35 YEARS OF PERFORMANCE MANAGEMENT

SMT for z/VM
Understanding and Using

- Barton@VelocitySoftware.com
- HTTP://VelocitySoftware.com

"If you can't Measure it,
I am Just Not Interested ™"

VELOCITY SOFTWARE

PROVEN PERFORMANCE

SMT Topics

SMT Theory – Capacity increases

Data Validation / Capture Ratios

Capacity Planning – what does SMT add?

- (Capacity always increases)

Chargeback – what are the metrics? Accurate?

SMT Theory

SMT is about using unused cycles

If one thread

- Cycles wasted waiting for L1/L2 cache update
- Cycles wasted waiting for DAT (Dynamic Address Translation)

If two threads

- Wasted cycles could be used by an alternate thread
- If there is contention for cache or DAT, work takes longer!
- Is there an increase in capacity?
- What is the performance impact?

SMT Theory

SMT Objective:

- Increases capacity (maybe at the cost of performance / response time)
- Better core utilization (more cycles for real work)

In theory: Processor cycles are sitting idle

- To execute an instruction, L1 cache is populated (data/instruction)
- Cycles wasted while L1 cache is loaded from L2/L3/L4/Memory
- SMT uses “wasted” cycles for another “thread”

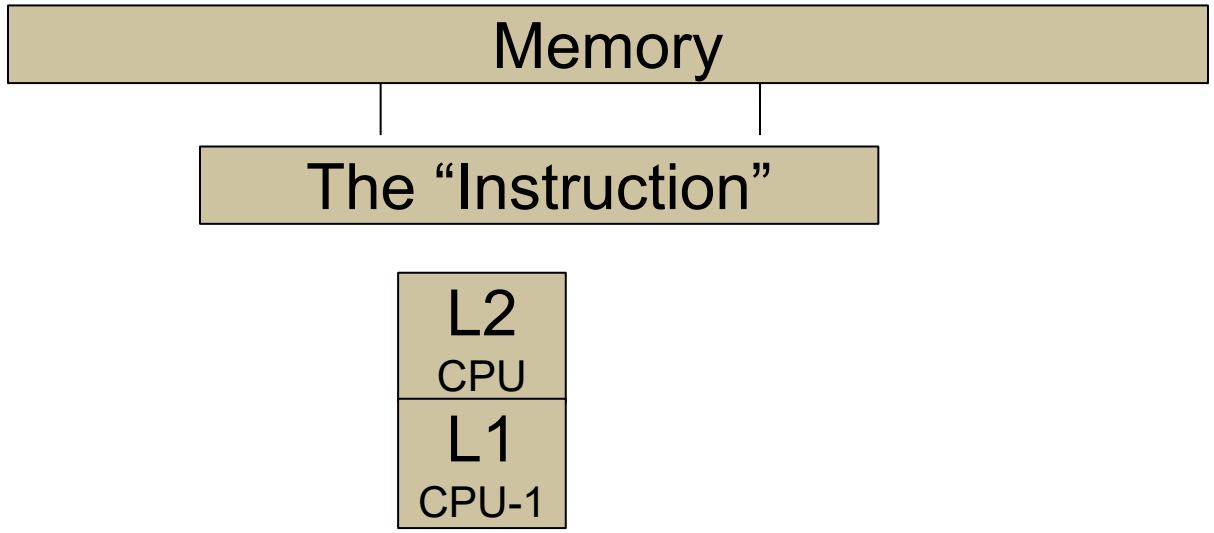
In practice:

- Two threads share one core – and cache
- More processes share core – and cache
- Cache has more contention
- Core has contention
- BUT, wasted cycles are now being used

SMT Theory – Why Cache is important

Hardware – the way it works

- Instructions executed on the CPU, from the L1 Cache
- Same on INTEL
- IBM z Implements levels of cache



z/VM Challenges

SMT on z/VM has challenges

- Why is SAP/Oracle better for SMT? (z13)
- A 30% ITR (Internal Throughput Rate) improvement with SMT in SAP production LPAR
- Why would z/OS do better with SMT? (It doesn't anymore)

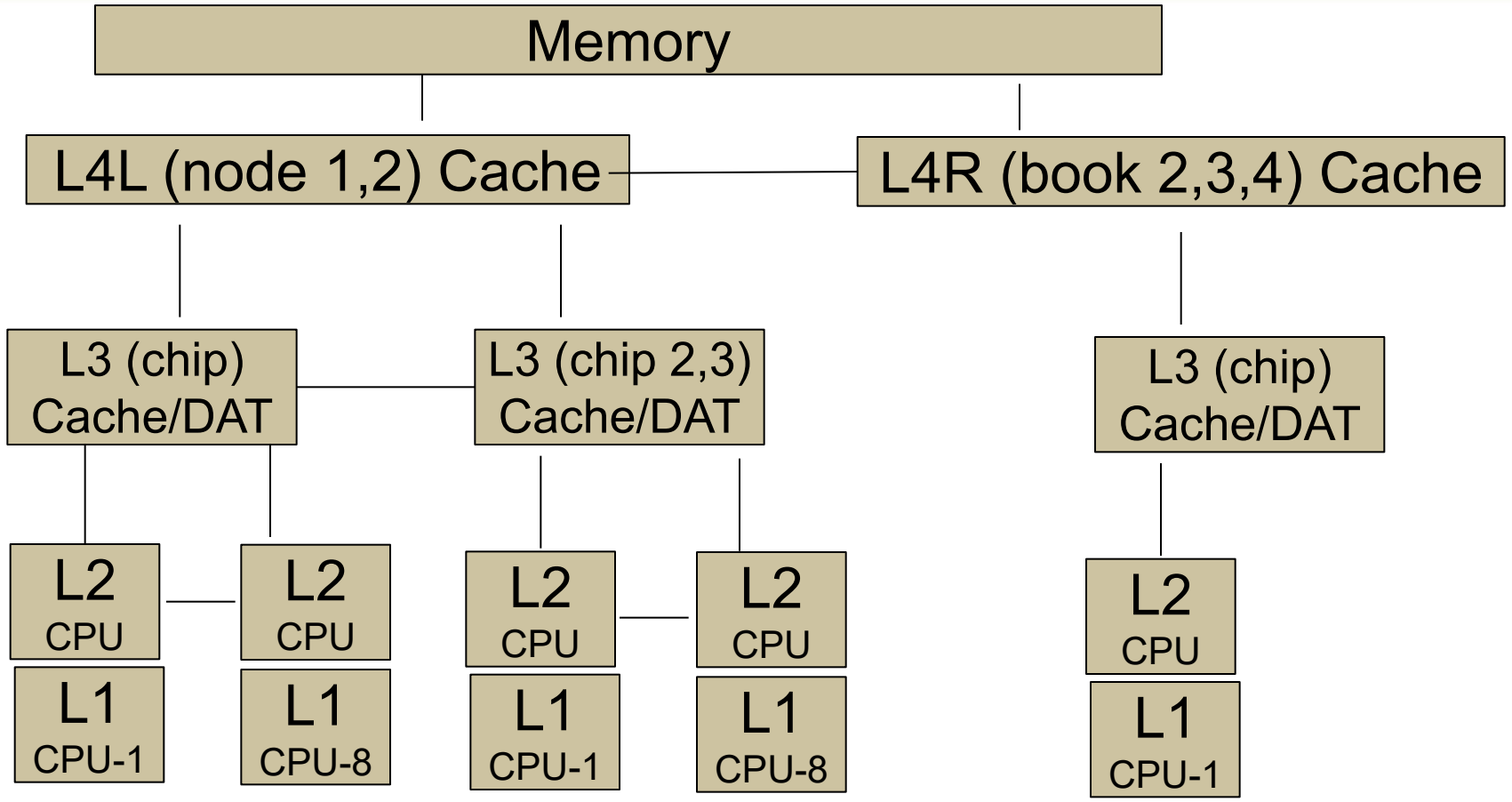
Dispatching 30,000 times per second on one thread

- How long is the task on CPU? (<30 microseconds)
- 30 microseconds -> 15,000 cycles – 5k instructions?
- How long does data remain in L1/L2 cache?
- The more references further out, the worse things get

Relative Nest Intensity – RNI (John Burg, WSC)

- Provides relative wait times
- Smaller means less time waiting for cache to be loaded

Why SMT?



For instructions to execute in CPU, must be in L1 cache
It takes time to load cache, no instructions execute during load

Cycle Requirements Per Source

SMT Analysis starts with Cycles Per Instruction

- CPU clocked at 5.2GHz, so 5.2 Billion CYCLES per second
- Instructions take 1-n cycles. (Less when pipelined)
- If data/instructions not in L1 cache, they get loaded (1-30 cycles wasted)
- Capacity is based on instructions executed
- Using “wasted” cycles for real instructions increases capacity

- Z13 was limited by address translation taking 30-50% of the cycles
- z14+ has 4 DAT units per core

- Objective is more instructions per given cycle

Cycle Requirements Per Source

What happens to RNI when a 2nd thread is added? (z13)

- Relative Nest Intensity derived by John Berg, WSC for z/OS as capacity measureg
- Average CPI (Cycles Per Instruction) went from 1.25 to 1.40 so some degradation)
- Average RNI (Relative Nest Intensity) went from .55 to .66 (cache contention)

Report: **ESAMFCA** MainFrame Cache Magnitudes R

		<CPU Busy>		<-------Processor------>				**RNI**
		<percent>		Speed/<-Rate/Sec->			**CPI**	From
Time	CPU	Totl	User	Hertz	Cycles	Instr	**Ratio**	**Burg**
09:47:00	0	10.9	10.6	5208M	569M	454M	1.254	0.53
09:48:00	0	11.9	11.6	5208M	621M	523M	1.187	0.42
09:49:00	0	9.3	9.0	5208M	487M	385M	1.265	0.56
09:50:00	0	9.5	9.2	5208M	497M	391M	1.270	0.54
09:51:00	0	9.5	9.1	5208M	497M	380M	1.309	0.65
09:52:00	0	10.0	9.5	5208M	520M	373M	1.394	0.62 ←SMT Enabled
09:53:00	0	11.2	10.8	5208M	587M	448M	1.312	0.48
09:54:00	0	9.8	9.3	5208M	512M	365M	1.403	0.68
09:55:00	0	10.5	10.0	5208M	550M	390M	1.411	0.66

CPU Measurement Facility with SMT

- Cycles by thread (total cycles used for both work and wait)
- Shows cycles used and instructions executed (thread CPI)
- Core CPI went down (instructions/capacity went up)
- Meaningful instructions per second – total (3.33G)
- Real cycles per instruction: 88% of 5208M / 3330M (1.37)

```
Report: ESAMFC          MainFrame Cache Magnitudes
Monitor initialized: 06/17/20 at 21:23:09 on 390
-----------------------------------------------
                <CPU Busy><-------Processor------>
                <percent> Speed/<-Rate/Sec->
Time      CPU Totl User Hertz Cycles Instr Ratio
-------- --- ---- ---- ----- ------ ----- -----
21:25:02   0 88.4 74.5 5208M  4607M 1652M 2.789 -> 1.37
           1 88.6 76.9 5208M  4617M 1678M 2.752
```

Back to – What is a CPU second?

- We charge for CPU seconds?
- Is it consistent? No!
- How much does it vary? (in instructions per second)
- Dependent on workload (cache residency)
- If more contention for cache, more time is spent waiting

SMT Data Points

System data points – hardware perspective

- Core time allocated to LPAR (both threads at a time)
- Thread busy vs thread idle (potential capacity)
- Instructions per second per core
- Cycles per instruction (low is good)
- Impacts of the LPAR definition

User data points

- Core time and thread time
- Change in thread time (response time)
- Change in cycles consumed (capacity)
- Does the data agree?

SMT – When to use it?

SMT was announced on z13 without much guidance

Some installations said "good stuff"

- Oracle, SAP workloads

Others said "not so good…"

- Java, Websphere workloads

The question is why?

And why is z14 (and up) so much better?

Does SMT provide more capacity?

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

**Illustrative numbers only*

Measurement:

- ”Person miles”?
- Per Minute?

Add lanes and…?

Does SMT Provide Contention?

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

**Illustrative numbers only*

(z/13) Not always faster…

Capture Ratios – z/VM – NO SMT

Compare LPAR (SYTCUP) to z/VM (SYTPRP): Capture 99%

- CPU by CPU comparison accurate
- Some scheduling time likely lost

ESACAPT Logical Partition Analysis

<----Logical Processor--->		<---CPU (percentages---->						Capture%
VCPU Addr	CPU Type	<---%Assigned--> Total	Ovhd	Emul	Total util	Emul time	User ovrhd	Sys ovrhd	LPAR -
0	IFL	15.7	0.5	15.2	14.9	12.0	1.3	1.6	0.98
1	IFL	18.8	0.5	18.3	17.9	16.0	1.5	0.5	0.98
2	IFL	20.7	0.4	20.3	20.0	18.1	1.4	0.5	0.98
3	IFL	25.1	0.4	24.7	24.4	22.5	1.5	0.4	0.99
4	IFL	27.2	0.4	26.8	26.5	24.6	1.4	0.5	0.99
5	IFL	38.4	0.4	38.0	37.7	35.5	1.7	0.6	0.99
6	IFL	64.8	0.6	64.3	64.0	60.4	2.8	0.8	1.00
7	IFL	1.1	0.2	0.9	0.7	0.1	0.1	0.5	0.76
8	IFL	0.8	0.0	0.7	0.7	0.6	0.0	0.1	0.95
Total	IFL	212.6	3.3	**209.3**	**206.9**	189.8	11.6	5.4 6	**0.99**

SMT Base Lines

Processor Utilization – No SMT

- Numbers agree and make sense
- Can capture virtual machine resources and believe it
- Have value for overheads

SMT Challenges

- Virtual machines share the CPU/core
- The more they share, the slower they go (in theory, how slow?)
- Numbers are likely not repeatable based on workload
- How much added capacity with SMT for YOUR workload?
- How do you charge?
- (You must charge for consumption)

Processor Capacity Planning Concepts

Processor utilization – what level is target?

- Performance – what level of performance is required?
- What level of performance management is required? Is available?
- Capacity Planning – what utilization level is needed financially?

Customer targets

- Target based on performance?
- 80% and higher hardware utilization requires management
- 50% CPU minimizes CPU queue – better performance – tradeoff
- Higher utilization is better financially

Capacity planning objective

- Provide resources to get work done in a timely fashion
- Meeting appropriate financial and performance objectives

Processor Measurement Concepts - Utilization

What is "CPU Utilization"? Need to agree on this first?

All zVPS numbers are measured in CPU seconds

- Percent is always based on CPU seconds divided by wall clock
- What is a CPU second if there are two threads with SMT?

Impacts the measurements of:

- LPAR (percent of processor assigned to the partition)
- z/VM Virtual Machines (percent of "thread" assigned to a virtual machine)
- Linux processes (percent of a vCPU)

BUT DO WE AGREE ON WHAT IS IMPORTANT?

- Is it processor utilization?
- Or work completed?

Capacity Benefits?

SMT adds how much capacity?

- How much more throughput?
- Workload dependencies
- How to predict

Z13/14/15/16/17 have larger cache sizes

- How long does cache last when 30,000 dispatches per second per processor?
- How much does enabling SMT impact cache?

Capacity Planning Thoughts

How much used capacity at the CEC level?

- Total IFL (Assigned) Utilization (ESALPARS/ESALPMGS)
- Totals by Processor type
- Shared processor total busy

```
<---------CPU-------> <-Shared Processor busy->
Type Count Ded shared  Total  Logical Ovhd Mgmt
---- ----- --- ------ ------ -------- ---- ----
CP      11   0     11  892.1    865.2 11.2 15.7
IFL     37   6     31 2466.7   2412.0 30.9 23.8  ← 80% utilization
```

Capacity Planning Thoughts

z/VM: One core – Two threads

- "Assigned" – 933.7% or 4.1%
- Two threads are not always both active –> thread idle time
- Subtract 138% thread idle -> (933% - 4) * 2 – 138% = **1720**% thread time (z/VM time)
- (Thread idle time is not necessarily excess capacity)

			<--------Logical Partition------->					
			Virt	CPU	<%Assigned>		<-Thread->	
Time	Name	Nbr	CPUs	Type	Total	Ovhd	Idle	cnt
21:25:00	Totals:	00	27	CP	876.3	11.2		
	Totals:	00	54	IFL	2443	30.9		
	ZVMQAXX	0B	**14**	IFL	**933.7**	4.1	**138.1**	2 ←

Capacity Planning Thoughts

Report: ESACPUU CPU Utilization Report

Time	<Users> Actv	In Q	Tran /sec	CPU	CPU Type	Total util	Emul time	User ovrhd	Sys ovrhd	Idle time	Steal time
21:25:00	194	399	0.5	0	IFL	88.4	74.5	1.7	12.2	10.5	1.1
				1	IFL	88.6	76.9	1.9	9.8	10.3	1.1
				2	IFL	89.2	77.7	2.4	9.1	9.7	1.1
				3	IFL	89.2	77.7	1.5	10.1	9.6	1.2
				4	IFL	89.6	78.0	1.7	9.9	9.4	1.0
				5	IFL	89.1	77.7	2.3	9.1	9.9	1.1
				22	IFL	67.2	58.5	1.5	7.1	11.6	21.2
				23	IFL	66.8	58.4	1.4	6.9	12.0	21.3
				24	IFL	74.9	66.4	1.5	7.0	13.9	11.1
				25	IFL	74.4	66.3	1.6	6.5	14.4	11.2
				26	IFL	76.4	68.3	1.3	6.8	12.6	10.9
				27	IFL	75.6	68.2	1.6	5.8	13.4	11.0
System:						1709	1499	36.2	173.6	332.2	759.1

How much used capacity in the z/VM LPAR?

- Total IFL Utilization (ESACPUU) 1,709% (capture 99%+)
- User billable – Traditional: (1499 + 36) – 1,535%?
- Steal time: Physical processor stolen

Processor Measurements - SMT

ESAUSR5/ESAUSP5 show SMT user data (raw/percent) - Three CPU measures:

- Traditional: Time assigned and dispatched on a thread
- MT-Equivalent: Time it would take if non-SMT (performance)
- MT Prorated: Cycles really used (estimated) for Capacity and Chargeback

What if some workloads perform better as non-SMT?

- Should you have a “performance LPAR”?

Report: **ESAUSR5** User SMT CPU Consumption Analysis

	<----Raw CPU Seconds Consumed (Total)---->					
UserID	<**Traditional**>		<MT-Equivalent>		<**MT Prorated**>	
/Class	Total	Virt	Total	Virtual	Total	Virtual
10:32:00	660.4	641.7	476.0	462.5	432.0	420.0
User Class Analysis						
TheUsers	660.2	641.6	475.9	462.4	431.9	419.9
CPU POOL User Analysis						
DB2	15.63	15.42	**12.13**	11.97	**12.23**	12.09
EEMSCSP	9.03	8.97	6.91	6.87	6.59	6.55
IIB	498.7	488.6	360.4	353.2	321.8	315.4

SMT – Not Always a Good Thing?

Workload helped by SMT? Is monitor user data valid?

- 1535% “thread time” (validated against CPU busy)
- 1192% core time
- 1051% “would be” time
- Used 1192% - could have been 1051
- Based on user data, less capacity because of SMT? (13%)
- But the hardware said 933% assigned and that data is validated
- And still there is thread idle – how to account for that?

```
Report: ESAUSP5       User SMT CPU Consumption Analysi
Monitor initialized: 06/17/20 at 21:23:09 on 3906 ser
-------------------------------------------------------
          <------CPU Percent Consumed   (Total)---->
UserID    <Traditional> <MT-Equivalent> <MT Prorated>
/Class    Total  Virt   Total  Virtual  Total Virtual
-------- ----- -----   -----  -------  ----- -------
21:25:00  1535   1499   1051     1026   1192    1163
```

Processor Measurements for SMT Validity

ESAUSR5/ESAUSP5 show SMT user data

- Traditional: Thread time (response time)
- Equivalent: Time it would take if non-SMT
  - (**PERFORMANCE ratio 1051 / 1535**) – 50% slower
- Prorated: Cycles really used (approximate/prorated)
  - (**Capacity and Chargeback**)
  - Want to charge for 933% (physical assigned time to LPAR)
  - Prorated metrics are too high (1192 / 933)

```
<------CPU Percent Consumed    (Total)---->
UserID   <Traditional> <MT-Equivalent> <MT Prorated>
/Class   Total  Virt   Total  Virtual  Total Virtual
-------- ----- -----   -----  -------  ----- -------
21:25:00  1535   1499   1051     1026   1192    1163
```

Processor Measurements – Valid Data?

ESAUSP5:

- CPU Percent Consumption:
  - Total for all users
  - By User
  - By Class

LPAR Assigned Time: 933.7%

z/VM Thread assigned time: 1720%

User time: (1499 + 36 = 1535)

- Traditional measurements are valid
- 100% capture ratio
- IBM SMT prorated numbers 30% off?
- Watch for “Velocity Prorates” next

```
Report: ESAUSP5      User SMT CPU Consumption Analysis
Monitor initialized: 06/17/20 at 21:23:09 on 3906 seri
-------------------------------------------------------
         <------CPU Percent Consumed    (Total)---->
UserID   <Traditional> <MT-Equivalent> <MT Prorated>
/Class   Total  Virt   Total  Virtual  Total Virtual
-------- ----- -----   -----  -------  ----- -------
21:25:00  1535  1499    1051     1026   1192    1163
 ***User Class Analysis***
Servers   0.04  0.00    0.03     0.00   0.03    0.00
ZVPS      2.38  1.57    1.66     1.04   2.14    1.37
TheUsers  1532  1497    1049     1025   1189    1162
-------------------------------------------------------
```

```
                <Processor>  Captur
                Utilization   Ratio
Time            Total Virt.  (pct)
--------        ----- -----  ------
21:25:00         1709  1499  100.00
21:26:00         1642  1438  100.00
21:27:00         1641  1381  100.01
21:28:00         1639  1329   99.99
21:29:00         1561  1332  100.00
21:30:00         1528  1305   99.99
**********************
Average:         1629  1389  100.00
```

Processor Measurements – SMT Validity

ESALPARS	ASSIGNED	
ZVMQA00	**933.7**	4.1
ZVMQA00	897.6	4.2
ZVMQA00	908.8	5.6
ZVMQA00	905.1	5.9
ZVMQA00	883.2	7.4
ZVMQA00	873.5	8.2
ZVMQA00	894.9	7.0
ZVMQA00	915.2	4.8
ZVMQA00	901.2	5.3
ZVMQA00	917.3	6.2
ZVMQA00	906.4	6.2
ZVMQA00	923.1	6.5

ESAUSP5	THREAD		MT-PRORATED	
21:25:00	**1535**	1499	**1192**	1163
21:26:00	1477	1438	1146	1116
21:27:00	1431	1381	1115	1076
21:28:00	1382	1329	1077	1035
21:29:00	1379	1332	1081	1044
21:30:00	1350	1305	1061	1025
21:31:00	1445	1402	1129	1095
21:32:00	1469	1427	1141	1107
21:33:00	1413	1364	1097	1058
21:34:00	1452	1405	1134	1097
21:35:00	1430	1383	1117	1080
21:36:00	1454	1406	1137	1099

Compare assigned time to thread time to “prorated”

- Target is assigned time, maybe subtract thread idle
- The Velocity Prorated will be in the next release

SMT Prorate Minute by Minute

Compute ESALPARS assigned and subtract thread idle

Prorate against ESAUSP5 total and get “new” prorate interval by interval (.56 - .59)

```
ratio:  0.563289902
ratio:  0.562660799
ratio:  0.583018868
ratio:  0.603183792
ratio:  0.577411168
ratio:  0.578814815
ratio:  0.560761246
ratio:  0.578012253
ratio:  0.591224345
ratio:  0.575172176
ratio:  0.576713287
ratio:  0.576925722
```

Chargeback for SMT – Step 1

Start with ESALPARS:

- Assigned time to LPAR (1900 – Dedicated)
- Thread idle
- Time to be charged: (1900 * 2 – 1471) / 2 = 1164%
- Thread time for comparison: 1164% * 2 = 2329%

```
Report: ESALPARS
Monitor 12/12/22
-------------------------------------------------------
          <------ical Partition------->
                   Virt CPU  <%Assigned>  <-Thread->
Time      Name     CPUs Type Total  Ovhd  Idle   cnt
--------  -------  ---- ---- -----  ----  ------ ---
22:02:00  Totals:    16  CP  126.4   1.4
          Totals:    64 IFL  282.2   4.0
          VMP1O3     19 IFL   1901   0.1   1471    2
```

Chargeback for SMT – Step 2

Compare to ESACPUU to validate capture ratio

- LPAR measurement thread time: 2329%
- Time to be charged: 1164%
- CPU (thread time): 2296% (98.5% capture)
- User thread time: 2207% + 39% = 2246%
- Time to be charged from ESALPARS: 1164%

Report: **ESACPUU**

		<--------CPU (percentages			
Time	CPU	Total util	Emul time	User ovrhd	Sys ovrhd
22:02:00	0	58.1	55.9	0.9	1.3
	1	63.8	61.8	0.9	1.1
	37	57.1	54.8	1.1	1.2
System:		**2296**	**2207**	**39.0**	50.4

Chargeback for SMT – Step 3

Calculate prorate factor: ESAUSP5

- Thread time: 2246% (100% capture ratio within z/VM)
- IBM “prorated time” – 1623% is incorrect
- Time to be charged from ESALPARS: 1164%
- Prorate using “traditional” – 1164% / 2246% = .51
- Charge back factor .51 against traditional times (.51-.53)

```
Report: ESAUSP5       User SMT CPU Consumption Analysis
-------------------------------------------------------
         <------CPU Percent Consumed   (Total)---->
UserID   <Traditional> <MT-Equivalent> <MT Prorated>
/Class   Total  Virt   Total  Virtual  Total Virtual
-------- ----- -----   -----  -------  ----- -------
22:02:00  2246   2207   1643     1614   1623    1597
***Top User Analysis***
xxxDBLP5 436.1  427.9  320.3    314.2  316.4   310.9
xxQDBLP1 350.3  346.5  256.7    254.0  266.4   263.6
xxDDBLP1 314.8  309.0  224.2    220.1  203.8   200.1
xxQDBLP3 282.8  280.1  213.8    211.7  235.8   233.6
xxDDBLP3 248.7  244.2  182.8    179.5  186.9   183.6
```

CPU by Component/Function

Some "better news" from z/VM based measurements

- CPU numbers are traditional, measured by Linux (thread time)
- Virtual Machine with SMT "prorate" are lower
- IBM SMT numbers do not match reality - Same in zCX
- See https://VelocitySoftware.com/tuneguide/smt.html for more details about SMT

```
Report: ESAUSP5       User SMT CPU Consumption Analys
-------------------------------------------------------
         <------CPU Percent Consumed    (Total)---->
UserID   <Traditional> <MT-Equivalent> <IBM Prorate>
/Class   Total  Virt    Total  Virtual  Total Virtual
-------- -----  -----   -----  -------  ----- -------
07:02:00 414.9  408.0   322.7    317.3  239.7   235.8

 ***User Class Analysis***
OpenShif 355.0  350.3   276.0    272.3  204.9   202.2

 ***Top User Analysis***
RHOSCP1  142.4  140.8   110.1    108.9  82.93   82.01
RHOSCP3  125.2  123.8   97.38    96.34  72.35   71.60
RHOSCP2  86.79  85.04   68.00    66.64  49.31   48.30
```

CPU by Component/Function

Some even "better news"

- CPU numbers are traditional, measured by Linux
- VSI Prorated is based on HMC data
  - Shows SMT is significantly better

Report: **ESAUSP5** User SMT CPU Consumption Analysis

	<------CPU Percent Consumed (Total)---->				<-TOTAL CPU-->			
UserID	**<Traditional>**		<MT-Equivalent>		<IBM Prorate>		**<VSI Prorated>**	
/Class	Total	Virt	Total	Virtual	Total	Virtual	Total	Virtual
07:02:00	414.9	408.0	322.7	317.3	239.7	235.8	208.2	204.7
User Class Analysis								
OpenShif	**355.0**	350.3	276.0	272.3	204.9	202.2	**178.1**	175.7
Top User Analysis								
RHOSCP1	142.4	140.8	110.1	108.9	82.93	82.01	71.43	70.65
RHOSCP3	125.2	123.8	97.38	96.34	72.35	71.60	62.80	62.14
RHOSCP2	86.79	85.04	68.00	66.64	49.31	48.30	43.55	42.67

Expectations of SMT?

IBM Monitor data "MT Prorated" is incorrect

IBM Monitor data "MT-Equivalent" is not validated

Need a validated prorate factor

Low utilization:

- Capacity is not really an issue
- Response time should not change

High utilization – Intense workloads (SAP, Oracle):

- Capacity should see improvements
- Cache utilized well (dedicate engines…)

High utilization – Polling workloads (WAS, DB2):

- Cache competition is very very high
- Response times WILL get worse
- Capacity may drop? Validate with MFC…

Summary

SMT Capacity Planning

- Your capacity improvements are “dependent”
- Enhancements to capacity are measurable
- Evaluate each LPAR for SMT value (use CPI)
- Evaluate each server for SMT impact

SMT Chargeback

- IBM provides bogus metrics for user chargeback
- This will likely result in overcharging
- Develop an added prorate metric