IBM S/390 Parallel Enterprise Server G5 fault tolerance: A historical perspective

by L. Spainhower
T. A. Gregg

Fault tolerance in IBM S/390® systems during the 1980s and 1990s had three distinct phases, each characterized by a different uptime improvement rate. Early TCM-technology mainframes delivered excellent data integrity, instantaneous error detection, and positive fault isolation, but had limited on-line repair. Later TCM mainframes introduced capabilities for providing a high degree of transparent recovery, failure masking, and on-line repair. New challenges accompanied the introduction of CMOS technology. A significant reduction in parts count greatly improved intrinsic failure rates, but dense packaging disallowed on-line CPU repair. In addition, characteristics of the microprocessor technology posed difficulties for traditional in-line error checking. As a result, system fault-tolerant design, particularly in CPUs and memory, underwent another evolution from G1 to G5. G5 implements an innovative design for a high-performance, fault-tolerant single-chip microprocessor. Dynamic CPU sparing delivers a transparent concurrent repair mechanism. A new internal channel provides a high-performance, highly available Parallel Sysplex® in a single mainframe. G5 is both the culmination of decades of innovation and careful implementation, and the highest achievement of S/390 fault-tolerant design.

1. Introduction

Fault-tolerant design in the IBM mainframe computers is motivated by two distinct forces: enterprise application programs and electronics technology. Even before the introduction of S/360 in 1964, IBM mainframe systems were used for mission-critical business applications. Systems were required to be available during normal business hours, to be able to be quickly repaired when a failure occurred, and to preserve data integrity. At the same time, they were large and complex, the intrinsic

©**Copyright** 1999 by International Business Machines Corporation. Copying in printed form for private use is permitted without payment of royalty provided that (1) each reproduction is done without alteration and (2) the *Journal* reference and IBM copyright notice are included on the first page. The title and abstract, but no other portions, of this paper may be copied or distributed royalty free without further permission by computer-based and other information-service systems. Permission to *republish* any other portion of this paper must be obtained from the Editor.

0018-8646/99/$5.00 © 1999 IBM

failure rates of their componentry were high, and transient or intermittent failures were frequent. The evolution of fault-tolerant techniques in IBM mainframes prior to the era of the thermal conduction model (TCM) is well-documented [1, 2]. The earliest commercial mainframes had error checking and checkpoint restart. Thirty years ago, some of the basic fault-tolerant techniques employed in G5 today, e.g., error-correction codes and CPU instruction retry, were first invented and implemented. Over time, in response to new failure mechanisms and an ever-increasing need for application uptime, S/390* error detection and recovery have become increasingly comprehensive, and new fault-tolerant techniques have been invented.

The remainder of this paper is organized in the following fashion: Section 2 contains a brief description of failure modes, an overview of fault-tolerant functionality during the TCM era, and a review of CMOS progress. Section 3 is a detailed description of the G5 microprocessor fault-tolerant design, discussing its unique industry position and the technology challenges posed by CMOS technology. Sections 4, 5, and 6 respectively are overviews of the G5 fault-tolerant design of the memory hierarchy, the I/O subsystem, and the power subsystem. Brief conclusions are presented in Section 7.

2. Background

From S/360 on, preserving data integrity has been a fundamental design principle. Early on, the prudent policy was to terminate operation upon error detection. Over time, a second design principle, transparent recovery, has gained equal importance. The technology failure model has remained fairly constant, but the implemented fault-tolerant designs have continually changed. Sometimes the changes have been gradual and evolutionary; sometimes, as with the transformation to CMOS, the changes have been more radical.

- *S/390 failure model*

Failure modes include permanent circuit faults, intermittent faults, and transient faults. Permanent or hard faults occur when a circuit no longer yields a correct output, given a specific set of inputs. Every time the specific input is repeated, the incorrect output is produced. An intermittent fault occurs when a specific event produces an incorrect result, but the same inputs at a different time may produce the correct result. An intermittent fault can occur as a result of design error or marginal circuits. Transient faults are random events which occur when environmental conditions, noise, or cosmic particles cause an incorrect result, but the circuitry itself functions correctly.

It is necessary to determine whether an error is caused by a permanent physical failure requiring repair or a transient failure that can be recovered without physical parts replacement. The logic structure supports the capability to back out and retry internal operations with appropriate thresholds for determining success. The vast majority of modern hardware failures—TCM and CMOS—are transient, and retry effectively negates their effects. It is possible, of course, that a permanent failure may be successfully retried because the machine is run in a non-overlapped state during the retry operation, thereby using different circuit combinations. Intermittent faults, on the other hand, could cause errors several times, only to disappear later. S/390 systems are designed to handle transient and permanent failures and are not specifically designed to handle intermittents. A very infrequent event will be retried and recovered; if the error occurs frequently and exceeds a certain threshold, it is considered permanent.

- *TCM overview*

During the era of TCM mainframe development from 1981 to 1993, each major system generation (308X, 3090*, 9020) was characterized by improved circuit failure rates. However, the enhanced functionality, scalability, and performance of each new generation required circuit growth rates that exceeded the reliability gains. Thus, just to deliver equivalent fault tolerance in the new product, additional system-level design was required. For two reasons, simple equivalence was inadequate. First, the uptime requirements and expected availability of mainframes increased steadily. Especially after the introduction of logical partitioning (LPAR) in 1988, permitting multiple concurrent instances of an operating system, unexpected downtime became less tolerable. Second, there was in place a corporate directive to improve the fault-tolerant performance of each generation over that of its predecessors. For these reasons, system designers were further pressured to invent and implement new detection, isolation, and recovery capabilities.

For TCM circuit-level fault tolerance, design engineers followed rules and guidelines to ensure consistent circuit-level error checking and isolation throughout the computer, regardless of the particular logic function. The goal was to detect and contain all failures of the hardware. Every register and latch was protected by error checking. Arrays, data paths, and control paths were protected either by parity or error-correcting code (ECC). Checking mechanisms employed for state machines, ALUs, and other control logic included illegal-state detection, parity prediction, and pinpoint explicit redundancy. Designers conducted lengthy design reviews to evaluate and improve coverage. Especially for control logic, circuit overhead was typically high, designs were complex, and effectiveness was difficult to verify.

Thermal conduction modules (TCMs) are water-cooled glass-ceramic substrates with a maximum of 100 or

121 emitter-coupled logic (ECL) chips. Prior to the introduction of the TCM, second-level packaging was implemented using card-on-board. Service personnel were provided with detailed logic diagrams and could diagnose failures by tracing signals back from an incorrect state while executing handwritten program loops or packaged diagnostic programs. TCM encapsulation eliminated the ability to employ oscilloscopes and logic analyzers to monitor low-level circuit behavior. As a result, the 308X fault-tolerant design effort concentrated on careful checker placement and first-failure data capture [3]. Emphasis was placed on identifying a single field-replaceable unit (FRU) when a failure occurred. Pages (two- or four-kilobyte sections) of memory could be deconfigured, and I/O channels could be varied off-line while program execution continued, but all system repair required all system resources. The 3090 design continued to improve failure isolation, with an additional focus on failure-rate reduction and increased degradation capabilities, specifically of individual CPUs [4]. Failure-rate models were developed which used individual component failure rates to build entire mainframes, factoring in capabilities such as ECC and instruction retry. Projections of anticipated failures shifted from a service cost to a customer impact perspective. Also, the relatively high failure rate of the processor controller led to a duplex design with concurrent maintenance capability.

In the late 1980s, improvement in application availability was required at a faster rate than the TCM mean time to failure (MTTF) learning curve allowed. Also, in a shift from earlier practices, the initial 9020 model scheduled for shipment was the highest-capacity shared multiprocessor (SMP). It was capable of twice the throughput of the largest 3090 predecessor and, using LPAR, was often employed as a consolidation vehicle. When previously isolated workloads, with different peak demands, are placed on the same physical mainframe, its uptime requirements increase greatly. Thus, a significant fault-tolerant design initiative was undertaken for 9020. Wherever possible, the inherent redundancy of the mainframe—multiple channels and CPUs, for instance—was exploited to allow continued operation subsequent to a failure in functional logic. The packaging of the logical elements permitted concurrent maintenance. Support subsystems such as power and cooling were outfitted with explicit redundancy [5, 6]. The 9020 is the high point of TCM fault-tolerant design, and **Figure 1** demonstrates the discontinuity resulting from this major design effort. Full field data for 1995 for a ten-way 9020, 9X2, shows an MTTF, with failure defined as an unplanned outage, of more than 12 years. Of all repairs, 71% were concurrent, 18% were deferred to a time selected by the customer, and only 11% were unplanned.

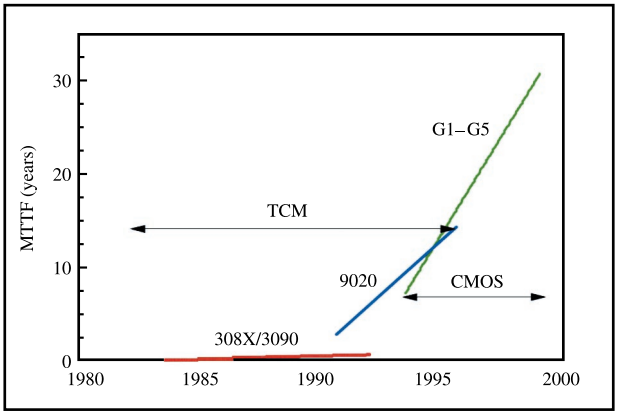

Figure 1

Trends in mean time to failure (where failure = hardware-caused system crash).

- *CMOS overview*

The large-scale integration and reliability of CMOS technology greatly improve the fault-avoidance characteristics of S/390 mainframes over those of predecessor TCM machines. Better intrinsic failure rates coupled with huge reductions in parts count, as shown in **Figure 2**, significantly increase MTTF. In addition, since one FRU contains multiple CPUs, I/O engines, and memory controllers, error checker placement can typically be much coarser than was acceptable for TCM technology. Reduced CMOS power requirements permit fault-tolerance improvements for power and cooling, including bulk power and battery backup.

When the S/390 transition to CMOS occurred in 1994, the G1 CPU logic did not include 9020-equivalent circuit-level detection and recovery. Instantaneous checking coverage was roughly equivalent to that of the 3090 generation, but less than the 98% targeted for 9020. There was no instruction-retry capability, although most large CPU arrays performed refetch when a parity check occurred, permitting transient fault recovery. On G1, some memory-array chip failure mechanisms could cause the system to fail, and there was no dynamic-memory chip sparing. Several factors allowed a different fault-tolerant design point. First, and foremost, was the intrinsic failure-rate reduction. The MTTF was predicted to be similar to that of 9020. Second, the capacity of the largest G1 mainframe was approximately half that of its TCM predecessor, so it was not a growth and consolidation vehicle, with the accompanying increased uptime demands. Third, the I/O subsystem utilized robust S/390 architecture, and redundancy was implemented throughout the support subsystems—power, cooling, and support

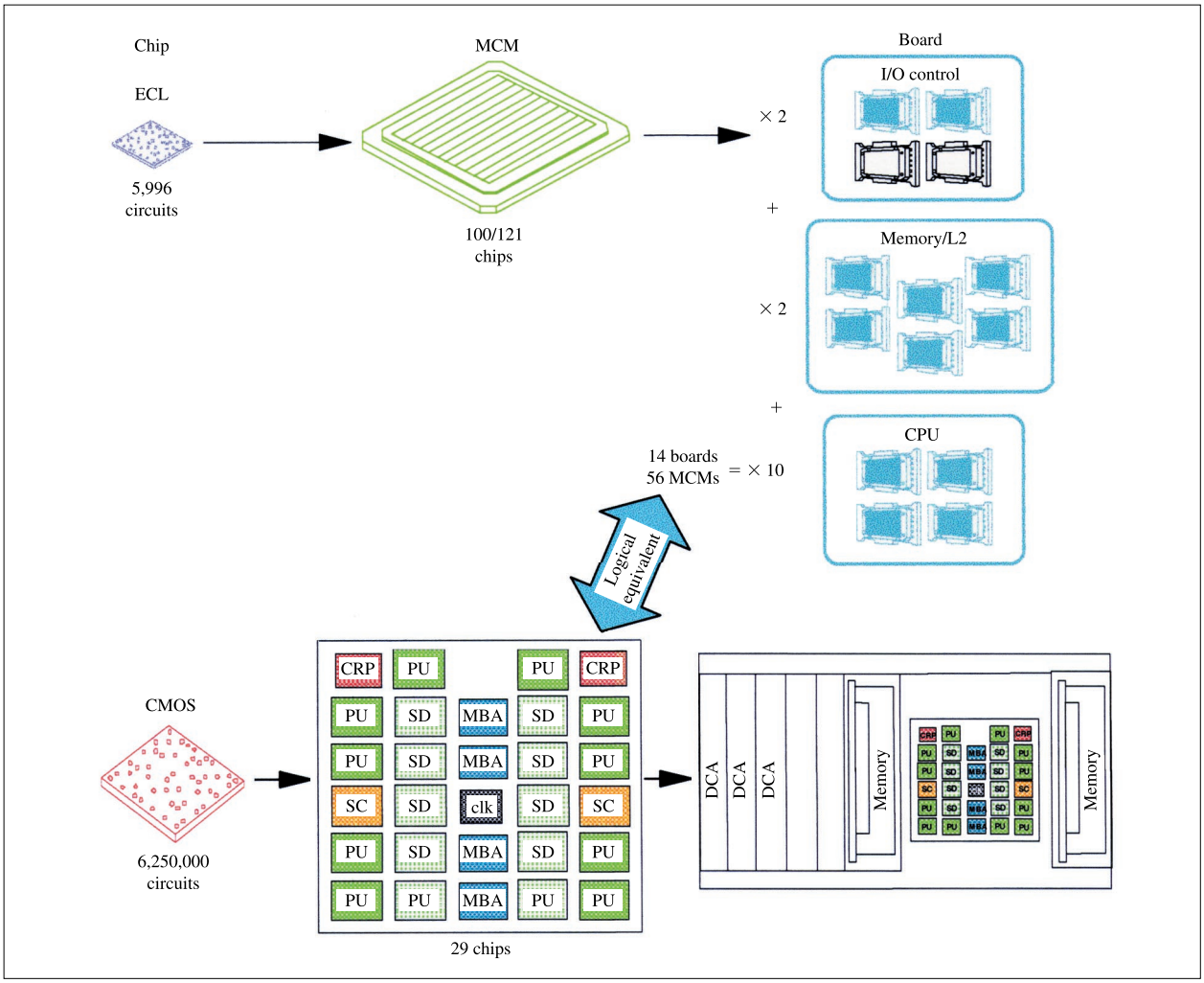

Figure 2

G5 technology integration (ten-way G5 vs. ten-way ECL).

processor. Fourth, a cluster solution, S/390 Parallel Sysplex*, was now available for the most demanding mission-critical applications [7].

Fault-tolerant enhancements have been added with each new generation of CMOS. G2 included dynamic-memory chip sparing. A more robust ECC was included on G3, which also had first-generation CPU sparing. Although it was not dynamic and required manual intervention, sparing permitted a system to be restored to full capacity without requiring an outage. G4 allowed some concurrent CPU sparing and was the first CMOS mainframe to include CPU instruction-level retry.

G5 is the first CMOS mainframe to greatly exceed the capacity of the most powerful TCM mainframe, 9X2. Its challenge was to provide superior function in fault tolerance and concurrent repair compared to the 9X2. However, the design constraints of modern high-performance microprocessors complicate the implementation of error checking. The ability to perform deferred and concurrent maintenance in the 9X2 depended to a large extent on the physical isolation of logical entities such as CPUs and I/O channels. Denser CMOS packaging requires new techniques for graceful degradation and on-line repair.

Today as we assess the evolution of CMOS reliability in the future, there is no certain indication of continued improvement. As the technology develops to smaller line widths and cell sizes and chip densities greatly increase, there are indications that chip failure rates (both hard and soft) may experience an upturn. However, the major

concern with future failure rates is soft errors. Today, soft errors are confined to arrays, permitting the use of well-known and relatively inexpensive techniques to ensure data integrity and prevent unnecessary system crashes, but it is expected that within the next few years transient failures will regularly occur in logic latches and some combinatorial logic. G5 and its successors have microprocessor designs capable of transparent recovery and protection from data corruption in the face of soft errors in logic. This is not typical in the industry.

3. Microprocessor fault-tolerant design

- *Common industry practices*

Other than S/390, existing single-chip CMOS microprocessors, whether RISC or CISC, have limited hardware error detection. In those few instances in which high-coverage error checking in microprocessors is implemented, it is done by duplicating chips and comparing the outputs. Although it is used only by a few specialty vendors, Intel builds into its chips functional redundancy-checking logic that permits master/checker Pentium** microprocessor pairs [8]. Tandem has designed similar off-chip logic to create perfectly checked microprocessor pairs for its Himalaya systems [9]. Both of these techniques require 100% or more logic overhead. Duplicate and compare is, however, adequate only for error detection. Detection alone is inadequate for S/390, which singularly requires dynamic CPU recovery.

Microprocessors contain a component to fetch and decode instructions (I-unit), one or more instruction-execution elements (E-unit), and a cache. Modern microprocessors usually include parity for cache data and data paths where data is generally moved without being altered. Checking of the control, arithmetic, and logical functions in the I-unit and E-unit is considered to be difficult and time-consuming, and to introduce performance penalties. As a result, with the exception of S/390, today's microprocessors leave these components unchecked.

In most instances other than S/390, the accepted wisdom is that single-chip microprocessors are sufficiently reliable that failures are rare and that other mechanisms (time-outs and software detection, for instance) will discover the few problems that do occur. This philosophy depends on continual reliability improvements; in reality, predicting chip technology failure mechanisms is problematic. Future CMOS technologies could be less reliable; latches and dynamic logic will become more susceptible to alpha particles and other cosmic radiation. This is not a concern unique to IBM; it is acknowledged throughout the industry [10]. S/390 design is sufficiently robust to recover from transient logic errors; other microprocessors are not.

In addition, however, there is a real exposure to malfunctions in the hardware that could result in incorrect data being written to memory. The model proposed by Horst et al. [11] projects a pessimistic bound of one in 75 unchecked microprocessors causing a data-corruption error each year. S/390 is designed to prevent data corruption by including extensive error checking in all functional elements—combinatorial logic as well as arrays. Horst's prediction is based on an optimistic evaluation of software detection; it is true that in some cases time-outs and software detection will prevent data integrity problems. However, software does not reliably fail quickly when errors are injected, resulting in corrupted data [12]. Even when the detection is successful, a system with no recovery will usually hang, causing application downtime. S/390 is required to recover transparently whenever feasible, which necessitates instantaneous detection and containment of failures.

The fundamental S/390 fault-tolerant design principles (to preserve data integrity and recover transparently from faults both permanent and transient) have resulted in a microprocessor design point that is unique in the industry. Incorrect results occurring because of technology faults are detected at the source, isolated to prevent propagation, and recovered.

- *G5 fault-tolerant design point*

The 9X2 was a superscalar ECL CPU which permitted out-of-order instruction execution. It was designed to optimize cycles per instruction (CPI). On the other hand, G5 has as its primary performance objective the fastest possible cycle time. The G5 CPU is a pipelined design with one floating-point and two fixed-point execution elements. Only one unit is executing at a time, and only one instruction can be decoded each cycle. The similar G4 design is described in detail in [13]. Primary fault-tolerance goals are to protect data integrity, recover transparently from transient failures, degrade gracefully from permanent failures, and, in most cases, repair permanent failures without application downtime.

ECL packaging permits efficient implementation of extensive in-line checking [5]. Each 9X2 CPU required approximately 400 chips. Often the chips were I/O-pin-limited, and logical functions did not always divide neatly. Developers could exploit the "leftover" logic for decode checkers, localized functional redundancy, and other high-overhead checking mechanisms without increasing the chip count. For the 9X2, in-line CPU error detection used about 30% circuit overhead.

Comparable in-line checking would have been one of the main inhibitors to achieving the low processor cycle times required for G5. A common method of in-line checking for detecting errors in ECL combinatorial logic uses parity prediction. Separate logic with the same inputs

(both data and parity) as the actual function (ALU, shifter, state machine, etc.) calculates the parity of the function output. Parity is checked in the same or a subsequent cycle. This method detects failures in the inputs and output as well as in the combinatorial logic. A simple implementation of a parity predictor is to duplicate the function and generate parity from the input parity and the output of the duplicated function. More optimized techniques have been designed for common functions such as ALUs and shifters. The path length through the parity-prediction logic for an ALU or shifter is longer than the path length through the actual ALU or shifter function. This is true for both ECL and CMOS, but the effects in CMOS can be even worse than in ECL, as described below:

1. Compared to CMOS, high fan-in and fan-out capabilities in ECL result in smaller differences between the actual function and the predicted parity path lengths. Higher capacitance associated with CMOS circuits as their fan-in and fan-out increase makes path lengths in this technology more susceptible to increases as the complexity of the function increases.
2. The added logic for the parity prediction itself increases the chip area. In ECL, the parity-prediction logic could often be packaged on the same chip as the main function, and chip-to-chip wiring was not affected.
3. Additional chip area in turn causes longer interconnection wiring, which also increases the path length.
4. The lower fan-out capabilities of CMOS further increase path length, since the prediction circuits must be connected to critical data buses.

In-line error checking would have been a serious problem for G5, because the primary performance objective is cycle-time reduction. Another design point, for instance a superscalar design that optimizes cycles per instruction (CPI), might find the adverse effects of in-line checking more tolerable. In addition to its impact on cycle time, in-line checking requires more development time. Placement of error checkers requires skill and diligence. Adequate automated tools are not available, so coverage is determined by exhaustive design "walkthroughs." Simulation of the checkers requires carefully placed error injection to ensure detection, and mistakes in the design sometimes cause false detection. Extra error-injection steps are also required during testing of the logic, to verify that the system recovers correctly from the error. Design errors in checking logic tend to be discovered very late in the development cycle, requiring unanticipated changes at critical points in the cycle.

Thus, performance and schedule requirements dictated a different approach for G5. The requirement for dynamic recovery, coupled with other design constraints (e.g., power, performance, second-level packaging) pointed to a highly checked single-chip microprocessor. However, in-line checking of control and arithmetic logic was prohibited by the performance and schedule penalties. The decision was made to duplicate the I-unit and E-unit and compare outputs. With a carefully laid-out floorplan, there are no cross-chip performance-limiting paths. The cycle time is the same as if the same design were implemented on a smaller chip with only one unchecked I-unit and E-unit. The compare-and-detect cycle is completely overlapped in the instruction execution pipeline, so the G5 achieves improved checking without incurring either cycle-time or CPI penalties.

Most of the microprocessor area is devoted to the cache, where data is primarily moved around unchanged. Updates to memory are maintained in an ECC-protected store buffer during instruction execution and transferred to an ECC-protected L2 cache when instruction execution completes. Thus, for the on-chip cache, low-overhead parity checking is sufficient.

Total circuit overhead for cache parity, register-unit ECC (the R-unit is described in the next section), and the duplicate I-unit and E-unit is about 35%. Design, verification, and testing are far simpler than with in-line checking. A larger chip area may result in lower yields and higher costs, but CMOS 6X yield characteristics let G5 exploit transistor densities without a penalty. In addition, the design facilitates meeting all of the key requirements: fault tolerance, performance, and schedule.

- *Recovery and on-line repair*

The key element of the microprocessor R-unit is an ECC-protected register file, the checkpoint array. The checkpoint array keeps track of the entire state of the CPU including register contents and instruction addresses. It operates as follows: 1) As instructions are executed, the results from the two I- and E-units are compared and, if equal, changes to the state of the CPU are placed in an update buffer in the R-unit. 2) Once the execution successfully completes, the update buffer contents are placed in the checkpoint array and L1-pending stores are placed in the store buffer. Step 2 is blocked if an error is detected during instruction execution. This ensures that any completed instruction is error-free and that the checkpoint array and the store buffer accurately describe the state of the machine at the successful completion of instruction execution.

S/390 mainframes have typically performed CPU retry on instruction boundaries. The 9X2 executed retry with a set of CPU microcode algorithms. The particular algorithm used was dependent on the particular instruction that failed. The CPU performed a scatter-gather type of operation in order to back up to a state

prior to the error. All potential faulty information had to be discarded. Certain instructions (e.g., Set Storage Key) could not be retried because the CPU state could not be precisely determined at all points during their execution. Simulation and verification of retry required creation of highly specialized test cases to determine whether recovery was performed correctly. Because there were different algorithms and microcode routines for different instructions, extensive customized testing was required.

The R-unit is a great advance in instruction retry. The recovery scenario is completely controlled by hardware and is identical for all instructions. When an error is detected, the R-unit manages instruction retry as follows: Except for the R-unit itself, the CPU is reset and store buffer contents are sent to L2. The state of the CPU is returned to what it was at the completion of the last prior instruction. If the checkpoint array is error-free, serialized instruction processing begins, retrying the instruction that caused the error. If the retry is successful, the failure is regarded as transient, and the CPU resumes pipelined instruction processing. If the retry is not successful, the error is considered to be permanent, and the CPU is stopped.

On the 9X2, graceful degradation was achieved by coordinated hardware and operating system action, called Processor Availability Facility. If the error was permanent, the CPU would signal the operating system to store the state information in the dispatch queue of another CPU in the system. The failed CPU would be dynamically varied out of the active configuration. The task it was executing would be restarted according to normal dispatch priorities. In addition, packaging permitted on-line repair. Each CPU contained four TCMs mounted on a specialized board with its own, unshared power source. If a CPU failed, it was powered down and the failed TCM was replaced. The CPU was then powered up and reintegrated into the running configuration, all concurrently with continued operation of the remaining CPUs. In contrast, all of the CPUs in G5 are packaged on one multichip module (MCM).

Because the single MCM design precludes a "hot-plug" approach, G5 needed a new method. Dynamic CPU sparing (DCS) improves upon the 9X2 design. Since G3, some CPUs have been designated as "spares." When one of the running G5 CPUs fails and instruction retry is unsuccessful, a transparent sparing operation is performed. Using the service element, DCS scans checkpoint state information from the R-unit of the failed CPU into the R-unit of the spare CPU. To ensure that the action is transparent to the operating system, special hardware tags the spare with the CPU address of the failed CPU. Now the spare is ready to begin executing at precisely the instruction where the other CPU failed. The hardware is effectively performing CPU retry across CPU boundaries. DCS is executed completely by hardware, with no operating system awareness. Applications continue to run without noticeable interruption. If DCS cannot be performed, either because there is no spare or because the checkpoint array is damaged, G5 will execute Processor Availability Facility. Both the 9X2 and G5 schemes keep the applications running. However, contrasted to 9X2 system on-line CPU repair, which requires service personnel travel and parts procurement before it can be carried out, DCS permits the system to be restored to full capacity in less than one second as opposed to hours.

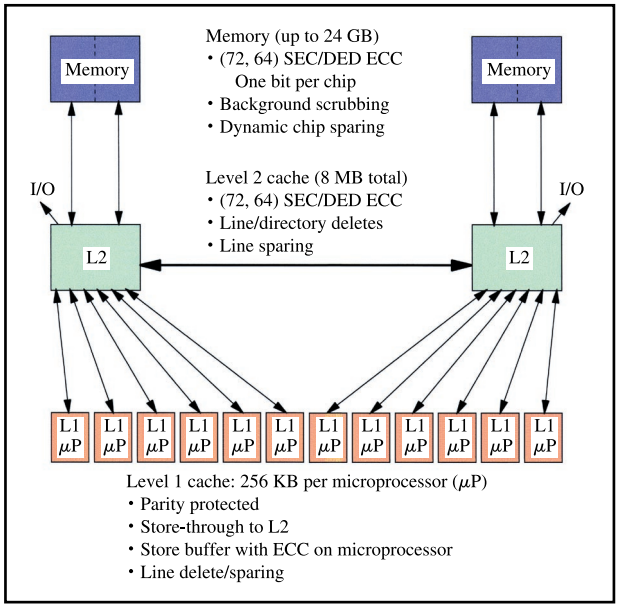

Figure 3

G5 memory hierarchy fault tolerance.

4. Memory hierarchy fault tolerance

The design objective in the memory hierarchy is to continue uninterrupted operation when data errors occur. An overview of the G5 memory hierarchy is shown in **Figure 3**. The failure model predicts a predominance of transient failures, so all levels of the hierarchy must transparently recover from them. Additional fault tolerance provides recovery from many permanent failures. Data redundancy is provided by two primary means, store-through (write-through) cache design and error-correcting codes (ECCs).

L1 is the store-through microprocessor cache. Pending instruction results are maintained both in L1 and in an ECC-protected store buffer. When instruction execution completes, updated results are immediately stored into L2.

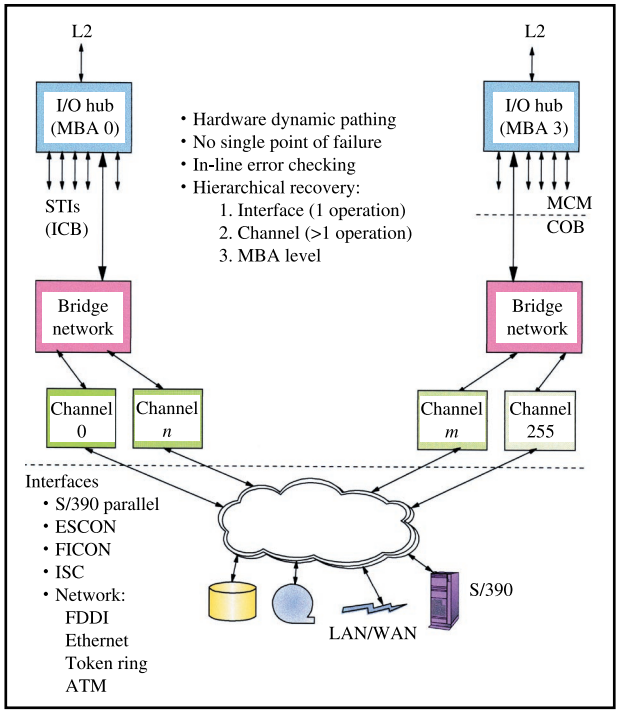

Figure 4

G5 I/O subsystem fault tolerance.

Because L1 data is always replicated, byte parity is adequate for protection. A transient L1 failure is recovered by CPU instruction retry. For a permanent failure, depending on the scope, a cache-line or quarter-cache delete is performed dynamically. A deleted line may be restored with a spare line at the next power-on. Also, if a CPU experiences a permanent failure, all of its changed data is accessible to other CPUs because L2 is shared and L1 is store-through.

Each of the two L2s is shared by six microprocessors. L2 maintains cache coherency for the system and may contain the only version of changed data. L2 enforces strict cache coherency for L1 and L2; data must be in an exclusive state in the L1 directory of a CPU, and no other copies except those in that CPU's L1 and its associated L2 can exist when the CPU modifies the data. A complete description of how L2 manages memory can be found in [14]. To prevent single-bit transient errors from resulting in lost data, L2 is protected by ECC. Permanent faults in L2 that might result in an uncorrectable data error can be avoided by using a cache-delete capability. Faulty locations either in the data array or in the address directory can be dynamically marked as invalid, and the system continues operating with a very slightly smaller L2. In addition, a spare line can be substituted for a failed one. Spare lines are commonly designed on array chips for use by chip manufacturers to increase yield. In G5 the mechanism has been modified to allow a cache line with a permanent fault to be replaced by a spare line at power-on.

L3 is the main memory in G5. It is possible that certain control-logic failures may result in unscheduled system downtime, but the design objective is that array failures will never result in a system failure or a customer outage. Coming full circle, G5 delivers this with a (72, 64) single-error-correct/double-error-detect (SEC-DED) error-correction code. TCM mainframes were designed with single-bit-per-chip ECC so that any failure mechanism of the chip (single cell, word line, bit line, or complete chip kill) would still be correctable by the code. The first two generations of CMOS continued to use a (72, 64) code, but four bits per chip were included in each ECC word. This is the most common memory configuration and ECC in the server industry. However, it is not as robust as the typical S/390 scheme, and some of the chip-failure mechanisms could result in uncorrectable errors and possibly crash the system.

When a chip is b bits ($b \geq 2$) wide, an access to a 64-bit data word may have a b-bit block or byte error. There are codes to variously correct single b-bit errors and detect double b-bit errors. For G3 and G4, a code with 4-bit correction capability (S4EC) was implemented. Because the system design included dynamic on-line repair of chips with massive failures, it was not necessary to design a (78, 64) code which could both correct one 4-bit error and detect a second 4-bit error (D4ED). Such a code would have required an extra chip per checking block. The (76, 64) S4EC/DED ECC implemented on G3 and G4 is designed to ensure that all single-bit failures of one chip (and a very high probability of double- and triple-bit failures) occurring in the same doubleword as a 1–4-bit error on a second chip are detected [15]. G5 returns to single-bit-per-chip ECC and is therefore able to again use a less costly (72, 64) SEC/DED code and still protect the system from catastrophic failures caused by a single array-chip failure.

Background scrubbing is performed on L3 data to reduce the frequency of transient single-bit failures. Automatic on-line repair of faulty DRAMs is done using built-in spare chips. Counts of correctable errors are maintained on a per-chip basis. When a threshold is exceeded, the data from the over-threshold chip is dynamically transferred to an error-free spare chip. A chip with systematic failures, e.g., word line or chip kill, will rapidly exceed threshold and be removed from the system.

5. I/O subsystem fault tolerance

The S/390 I/O subsystem is system hardware that connects main memory and CPUs to peripheral devices and their controllers over various standard interfaces. The design is

described completely in [16]. The fault-tolerant design objectives of the I/O subsystem are to exploit the redundant paths between all devices and main memory and to minimize the scope of any failures [17].

The S/390 I/O subsystem provides high availability through multiple paths to I/O devices. As shown in **Figure 4**, devices can have completely redundant paths through the hardware. All of the paths are normally active and are used to enhance performance as well as to provide backup for one another if one should fail. The I/O channel adapters perform direct memory access with robust memory protection on behalf of I/O devices such as disk and tape storage, network communications, and server-to-server cluster (Parallel Sysplex) services. They prevent I/O devices from unauthorized memory read operations and from memory write operations into arbitrary memory locations. I/O channel adapters also provide error isolation by preventing the propagation of interface errors into the system.

The G5 server introduces fibre channel (FICON*) as the primary I/O interface. It preserves the data integrity and fault tolerance of the S/390 I/O architecture and programming model by introducing a new fibre channel upper-layer protocol for S/390. The parallel channel is remapped from five chips mounted on an MCM to a single chip, greatly reducing the failure rate. New on G5 is the Integrated Cluster Bus (ICB), which provides a dramatic reduction in failure rate by incorporating the ISC I/O channel adapter function into the hub chip [18]. The hardware failures of separate I/O adapter and bridge hardware are eliminated. In addition, superior performance reduces the total number of interfaces needed, thus achieving even lower failure rates.

Parallel Sysplex is the S/390 cluster. It can be configured for near-continuous availability with two or more interconnected mainframes. Alternatively, it can be built of separate partitions within one mainframe and assume the hardware availability characteristics of the mainframe. The advantage of a single-mainframe sysplex is that, unlike a standalone G5, it is tolerant of software failures. The internal coupling (IC) channel delivers a practical Parallel Sysplex in a single G5 mainframe for the first time. Before G5, single-mainframe Parallel Sysplex implementations required communication between the operating system and the coupling facility partitions to be done either by interconnected pairs of coupling channels or by the integrated cluster migration facility (ICMF), designed specifically for test environments. Interconnecting coupling channels have acceptable performance, but require additional hardware that increases cost and failure rates. ICMF does not require any coupling-channel hardware, but it does require interaction with the logical partitioning (LPAR) hypervisor. The LPAR task switches required to communicate between the partitions greatly increase path lengths, leading to a solution with performance that is acceptable only in a limited test environment. In contrast, the IC uses processor microcode to communicate between partitions. The LPAR hypervisor is not involved, and the resulting path lengths of the IC are much shorter, with much superior performance. With the IC, enterprise applications can take advantage of the hardware fault tolerance of a standalone G5 combined with the software fault tolerance of a distributed Parallel Sysplex.

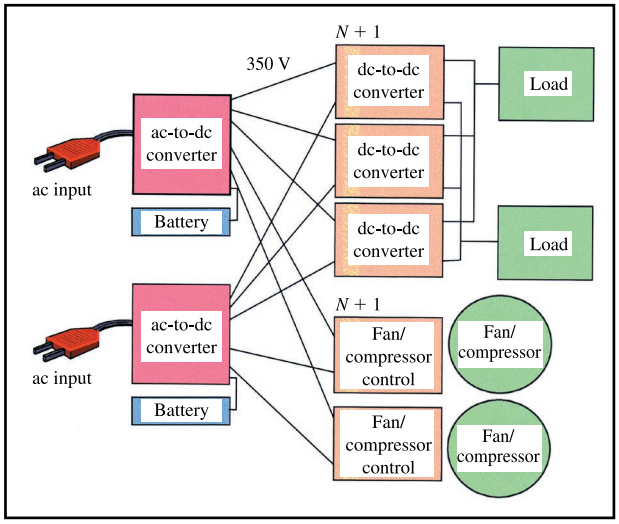

Figure 5

G5 power/cooling fault tolerance.

6. Power and cooling fault tolerance

To meet mainframe availability requirements, the power-and-cooling subsystem is designed with no single points of failure and concurrent repair capability for virtually all components. In G5, lower power consumption makes meeting these requirements easier than in the 9X2; the maximum power for the largest G5 is 5 kW, while the maximum power of the largest 9X2 is 144 kW. In TCM mainframes, the high cost of high-power components made duplication of bulk power elements prohibitively expensive, but for CMOS it becomes practical.

Figure 5 shows the major components of the power and cooling subsystem. Separate ac power inputs feed dual ac-to-dc converters, and each of these has several 350-V dc outputs. Also, each can be connected to a 350-V gel cell battery. The lower power requirements of the CMOS mainframes allow battery backup to be attractive, and the batteries provide at least ten minutes of backup power in the largest mainframes. The ac-to-dc converters are fully

redundant, and when they are both on line, the load is shared. If one of them fails, the other takes over the 350-V load. Also, three-phase ac input current is used, and one of the phases can fail without causing an outage. If one of these converters fails, it can be concurrently replaced.

Both ac-to-dc converters feed all dc-to-dc converters and controllers for the fan and compressor motors through individual point-to-point 350-V cables. The dc-to-dc converters provide multiple output voltages for the circuitry, the fans move air for cooling the circuitry, and the compressors are part of a refrigeration system that cools the processor module of the largest G5. The dc-to-dc converters, controller/fan assemblies, and controller/compressor assemblies are all in an $N + 1$ configuration. For example, if two dc-to-dc converters are required to supply the load, a third is added for on-line redundancy. In the case of the fans and compressors, two are always used. All of these components can be concurrently replaced. In the case of the fan and compressor motors, some failures are detected by measuring the back EMF of the motor field coils. When a fan fails for any reason, the speed of the second fan is increased to compensate for the loss. Under normal operating conditions, the speed of the fans is kept to a minimum to reduce acoustic noise. In contrast, only one of the pair of compressors operates at a time. A switchover is made every 24 hours to ensure that both assemblies are operational. When a controller/compressor fails, it can be concurrently maintained. Quick disconnects are provided at the evaporators for the refrigerant hoses. Coordinated control of all power converters and fan/compressor controllers uses duplicated communication paths. The control interface to the power and cooling subsystem is provided by an IBM ThinkPad* notebook computer.

The bulk of the I/O subsystem hardware is packaged into various circuit cards that provide many different I/O interfaces, and most of these cards can be concurrently replaced. Live insertion of circuit cards causes large noise spikes on the power supply, so a low-noise mechanism for applying power is required. The 9X2 has only a few different I/O card types, and during concurrent removal and insertion, the power to the individual cards is ramped up and down by a system of multiple pin lengths on the card connector. The longest pins supply bulk power, while shorter pins control solid-state switches (FETs) on the card. These switches supply current for ramping the power up and down. To meet the modern I/O requirements of the CMOS mainframes, many more card types of widely varying power requirements are needed. For example, ESCON* I/O cards require several times less power than ATM cards. These requirements have motivated a more flexible design called the "soft switch." Instead of multiple-length pins, each I/O card position has an interface that controls the soft switch on the card. Before an I/O card is removed, the soft switch is deactivated, causing the power on the card to drop. After a card is inserted, the soft switch is activated and the power to the card is turned on.

To reduce human error during concurrent card replacement, a group of LED indicators, one for each card position, is used to positively identify the card to be replaced. The 9X2 uses indicators on the cards themselves. On G5, these indicators have been moved off the cards and onto the card cage. This is an improvement over the 9X2 design, because the indicator can also flag the empty card position that will receive the new card, and because the indicators are provided in a uniform way for all card types.

7. Conclusions

S/390 mainframes have continually improved their fault tolerance, but this improvement has not been steady, incremental growth. The introduction of TCM technology was accompanied by greatly improved instantaneous error detection, first-failure data capture, and single-FRU isolation. The demands of mission-critical applications for significantly improved uptime resulted in the introduction of transparent recovery and on-line repair in all major subsystems of the 9020. The dense packaging of CMOS required new methods to be invented for error detection and concurrent repair in CPUs. The key innovations in G5 deliver an industry-unique microprocessor which detects errors, recovers from both transient and permanent faults, and restores full processing capacity after a permanent fault, all under hardware control and without intervention or noticeable interruption in enterprise applications. In addition, CMOS densities are exploited, and there is no cycle time or CPI penalty. Internal Channel permits a high-performance Parallel Sysplex within a single mainframe without additional hardware. This delivers software fault tolerance in addition to the G5 hardware fault tolerance. Although quite different in implementation, both the CPU and the IC exemplify the S/390 fault-tolerant design approach of reducing the effects of electronics technology failures while optimizing the uptime of application programs. First-quarter 1999 full-field data indicates that G5 is delivering an MTTF of more than 45 years. About 84% of all repairs are concurrent, 14.6% can be scheduled, and only 1.4% result in immediate loss of applications.

Acknowledgments

The authors wish to thank Guru Rao, Ram Chillarege, and Bill Shen for their careful reviews, thoughtful comments, and recommendations. They also thank Scott Swaney, Pak-kin Mak, Lisa Heller, and John Kinnear, for

respectively describing design details of L1, L2, dynamic CPU sparing, and the power/cooling subsystem.

*Trademark or registered trademark of International Business Machines Corporation.

**Trademark or registered trademark of Intel Corporation.

References

1. A. Avizienis, H. Kopetz, and J. C. Laprie, *Dependable Computing and Fault-Tolerant Systems*, Springer-Verlag, New York, 1987, pp. 1–36.
2. M. Y. Hsiao, W. C. Carter, J. W. Thomas, and W. R. Stringfellow, "Reliability, Availability, and Serviceability of IBM Computer Systems: A Quarter Century of Progress," *IBM J. Res. Develop.* **25,** No. 5, 453–465 (1981).
3. D. C. Bossen and M. Y. Hsiao, "Model for Transient and Permanent Error-Detection and Fault-Isolation Coverage," *IBM J. Res. Develop.* **26,** No. 1, 67–77 (1982).
4. D. P. Siewiorek and R. S. Swarz, *Reliable Computer Systems*, Digital Press, Bedford, MA, 1992, pp. 485–507.
5. L. Spainhower, J. Isenberg, R. Chillarege, and J. Berding, "Design for Fault-Tolerance in System ES/9000 Model 900," *Proceedings of the 22nd Annual International Symposium on Fault-Tolerant Computing*, 1992, pp. 38–47.
6. L. Spainhower, T. A. Gregg, and R. Chillarege, "IBM's ES/9000 Model 982's Fault-Tolerant Design for Consolidation," *IEEE Micro* **14,** No. 1, 48–59 (1994).
7. J. M. Nick, B. B. Moore, J.-Y. Chung, and N. S. Bowen, "S/390 Cluster Technology: Parallel Sysplex," *IBM Syst. J.* **36,** No. 2, 172–201 (1997).
8. *Pentium Family User's Manual, No. 1: Data Book*, Order No. 241428, Intel Corporation, Mt. Prospect, IL, 1994, pp. 12-7–12-8.
9. Tandem Computers Incorporated, "NonStop Himalaya Range: K200, K2000, and K20000 Servers," *NonStop Servers Product Description*, 1995.
10. J. Robertson, "Alpha Particles Worry IC Makers as Device Features Keep Shrinking," *Semicond. Business News*, October 21, 1998.
11. Robert Horst, Doug Jewett, and Daniel Lenoski, "The Risk of Data Corruption in Microprocessor-Based Systems," *Proceedings of the 23rd Annual International Symposium on Fault-Tolerant Computing*, 1993, pp. 576–585.
12. S. Chandra and P. M. Chen, "How Fail-Stop Are Faulty Programs?" *Proceedings of the 28th Annual International Symposium on Fault-Tolerant Computing*, 1998, pp. 240–249.
13. C. F. Webb and J. S. Liptay, "A High-Frequency Custom CMOS S/390 Microprocessor," *IBM J. Res. Develop.* **41,** No. 4/5, 463–473 (1997).
14. P. R. Turgeon, P. Mak, M. A. Blake, M. F. Fee, C. B. Ford III, P. J. Meaney, R. Seigler, and W. W. Shen, "The S/390 G5/G6 Binodal Cache," *IBM J. Res. Develop.* **43,** No. 5/6, 661–670 (1999, this issue).
15. C. L. Chen and M. Y. Hsiao, "Error Detection and Correction for Four-Bit-per-Chip Memory System," U.S. Patent 5,757,823, 1998.
16. T. A. Gregg, "S/390 CMOS Server I/O: The Continuing Evolution," *IBM J. Res. Develop.* **41,** No. 4/5, 449–462 (1997).
17. L. Spainhower and T. A. Gregg, "G4: A Fault Tolerant CMOS Mainframe," *Proceedings of the 28th Annual International Symposium on Fault-Tolerant Computing*, 1998, pp. 432–440.
18. T. A. Gregg, K. M. Pandey, and R. K. Errickson, "Integrated Cluster Bus for the IBM S/390 Parallel Sysplex," *IBM J. Res. Develop.* **43,** No. 5/6, 795–806 (1999, this issue).

Received December 17, 1998; accepted for publication May 27, 1999

Lisa Spainhower *IBM Server Development, 522 South Road, Poughkeepsie, New York 12601 (lisa@us.ibm.com).* Ms. Spainhower is a Senior Technical Staff Member in Server Design. She is responsible for technical competitive analysis and server strategy, with a focus on high availability and fault tolerance. She received a B.A. degree from the University of Michigan. Ms. Spainhower is a member of the IEEE and the IBM Academy of Technology.

Thomas A. Gregg *IBM System/390 Division, P.O. Box 950, Poughkeepsie, New York 12602 (tomgregg@us.ibm.com).* Mr. Gregg is a Senior Technical Staff Member in the S/390 System Design group. He received an SC.B. degree in engineering from Brown University in 1972 and continued under a university fellowship, receiving an SC.M. degree in electrical engineering in 1974. He joined IBM at the Poughkeepsie Laboratory in 1973. Mr. Gregg has held various technical positions in the area of I/O subsystem design. He holds numerous patents utilized in IBM ESCON and Parallel Sysplex channel products, and has received nine IBM Invention Achievement Awards. He received an IBM Corporate Award and an IBM Outstanding Innovation Award for work on ESCON products, and three IBM Outstanding Innovation Awards for work on Parallel Sysplex. He is a member of the IEEE.