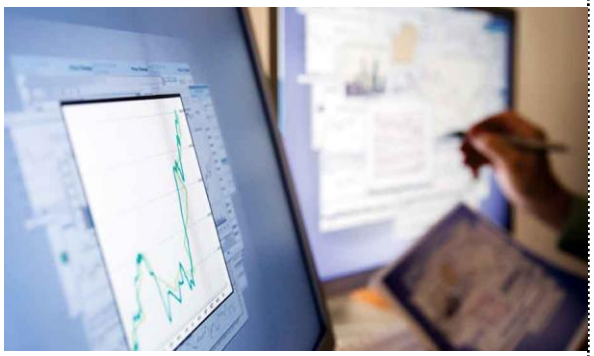
- New interface to specify APs and domains for shared use
  - **CRYPTO APVIRTUAL** system configuration statement
    - Assigns all specified domains that are available to the shared-use pool
  - If **CRYPTO APVIRTUAL** is not specified, maximum of two domains are used for sharing

# Summary



## Leadership

z/VM continues to provide additional value to the platform as the strategic virtualization solution for z Systems. Virtual Switch technology in z/VM is industry leading.



## Innovation

z/VM 6.3 added HiperDispatch, allowing greater efficiencies to be realized. Now the adding SMT with topology awareness raises the bar again.

## Growth

z/VM 6.3 increases the vertical scalability and efficiency to complement the horizontal scaling introduced in z/VM 6.2, because we know our customers' systems continue to grow. This year we continue to extend the limits with processor scalability improvements.

# Additional Information

- z/VM 6.3 resources
  - http://www.vm.ibm.com/zvm630/
  - http://www.vm.ibm.com/zvm630/apars.html
  - http://www.vm.ibm.com/events/
  - http://www.vm.ibm.com/service/vmreqz13.html

- z/VM 6.3 Performance Report
  - http://www.vm.ibm.com/perf/reports/zvm/html/index.html

- z/VM Library
  - http://www.vm.ibm.com/library/

- Live Virtual Classes for z/VM and Linux
  - http://www.vm.ibm.com/education/lvc/

# *Thanks!*

John Franciscovich
IBM
z/VM Design and Development
Endicott, NY
francisj@us.ibm.com

***Session 16453***