IBM

Technical Forum & Executive Briefing

17 al 21 Octubre 2011

Imagine **PODER** Imagine **CAPACIDAD**

Session title – Analyzing CPU Performance

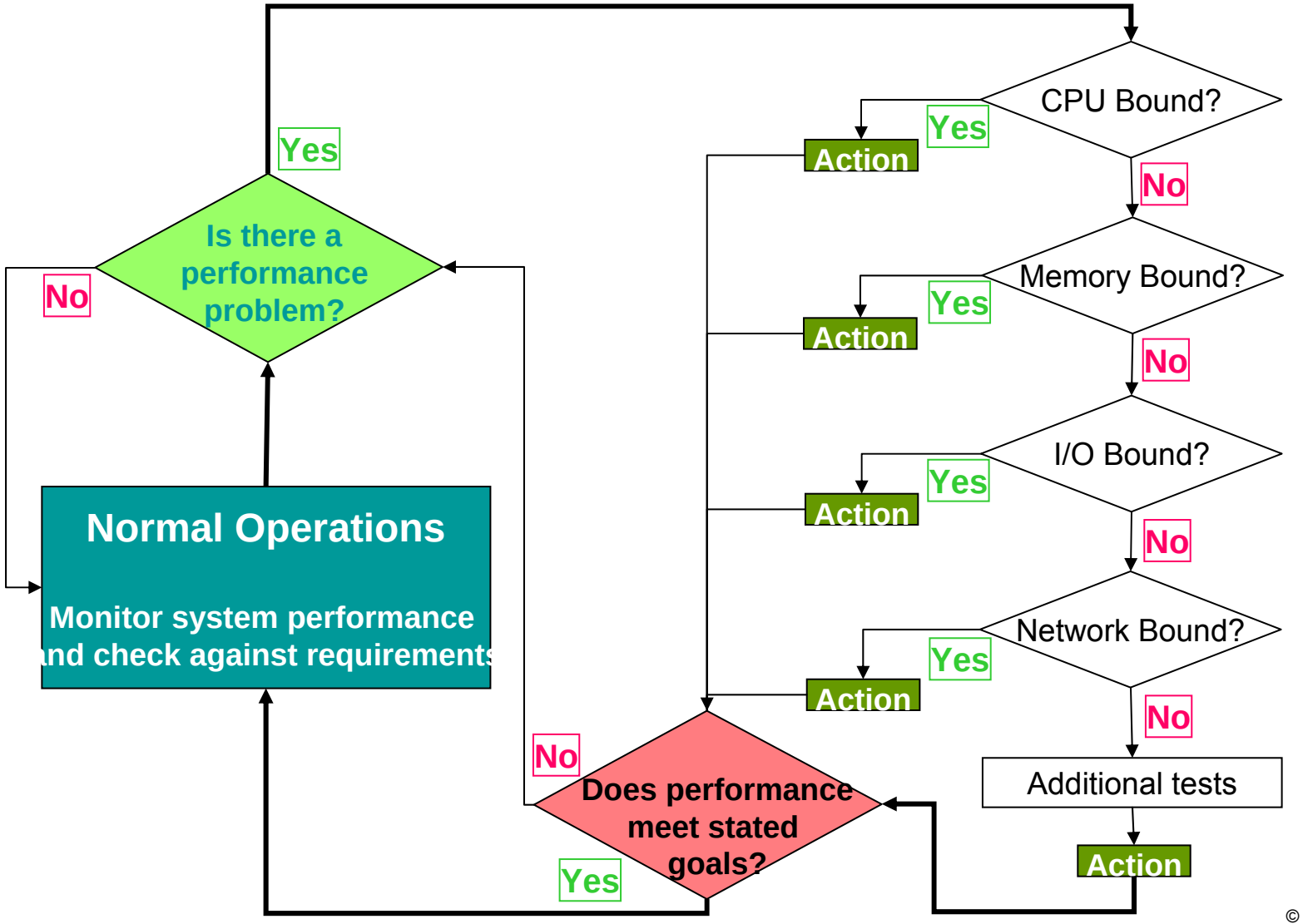
Speaker name – Scott “Tex” Nance

nancet@us.ibm.com

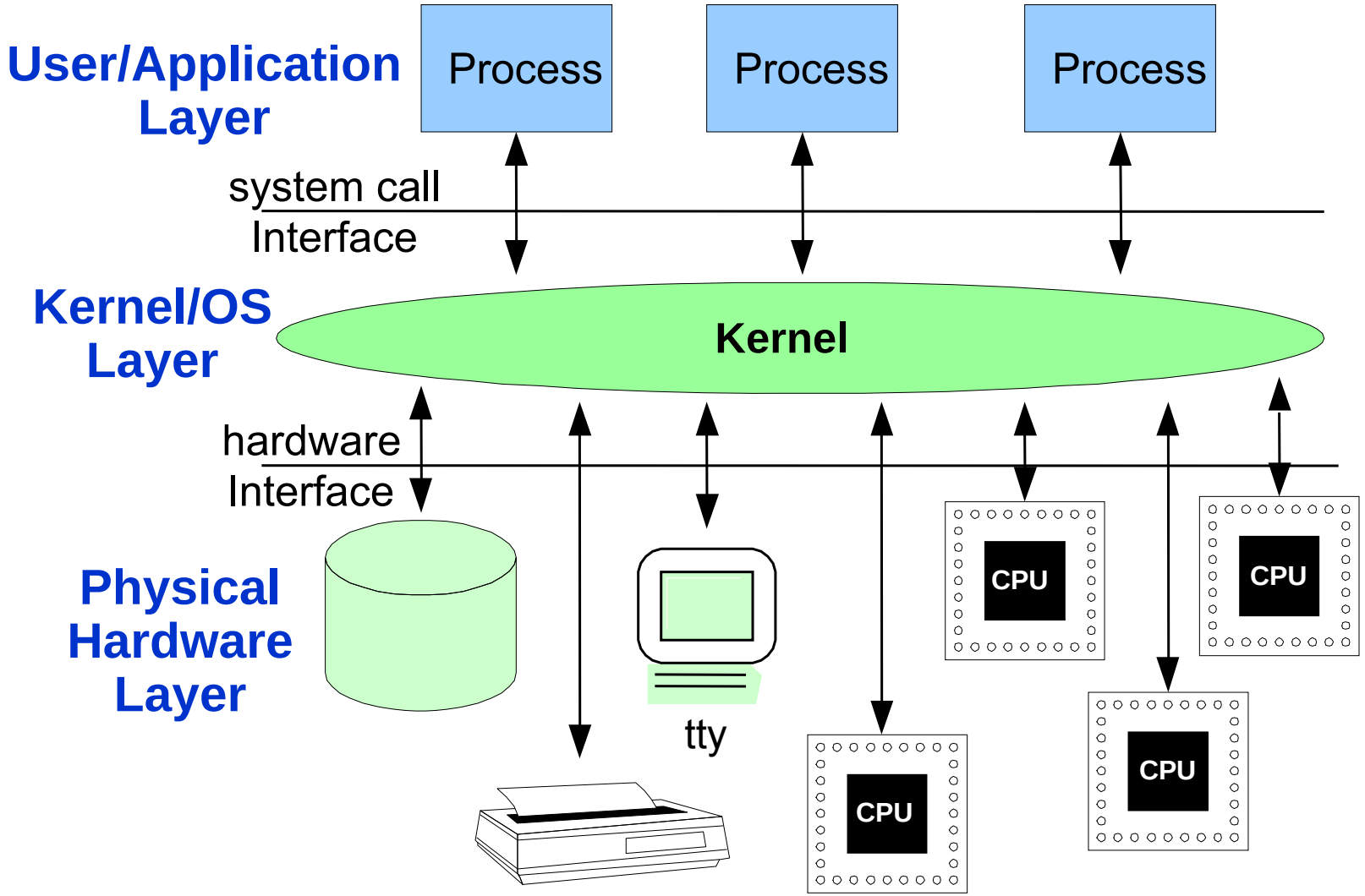
Session Objectives

- Use the output of the following AIX tools to determine symptoms of a CPU bottleneck:
  - **vmstat**, **sar**, **ps, topas, tprof,** and **curt/trace**
- Interpret the trace output and identify routines and threads being executed
- Identify causes and impacts of context switches
- Tune the Priorities of Processes

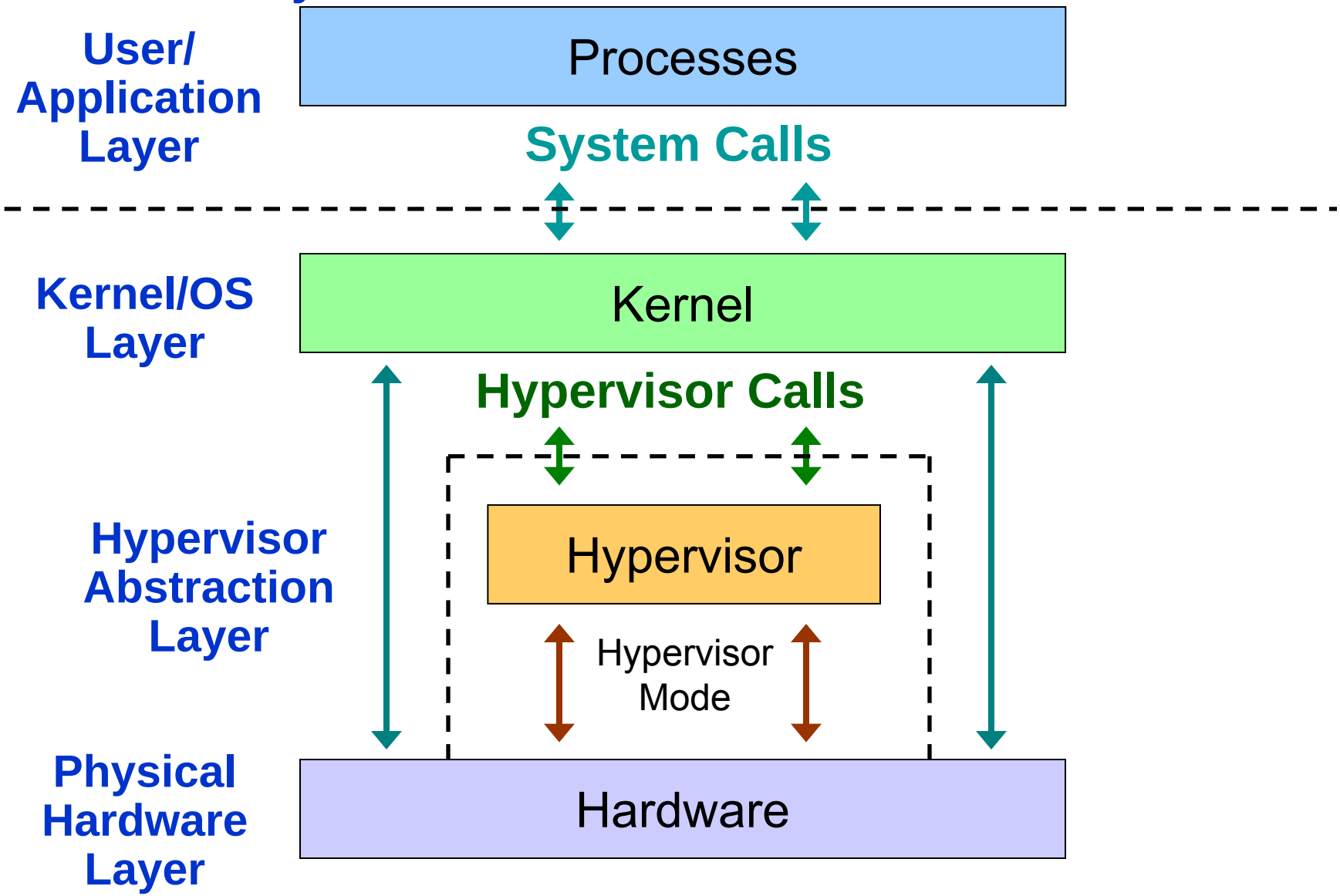
Performance Analysis Flowchart



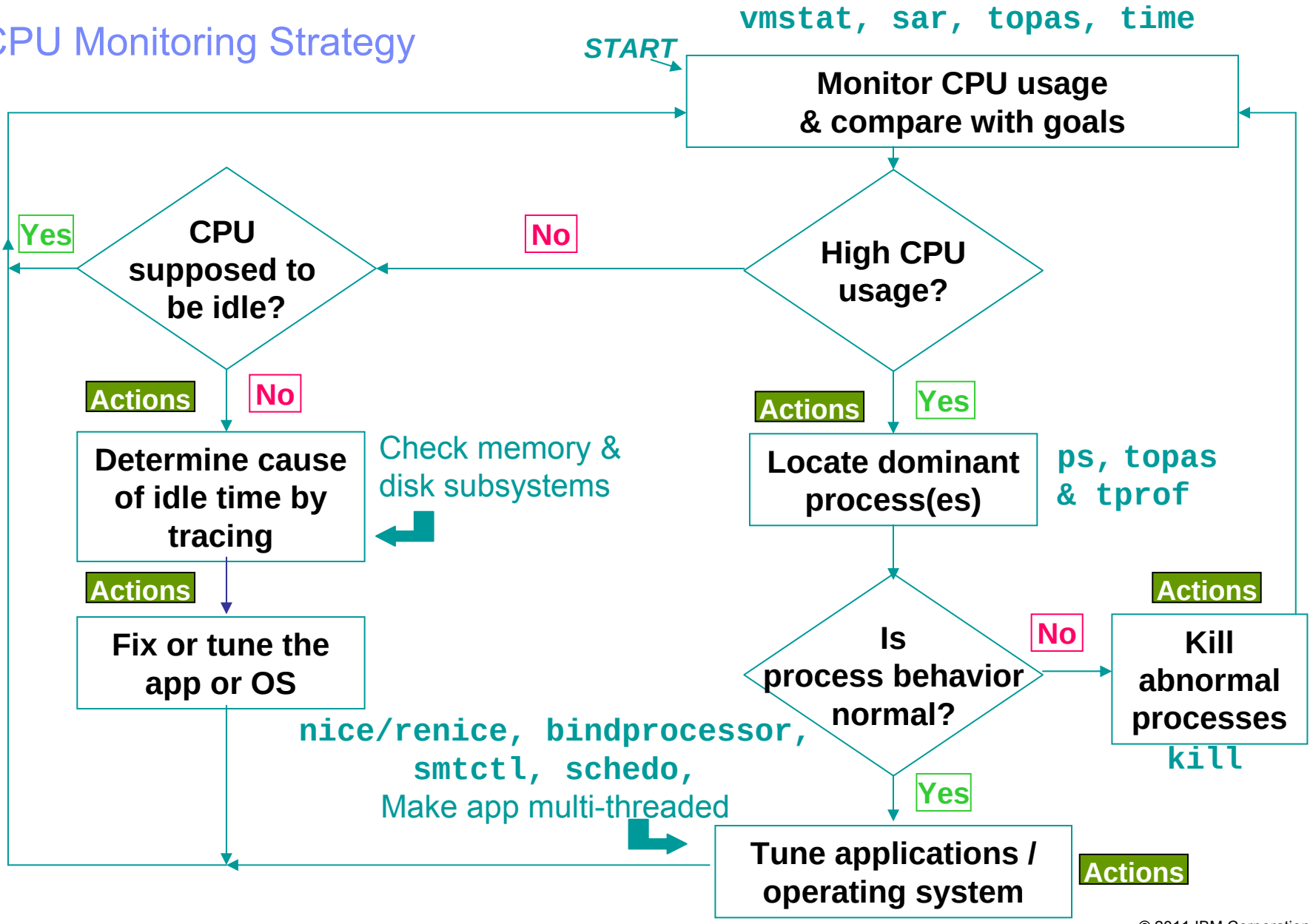
Traditional System Architecture



Partitioned System Architecture



CPU Monitoring Strategy



Monitoring CPU Usage with vmstat

```
# vmstat 5 3
System configuration: lcpu=4 mem=1024MB

kthr     memory             page                   faults        cpu
------- ------------ ----------------------- ------------ -----------
 r  b   avm   fre   re  pi  po  fr  sr  cy   in    sy   cs  us sy id wa
19  2 127005 758755  0   0   0   0   0   0 1692 10464 1070  48 52  0  0
19  2 127096 758662  0   0   0   0   0   0 1397 71452 1059  28 72  0  0
19  2 127100 758656  0   0   0   0   0   0 1361 72624 1001  28 72  0  0
```

- Runnable threads simply show total number of threads in queue:
  - High number could simply mean your system is efficiently running lots of threads
  - If the high number is abnormal, look at what processes are running and if total CPU utilization is higher than normal
- If **us** + **sy** = 100%, then there may be a CPU bottleneck:
  - Compare interrupt, system call, and context switch rates to baseline

Identifying Potential CPU Bottlenecks

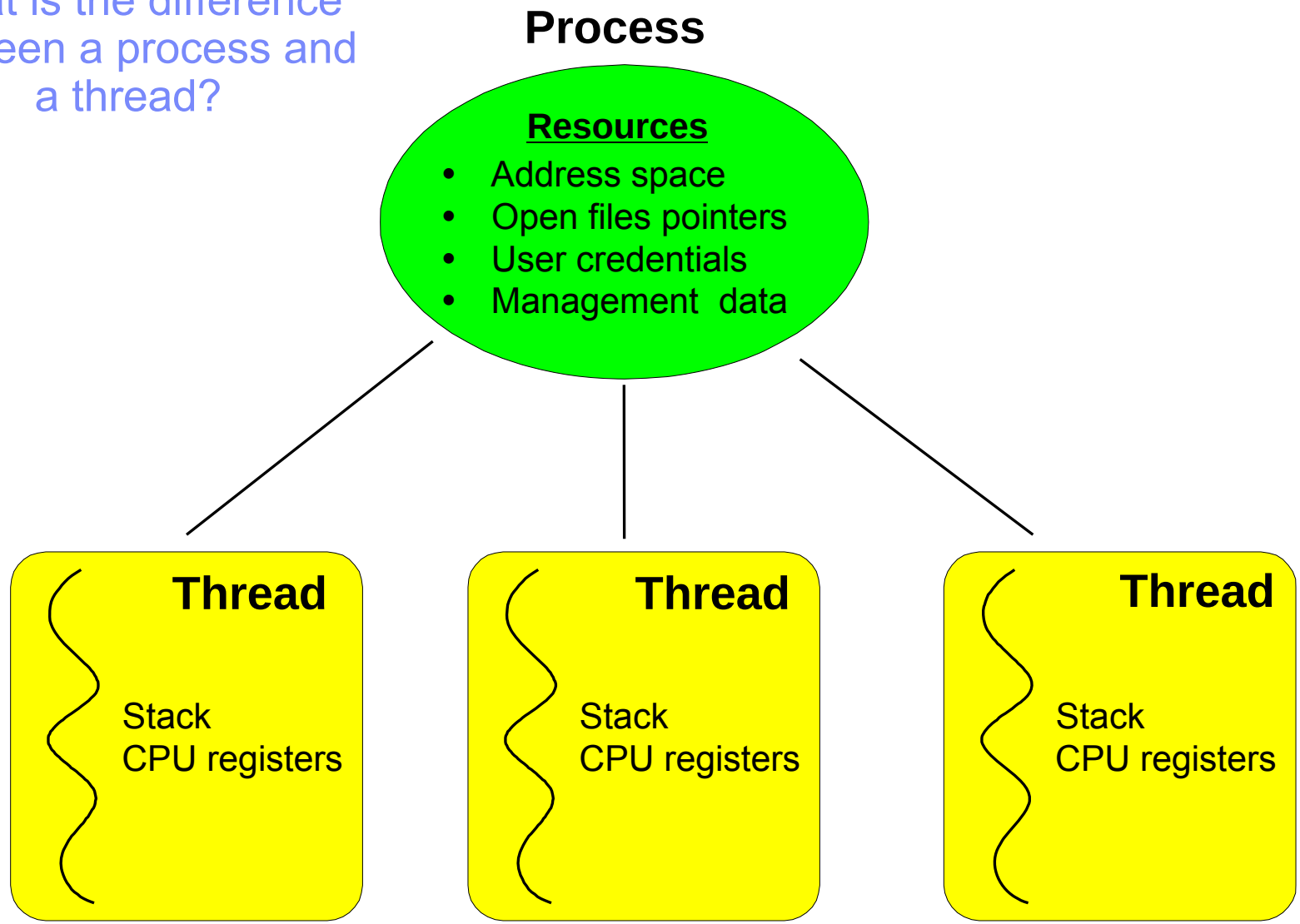
- Use **`vmstat`** to see run queue & context switches
- A context switch is when one thread is taken off a CPU and another thread is dispatched onto the same CPU
- Context switches are normal for multi-processing systems:
  - What is abnormal? Check against baseline
  - High context switch rate could be indication of lock contention
- Example:

```
# vmstat 1 5

System configuration: lcpu=2 mem=512MB

kthr     memory             page                   faults         cpu
----- ------------ ------------------------ -------------- ----------
 r  b   avm    fre  re  pi  po  fr   sr  cy    in   sy   cs  us sy id wa
33  3 257759  3371   0   0   0 102  208   0  6061 3355 13551  1 41  8 50
38  1 257760  3292   0   0   8 666 1398   0  5286 1091 23504  0 62  7 31
35  1 257760  3437   0   0   0 514 1039   0  5507 2363 11142  1 39  7 53
34  6 257760  3356   0   0   0 513  982   0  8927 1264 19310  1 55  4 40
36  2 258259  3289   0   0   0 256  516   0  7161 5649 16332  1 46  5 48
```

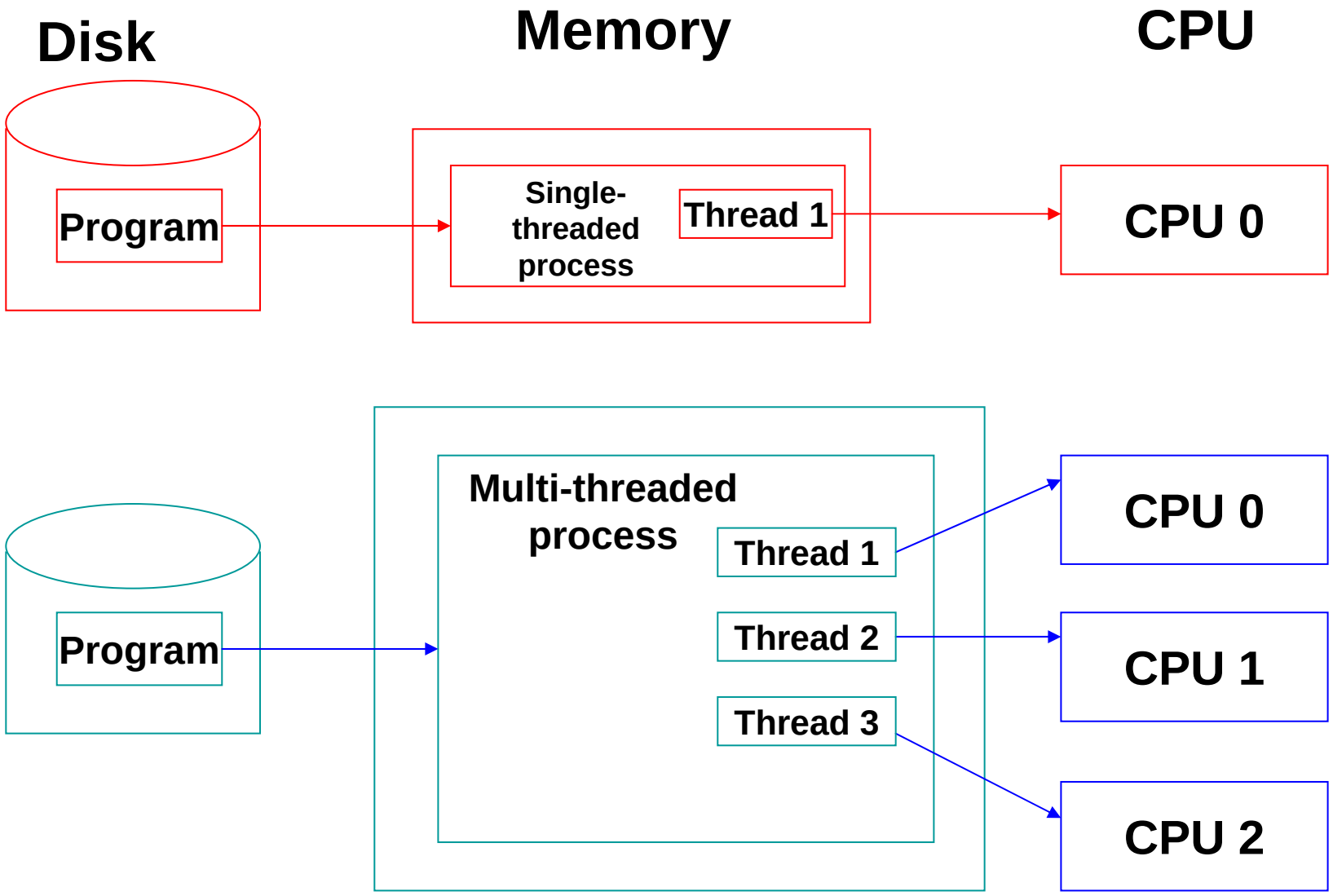
The topas Command

```
Topas Monitor for host:    woolf222            EVENTS/QUEUES    FILE/TTY
Tue Feb  3 19:43:13 2009   Interval: 10        Cswitch     165  Readch     1373
                                               Syscall  949.2K  Writech     335
CPU  User%  Kern%  Wait%  Idle%                Reads    949.3K  Rawin         0
ALL   22.4   77.6    0.0    0.0                 Writes        0  Ttyout       64
                                               Forks         0  Igets         0
Network  KBPS   I-Pack  O-Pack   KB-In  KB-Out  Execs         0  Namei         5
Total     0.2      1.0     0.4     0.1     0.1  Runqueue    1.2  Dirblk        0
                                               Waitqueue   0.0
Disk    Busy%     KBPS     TPS KB-Read KB-Writ                  MEMORY
Total     0.0      0.0     0.0     0.0     0.0  PAGING           Real,MB    1024
                                               Faults       17  % Comp     68.8
FileSystem        KBPS     TPS KB-Read KB-Writ  Steals        0  % Noncomp  11.1
Total              1.1     1.0    1.1    0.0    PgspIn        0  % Client   11.1
                                               PgspOut       0
Name            PID  CPU%  PgSp Owner           PageIn        0  PAGING SPACE
cpuprog      503892  99.7   0.1 root            PageOut       0  Size,MB     512
getty        213180   0.1   0.5 root            Sios          0  % Used      1.1
topas        262368   0.0   1.3 root                             % Free     99.9
java         204836   0.0  70.0 pconsole        NFS (calls/sec)
gil           57372   0.0   0.9 root            SerV2         0  WPAR Activ    0
java         114916   0.0  37.9 root            CliV2         0  WPAR Total    0
rpc.lock      81986   0.0   1.2 root            SerV3         0  Press: "h"-help
ksh          290824   0.0   0.5 root            CliV3         0         "q"-quit
rmcd         266382   0.0   2.5 root
aixmibd      225438   0.0   1.1 root
sendmail     217244   0.0   1.1 root
xmgc          45078   0.0   0.4 root
```

What is the difference between a process and a thread?



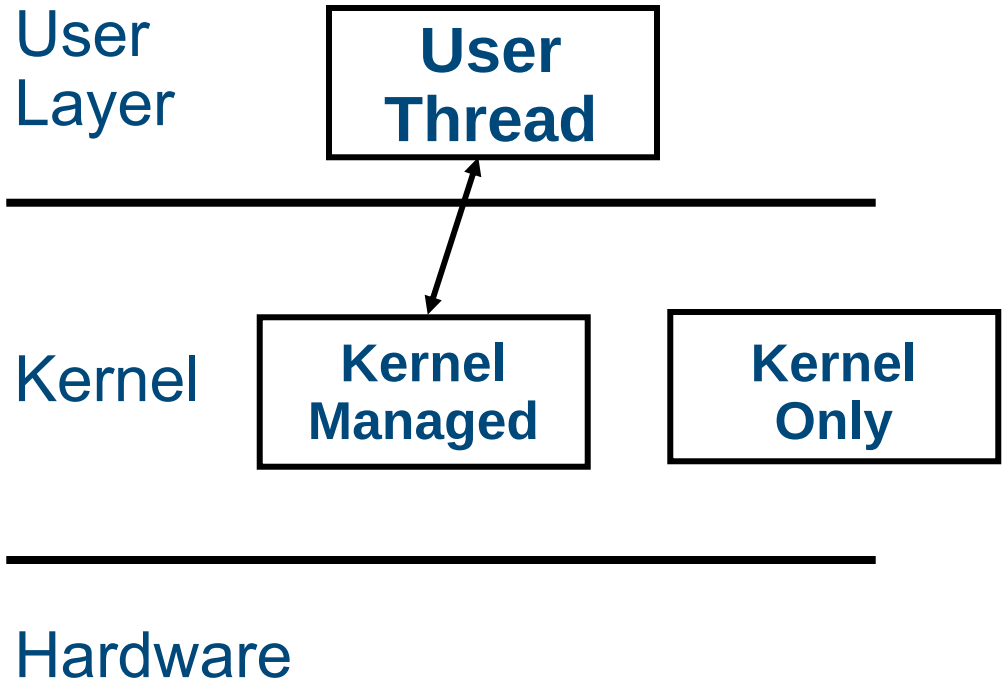
Processes and Threads



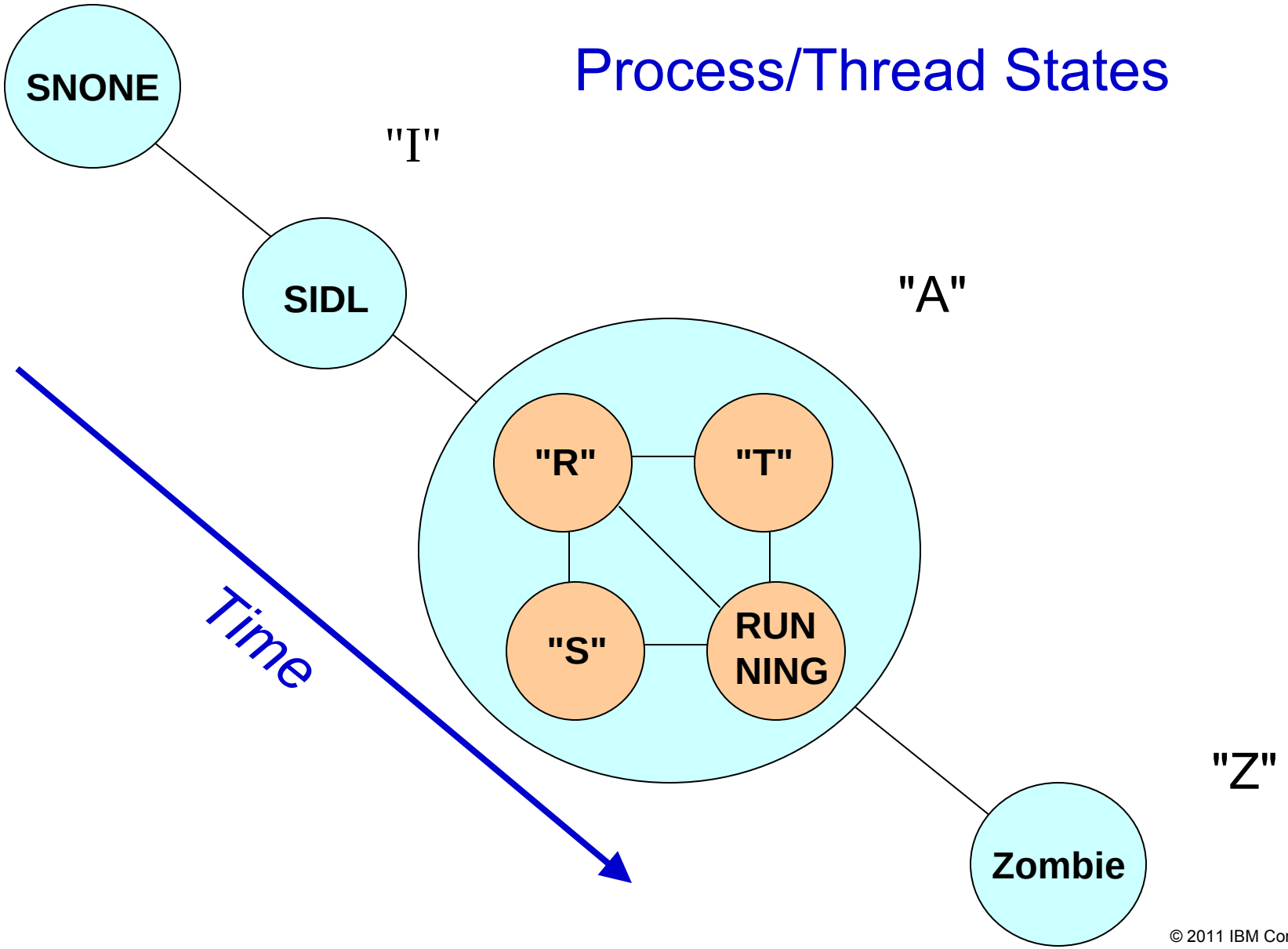
Types of Threads

- POSIX threads also called pthreads, user thread, or user-level threads
  - Kernel has no knowledge of their existence
  - Must be mapped to kernel threads for execution
- Kernel threads created specifically for user processes
- Kernel threads of kernel processes

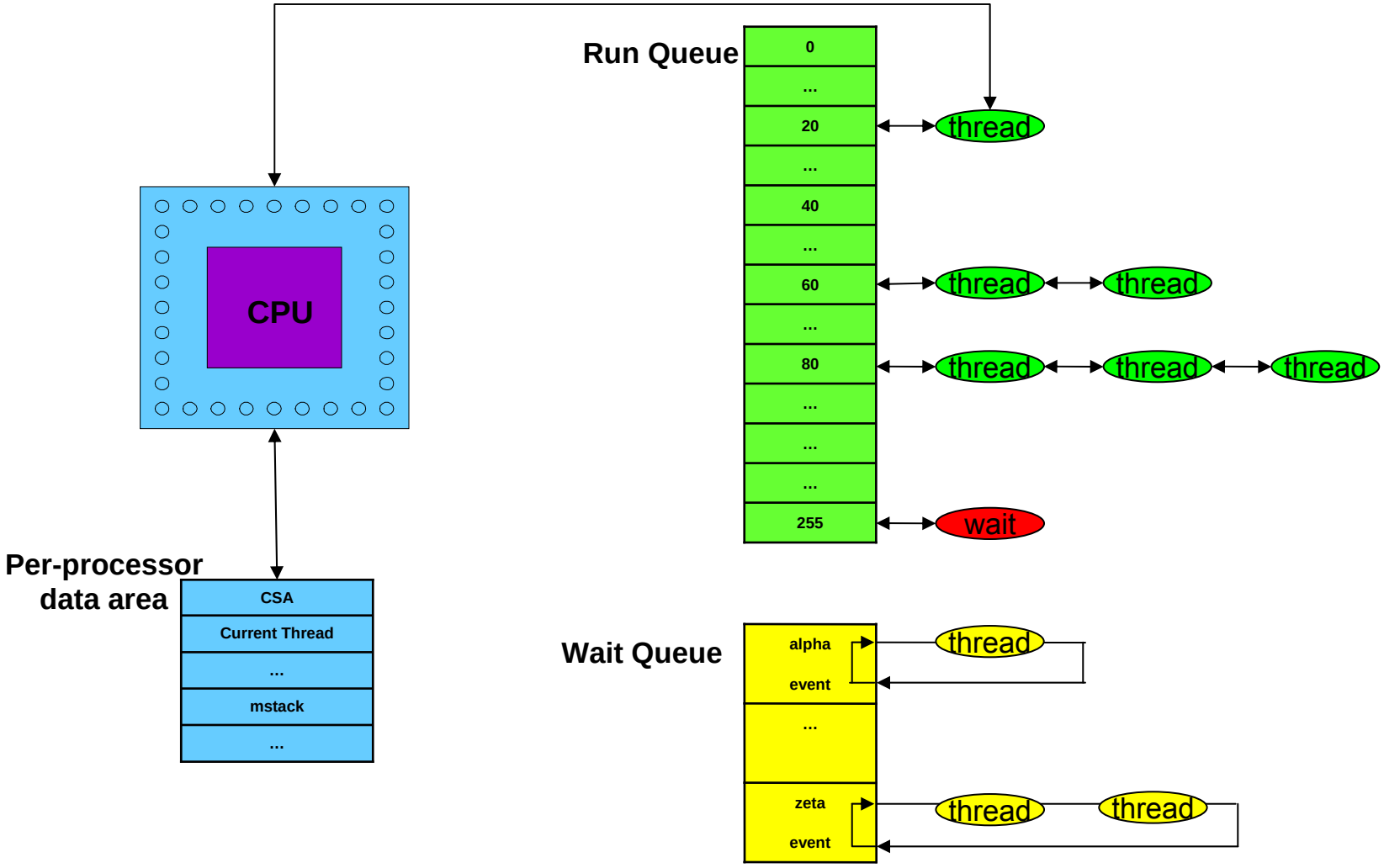
Architecural Design Example



Process/Thread States



Run Queue and Sleeping Queues



Viewing Process/Thread Priorities

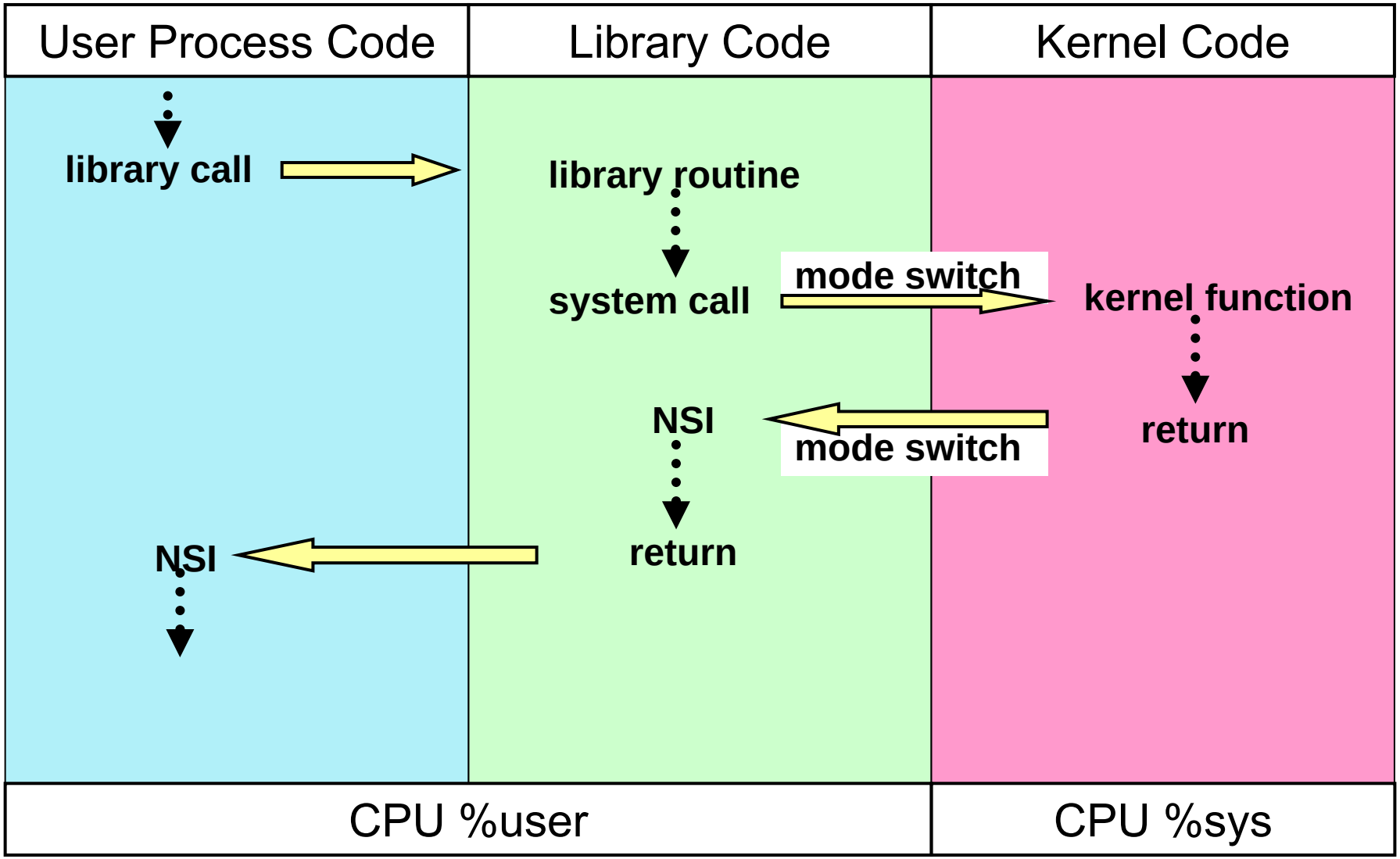
```
# ps -elk
      F S UID    PID   PPID   C PRI NI ADDR        SZ    WCHAN            TTY  TIME CMD
    303 A   0      0      0 120  16 -- 15004190   384                       -  0:01 swapper
 200003 A   0      1      0   0  60 20 10001480   708                       -  0:00 init
    303 A   0   8196      0   0 255 -- 17006190   384                       - 20:31 wait
    303 A   0  12294      0   0  17 -- 19008190   448                       -  0:00 sched
    303 A   0  16392      0   0  16 -- 1b00a190   512 f100080009786c08       -  0:00 lrud
    303 A   0  49176      0   0 255 -- 1d02c190   384                       - 20:11 wait
    303 A   0  53274      0   0 255 -- 1f02e190   384                       - 20:35 wait
    303 A   0  57372      0   0 255 -- 1030190    384                       - 20:14 wait
    303 A   0  61470      0   0  36 -- 2033190    448                       -  0:00 netm
    303 A   0  65568      0   0  37 -- 4035190    960           *           -  0:01 gil
    303 A   0  69666      0   0  16 -- 9038190    512  3f2af70              -  0:00 wlmsched
  40201 A   0  81986      0   0  60 20 170a6190   448                       -  0:00 lvmbb
 240001 A   0 106618      1   0  60 20 1c14d480   552           *           -  0:00 syncd
 240001 A   0 180346 151706   0  60 20 f1de480    376                       -  0:00 syslogd
 240001 A   0 192764 204958   0  60 20 1fb2e480   824 f100070000159c78  pts/2  0:00 ksh
 200001 A   0 262372 192764  34  87 24 1cb4d480    92                   pts/2 52:37 myprog
 200001 A   0 286896 192764  35  87 24 1fb4e480    92                   pts/2 52:32 myprog2
 200001 A   0 290950 356386   0  60 20 15b64480   732                   pts/1  0:00 ps
```

```
# ps -L 192764 -l
      F S UID    PID   PPID   C PRI NI ADDR        SZ    WCHAN            TTY  TIME CMD
 240001 A   0 192764 204958   0  60 20 1fb2e480   824 f100070000159c78  pts/2  0:00 ksh
 200001 A   0 262372 192764  40  90 24 1cb4d480    92                   pts/2 55:02 myprog
 200001 A   0 286896 192764  40  90 24 1fb4e480    92                   pts/2 54:55 myprog2
```

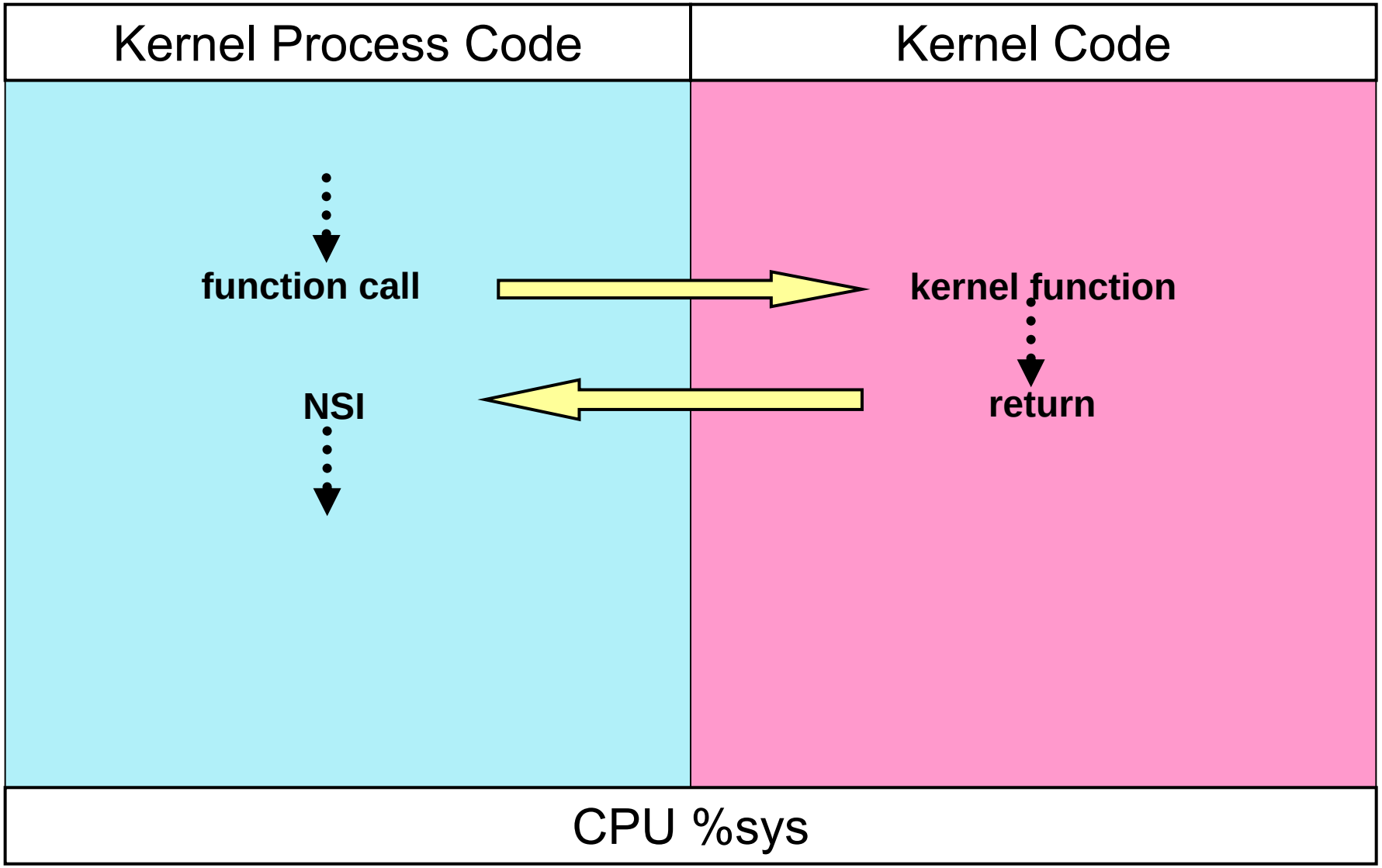
```
# ps -ekmo THREAD
    USER    PID   PPID      TID ST  CP PRI SC    WCHAN             F     TT BND COMMAND
    root      0      0        - A  120  16  1        -           303      -   - swapper
       -      -      -        3 S  120  16  1        -          1000      -   - -
    root      1      0        - A    0  60  1        -        200003      -   - /etc/init
       -      -      -     4099 S    0  60  1        -        410400      -   - -
    root   8196      0        - A    0 255  1        -           303      -   0 wait
       -      -      -     8197 R    0 255  1        -          3000      -   0 -
    root  16392      0        - A    0  16  2 f100080009786c08   303      -   - lrud
       -      -      -    16393 S    0  16  1 f100080009786c08  1004      -   - -
       -      -      -    45079 S    0  16  1        -          1004      -   - -
    root  49176      0        - A    0 255  1        -           303      -   2 wait
       -      -      -    77863 R    0 255  1        -          3000      -   2 -
    root  53274      0        - A    0 255  1        -           303      -   1 wait
       -      -      -    81961 R    0 255  1        -          3000      -   1 -
    root  57372      0        - A    0 255  1        -           303      -   3 wait
       -      -      -    86059 R    0 255  1        -          3000      -   3 -
    root 262372 192764        - A   58 100  0        -        200001 pts/2   - ./cpuprog
       -      -      -   733299 R   58 100  0        -             0      -   - -
    root 286896 192764        - A   63 103  1        -        200001 pts/2   - ./cpuprog
       -      -      -   737399 R   63 103  1        -             0      -   - -
    root 192764 204958        - A    0  60  1 f100070000159c78 240001 pts/2   - -ksh
       -      -      -   692347 S    0  60  1 f100070000159c78  10400      -   - -
```

```
# ps -mo THREAD -p 192764
    USER    PID   PPID      TID ST  CP PRI SC    WCHAN             F     TT BND COMMAND
    root 192764 204958        - A    0  60  1 f100070000159c78 240001 pts/2   - -ksh
       -      -      -   692347 S    0  60  1 f100070000159c78  10400      -   - -
```

User and Kernel Managed Thread Execution



Kernel Only Thread Execution



CPU Performance Statistics and Tuning

- Basic statistics (system wide or per CPU)

```
% user    % sys     % idle    % iowait
 8.4      2.6       88.5       0.5
```

- To identify the dominant processes and subroutines:
  - Use the **`ps, topas, and tprof`** command's
- Once the dominant processes and subroutines are identified:
  - Schedule work to run during low utilization period, prioritize work with nice value or use a workload manager
  - Fix, redesign, or tune the application
- Is the workload uneven among the processors?
  - With SMT, it is normal to see uneven CPU usage between the primary and secondary logical processors
- Application design issues
  - Inefficient user code, single threaded processes, excessive voluntary context switching, excessive or poorly managed system calls
- CPU capacity issues
  - Add more CPUs, CPU entitlement, or upgrade to faster CPUs

Detailing Source of CPU Consumption

HIGH % user

- Related to user code or library routines
  - High library routine usage is likely to be related to application design

HIGH % sys

- Often related to application design
  - Low processing in user code but high system call rate
  - Choice to use services with longer execution paths
  - High number of active threads for an application and a high rate of voluntary context switching
- Also, it may be related to
  - High adapter interrupt rates
  - High consumption by kernel processes

Subroutine Analysis with `tprof` (1 of 3)

- A system was exhibiting poor performance: **`vmstat`** showed the system as being consistently CPU bound with approximately 75% user and 25% sys

```
# pg tprof.sum
```

Process	FREQ	Total	Kernel	User	Shared	Other
db2sysc	318	225773	73906	3	151864	0
db2bp	14	26674	2901	0	23773	0
PID.6200	1	6021	6021	0	0	0

- The **`db2sysc`** process dominates the CPU with most of the CPU cycles being used in the shared library routines

Subroutine Analysis with `tprof` (2 of 3)

- The shared library section of the **`tprof`** report shows which shared libraries are being used the most

```
# pg tprof.sum
```

```
Total Ticks For All Processes (SH-LIBs) = 176740

Shared Object                              Ticks   %    Address   Bytes
=============                              ===== =====  ========  ======
/usr/lpp/db2_07_01/lib/libdb2e.a/shr.o    127103  46.7  d14900c0 1d22265
/usr/lib/libc.a/shr.o                      28789  10.6  d0160720  1c358f
```

- This report shows that a DB2 library is using the most CPU cycles

Subroutine Analysis with `tprof` (3 of 3)

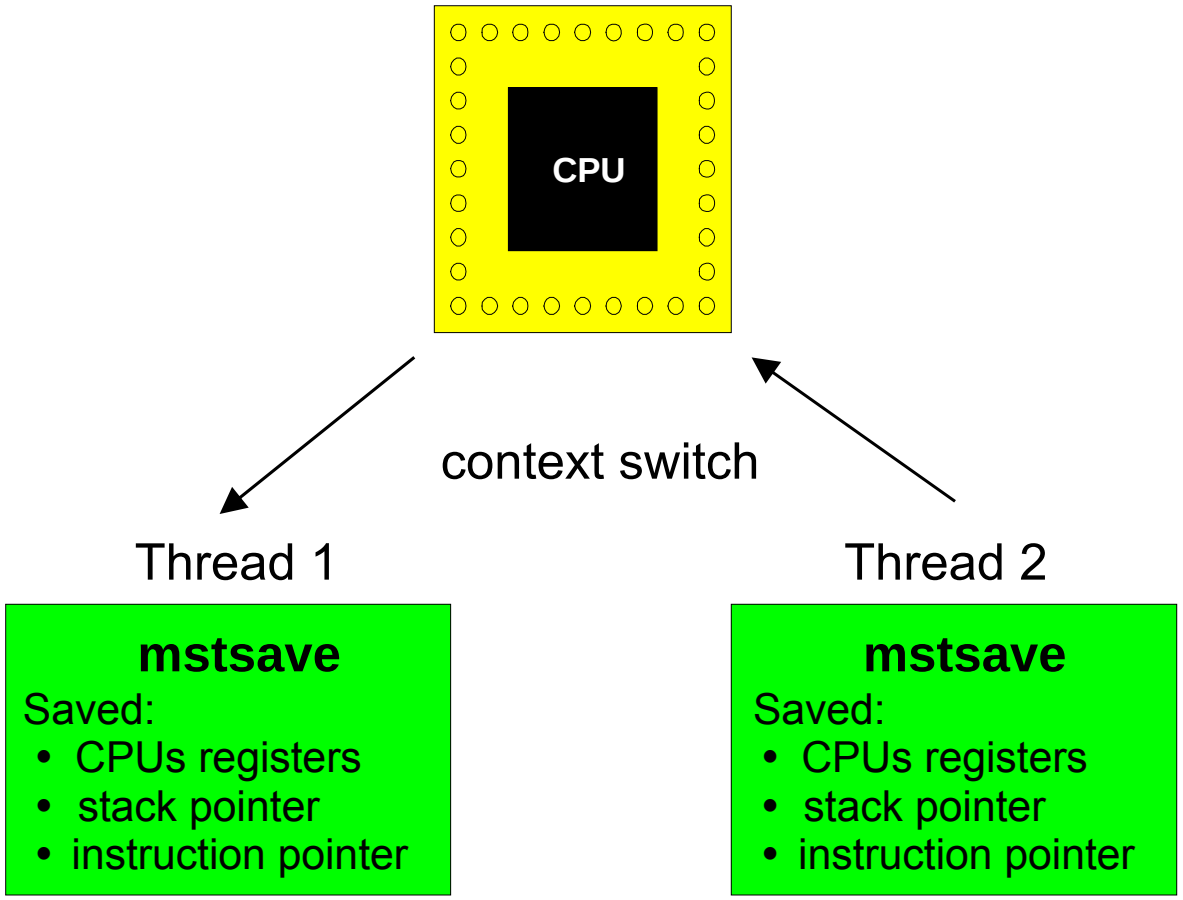
- The next step would be to examine the **`tprof`** report for the CPU utilization of the subroutines in that specific library
- Most of the CPU cycles are used in SQL processing, either a DB2 or an SQL user

```
# pg tprof.sum
```

```
Profile: /usr/lpp/db2_07_01/lib/libdb2e.a[shr.o]
Total Ticks For All Processes (/usr/lpp/db2_07_01/lib/libdb2e.a[shr.o]) = 127103
Subroutine                                    Ticks   %  Source      Address   Bytes
==========                                    ===== ==== ======      =======   =====
.sqldEvalDataPred(SQLD_DFM_WORK*,SQLD_DPRED*) 20705  7.6 sqldfrd.C  d1c81cec     6b0
.sqldReadNorm(SQLD_DFM_WORK*,int)             16819  6.2 sqldfrd.C  d16f4608     26c
.sqldReadNorm(SQLD_DFM_WORK*,int)             11054  4.1 sqldfrd.C  d16f4994     5d0
```

- This technique of drilling down in the **`tprof`** reports can help isolate a variety of problems, but cannot tell which process used which specific library subroutines

What are Context Switches?



Context Switches and Performance

- Context switches are a normal part of a multi-processing operating system
  - Saving and restoring contexts has an overhead cost
    - Unnecessary context switches will increase this cost
  - Undispatched thread may have cached data or instructions
    - Redispatch delay can result in loss of "cache warmth"

- Involuntary context switches
  - Preemptions due to priority can be long, but rate is usually limited, unless due to lock related priority boosts

- Voluntary context switches
  - Can be for long periods of time and can be at a very high rate

- Device interrupts can have a high rate, but:
  - Are usually brief
  - Interrupted routine does not wait on run queue to execute

Trace Disclaimer

- Trace hooks are basically debugging information in the kernel put there by the developers
- Trace hooks and related data associated with those hooks are not documented
- The details of a kernel trace can change at ANY time
- IBM does not document any specific trace hooks in detail
- Inclusion of trace data in this class does NOT constitute documentation
- We will only be focusing on a few trace hooks

Trace Example: Typical Preemption

```
ID  PROCESS NAME CPU TID      I ELAPSED APPL SYSCALL KERNEL  INTERRUPT

106 java         1   15126531   0.104927             dispatch:   cmd=java pid=1101874         tid=15126531
  priority=60 old_tid=16023735 old_priority=126 CPUID=1 [40 usec]

200 java         1   15126531   0.104930             resume  java iar=13BA08 cpuid=01

100 java         1   15126531   0.104932                     DECREMENTER INTERRUPT           iar=93BC cpuid=01

200 java         1   15126531   0.104940             resume  java iar=93BC cpuid=01



. . . < some output deleted >. . .



100 java         1   15126531   0.554878                     DECREMENTER INTERRUPT           iar=37E4 cpuid=01

4B0 java         1   15126531   0.554904             undispatch: old_tid=15126531            CPUID=1

106 java         1   8966175    0.554905             dispatch:   cmd=java pid=823506         tid=8966175
  priority=60 old_tid=15126531 old_priority=60 CPUID=1 [14 usec]

200 java         1   8966175    0.554912             resume  java iar=13BA08 cpuid=01
```

Context Switches Statistics: vmstat

```
# vmstat 10 . . .

System configuration: lcpu=10 mem=12288MB ent=0.50

kthr       memory             page                    faults                cpu           hypv-page          time
r  b p     avm   fre  fi  fo pi  po   fr   sr    in     sy    cs   us sy id wa   pc    ec hpi hpit  pmem loan hr mi se
35 1 0 2054490 16359 508 338  0 501    0    0  11372  69572 30843  86 14  0  0 4.50 900.9   0    0  12.00 0.00 00:58:22
33 1 0 2069741  9099 444 132  0 546  904  904  11857  65059 32961  85 15  0  0 4.54 908.2   0    0  12.00 0.00 00:58:32
32 1 0 2059662 21159 426 139  0 460  319  320  10660  55068 30024  85 15  0  0 4.26 852.1   0    0  12.00 0.00 00:58:42
32 0 0 2039644 40427 282 156  0 398    0    0   8468  64639 25552  84 16  0  0 3.94 788.3   0    0  12.00 0.00 00:58:52
35 1 0 2058986 20441 258  89  0 227    0    0   5601  40390 19349  86 14  0  0 3.17 634.3   0    0  12.00 0.00 00:59:02
40 1 0 2059440 19361 295  91  0 265    0    0   5956  44368 19959  85 15  0  0 3.33 666.6   0    0  12.00 0.00 00:59:12
39 0 0 2074561  5193 339 141  0  85  217  217   5904  40380 19081  86 14  0  0 2.99 598.3   0    0  12.00 0.00 00:59:22
32 0 0 2090388 10638 399 121  0 321 2239 2240   8724  55857 27763  86 14  0  0 4.19 838.2   0    0  12.00 0.00 00:59:32
31 1 0 2089548 10277 411 136  0 288    0    0  10595  63750 29099  86 14  0  0 4.37 874.8   0    0  12.00 0.00 00:59:42
23 1 0 2101187  6915 547 142  0 292  921  921  10400  50531 30067  85 15  0  0 4.23 845.5   0    0  12.00 0.00 00:59:52
29 1 0 2074794 35380 263 104  0 243  294  294   5739  42169 20056  84 16  0  0 3.01 602.0   0    0  12.00 0.00 01:00:02
32 1 0 2075872 33663 248  83  0 255    0    0   4679  44195 16170  82 18  0  0 2.49 498.7   0    0  12.00 0.00 01:00:12
33 1 0 2081620 27231 191 120  0 157    0    0   3445  21039 11745  83 16  0  0 1.62 324.6   0    0  12.00 0.00 01:00:22
29 1 0 2089875 20557 291  93  0 239  256  256   5340  37528 19089  86 14  0  0 2.96 592.3   0    0  12.00 0.00 01:00:32
. . . < some output deleted >. . .
```

```
# vmstat -s
          . . . < some output deleted >. . .
        71601491 cpu context switches
        16478909 device interrupts
         7603571 software interrupts
       120791875 decrementer interrupts
           20762 mpc-sent interrupts
           20762 mpc-received interrupts
        14983433 phantom interrupts
               0 traps
       342548645 syscalls
          . . . < some output deleted >. . .
```

Context Switches Statistics: mpstat

```
# mpstat 10 . . .

System configuration: lcpu=10 ent=0.5 mode=Uncapped

cpu   min maj mpcs mpcr   dev soft  dec   ph    cs   ics  . . .   sysc    us    sy   wa   id    pc    %ec   ilcs   vlcs
  0   580   7    0    0  1151   14  113  468  3066   883  . . .   6210  84.8  15.1  0.0  0.0  0.45  10.0    574     23
  1   677   9    0    0  1127   12  113  465  3168   930  . . .   6336  85.9  14.0  0.0  0.0  0.45  10.0    573     20
  2   690   9    0    0  1158   54  204  466  3285   921  . . .   7311  85.2  14.8  0.0  0.0  0.45  10.0    580     14
  3   596   9    0    0  1125    6  111  467  3310   972  . . .   6395  85.9  14.1  0.0  0.0  0.45  10.0    578     20
  4   772  12    0    0  1146   12  112  471  3390   944  . . .   8107  83.9  16.1  0.0  0.0  0.45  10.0    570     21
  5   597  10    0    0  1109    7  110  461  3305   932  . . .   6772  85.2  14.8  0.0  0.0  0.45  10.1    570     22
  6   750   8    0    0  1152    9  113  460  3017   873  . . .   6409  86.5  13.5  0.0  0.0  0.45  10.0    567     19
  7   726   8    0    0  1114    9  108  456  2698   784  . . .   6761  87.0  13.0  0.0  0.0  0.45  10.0    567     19
  8   816   9    0    0  1157    3  111  461  2738   822  . . .   8823  85.7  14.3  0.0  0.0  0.45  10.0    575     14
  9   998   7    0    0  1120    9  113  462  2951   889  . . .   6468  87.5  12.4  0.0  0.0  0.45  10.0    576     22
ALL  7202  88    0    0 11359  135 1208 4637 30928  8950  . . .  69592  85.7  14.2  0.0  0.0  4.51 901.2   5730    194
------------------------------------------------------------------------------------------------------------------------
  0   911  11    0    0  1194   18  118  492  3848  1139          7611  82.3  17.6  0.0  0.1  0.45  10.0    550     55
  1   551  12    0    0  1170    9  109  480  3725  1166     0    6336  84.0  15.8  0.0  0.1  0.45  10.0    551     45
  2  1301   7    0    0  1195   59  207  487  3843  1169     0    6694  81.9  18.0  0.0  0.1  0.45  10.0    548     59
  3   733  10    0    0  1168    7  107  484  3285   967     0    5686  84.7  15.2  0.0  0.1  0.45  10.0    547     50
  4  1463   8    1    0  1208    8  109  490  3237   915     0    6258  84.5  15.5  0.0  0.1  0.46  10.1    546     46
  5  1212  10    0    0  1171    8  112  488  3705  1118     0    6752  83.2  16.7  0.0  0.1  0.45   9.9    545     59
  6   746   9    0    0  1218    9  109  485  3176   966     0    5385  86.2  13.6  0.0  0.1  0.45  10.0    545     56
  7   601   9    0    0  1170    8  111  485  3053   924     0    5645  87.2  12.7  0.0  0.1  0.45  10.0    549     35
  8  1230   6    0    0  1195    8  114  494  2671   802     0    7225  87.5  12.5  0.0  0.1  0.46  10.2    548     41
  9  1310   7    0    0  1173    5  116  484  2779   861     0    7482  86.5  13.5  0.0  0.1  0.45   9.8    545     48
ALL 10058  89    1    0 11862  139 1212 4869 33322 10027     0   65074 84.8  15.1  0.0  0.1  4.54 908.3   5474    494
------------------------------------------------------------------------------------------------------------------------
. . . < some output deleted >. . .
```

Context Switches and Calls Statistics: sar

```
# sar -w
System configuration: lcpu=10 ent=0.50 mode=Uncapped

00:58:12 cpu cswch/s
00:58:22   0     3067
           1     3162
           2     3298
           3     3264
           4     3400
           5     3309
           6     3028
           7     2706
           8     2721
           9     2923
           -    15428

. . . < some output deleted >. . .
```

```
# sar -c
System configuration: lcpu=10 ent=0.50 mode=Uncapped

00:58:12 cpu scall/s sread/s swrit/s  fork/s  exec/s rchar/s wchar/s
00:58:22   0     6210     623      75    0.70    0.90  489983   32856
           1     6316     446      90    1.90    1.40  642590   20586
           2     7301     651      88    1.10    1.30 1089182   14556
           3     6405     384      78    1.60    0.90  684584   15839
           4     8108     470      85    0.50    0.70 1526234  439241
           5     6791     297     135    0.30    0.50  750652   51197
           6     6409     302      88    1.30    1.20  457957   81432
           7     6715     480      95    1.50    1.50  545720  115235
           8     8831    1150      81    1.60    1.40 2130405   87166
           9     6466     700      79    0.50    1.40  922841   19607
           -    34800    2748     448    5.60    5.65 4619638  438944

. . . < some output deleted >. . .
```

Using `curt`

- CPU Utilization Reporting Tool (**`curt`**)
  - Based on an existing raw trace file
  - Provides a more detailed analysis of subroutines calls
- Summary reports:
  - **System Summary** categorizes cycles for entire system:
    - Application, syscall, kproc, FLIH, SLIH, dispatch
  - **Application and Kproc** ranks threads of each type
  - **System Calls** provides statistics on service calls
  - **Pending System Calls** shows the system calls that did not complete by the end of the trace
  - **FLIH and SLIH** statistics categorizes the types of interrupts
- Optional report information:
  - Elapsed times for system calls
  - Errors returned by system calls
  - Detailed reports for each thread or for each process

curt: System Summary

```
                  System Summary
                  --------------


 processing       percent       percent
 total time     total time     busy time
     (msec)   (incl. idle)   (excl. idle)  processing category
===========    ===========    ===========  ==================
    9044.33         67.71         70.37  APPLICATION
    1233.02          9.23          9.59  SYSCALL
     304.24          2.28          2.37  HCALL
    1575.00         11.79         12.25  KPROC (excluding IDLE and NFS)
       0.00          0.00          0.00  NFS
     173.94          1.30          1.35  FLIH
     413.19          3.09          3.21  SLIH
     108.34          0.81          0.84  DISPATCH (all procs. incl. IDLE)
       0.01          0.00          0.00  IDLE DISPATCH (only IDLE proc.)
-----------    ----------       -------
   12852.05         96.22        100.00  CPU(s) busy time
     504.54          3.78                IDLE
-----------    ----------
   13356.59                             TOTAL

Avg. Thread Affinity = 0.91

Total Physical CPU time (msec) = 13430.57
Physical CPU percentage        = 65.87
```

curt: Application and Kproc Summaries

```
                Application Summary (by Tid)
                ----------------------------
  -- processing total (msec) --          -- percent of total processing time --
 combined  application      syscall     combined  application      syscall  name (Pid  Tid)
 ========  ===========      =======     ========  ===========      =======  ==================
 511.1163     510.8494       0.2669       3.8267       3.8247       0.0020  java(1425506  9736363)
 376.9658     376.8221       0.1437       2.8223       2.8212       0.0011  java(1445898  6656211)
 366.3085     366.1054       0.2031       2.7425       2.7410       0.0015  java(643138  6869083)
 . . . < some output deleted >. . .
```

```
                  Application Summary (by process type)
                  ---------------------------------
   -- processing total (msec) --          -- percent of total processing time --
  combined  application      syscall     combined  application      syscall  name (thread count)
  ========  ===========      =======     ========  ===========      =======  ==============
10266.6747    9036.6816    1229.9930      76.8660      67.6571       9.2089  java(630)
    1.7188       1.1906       0.5282       0.0129       0.0089       0.0040  /usr/bin/sleep(1)
    1.5208       1.5208       0.0000       0.0114       0.0114       0.0000  getty(1)
    0.6630       0.0452       0.6178       0.0050       0.0003       0.0046  gmond(1)
    0.5024       0.1132       0.3892       0.0038       0.0008       0.0029  ksh(1)
. . . < some output deleted >. . .
```

```
                  Kproc Summary  (by Tid)
                 ----------------------
  -- processing total (msec) --          --   percent of total  time   --   name (Pid Tid Type)
 combined       kernel    operation     combined       kernel    operation
 ========       ======  ===========     ========       ======  ===========  ==================
 861.5990     861.5990       0.0000       6.4507       6.4507       0.0000  wait(81960  122941 W)
 584.8921     584.8921       0.0000       4.3791       4.3791       0.0000  wait(77862  118843 W)
  70.9009      70.9009       0.0000       0.5308       0.5308       0.0000  rtcmd(209020  2240613 -)
  10.7983      10.7983       0.0000       0.0808       0.0808       0.0000  swapper(0  3 -)
  10.2881      10.2881       0.0000       0.0770       0.0770       0.0000  kbiod(139434  16781379 -)
 . . . < some output deleted >. . .
```

curt: System Calls Summary and Errors

–e option shows additional columns

```
                System Calls Summary
                --------------------
Count TotalTime % sys AvgTime MinTime MaxTime  TotETime AvgETime MinETime MaxETime SVC (Address)
         (msec)  time  (msec)  (msec)  (msec)    (msec)   (msec)   (msec)   (msec)
===== ========= ===== ======= ======= =======  ======== ======== ======== ======== ================
11084  277.8998 2.08%  0.0251  0.0000  0.9215  4014.4147   0.3622   0.0000 477.0447 _esend(2999e38)
22023  242.3083 1.81%  0.0110  0.0034  2.0658 45537.7396   2.0677   0.0034 296.9161 _erecv(2999d48)
  111  235.9864 1.77%  2.1260  1.4061  3.9727  2050.5125  18.4731   4.6201  94.3297 rename(299ee18)
 1244  178.9950 1.34%  0.1439  0.0021  4.1999  4323.2819   3.4753   0.0021 301.4679 getdiret(299e9b0)
  976   61.8068 0.46%  0.0633  0.0167  1.1531  3066.4633   3.1419   0.0167 296.8303 statx(299ee90)
  403   59.8998 0.45%  0.1486  0.0221  1.5419  2193.1022   5.4419   0.0221 136.6113 kopen(299ed58)
. . . < some output deleted >. . .
```

```
                  Errors Returned by System Calls
                  -------------------------------
  Errors (errno : count : description) returned for System Call: _erecv(0x2999d48)
      4 :        1 : "Interrupted system call"
   Errors (errno : count : description) returned for System Call: statx(0x299ee90)
      2 :      771 : "No such file or directory"
  Errors (errno : count : description) returned for System Call: kopen(0x299ed58)
      2 :        1 : "No such file or directory"
  Errors (errno : count : description) returned for System Call: thread_tsleep(0x29987f0)
      4 :        2 : "Interrupted system call"
  Errors (errno : count : description) returned for System Call: __loadx(0x29a59e8)
      2 :        1 : "No such file or directory"
    109 :      208 : "Function not implemented"
  Errors (errno : count : description) returned for System Call: _nsleep(0x2998be0)
      4 :        1 : "Interrupted system call"
  Errors (errno : count : description) returned for System Call: kioctl(0x299ea58)
     25 :      422 : "Not a typewriter"
  Errors (errno : count : description) returned for System Call: ngetsockname(0x2999c70)
     57 :        1 : "Socket operation on non-socket“
. . . < some output deleted >. . .
```

curt: Interrupt Handler Summaries

```
                    Global Flih Summary
                    -------------------
 Count  Total Time    Avg Time    Min Time    Max Time  Flih Type
           (msec)      (msec)      (msec)      (msec)
====== =========== =========== =========== ===========  =========
    21      0.2899      0.0138      0.0044      0.0330    4(INSTR_PG_FLT)
   511      3.8289      0.0075      0.0004      0.1075   32(QUEUED_INTR)
  2416     44.5332      0.0184      0.0004      0.0994   31(DECR_INTR)
    93      2.9237      0.0314      0.0032      0.3101    9(PHANTOM)
 14873     14.8839      0.0010      0.0003      0.0115    5(IO_INTR)
  9303     27.7094      0.0030      0.0012      0.2519   64(UNRECOGNIZED INTR TYPE)
  5482     79.7689      0.0146      0.0015      0.3260    3(DATA_ACC_PG_FLT)
```

```
                    Global Slih Summary
                    -------------------
 Count  Total Time    Avg Time    Min Time    Max Time Slih Name(Address)
           (msec)      (msec)      (msec)      (msec)
====== =========== =========== =========== =========== =================
     1      0.1934      0.1934      0.1934      0.1934 (unknown)(a012f0d0)
   132     18.0682      0.1369      0.0235      2.7573 hea_eq_intr(45e3508)
```

curt: Detailed Report for Threads (1 of 2)

```
--------------------------------------------------------------------------------
Report for Thread Id: 9515057 (hex 913031) Pid: 880716 (hex d704c)
Process Name: java
--------------------
 Total Application Time (ms):  136.710346
 Total System Call Time (ms):  29.749092
 Total Hypervisor Call Time (ms):  5.547684

                         Thread System Call Summary
                         --------------------------
 Count  Total Time Avg Time Min Time Max Time Tot ETime Avg ETime Min ETime Max ETime    SVC (Address)
           (msec)   (msec)   (msec)   (msec)    (msec)    (msec)    (msec)    (msec)
====== =========== ======== ======== ======== ========= ========= ========= ========= ================
   446     10.7584   0.0241   0.0159   0.1124  193.2016    0.4332    0.0159   33.3568 _esend(2999e38)
   892      9.3878   0.0105   0.0046   0.0449 1390.4841    1.5588    0.0046   73.2734 _erecv(2999d48)
    27      4.9114   0.1819   0.0028   2.4620   37.3978    1.3851    0.0028    7.0277 getdir(299e9b0)
     1      2.5966   2.5966   2.5966   2.5966   19.6231   19.6231   19.6231   19.6231 rename(299ee18)
     3      1.0431   0.3477   0.2249   0.5877   23.1601    7.7200    1.2670   17.3201 kopen(299ed58)
     3      0.5979   0.1993   0.1696   0.2568    3.8783    1.2928    0.9070    1.5025 statx(299ee90)
. . . < some output deleted >. . .

  Errors (errno : count : description) returned for System Call: statx(0x299ee90)
      2 :        1 : "No such file or directory“
```

curt: Detailed Report for Threads (2 of 2)

```
. . . < some output deleted >. . .

                     processor affinity:  0.914839

Dispatch Histogram for thread (CPUid : times_dispatched).
    CPU 0 : 32
    CPU 1 : 64
    CPU 4 : 254
    CPU 5 : 383
    CPU 7 : 7
    CPU 8 : 11
    CPU 9 : 24

 total number of dispatches:  774
 total number of redispatches due to interupts being disabled:  1
 avg. dispatch wait time (ms):  2.255395

      Data on Interrupts that Occured while Thread was Running
           Type of Interrupt       Count
   ============================== ==========================
    Data Access Page Faults (DSI): 4
   Instr. Fetch Page Faults (ISI): 0
          Align. Error Interrupts: 0
         IO (external) Interrupts: 263
         Program Check Interrupts: 0
        FP Unavailable Interrupts: 0
          FP Imprecise Interrupts: 0
               RunMode Interrupts: 0
           Decrementer Interrupts: 36
   Queued (Soft level) Interrupts: 11
      Perf. Monitoring Interrupts: 0
```

Reducing Kernel Mode CPU Consumption

- Reduce the number of system calls and avoid unneeded functions
- Disk I/O services:
  - Use large blocks for reads and writes
  - Use raw I/O, direct I/O, or concurrent I/O
- Network I/O:
  - Use larger sends and receives
  - Eliminate small MTUs, reduce segmentation and fragmentation
  - Tune TCP `sendspace` and `recvspace` values
  - Offload work to network adapters

Tuning the Priorities of Processes

- If CPU resources are already constrained, the setting of priorities can help use more CPU resources for the more important processes:
  - Decrease priority value on the most important processes
  - Increase priority value for the least important processes
- If most important processes use a lot of CPU time, could change CPU usage priority decay rate
  - Configure the CPU aging/decay and the CPU usage penalty options with **schedo**
- Consider using a workload manager to balance the CPU resources

CPU Usage Penalty (sched_R)

- Thread priority is calculated by the formula:

  Priority = x_nice + (Current CPU ticks * R / 32 * (x_nice + 4 / 64))

  Where:
  - p_nice = nice value + base priority
  - If p_nice > 60
    then x_nice = 20 + base priority + (2 * (nice - 20))
    else x_nice = p_nice
  - CPU penalty = CPU usage * R/32 *(Default R value is 16)*
    (R/32 is the CPU-penalty-to-recent-CPU-usage ratio)

- CPU usage is incremented by one for the thread in control of the CPU when a time interrupt occurs
- Thread priority is also recalculated whenever a thread is dispatched

CPU Aging/Decay (sched_D)

- CPU usage is *decayed* once per second by the kernel **swapper** process using the formula:

  CPU usage * D/32 *(Default D value is 16)*
  (D/32 is the recent-CPU-usage-decay factor)

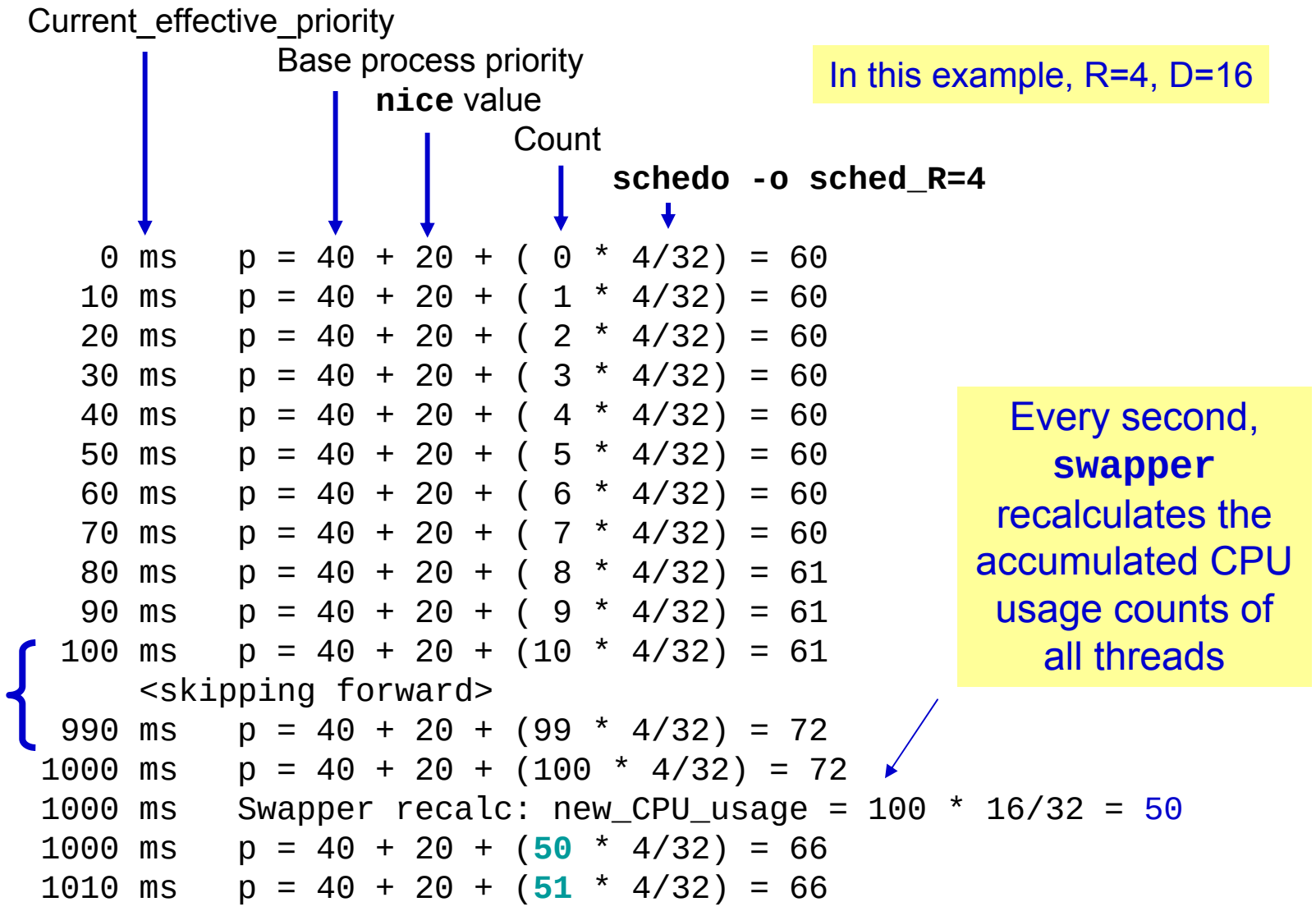
- For example, if a thread's priority is currently 72 after one second, where D=4 and R=16:

```
1000 ms    p = 40 + 20 + (100 * 4/32) = 72
1000 ms    Swapper recalc: new_CPU_usage = 100 * 16/32 = 50
1000 ms    p = 40 + 20 + (50 * 4/32) = 66
```

"R"

"D"

Priority Calculation Example



Session Summary

- Use the output of the following AIX tools to determine symptoms of a CPU bottleneck:
  - **`vmstat`**, **`sar`**, **`ps`**, **`topas`**, **`tprof`**, and **`curt/trace`**
- Interpret the trace output and identify routines and threads being executed
- Identify causes and impacts of context switches
- Tuning the Priorities of Processes

Ya'll have fun now, hear!

Gracias!!!