HP Adaptive Enterprise

Marty Poniatowski

Session ID #: 1642

Title: HP Adaptive Infrastructure

Room: B217

Date: Tuesday, 8/12/2003

Time: 4:50 PM

HP WORLD 2003
Solutions and Technology Conference & Expo

Technology Update

1) Enterprise Systems (PA-RISC and Itanium)

2) Itanium and Multiple Operating Systems: HP-UX, Linux, Windows

3) Partition Technology in the Adaptive Enterprise

nPartitions: detail

vPars: detail

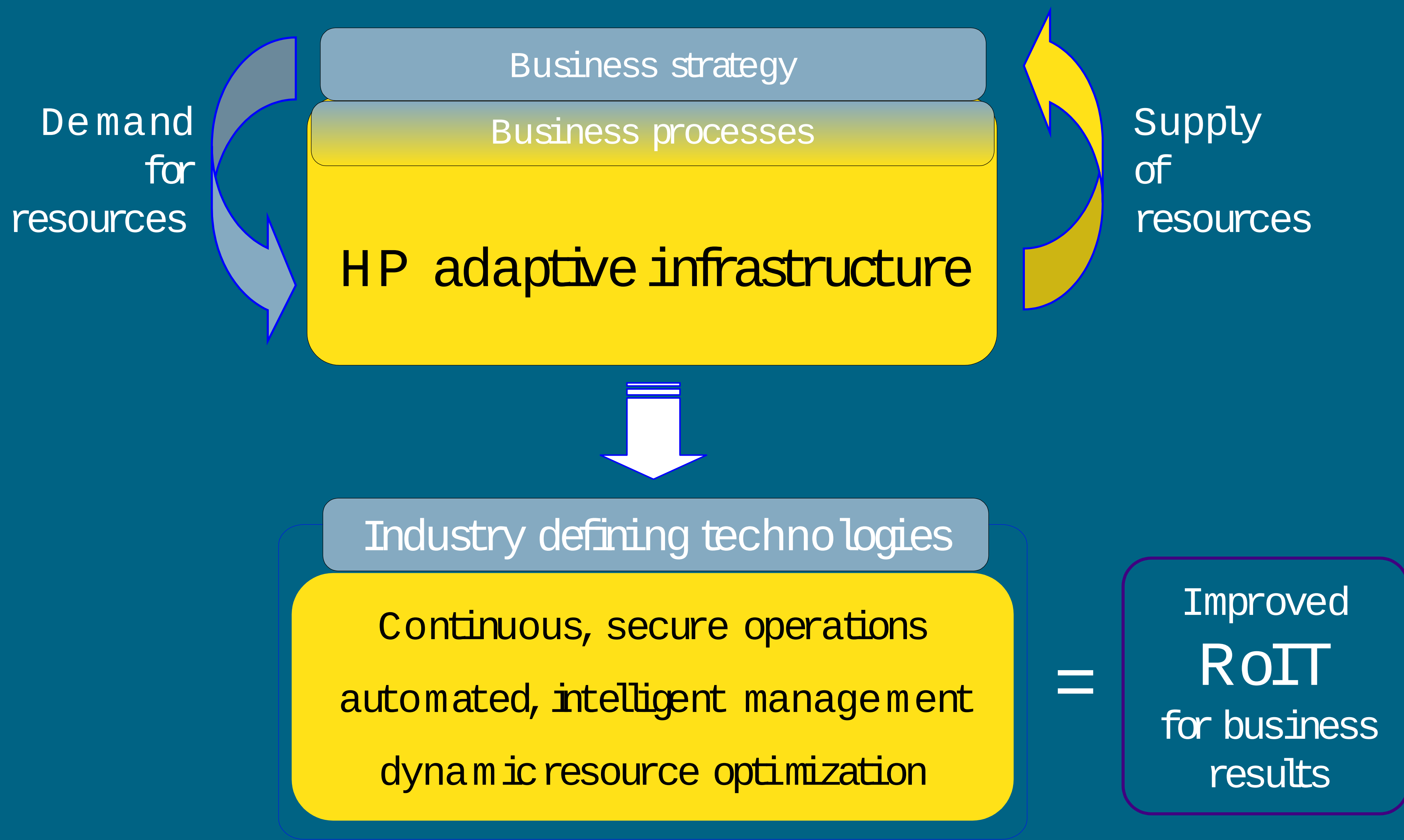
PRM: detail

WLM: description

iCOD, TiCOD,PPU, and psets description

Business agility requires an HP adaptive infrastructure

Immediate knowledge, intelligent action



servers

HP-UX product family

high-end

HP 9000 (PA-RISC)
Integrity (Itanium)

Superdome (Itanium)

mid-range

rp7410(rx7610-Itanium)

rp8400(rx8600-Itanium)

entry-level

rp2400

rp5400

Itanium

Rx2600
(new 2-way coming)

Coming
rx4640 – 3u

Itanium

rx5670
(rx4640 coming)

hp server naming decoder ring
numeric digits

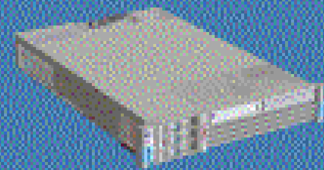
hp server aadddd

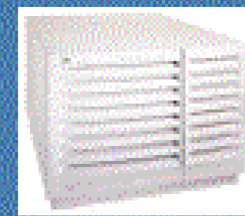
00 - 90 relative capacity & “newness” (upgrades, etc.)

Unique number for each architecture to ensure different systems do not have the same numbering across architectures

1-9 identifies family and/or relative positioning

a closer look at the rx2600 and rx5670





	rx2600	rx5670
processors	1-2 way 900MHz and 1 GHz IPF CPU	1-4 way 900 MHz and 1 GHZ IPF CPU
memory	up to 12GB DDR SDRAM	up to 48GB DDR SDRAM
bandwidth	6.4 GB/s system; 5.5 GB/s memory; 4.0 GB/s I/O	6.4 GB/s system; 12.8 GB/s memory; 4.0 GB/s I/O
pci-x/pci slots	4 PCI-X @ 133MHz	9 PCI-X (3 @ 133MHz, 6 @ 66MHz); 1 PCI (33MHz)
internal storage	up to 219GB	up to 292GB
operating system	HP-UX 11i ver 1.6, Linux, Windows Advanced Server LE	HP-UX 11i ver 1.6, Linux, Windows Advanced Sever LE
positioning	2-way IPF price/performance leader	4-way IPF server solutions leader

HP Itanium® 2-based systems for superior application performance

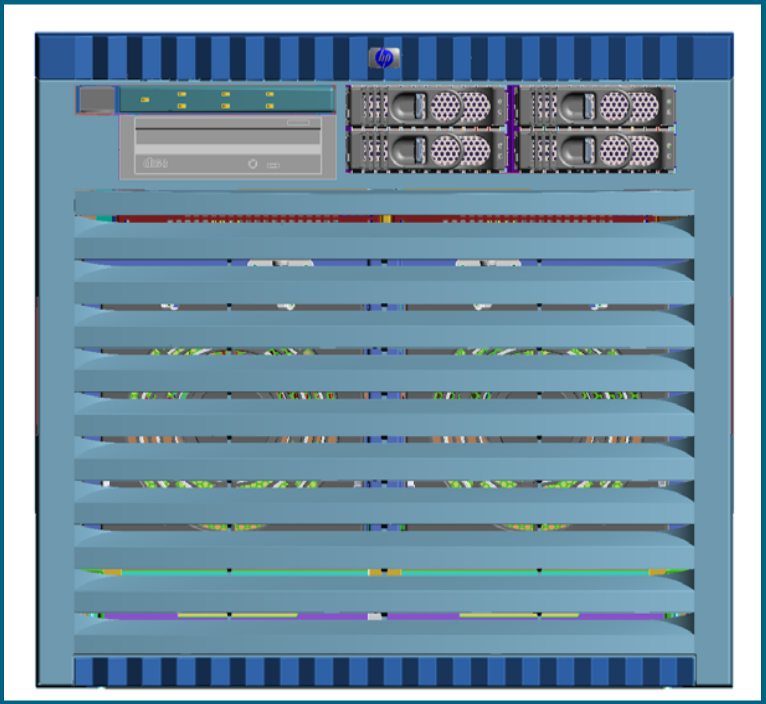
	typical IA-32 system	typical RISC system	Itanium® 2-based hp system	benefits:
CPU bus bandwidth	1-3 GB/s	2-4 GB/s	6.4 GB/s	→ faster OLTP
I/O bandwidth	1 GB/s	2 GB/s	4 GB/s	→ quicker web serving
on-chip resources	8 general registers	32 general registers	128 general registers	→ faster secure transactions
parallel execution	1 instruction per cycle	2-4 instructions per cycle	6 instructions per cycle	→ better decision support performance

delivering on our promise
"investment protection only hp can deliver"

the world's only in-box upgrade from an existing RISC server to an Itanium 2-based server!

rp7410 8-way

(8 processor
1 and 2-way,
and 8-way
already covered)



Hewlett-Packard introduces
the hp server rp7410

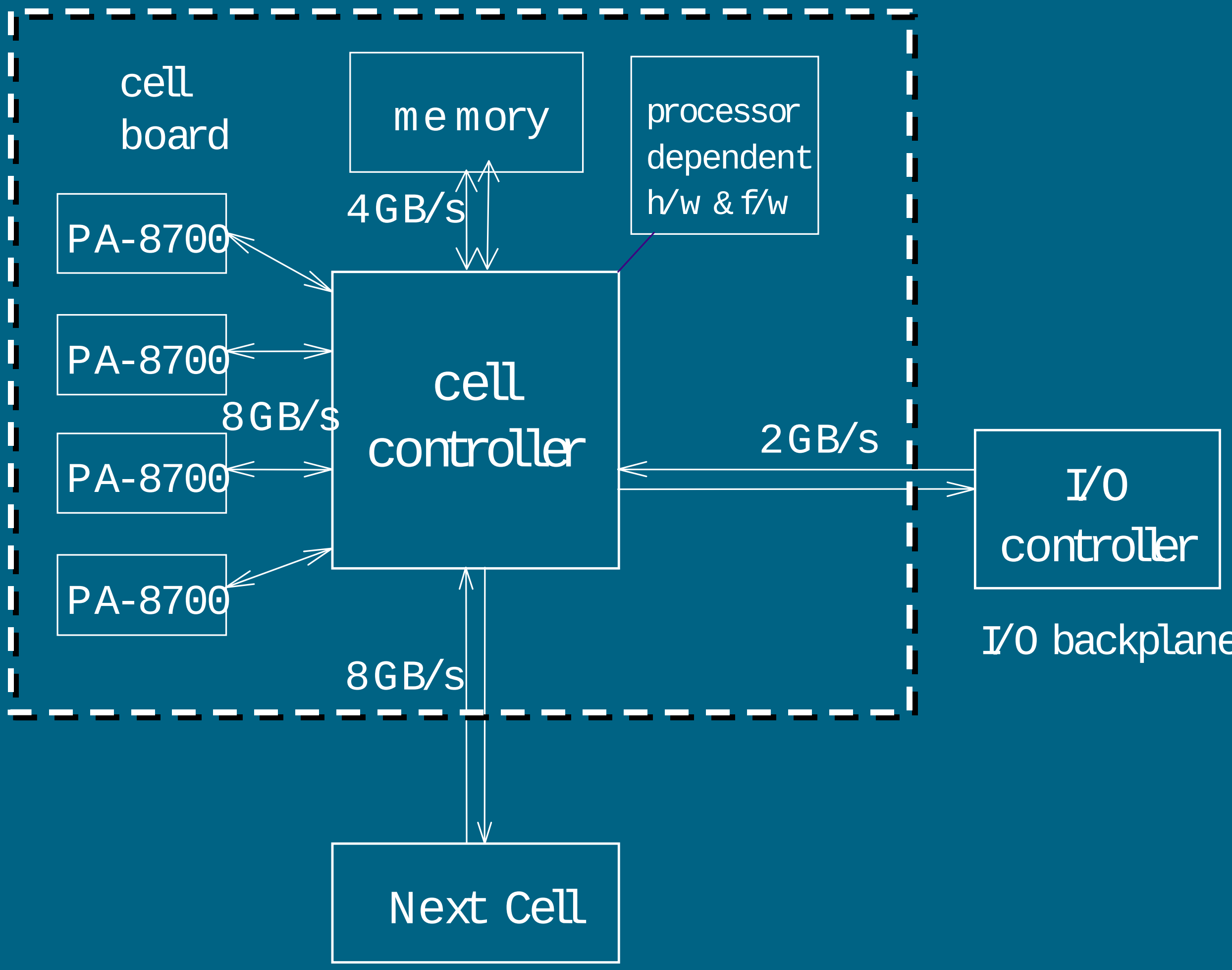
rp7410 system architecture building blocks: cell board

rp7410 is a cell-based system

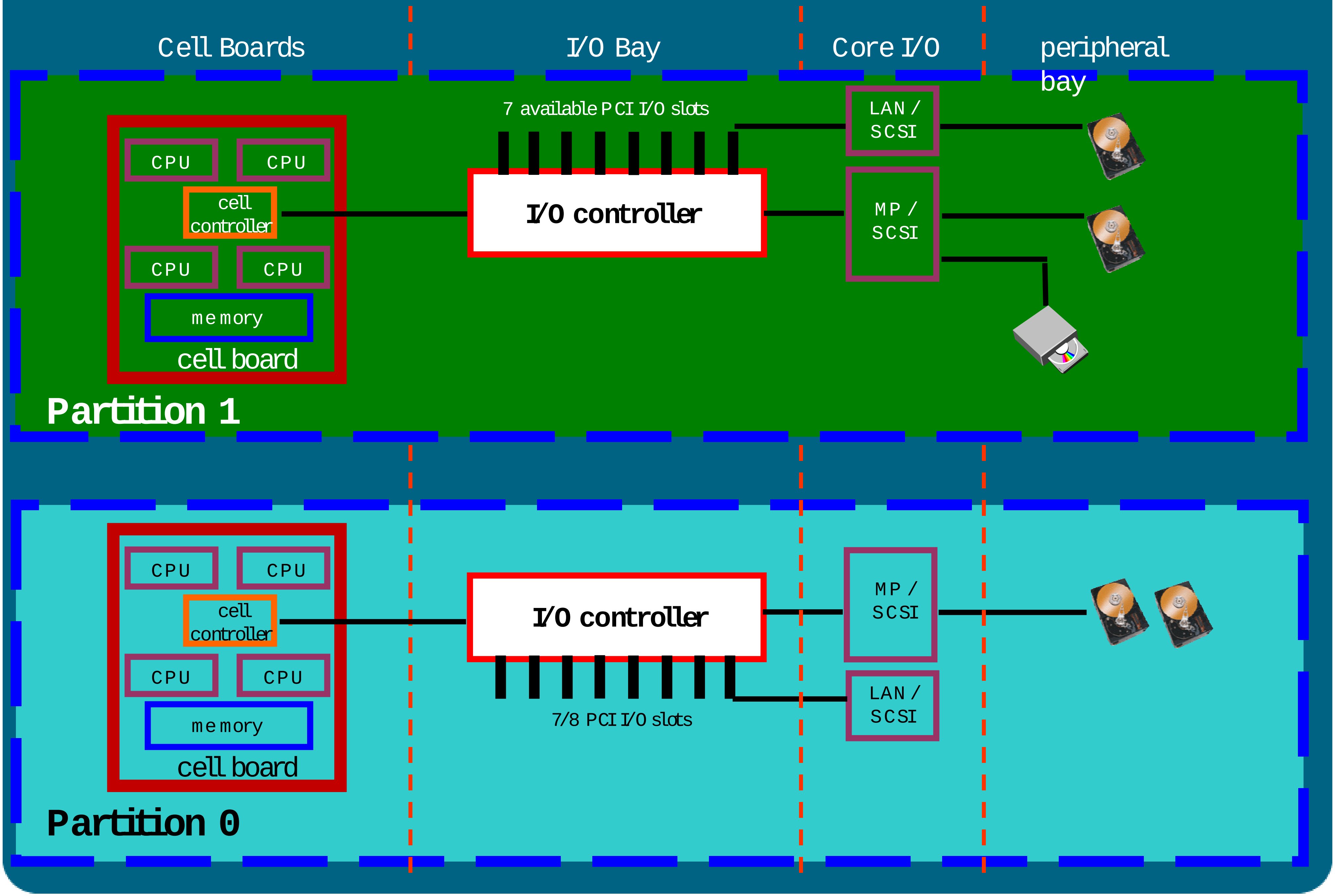
Interchangeable with the rp8400 cell

a cell consists of:

- 4 CPUs
- 2 to 16 GB of memory with 128-MBit DRAMs)
- link to PCI I/O slots and adjacent cell



rp7410 system architecture - Partitioned

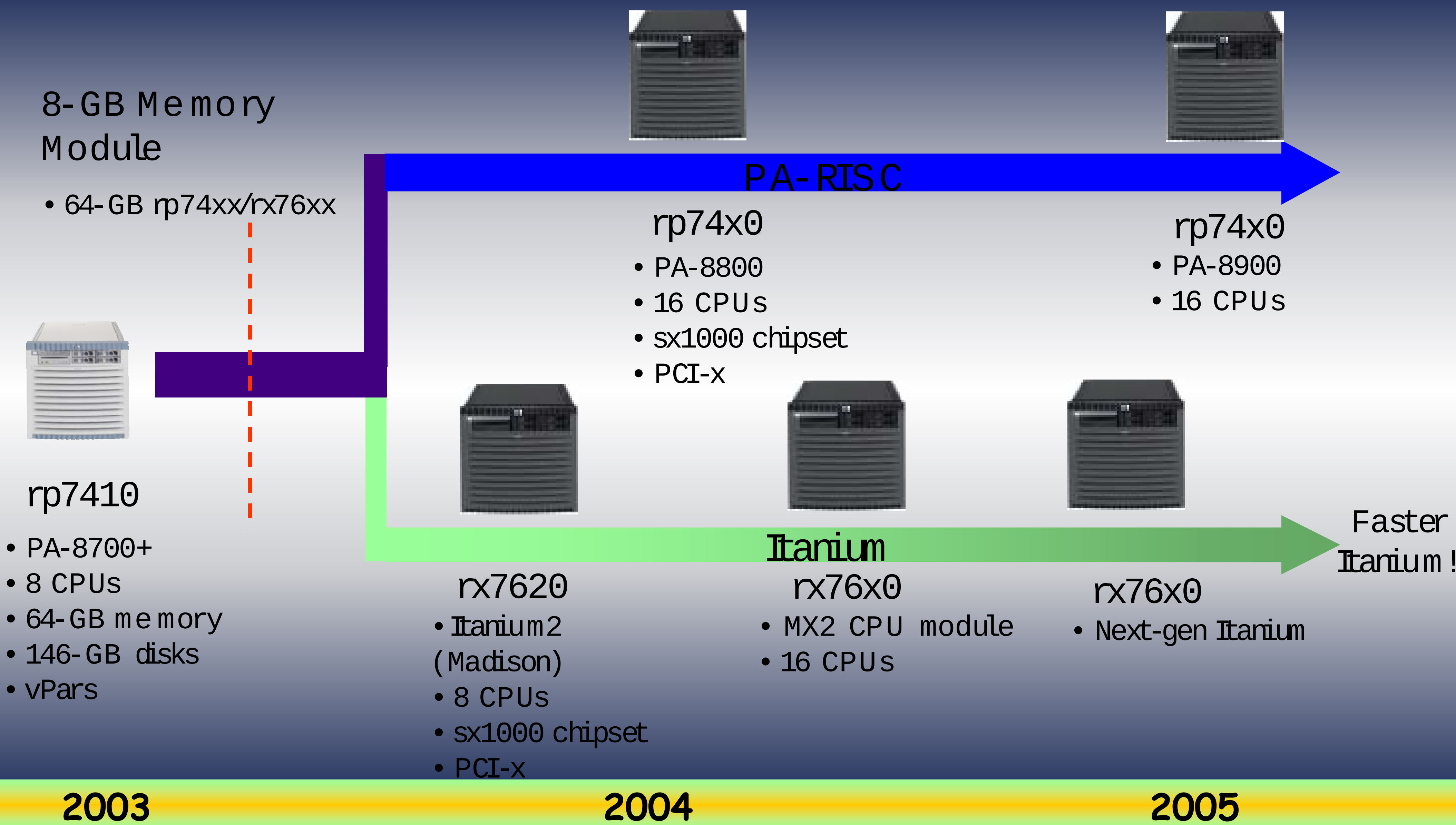


rp7410
evolution in hp's 8-way leadership

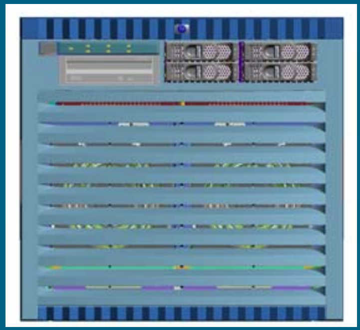
better performance and functionality with a flexible growth path!

	rp7400	rp7410
partitions	virtual	virtual and hardware
I/O	12 PCI slots	15 PCI slots
core I/O	1	2 with fail-over capabilities
memory	32 GB	64 GB*
aggregated bandwidth	20 GB/s	32 GB/s
serviceability/ accessibility	requires all-sides access	front access no tools
depth	35 inch.	29 inch.
architecture	bus architecture	high-end cell-board
power solution	N+1 solution	2N+1 dual grid solution
internal peripherals	2 HDDs	4 HDDs and 1 removable

rp7410 roadmap



rp8400 16-way—the technology (8 processor 1 and 2-way, 4 way,and 8-way already covered)



winning physical specifications

- high density (17U form factor)
- rack-optimized and stand-alone
- optimum power requirements
- fits into 3rd-party racks
- optimum upgrade/service time
- front and back server access
- sophisticated cable mgmt.

unmatched system features

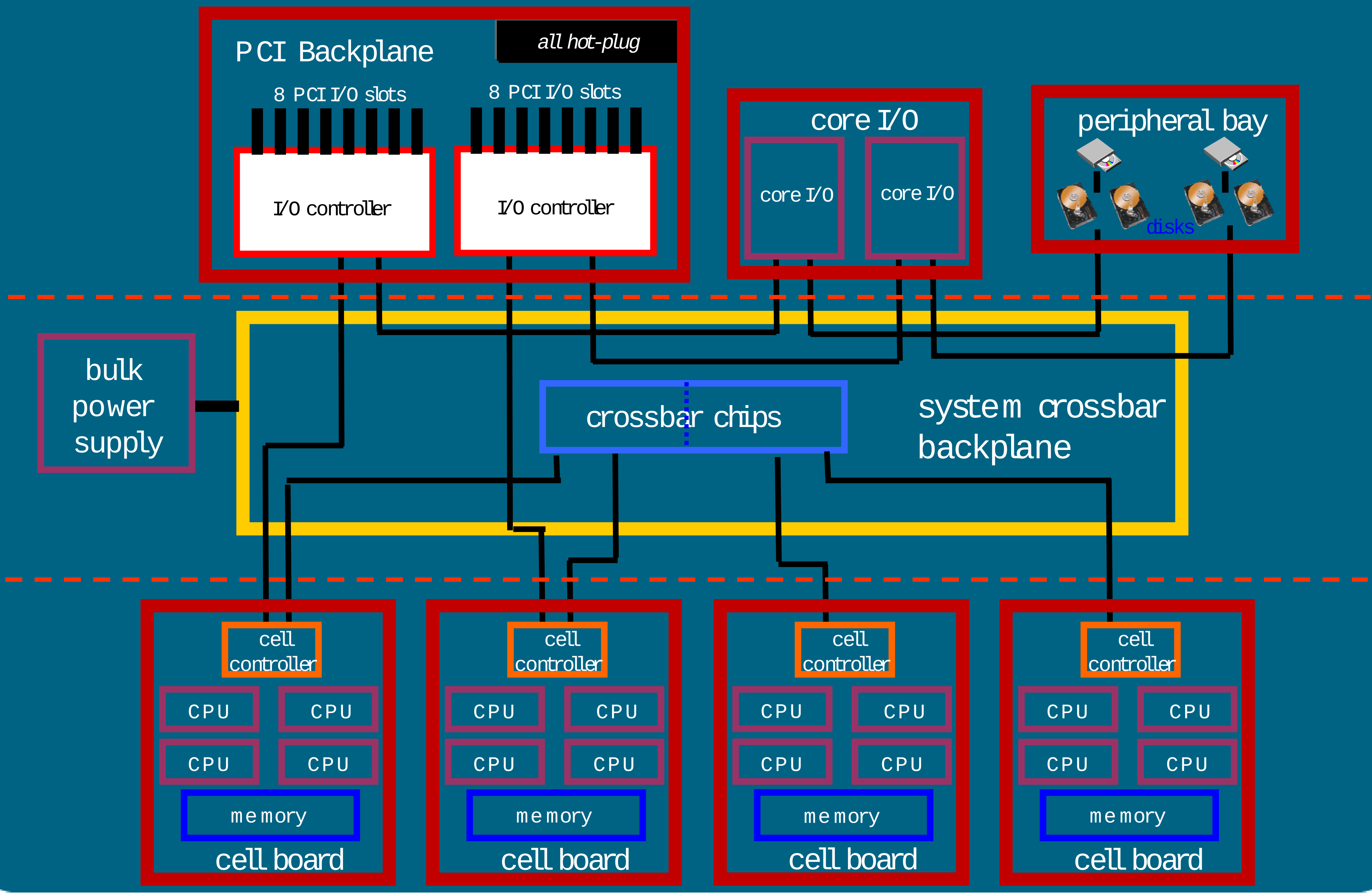
- 2- to 16-way industry-leading PA-8700 CPUs at 650,750 and 875 MHz
- superdome high-end cell board architecture
- hardware and virtual partitions
- 64-GB main memory
- 16 PCI slots and 2 core I/Os
- 4 internal hot-plug HDDs
- 2 internal hot-plug removable media peripherals
- 2N+1 power supply solution

high availability

- hot-plug cell boards
- OLAR PCI cards
- doorbell PCI card functionality*
- N+1 OLR fans
- 2N+1 OLR power supplies
- failover system console functionality*
- ECC on all CPU, memory and bus paths
- CPU and memory deallocation
- memory chip-kill-like technology
- EMS monitor/diagnostic
- 2N input power – dual grid support

*estimated availability 1H02

rp8400 system architecture



rp8400 roadmap

8-GB Memory Module

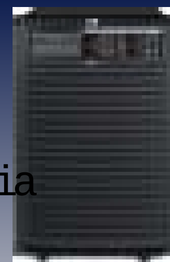
- 128-GB rp84xx/rx86xx

rp8400

- PA-8700+
- 16 CPUs
- 128-GB memory
- 146-GB disks
- vPars

Ser.Exp.Unit

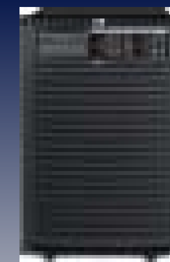
- 16 added PCI-X slots
- 4 added disk drives
- 2 added removable media
- 2 added nPars/core I/O



PA-RISC

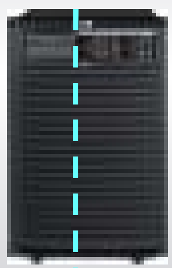
rp84x0

- PA-8800
- 32 CPUs
- sx1000 chipset
- PCI-x
- SEU



rp84x0

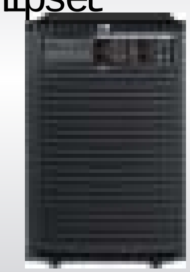
- PA-8900
- 32 CPUs



rx8620

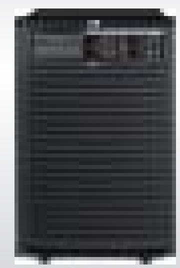
- Itanium2 (Madison)
- 16 CPUs
- sx1000 chipset
- PCI-x
- SEU

Itanium



rx86x0

- MX2 CPU module
- 32 CPUs



rx86x0

- Next-gen Itanium

Faster Itanium

2003 2004 2005

superdome: built for the future with investment protection today

- pa-8700
  - 98% performance increase to 389k TPC-C

- PA-8700+
  - 875 MHz
  - 64 CPUs
  - 256GB RAM
  - New I/O Cards

- Itanium
  - Madison
  - (1 CPU/chip)
  - 512GB RAM
  - PCI-X
  - linux
  - HP-UX

- Itanium mx2
  - Madison
  - (2 CPUs/socket)
  - 128 CPUS
  - 1 TB RAM
- pa-8800
  - (2 CPUs/chip)
  - 128 CPUs
  - 1 TB RAM

- Itanium
  - Montecito
  - (2 CPUs/chip)
  - 128 CPUs
  - 1 TB RAM
  - linux
  - windows
  - HP-UX

2h 2001 | mid 2002 | July 2003 | 1Q 2004 | 2h 2004

subject to change without notice

hp superdome

Performance & scalability
- single cabinet:
  - 32, 64 CPUs
  - 64, 128, 256 GBs
- 48, 96, 192 PCI slots
- HP-UX 11i OS
- management, security and e-services software

Partitioning continuum
- hp hyperplex
- nPartitions (up to 16)
- virtual partitions
- resource management

Utility technology & pricing
- iCOD
- utility pricing

High availability
- N+1 OLR fans
- N+1 OLR power supplies
- dual power source
- OLAR CPU, memory
- OLAR PCI I/O cards
- parity protected I/O data paths
- ECC on all CPU and memory paths
- dynamic processor resilience
- dynamic memory resilience

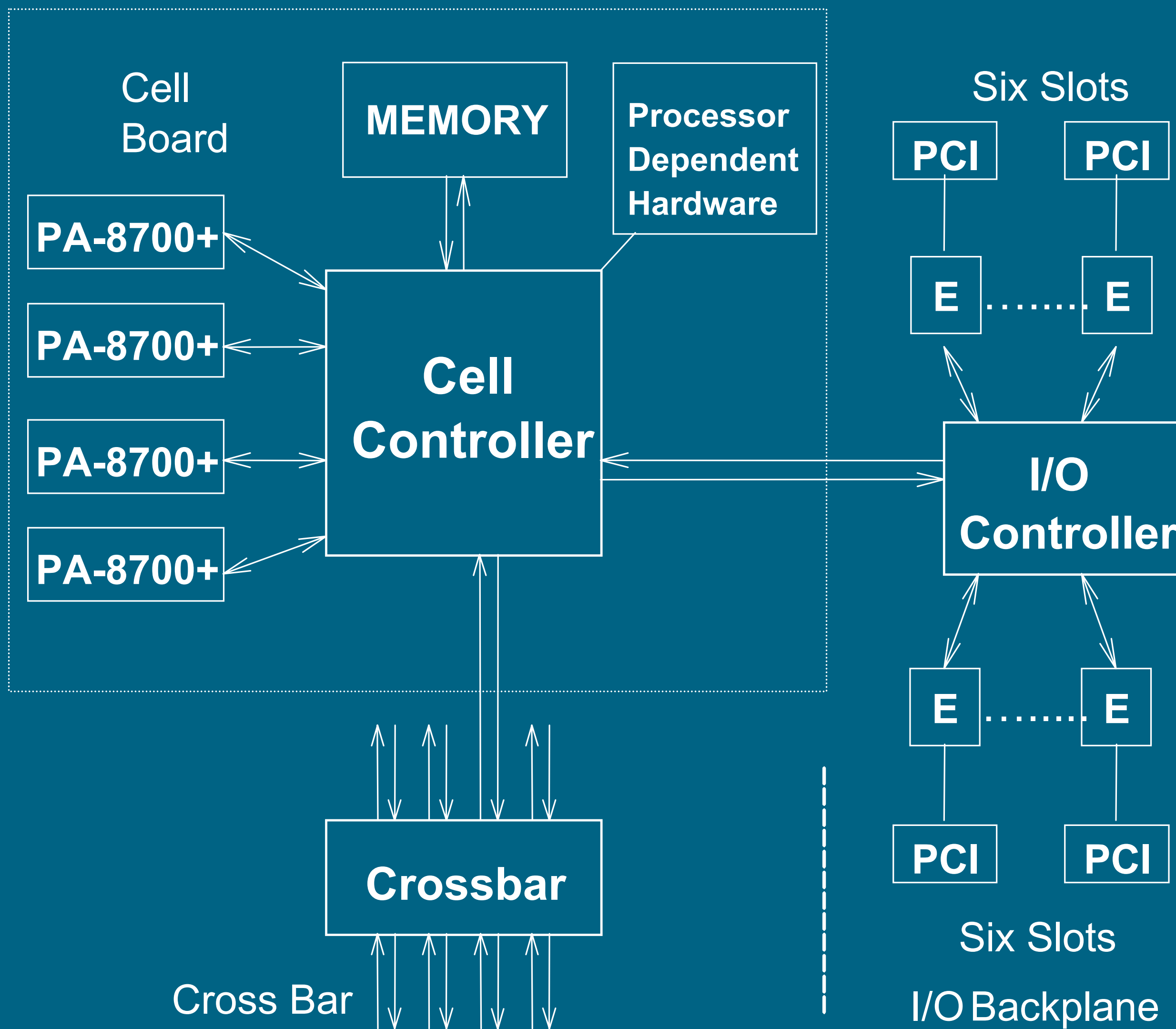
Built for the future
- initial release: PA-8700
- future releases: PA-RISC & Itanium
- Multi-OS: HP-UX, Linux and Windows

superdome cells

superdome is a cell-based hierarchical cross-bar system.

A cell consists of

- 4 CPUs
- 2 to 16GBs of Memory
- A link to 12 PCI I/O Slots (optional)



"Pinnacles" generation superdome cell

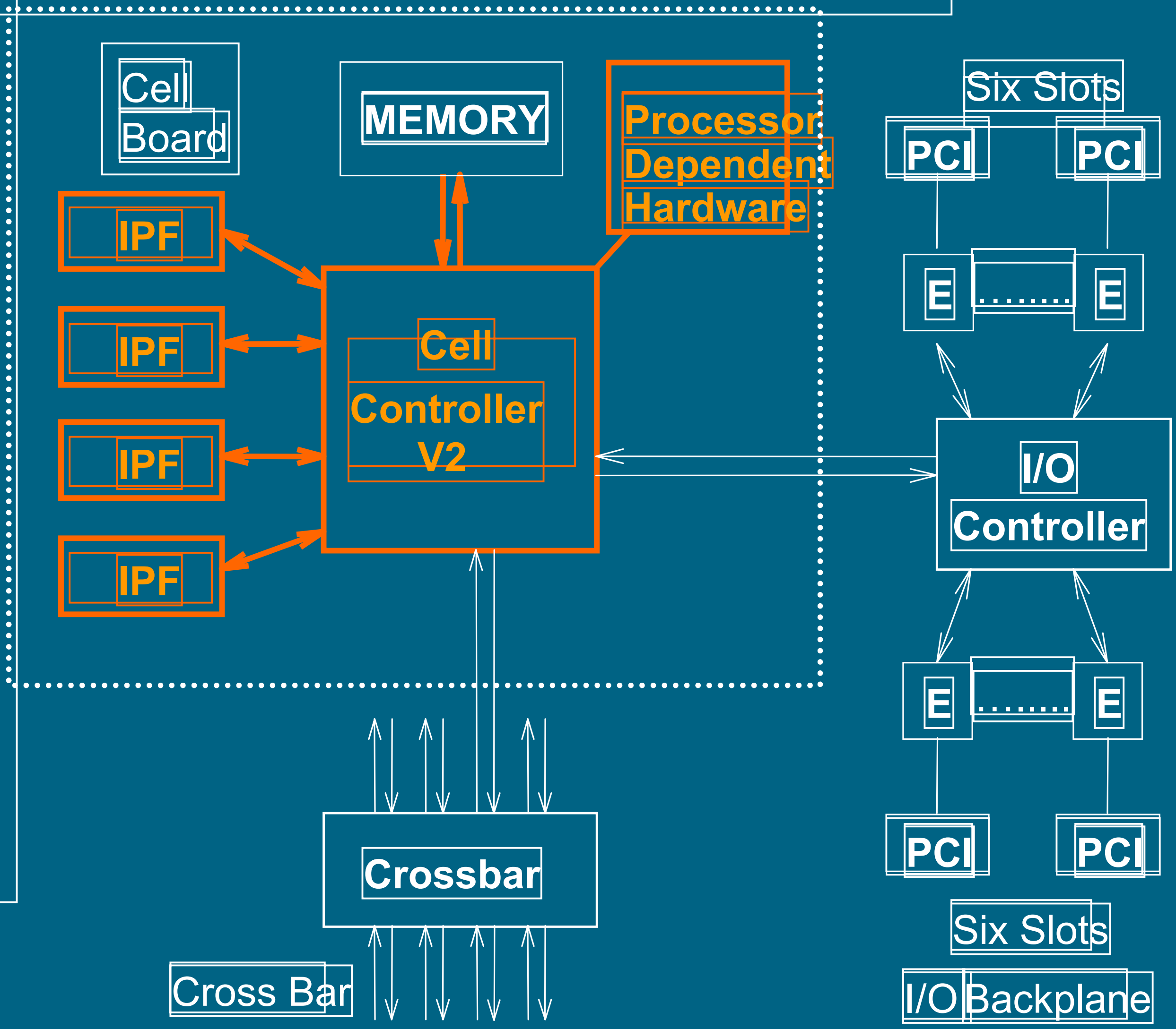
Supports PA 8800, 8900, and all Itanium® CPUs

Cell controller, cell board, and CPUs change (orange)

Memory DIMMS, I/O connection, and crossbar connection remain the same

All other system infrastructure (frame, backplane, I/O chassis, etc.) is preserved

Optional PCI-X I/O slots can also be used

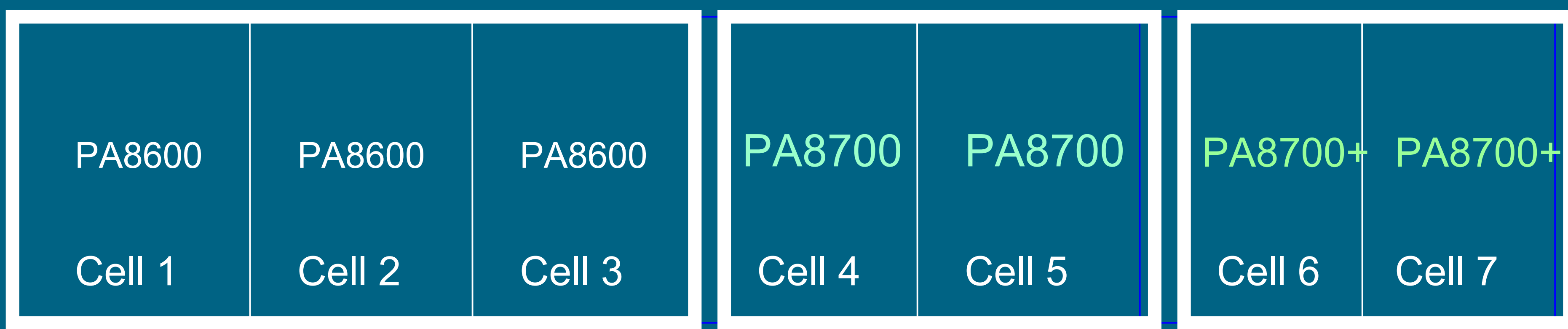


Superdome Investment Protection and Upgrade Example

Partition 1
12 CPUs

Partition 2
8 CPUs

Partition 3
8 CPUs



Partition 1: keep PA8600s for investment protection

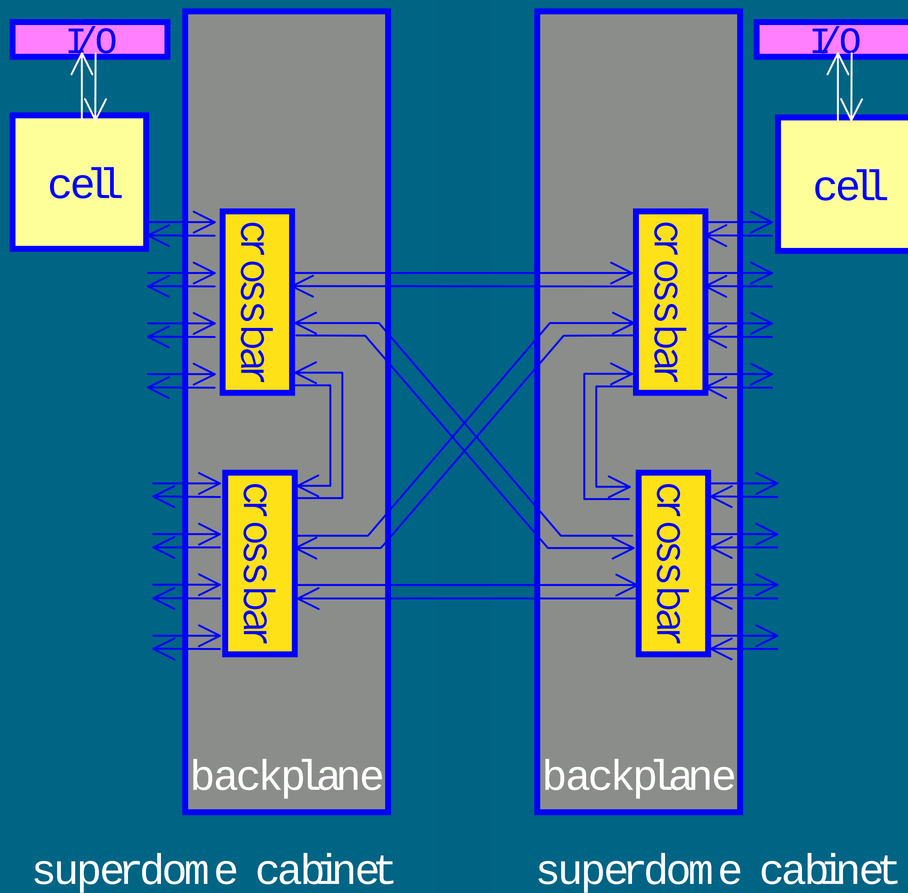
Partition 3: upgrade to PA8700+ in month 4

Partition 2: upgrade to PA8700 in month 1

Can upgrade to PA8700 on line <u>one partition at a time</u> so applications running in other partitions can keep running.

Interconnect Fabric: Crossbar Mesh

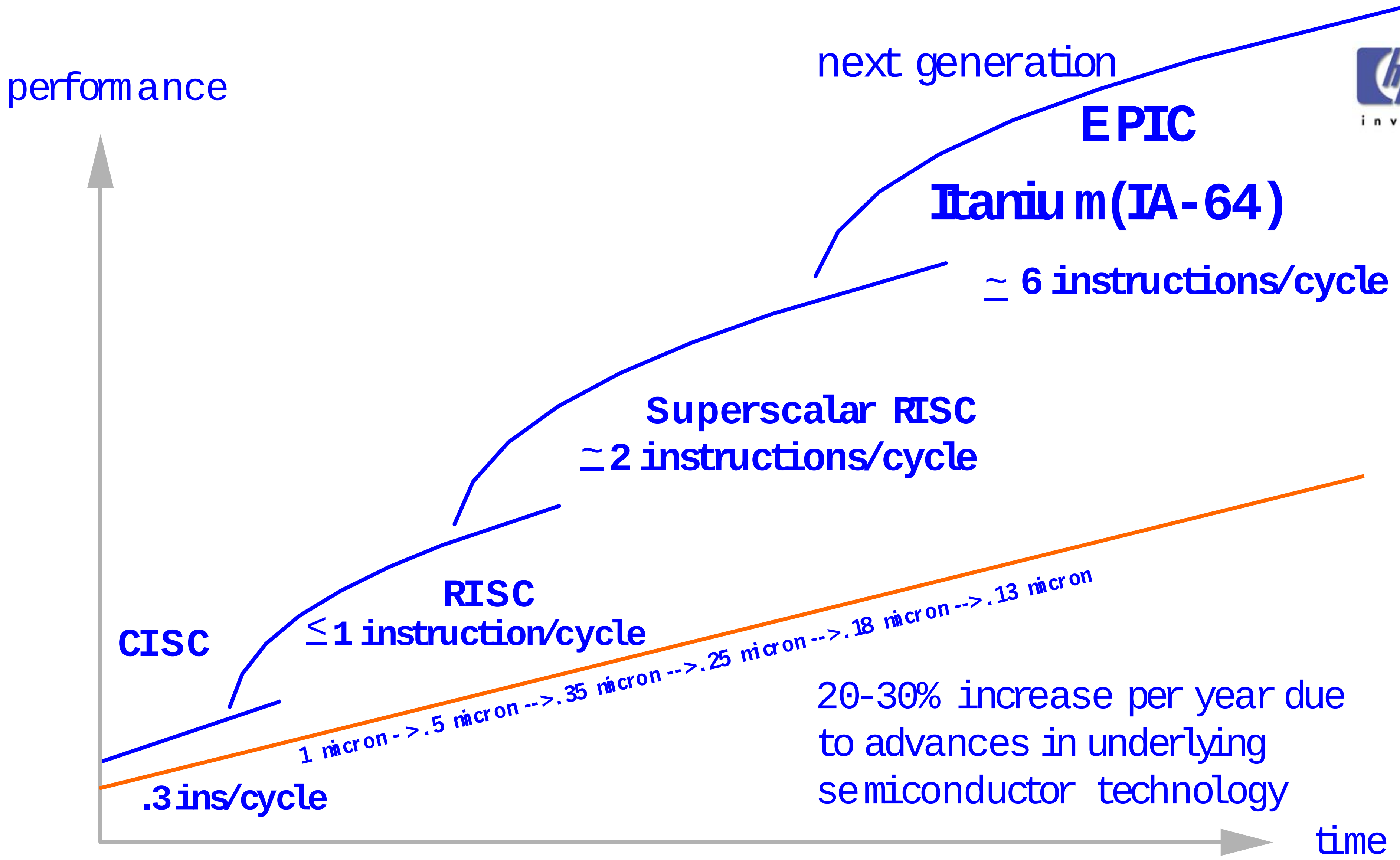
- Fully-connected crossbar mesh
  - Four crossbars
  - Four cells per crossbar
- All links have equal bandwidth and latency
  - Minimizes latency
  - Maximizes usable bandwidth
- Implements point-to-point packet filtering and routing network
  - Allows hardware isolation of all faults
- Interconnect 16 cells with 3 latency domains
  - Cell local
  - Crossbar local
  - Remote crossbar



Processors	SuperDome	UE10K	S80	GS320
4	200	600	?	325
8	250	600	?	635
16	275	600	?	790
32	315	600	X	870
64	335	600	X	X

HP's Itanium Strategy

processor evolution



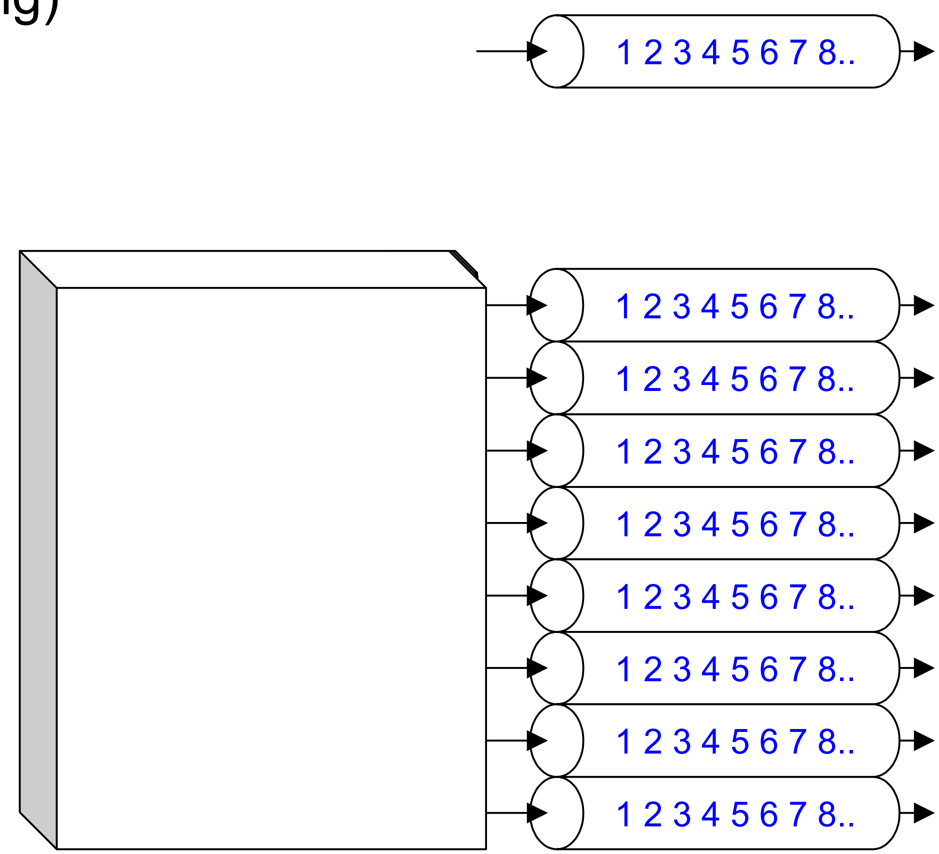
CPU architectures

RISC (reduced instruction set computing)

- Pipeline stages run in parallel

Superscalar RISC

- Multiple parallel pipelines
- Hardware schedules instructions and evaluates potential conflicts
- code parallelisation at runtime

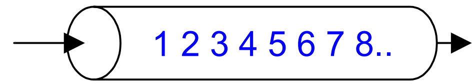


Scheduler area grows as the square of the number of pipelines

CPU architectures

RISC (reduced instruction set computing)

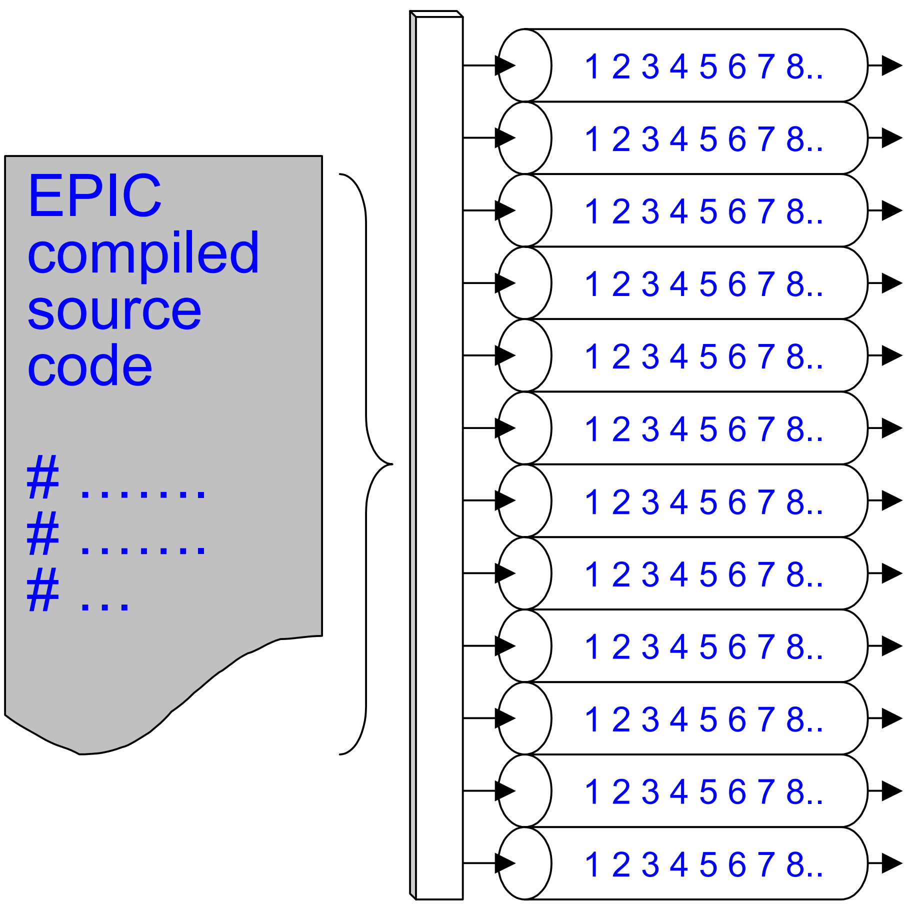
- Pipeline stages run in parallel



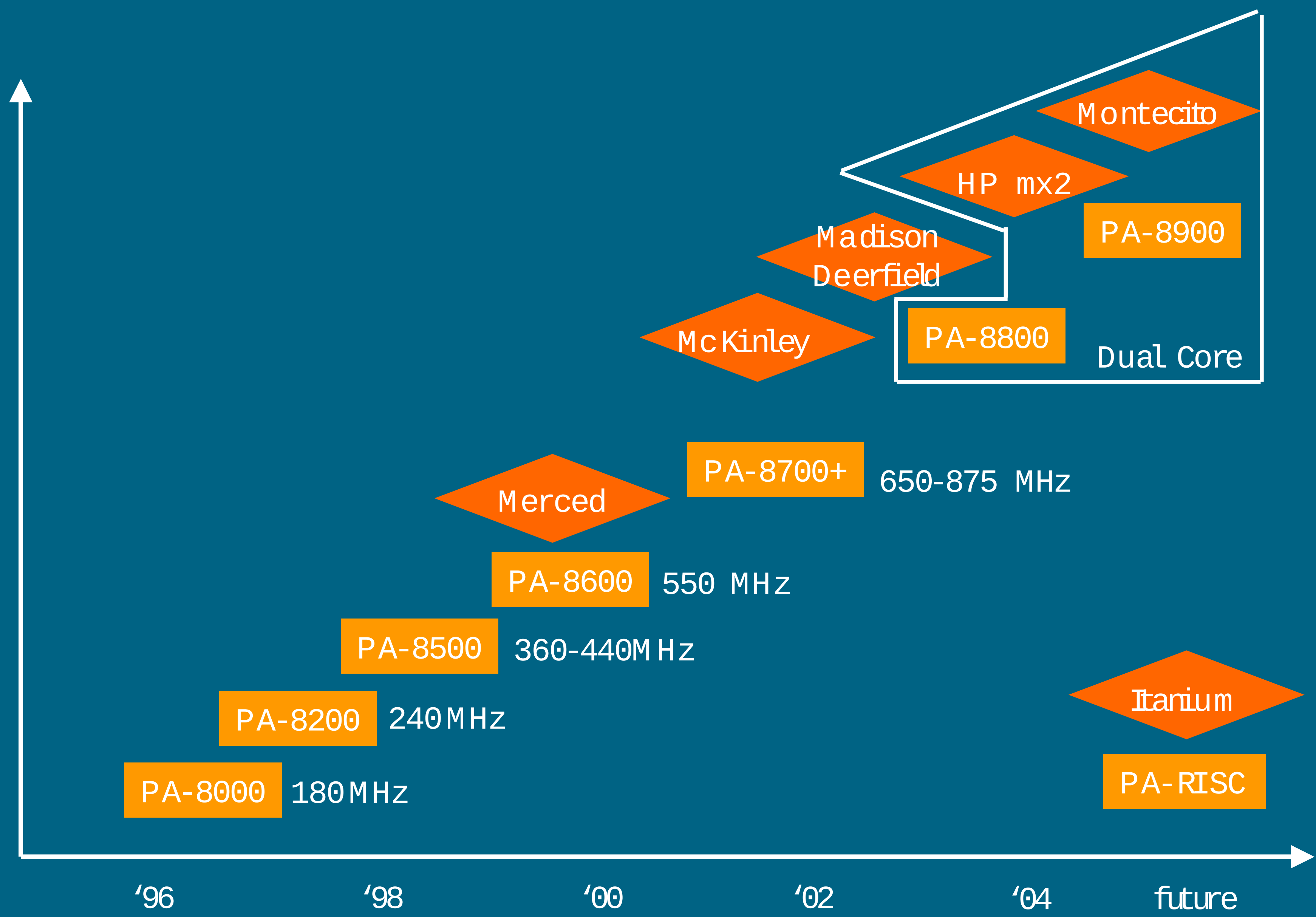
EPIC

(explicit parallel instruction comp.)

- Compiler schedules instructions and guarantees independence
- very large number of parallel pipelines possible
- code parallelisation at compiling



Microprocessor Roadmap
PA-RISC and Itanium



Itanium® Processor Family Roadmap

	2002	2003	2004	2005
Intel® Itanium® 2 Processor	Itanium® 2 Processor (1 GHz, 3MB L3)	Itanium® 2 Processor (Madison) (1.5GHz, 6MB L3)	Itanium® 2 Processor (Madison 9M) (>1.5GHz, 9MB L3)	Montecito (Dual Core)
Low Voltage Intel® Itanium® 2 Processor		Low Voltage Itanium® 2 Processor (Deerfield) (1.0GHz, 1.5MB L3, DP)	Low Voltage Itanium® 2 Processor (Deerfield refresh) (>1.0GHz, same platform)	Deerfield Follow-on (Same or lower power envelope, enhanced microarchitecture)

Silicon Process

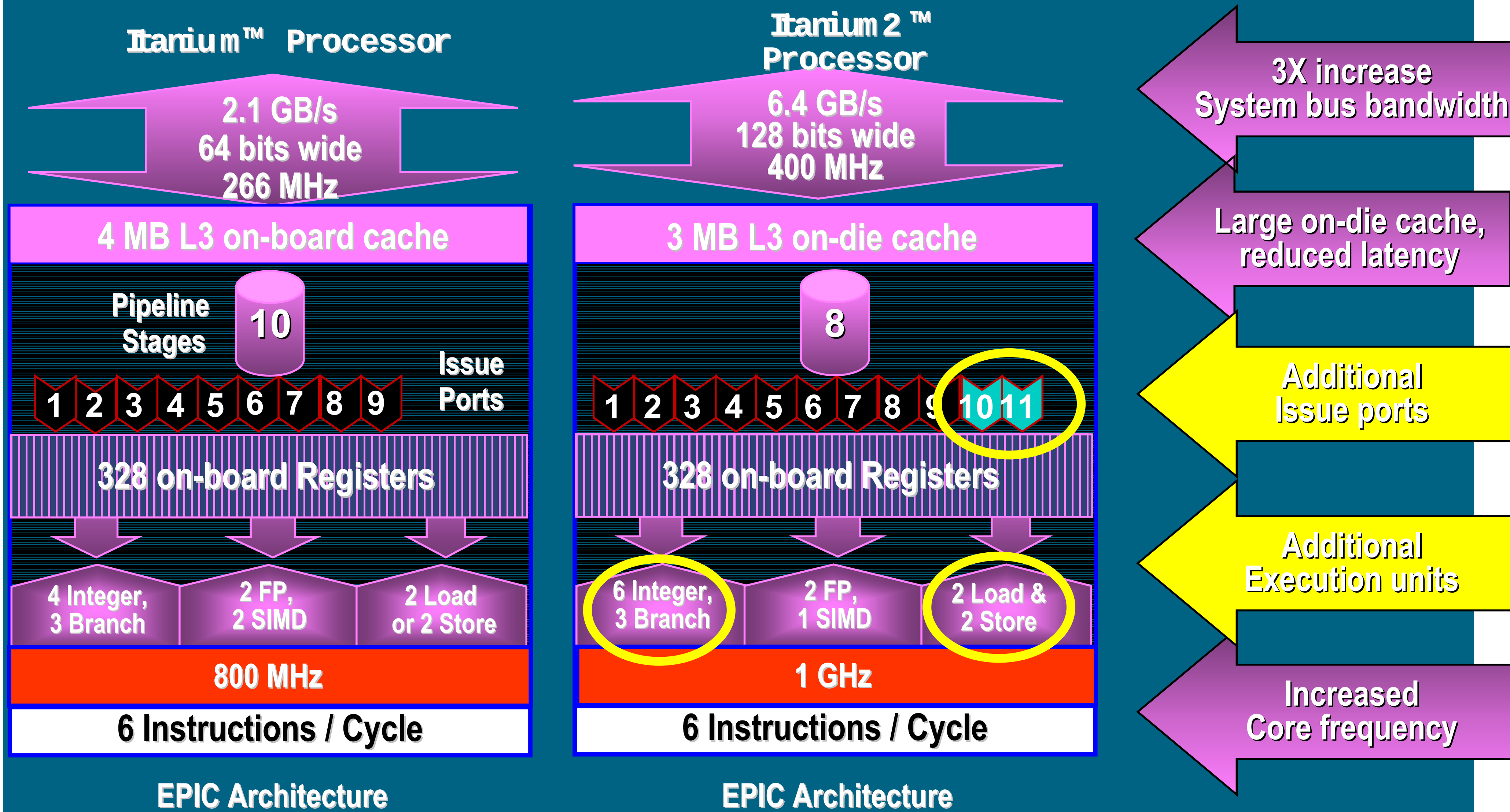
0.18 µm
0.13 µm
90 nm

- Next Itanium® 2 processor (Madison) on track for production in June-July
- New Low Voltage Itanium® 2 processor (Deerfield) follows in 2H'03
- Itanium® 2 platform maintains same socket, bus and software compatibility
- Intel will enhance Itanium® 2 processors (Madison and Deerfield) in 2004
- Montecito processor will enable dual-core technology and enhanced microarchitecture in 2005

All dates specified are target dates, are provided for planning purposes only and are subject to change.

Roadmap maintains world class performance

Building Out the Itanium™ Architecture



McKinley enhancements build on Itanium Architecture foundation

Making multi-operating systems work

HP-UX 11i

Non-Stop Kernel

OpenVMS

Microsoft Windows

We are investing in

- HP-UX
  - Incorporate the best of Tru64 UNIX functionality into HP-UX
- Windows®
  - Lead the migration to .NET®
- Linux
  - Contribute IP to Linux community
- OpenVMS
- Non-Stop Kernel

Multi-OS capabilities

- IT virtualization technologies
- Security
  - Single sign-on
- High-availability
- Common management
  - One system management environment

hp zx1 chipset unleashes the full power of Itanium 2

high memory bandwidth, low memory latency

- enables top application performance
- consistent response times
- supports more users and processes

high memory capacity

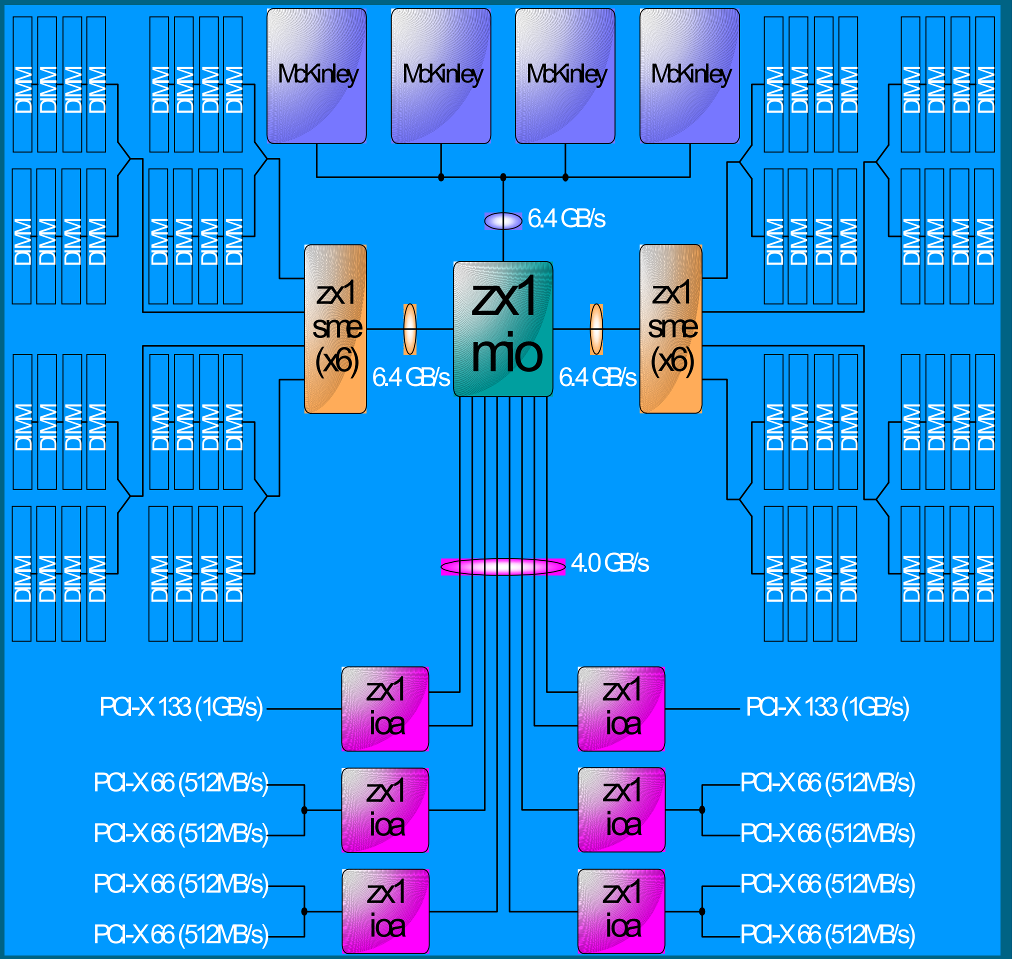
- supports DDR RAM
- enables optimum performance for large models/databases

high I/O bandwidth and capacity

- consolidate applications to reduce number of servers
- very large databases or multiple large DB
- four high-speed channels provide ~4 GB/s available bandwidth

scalability

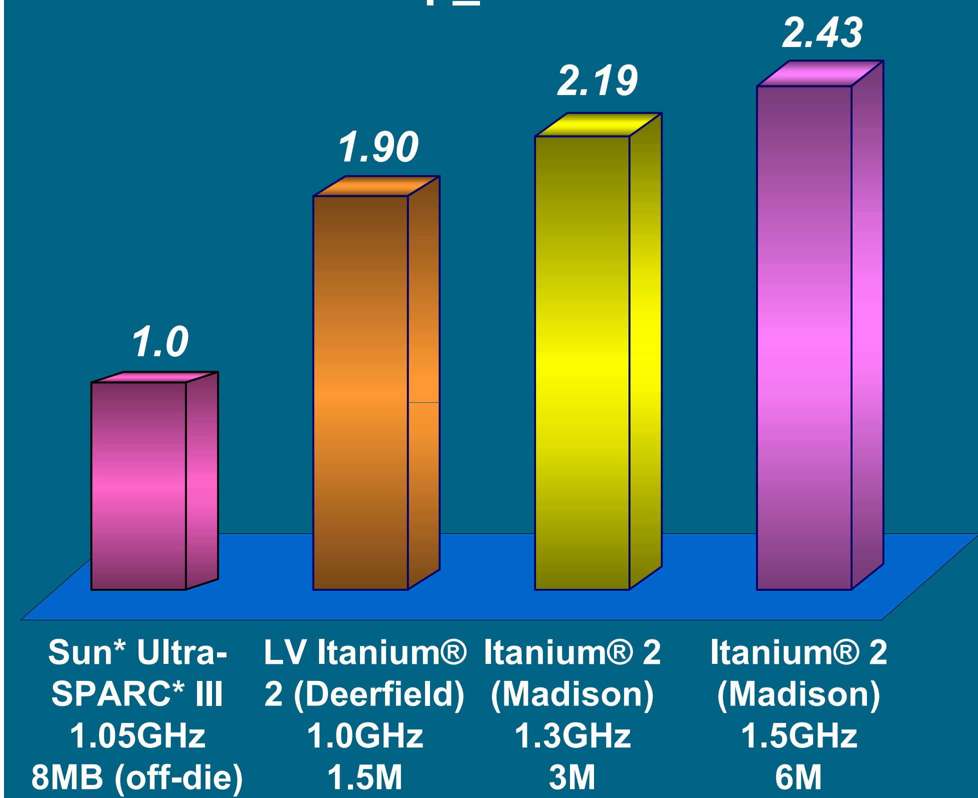
- enables a family of systems to be tuned to meet a variety of needs



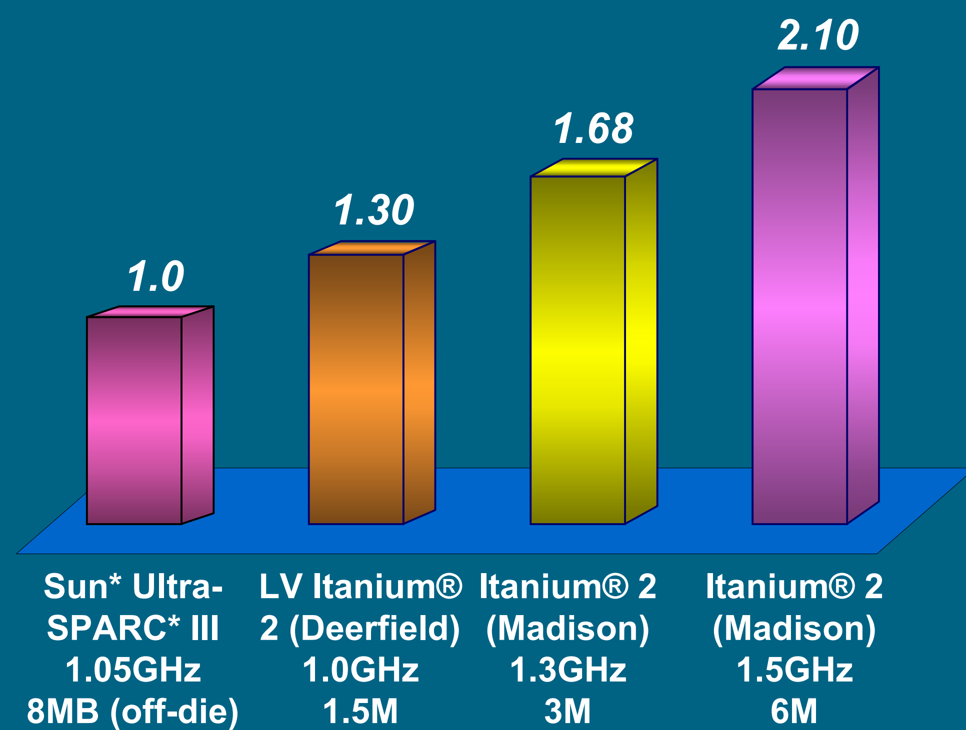
the fastest Itanium 2 platforms available today

Outstanding Performance from New Intel® Itanium® 2 Processors

Floating Point Performance
SPECfp_base2000



Integer Performance
SPECint_base2000



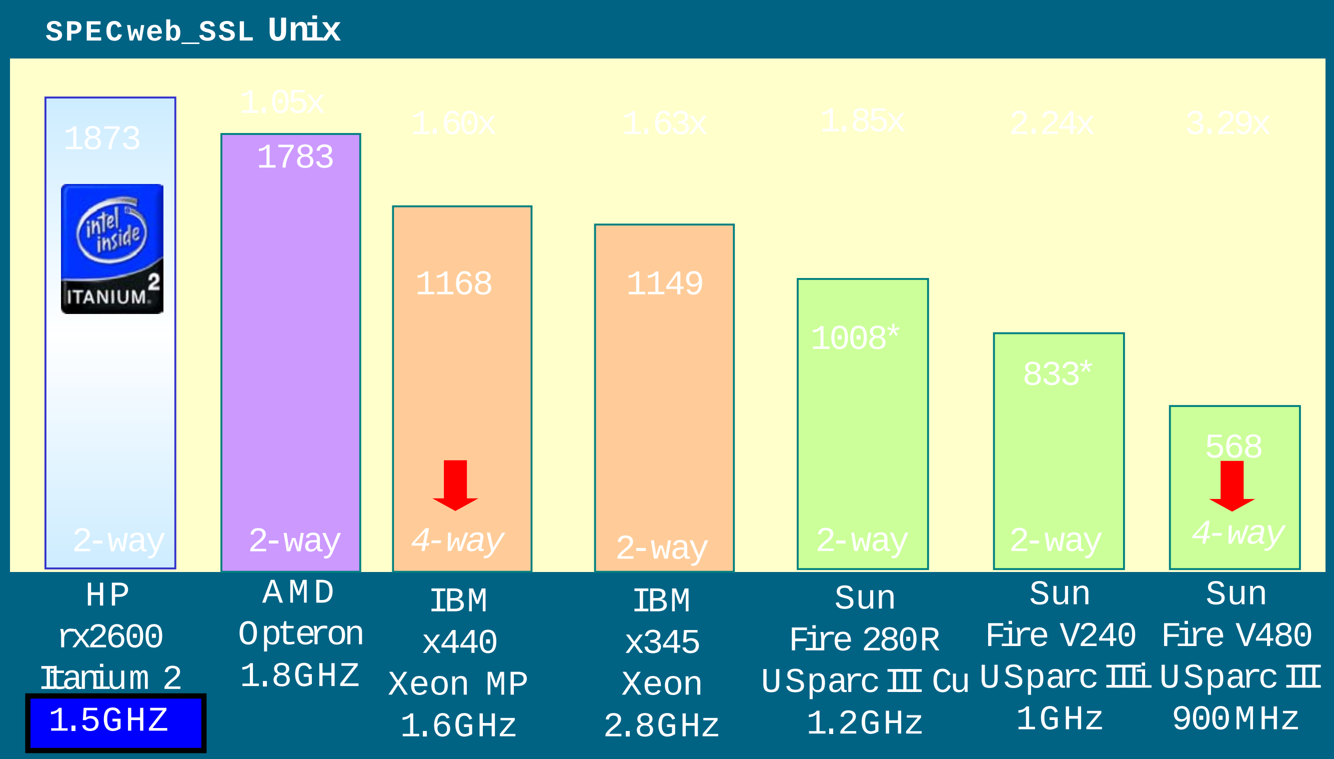
Sources: Sun – www.spec.org; Intel – Intel estimate (12/06 Munce)

Deerfield significantly outperforms Sun in both floating point and integer performance

2-way SLL encryption comparison

#1 2-way for encryption/decryption

HP Server rx2600 with next generation Itanium 2 processors tops all other 2-way servers for SSL performance



*Sun result achieved through use of dedicated crypto accelerators

Hard Partition (nPar) Terms and Comands:

- Base Cells
- Assigned Cells
- Unassigned Cells
- Core Cells
- Active Cells
- Inactive Cells
- Genesis nPartition
- Partition Numbers

- ioscan
- rad
- parmgr
- parcreate
- parmodify
- parremove
- parstatus
- parunlock
- fruled
- frupower

parstatus -P

parstatus -P to get some high-level information about the nPartitions in our system:

```
mtvnhp01:/>parstatus -P
[Partition]
Par                # of  # of I/O
Num Status         Cells Chassis  Core cell  Partition Name
=== ============ ===== ======== ========== ==============
 0  active         2       1     cab0,cell0 mtvnhp01
 1  active         1       1     cab0,cell1 mtvnhp02
mtvnhp01:/>
```

This output shows that we have two nPartitions on our rp8400.
The first of the nPartitions has two cells and the second has one.

Next, we'll issue the **parstatus** command with a verbose output of nPartition *p0:*

parstatus –V –p0

```
# parstatus -V -p0
[Partition]
Partition Number        : 0
Partition Name          : P1
Status                  : active
IP address              : 0.0.0.0
Primary Boot Path       : 0/0/1/0/0.0.0
Alternate Boot Path     : 0/0/6/0/0.1.0
HA Alternate Boot Path  : 0/0/1/0/0.6.0
PDC Revision            : 32.5
IODCH Version           : 5E70
CPU Speed               : 750 MHz
Core Cell               : cab0,cell0

[Cell]
                        CPU     Memory                                    Use
                        OK/     (GB)                              Core    On
Hardware   Actual       Deconf/ OK/                               Cell    Next Par
Location   Usage        Max     Deconf    Connected To            Capable Boot Num
========== ============ ======= ========= ======================= ======= ==== ===
cab0,cell0 active core  4/0/4    4.0/12.0 cab0,bay1,chassis3      yes     yes  0
cab0,cell1 active base  4/0/4    4.0/12.0 cab0,bay1,chassis1      yes     yes  0
cab0,cell2 active base  4/0/4    4.0/12.0 -                       no      yes  0
cab0,cell3 active base  4/0/4    4.0/12.0 -                       no      yes  0
cab0,cell4 active base  4/0/4    4.0/12.0 cab0,bay0,chassis1      yes     yes  0

[Chassis]
                                 Core Connected  Par
Hardware Location   Usage        IO   To         Num
=================== ============ ==== ========== ===
cab0,bay1,chassis3  active       yes  cab0,cell0 0
cab0,bay1,chassis1  active       yes  cab0,cell1 0
cab0,bay0,chassis1  active       yes  cab0,cell4 0
```

parstatus –w
parstatus –C

To determine the local nPartition on a system we'd issue **parstatus -w** as shown in the following output:

```
mtvnhp01:/>parstatus -w
The local partition number is 0.
mtvnhp01:/>
```

The next command we'll issue is **parstatus -C** to get details about the way in which our cells are configured, as shown in the fo

```
mtvnhp01:/>parstatus -C
[Cell]
                        CPU     Memory                                Use
                        OK/     (GB)                          Core    On
Hardware   Actual       Deconf/ OK/                           Cell    Next Par
Location   Usage        Max     Deconf    Connected To        Capable Boot Num
========== ============ ======= ========= =================== ======= ==== ===
cab0,cell0 active core  4/0/4    8.0/ 0.0 cab0,bay0,chassis0  yes     yes  0
cab0,cell1 active core  4/0/4    4.0/ 0.0 cab0,bay0,chassis1  yes     yes  1
cab0,cell2 active base  4/0/4    8.0/ 0.0 -                   no      yes  0
cab0,cell3 absent       -       -         -                   -       -    -

mtvnhp01:/>
```

The **parstatus** command without any options produces a more detailed list of results, as shown in the following output:

parstatus (No option)

```
mtvnhp02:/>parstatus
Warning: No action specified. Default behaviour is display all.
Compute Cabinet (4 cell capable) : 1
   Active GSP Location : cabinet 0
The total number of Partitions Present : 2

[Cabinet]
                 Cabinet   I/O       Bulk Power
                 Fans      Fans      Supplies
                 OK/       OK/       OK/
Cab              Failed/   Failed/   Failed/
Num Cabinet Type N Status  N Status  N Status     GSP
=== ============ ========= ========= ==========   ======
 0  S16K-A       21/ 0/ N+  6/ 0/ N+  6/ 0/ N+    active

[Cell]
                        CPU     Memory                                  Use
                        OK/     (GB)                            Core    On
Hardware   Actual       Deconf/ OK/                             Cell    Next Par
Location   Usage        Max     Deconf    Connected To          Capable Boot Num
========== ============ ======= ========= ===================== ======= ==== ===
cab0,cell0 active core  4/0/4    8.0/ 0.0 cab0,bay0,chassis0    yes     yes  0
cab0,cell1 active core  4/0/4    4.0/ 0.0 cab0,bay0,chassis1    yes     yes  1
cab0,cell2 active base  4/0/4    8.0/ 0.0 -                     no      yes  0
cab0,cell3 absent       -       -         -                     -       -    -

[Chassis]                                Core Connected  Par
Hardware Location   Usage        IO   To         Num
=================== ============ ==== ========== ===
cab0,bay0,chassis0  active       yes  cab0,cell0 0
cab0,bay0,chassis1  active       yes  cab0,cell1 1

[Partition]
Par              # of  # of I/O
Num Status       Cells Chassis  Core cell  Partition Name (first 30 chars)
=== ============ ===== ======== ========== ===============================
 0  active         2      1     cab0,cell0 mtvnhp01
 1  active         1      1     cab0,cell1 mtvnhp02
mtvnhp02:/>
```

```
# rad -q
                                                                               Driver(s)
Slot         Path         Bus    Speed    Power    Occupied     Suspended     Capable
0-0-1-0      4/0/0        0      33       On       Yes          No            No
0-0-1-1      4/0/1/0      8      33       On       Yes          No            Yes
0-0-1-2      4/0/2/0      16     33       On       Yes          No            Yes
0-0-1-3      4/0/3/0      24     33       On       Yes          No            Yes
0-0-1-4      4/0/4/0      32     66       On       Yes          No            Yes
0-0-1-5      4/0/6/0      48     33       On       Yes          No            Yes
0-0-1-6      4/0/14/0     112    66       On       Yes          No            Yes
0-0-1-7      4/0/12/0     96     33       On       Yes          No            Yes
0-0-1-8      4/0/11/0     88     66       On       Yes          No            Yes
0-0-1-9      4/0/10/0     80     33       On       Yes          No            Yes
0-0-1-10     4/0/9/0      72     33       On       Yes          No            Yes
0-0-1-11     4/0/8/0      64     33       On       Yes          No            Yes
0-0-3-0      5/0/0        0      33       On       Yes          No            No
0-0-3-1      5/0/1/0      8      33       On       Yes          No            Yes

          .
          .
          .
0-1-3-8      0/0/11/0     88     66       On       Yes          No            Yes
0-1-3-9      0/0/10/0     80     33       On       Yes          No            Yes
0-1-3-10     0/0/9/0      72     33       On       Yes          No            Yes
0-1-3-11     0/0/8/0      64     33       On       Yes          No            Yes
#
```

rad -q

The first field is the slot information, which is in the following form:

```
Cabinet-Bay-Chassis-Slot        such as 0-0-1-0 for the first entry
```

The second field, which is the *Path*, contains the following

```
Cell/SBA/LBA/Device             such as 0/0/1/1 for the second entry
```

The following listing shows an **ioscan** output of the second nPartition in the same rp8400:

ioscan –f (second nPar)

```
mtvnhp02:/>ioscan -f
Class        I  H/W Path       Driver    S/W State   H/W Type     Description
==============================================================================
root         0                 root      CLAIMED     BUS_NEXUS
cell         0  1              cell      CLAIMED     BUS_NEXUS
ioa          0  1/0            sba       CLAIMED     BUS_NEXUS    System Bus Adapter
ba           0  1/0/0          lba       CLAIMED     BUS_NEXUS    Local PCI Bus Adapter
unknown     -1  1/0/0/0/0                UNCLAIMED   UNKNOWN      PCI BaseSystem
tty          0  1/0/0/0/1      asio0     CLAIMED     INTERFACE    PCI Serial
lan          0  1/0/0/1/0      gelan     CLAIMED     INTERFACE    HP A3639-60019
 1000Base-T Built-in I/O
ext_bus      0  1/0/0/2/0      c720      CLAIMED     INTERFACE    SCSI C87x Ultr
a Wide Single-Ended
target       0  1/0/0/2/0.6    tgt       CLAIMED     DEVICE
disk         0  1/0/0/2/0.6.0  sdisk     CLAIMED     DEVICE       HP 36.4GST3367
06LC

        .
        .
        .
ba           9  1/0/14         lba       CLAIMED     BUS_NEXUS    Local PCI Bus
Adapter (782)
memory       0  1/5            memory    CLAIMED     MEMORY       Memory
processor    0  1/10           processor CLAIMED     PROCESSOR    Processor
processor    1  1/11           processor CLAIMED     PROCESSOR    Processor
processor    2  1/12           processor CLAIMED     PROCESSOR    Processor
processor    3  1/13           processor CLAIMED     PROCESSOR    Processor
mtvnhp02:/>
```

This **ioscan** output includes all components in nPartition *0* in this rp8400. Cell *1* is shown in this **ioscan** output which we'll confirm later. The **ioscan** outputs produced a lot of components for our local partition. The form of the **ioscan** output for an nPartition looks like the foll

```
    Field 1          Field 2     Field 3     Field 4     Field 5   Field 6

Global cell no./proc, mem, or SBA/LBA/Card address/Function/dev addr
```

The following **parcreate** commands create a new nPartition:

parcreate and parmodify

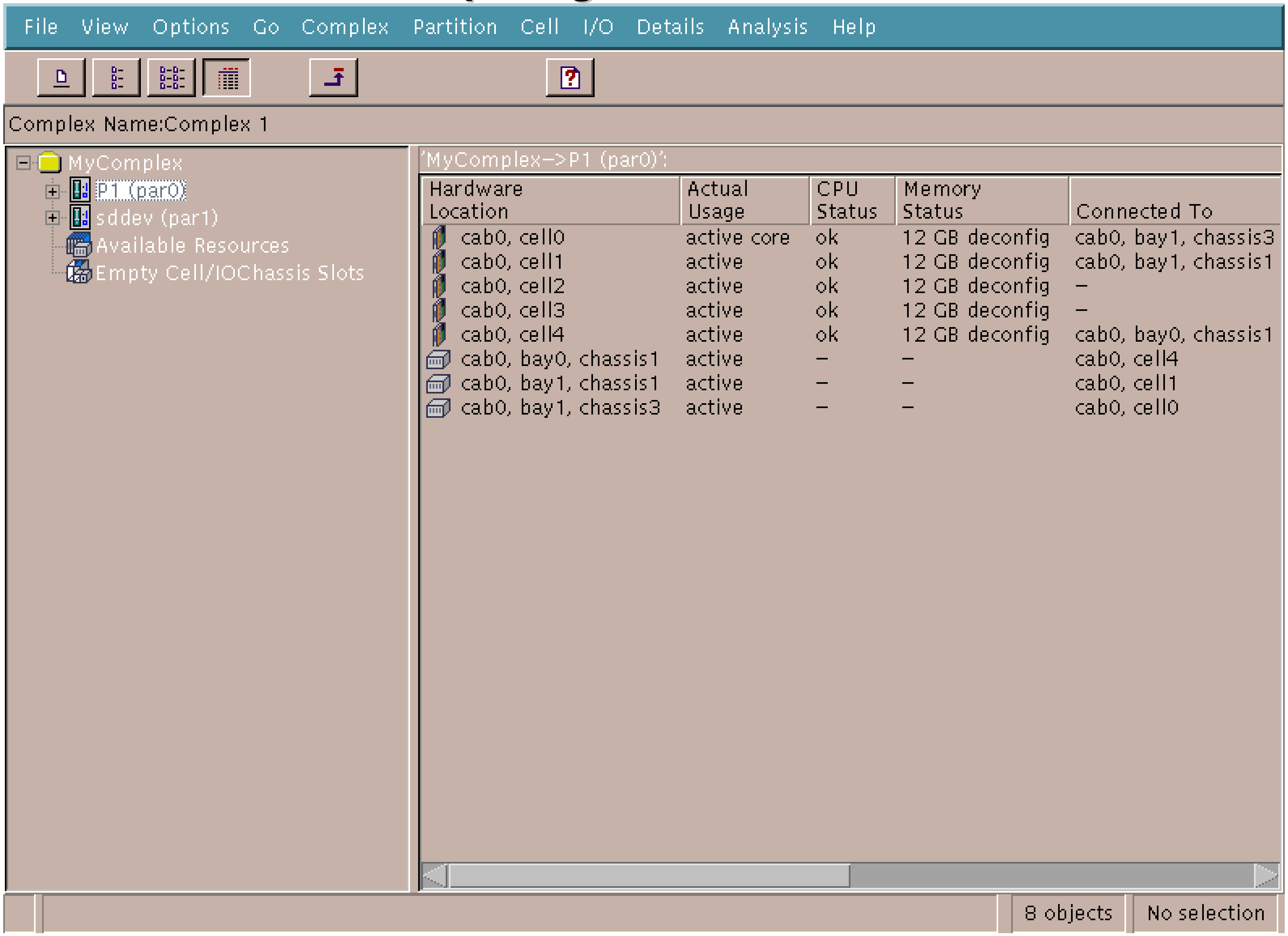
```
# parcreate -P sddev -c5:base:y:ri -c6:base:y:ri -b 0/0/1/0/0.1
Partition Created. The partition number is: 1
#
```

The two cell boards are added with *base* (*base* cell is the only valid type at this time,) *y* (which means that the cell will p participate in the reboot), and *ri* (which is *reactive with interleave* which is the only valid value at this time). We left *cel* *Cell 7* unused by not including it in *sddev*.

Next we'll remove *cell7* from this nPartition with **parmodify**. Notice in the following listing that after we run **parmodify** on *p0*, **shutdown -R** is required to make the removal of *cell7* from *p0*:

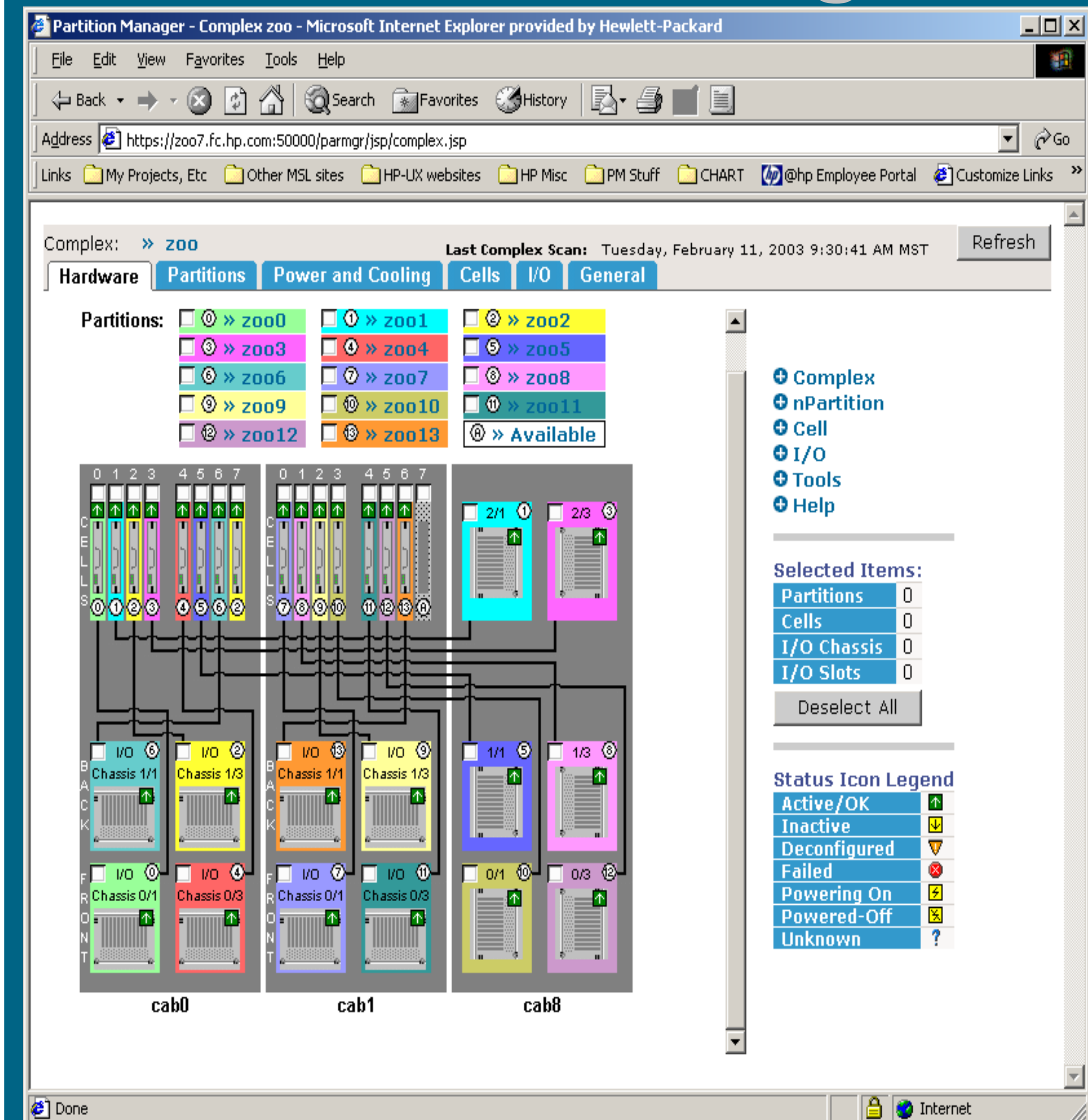
```
# parmodify -p0 -d7 -B
Cell 7 is active.
Use shutdown -R to shutdown the system to ready for reconfig state.
Command succeeded.
#
```

parmgr Main Screen

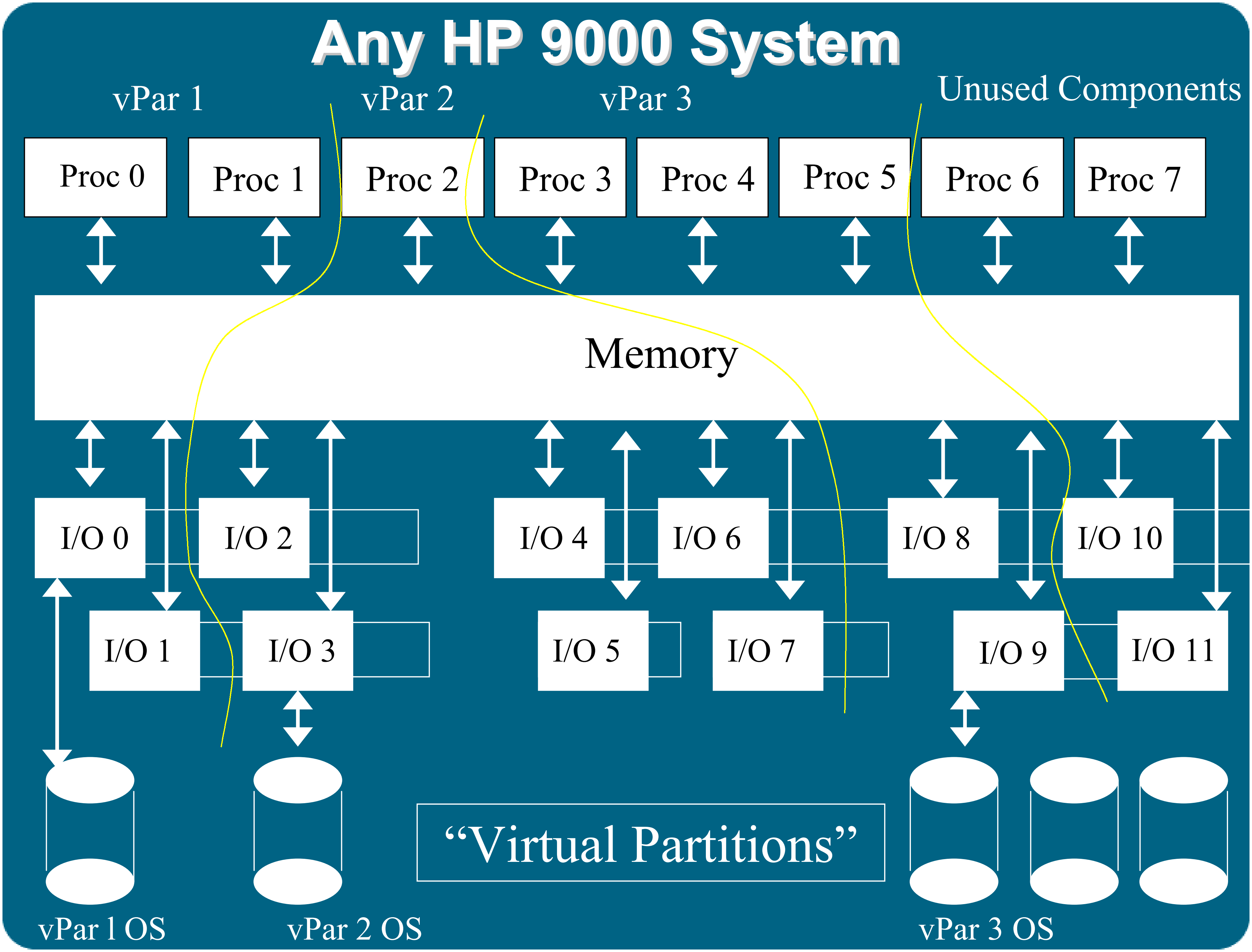


The left side of the **parmgr** screen shows the first nPartition (*P1*), the second nPartition (*sddev*), *Available Resources*, and *Empty* components. When one of these is selected, its details are shown in the right window. *P1* is selected in the left window so we see its components in the right window. Note that only *cell0-4* are part of *P1* because we removed *cell7* from it earlier at the command line.

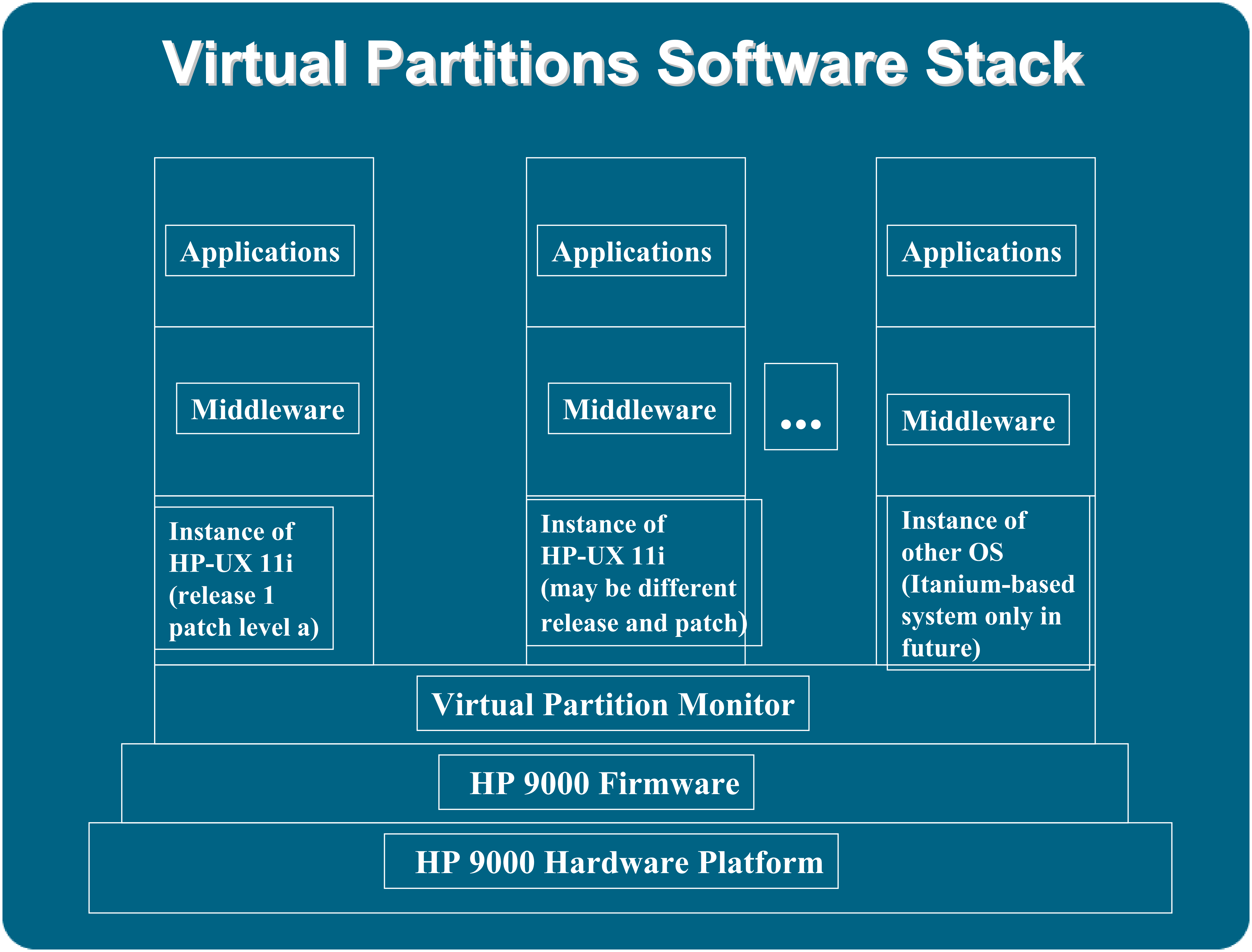
Partition Manager New Features



- ✓ New web interface
- ✓ Graphical "big picture" views of
  - nPars
  - Hardware in Complex
- ✓ Supports new OS/HW features
  - Cell local memory for HP-UX 11i v.2 partitions
  - Inter-partition security
- ✓ Remote admin of Superdome Madison complex
- ✓ Compatible with iCOD/Pay-Per-Use
- ✓ Increased integration with SCM 3.0
- ✓ Native on Windows (2H03)
- ✓ J2EE app runs in tomcat web server



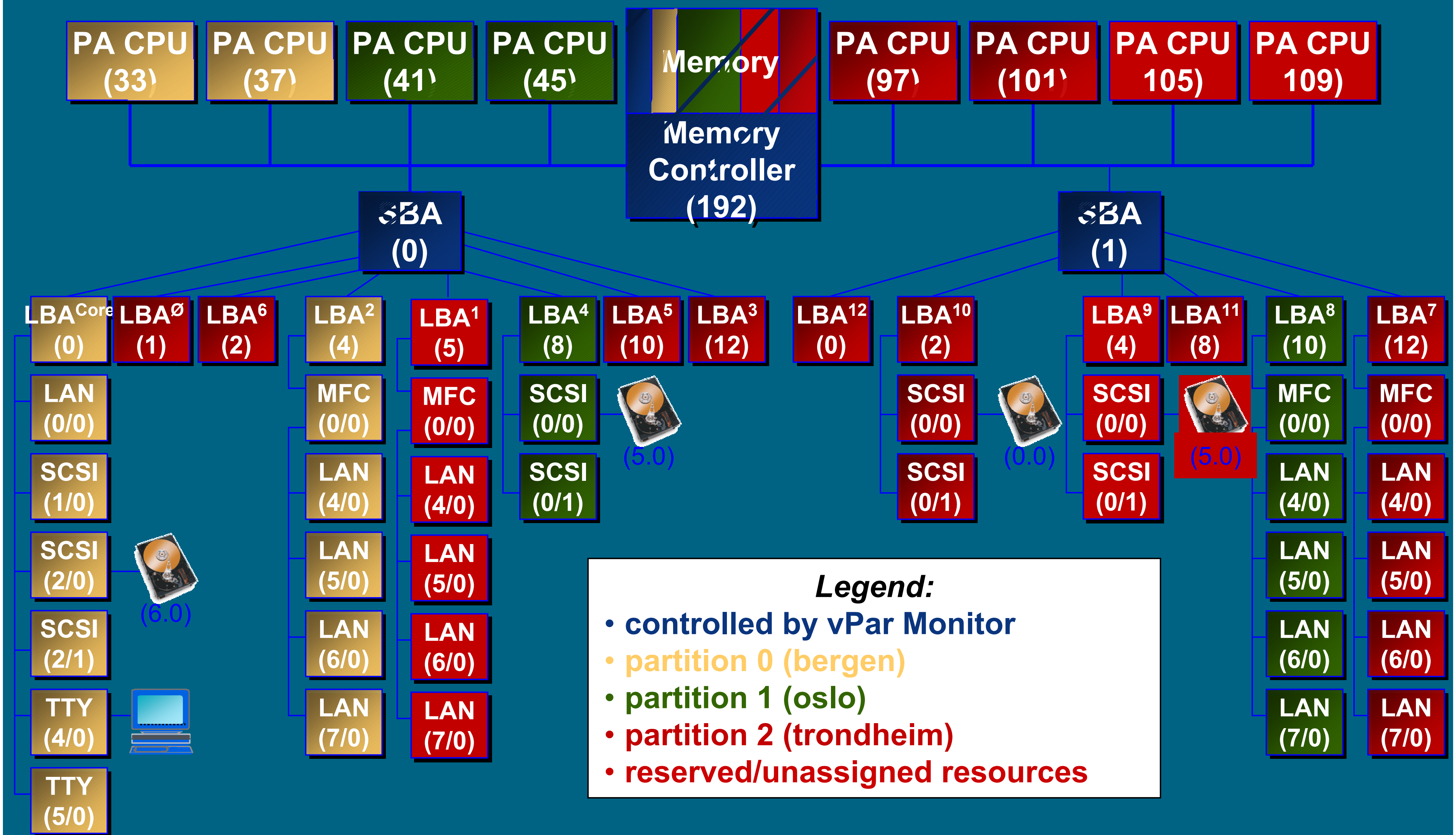
Any HP 9000 System
vPar 1
vPar 2
vPar 3
Unused Components
Proc 0
Proc 1
Proc 2
Proc 3
Proc 4
Proc 5
Proc 6
Proc 7
Memory
I/O 0
I/O 2
I/O 4
I/O 6
I/O 8
I/O 10
I/O 1
I/O 3
I/O 5
I/O 7
I/O 9
I/O 11
"Virtual Partitions"
vPar 1 OS
vPar 2 OS
vPar 3 OS


Virtual Partitions Software Stack
Applications
Applications
Applications
Middleware
Middleware
...
Middleware
Instance of HP-UX 11i (release 1 patch level a)
Instance of HP-UX 11i (may be different release and patch)
Instance of other OS (Itanium-based system only in future)
Virtual Partition Monitor
HP 9000 Firmware
HP 9000 Hardware Platform

rp7400 partition plan

vPar Number	0	1	2
vPar Name	Bergen	Oslo	Trondheim
CPUs	33, 37	41, 45	105, 109
Memory Ranges	0x01000000 to 0x07ffffff (112MB) 0x40000000 to 0x5fffffff(512MB)	0x08000000 to 0x0fffffff (128MB) 0x60000000 to 0x9fffffff (1024MB)	0x10000000 to 0x17ffffff (128MB) 0xA0000000 to 0xdfffffff (1024MB)
I/O Paths (LBAs)	0/0 0/4	0/8 1/10	0/5 1/4
Boot Path	0/0/2/0.6.0	0/8/0/0.5.0	1/4/0/0.5.0
Console	0/0/4/0 (Virtual)	Virtual	Virtual
Kernel Image	/stand/vmunix	/stand/vmunix	/stand/vmunix
Autoboot	On	On	On

partitioned rp7400 block diagram



note: command names subject to change

configuration & management commands

- *vparcreate* – create a new partition definition, with or without resources
- *vparremove* – destroy an existing partition definition
- *vparmodify*
  - add resources to an existing partition
  - remove resources from an existing partition
  - modify the attributes (e.g. boot path) of an existing partition
- *vparboot* – load and launch an operating system within an existing partition
- *vparreset* – stop/reset a partition
- *vparstatus*
  - display one or more partition definition(s) in human readable form
  - check the status of one or more partitions and/or the monitor

Virtual Partitions (vPars) Commands:

vparload
Load Virtual Partitions from *MON>* prompt only.

vparboot
Boot a Virtual Partition from the command line only.

vparcreate
Create a Virtual Partition.

- **vparmodify**
Modify the attributes of a Virtual Partition.

vparremove
Delete a Virtual Partition.

vparreset
Reset a Virtual Partition.

vparresources(5) man page
Provides description of Virtual Partitions and their resources.

vparstatus
Display the status of Virtual Partitions.

vpartition man page
Display information about the Virtual Partition Command Line Interface.

vPars **setboot** Options:
-a
-b
-p
-s
no options

At boot time:

Virtual Partitions Monitor is loaded from *ISL>* with:
`ISL>` **`hpux /stand/vpmon`**

To load one vPar from *MON* , use:
`MON>` **`vparload`** *`vPar_name`*

List components that you would like in a vPar:

```
name      cable1
processors        min of one (bound) max of three (two unbound)
                  with num (bound + umnbound) equal to one
memory    1024 MB
LBA Core I/O 0/0 (all components on 0/0 are implied)
LAN       0/0/0/0 (not specified explicitly, on 0/0)
boot disk         0/0/1/1.2.0
kernel    /stand/vmunix (this is default)
autoboot off (manual)
console   0/0/4/0 (not specified explicitly, on 0/0)
```

Rad -q

To create a Virtual Partition with three processors (*num*) total, two bound (*min*) , 2048MB RAM, all components on 0/0, boot disk at 0/ 0/1/1.2.0, with a kernel of **/stand/vmunix**, autoboot on, and console at 0/0/4/0:

```
# vparcreate -p vPar_name -a cpu::3 -a cpu:::2:4
  -a mem::2048 -a io:0/0 -a io:0/0/1/1.2.0:boot
  -b /stand/vmunix -B auto
```

To add processor at path *109* (adds this proc to those already assigned):

```
# vparmodify -p vPar_name -a cpu:109
```

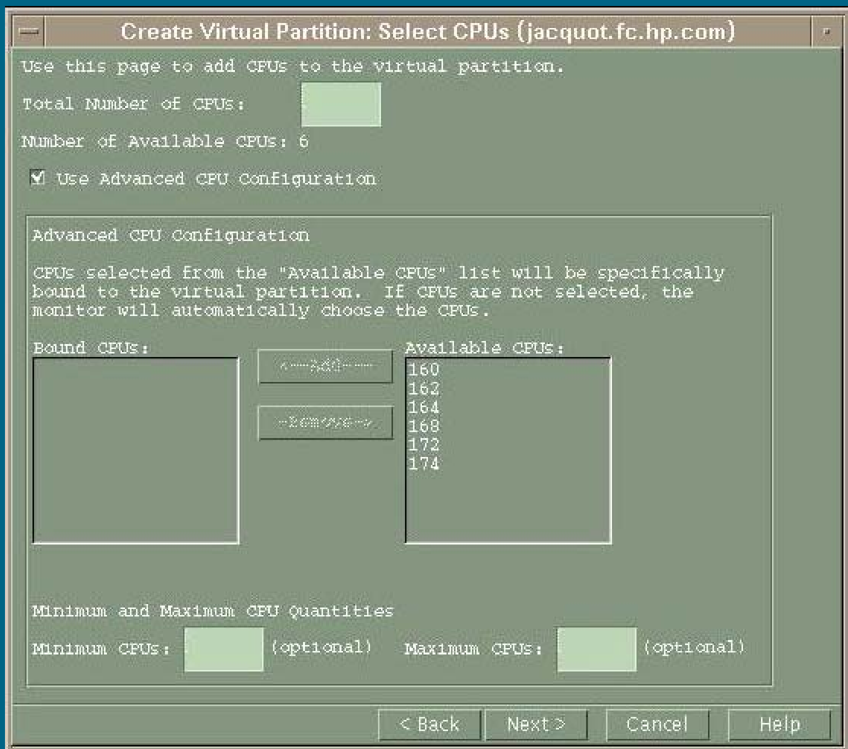
To delete a Virtual Partition in the currently running database:

```
# vparremove -p vPar_name
```

To display the status of a Virtual Partition in verbose mode:

```
# vparstatus -v -p vPar_name
```

virtual partition manager (vparmgr): GUI for managing virtual partitions

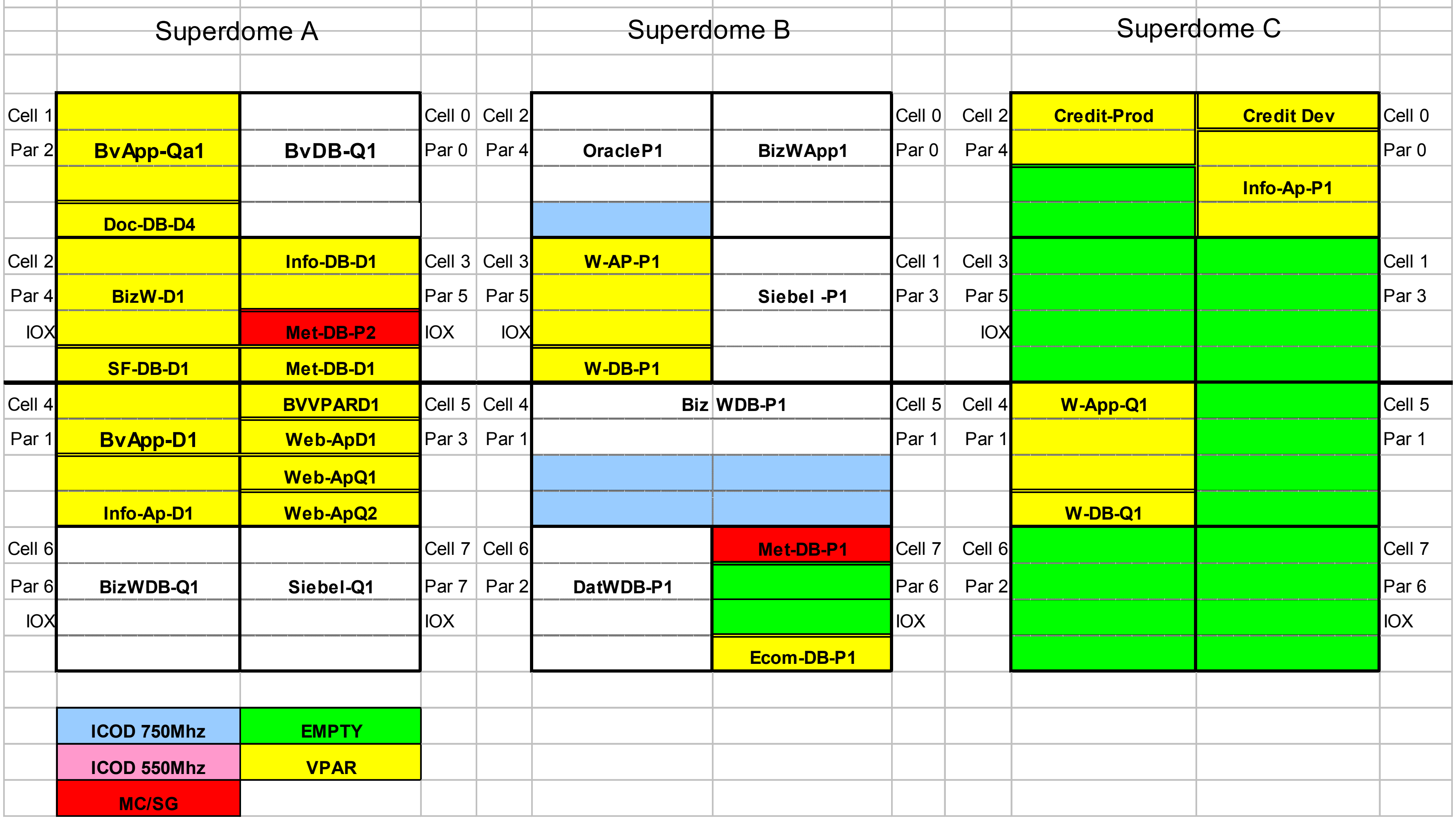


- Create, modify and delete virtual partitions (vpars)*
- Display assigned resources, attributes, and status of vpar
- Display vpar event log and samlog
- Boot and reset a vpar
- Direct invocation of task screens
- Preview create/modify vpar command lines prior to execution

✓ *parmgr is vPar aware!*

(it doesn't do vPars configuration at this point, but the 2 are planned to be integrated in the future)

Pitney Bowes SuperDome Architecture Present



PRM Commands

prmanalyze

prmavail

prmconfig

prmlist

Work at the command line or in xprm

prmloadconf

prmmonitor

prmmove

prmrecover

prmrun ps -efP

xprm – We’ll use this in our example but chapter uses command line.

/etc/rc.config.d/prm

/etc/prmconf time

glance

PRM Steps

Install PRM (component of MCOE)

↓

Set PRM startup variables in **(etc./rc.config.d/prm)**

↓

Run prmanalyze (requires accounting) or other method of analyzing system resources.

↓

Copy and modify **prmconf** to meet your resource management needs (**xprm** can be used to accomplish many tasks)

↓

Run **prmconfig –i -f** *prmconf.file* to initialize configuration file

(can also use **xprm**)

↓

Run **prmconfig –e** to enable PRM if required

↓

Check PRM groups with **ps -efP**

↓

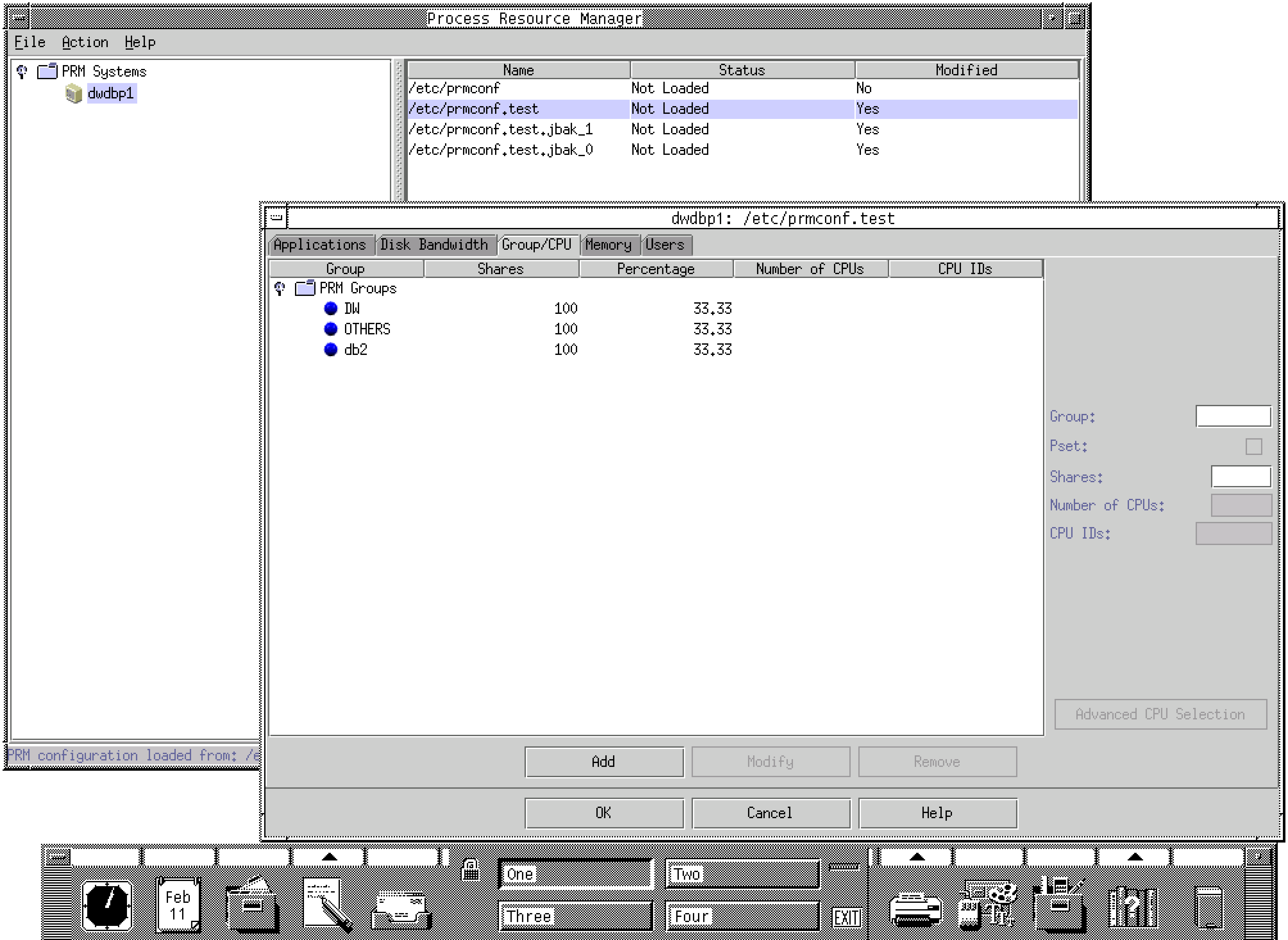
Use HP GlancePlus/UX to view PRM groups and system resources consumed

```
root@dwdbp1[/etc] > cat prmconf.test
  #
  # Group/CPU records
  #
  DW:2:100::
  OTHERS:1:100::
  db2:3:100::
  #
  # Memory records
  #
  # Application records
  #
  # Disk bandwidth records
  #
  # User records
  #
  db_hp::::db2
  adm::::OTHERS
  bin::::OTHERS
  daemon::::OTHERS
  hpdb::::OTHERS
  lp::::OTHERS
  nobody::::OTHERS
  nuucp::::OTHERS
  opc_op::::OTHERS
  smbnull::::OTHERS
  sys::::OTHERS
  uucp::::OTHERS
  webadmin::::OTHERS
  www::::OTHERS
  root@dwdbp1[/etc] >
```

Equal shares in prmconf Of 1/3 each

The PRM group *db2* has a PRMID of *3* and a share of *100* in this example. There is one user in the group db2 with a With a name *db_hp*.

xprm shows equal shares

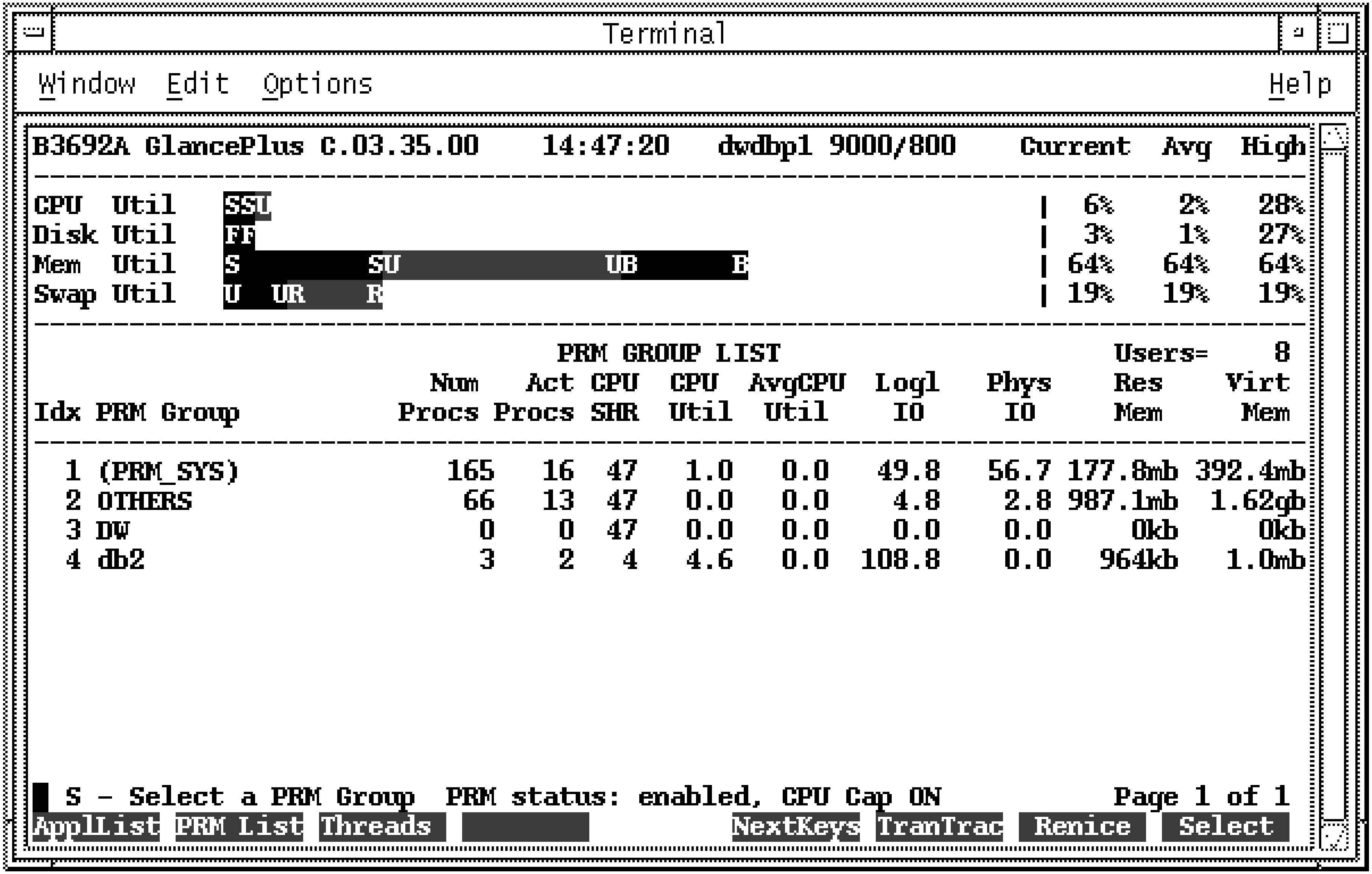


After editing the file and reducing the share of *db2* from *100* to *10,* we'll run the **prmconfig** command shown b

```
# prmconfig -i -f /etc/prmconf.test

PRM configured from file:  /etc/prmconf.test
File last modified:        Mon Feb 11 12:43:01 2002
PRM CPU scheduler state:  Enabled, CPU cap ON
PRM Group                      PRMID    CPU Entitlement
-------------------------------------------------------
DW                                 2             47.62%
OTHERS                             1             47.62%
db2                                3              4.76%

PRM memory manager state:  Not Running

PRM User                       Initial Group       Alternate Group(s)
--------------------------------------------------------------------
adm                            OTHERS
bin                            OTHERS
daemon                         OTHERS
db_hp                          db2
hpdb                           OTHERS
lp                             OTHERS
nobody                         OTHERS
nuucp                          OTHERS
opc_op                         OTHERS
root                           (PRM_SYS)
smbnull                        OTHERS
sys                            OTHERS
uucp                           OTHERS
webadmin                       OTHERS
www                            OTHERS
PRM application manager state:  Enabled  (polling interval: 30 seconds)
PRM application manager logging state:  Disabled
Disk manager state:  Disabled
#
```

Change db2 to 10 share from 100

Glance confirms the change from 100 share to 10 share
Notice that PRM_SYS share exists by default



Terminal

Window Edit Options Help

```
B3692A GlancePlus C.03.35.00    14:47:20   dwdbp1 9000/800    Current  Avg  High
--------------------------------------------------------------------------------
CPU  Util   SSU                                                |  6%     2%   28%
Disk Util   FF                                                 |  3%     1%   27%
Mem  Util   S        SU              UB       B                | 64%    64%   64%
Swap Util   U  UR    R                                         | 19%    19%   19%
--------------------------------------------------------------------------------
```

PRM GROUP LIST Users= 8

Idx	PRM Group	Num Procs	Act Procs	CPU SHR	CPU Util	AvgCPU Util	Logl IO	Phys IO	Res Mem	Virt Mem
1	(PRM_SYS)	165	16	47	1.0	0.0	49.8	56.7	177.8mb	392.4mb
2	OTHERS	66	13	47	0.0	0.0	4.8	2.8	987.1mb	1.62gb
3	DW	0	0	47	0.0	0.0	0.0	0.0	0kb	0kb
4	db2	3	2	4	4.6	0.0	108.8	0.0	964kb	1.0mb

S - Select a PRM Group PRM status: enabled, CPU Cap ON Page 1 of 1

ApplList PRM List Threads NextKeys TranTrac Renice Select

WLM Service Level Objectives

SLO's use goals, constraints, and conditions.

An SLO consists of:

- A workload (PRM group)
- Constraints (min, max cpu)
- A goal
- Priority
- Conditions (time of day, event, etc)

Group A

Min CPU: 20%

Max CPU: 50%

Group A receives 3 shares for each additional user.

Policy applies 9am to 5pm AND

when ServiceGuard Package XYZ

WLM goal types

- Any of the following can be used to allocate resources to a workload:
  - resource utilization
    - CPU entitlement based on utilization of current entitlement
  - direct measurement of the performance of the workload
    - response time
    - throughput
  - measurement of load on application
    - number of users/processes
    - queue length

Financial Services: Consolidating Oracle and WebLogic on Superdome using HP-UX WLM

Large Financial Services Company

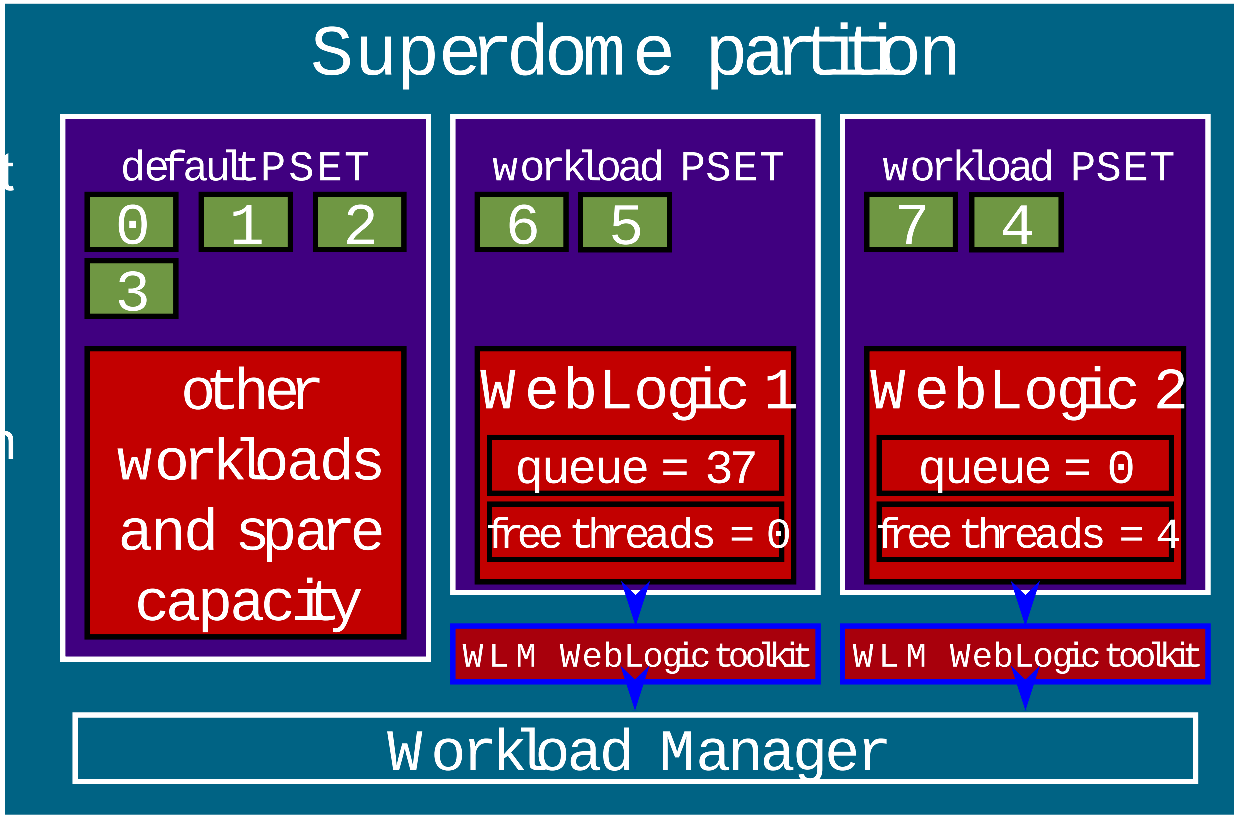
- 29,000 employees in 40 countries
- 82-year history
- Earned nearly $1.8 billion in 2001

The solution

- Processor Sets (pSets) provide the optimal performance and throughput for WebLogic-based applications
- Current queue length and the number of idle threads in the associated thread pool are used as performance metrics for Workload Manager
- Workload Manager will dynamically resize processor

The challenge

- incremental TCO benefits for HP Superdomes beyond hard partitioning
- improve system utilization

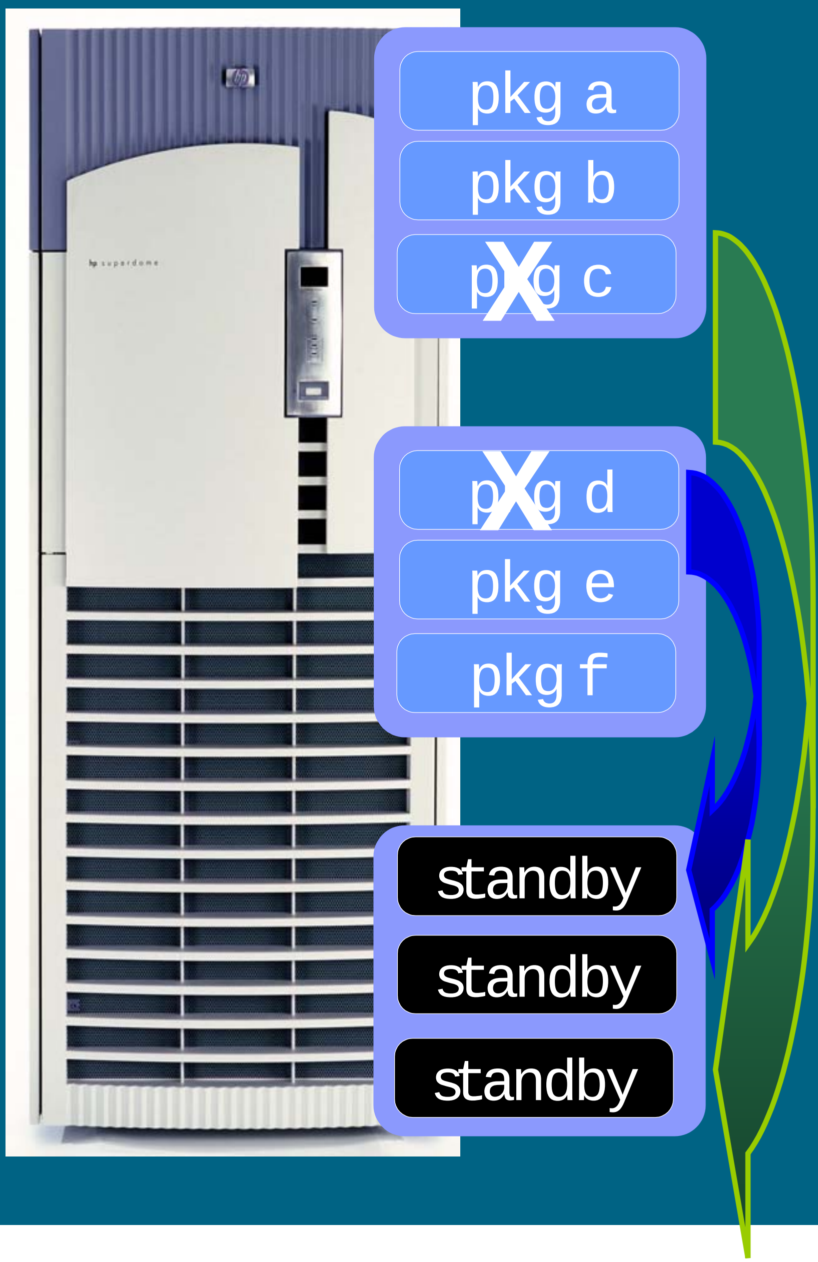


hp serviceguard with partitioning provides high availability for IT consolidation demands

Serviceguard supports hard-partitions (nPar) & soft-partitions (vPar) expanding application failover capabilities across chip-level, board-level, partition-level, system-level and data center level

benefits

- lower admin costs
- multiple use of single standby resource



On-Demand Resource Definitions

iCOD For thoses with strong growth plans Customer pays right-to-access fee, then enablement fee when CPU is activated and purchased.
Right to access (20%) + enablement (80%) = regular CPU purchase price at time of activation.

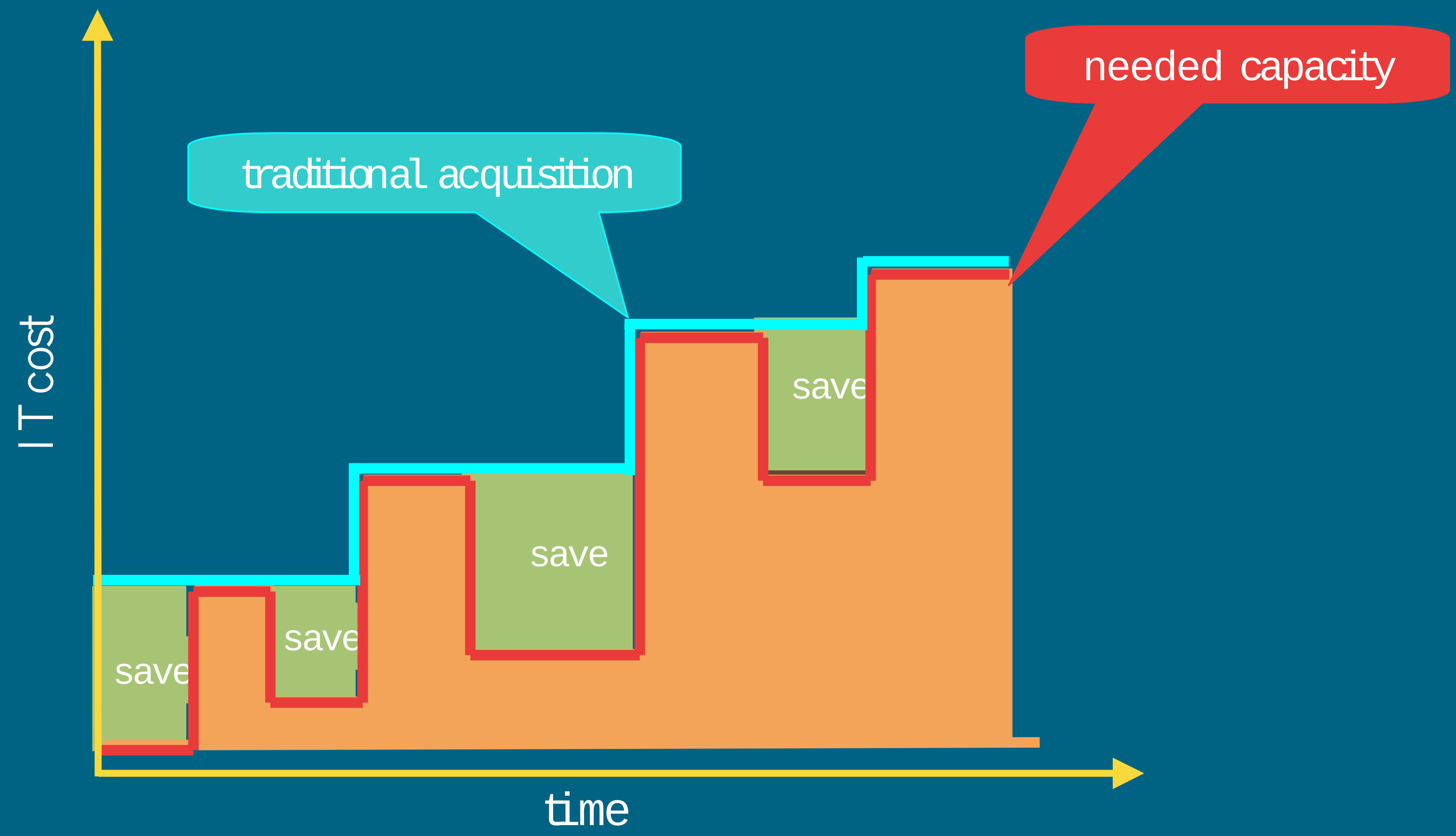
Temporary iCOD
Enables you to activate iCOD processors for 30 CPU days (measured in CPU minutes.) This program includes an HP-UX temporary operating environment right-to-use license and temporary hardware and software support.

Cellboard iCOD
Enables the you to add memory into the iCOD solution offering. Cell iCOD is a complete cell board that consists of base cell board, CPUs, and memory as one iCOD unit.

Pay-per-Use: Active CPU You control the number of active CPUs in your system PPU Metering software runs on server and measures active CPUs Billing based on monthly average of daily average of active CPUs Ideal for those who want control over their capacity or Those that have significant ISV software that is per-processor licensed.

Pay-per-Use: Percent Utilization Fully-automated, all CPUs are active.
Metering Appliance and software measures usage of each CPU 5 minutes and averages Billing based on monthly average of daily average % CPU utilization Ideal for those who use vPars or that want an automated solution with all CPUs active for better performance scaling.

Why TiCOD and PPU have a compelling value?



Industry average utilization

- •IDC, Meta, Gartner’s research shows industry average Unix server utilization is in the range of 30-40%
- •HP-UX survey shows most Unix customer have to support peaks of 2-3 x the average workload
- •HP Pay per use research on sample of 600+ mission critical customers had an average usage of 29% (+/-2%)