



POWER7[®] System RAS
**Key Aspects of Power Systems[™] Reliability,
Availability, and Serviceability**

November 1, 2010

IBM Server and Technology Group
Daniel Henderson, Jim Mitchell, and George Ahrens

Table of Contents

Introduction	5
<i>The Decade of Smart</i>	5
<i>Smarter Systems for a Smarter Planet™</i>	5
<i>Describing Server Reliability, Availability, Serviceability (RAS)</i>	6
A Definition of RAS	7
<i>Reliability</i>	7
<i>Availability</i>	8
<i>Serviceability</i>	8
POWER7 Reliability	9
Overview	9
Transient Faults	10
<i>Design and Test</i>	10
Design Defects	11
<i>Continuous Reliability Improvement</i>	12
POWER7 Reliability Design Improvements	13
Error Detection and Fault Isolation Design	14
Introduction	14
First Failure Data Capture (FFDC) with Dedicated Service Processors	15
Central Electronics Complex Availability	16
<i>Designing for Availability</i>	16
POWER7 Processor	17
A POWER7 Processor Core	17
<i>Core Recovery – Dynamic Processor Deallocation, Processor Instruction Retry, and Alternate Processor Recovery</i>	18
Predictive Processor Deallocation	19
Dynamic Processor Sparring	20
Partition Availability Priority	21
<i>Additional POWER7 Processor Core Recovery Design Attributes</i>	21
Level 1 Caches (Instruction and Data Cache Handling)	21
Level 2 and Level 3 Caches	22
<i>Special Uncorrectable Error Handling</i>	23
<i>CEC nodes and Processor to Processor Interfaces (Fabric Busses)</i>	25
Memory Subsystem	26
<i>Memory Bus Interfaces</i>	26
<i>Memory Buffer</i>	27
<i>Memory DIMMs and ECC Words</i>	27
<i>Scrubbing, Page and Logical Memory Block Deallocation</i>	29
<i>Active Memory Mirroring for Hypervisor</i>	30
Persistent Deallocation of Components and System IPL	31
Detecting and Deallocating Failing Components	31
Persistent Deallocation	31

I/O Subsystem Availability	32
Base Design	32
Later Developments	34
<i>Transitions from PCI to PCI-X and PCI-E and from RIO to RIO-G to InfiniBand.....</i>	34
<i>I/O Hub RAS and I/O Special Uncorrectable Error Handling.....</i>	34
12x DDR InfiniBand Connected I/O Hub Adapters and “Freeze” Mode.....	34
POWER7 I/O Enclosures and Integrated I/O	35
<i>Attached I/O Availability.....</i>	35
<i>Integrated I/O.....</i>	35
Reliability and Availability of the System Infrastructure	36
General System Environments.....	36
<i>Single Server/Operating System Environment.....</i>	36
<i>BladeCenter™ and Blades Environment.....</i>	37
Evaluating the Failure Rate of a “Redundant” and “Passive” Infrastructure	37
<i>Standalone Larger SMP Virtualized System Environment.....</i>	38
<i>Multiple Large Standalone Systems or a Physically Partitioned Single System.....</i>	39
Infrastructure Approach for POWER7 Servers.....	41
<i>PowerVM / POWER Hypervisor Design.....</i>	41
<i>Processor Resource Virtualization</i>	41
<i>Memory Resource Virtualization</i>	43
Assigning memory to virtual machines.....	43
Active Memory Expansion for the Power 795	43
Mirrored Memory Protects the POWER Hypervisor.....	43
Memory Deconfiguration and Sparing.....	43
<i>I/O Adapter Virtualization</i>	44
Dedicated I/O.....	44
Shared (Virtual) I/O.....	44
<i>Live Partition Mobility.....</i>	44
Other Infrastructure Reliability/Availability Design Elements	45
<i>Power/Cooling</i>	45
<i>TPMD.....</i>	45
<i>Clocks.....</i>	45
<i>Service Processors.....</i>	45
<i>Firmware Updates.....</i>	46
<i>Hardware Management Consoles.....</i>	46
Availability and Virtualization beyond the Hardware	47
Operating System and Application Features	47
<i>Storage Protection Keys.....</i>	47
<i>High Availability Solutions</i>	47
A Smarter Planet — Instrumented, Interconnected, and Intelligent	49
<i>Appendix A: System Support for Selected RAS Features [● = available, ○ = optional]</i>	51
<i>Appendix B: Operating System Support for Selected RAS Features [● = available, ○ = optional]</i>	53

Introduction

The Decade of Smart

In 2008, the IBM leadership team, recognizing the vast impact of technology on our society, began a dialog on the best ways to create a smarter planet. In a *smarter planet*, intelligence is infused into the systems and processes that make the world work. These technologies encompass both traditional computing infrastructures and the rapidly growing world of intelligent devices: smart phones, global positioning systems (GPS), cars, appliances, roadways, power grids, water systems.

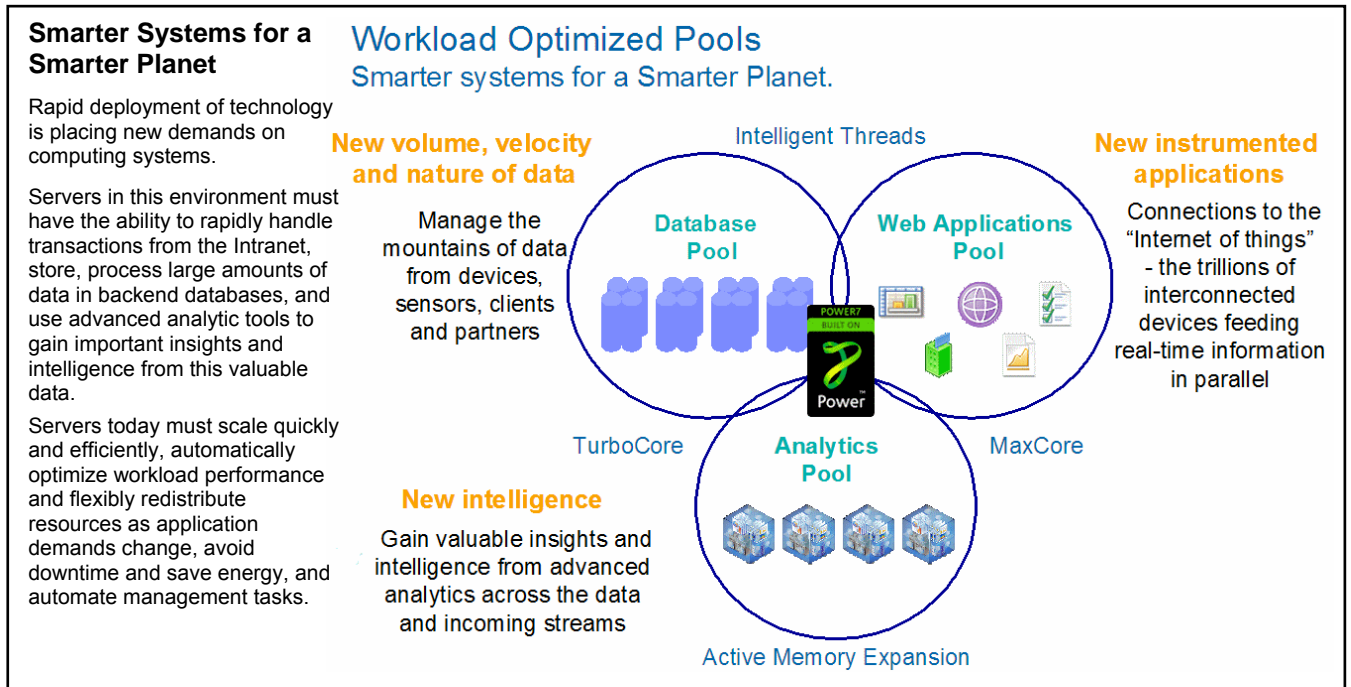
By 2011 we expect almost a trillion devices to be connected to the Internet. These Internet connected devices allow both new modes of social interactions and new ways for businesses to connect to their clients, their employees, and their suppliers. At the same time, these interactions produce enormous amounts of data — raw information about the way people use their resources and how data flows through the marketplace — that can be used to understand how societies operate.

Servers in this environment must have the ability to rapidly process transactions from the Internet, store, and process large amounts of data in backend databases, and support analytical tools allowing business and community leaders to gain the insight needed to make sound, timely decisions within their areas of responsibility.

Smarter Systems for a Smarter Planet™

In February 2010, IBM announced the first models in a new generation of Power Servers based on the POWER7™ microprocessor. While these new servers are designed to meet traditional computing demands of improved performance and capacity, the transition to “smarter” solutions requires that smarter servers also: scale quickly and efficiently, automatically optimize workload performance and flexibly manage resources as application demands change, avoid downtime and save energy, and automate management tasks.

Therefore, IBM has vastly increased the parallel processing capabilities of POWER7 systems -- integrated across hardware and software -- a key requirement for managing millions of concurrent transactions. As expected, the new Power Systems continue to enhance the proud heritage of IBM Power servers — delivering industry-leading transaction processing speed in designs built to efficiently



process the largest database workloads. In addition, these new offerings, optimized for running massive Internet workloads, deliver a leap forward in “throughput” computing.

Emerging business models will gather large amounts of data (from the Internet, sensors in electric grids, roads, or the supply chain for example). This data will be deposited in large databases and scrutinized using advanced analytical tools — glean information that can provide competitive advantage. Pools of high speed multi-threaded POWER7 processor-based servers can be deployed in optimized pools to efficiently process Internet workloads, store and process large amounts of data in databases, and employ specialized analytic tools (like the IBM Smart Analytics System 7700) to derive information useful to the business. These three computing modes — massive parallel processing, “throughput” computing, and analytics capabilities — are integrated and managed consistently with IBM Systems Director software.

Describing Server Reliability, Availability, Serviceability (RAS)

Since the early 1990’s, the Power Development team in Austin has aggressively pursued integrating industry-leading mainframe reliability technologies in Power servers. Arguably one of the most important capabilities, introduced first in 1997, is the inclusion of a hardware design methodology called First Failure Data Capture (FFDC) in all IBM Power System servers. This methodology uses hardware-based fault detectors to extensively instrument internal system components. Each detector is a diagnostic probe capable of reporting fault details to a dedicated service processor. FFDC, when coupled with automated firmware analysis, is used to quickly and accurately determine the root cause of a fault the first time it occurs, regardless of phase of system operation and without the need to run “recreate” diagnostics. The overriding imperative is to identify *which component* caused a fault — *on the first occurrence of the fault* — and to prevent any reoccurrence of the error. This feature has been described in detail in a series of RAS technical articles and “white papers” that provide technical detail of the IBM Power design.

The article “Fault-tolerant design of the IBM pSeries® 690 system using POWER4™ processor technology¹” emphasized how POWER systems were designed — from initial RAS concept to full deployment. In the nine years since the introduction of POWER4, IBM has introduced several successive generations of POWER processors, each containing new RAS features. Subsequent white papers^{2,3} included descriptions of how RAS attributes were defined and measured to ensure that RAS goals were in fact being met and detailing how each new feature contributed to reliable system operations. In general, these documents outlined the core principles guiding IBM engineering design that are reflected in the RAS architecture. A user should expect a server to provide physical safety, system integrity, and automated fault detection and identification.

1. Systems should be reliable in a broad sense: they should
 - produce accurate computational results
 - be free of physical hazards
 - be free of design flaws that impact availability or performance
 - have few hardware failures over their serviceable life
2. Systems should be configurable to achieve required levels of availability
 - Without compromising performance, utilization or virtualization
 - Utilizing Industry leading availability design and practices
 - Capable of taking advantage of clustering techniques to achieve the highest levels of availability
3. Systems should diagnose problems automatically and proactively

¹ D.C.Bossen A. Kitamorn, K.F. Reick, and M.S. Floyd, “Fault-tolerant design of the IBM pSeries 690 system using POWER4 processor technology”, IBM Journal of Research and Development, VOL.46 NO.1, January 2002.

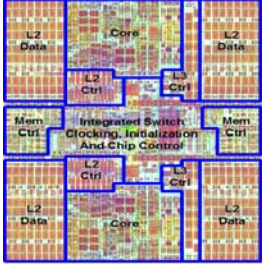
² J. Mitchell, D. Henderson, G. Ahrens, “IBM @server p5: A Highly Available Design for Business-Critical Applications”, p5tiPOWER5RASwp121504.doc, December 2004.

³ J. Mitchell, D. Henderson, G. Ahrens, and J. Villarreal, “Highly Available IBM Power Systems Servers for Business-Critical Applications”, PSW03003-USEN-00, April 2008.

- Taking advantage of self-healing and selective deconfiguration where possible to keep applications running
- Without relying on diagnostics to recreate problems, or manual analysis of dumps
- Without relying on assistance from the operating system or applications

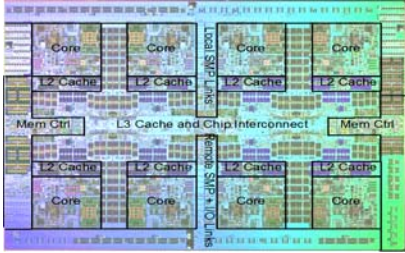
The intent of this white paper, then, is to highlight POWER7 server design features that extend the inherent Power Architecture® RAS capabilities delivered in previous server generations.

Quick Comparison: POWER6 and POWER7



POWER6 (2007)

- 65 nm technology – 341 mm²
- 0.79B transistors
- 2 Cores
 - 2 SMT threads/core
- 9 execution units/core
 - 2 integer and 2 binary floating-point units
 - 1 vector and 1 decimal floating-point unit
 - 2 load/store, 1 branch
- Integrated L2 cache
- L3 directory & controller (off chip L3 cache)
- 2 memory controllers



POWER7 (2010)

- 45nm technology – 567 mm²
- 1.2B transistors
- 8 Cores
 - 4 SMT threads/core
- 12 execution units/core
 - 2 integer and 4 binary floating-point units
 - 1 vector and 1 decimal floating-point unit
 - 2 load/store, 1 branch, 1 condition register
- Integrated L2 cache
- Integrated L3 cache
- 2 memory controllers

The POWER7 module is a much denser chip than its POWER6® predecessor, containing eight cores instead of two, and featuring 32 MB of integrated, not external, L3 cache. Compared to POWER6 it also includes substantially more function, with higher levels of simultaneous multi-threading (SMT) per core.

In addition to advances in density, virtualization, and performance the processor module also contains substantial new features related to reliability and availability, building on the rich heritage of prior generations.

Emphatically, a system's reliability, and the availability of the applications it supports, is a function of much more than just the reliability of the processors, or even of the entire system hardware. A full description of a system design for RAS must include all of the hardware, the firmware, the operating system, middleware, applications, operating environment, duty cycle, and so forth.

A Definition of RAS

As there are many ways to define reliability, availability, and serviceability, perhaps a good way to begin this discussion is to describe the how the IBM engineering team views RAS. A somewhat technically imprecise but useful definition for server hardware:

Reliability

- How infrequently a defect or fault is seen in a server.

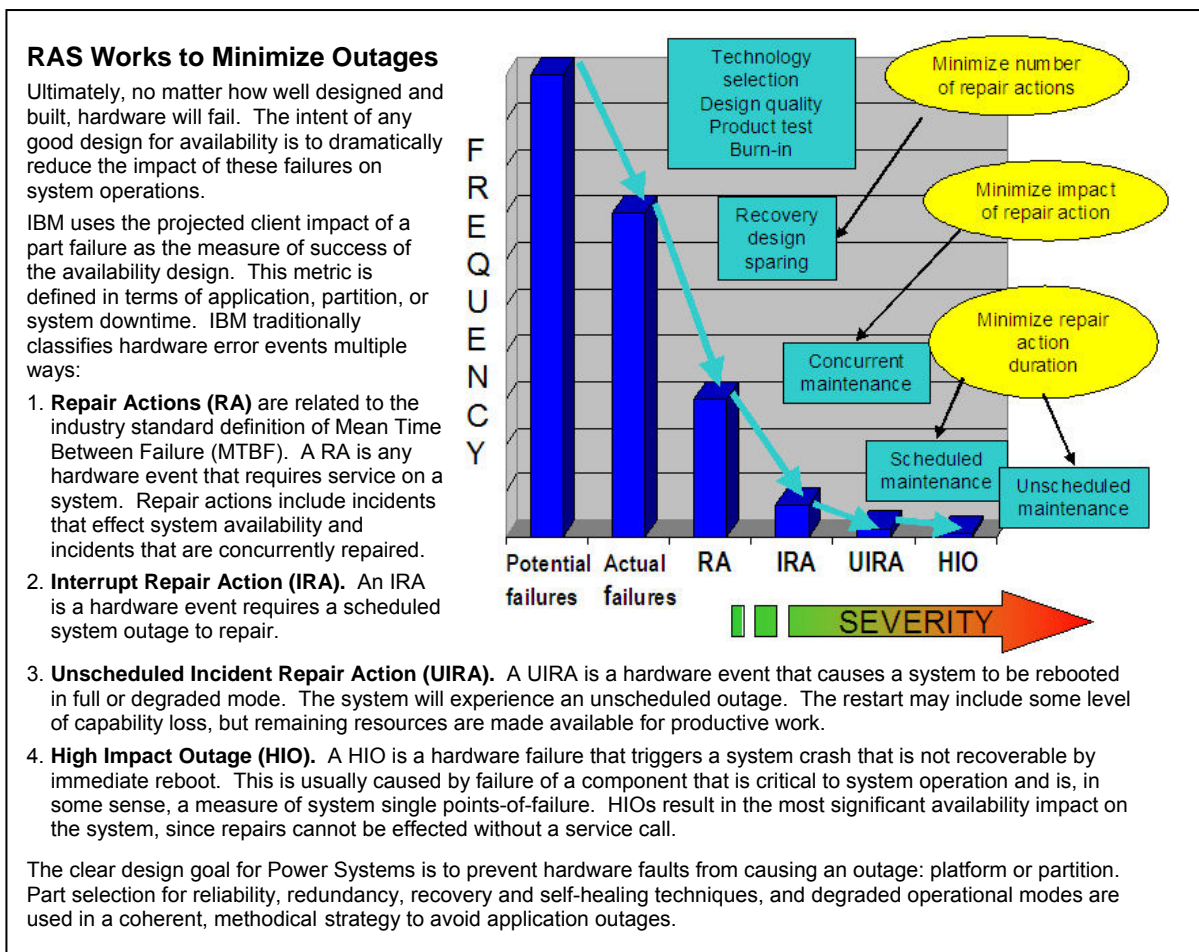
Availability

- How infrequently the functionality of a system or application is impacted by a fault or defect.

Serviceability

- How well faults and their impacts are communicated to end-users and servicers and how efficiently and non-disruptively they are repaired.

Defined this way, reliability in hardware is all about how often a hardware fault requires a system to be serviced – the less frequent the failures, the greater the reliability. Availability, then, is how infrequently such a failure impacts the operation of the system or application. For high levels of availability, correct system operation must not be adversely affected by hardware faults. In other words, a highly available system design insures that most hardware failures will not result in an application outage. Serviceability concerns itself with identifying what fails and ensuring an efficient repair (of that component, firmware, or software).



POWER7 Reliability

Overview

A variety of metrics can be used to describe the reliability or availability of a system or component. One of the most common is Mean Time between Failure (MTBF) which is traditionally thought of as how long, on average, something can be used before it stops working correctly.

When component description states that it has a MTBF of 50 years, the intent is not to claim it would be expected to run 50 years before seeing a failure. Rather, this statistical measurement indicates that in a large population of components, while the likelihood of any individual part failure is small, one could expect that 2% of the components would fail each year (for example, on a server with a published 50 year MTBF, if 100 servers were in operation for a year, an average of two server failures would be experienced).

This can be understood best by looking at the way hardware in a system is intrinsically expected to fail. Experience indicates that newly manufactured parts experience a relatively high failure rate. After an initial period of “early-life” fallout, there is an extended period of time where parts will fail at a fairly steady, but slow, rate. Then as components begin to wear-out there is a period of time where failures occur at an increasing rate.

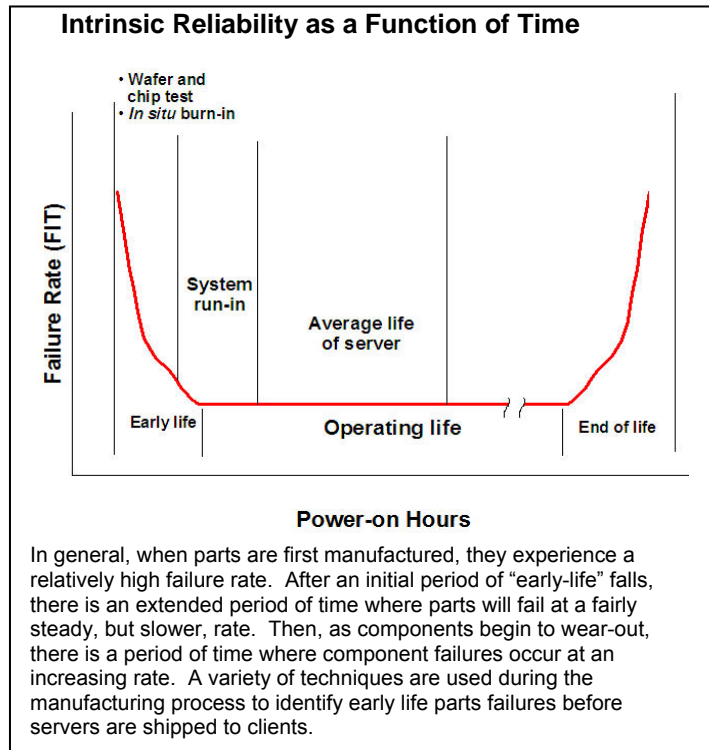
This can be illustrated by charting failures over time. A graph of typical failure rates is informally called a “bath-tub” curve — reflecting its unique shape.

It is the intent of the RAS design for POWER servers that systems be tested, screened, and if necessary “burned-in” such that when systems are delivered to clients for installation, components that can cause an outage are already “past” the early-life failure portion of the bath-tub curve.

In addition, the RAS design is intended to ensure that the wear-out portion for outage producing parts would be encountered after systems have been retired from production.

A number of different elements come in to play to make the latter expectation a reality. To mention a few:

- Components with a significant expectation of wearing out: fans, power components, etc. are made redundant and concurrently repairable or otherwise accounted for in the design such that their failures do not lead to planned or unplanned outages.
- The thermal environments of systems are regulated and monitored to ensure that components do not wear out quickly due to thermal stress.
- Spare connector pins are added to connectors that might show a propensity to fail after extended use (due to corrosion and other factors).



Transient Faults

Component failures (“hard” failures related to their intrinsic reliability) are not the only source of hardware outages. A variety of temporary or transient error may occur when the underlying hardware is not defective. These include faults caused by transient external upsets such as power line disturbances, static electrical discharge, energy released into circuits by cosmic radiation, and occasional circuit disturbance due to other sources of random noise.

Designing reliable systems require that the sources of these potential transient upsets be understood — so that hardware can be appropriately protected from them.

To this end, engineers use components and design techniques that inherently protect components and subsystems against transient errors: for example, using a processor latch design that is generally tolerant of the energy released by a cosmic ray strike.

Techniques are also employed to allow recover from problems as they occur. For example, by using a double error detect/single error correction (DEC/SEC ECC) code in memory; a temporary upset in a single bit can be automatically corrected.

Design and Test

Resilient system design techniques, component selection for reliability, thermal management, and controlling voltage regulation are some of the techniques used by the IBM system engineering team to meet the challenge of transient faults.

A Test to Verify Automatic Error Recovery⁴

To validate the effectiveness of the RAS techniques in the POWER6 processor, an IBM engineering team created a test scenario to “inject” random errors in the cores.

Using a proton beam generator, engineers irradiated a POWER6 chip with a proton beam, injecting high-energy protons into the chip at many orders of magnitude higher flux than would normally be seen by a system in a typical application. The team employed a methodical procedure to correlate an error coverage model with measured system response under test.

The test team concluded that the POWER6 microprocessor demonstrated dramatic improvements in soft-error recovery over previously published results.

Extending the testing to POWER7

This year, an IBM engineering team conducted a series of proton beam and “hot underfill” tests on the POWER7 microprocessor to validate the system design and the inherent recovery procedures.

POWER7 includes extensive soft error recovery techniques including Instruction Retry and Alternate Processor Recovery, providing “immunity” from errors that could cause outages on other processor designs.

In one test, engineers underfilled six POWER7 modules with radioactive material and ran them in Power 780 systems, amassing thousands of hours of highly accelerated runtime. This method speeds testing by dramatically increasing the likelihood of experiencing soft errors in the POWER7 chip. This test — equivalent to running for approximately a million years in a normal operating environment — generated almost 26,600 induced transient errors. Only 3 resulted in a UIRA — this is an equivalent soft error MTBF UIRA of greater than 330,000 years.

This type of system checking allow the engineering team to extensively identify issues with the recovery strategy and implementation, and validate checker coverage, checker priorities, and the latch and array designs.



POWER7 test system mounted in beam line.

Proton beam injection

- Irradiated a POWER7 module with proton beam
 - Injecting 720 billion high-energy protons
 - Many orders of magnitude higher flux than normal
- Validating detection/resilience and recovery mechanisms in the module

Hot under-fill testing

- Replaces a radiation-free under-fill material (Intended to shield against alpha-particles) with an irradiated under-fill intended to expose modules to alpha-particle energy
- Tests operating modules in systems over an extended time period
 - Validating the server’s soft errors correction facilities

⁴ Jeffrey W. Kellington, Ryan McBeth, Pia Sanda, and Ronald N. Kalla, “IBM POWER6 Processor Soft Error Tolerance Analysis Using Proton Irradiation”, SELSE III (2007).

However, simply designing for reliability is not sufficient to ensure robust protection against transient faults. IBM has long recognized the need to test systems to validate their resistance to sources of soft error events. Compliance testing of systems at IBM includes subjecting systems to power line disturbances of various different intensities and durations, including those that might be created by nearby lightning strikes, systematically exposing systems to electrical static discharge, doing thermal testing and profiling, and so forth.

In more recent years IBM has also expended considerable effort in exposing electronics to large doses of radiation (many orders of magnitude more intense than seen in the field) to ensure that the designs provide the intended level of protection from cosmic and alpha particle emissions.

Design Defects

Sometimes, a system design flaw (a “bug” in the hardware or firmware) is manifested as an unplanned outage. This is most typically the case for those infrequent design defects that occur only under a very specific set of operating conditions. These rarely seen situations can be very difficult to identify and repair. In other cases, defects in the methods used to handle error detection or recovery can themselves be defective, resulting in cascading errors and leading to system outages.

Predicting system outages (determining the system reliability) caused by intrinsic failures is complex but relatively straightforward. Reliability calculations can also account for the predicted impact of transient errors. It is very difficult, however, to include the impact of design defects in these computations.

Rather than projecting the impact of design defects, a good reliability design strives to eliminate them. For this reason processors and other custom modules designed by IBM are extensively simulated before parts are fabricated. Simulation is intended to validate both the function of the devices, and the system’s ability to detect and handle errors as they occur. Component level simulation and testing is the starting point in a multi-tiered system verification strategy performed before a product is released. Engineers create full system computational environments in which entire stacks — hardware, firmware, operating system, and applications — are functionally tested in the ways users are expected to deploy systems.

IBM has invested heavily in a hardware design that supports error detection and correction during normal operations. This approach also incorporates the ability to inject errors as needed during the design bring-up and validation phase to efficiently and thoroughly confirm a system’s functions and its’ error handling capabilities.

Power Systems use a FFDC methodology, employing an extensive array of error checkers and Fault Isolation Registers (FIR) to detect, isolate, and identify faulty conditions in a server. This type of automated error capture and identification is especially useful in allowing quick recovery from unscheduled hardware outages. While this data provides a basis for component failure analysis, it can also be used to both improve the reliability of the part and as a starting point for design improvements in future systems.

IBM RAS engineers use specially designed logic circuitry to create faults that can be detected and stored in FIR bits, simulating internal chip failures. This technique, called error injection, is used to validate server RAS features and diagnostic functions in a variety of operating conditions (power-on, boot, and operational run-time phases). Error injection is used to confirm both execution of appropriate analysis routines and correct operation of fault isolation procedures that report to upstream applications (the POWER Hypervisor™, operating system, and Service Focal Point and Service Agent applications). Further, this test method verifies that recovery algorithms are activated and system recovery actions take place. Error reporting paths for client notification, pager calls, and call home to IBM for service are validated and RAS engineers substantiate that correct error and extended error information is recorded. A test servicer, using the maintenance package, then “walks through” repair scenarios associated with system errors, helping to ensure that all the pieces of the maintenance package work together and that the system can be restored to full functional capacity. In this manner, RAS features and functions, including the maintenance package, are verified for operation to design specifications.

Continuous Reliability Improvement

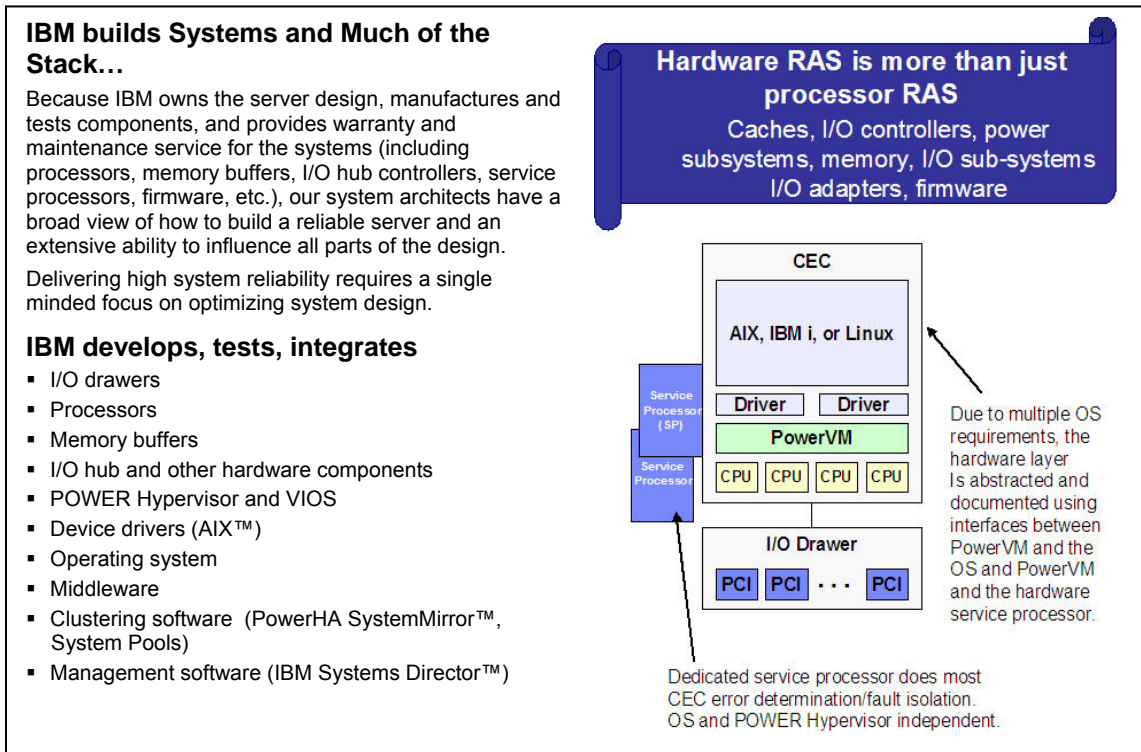
Of course, setting failure rate reliability targets for components, designing systems to meet specific availability metrics, thoroughly validating and testing the reliability design, and projecting the expected resulting performance all help create a reliable server design. However, simply setting targets and simulating the design is not sufficient.

It is also vital for long term system reliability that the performance of deployed systems be monitored, deficiencies identified, and corrective actions taken to ensure continued reliability of systems over time.

IBM field engineering teams track and record repairs of system components covered under warranty or maintenance agreement. Failure rate information is gathered and analyzed for each part by IBM commodity managers, who track replacement rates. Should a component not be achieving its reliability targets, the commodity manager will create an action plan and take appropriate corrective measures to remedy the situation.

Aided by IBM's FFDC methodology and the associated error reporting strategy, commodity managers build an accurate profile of the types of field failures that occur and initiate programs to enable corrective actions. In many cases, these corrections can be initiated without waiting for parts to be returned for failure analysis.

The IBM field support team continually analyzes critical system faults, testing to determine if system firmware, maintenance procedures, and tools are effectively handling and recording faults. This continuous field monitoring and improvement structure allows IBM engineers to ascertain with some degree of certainty how systems are performing in client environments rather than just depending upon projections. If needed, IBM engineers use this information to undertake "in-flight" corrections, improving current products being deployed. This valuable field data is also used to plan and design future servers.



Because IBM owns the server design, manufactures and tests components, and provides warranty and maintenance service for the systems (including processors, memory buffers, I/O hub controllers, service processors, firmware, etc.), IBM system architects have a broad view of how to build a reliable server

and an extensive ability to influence all parts of the system design. When designing a specific component there is a natural tendency to want to build the “best” or the “fastest.” The risk is that while a component may be optimized by some criteria, it may not operate well within a system’s architecture. This type of design process is referred to as sub-optimization — components work well by themselves but the overall server performance or reliability suffers. This type of server design can result when a vendor receives components from a variety of sources and tries to “glue” systems together.

A primary objective of the IBM Power design team is to deliver “smarter” systems that provide balanced performance (coupling processor speed, with memory capacity and I/O bandwidth) and that integrate exceptionally well with systems software (operating system, middleware, applications).

IBM is uniquely positioned in the computing industry in that IBM designs and builds, (or IBM engineers specify the design and monitor and verify the manufacture), integrates, and tests system key components — transforming raw silicon to servers, integrating OS, middleware, and management software. This control extends to IBM engineers a very efficient means to optimize solutions — whether a change is needed in the hardware or firmware design, or in how it is manufactured, tested, or serviced.

POWER7 Reliability Design Improvements

From a RAS standpoint, the system design process begins with the definition of a design for reliability. An IBM corporate directive dictates that new products must deliver equal or better reliability than their predecessors. This outcome is expected even though each new server generation typically delivers improved performance and higher capacity — often at the “cost” of more components that can fail (more cores, memory, I/O channels).

The transition from POWER6 to POWER7 presented an interesting challenge for the RAS team. On one hand, the dramatic increase in number of cores per chip allows IBM to deliver servers with the same number of cores but with many fewer processor chips (fewer potential points of failure). In addition, the POWER7 incorporates an integrated L3 cache — reducing the number of chips required in larger server designs. On the other hand, IBM is also delivering servers with many more cores (up to 256 in the Power 795) — along with dramatic increases in the memory and I/O required to keep those cores working at optimal levels of performance.

As an additional challenge to RAS engineers, stepping down the processor feature size to 45nm technology brought a potential increase in exposure to transient errors. IBM introduced both Instruction Retry and Alternate Processor Recovery in POWER6 processor-based servers to help avoid these types of soft errors. POWER7 continues this technology and extends its overall resistance to these types of events by using a “stacked latch” design which is inherently less likely to change state when an energy discharge from a cosmic ray or alpha particle is encountered.

Continuing to focus on avoiding soft errors, the memory bus on POWER7 processor-based servers features a differential design with the ability to tolerate soft errors by detecting faults with CRC checking and retrying faulty operations. The bus maintains the ability introduced in POWER6 to replace a faulty data bit line on the bus with a spare bit line dynamically.

Significantly, given the expectations for dramatically increasing the overall amount of memory available in the system, the amount of spare memory capacity on custom DIMMs for larger servers has also increased in POWER7. Using spare DRAM chips, selected high end POWER7 servers provide a “self-healing capability” in memory, automatically moving data from failed DRAM chips to available spares. This design permits a pair of DIMMS, on a rank basis, to experience one and sometimes two DRAM failures without replacing the DIMMs and still maintain Chipkill detection and correction capabilities (potentially surviving three DRAM chip failures for every rank pair on a DIMM pair). For details, refer to Memory DIMMS and ECC Words on page 27.

Error Detection and Fault Isolation Design

Introduction

In a broad sense, for systems to be reliable, faults which could impact computations must be detected and corrected so that incorrect results are not propagated to client applications. Systems designed to accomplish this task require an extensive architecture for error detection, recovery, and fault isolation.

Such an infrastructure must include the ability to detect and, where possible, correct errors in data wherever it is stored. Maintaining at least double bit detection and single bit correction ECC codes to verify stored data can help meet that objective.

However, while the sheer number of circuits devoted to such ECC correction codes can be large, error detection should include much more than just ECC data checking. To be effective, it should also include:

- The ability to detect cases where data is stored in, or retrieved from, an incorrect location, or where there was no response to a request to store or retrieve data.
- Mechanisms to validate that the logic function of various processing components and those devoted to data movement were also free of flaws. For example:
 - Error detection and correction of data in latches and arrays used to store logic state.
 - Processes that flag invalid state machine transitions caused by component failures and transient conditions
 - Mathematical mechanisms for statistically validating the results produced by computation units (e.g., using techniques like residue checking).

To ensure efficient servicing of systems having uncorrectable errors, a complete design methodology must not only detect an error, it must also include enough data to determine the source of the error — to correctly identify a specific component (or components). These parts may need to be deconfigured to prevent a re-occurrence of a problem, and ultimately replaced as required.

This architecture mandates that error checkers be distributed throughout the system to allow identification of the root cause of a failure. For instance, it is not sufficient to simply check the validity of data (via a mechanism such as ECC) when it is loaded into a processor data cache or written to memory. The data must be validated at every point where it crosses a component boundary (everywhere it could be corrupted).

For the highest levels of error detection and recovery, a server must both identify a fault when it occurs *and* incorporate a mechanism to automatically handle the failure *at that time*.

For example, such a design might mandate that a floating point execution unit complete the residue checking of an executing floating point instruction before the next instruction begins, so that a failed instruction can be retried immediately should an error occur.

Finally, error detection and fault isolation should be:

- Accomplished automatically in the hardware and firmware without the need for errors to be recreated by a diagnostic program and without the need for any sort of human interpretation of error indicators.
- Completed in hardware/firmware *independent of any operating system*, and ideally also independent of system processing resources (so that the server hardware can efficiently support multiple operating systems and avoid reducing the computational resources available to the system).

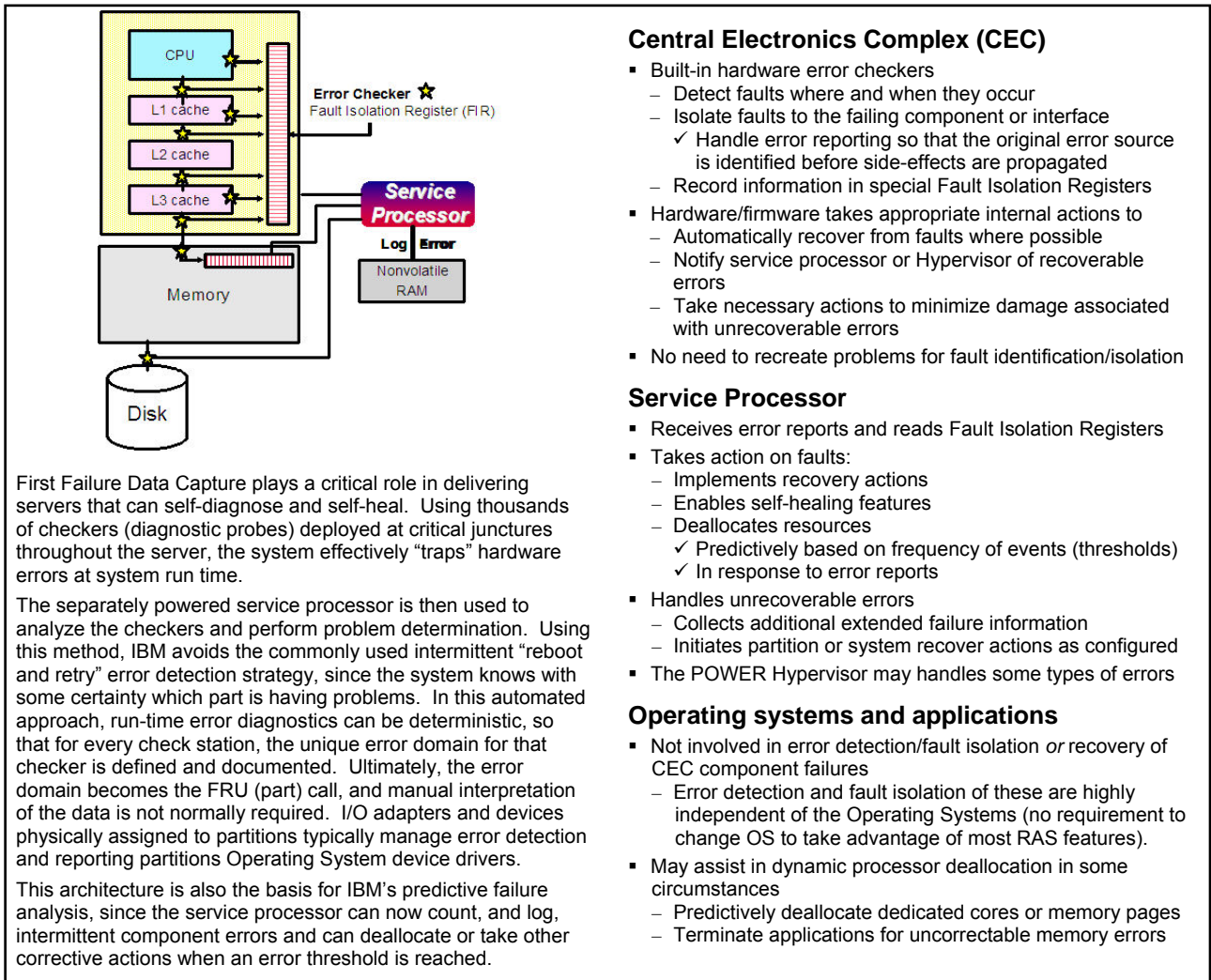
First Failure Data Capture (FFDC) with Dedicated Service Processors

Components in the central electronics complex (processors, memory buffers, and I/O hub controllers) use hardware-based fault detectors to extensively instrument internal system components.

Called First Failure Data Capture, this architecture requires that a server “build-in” hardware error check stations that capture and help to identify error conditions. A 256-core Power 795 server, for example, includes over 598,000 checkers to help capture and identify error conditions. These are stored in over 96,000 Fault Isolation Register bits. Each detector is a diagnostic probe capable of reporting fault details to a dedicated service processor. Pertinent error data related to the fault is captured and collected in fault isolation registers and processed with specialized “who’s on first” logic for analysis.

With this technique, if a fault is detected in a server, the root cause of the fault will typically be captured without the need to recreate the problem or run any sort of extended tracing or diagnostics program. For the vast majority of faults, a good FFDC design means that the root cause can also be detected automatically without service intervention.

The service processor is a dedicated microprocessor, independent of the POWER7 microprocessors, used to correlate fault information and to orchestrate error recovery mechanisms in conjunction with the hardware and POWER Hypervisor (firmware). The service processor is signaled whenever the status of the fault isolation registers change and can access the register information at any time without impacting the performance of these modules.



The RAS architecture, then, couples FFDC with *automated* firmware analysis, to quickly and accurately determine the root cause of a fault the first time it occurs, regardless of phase of system operation and without the need to run “recreate” diagnostics. The overriding imperative is to identify *which component* caused a fault — *on the first occurrence of the fault* — and to prevent any reoccurrence of the error.

IBM engineers control the entire CEC design (from silicon to architecture) and the RAS architects correlate error events over the entire system. To help insure accurate error coverage and identification, Power Servers include an *additional* error infrastructure to help identify the first cause of a fault even if it propagates to multiple components (e.g. data from memory having uncorrectable errors).

In addition to accessing fault isolation registers, when a severe error occurs, the service processor also has the ability to scan the state of functional latches in modules, to create a snapshot of the state of the processors and other components at the time of a failure. This additional information, referred to as extended error data, can be transmitted to IBM when a failure occurs, using automated call-home functions (when enabled by the system administrator), so that IBM servicers can ascertain the circumstances surrounding a fault event. This process enables detailed failure analysis, helping to ensure that systematic issues can be quickly identified and dealt with as part of a continuous quality improvement process [See [Continuous Reliability Improvement](#) on page **Error! Bookmark not defined.**].

Central Electronics Complex Availability

Designing for Availability

The IBM Power availability strategy is deeply rooted in an extensive history spanning decades of mainframe development. By leveraging an extensive background in Predictive Failure Analysis™ and dynamic system adaptation, the IBM availability team has helped to create a unique processor that unleashes significant value to the client.

Together these attributes form a cornerstone of the POWER7 availability strategy as delivered in the, Power Server family. These capabilities enable an unprecedented level of self-awareness in POWER7 processor-based systems. The FFDC method allows the system to proactively analyze situations that indicate an impending failure, in many cases removing the questionable component from use before it can cause a threat to system stability.

To qualitatively understand the availability design of Power Systems it's necessary to identify the availability characteristics of each component of the system. This is true of hardware – processors, memory, I/O adapters, power and cooling subsystems, chassis and firmware as it relates to the function of the hardware — and of each software component used on the server. Thus, even if a RAS discussion is limited to hardware elements, it must be investigated in the context of the entire system — and not just focus on one element, such as a processor core, or memory component.

In the sections that follow, the discussion of RAS capabilities will specifically spotlight the capabilities delivered in the high-end enterprise class systems. Server design methodologies commonly focus first on the very stringent availability requirements of large, highly virtualized, high performance systems like the Power 795. RAS features developed for this class of systems will then frequently “flow down” through the product line, exploiting investments in processor or firmware recovery design, and delivering “high end” availability attributes to “entry” class servers. A RAS feature matrix by model is provided in [Appendix A: System Support for Selected RAS Features](#) [page 51].

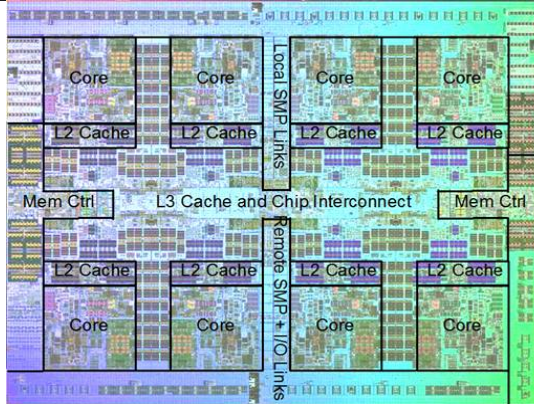
POWER7 Processor

POWER7 Processor Chip

The POWER7 chip features eight cores each capable of single- and simultaneous multithreading execution (including the ability to execute four threads on the same core at the same time). POWER7 maintains binary compatibility with existing POWER6 processor-based systems to ensure that binaries continue executing properly on the newer systems. Supporting virtualization technologies like its POWER6 predecessor, the POWER7 technology has improved availability and serviceability at both chip and system levels. To support the data bandwidth needs of an eight-core processor running

at 4 GHz (or in TurboCore mode at 4.25 GHz), the POWER7 chip uses embedded dynamic RAM (eDRAM) technology to include a 32-MB intelligent L3 cache on the processor chip. This cache dramatically reduces the latency of data access when compared to an external cache implementation, while taking roughly 1/3 the space and 1/5 the standby power while delivering significant improvements in resistance to soft errors when compared to conventional static RAM (SRAM).

Based on a superscalar design, each 4W SMT core in the POWER7 microprocessor includes twelve (12) instruction execution units. Delivering recovery hardware for processor Instruction Retry for automatic restart of workloads on the same, or alternate, core in the same server, the POWER7 chip also uses a “stacked latch” design for improved soft error immunity and includes a wide variety of features for energy management including “sleep” and “nap” modes and dynamic frequency and voltage management.



The POWER7 processor chip consists of eight processor cores, each with a 32-KB 8-way set associative data cache, a 32-KB 2-way set associative instruction cache, and a tightly coupled 256-KB 8-way set associative level 2 cache.

The chip also uses embedded dynamic RAM technology (eDRAM) to deliver 32MB of *internal* L3 Cache. This onboard L3 cache is very important to delivering a balanced system design as it

provides:

- Up to a 6 to 1 latency improvement for L3 accesses vs. external L3 since there are no off-chip drivers or receivers in the L3 access path.
- A 2 times bandwidth improvement with on-chip interconnect.
- A space reduction over conventional SRAM.
 - ✓ 1/3 the space of conventional SRAM implementation.
 - ✓ 1/5 the standby power.
- An innovative, intelligent cache design, with a 4-MB fast L3 region per processor core where the system:
 - ✓ Will automatically migrate private footprints (up to 4-MB) to this fast local region (per core) at ~5X lower latency than full L3 cache.
 - ✓ Can automatically clone shared data to multiple private regions.

POWER7 also includes two memory controllers (each with 4 high speed memory channels), and a series of bus controllers that manage interfaces to other processor modules, to I/O hub controllers, and to a service processor.

A POWER7 Processor Core

Each 64-bit processor core features “out-of-order” instruction execution, and includes the ability to dispatch 6 instructions per cycle and issue 8 instructions per cycle. Each core supports 12 instruction execution units, including 2 for fixed point (integer) processing (FXU), 2 load/store (LSU), 4 double precision floating point processors (FPU), 1 vector processor (VMX), 1 branch (BRU), 1 condition register (CRU), and 1 decimal floating point unit (DFU).

A POWER7 Processor Core

Each 64-bit processor core features “out-of-order” instruction execution, and includes the ability to dispatch 6 instructions per cycle and issue 8 instructions per cycle. Each core supports 12 instruction execution units, including 2 for fixed point (integer) processing, 2 load/store, 4 double precision floating point processors, 1 vector processor, 1 branch, 1 condition register, and 1 decimal floating point unit.

Each variable frequency core (up to 4.25 GHz) can process instruction streams in single threaded, or 2- or 4-way simultaneous multi-threaded (SMT) mode. SMT modes may be dynamically set per virtual machine or changed by the operating system at run-time for optimal performance.

To achieve the highest levels of server availability and integrity, FFDC and recovery safeguards must protect the validity of user data anywhere in the server, including all the internal storage areas and the buses used to transport data. It is equally important to authenticate the correct operation of internal latches (registers), arrays, and logic within a processor core that comprise the system execution elements and to take appropriate action when a fault (“error”) is discovered.

The POWER7 microprocessor includes circuitry (FFDC) inside the processor core to spot and correct these types of errors. A wide variety of techniques is employed, including built-in precise error check logic to identify faults within controller logic and detect undesirable conditions within the server. In addition, Power Servers can use Predictive Failure Analysis techniques to vary off (dynamically deallocate) selected components before a fault occurs that could cause an outage (application, partition, or server).

Selected Core RAS Features

- Instruction and data cache retry and set delete
- Processor Instruction Retry
- Alternate Processor Recovery
- Hang recovery
- GPR ECC with recovery
- Core contained checkstop

Core Recovery – Dynamic Processor Deallocation, Processor Instruction Retry, and Alternate Processor Recovery

The POWER6 processor-based servers introduced a core design that included extensive facilities to detect and isolate a wide variety of errors during instruction processing. Maintaining an accurate copy of CPU state, the core could retry a failing instruction, avoiding many soft failures.

These systems introduced a number of features enabling dynamic adjustments when issues arise to threaten availability. Most notably, they implement the Processor Instruction Retry suite of tools, which includes Processor Instruction Retry, Alternate Processor Recovery, Partition Availability Prioritization, and Single Processor Checkstop. Taken together, in many failure scenarios these features allow a Power Server (POWER6 or POWER7) to recover transparently without an impact on a partition using the core. This feature set is described in detail in a POWER6 RAS white paper.⁵

Processor Instruction Retry (retrying a failed operation caused by a transient error), is a primary means by which a POWER6 or POWER7 system recovers from “soft” faults in the processor core. These processor chips also include soft error fault handling mechanisms that operate in cooperation with the POWER Hypervisor. These transient error recovery techniques are accomplished without operating system involvement.

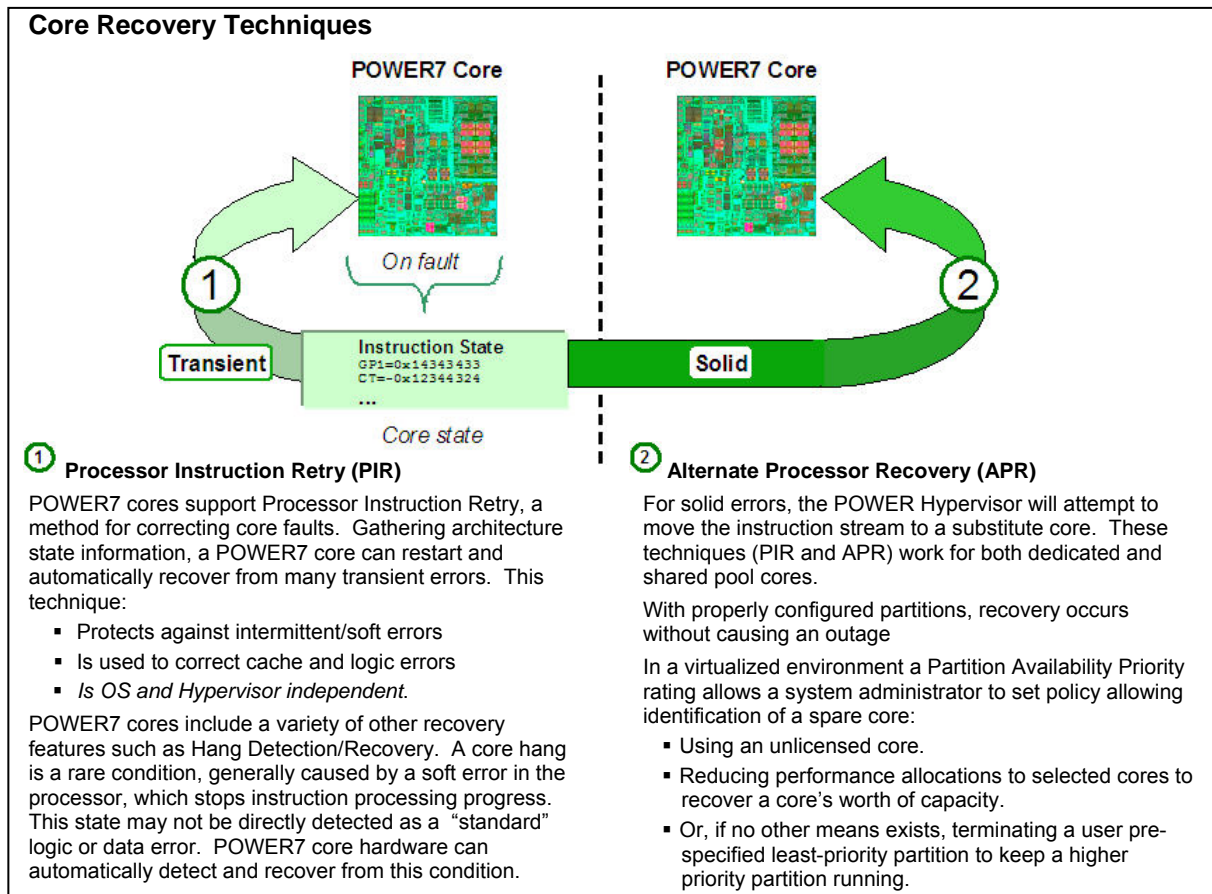
However, while protecting against transient or soft errors is an important availability concern, it is not sufficient for a complete RAS design. Engineers must also take into account the “solid” failure rate of components. For this reason, POWER6 incorporated a mechanism that created a checkpoint describing the core operational state before an instruction was executed. Then, in many cases, the POWER Hypervisor could load the state information from a solidly failing processor core into an alternate

⁵ Jim Mitchell, Daniel Henderson, George Ahrens, and Julissa Villarreal, “IBM Power Platform Reliability, Availability, and Serviceability (RAS). Highly Available IBM Power Systems Servers for Business-Critical Applications”, POW03003-USEN-03 (June 5 2009).

processor core — dynamically moving thread execution without an application outage. This technique is called *Alternate Processor Recovery*. Alternate processor recovery allowed systems to recover from a number of solid faults in the processor that would otherwise have caused outages of either partitions or entire systems. This feature is now supported in POWER7 servers.

A key aspect of this design is that all of this capability for recovery, for both hard and soft failures, is contained in the hardware and POWER Hypervisor. Recovery is expected to be transparent to applications and independent of the operating systems deployed in partitions.

If a processor-detected fault cannot be recovered by Processor Instruction Retry or Alternate Processor Recovery, then the POWER Hypervisor will terminate (checkstop) the partition using the processor core at the time of the fault. This is known as *core contained checkstop*. In general, this limits the outage to the single partition. However, if a fault can not be contained to a single instruction, or if a failure occurred on a POWER Hypervisor instruction, the server will be rebooted.



Predictive Processor Deallocation

Prior to the introduction of the advanced core recovery features in POWER6, Predictive Processor Deallocation was the primary means for dealing with core errors. This feature considers patterns of correctable errors in processor function as predictive of a future uncorrectable processor error.

Dynamic Processor Deallocation enables automatic deconfiguration of an error-prone processor core before it causes an unrecoverable system error (unscheduled server outage). Dynamic Processor Deallocation relies upon the service processor’s ability to use FFDC-generated recoverable error information and to notify the POWER Hypervisor when the processor core reaches its predefined error limit. The POWER Hypervisor, in conjunction with the operating system (OS), will then “drain” the run-

queue for the offending core, redistribute the work to the remaining CPUs, deallocate the offending CPU, and continue normal operation, although potentially at a lower level of system performance

This feature could be coupled with Dynamic Processor Sparing to automatically and dynamically substitute available spare processing capacity prior to deallocation of the failing core.

While this technique is still used in POWER7 servers, it has some significant drawbacks when compared to Processor Instruction Retry and Alternate Processor Recovery:

- When processors are dedicated to partitions, cooperation in the deallocation by the operating system and or applications is required to ensure that the thread currently running on the processor is completed before the deallocation takes place.
- This mechanism recovers from predictable, recoverable errors. Unpredictable intermittent or solid errors can not be handled.

Dynamic Processor Sparing

IBM's Power virtualization environment built on PowerVM™ offers a secure virtualization environment that can help eliminate underutilized servers by pooling resources and optimizing their use across multiple application environments and operating systems. Through advanced dynamic logical partitioning (LPAR) capabilities, a single partition can act as a completely separate AIX®, i, or Linux™ operating environment. Partitions can have dedicated or shared processor resources. With shared resources, PowerVM can automatically adjust pooled processor resources across multiple operating systems, borrowing processing power from idle partitions to handle high transaction volumes in other partitions.

This very powerful approach to partitioning maximizes partitioning flexibility and maintenance. In addition to enabling fine-grained resource allocation, these PowerVM capabilities provides all the servers in the Power Systems family the capability to individually assign or reassign any resource (processor core, memory segment, I/O slot) to any partition in any combination — or to dynamically share resources among multiple virtual machines.

In a logically partitioning architecture, all of the server memory is physically accessible to all the processor cores and all of the I/O devices in the system, regardless of physical placement of the memory or where the logical partition operates. The POWER Hypervisor is designed to ensure that *any* code running in a partition (operating systems and firmware) only has access to the physical memory allocated to the dynamic logical partition. Power Systems models also have IBM-designed PCI-to-PCI bridges that enable the POWER Hypervisor to restrict DMA (Direct Memory Access) from I/O devices to memory owned by the partition using the device. The single memory cache coherency domain design is a key requirement for delivering the highest levels of SMP performance. Since it is IBM's strategy to deliver hundreds of dynamically configurable logical partitions, allowing improved system utilization and reducing overall computing costs, these servers must be designed to avoid or minimize conditions that would cause a full server outage.

IBM's availability architecture provides a high level of protection to the individual components making up the memory coherence domain; including the memory, caches, and fabric bus. It also offers advanced techniques designed to help contain failures in the coherency domain to a subset of the server. Through careful design, in many cases failures are contained to a component or to a partition, despite the shared hardware system design. Many of these techniques are described in this document.

Not only does this architecture allow exceptional configuration flexibility, it delivers capabilities not available in other, less flexible, architectures. Because processors, and processor cycles, can be dynamically reassigned between logical partitions, *processor sparing* includes not only reassigning full processor cores but also reassigning cycles (performance) between partitions helping to avoid outages when extra (spare) cores are not available.

By definition, Alternate Processor Recovery requires spare resource (equal to a core) if workload is to be moved from a failed processor. The Predictive Processor Deallocation technique also includes the ability to use a spare core ensure that partitions continue to operate without loss of performance.

Leveraging virtualization, the system takes an expansive view when defining a “spare” core. With PowerVM, a Power Server has considerable flexibility in collecting unused resources, even fractions of processing capacity, to create the “spare.”

Finding spare capacity, in priority order:

- *An unlicensed Capacity on Demand (CoD) core.* The most obvious spare core in a system is one that is currently unlicensed in the system – one held in reserve for future use. This unlicensed core can be used to automatically “back-fill” for the deallocated bad processor core. In most cases, this operation is transparent to the system administrator and to end users, and the server continues normal operation with full functionality and performance. CoD sparing does not require an additional license and does not impact subsequent core licensing when the system is repaired.

This approach has some key advantages:

1. the spare can come from anywhere in the system, it need not be in the same physical enclosure or partition as the failing core and
 2. a single spare is sufficient to allow for a deallocation event anywhere within the entire server.
- *If no CoD processor core is available,* the POWER Hypervisor attempts to identify an un-allocated licensed core somewhere in the system. This licensed core will be substituted and the failing CPU deallocated.
 - *If no spare processor core is available,* the POWER Hypervisor attempts to identify a full core processor capacity equivalent from a shared processor pool (when enabled) – a capability enabled by logical partitioning, because resources can be shared anywhere – and redistributes the workload. In this case, the POWER Hypervisor reduces the allocation of processing resources given to partitions to account for the loss of a processor. This can be done at the granularity of a fraction of a processor.
 - *If the requisite spare capacity is not available,* such as when the partitions in aggregate require more processing resources than would be available when a processor is deallocated, or if a shared processor pool is not used.
 1. Predictive Processor Deallocation:
 - Shared pool partitions: a core will not be deallocated until the next server reboot.
 - Dedicated processor partitions: core deallocation will proceed (2 or more cores in partition) but no sparing will occur. In a uniprocessor partition, no core is deallocation until server reboot.
 2. Alternate Processor Recover: At least one partition will be terminated – not necessarily the one using the unreliable core. Core selection is determined by the POWER Hypervisor using Partition Availability Priority.

Partition Availability Priority

Power Systems give users the ability to prioritize partitions in advance. When partition availability priorities are specified, the POWER Hypervisor is able to terminate a lower priority partition (or partitions if necessary) and use the now available compute resource to keep higher priority partitions active.

When partitions are prioritized, even in those instances in which a partition need not be terminated, the priorities guide the order of deallocation of fractional resources from processors.

Additional POWER7 Processor Core Recovery Design Attributes

Level 1 Caches (Instruction and Data Cache Handling)

Instruction and data caches, typically considered part of a processor core, are designed so that detected errors are corrected, if necessary, by fetching faulty data or instructions from elsewhere in the memory hierarchy. The caches are organized into multiple data sets. If a persistent error is discovered in a set, the core can stop using that set — effectively repairing the problem. Multiple *set delete* events can also trigger Predictive Processor Deallocation, removing the bad cache (and core) before a catastrophic error is encountered.

Partition Availability Priority:

You can change the partition availability priority for the following partitions by first selecting one or more partitions and then choosing an availability priority from the field below the table. Please press OK button to submit your changes.

Select	Partition Name	Partition Type	Processing units	Processing Mode	Availability priority
<input type="checkbox"/>	Sys_Dump_1	AIX or Linux	0	Shared	127
<input type="checkbox"/>	lpar4_RH_all	AIX or Linux	0	Shared	127
<input type="checkbox"/>	lpar5_AIX_all	AIX or Linux	0	Shared	127
<input type="checkbox"/>	lpar6_SuSe_all	AIX or Linux	0	Shared	127
<input type="checkbox"/>	lpar_1	AIX or Linux	0	Shared	63
<input type="checkbox"/>	lpar_2	AIX or Linux	2.0	Shared	127
<input type="checkbox"/>	lpar_3	AIX or Linux	0	Shared	127
<input type="checkbox"/>	lpar_3a	AIX or Linux	4.0	Shared	127
<input type="checkbox"/>	lpar_7	AIX or Linux	0	Shared	127

Availability priority:

Partition Availability Priority

If Processor Instruction Retry does not successfully recover from a core error, the POWER Hypervisor will invoke Alternate Processor Recovery, using spare capacity (CoD or unallocated core resources) to move workloads dynamically. This technique can maintain uninterrupted application availability on a POWER6 processor-based server.

Should a spare core not be available, administrators can manage the impact of Alternate Processor Recovery by establishing a Partition Availability Priority. Set via HMC configuration screens, Partition Availability Priority is a numeric ranking (ranging from 0 to 255) for each partition.

Using this rating, the POWER Hypervisor takes performance from lower priority partitions (reducing their entitled capacity), or if required, stops lower priority partitions so that high priority applications can continue to operate normally.

Level 2 and Level 3 Caches

The level 2 and level 3 caches in the POWER7 processor use at least double error detect/single error correct (DED/SEC) ECC codes to ensure that a two bit error in any ECC word of data can be detected and that a single bit error in any ECC word can be corrected. This mechanism, when combined with the layout of the various bits of data within the ECC word provides primary protection against a soft error in either cache.

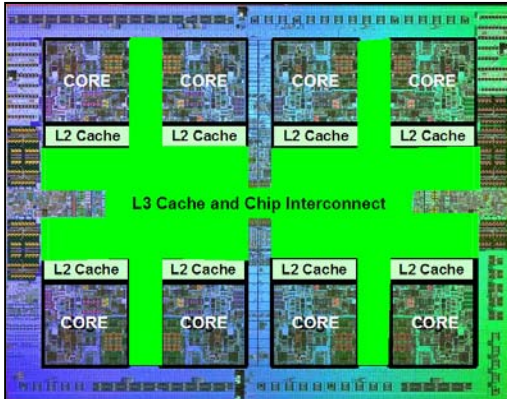
When data in a cache is read, errors are monitored by the

cache controller. If a set of cache data — a cache line — appears to have a persistent, correctable error, the cache controller will predictively deallocate the cache line through a procedure known as *cache line delete*.

In this process, the cache line is identified in the cache line directory as unavailable. Before the line is deleted, if any data from the cache needs to be written to memory, because it was modified in the cache, the data will be corrected using ECC and written to memory.

If, despite these precautions, data from a cache line is found to have an error that can not be corrected by ECC, then the cache line will still be deleted. If the data in the cache was unmodified, it will be “recovered” by fetching another copy from main memory. However, if the data in the cache line was modified in the cache, then the data will be written into main memory. Any faults that can be corrected by ECC will be corrected during the write. Any faults that can not be corrected will be identified as the data is stored by writing a special code into the ECC check bits of the words that were uncorrectable.

The firmware will not terminate any processes or partitions simply because an uncorrectable error was discovered. Instead, the use of the marked bad data will be monitored in the system through a process known as *Special Uncorrectable Error* handling. If the data is never used, computing will continue without interruption, however, if the data is used, in many cases the impact may be limited to just an application or partition termination.



L2 and L3 Cache Data Handling

The POWER7 processor chip consists of eight processor cores, each with a 32-KB 8-way set associative data cache, a 32-KB 2-way set associative instruction cache, and a tightly coupled 256-KB 8-way set associative level 2 cache.

The chip also uses embedded dynamic ram technology (eDRAM) to deliver 32MB of internal L3 Cache. This onboard L3 cache is very important to delivering a balanced system design.

The level 2 and level 3 caches in the POWER7 processor use at least double error detect/single error correct ECC codes to ensure that a two bit error in any ECC word of data can be detected and that a single bit error in any ECC word can be corrected. This mechanism, when combined with the layout of the various bits of data within the ECC word provides primary protection against a soft error in either cache. A variety of other mechanisms are used to protect data.

Single bit soft errors

- Corrected with ECC

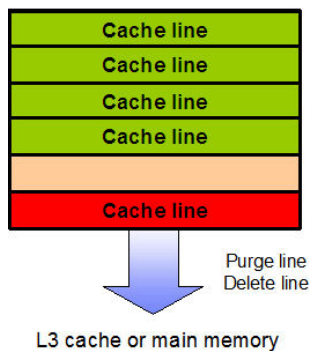
Single bit hard errors

- Cache line purged
 - Data unmodified in cache is invalidated
 - Data modified in cache is written to memory
 - ✓ Marked by HW as uncorrected (Special UE mark)
- Cache line dynamically deleted
 - Cache line no longer used, no future issues

Multi-bit error

- Cache line purged
 - Data unmodified in cache is invalidated
 - Data modified in cache is written to memory
 - ✓ Any uncorrectable data will be marked with a special UE code and handled when/if data is used
- Cache line dynamically deleted
 - Cache line no longer used, no future issues

L2 or L3 Cache



An Uncorrectable data Error (UE) has potentially severe consequences. If a UE is encountered, the system firmware will request that the part containing the cache be replaced. However, if the fault is isolated to a single cache line, the cache line delete function is sufficient to prevent further problems associated with the error.

The L2 and L3 caches are controlled by much smaller arrays called directories which include information about the contents of particular cache lines. These directories are covered by a DED/SEC ECC code.

Special Uncorrectable Error Handling

While it's a rare occurrence, an uncorrectable data error can occur in memory or a cache despite all precautions built into the server. The goal of all Power Systems is to limit, to the least possible disruption, the impact of an uncorrectable error by using a well-defined strategy that begins with considering the data source.

Sometimes an uncorrectable error is transient in nature and occurs in data that can be recovered from another repository. For example: An L3 cache can hold an unmodified copy of data in a portion of main memory. In this case, an uncorrectable error in the L3 cache would simply trigger a "reload" of a cache line from main memory. This capability is also available in the L2 cache.

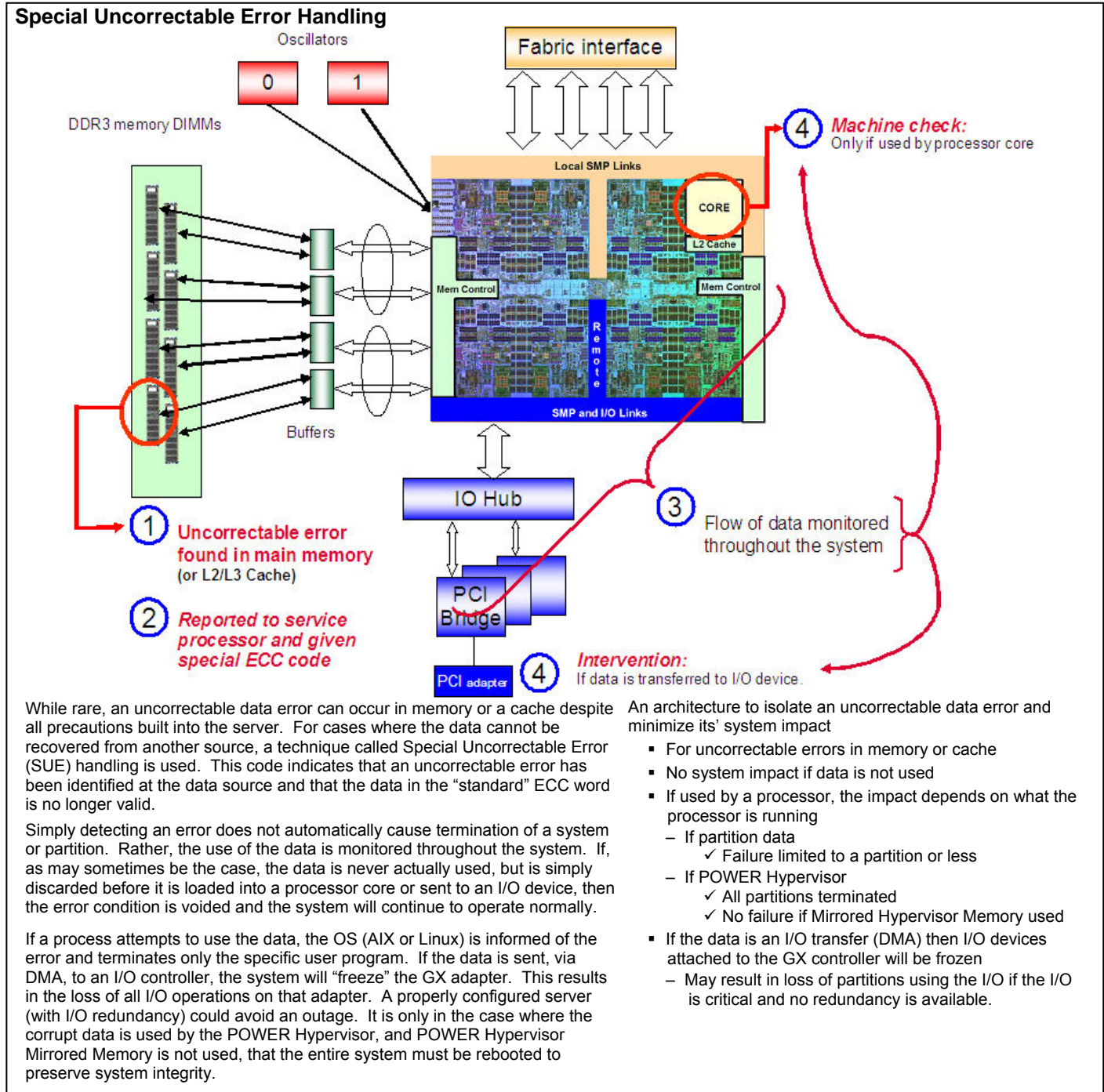
For cases where the data cannot be recovered from another source, a technique called Special Uncorrectable Error (SUE) handling is used. A SUE can be detected a number of ways:

1. ECC words containing uncorrectable errors in Level 2 and Level 3 caches will eventually be written to memory with a Special Uncorrectable Error (SUE) code indicating that the data is not reliable.

- Faults in other parts of the system, such as main memory, can also cause data words to be marked with the SUE code.

This code indicates that an uncorrectable error has been identified at the data source and that the data in the “standard” ECC word is no longer valid. The check hardware also signals the service processor and identifies the source of the error. The service processor initiates actions to identify the component(s) containing the fault so that they will be called out for repair. The service processor also takes appropriate action to handle the error.

Simply detecting an error does not automatically cause termination of a system or partition. Rather, the



use of the data is monitored throughout the system. If, as may sometimes be the case, the data is never actually used, but is simply discarded before it is loaded into a processor core or sent to an I/O device, then the error condition can safely be voided and the system will continue to operate normally.

If processor core tries to load the bad data, however, the request causes generation of a synchronous machine check interrupt. The firmware provides a pointer to the instruction that referred to the corrupt data. This data is often useful in mitigating the problem:

1. If the referenced data is owned by a user application (user space code), the operating system (currently supported versions of AIX and Linux) has the option of just terminating the application owning the data. If the data is operating system kernel data, then the partition will typically be terminated, but the rest of the system can continue to operate.
2. Only in the case where the corrupt data is owned by the POWER Hypervisor, or in a critical area of POWER Hypervisor memory, would the entire system be terminated and automatically rebooted, preserving overall system integrity.

If the data is sent, via DMA, to an I/O controller, the system will “freeze” the GX adapter. This results in the loss of all I/O operations on that adapter. A properly configured server (with I/O redundancy) could avoid an outage.

This design for handling a random UE is tuned to maximize the availability of applications and partitions since nothing is terminated *until* and *unless* necessary to prevent the system from using incorrect data. When termination does occur, *only the programming element owning the bad data need be terminated.*

Power 795: A Multi-Node Server with a Fabric Interconnect

The IBM Power 795 server is designed to support large-scale transaction processing and database applications within a highly virtualized system infrastructure, enabling new levels of workload consolidation, resource utilization, and efficiency. As the most powerful member of the IBM Power Systems family, this server provides exceptional performance, massive scalability, and bandwidth to efficiently and concurrently support a full range of complex, mission-critical applications.

Equipped with up to 256 POWER7 processor cores, the Power 795 server can scale rapidly and seamlessly to address the changing needs of today's business climate. The Power 795 server is designed to grow with a business by offering extraordinary scalability and configuration flexibility. Processor books, memory, I/O drawers, adapters, and disk bays can be easily added to realize the potential power and capacity of the system.

Extensive mainframe-inspired reliability, availability and serviceability (RAS) features in the Power 795 help ensure that mission-critical applications run reliably around the clock. All POWER7 processor modules are connected together using processor-to-processor fabric busses. Multiple fabric bus interfaces in each processor module allow module-to-module point-to-point connections within a node and point-to-point connections between nodes — all without intervening switching or cross-bar modules. In a multi-node server, this architecture allows a processor book to be deconfigured or reconfigured within a system while the rest of the processor books stay connected. This operation can be accomplished without a loss of performance, connectivity, or need to rebalance bus traffic among the fabric busses connecting the remaining nodes.

Using local and remote SMP links, up to 32 POWER7 chips are connected.

CEC nodes and Processor to Processor Interfaces (Fabric Busses)

The Power System 770, 780, and 795 servers use a “building block” or “node” hardware architecture in the Central Electronics Complex (CEC). In these servers, a system can include one or more nodes each containing one or more POWER7 chips, memory DIMMs, I/O hub chips, and associated components.

In a Power 770 or 780, a node is a CEC drawer. Up to four CEC drawers are connected together by fabric bus cables into a highly scalable system.

The Power 795 is constructed with from one to eight processor/memory books. Each book is inserted into a CEC rack (like books on a bookshelf) and connected electrically via a passive (i.e. it contains no active electronic components) printed circuit board called the “midplane”.

All POWER7 processor modules are connected together using processor-to-processor fabric busses. Multiple fabric bus interfaces in each processor module allow module-to-module point-to-point connections within a node and point-to-point connections between nodes — all without intervening switching or cross-bar modules.

In a multi-node server, this architecture allows a processor book to be deconfigured or reconfigured within a system while the rest of the processor books stay connected. This operation can be accomplished without a loss of performance, connectivity, or need to rebalance bus traffic among the fabric busses connecting the remaining nodes.

In this way the node-structure hardware supports *dynamic removal of a node* within a system (after fabric activity has been suspended) as well as *dynamic integration of a repaired node* or *addition of a new node* during an upgrade activity.⁶

In addition to the architectural attributes of the fabric (connectivity, performance), this design is also well engineered from a transmission view. The fabric bus is very stable in terms of signaling characteristics — there is no expected soft error rate or intermittent failure rate of the bus due to electrical noise or timing issues, even on a fully configured system. This is due in part to IBM’s design ownership of all the components carrying the fabric bus signals. IBM engineers design, tune, and customize the components for each individual bus and system implementation.

An inherent objective of the fabric bus RAS design is to tolerate a failure of an individual data bit-lane without causing an uncorrectable error or a signal capacity degradation which could, in turn, cause performance issues. This is accomplished using a DED/SEC ECC code on the fabric bus which corrects both transient and solid errors on a data bit line.

Memory Subsystem

The memory subsystem of a POWER7 system contains three main parts: DRAM chips which store the data and comprise the “main memory” of the system, memory controllers (two per processor module) which manage the processor interface to memory, and memory buffer chips which interface between the memory controllers and the DRAM chips.

Two basic DIMM (Dual In-line Memory Module) designs are used:

1. A custom design that imbeds the memory buffer and DRAM chips on a DIMM designed to IBM specifications.
2. A design that uses industry standard memory DIMMs and places buffer chips on a separate printed circuit board.

Memory Bus Interfaces

Unlike the processor-to-processor fabric bus interfaces, the bus between the processor memory controllers and the buffer chips can be expected to exhibit an occasional soft failure due to electrical noise, timing drift, and a variety of other factors.

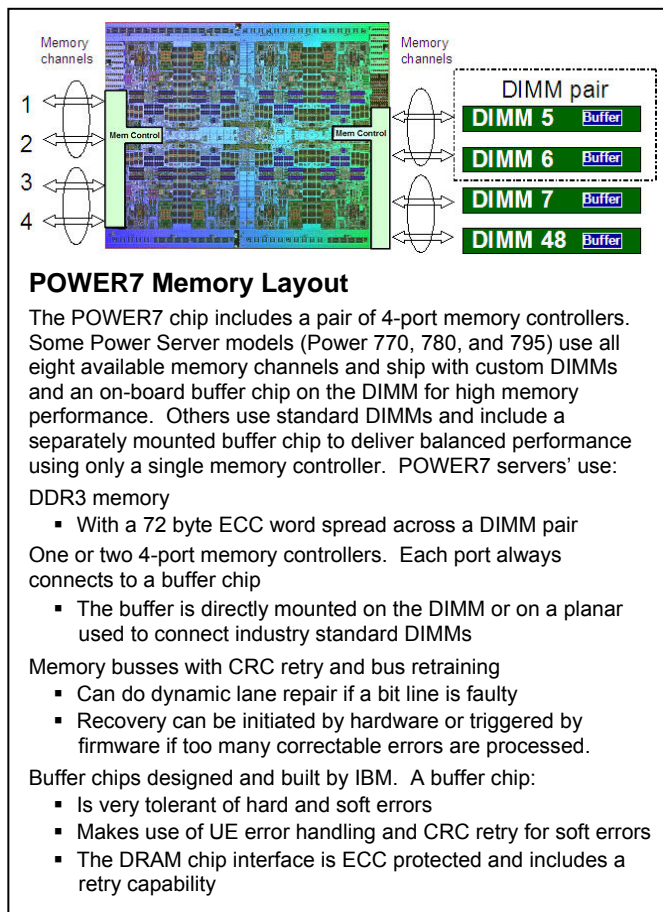
Because of this, the memory bus design differs significantly from the fabric bus interface. In this case, the design uses a cyclic redundancy check (CRC) code for error detection. This method has a robust ability to detect the types of multiple bit errors that might be expected due to soft bus error issues.

⁶ Refer to IBM United States Hardware Announcement 110-158, dated August 17, 2010 and IBM United States Hardware Announcement 110-025, dated February 9, 2010 for availability dates

Such codes typically provide detection but no correction of errors. On the memory bus, data can be corrected, however, by memory controller's ability to *retry a faulty operation*.

Each time a system is powered on, the memory fabric bus undergoes a series of tests designed to optimize data transfer performance. If a bus experiences multiple CRC errors that must be corrected by retry, the memory controller can be *dynamically retrained* to re-establish optimal bus performance.

If retraining a bus does not correct a persistent soft error, the fault could be due to a faulty bit line on the bus itself. The memory bus contains a *dynamic spare bit line* (dynamic memory channel repair) function which allows the memory controller to identify a persistently faulty bit line and to substitute a spare.



Memory Buffer

Considerable attention has been given to the RAS characteristics of the memory buffer chip. This IBM designed chip was built with the same sort of error detection and fault isolation techniques used in the POWER7 processor module. In both cases, a primary design goal was detection and recovery from soft errors.

When necessary, while waiting for a previous error recovery operation to complete, the memory buffer chip may temporarily "hold off" pending data transfers between itself and the memory controller by *signaling a CRC error* whenever there is data to be transferred. This means that a POWER7 memory subsystem can recover from soft errors in the buffer chips that could be fatal in other memory designs. The data bus between the memory buffer and the DRAM chips on a DIMM uses ECC protection: an error will result in retry of operations to correct transient bus errors

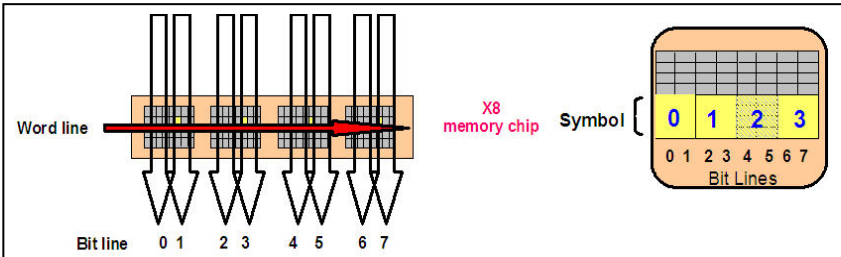
Memory DIMMs and ECC Words

Main memory in a POWER7 system consists of memory DIMMs which are in turn composed of a number of DRAM chips. Most POWER7 systems use a "by 8" DRAM design, indicating that an individual DRAM chip supplies eight bits of data at a time. Each "bit" being read from a DRAM chip is called a *bit line*.

Data is read from and stored into memory in ECC words. An ECC word is a collection of data and associated check bits used to determine if data was correctly stored and accessed.

In POWER7 an ECC word consists of seventy-two bytes of data. Of these, sixty-four bytes are used to hold data. The remaining eight bytes are used to hold check bits and additional information about the ECC word.

POWER7 systems access memory DIMMs in pairs — with the seventy-two bytes of an ECC word spread equally across two different DIMMs — such that thirty-six bytes of an ECC word come from each DIMM. Each DIMM is further divided into multiple ranks of data. A *memory rank* is a unit of data created using some or all the memory chips on an individual memory DIMM. This design is used primarily to help deliver improved performance, but it also provides certain RAS advantages such as simplifying the detection of a bad memory buffer chip or DIMM in an ECC word.



DRAM Chip Organization

This logical diagram portrays one way to arrange a memory DRAM chip. The DRAM consists of an array of thousands of memory cells. These cells may be accessed in many ways, depending on the organization of the chip. This is an example of an x8 (read: by 8) memory chip. Data is simultaneously accessed in eight bit lines. Bit lines are aggregated from multiple DRAM chips to form memory words.

The POWER7 ECC scheme uses four bits from each of two bit lines to form an eight bit symbol. Each DRAM chip provides four symbols (or four bytes) for the ECC code. In a POWER7 memory subsystem, eighteen DRAM chips would be accessed simultaneously to gather the seventy-two bytes needed for the ECC scheme. These would come from two DIMMs (a DIMM pair).

The basic ECC design is set up to ensure that if an entire memory chip on a DIMM is faulty, it can be permanently identified and marked as bad. Then the ECC algorithm, knowing that a DRAM chip is bad, can continue to process data while ensuring that the remaining data in the ECC word is checked, corrected (if necessary), and transmitted properly. This level of error correction is known as *Chipkill* correction. Chipkill is the foundation for memory RAS because it ensures that a system can continue to operate correctly even if a single permanent DRAM chip error occurs.

This level of DRAM protection was provided on POWER6 systems; however, while POWER6 systems could correct an entire Chipkill event, in some configurations, it was possible that a single fault elsewhere in an ECC word could lead to an uncorrectable memory error. To protect against this event, POWER6 systems provided redundancy in the memory such that, up to the redundancy provided, a faulty bit line on a DRAM chip could be replaced by a spare bit line from another DRAM chip.

For POWER7 servers, the basic ECC design itself has been improved for all configurations such that once a DRAM chip has been marked as bad, the ECC algorithm can *correct* an error even if *another nibble (bit line)* experiences a fault.

This error correction capability allows a DIMM to remain in a system even with a faulty DRAM chip, since

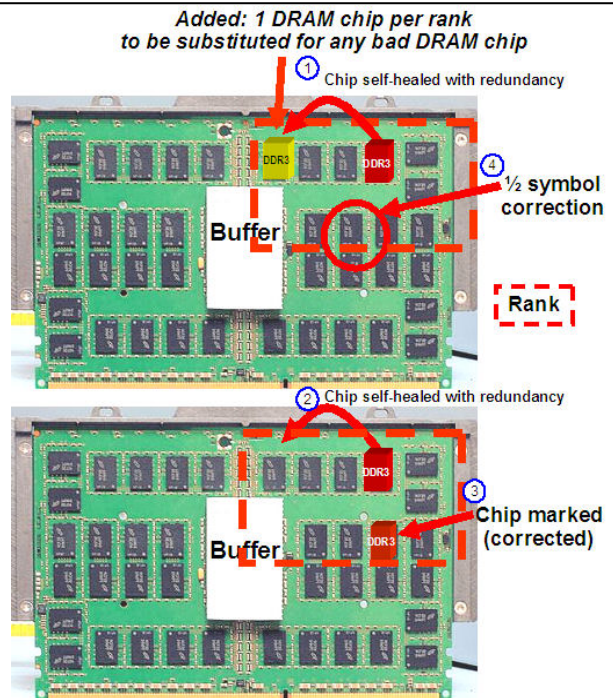
POWER7 DIMMs with Integrated Buffer Chips — a “Self-healing” Memory Design

For POWER7 servers, the basic ECC design itself has been improved for all configurations such that once a DRAM has been marked as bad, the ECC algorithm can correct an error even if another bit line experiences a fault.

This error correction capability allows a DIMM to remain in a system even with a faulty DRAM chip, since a random fault in another ½ symbol (nibble) can still be corrected

This photograph shows engineering models of the types of DIMMs used in Power 770, 780, and 795 servers. Ranks may be organized differently than depicted here. These models use custom DIMMs that include spare capacity.

- These include an extra DRAM chip per rank per DIMM that can be substituted for a bad DRAM chip.
- After substitution they have all the error detection and correction of the basic configuration
- A single DIMM can be composed of multiple ranks, each rank having spare capacity. The number of ranks and amount of spare capacity varies depending on memory DIMM size.



a random fault in another ½ symbol (nibble) can still be corrected.

However, because of the need to minimize both planned and unplanned outages, in systems with custom DIMMs (Power 770, 780, and 795), the RAS capabilities of DIMMs have been extended beyond this base protection with the addition of *spare memory chips*. In these servers, a memory rank consists of ten DRAM chips. Each DRAM chip provides eight bit lines of data. For ECC purposes, bit lines are organized in pairs and data is accessed as ECC symbols from each bit line pair. A symbol, in this case, consists of 4 bits per bit line (a nibble), or a byte per bit line pair. Thus, each DRAM chip delivers four bytes of data to the ECC word. Nine DRAM chips would deliver thirty-six bytes and the DIMM pair provides the full seventy-two bytes used by the error correction scheme. The “extra” DRAM chip (number ten) provides a spare DRAM chip for every memory rank on each DIMM. If a solid fault occurs on a DRAM chip, the chip is marked as “bad” in the ECC scheme. The data from the faulty DRAM chip is then migrated to the spare DRAM, and the system is restored to the initial ECC state (i.e. with full ECC protection). This is an excellent example of a “self-healing” architecture — in which the server automatically substitutes good components for failing ones, with no loss of performance or capabilities.

Since an ECC word is spread across a rank pair, there are two spare DRAM chips per rank pair. Each of these extra DRAM chips can be used as a spare for a faulty DRAM chip on the same DIMM. Thus, it is possible that in a DIMM pair, one DRAM chip could be faulty and spared on one DIMM, followed later by a similar DRAM chip fail and spare on the other DIMM in the pair. With this design, the system could tolerate yet another DRAM chip failure and still correct a bit line failure on one of the DRAM chips in the rank pair.

With this amount of sparing capacity it is expected that even planned outages for replacement of DIMMs due to DRAM chip failures will be greatly reduced.

Scrubbing, Page and Logical Memory Block Deallocation

In addition to simply detecting and correcting faults when they occur, POWER7 system memory controllers proactively scrub memory under the direction of the service processor. Memory scrubbing involves periodically reading all the memory in the system, looking for data errors. This scrubbing involves all the memory in the system, whether licensed or unlicensed.

Through this process if a single cell (single bit at a single address) is found to be persistently faulty, the service processor can work through the POWER Hypervisor to request that the page of memory containing the address not be assigned to any partition or hypervisor code, and if already assigned, request that the page be deallocated. This *dynamic page deallocation* is designed so that a persistent fault in a single memory cell on a DRAM chip does not align with a fault on a different DRAM chip in the same memory word at the same address (thus avoiding double bit errors).

In hardware, memory is collected into ECC words of seventy-two bytes spread across two DIMMs. At the OS level, memory is accessed in memory pages, which are collections of data on the order of a few kilobytes. Memory page size is defined by the operating system. Memory is assigned to operating systems in logical memory blocks (LMB) where a logical memory block is a collection of memory pages on the order of 16 to 256 MB in size.

For optimal performance, a single LMB typically will contain data from all the DIMM pairs connected to a processor module. For an enterprise class system like a Power 795, each of LMB in a system may contain data from all 8 DIMMs connected to a POWER7 processor chip.

If an uncorrectable memory error is detected during a scrubbing operation, the POWER Hypervisor will request that the associated LMB *be deallocated* from the system and not-reused. If the LMB was assigned to an active partition, the partition will be terminated before the LMB is deallocated. The service processor will also request that the memory DIMM pair containing the uncorrectable error be replaced.

In a similar fashion, if an uncorrectable error in memory is encountered by the POWER Hypervisor or an application during normal system operations, the special uncorrectable error handling mechanism will be used to minimize to the extent possible any associated outage due to the UE, and the LMB containing the uncorrectable error will be deallocated as indicated above.

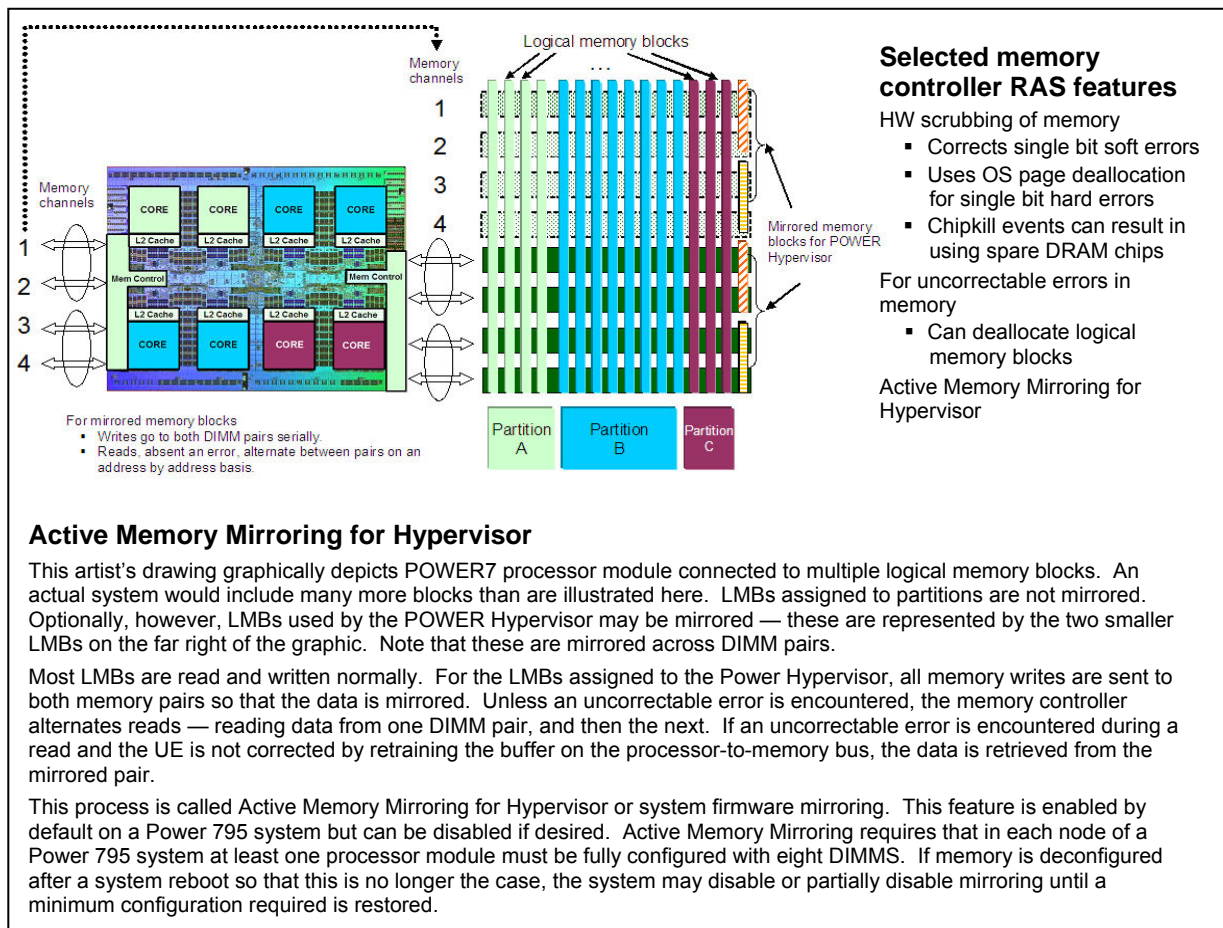
Active Memory Mirroring for Hypervisor

For systems with custom DIMMs, the probability of encountering an uncorrectable error in memory caused by faulty DRAM chips is very low. This is due to the self-healing design which couples an advanced ECC scheme with spare DRAM chips to automatically detect and correct memory errors and to move data to backup chips when necessary. As described earlier, servers equipped with these DIMMS included the ability to successively correct two DRAM chips. In fact, in some cases faulty data from three DRAM chips in a DIMM pair plus at least an additional bit line may be corrected (for every rank pair in the DRAM pair). The probability of a UE is even further reduced if, when the service processor indicates that a DIMM needs to be replaced, the repair is completed in a timely fashion.

Another potential source of a memory UE is a failure in DIMM logic managing the DRAM chips or involved in data movement. A fault in this “other” circuitry could include a catastrophic failure of the buffer chip, or some other miscellaneous fault on the DIMM such as a voltage regulator. Such a fault has the potential to cause multiple uncorrectable errors not contained to a single LMB.

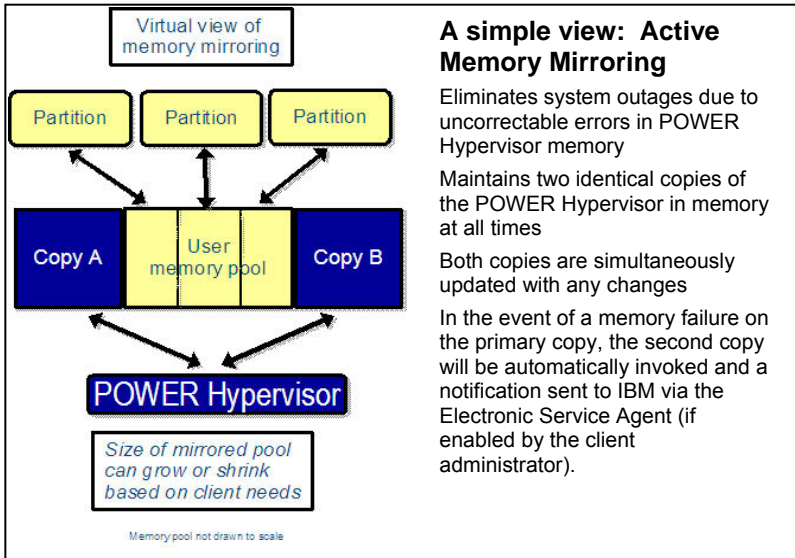
The memory buffer chip (page 27) is designed to detect and correct soft errors and IBM controls the manufacturing process of the buffer chips to assert control over chip quality. Nonetheless, if a catastrophic DIMM level failure would occur, then all the partitions in the system assigned an LMB on the faulty DIMM could experience outages. In addition, if the DIMM contains even one LMB storing critical data from the POWER Hypervisor, then a system-wide outage could result.

To guard against this last possibility, the memory controller of the Power 795 has been designed to allow mirroring ECC words in one DIMM pair with ECC words in another DIMM pair. The function is selective in that it does not require that all or even a substantial portion of the memory to be mirrored. In practice the



function is used to mirror only the LMBs used by the POWER Hypervisor.

Most LMBs are read and written normally. For the LMBs assigned to the POWER Hypervisor, all the writes to memory are sent across both memory pairs so that the data is mirrored. Unless an uncorrectable error is encountered, the memory controller alternates reads — reading data from one DIMM pair, and then the next. If an uncorrectable error is encountered during a read and the UE is not corrected by retraining the buffer on the processor- to-memory bus, the data is retrieved from the mirrored pair.



This process is called *Active Memory Mirroring for Hypervisor* or system firmware mirroring. This feature is enabled by default on a Power 795 system but can be disabled if desired. Active Memory Mirroring requires that in each node of a Power 795 system at least one processor module must be fully configured with eight DIMMS. If memory is deconfigured after a system reboot so that this is no longer the case, the system may disable or partially disable mirroring until a minimum configuration required is restored.

While this feature prevents system-wide outages due to a UE in POWER Hypervisor memory,

partitions may still experience outages. Exactly what and how partitions are affected depends on specific partition LMB assignments when the DIMM fault occurs.

Further, after a DIMM failure partition outage, restarting affected LMBs may not be possible or even advisable. If the UE was due to a catastrophic DIMM failure, all LMBs associated with the DIMM may eventually experience UE events. If a UE occurs during the reboot of a partition it could prevent that partition from rebooting.

Persistent Deallocation of Components and System IPL

Detecting and Deallocating Failing Components

Run-time correctable/recoverable errors are monitored to determine if there is a pattern of errors or a “trend towards uncorrectability.” Should a component reach a predefined error limit, the service processor will initiate an action to deconfigure the “faulty” hardware, helping avoid a potential system outage and enhancing system availability. This process has been illustrated via a number of examples (e.g., dynamic processor deallocation, cache line delete, dynamic page deallocation, memory block delete). Error limits are preset by IBM engineers based upon historic patterns of component behavior in a variety of operating environments. Error thresholds are typically supported by algorithms that include a time-based count of recoverable errors; that is, the service processor responds to a condition of too many errors in a defined time span.

Persistent Deallocation

To enhance system availability, if a fault is detected and a component is called out for repair, the system will mark the failing element for deconfiguration on subsequent system reboots. This is known as persistent deallocation and is intended to ensure that a failing component will not be put back into service if the system is rebooted before the part is repaired.

Component removal can occur either dynamically (while the system is running) or at boot-time (IPL), depending both on the type of fault and when the fault is detected.

In addition, components experiencing run-time unrecoverable hardware faults can be deconfigured from the system after the first occurrence. The system can be rebooted immediately after failure and resume operation on the remaining good hardware. This prevents the same “faulty” hardware from again affecting the system operation so that the repair action can be deferred to a more convenient, less critical time.

For various reasons, including performance and preparation for maintenance, the set of components deconfigured while a system is running is not necessarily the same set deconfigured on system reboot. For example, if an uncorrectable error occurs in a level 2 processor cache during runtime, the cache line containing the error will be deleted. If a large number of cache lines are deleted, on reboot it may be more efficient and better for performance to deconfigure the entire cache and its’ associated processor core.

Likewise, while individual LMBs of main memory may be deconfigured during system operation, on a reboot an entire DIMM will be deconfigured. Since ECC words are always spread across two DIMMs, the other DIMM in the DIMM pair will also be deconfigured by association. Though this removes two entire DIMMs from the system, it allows for reallocation of logical memory blocks on the remaining DIMMs and is often the most efficient way to configure the system for performance and memory capacity when multiple LMBs are involved in an outage.

Further, though not the default mode of operation, in some cases it might be useful to deconfigure an entire node on reboot if a component in the node has failed. This would allow for the rest of the nodes in the system to be brought up, while the deconfigured node is repaired and reintegrated at a later time.

Since systems are deployed and managed in many different environments, the user sets a policy that governs whether certain faults cause deconfiguration during run-time and reboot, and whether to deconfigure an entire node on reboot for a fault within the node.

Finally, if an established policy would cause a condition in which insufficient resources would be available to allow any part of a system to activate on a reboot, some components previously deconfigured may be restored to service — allowing continued system operation after a failure.

I/O Subsystem Availability

Base Design

Clearly, a system encompasses much more than a Central Electronics Complex — and so should the RAS design. Building a solid, reliable, available I/O subsystem is key to achieving a high level of system RAS. Unfortunately, the enormous variety and complexity of available I/O devices makes it difficult to cover the RAS characteristics of each device in depth. However, a description of how the system availability is impacted by the I/O subsystem is important to understanding application availability — and well within the scope of this document.

POWER7 servers are designed to allow continued availability of systems, partitions, and adapters. This is primarily accomplished by:

1. Building devices that connect the system to the I/O devices using reliable components that include a level of error resilience (that is, components that can tolerate some expected error types)
2. Offering redundancy in I/O adapters and devices
3. Insuring that devices and device drivers are built to exacting specifications and thoroughly tested prior to deployment.

When the first POWER4 system was introduced, the I/O adapters provided were called PCI adapters as the adapters supported the Peripheral Connect Interchange specification that was standard at the time.

These PCI adapters were intended to be connected to systems that would scale to support multiple I/O adapters in multiple I/O drawers. IBM, therefore, created a hierarchical I/O structure to support them.

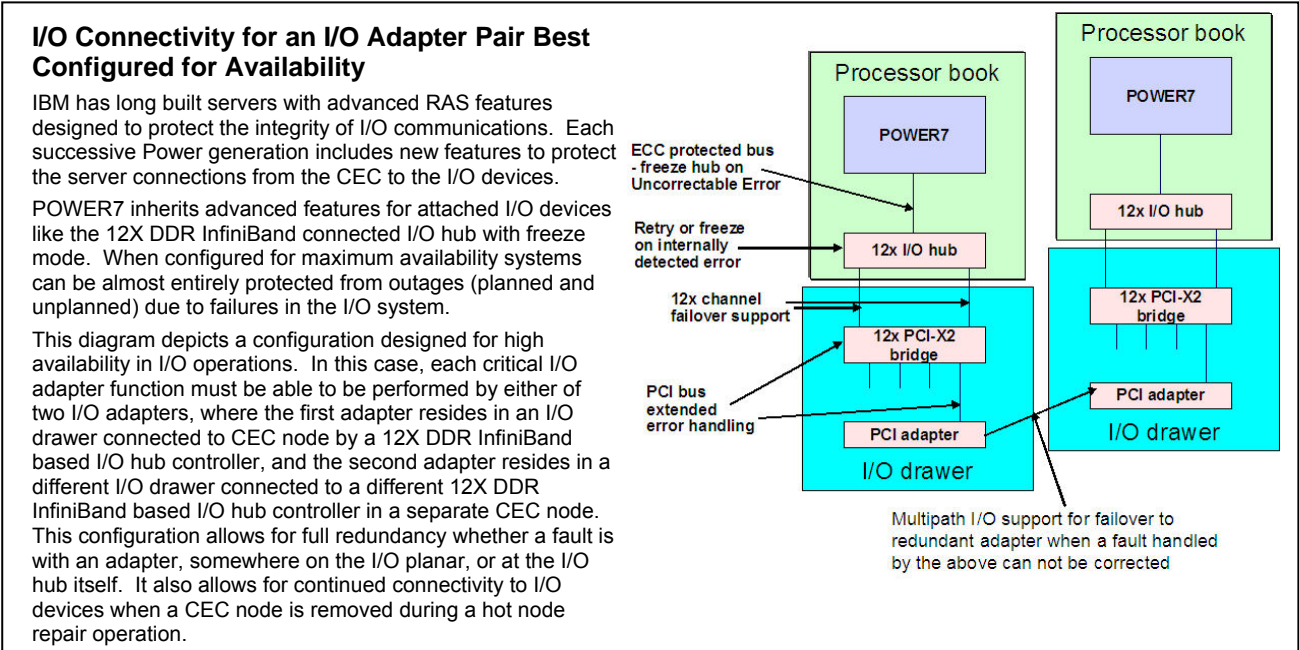
In this architecture, the system processor modules communicate by a high speed bus (named the GX bus) to an I/O hub card, known as a GX bus adapter.

I/O hub cards were designed with redundant links to I/O drawers that housed a set of PCI adapters. The bus between the I/O hub and I/O drawers was a high-speed link (called Remote I/O or simply RIO) that supported advanced RAS characteristics including retry of failed operations and redundancy with a failover capability that allowed I/O drawers to be unplugged and unplugged concurrently with the rest of system operation. Each I/O drawer had modules that connected to this bus and in turn performing a PCI Host Bridge function.

It was a major RAS goal for that system to support *dynamic removal and replacement of PCI adapters*. Achieving this objective required electrical isolation of PCI adapters (each adapter was placed in a separate power domain) so that the rest of the system was protected against possible transient electrical “glitches” occurring during a plugging operation. For this reason, IBM created a controller chip (the EADS controller) which received data from the PCI host bridge and created an *electrically isolated PCI bus* for each I/O slot it controlled.

The PCI specification at the time allowed for parity checking on the PCI bus. This allowed bus faults to be detected, but not corrected by the checking code. Given the large number of PCI adapters that might occupy an enterprise system, it was important to not only detect but also correct data bus errors in the I/O system. So IBM engineers included support in the EADS chip that allowed a device driver to be signaled when a PCI bus error occurred, permitting the device driver to reset the slot and retry the failed operation. If retries were not attempted or did not succeed, the design permitted the device driver to terminate use of the PCI adapter experiencing the faults without directly terminating a partition or system.

The basic concept for handling errors on the PCI bus was called *Enhanced Error Handling (EEH)* for I/O adapters.



Later Developments

This basic concept for I/O error handling was expanded in a number of ways over successive generations of adapters. The ability of an operating system to fail over to a redundant adapter in another I/O drawer of a system using techniques like *multi-path I/O* was added for cases where retrying operations would not fix a problem.

This mechanism was primarily designed to allow continued operations when a single PCI adapter suffered an unrecoverable hardware error. Over time, the EEH behavior for I/O adapters was expanded to include faults in the Processor Host Bridge, with the understanding that when a *PHB freezes*, all of the adapters underneath the PHB either need to be reset and I/O operations retried, or else the use of the adapters terminated.

Transitions from PCI to PCI-X and PCI-E and from RIO to RIO-G to InfiniBand

While these base availability improvements in the I/O subsystem were being made, some of the underlying structure of the I/O subsystem was also advancing. For example:

- Options were added to support PCI-X and PCI-E based I/O slots. These included support for new RAS characteristics inherent in the bus architectures.
- The I/O bus used to connect a CEC to its' I/O drawers was updated with improved performance when it migrated from a RIO bus, to a RIO-G bus, and eventually to a 12x Channel (that uses InfiniBand bus protocols).

I/O Hub RAS and I/O Special Uncorrectable Error Handling

The bus connecting a processor chip to an I/O hub card (the GX bus adapter) has been improved several times to increase speed and to add functional capabilities. The bus versions were identified by simply adding "+" or "++" after the bus name. However, whatever the version, a GX adapter in POWER7 systems maintains a number of important RAS features. The bus is ECC protected insuring that a single bit error could be continuously protected.

In addition, if data, marked with a Special Uncorrectable Error code, is transferred across the GX bus, the SUE is processed in the I/O hub controller. The hub controller ensures that the faulty data is discarded before it reaches an I/O adapter. Once discarded, all subsequent I/O operations to any device beneath (on the PCI bus "side") the hub are "frozen," causing device drivers to lose access to their devices. These failures are handled as persistent EEH events (and normal system operations can continue if proper redundancies are in place).

Faults on the interface between the I/O hub adapter and an I/O drawer are generally handled using redundant paths to the I/O drawers and retry mechanisms available in the bus protocol. However, due to the nature of the GX bus, an older I/O hub adapter (designed for no longer available RIO I/O drawers) could cause outages if an uncorrectable error originating on the bus or if certain other internal faults occurred in the hub. This design was employed to help insure data integrity — since incorrect data of this nature could be visible to not only the I/O hub device but also to the other processor components that comprise the cache coherency domain.

12x DDR InfiniBand Connected I/O Hub Adapters and "Freeze" Mode.

During the POWER6 timeframe, a new I/O hub adapter was introduced to connect I/O drawers using an optional dual 12x DDR InfiniBand connection (12x Channel Adapter). This adapter supported another major advancement in I/O subsystem protection.

The I/O hub controller was redesigned to work with the POWER processor so that errors on the hub related to either the GX bus or the processor can, in some instances, be re-tried, avoiding an outage for transient events. If retry is inappropriate for a hub error, or retry can't fix the problem, then in a fashion similar to the SUE case mentioned above, the I/O hub can freeze all connected I/O adapters. This *freeze mode* can be combined with I/O redundancy to prevent outages.

POWER7 I/O Enclosures and Integrated I/O

Attached I/O Availability

POWER7 inherits the RAS characteristics for attached I/O devices developed in previous processor generations. This includes options using the 12X DDR InfiniBand connected I/O hub with freeze mode. When configured for maximum availability systems, can be almost entirely protected from outages (planned and unplanned) due to failures in the I/O system.

To achieve this level of availability, each critical I/O adapter function must be able to be performed by either of two I/O adapters in a system where the first I/O adapter resides in an I/O drawer connected to CEC node by a 12X DDR InfiniBand based I/O hub controller, and the second adapter resides in a different I/O drawer connected to a different 12X DDR InfiniBand based I/O hub controller in a separate CEC node. This configuration allows for full redundancy whether a fault is with an adapter, somewhere on the I/O planar, or at the I/O hub itself. It also allows for continued connectivity to I/O devices when a CEC node is removed during a hot node repair operation.

It is also important to consider the RAS characteristics of the entities connected to these I/O adapters. For failover to work, of course, it is necessary that the network or devices that the adapters connect to can be controlled from multiple I/O controllers. It is also assumed that any critical storage is in some way redundant such as through DASD mirroring or RAID.

It should also be noted that each attached I/O drawer includes redundant power and cooling, providing improved levels of drawer availability.

Integrated I/O

In the previous discussion it was largely presumed that all I/O adapters in a system reside in separate I/O drawers. Different system models, however, allow I/O adapters and devices to reside in the CEC nodes themselves.

Each CEC drawer of a Power 780, for example, internally supports six PCI-E 8x slots, quad 1Gb Ethernet ports or optionally dual 10Gb optical/Twin Ax with dual 1Gb Ethernet ports, 2 SAS DASD controllers with internal RAID optional, a SATA media controller, and miscellaneous support for serial, USB, and Hardware Management Console connections.

While this “integrated I/O” contains many of the RAS characteristics described for separate I/O drawers, there are certain RAS limitations. For example:

- If a CEC node becomes unavailable for any reason, then all the integrated I/O in the CEC node also becomes unavailable.
- Power 770 and 780 systems use an integrated I/O hub controller which does not support I/O hub freeze mode. The integrated I/O Hub controller supports freeze at the slot level.
- While an internal RAID option is supported in each enclosure, using RAID does not eliminate the possibility of losing access to the integrated DASD devices when certain CEC outages occur, including planned maintenance using hot node repair.

Reliability and Availability of the System Infrastructure

General System Environments

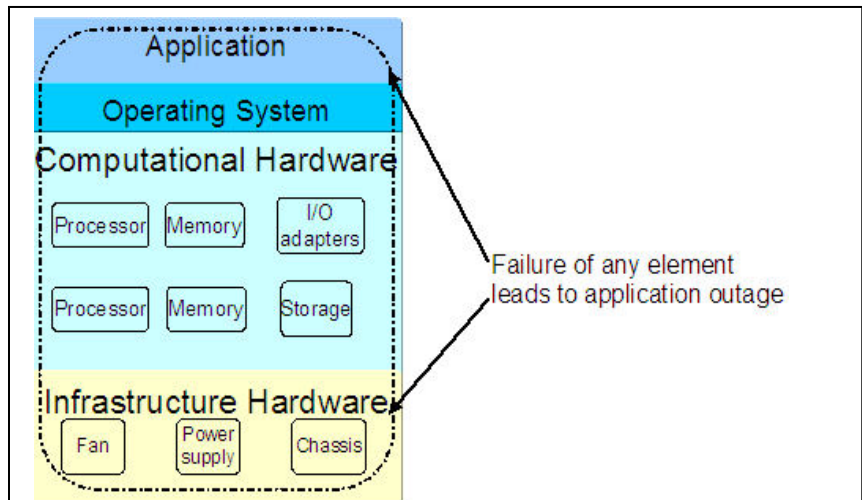
The preceding sections focused on RAS characteristics the computational resources of a computer system — processors, memory, and I/O. System hardware also includes infrastructure resources — power supplies, voltage regulators, fans and cooling components, clocking, and the infrastructure to support service processors.

System application availability depends on the reliability and availability of all of these resources, though the impact of the failure of such a resource may depend on how system resources are deployed and the hardware virtualization employed

Before focusing on infrastructure availability, it's useful to quickly examine various server environments to see how reliability of the data center infrastructure affects a design for availability.

Single Server/Operating System Environment

In a simple, standalone system, the failure of practically any element in the system could cause an application outage. Hence the availability of an application very directly depends on the availability of the computation components, infrastructure, OS, and so forth.

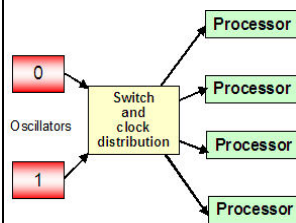


Hidden “Single Points of Failure” in Stand-alone Server Design

The goal of any server built to run “mission critical” applications is easy: keep applications running effectively. System level availability is a function of the reliability of the underlying hardware and the techniques used to mitigate the faults that do occur. To this end, engineers use highly reliable components in a highly available system design that insures most hardware failures will not result in an application loss. Therefore, engineers strive to:

1. Built a system environment conducive to the proper operation of components.
2. Identify components that have the higher failure rates. Design the server to recover from intermittent errors in these components and/or failover to redundant components. Use strategies like automated retry recovery or sparing (redundancy) for error recovery.

In a single system environment, while care may be taken to reduce the number of single points of failure (SPOF), these types of conditions are not always easily discernable. The example below shows two server designs that provide “redundant clocks.” Clearly, in one instance, while the server is protected from an oscillator failure, a failure in the clock switch component could still cause a server outage. Server designs are rife with such examples (see [I/O Connectivity for an I/O Adapter Pair Best Configured for Availability](#) on page 33).



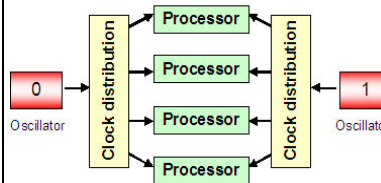
Clock Distribution Designs

Redundant oscillator design 1

- No outage on oscillator failure.
- Failure of switch/clock distribution component results in failure of all processors.
- If clock interface in processor fails, outage may be limited to the processor, depending on system design.
- Solid switch failure keeps system down even after reboot.

Redundant oscillator design 2

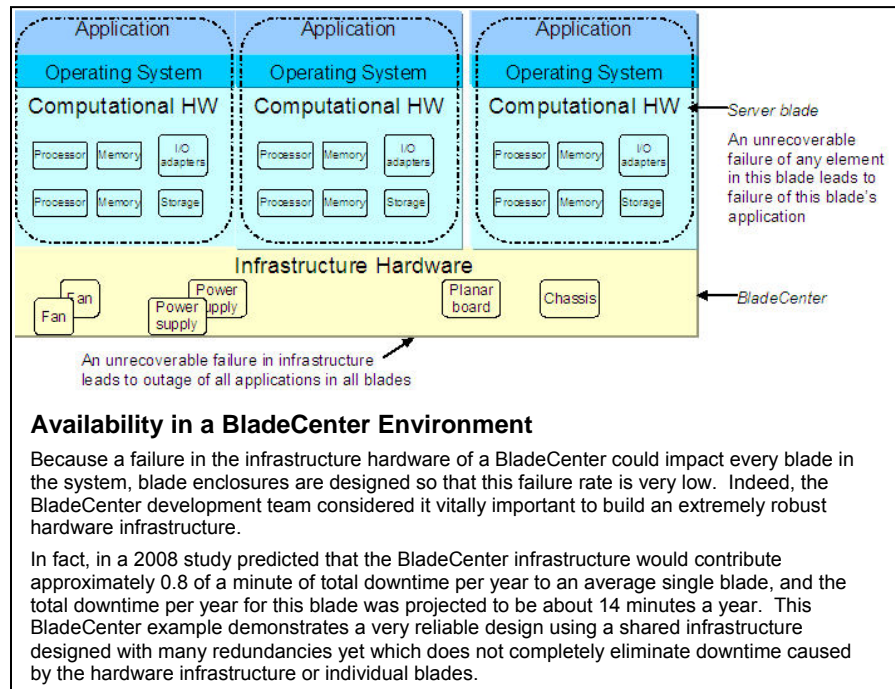
- No outage on oscillator or clock distribution component failure.
- If clock interface in processor fails, outage may be limited to the processor, depending on system design.
- No single point of failure prevents reboot.



BladeCenter™ and Blades Environment

An alternative to deploying multiple small standalone systems, some clients with many, relatively small workloads will deploy “blade” servers that can be packaged together in a single enclosure. Because of their general form factor (often a blade is built on single printed circuit card), these servers are can be easily integrated into a standard enclosure like that provided in the IBM BladeCenter™ environment.

In a BladeCenter, multiple independent server blades share a common infrastructure composed of common power supplies, central cooling elements, and facilities to provide interconnection between blades, a common chassis, and so forth.



Because a failure in the infrastructure hardware, could impact every blade in the system, blade enclosures are designed so that the failure rate of the infrastructure hardware is very low. This is typically accomplished by using components with very low intrinsic failure rates (e.g. passive planar boards with no active components), or by making use of redundancy (including, for example, redundant power supplies and fans).

Even in designs using all of these techniques, it is common to expect some failures or combinations of failures within the infrastructure hardware that can lead to both planned and unplanned outages.

Evaluating the Failure Rate of a “Redundant” and “Passive” Infrastructure

The BladeCenter development team considered it vitally important to build an extremely robust hardware infrastructure. Indeed, they proved to be quite successful and a casual view of the design description might lead to the conclusion that infrastructure failures never lead to outages of applications running in blades.

However, in 2008 IBM published an IBM Systems Journal article entitled “Availability analysis of blade server systems”⁷ that considered in depth the design and reliability of the infrastructure of a Blade/BladeCenter environment based on BladeCenter HS20 servers configured with Intel™ Xeon processors. The article showed how with considerable design focus, outages associated with a BladeCenter infrastructure could be predicted to be as little as approximately eight-tenths of a minute per year, this for both planned and unplanned outages.

While the infrastructure would be considered very robust, the paper described a number of reasons as to why zero downtime can not be expected from a common infrastructure.

⁷ W.E. Smith, K.S. Trivedi, L.A. Tomek, J. Ackaret, “Availability analysis of blade server systems”, IBM Systems Journal, Vol 47, No 4, 2008.

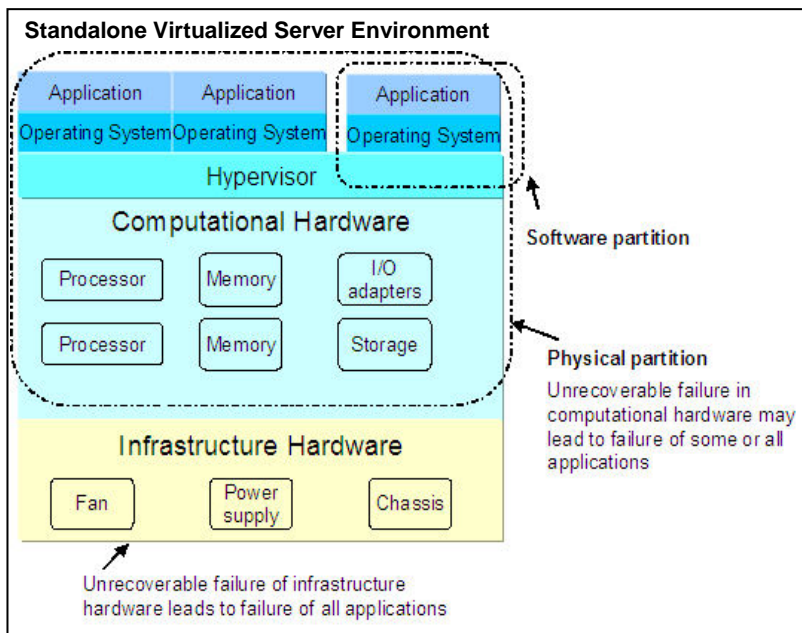
There are a variety of reasons for this. For example:

1. One of a pair of redundant components may fail, and then the other of the pair may also fail before the first one is replaced.
2. The failure of the first component in a redundant pair may uncover a hidden defect in the redundant pair, particularly if there is insufficient means in a running system to ascertain the health of the pair.
3. Failover to a redundant device might require code, and the code might have a defect.
4. Finally even in cases where two components are meant to be redundant there is often a shared component or connection between the redundant parts. The failure of that shared component can lead to an outage. A possible example of that might be the case of two fans in a system where each fan is powered by both power supplies in a system to ensure that when one power supply fails, both fans can still operate. If one of the fans failed in a system in such a way that an over-current (or electrical short) occurs, then both power supplies might see the over current condition and remove power to both fans. While engineers have devised ways to avoid or work-around many of these common mode failures, typically some such conditions remain.

Again, however, it is important to understand that while the impact the infrastructure has on blade outages can be minimized, the computational hardware is expected to be the primary hardware cause of blade outages.

While the infrastructure was predicted to contribute to approximately 0.8 of a minute of total downtime per year looking at the average single blade, the total downtime per year of a single blade was projected to be more like 14 minutes a year. This BladeCenter example demonstrates a *very reliable design* using a shared infrastructure designed with many redundancies yet which does not completely eliminate downtime caused by the hardware infrastructure or individual blades.

Standalone Larger SMP Virtualized System Environment



If there are sufficient processor and memory resources in a standalone server, it is possible to run multiple instances of an operating system within the single server.

This is typically accomplished using a layer of code called a "hypervisor" which resides in system memory and runs on the system processors. A hypervisor directly manages the hardware resources and provides a standard interface and a core set of services to operating systems. The hypervisor, sitting between an OS and the hardware, can "virtualize" the hardware — that is, it can provide the operating systems with a set of hardware resources (processors, memory,

and I/O) that may be dedicated (e.g., a processor core) or shared (e.g. a time shared core, where each "sharing" OS receives a percentage of available cycles on a core).

One straightforward approach to creating a hypervisor is to augment an existing operating system (which already knows how to schedule and manage hardware resources) with code that allows additional OS images to be booted. This type of hypervisor generally includes facilities to allow fine grained allocation

of resources to separate operating system images (partitions) and to provide security and isolation between images. These types of partitions are typically called *software partitions* or *soft partitions*.

While this structure generally provides software isolation between OS images (and applications), a catastrophic failure in either the hypervisor software or the infrastructure hardware would cause all of the “soft” partitions in a system to fail. In addition, a failure of any of the computational resources could cause a full system outage.

A more sophisticated hypervisor design would look less like an operating system — it would be smaller and thus less prone programming errors (“bugs”) — and would use specialized hardware facilities to allow better isolation of hardware failures to individual partitions. Even in this case, the extent to which isolation could be achieved would still to be limited by the overall hardware design.

As the BladeCenter example indicated, by using techniques to include redundancy, passive planars, and so forth, it should be possible to minimize the outages associated with the common infrastructure hardware. A very high MTBF for the hardware infrastructure should be possible.

However it would be misleading to expect that the MTBF of the applications would also be in the same range since this value refers to the infrastructure alone. All failures of the computational components (processors, memory, etc.) can also lead to outages of one or more partitions, depending on the hypervisor and hardware design. Failures of this class would tend to be the predominant source of application outages (although user, administrator, and software errors are often the most significant sources of application outages).

Multiple Large Standalone Systems or a Physically Partitioned Single System

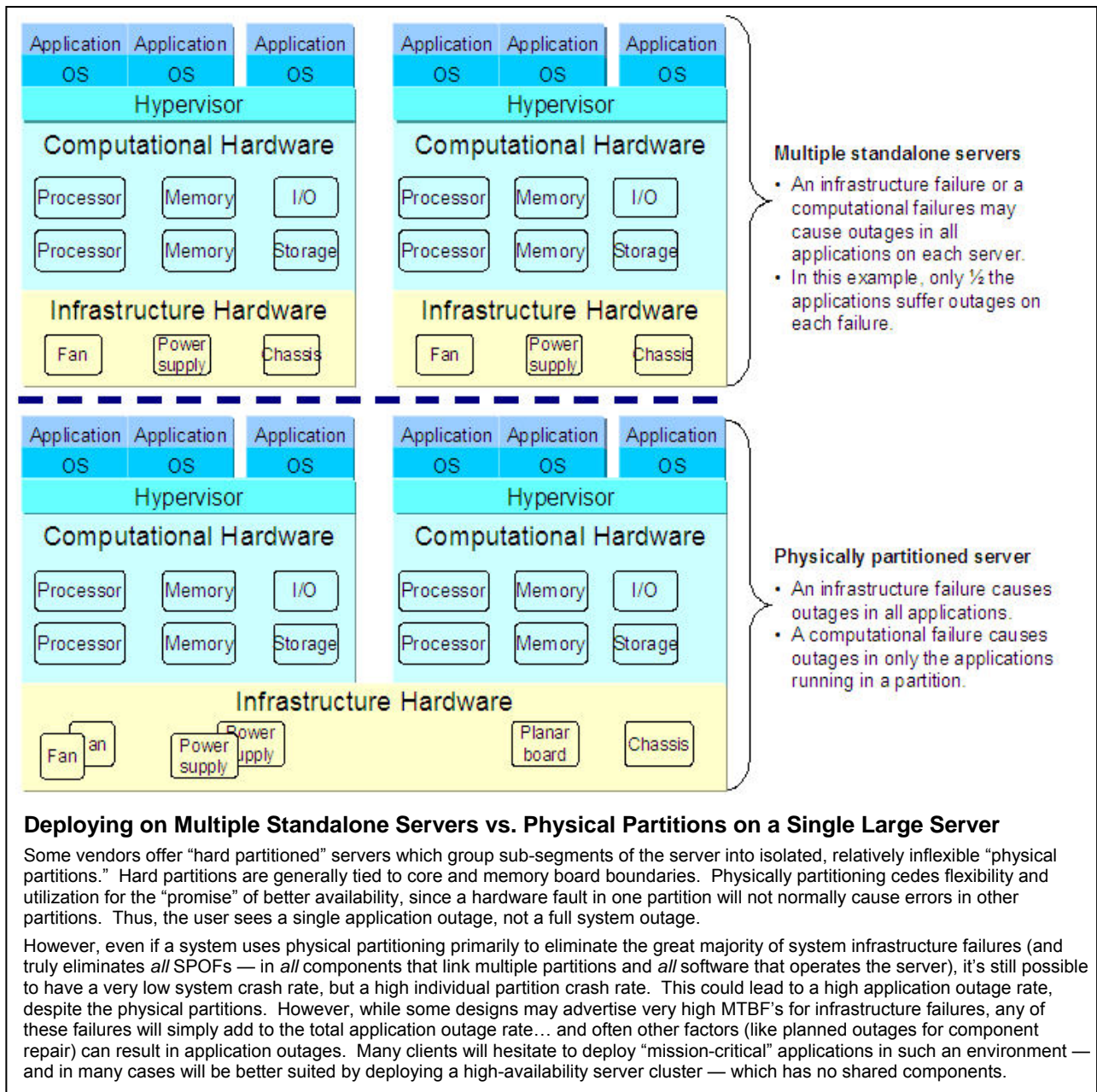
A large enterprise with aggressive requirements for application availability might deploy multiple large standalone systems to provide necessary computational resources, as well as to ensure necessary availability of critical applications.

Alternatively, much as a BladeCenter environment allows multiple blades to run under a single chassis for management purposes, a system design might also allow multiple larger SMP servers to be consolidated under a single chassis sharing a common hardware infrastructure. In such an environment the individual consolidated hardware elements could be considered as “physical” or “hard” partitions, just as the individual partitions running on top of a physical partition could be considered soft partitions.

In either of the two environments, as before, the availability of any individual application is a function of both the computational hardware and infrastructure hardware.

If the computational hardware in a physical partition is the same as that of a standalone server, the application availability due to hardware faults will be the same for both configurations. And, clearly, if there is a failure in the infrastructure of a consolidated server, all the applications will fail. However, an infrastructure error in a partitioned server can have a larger impact on operations since every application in every partition will incur an outage.

To illustrate this, assume that a data center deploys three servers, all with an infrastructure MTBF of 5 years. In this example, one large server is divided into two equal partitions, where each partition delivers performance and capacity equal to two smaller servers. The smaller servers are used to run a single business critical application, while the partitioned servers run one application per partition. In a five year period, each of these servers would be expected to experience a failure. Thus, in this time frame, total number of application failures (2 servers vs. partitioned server) would be exactly the same (the larger server fails once with 2 application failures, together the smaller servers also suffer an application failure each). However, there may be a larger impact to data center operations when the larger server fails and all critical applications suffer outages.



In a consolidated environment, if the argument is made that individual application failing means nothing, and that the only failure of interest is when every application in a complex fails, then only MTBF the infrastructure (and other faults that take down an entire system) matters. At best, the consolidated environment can only approximate the MTBF of two standalone servers; however, since even, say a 300 year MTBF for a consolidated systems infrastructure is not going to approach the mean time between the simultaneous outage of two standalone systems.

Generally application outages do matter, and application availability in a well designed system is primarily the function of the availability of the computational elements including not only how often failures have an impact but also to what extent those computational failures can be isolated to a single application.

Infrastructure Approach for POWER7 Servers

Power Systems do not segregate hardware into multiple physical partitions such that a single consolidated system might approximate the availability that could be achieved by running two, four, or eight separate, standalone systems. Rather it is expected that enterprises derive the best value from their assets by deploying servers in highly virtualized environments — where they can dynamically and flexibly access system resources, taking advantage of server performance by maximizing server utilization — and deriving a high return on their server investments.

The requirements for servers to scale quickly and efficiently, automatically optimize workload performance and flexibly flow resources as application demands change, avoid downtime and save energy, and automate management tasks define the key attributes of servers designed for a Smarter Planet. These requirements also drive a Power RAS design strategy designed to optimize availability regardless of system configuration.

This paper has covered, in some detail, how the unique reliability and availability design of the Power computational components assist in achieving application availability. In addition, the by designing the hardware and firmware together, IBM engineers have been able to include hardware features that support more reliable operations of the system firmware. For example, Active Memory Mirroring for Hypervisor [page 30] improves system availability by eliminating a rare, but significant, potential system SPOF.

The POWER Hypervisor is also critical component to both achieving virtualization goals and enhanced server availability.

PowerVM / POWER Hypervisor Design

The advanced virtualization techniques available with POWER technology require powerful tools allowing a system to be divided into multiple partitions, each running a separate operating system image instance. This is accomplished using firmware known as the POWER Hypervisor. The POWER Hypervisor provides software isolation and security for all partitions. The POWER Hypervisor is a “thin” layer of code, an abstraction level (code interface), between the operating system and the hardware. The POWER Hypervisor is limited in size and complexity when compared to a full operating system implementation, and therefore can be considered better “contained” from a design and quality assurance viewpoint.

The POWER Hypervisor code is stored in discrete flash memory storage areas and is active in all systems, even those containing just a single partition. In many ways, the Power Hypervisor is treated as simply another infrastructure component and, since it runs on system hardware, to some extent its reliability depends on the core system reliability.

Not only does PowerVM enable fine-grained resource allocations, the POWER Hypervisor firmware allows almost any system resource (processor cores and cycles, memory segments, I/O slots) to be dynamically assigned (dedicated or shared) to any partition in any combination. This allows exceptional configuration flexibility and enables many high availability functions.

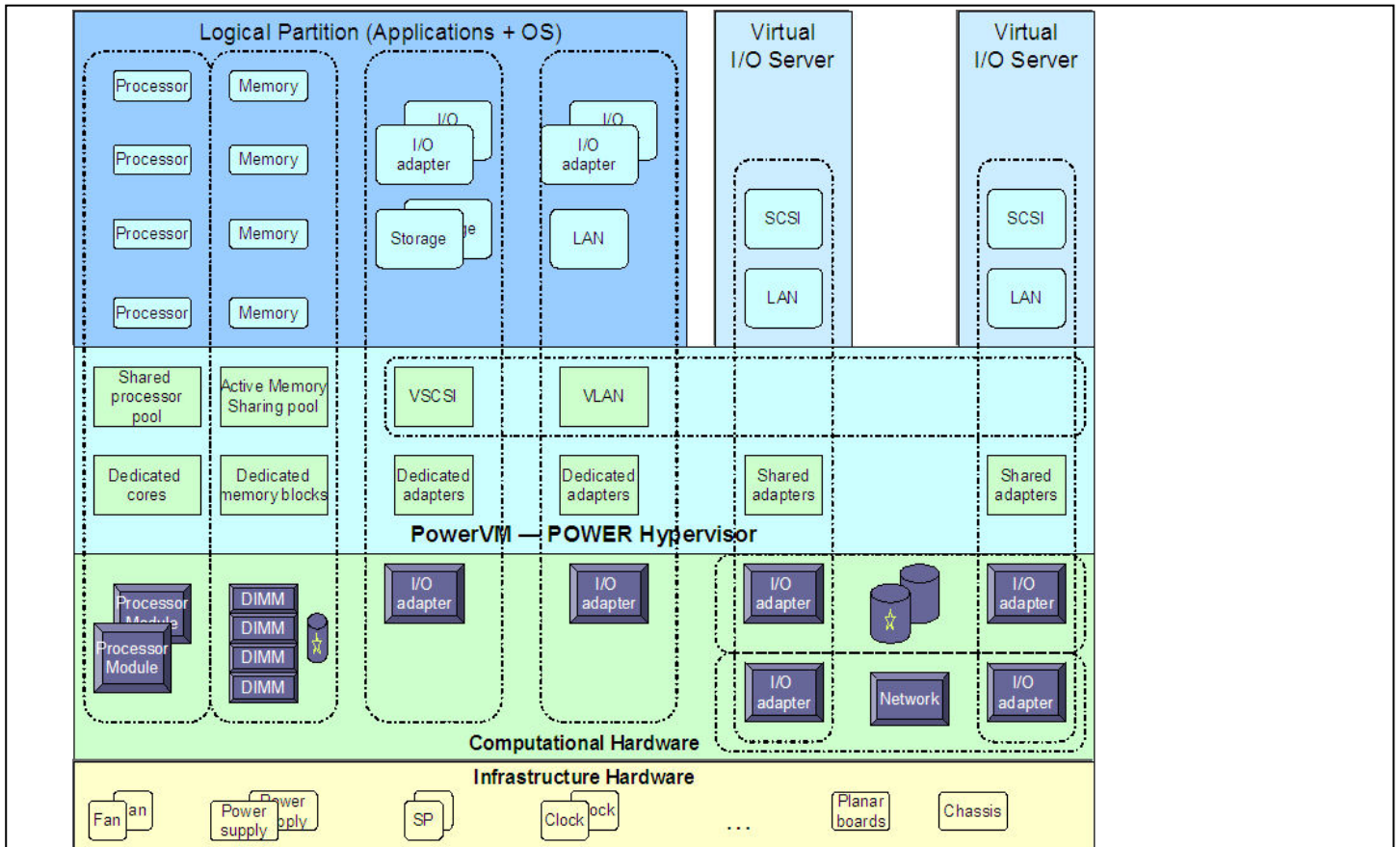
Processor Resource Virtualization

A variety of PowerVM editions are available, providing capabilities such as Micro-Partitioning™, Virtual I/O Server, Multiple Shared Processor Pools, and Shared Dedicated Capacity, which are designed to allow businesses to increase system utilization while helping to ensure applications continue to get the resources they need.

With Micro-Partitioning, PowerVM supports a variety of logical partitions (virtual machines):

1. **Dedicated:** processor cores are assigned are used only by the operating system running on the partition — no other OS can use these cores.

2. Shared: processors cores that assigned to a shared processor pool are shared resources. The POWER Hypervisor allocates processor cycles on a time-slicing basis to the partitions using these cores. A virtual machine receives a portion of the available cycles (in the entire pool) based on policy defined by a Hardware Management Console or Integrated Virtualization Manager. Resource allocation is dynamic and automatic.
3. Shared Dedicated: Shared Dedicated capacity allows for the “donation” of spare CPU cycles from dedicated processor partitions to a Shared Processor Pool. The dedicated partition maintains



A Simplified View of System Virtualization with PowerVM™

This illustration roughly depicts the different layers of hardware and software available on Power 770, 780, or 795 servers supporting partitioning and virtualization of a Power System using the POWER Hypervisor.

In all Power Systems servers, user applications are executed in dynamic logical partitions where each partition is running its own instance of an operating system. Called Micro-Partitioning™ technology, this architecture allows individual cores to run as many as 10 copies of the operating system. Processor cores will support both dedicated processor logical partitions and shared processor dynamic LPARs. In a dedicated processor partition, one or more physical cores are assigned to the partition. In shared processor partitions, a “shared pool” of physical processor cores is defined. This shared pool consists of one or more physical processor cores. Up to 10 logical partitions can be defined for every physical processor core in the pool. This technology is known as PowerVM.

IBM PowerVM is the family of Power technologies, capabilities, and offerings that deliver secure, industry-leading virtualization. It includes base components provided with Power Systems firmware, and optional components — PowerVM Editions — designed to pool resources and optimize their use across application environments and operating systems to deliver the power and flexibility to address multiple system requirements in a single machine. Other editions (Standard and Advanced) offer a variety of capabilities such as Micro-Partitioning™, Virtual I/O Server, Multiple Shared Processor Pools and Shared Dedicated Capacity, plus Live Partition Mobility (LPM), and Active Memory Sharing.

Active Memory Expansion is a new POWER7 technology which allows the effective memory capacity of the system to be much larger than the true physical memory. Innovative compression/decompression of memory content can target memory expansion factors of up to two times. Active Memory Expansion is available for partitions running AIX® 6.1 or later.

absolute priority for dedicated CPU cycles. Enabling this feature may help to increase system utilization without compromising the computing power for critical workloads in a dedicated processor.

Server configurations may also include Multiple Shared Processor Pools. These allow automatic non-disruptive balancing of processing power between partitions assigned to shared pools, resulting in increased throughput. It also provides the ability to cap the processor core resources used by a group of partitions to potentially reduce processor-based software licensing costs.

In any of these cases, if a processor must be deconfigured (removed) for either a hard or series of soft errors, there are mechanisms available to allow running programs to be migrated off the processor core to available spare capacity in the server. This spare capacity can be acquired from anywhere in the system without any need, for example, to dedicate a spare processor core to each partition. Where shared processor pools are utilized, no cooperation from the operating system or application is needed to perform these functions. [See [Dynamic Processor Sparing](#) on page 20].

It should also be noted that in a POWER7 System, each processor core is able to run up to 4 simultaneous threads using Simultaneous Multi-Threading. When multi-threading is enabled, these individual threads are also virtualized.

Memory Resource Virtualization

Assigning memory to virtual machines

Partitions are assigned memory in logical memory blocks of a predefined size (e.g. a 256 MB logical memory block). Once assigned, a logical memory block is dedicated to a partition. No other partition has access to data stored in this logical memory block.

Alternatively, memory can be allocated in a shared memory pool and dynamically assigned to partitions using the Active Memory™ sharing feature. Active Memory sharing enables the sharing of a pool of physical memory among logical partitions on a single server, helping to increase memory utilization and drive down system costs. The POWER Hypervisor manages the relationship between the logical memory blocks and physical memory hardware, including using a Virtual I/O Server to write seldom used memory blocks to DASD storage, enabling assignment of more logical memory blocks. Memory is dynamically allocated among the partitions as needed, to help optimize the overall physical memory usage in the pool.

Active Memory Expansion for the Power 795

Active Memory Expansion is a new POWER7 technology which allows the effective memory capacity of the system to be much larger than the true physical memory. Innovative compression/decompression of memory content can target memory expansion factors of up to two times. This can allow a partition to do significantly more work with the same physical amount of memory or a server to run more partitions and do more work for the same physical amount of memory. Active Memory Expansion is available for partitions running AIX® 6.1 or later.

Mirrored Memory Protects the POWER Hypervisor.

Active Memory Mirroring for Hypervisor [page 30] is designed to prevent a system outage even in the event of an uncorrectable error in memory.

Memory Deconfiguration and Sparing

If a memory fault is detected by the service processor at boot time, the affected memory will be marked as bad and will not be used on this or subsequent IPLs (Memory Persistent Deallocation).

If a partition crashes due to a UE, on the next reboot the POWER Hypervisor will take any unallocated memory (from anywhere in system), including Capacity on Demand (CoD) to allow the partition to restart. If there isn't enough free memory to replace the bad LMBs, the POWER Hypervisor will still start a partition as long as the available memory meets the user specified minimum for the partition.

In either case, an error message will be generated by the POWER Hypervisor, triggering a service call to replace the failed memory.

I/O Adapter Virtualization

Dedicated I/O

The POWER Hypervisor can assign either PCI-x or PCIe I/O adapters to partitions. These adapters are solely used by a single partition and are called dedicated I/O adapters. A wide variety of adapters are available for Power Servers to allow disk, LAN, or storage connections to the server. With operating system support, adapter availability can be enhanced by using redundant adapters a variety of recovery schemes like multi-path I/O with automatic fail-over.

Shared (Virtual) I/O

Power Servers support sharing of I/O devices by defining a special partition called a Virtual I/O Server (VIOS) partition. While I/O adapters (and their attached resources) are dedicated to a VIOS partition, the Virtual I/O Server includes code that allows PCI-x or PCIe adapters to be shared by device drivers running in different virtual machines (logical partitions). VIOS allows for the sharing of disk and optical devices as well as communications and Fiber Channel adapters to help drive down complexity and systems/administrative expenses.

For enhanced availability each Virtual I/O Server can use redundant I/O adapters with dynamic failover capabilities. Dynamic failover is independent of the operating systems and user applications. This configuration can be used to help avoid outages caused by I/O adapter failures.

With operating system support, a partition can access the same I/O subsystem using redundant Virtual I/O Servers. By employing I/O redundancy available in the deployed Virtual I/O Servers, client partitions can maintain continued access to I/O adapters even if there is an outage in a VIOS partition.

Live Partition Mobility

PowerVM supports the movement of a running AIX or Linux partition from one physical server to another compatible server without application downtime, helping to avoid application interruption for planned system maintenance, provisioning, and workload management. Called Live Partition Mobility (LPM), this feature uses many of the inherent features of the Power virtualization architecture (virtual processors, VIOS, dynamic reassignment of resources, etc.) to couple two servers sharing common I/O and storage — allowing dynamic movement of virtual machines between servers without application outage. Live Partition Mobility is supported between POWER6 servers, POWER7 servers, and between POWER6 and POWER7 servers. With LPM, migrating applications between server generations (POWER6 to POWER7) can be non-disruptive to data center operations as easy as simply moving the applications to the new server.

This capability can be used to balance workloads across systems or to save energy by consolidating servers during low demand times and powering down idle servers. In addition, data center managers can better manage planned downtime with the Live Partition Mobility function of PowerVM Enterprise Edition. Workloads can be moved from server to server enabling routine maintenance whenever it is most convenient.

Other Infrastructure Reliability/Availability Design Elements

Power/Cooling

Power 770, Power 780, and Power 795 servers include redundant power supply components, cooling components such as blowers or fans, and voltage regulator module outputs. In addition dual alternating current (AC) power line inputs are supported. Other systems support options for redundant power and cooling [See [Appendix A](#), page 51].

TPMD

All POWER7 systems have at least one Thermal Power Management Device. This TPMD monitors thermal and power utilization of nodes. The TPMD, a critical component of IBM's EnergyScale™ technology⁸, is used by IBM Systems Director Active Energy Manager to allow system administrators to establish policies that balance performance, cooling, and power utilization in a system, including support of various power saving modes.

The intent of the RAS design of the system is to tolerate a complete failure of a TPMD, using a “fail safe” mode allowing continued systems operations. The Power 795 servers include redundant TPMD modules, further reducing the chance for loss of energy management functions due to TPMD failures.

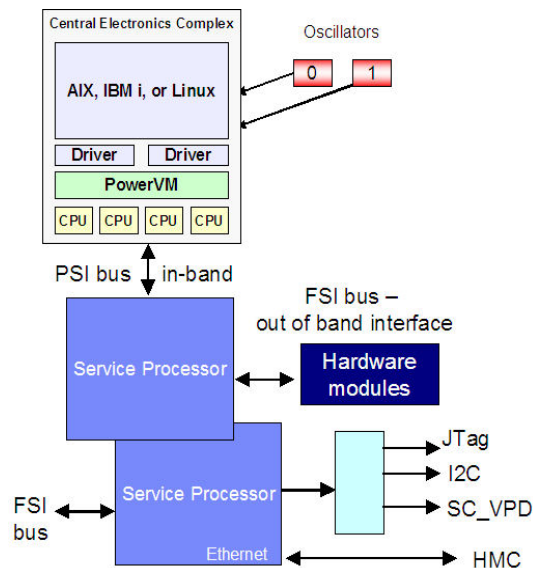
Clocks

Power 795 systems and multi-node Power 770 and 780 systems provide *redundant processor reference clocks* to each processor, supporting dynamic clock fail-over (without causing a system outage) if a fault is detected in the clocking components. Since switching is accomplished within the processor modules, (with no external switching element), the reference clock is considered a computational, rather than infrastructure, element.

Service Processors

POWER7 systems require a single service processor to enable system power-on and to monitor error events during operation. The service processor is a separately powered microprocessor, separate from the main instruction-processing complex. The service processor enables POWER Hypervisor and Hardware Management Console

Clock and Service Processor Connections



The service processor is a separately powered microprocessor, separate from the main instruction-processing complex. The service processor enables POWER Hypervisor and Hardware Management Console surveillance, selected remote power control, environmental monitoring, reset and boot features, remote maintenance and diagnostic activities, including console mirroring.

Each service processor communicates to the POWER Hypervisor using a processor service interface (PSI) connection to a POWER7 processor. This bus is ECC protected, and multiple service processors in a system can communicate to multiple POWER7 processors through redundant PSI busses.

Servers with redundant service processors or clocks support dynamic failover from the primary to backup without disrupting normal system operations.

⁸ Martha Broyles, Chris Francois, Andrew Geissler, Michael Hollinger, Todd Rosedahl, Guillermo Silva, Jeff Van Heuklon, Brian Veale, “IBM EnergyScale for POWER7 Processor-Based Systems”, <http://www-03.ibm.com/systems/power/hardware/whitepapers/energyscale7.html>, POW03039-USEN-03, August 2010.

surveillance, selected remote power control, environmental monitoring, reset and boot features, remote maintenance and diagnostic activities, including console mirroring. On systems without a Hardware Management Console, the service processor can place calls to report surveillance failures with the POWER Hypervisor, critical environmental faults, and critical processing faults even when the main processing unit is inoperable. The service processor provides services common to modern computers such as environmental monitoring, mutual surveillance, and a variety of availability features.

If a service processor experiences a run-time transient fault in either the hardware or firmware, it may be automatically rebooted while the system stays up and active. There will be no server application impact for service processor transient errors. If the service processor encounters a code “hang” condition, the POWER Hypervisor can detect the error and direct the service processor to reboot, avoiding other outage. This is called the *reset/reload* capability.

To support improved availability, separate copies of service processor microcode and the POWER Hypervisor code are stored in discrete flash memory storage areas. Code access is CRC protected. The servers also support dynamic firmware updates, in which applications remain operational while IBM system firmware is updated for many operations. Maintaining two copies allows a new firmware image to be loaded into the service processor while it continues to run with the older image and ensures that the service processor can run even if a flash memory copy becomes corrupted.

Power 770 and 780 systems provide redundant service processors in multi-node systems with the ability to dynamically failover from the primary service processor to a secondary for solid service processor faults.

In a Power 795 server, the service processor function is distributed between system controllers and node controllers. The system controllers, one active and one backup, act as supervisors providing a single point of control and performing the bulk of the traditional service processor functions. The node controllers, one active and one backup per node, provide service access to the node-hardware. All the commands from primary system-controller are routed to both the primary and redundant node-controller. Should a primary node controller fail, the redundant controller will automatically take over all node control responsibilities.

System controllers communicate via redundant LANS connecting the power controllers (in the bulk power supplies), the node controllers, and one or two HMCs. This design allows for automatic failover and continuous server operations should any individual component suffer an error.

Firmware Updates

Given the robust design for managing hardware failures in the service processor it's important that defects in the service processor firmware, POWER Hypervisor, and other system firmware can be fixed without experiencing an outage to patch the firmware.

The firmware structure for the service processor, power subsystem, and POWER Hypervisor all support a *concurrent patch strategy*, allowing administrators to install and activate many firmware updates without cycling power or rebooting the server. Since changes to some server functions (for example, changing initialization values for chip controls) cannot occur during operation, a patch in this area will require a system reboot for activation.

Activating new firmware functions requires installation of a firmware release level. This process is disruptive to server operations and requires a scheduled outage and full server reboot.

Hardware Management Consoles

The Hardware Management Console supports the IBM virtualization strategy and includes a wealth of improvements for service and support including automated install and upgrade, and concurrent maintenance and upgrade for hardware and firmware. The HMC also provides a focal point for service receiving, logging, tracking system errors, and, if enabled, forwarding problem reports to IBM Service and

Support organizations. While the HMC is an optional offering for some configurations, it may be used to support any server⁹ in the IBM Power Server product families.

Hardware Management Consoles are separate appliances connected to a system by an Ethernet network. Power Servers system support connection to two HMCs for redundancy⁹.

Availability and Virtualization beyond the Hardware

The focus of this paper is a discussion of RAS attributes in the Power hardware to provide for availability of the system hardware itself. Operating systems, middleware, and applications provide additional key features concerning their own availability that is outside the scope of this hardware discussion.

It is worthwhile to note, however that hardware and firmware RAS features can provide key enablement for selected software availability features

Operating System and Application Features

There are many error detection/fault isolation and availability features built into operating systems and applications that attempt to detect and diagnose flaws in the code unrelated to faults in the underlying hardware.

While most of these features are hardware independent, some make use of hardware or firmware related features. For example, on partition termination AIX will collect “dump” information related to the problem that caused the termination. The POWER Hypervisor assists in the collection of that information.

Storage Protection Keys

Inadvertent memory overlays – where one set of code inadvertently overwrites memory owned by a different code set — are a common source of application and OS outages.

By design, an operating system may prevent one application from directly over-writing the code or data of another, presuming the application is not sharing memory. However this still leaves open the possibility that one part of an application can overlay memory of an unrelated part of the same application.

Power Storage Protection Keys provide hardware-enforced access mechanisms for memory regions. Only programs that use the correct key are allowed to read or write to protected memory locations. AIX¹⁰ allows each virtual page of memory to be assigned a key that defines, by code segment, the kind of access allowed to the page (e.g. read-only or read-and-write). POWER7 servers allow a direct mapping of virtual keys to hardware keys so that the processor hardware itself is able to guard against unauthorized access.

This new hardware allows programmers to restrict memory access within well-defined, hardware-enforced boundaries, protecting critical portions of AIX¹⁰ and applications software from inadvertent memory overlay. Storage protection keys can reduce the number of intermittent outages associated with undetected memory overlays inside the AIX kernel. Programmers can also use the Power memory protection key feature to increase the reliability of large, complex applications running under AIX V5.3 and later releases.

High Availability Solutions

Throughout the whitepaper, emphasis has been on application availability within the environment of a single system. While application outages due to hardware can be made rare in this environment, they cannot be completely eliminated. It is also true that outages can be caused by failures in the software stack. This paper has covered many availability features that are automatically invoked when needed.

⁹ Except Power blades.

¹⁰ AIX V6.1 and later.

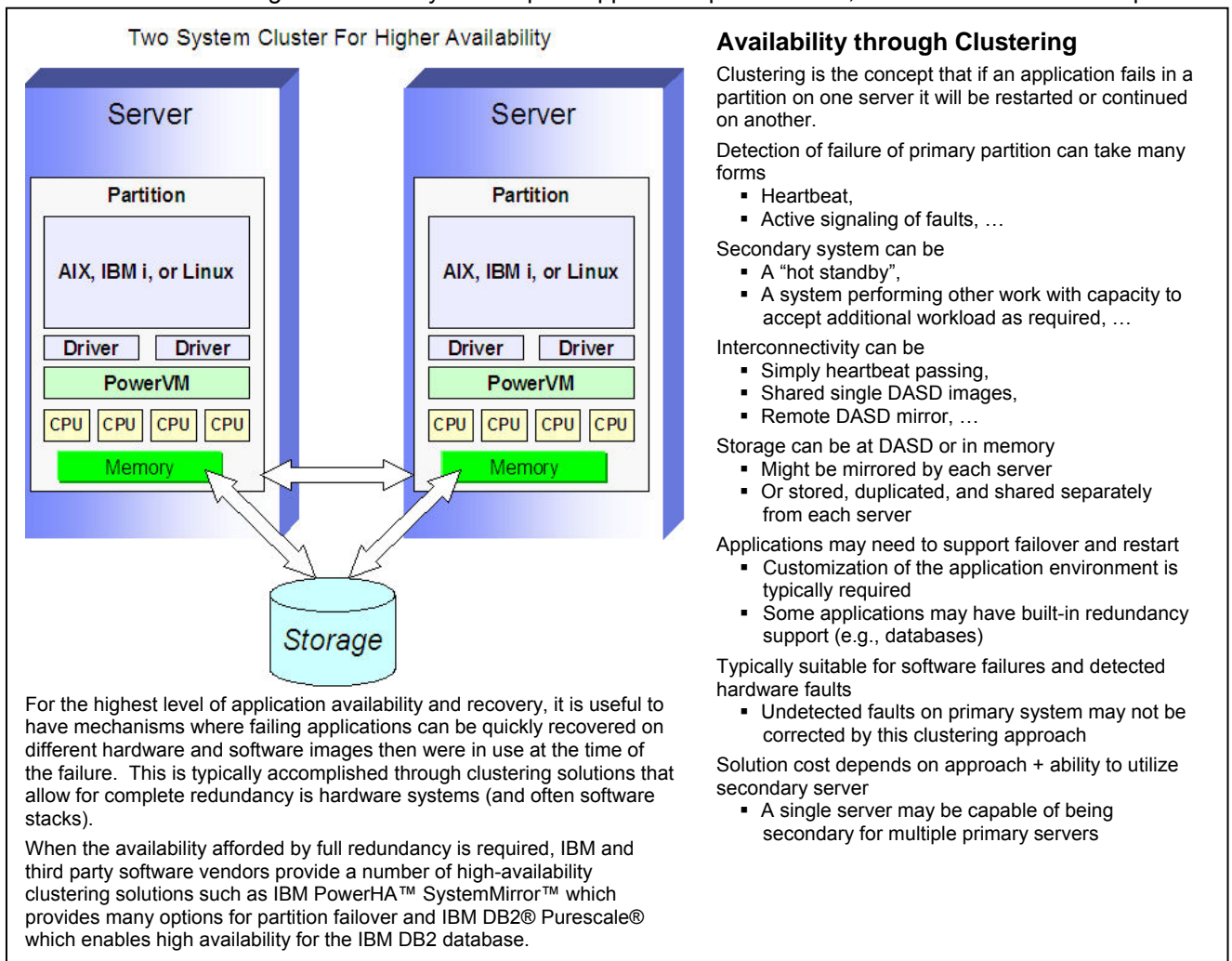
However, proper planning of server configurations can help maximize system availability. Correctly configuring I/O devices for redundancy, and creating partition definitions constructed to survive a loss of core or memory resource can improve overall application availability.

For the highest level of application availability and recovery, it is useful to have mechanisms allowing failing applications to be quickly recovered: ideally on different hardware and software images than those in use at the time of the failure. This is typically accomplished through clustering solutions that allow for complete redundancy in hardware systems (and often software stacks). These designs incorporate some method for fault monitoring so that automated restart of applications can be invoked on system failure.

When the availability afforded by full redundancy is required, IBM and third party software vendors provide a number of high-availability clustering solutions such as IBM PowerHA™ SystemMirror™ which provides many options for partition failover and IBM DB2® Purescale® which enables high availability for the IBM DB2 database.

A wide variety of clustering methods, involving duplicated hardware, firmware, and applications, also exist or has been proposed. Yet, in all cases, single server, enterprise-class reliability is the foundation for a truly reliable computing solution.

1. One obvious reason for this is that deploying complex applications using multi-system redundancy can be an expensive alternative and may not deliver an optimal means for achieving a desired level of availability.
2. Some clustering solutions may deliver poor application performance, and often failover techniques



involve system disruption. Poorly planned configurations may involve critically important delays in fault detection and invoked failover and subsequent application restart may involve a significant amount of time. Indeed, it's possible that a single server design could identify a fault, take an outage, deconfigure the failing component, and restore application availability faster than a clustered solution.

3. Even in an environment offering redundancy, it is important to fully restore the failed hardware after a failover. This can only be done successfully when the proper investment is made in error detection and fault isolation.
4. Experience has shown that failover techniques only work properly if they are adequately tested when deployed and retested on a periodic basis. Often organizations are loathe to cause an outage in production systems to prepare for some future disaster scenario.

It is also important to consider that failover techniques are reliable only when a fault is properly distinguished. Reliable hardware with a high level of error detection and fault isolation is vital to ensuring that faults are identified and isolated so that the failover will correctly recover without propagating incorrect results.

A Smarter Planet — Instrumented, Interconnected, and Intelligent

The mantra for the electronics industry for the past couple of decades has been “smaller, faster, denser, cheaper.” As a result, this year there are estimated to be over a billion transistors per human on our planet... each costing less than one ten-millionth of a US penny. This widely available technology is allowing almost anything and everything to be instrumented — enabling collection of key information about the environment, the packaging and condition of items, usage, thermals, traffic flow, or any other number of factors. Millions of smart devices are being deployed and interconnected. The Internet has fueled a metamorphosis in communications and the ways in which people interact with each other and the environment. Individuals and businesses are creating millions of new digital footprints with their multi-channel transactions, online communities, registrations, and movements — and creating mountains of data. This data holds jewels of intelligence... information that can be mined to gain insights that can make the world better.

Enterprises are handling more information than ever before, but this tremendous explosion in data volume is leading to waste, inaccuracy, and missed opportunities. To address this problem, enterprises are deploying smarter computing models using new flexible infrastructures that couple large, highly scalable, workload-optimized computer systems with open, dynamic software. This transition to “smarter” solutions places a new set of demands on server designers. Today, servers must be able to scale quickly and efficiently while automatically optimizing workload performance and flexibly reallocating resources as application demands change — all while delivering high levels of application availability.

To meet these demands, IBM has designed the POWER7 servers with exceptional capabilities to gracefully handle three modes of computing: massive parallel processing, “throughput” computing, and analytics capabilities. These workload optimized servers use dynamic virtualization capabilities allowing better use of computing capacity (driving high levels of utilization and sharing memory and I/O resources). In Power Systems, hardware is linked with systems software to deliver optimized performance for large numbers of virtual machines supporting a wide variety of applications. In this environment, system reliability and availability become a basic design requirement — servers must be engineered for reliability, availability, and serviceability for not only the infrastructure components, but also the compute elements.

All Power Systems servers use an architecture-based strategy designed to avoid planned and unplanned outages. These servers include a wide variety of features to automatically analyze, identify, and isolate failing components so that repairs can be made as quickly and efficiently as possible. System design engineers incorporate state-of-the-art components and advanced packaging techniques, selecting parts with low intrinsic failure parts rates, and surrounding them with a server package that supports their reliable operation. Care has been taken to deliver rugged and reliable interconnects, and to include features that ease service; like card guides, PCI adapter carriers, cable straps, and “positive retention”

connectors. Should a hardware problem actually occur, these servers have been designed to be fault resilient, to continue to operate despite the error. Every POWER7 server includes advanced availability features like Processor Instruction Retry, Alternate Processor Recovery, Dynamic Processor Deallocation, PCI bus error recovery, Chipkill memory, L2 and L3 cache line delete, dynamic firmware update, redundant hot-plug cooling fans, and hot-plug N+1 power, and power cords (optional in some configurations).

Many of these functions rely on IBM First Failure Data Capture technology, which allows the server to efficiently, capture, diagnose, and respond to hardware errors – the first time that they occur.

These availability techniques are backed by service offerings such as automated install, upgrade, and maintenance that can be employed by IBM servicers or IBM clients (for selected models), allowing servicers from either organization to install new systems or features and to effectively diagnose and repair faults on these systems. At the heart every POWER7 server is the POWER Hypervisor which not only provides fine-grained allocation of system resources supporting advanced virtualization capabilities, but also delivers many availability improvements. The POWER Hypervisor enables: resource sparing (CPU and memory pools), automatic redistribution of capacity on N+1, redundant I/O across LPAR configurations, the ability to reconfigure a system “on the fly,” automated scale-up of high availability backup servers, serialized sharing of devices, sharing of I/O devices through I/O server partitions, and moving “live” partitions from one Power server to another.

Borrowing heavily from predecessor system designs and incorporating many advanced techniques pioneered in IBM mainframes, Power Systems are designed to deliver leading-edge reliability, availability, and serviceability.

Appendix A: System Support for Selected RAS Features [● = available, ○ = optional]

RAS Feature	BladeCenter PS700, 701, 702 Express	Power 710 and 730 Express	Power 720 and 740 Express	Power 750 Express and 755	Power 770 and 780	Power 795
Processor						
Processor fabric bus protection	●	●	●	●	●	●
Dynamic Processor Deallocation	●	●	●	●	●	●
Dynamic Processor Sparing						
◆ Using CoD cores					○	○
◆ Using capacity from spare pool	○	○	○	○	○	○
Core Error Recovery						
Processor Instruction Retry	●	●	●	●	●	●
◆ Alternate Processor Recovery	●	●	●	●	●	●
◆ Partition core contained checkstop	●	●	●	● ¹¹	●	●
Persistent processor deallocation	●	●	●	●	●	●
I/O subsystem						
GX+ bus persistent deallocation ¹²		○	○	○	○	○
Optional ECC I/O hub with freeze behavior		○	○	○	○	○
PCI bus enhanced error detection	●	●	●	●	●	●
PCI bus enhanced error recovery	●	●	●	●	●	●
PCI-PCI bridge enhanced error handling		●	●	●	●	●
Redundant 12x Channel link			○ ¹³	○ ¹⁴	○	○
Clocks and service processor						
Dynamic SP failover at run-time					● ¹⁵	● ¹⁶
Clock failover at runtime					● ¹⁵	●
Memory availability						
ECC memory, L2, L3 cache	●	●	●	●	●	●
Error detection / correction						
◆ Chipkill memory plus additional ½ symbol correct	●	●	●	●	●	●
◆ Memory DRAM sparing					●	●
Memory sparing with CoD at IPL time					○	○
CRC plus retry on memory data bus (CPU to buffer)	●	●	●	● ¹¹	●	●
Data bus (memory buffer to DRAM) ECC plus retry	●	●	●	●	●	●
Dynamic memory channel repair	●	●	●	●	●	●
Processor memory controller memory scrubbing	●	●	●	●	●	●
Memory page deallocation	●	●	●	●	●	●
L1 parity check plus retry / set delete	●	●	●	●	●	●
L2 cache line delete	●	●	●	●	●	●
L3 cache line delete	●	●	●	●	●	●
Special Uncorrectable Error handling	●	●	●	●	●	●
Active Memory Mirroring for Hypervisor						○ ¹⁷
Fault detection and isolation						
FFDC for fault detection and isolation	●	●	●	●	●	●
Storage Protection Keys	●	●	●	●	●	●
Error log analysis	●	●	●	●	●	●

¹¹ Availability may depend on system processors and firmware level

¹² Two or more GX busses

¹³ Power 740 only

¹⁴ Only Power 750 with at least 2 sockets

¹⁵ Requires two or more nodes

¹⁶ Also: redundant node controllers w/dynamic failover at runtime

RAS Feature	BladeCenter PS700, 701, 702	Power 710 and 730	Power 720 and 740	Power 750 and 755	Power 770 and 780	Power 795
Serviceability						
Boot-time progress indicators	●	●	●	●	●	●
Firmware error codes	●	●	●	●	●	●
Operating system error codes	●	●	●	●	●	●
Inventory collection	●	●	●	●	●	●
Environmental and power warnings	●	●	●	●	●	●
PCI card hot-swap			○ ¹⁸	○ ¹⁹	● ¹⁹	○ ¹⁸
Hot-swap DASD / media	● ²⁰	●	●	●	●	● ²¹
Dual disk controllers / split backplane			○	○	●	○ ¹⁸
Extended error data collection	●	●	●	●		●
SP "call home" on non-HMC configurations		●	●	●	N/A ²²	N/A
I/O drawer redundant connections			●	●	●	●
I/O drawer hot add and concurrent repair			●	●	●	●
Hot GX adapter add and repair				●	●	● ²³
Concurrent add of powered I/O rack			●	●	●	●
SP mutual surveillance w/ POWER Hypervisor	●	●	●	●	●	●
Dynamic firmware update with HMC		○	○	○	●	●
Service Agent Call Home application	●	●	●	●	●	●
Service indicators – guiding light or light path LEDs	Light path	Light path	Light path	Light path	Guiding	Guiding
Service processor support for BIST for logic/ arrays, wire tests, component initialization	●	●	●	●	●	●
System dump for memory, POWER Hypervisor, SP	●	●	●	●	●	●
InfoCenter / Systems Support Site service publications	●	●	●	●	●	●
Repair and Verify Guided Maintenance	N/A	○	○	○	●	●
Operating system error reporting to HMC SFP app.	N/A	○	○	○	●	●
RMC secure error transmission subsystem	N/A	○	○	○	●	●
Health check scheduled operations with HMC	N/A	○	○	○	●	●
Operator panel (real or virtual)	●	●	●	●	●	●
Concurrent operator panel maintenance	N/A				●	N/A
Redundant HMCs	N/A	○	○	○	○	○
Automated server recovery/restart	●	●	●	●	●	●
Hot-node add/ cold node repair	Hot Swap Blades				● ²⁴	● ²³
Hot-node repair / hot memory upgrade	Hot Swap Blades				● ²⁴	● ²³
PowerVM Live Partition / Live Application Mobility	○	○	○	○	○	○
Power and cooling						
Redundant, hot swap fans and blowers for CEC	● ²⁵	●	●	●	●	●
Redundant, hot swap power supplies for CEC	● ²⁵	710 ○ 730 ●	720 ○ 740 ●	●	●	●
Redundant voltage regulator outputs				●	●	●
TPMD for system power and thermal management	●	●	●	●	●	●
CEC power/thermal sensors (CPU and memory)	●	●	●	●	●	●
Redundant power for I/O drawers			● ¹⁸	● ¹⁸	● ¹⁸	● ¹⁸

¹⁷ Refer to IBM United States Hardware Announcement 110-158, dated August 17, 2010 for availability dates

¹⁸ Supported in attached I/O drawers only

¹⁹ Supported in base unit and I/O drawers (Note: I/O drawers not supported on Power 755)

²⁰ In BladeCenter S Chassis

²¹ Capability in I/O drawers

²² NA = Not Applicable

²³ Refer to IBM United States Hardware Announcement 110-158, dated August 17, 2010 for availability dates

²⁴ Refer to IBM United States Hardware Announcement 110-025, dated February 9, 2010 for availability dates

²⁵ In BladeCenter Chassis

Appendix B: Operating System Support for Selected RAS Features²⁶ [● = available, ○ = optional]

RAS Feature	AIX V5.3	AIX V6	AIX V7	IBM i V6	IBM i V7	RHEL 5.5 ^{27,28}	SLES ²⁹ 10	SLES ³⁰ 11
Processor								
Processor fabric bus protection	●	●	●	●	●	●	●	●
Dynamic Processor Deallocation	●	●	●	●	●	●	●	●
Dynamic Processor Sparing	●	●	●	●	●	●	●	●
◆ Using CoD cores	●	●	●	●	●	●	●	●
◆ Using capacity from spare pool	●	●	●	●	●	●	●	●
Core Error Recovery								
Processor Instruction Retry	●	●	●	●	●	●	●	●
◆ Alternate Processor Recovery	●	●	●	●	●	●	●	●
◆ Partition core contained checkstop	●	●	●	●	●	●	●	●
Persistent processor deallocation	●	●	●	●	●	●	●	●
I/O subsystem								
GX+ bus persistent deallocation	●	●	●	●	●			
Optional I/O hub with freeze behavior	●	●	●	●	●	●	●	●
PCI bus enhanced error detection	●	●	●	●	●	●	●	●
PCI bus enhanced error recovery	●	●	●	●	●	Limited	Limited	Limited
PCI-PCI bridge enhanced error handling	●	●	●	●	●			
Redundant 12x channel link	●	●	●	●	●	●	●	●
Clocks and service processor								
Dynamic SP failover at run-time	●	●	●	●	●	●	●	●
Redundant SP and node controllers w/dynamic failover at runtime	●	●	●	●	●	●	●	●
Clock failover at runtime	●	●	●	●	●	●	●	●
Memory availability								
ECC memory, L2, L3 cache	●	●	●	●	●	●	●	●
Error detection / correction	●	●	●	●	●	●	●	●
◆ Chipkill memory plus additional ½ symbol correct	●	●	●	●	●	●	●	●
◆ Memory DRAM sparing	●	●	●	●	●	●	●	●
Memory sparing with CoD at IPL time	●	●	●	●	●	●	●	●
CRC plus retry on memory data bus (CPU to buffer)	●	●	●	●	●	●	●	●
Data bus (memory buffer to DRAM) ECC plus retry	●	●	●	●	●	●	●	●
Dynamic memory channel repair	●	●	●	●	●	●	●	●
Processor memory controller memory scrubbing	●	●	●	●	●	●	●	●
Memory page deallocation	●	●	●	●	●	●	●	●
L1 parity check plus retry / set delete	●	●	●	●	●	●	●	●
L2 cache line delete	●	●	●	●	●	●	●	●
L3 cache line delete	●	●	●	●	●	●	●	●
Special Uncorrectable Error handling	●	●	●	● ³¹	● ³¹	●	●	●
Active Memory Mirroring for Hypervisor	●	●	●	●	●	●	●	●
Fault detection and isolation								
FFDC for fault detection/error isolation	●	●	●	●	●	●	●	●
Storage Protection Keys	●	●	●	●	●	●	●	●
Error log analysis	●	●	●	●	●	●	●	●

²⁶ See the Power Systems Facts and Features (<http://www-03.ibm.com/systems/power/hardware/reports/factsfeatures.html>) document for POWER7 servers (POB03022-USEN-05) for more detailed information.

²⁷ RHEL 5.5 = Red Hat Enterprise Linux 5.5

²⁸ IBM Statement of Direction to support POWER7 based Power Systems servers and blades in Red Hat's upcoming version, Red Hat Enterprise Linux 6. For additional questions on the availability of this release, please contact Red Hat.

²⁹ SLES 10 = Novell SUSE LINUX Enterprise Server 10

³⁰ SLES 11 = Novell SUSE LINUX Enterprise Server 11

³¹ Partition checkstop

RAS Feature	AIX V5.3	AIX V6	AIX V7	IBM i V6	IBM i V7	RHEL 5.5	SLES 10	SLES 11
Serviceability								
Boot-time progress indicators	●	●	●	●	●	Limited	Limited	Limited
Firmware error codes	●	●	●	●	●	●	●	●
Operating system error codes	●	●	●	●	●	Limited	Limited	Limited
Inventory collection	●	●	●	●	●	●	●	●
Environmental and power warnings	●	●	●	●	●	● ³²	● ³²	● ³²
Hot-swap DASD / media / PCI adapters	●	●	●	●	●	●	●	●
PCI card hot-swap	●	●	●	●	●	●	●	●
Dual disk controllers / split backplane	●	●	●	●	●	●	●	●
Extended error data collection	●	●	●	●	●	●	●	●
SP "call home" on non-HMC configurations	●	●	●	●	●	●	●	●
I/O drawer redundant connections	●	●	●	●	●	●	●	●
I/O drawer hot add and concurrent repair	●	●	●	●	●	●	●	●
Hot GX adapter add and repair	●	●	●	●	●	●	●	●
Concurrent add of powered I/O rack	●	●	●	●	●	●	●	●
SP mutual surveillance w/ POWER Hypervisor	●	●	●	●	●	●	●	●
Dynamic firmware update with HMC	●	●	●	●	●	●	●	●
Service Agent Call Home Application	●	●	●	●	●	●	●	●
Service indicators – guiding light or light path LEDs	●	●	●	●	●	● ³²	● ³²	● ³²
Service processor support for BIST for logic/ arrays, wire tests, component initialization	●	●	●	●	●	●	●	●
System dump for memory, POWER Hypervisor, SP	●	●	●	●	●	● ³²	● ³²	● ³²
InfoCenter / Systems Support Site service publications	●	●	●	●	●	●	●	●
Repair and verify guided maintenance	●	●	●	●	●	Limited	Limited	Limited
System Support Site education	●	●	●	●	●	●	●	●
Operating system error reporting to HMC SFP app.	●	●	●	●	●	● ³²	● ³²	● ³²
RMC secure error transmission subsystem	●	●	●	●	●	●	●	●
Health check scheduled operations with HMC	●	●	●	●	●	●	●	●
Operator panel (real or virtual)	●	●	●	●	●	●	●	●
Concurrent operator panel maintenance	●	●	●	●	●	●	●	●
Redundant HMCs	●	●	●	●	●	●	●	●
Automated server recovery/restart	●	●	●	●	●	●	●	●
High availability clustering support	●	●	●	●	●	●	●	●
Concurrent kernel update	●	●	●	●	●	●	●	●
Hot-node add/ cold node repair	●	●	●	●	●	●	●	●
Hot-node repair / hot memory upgrade	●	●	●	●	●	●	●	●
PowerVM Live Partition / Live Application Mobility	○	○	○			○	○	○
Power and cooling								
Redundant, hot swap fans and blowers for CEC	●	●	●	●	●	●	●	●
Redundant, hot swap power supplies for CEC	●	●	●	●	●	●	●	●
Redundant voltage regulator outputs	●	●	●	●	●	●	●	●
TPMD for system power and thermal management	●	●	●	●	●	●	●	●
CEC power/thermal sensors (CPU and memory)	●	●	●	●	●	●	●	●
Redundant power for I/O drawers	●	●	●	●	●	●	●	●

³² All systems except blades

About the authors:

Jim Mitchell is an IBM Senior Engineer and Certified IT Infrastructure Consultant. He has worked in microprocessor design and has managed an operating system development team. An IBM patent holder, Jim has published numerous articles on floating-point processor design, system simulation and modeling, and server system architectures. Jim is currently assigned to the staff of the Austin Executive Briefing Center.

Daniel Henderson is an IBM Senior Technical Staff Member. He has been a part of the design team in Austin since the earliest days of RISC based products and is currently the lead availability system designer for IBM Power Systems platforms.

George Ahrens is an IBM Senior Technical Staff Member. He is responsible for the Power Systems service strategy and architecture. He has published multiple articles on RAS modeling as well as several whitepapers on RAS design and availability best practices. He holds numerous patents dealing with RAS capabilities and design on partitioned servers. George currently leads a group of service architects responsible for defining the service strategy and architecture for IBM Systems and Technology Group products.

Information concerning non-IBM products was obtained from the suppliers of these products or other public sources. Questions on the capabilities of the non-IBM products should be addressed with the suppliers.

IBM hardware products are manufactured from new parts, or new and used parts. In some cases, the hardware product may not be new and may have been previously installed. Regardless, our warranty terms apply.

Photographs show engineering and design models. Changes may be incorporated in production models.

Copying or downloading the images contained in this document is expressly prohibited without the written consent of IBM. This equipment is subject to FCC rules. It will comply with the appropriate FCC rules before final delivery to the buyer.

Information concerning non-IBM products was obtained from the suppliers of these products. Questions on the capabilities of the non-IBM products should be addressed with the suppliers.

Many of the IBM Power Systems features described in this document are operating system-dependent and may not be available on Linux. For more information, please, visit <http://www-03.ibm.com/systems/power/software/linux/index.html>

All performance information was determined in a controlled environment. Actual results may vary. Performance information is provided "AS IS" and no warranties or guarantees are expressed or implied by IBM

For details on model specific features, refer to the IBM POWER7 Systems Family Quick Reference Guide <http://www-03.ibm.com/systems/power/hardware/>.

All statements regarding IBM's future direction and intent are subject to change or withdrawal without notice and represent goals and objectives only



© IBM Corporation 2010
IBM Corporation
Systems and Technology Group
Route 100
Somers, New York 10589

Produced in the United States of America
August 2010
All Rights Reserved

This document was developed for products and/or services offered in the United States. IBM may not offer the products, features, or services discussed in this document in other countries. The information may be subject to change without notice. Consult your local IBM business contact for information on the products, features, and services available in your area.

The Power Architecture and Power.org wordmarks and the Power and Power.org logos and related marks are trademarks and service marks licensed by Power.org.

IBM, the IBM logo, the e-business logo, Active Memory, AIX, AIX 5L, AIX 6, TurboCore, PowerHA, PowerHA SystemMirror, DB2, DB2 PureScale, Micro-Partitioning, Power, POWER, POWER4, POWER5, POWER5+, POWER6, POWER7, Power Systems, pSeries, PowerVM, Power Systems Software, Smarter Planet, System i, Systems Director, TotaStorage, Energy Scale, and Virtualization Engine are trademarks or registered trademarks of International Business Machines Corporation in the United States or other countries or both. A full list of U.S. trademarks owned by IBM may be found at: <http://www.ibm.com/legal/copytrade.shtml>.

UNIX is a registered trademark of The Open Group in the United States, other countries or both. Linux is a trademark of Linus Torvalds in the United States, other countries or both.

Other company, product, and service names may be trademarks or service marks of others.

The Power Systems page can be found at: <http://www-03.ibm.com/systems/power/>

The IBM Systems Software home page on the Internet can be found at: <http://www-03.ibm.com/systems/software/>

POW03056-USEN-03