# Design methods for attaining IBM System z9 processor cycle-time goals

G. Mayer
G. Doettling
R. F. Rizzolo
C. J. Berry
S. M. Carey
C. M. Carney
J. Keinert
P. Loeffler
W. Nop
D. E. Skooglund
V. A. Victoria
A. P. Wagstaff
P. M. Williams

*Cycle-time targets were set for the IBM System z9™ processor subsystem prior to building the system, and achieving these targets was one of the biggest challenges we faced during hardware development. In particular, although the processor-subsystem cycle-time improvement was driven primarily by the technology migration from CMOS 9S (130-nm lithography) for the prior IBM System z990 to CMOS 10S0 (90-nm lithography) for the new system, the cooling capability for the System z9 resulted from a direct migration of the System z990 implementation with very limited improvements. The higher device current leakage and power associated with the technology migration, combined with the fixed cooling capability, created a technology challenge in which the subsystem cycle time and performance were potentially limited by cooling capability. Our solution emphasized silicon technology development, chip design, and hardware characterization and tuning. Ultimately, the System z9 processor subsystem achieved operation at 1.7 GHz, which exceeded the original target.*

## Introduction

The IBM System z9* processor subsystem is a redesign of the IBM System z990 processor subsystem, and the technology change from 130-nm to 90-nm CMOS silicon-on-insulator (SOI) technology is the primary cause of subsystem performance improvement. **Table 1** shows some of the technology parameters for the 90-nm technology. In addition, Elastic Interface 2 (EI-2) [1], which is essentially a synchronous wave-pipelined interface, was required for nearly all of the chip-to-chip communications in order to enable the higher subsystem chip frequencies. A number of architectural enhancements improved performance; however, these enhancements were minor compared with the performance boost provided by the technology change.

The System z9 processor subsystem is based on the System z990 implementation that was described by Mak and colleagues in 2004 [2]. The subsystem consists of eight dual-core processor chips, connected through a centralized switch and coherency manager known as the system control element (SCE). The SCE includes a second-level cache (L2 cache) and a pipelined switch that manages data routing and maintains strong storage coherency across the multiprocessing system [2]. The main storage controller (MSC) connects to the SCE and provides the subsystem interfaces to the third-level cache (L3 cache) and the I/O subsystem. The combination of SCE and MSC is referred to as the nest, and it runs at half the frequency of the processors. The subsystem chips, which include such elements as the processor, SCE, MSC, and clock synchronization, are packaged on a single multichip module (MCM) to minimize interconnect delay.

The cooling capability for the System z9 processor subsystem resulted from a direct migration from the previous System z* generation, with limited enhancements. The technology migration, combined with the fixed cooling capability, created a technological challenge in which the subsystem cycle time and performance would likely be limited by the cooling capability. This paper describes the innovations and enhancements in design, characterization, and hardware tuning that were necessary to address this challenge.

©**Copyright** 2007 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.

**0018-8646/07/$5.00 © 2007 IBM**

**Table 1** Chip technology parameters for 90-nm technology. ($L_{eff}$: effective transistor device length; $T_{ox}$: transistor gate oxide thickness; $M_n$: $n$th metal level.)

| *Characteristics* | *Value* |
|---|---|
| Lithography | 90 nm |
| $L_{eff}$ | 40 nm |
| $T_{ox}$ | 11 Å |
| $M_1$ pitch | 0.245 $\mu$m |
| $M_2$–$M_4$ pitch | 0.280 $\mu$m |
| $M_5$–$M_6$ pitch | 0.560 $\mu$m |
| $M_7$–$M_8$ pitch | 1.120 $\mu$m |
| $M_9$–$M_{10}$ pitch | 1.680 $\mu$m |
| Power-supply voltage | 1.0 V |

When we use the phrase *cycle time* in this paper, we refer to the fundamental clock period of the subsystem chip under development. Generally speaking, this cycle time represents the time limit for data to propagate from one state latch to another state latch. Extensive tests have indicated that actual physical chips can have speeds that are significantly different from those predicted by designers, and such unpredictabilities may result from timing-tool inaccuracies, the manufacturing process, and other factors.

## Processor design considerations

### *Dual processor chip floorplan*

The System z9 dual microprocessor chip is implemented in a 90-nm process, with a ten-level metal stack and a die area of 185 $mm^2$ (11.844 mm by 15.778 mm). For comparison, the previous-generation System z990 dual microprocessor used a 130-nm process with eight levels of metal and a die area of 265 $mm^2$ (14.1 mm by 18.9 mm).

The ten levels of metal wire resources were allocated in such a way as to enable concurrent design on all four levels of the physical hierarchy, including chip, cores, units, and macros [3]. As is well known to circuit designers, the term *macro* refers to the logic function design on a chip that specifies how certain logic elements are interconnected and that also specifies the physical pathways and wiring patterns between components. The term *unit* refers to such entities as the instruction unit (IU) or floating-point unit (FPU), which form the core central processor (CP) of the system. Chip infrastructures such as clock H-tree and I/O received metal levels $m_9$ and $m_{10}$ over the entire die. The clock grid was routed on $m_7$ and $m_8$. Power was routed on all levels of metal in a regular pattern. Signal interconnects over the chip silicon area used $m_1$ through $m_{10}$. In the processor core area, the $m_6$–$m_{10}$ metal levels were allocated among chip, core, and unit levels of the physical hierarchy. Nets with more negative timing margins were assigned to the lower-delay wire classes, where wire class is defined by metal level, wire width, and spacing.

For a variety of reasons, the die size was not reduced by the full lithographic shrink factor, which is 70% for a change from the 130-nm process to the 90-nm process. For example, higher-performing metals were not reduced by a 70% factor in order to ensure that robust clock and power design requirements were met. The I/O C4 (controlled collapse chip connection) shrinkage was 88% to support the MCM chip interface. (A C4 is also often referred to as a *C4 solder ball*.) To support the higher-bandwidth requirements on the MCM, EI-2 [1] was introduced to the System z9, and this introduction required a much more stringent resistance/capacitance (*RC*) specification for the I/O wires. As a result, the I/O drivers and receivers were grouped in smaller numbers in order to place them closer to their associated C4 pads. The lower levels of metal ($m_1$ through $m_4$) were affected by a titanium liner, which made signal nets that were noncritical in the System z990 critical for the System z9, and those nets had to be moved upward in the metal stack to the higher-performing metal levels. To keep the core function tightly packed and to ensure that timing paths did not become constrained by high densities of wires, the full ten-level metal stack was used. Some I/O macros and support logic were placed on the side of the core to make higher-performing metal available for timing-critical function in the core.

### *Design approach to account for degradation effects*

Lessons learned from our experience with the prior-generation System z990 CP chip made it obvious that a close collaboration between IBM teams would be required in order to solve the challenges of power dissipation and process or parametric variability when migrating to the next generation of deep-submicron CMOS technology. Process and parametric variations had driven significant hardware analysis efforts and reduced the circuit-limited yield (CLY) for the System z990 CP chip. In response to these prior challenges, all anticipated issues were addressed early in the design of the System z9 processor. We introduced several array yield improvements, such as the inclusion of additional programmable timing bits, a more robust content-addressable memory (CAM) cell design, and additional eFUSE disable and redundancy repair features. (IBM eFUSE technology combines unique software algorithms and microscopic electrical fuses.) Aside from functional yield issues, process variation and device degradation effects are likely to cause significant cycle-time limitations

and produce a significant difference between design cycle time and cycle times available in the final product.

Negative bias temperature instability (NBTI) has become one of the most important degradation problems for today's CMOS technologies [4, 5]. NBTI is caused by the generation of interface traps due to the electrical field across the thin oxide of negatively biased p-FET transistors. Over time, this causes the threshold voltage ($V_t$) of the p-FET transistors to increase and therefore results in a reduced drive current [6]. The amount of degradation actually depends on various factors, such as temperature, the supply voltage $V_{dd}$, switching factor, switching history, and p-FET location in the stack. Therefore, we do not have a simulation model that would include every detail of all degradation effects for a practical design timing methodology used for a microprocessor and its components. However, a carefully chosen constant-$V_t$ adder can be applied to every p-FET transistor in the design, and this provides a very accurate estimate for the end of live (EOL) degradation. The effect of EOL degradation can be empirically determined by measuring the maximum chip frequency before and after burn-in (BI) stress runs that represent the lifetime stress of the device under test. (The term *burn-in* refers to early tests that are run at elevated temperatures to stress and test the chip.) The use of a $V_t$ adder in the design phase has a clear benefit because it allows us to tune NBTI-sensitive critical paths, resulting in a minimum of EOL degradation for the maximum chip operation frequency. **Table 2** shows the different $V_t$ adders used for testing specific conditions of voltage, temperature, and time.

Because of the high-reliability demands for the System z9, the processor subsystem chips are subject to a dynamic BI that results in a p-FET $V_t$ degradation that can be modeled by a $V_t$ adder of 52 mV (see Table 2). Later, during the chip-attach process, the chips go through an elevated temperature reflow step that liquefies the solder balls on the chips and substrate, causing them to join. Since the transistors receive no power during this step, this reflow eliminates some of the surface traps, thus reducing the p-FET degradation. Results from the previous 130-nm technology showed a significant reduction (~80%) in p-FET degradation, or NBTI, and the calculated reductions for the 90-nm technology (shown in Table 2) are based on this. The System z9 design targeted the Use 6 condition in Table 2, which predicts a final p-FET degradation modeled by a $V_t$ adder of 12 mV, which was used during the initial design phase. For the final design phase timing work, the targeted $V_{dd}$ was raised to 1.05 V, and the $V_t$ adder was adjusted to 17 mV accordingly. Pre- and post-BI measurements proved that the remaining NBTI-induced frequency degradation was significantly less than 1%. This enabled the final product to be run at a faster cycle time because of a reduction in device degradation.

**Table 2** p-FET $V_t$ adder calculations to model p-FET NBTI degradation through end of life of the system and chips (100% duty cycle). The various use conditions assume no burn-in or parts subjected to post-burn-in reflow. To obtain the $V_t$ adder for 50% duty factor, multiply the adders below by 0.6.

| *Conditions* | *$V_t$ adder* (mV) |
|---|---|
| Burn-in: 1.575 V, 140°C, 48 hr | 52 |
| Use 1: 1.20 V, 55°C, 100K hr | 27 |
| Use 2: 1.15 V, 55°C, 100K hr | 24 |
| Use 3: 1.10 V, 55°C, 100K hr | 20 |
| Use 4: 1.05 V, 55°C, 100K hr | 17 |
| Use 5: 1.00 V, 55°C, 100K hr | 14 |
| Use 6: 0.95 V, 55°C, 100K hr | 12 |

With respect to prior product chips, we also observed that achievable subsystem frequency was limited as a result of the natural variability of metal-layer sheet resistances and line capacitances. Therefore, the System z9 design also focused on accounting for this degradation effect in early formulations of the timing methodology. This was accomplished by increasing (i.e., "uplifting") the wire *RC* values by constant multiplicative factors, called uplift factors, as discussed shortly.

Resistance ($R$) and capacitance ($C$) values for metal layers exhibit a statistical distribution of values and can vary up to ±40% from their nominal value. Because $R$ and $C$ values are not independent of each other, the impact of this variation on the wire delay ($RC$) is significantly smaller and depends on the width and spacing of the metal layer.

We used Monte Carlo simulations, performed by PowerSPICE (the IBM circuit simulator tool), to generate individual *RC* uplift factors for each wire class that was used in the subsystem design. These uplift factors were chosen in such a way so that they covered 85% of the statistical distribution of the *RC* delay of each wide wire code and ranged from 6% to 13%.

As the chip design progressed from high-level design to detailed implementation, higher-accuracy wiring modeling was utilized. During the early (high-level) design phase, timing is calculated without wire data, such as wire delay and parasitic data. This provides rough timing feedback information to designers, even if the detailed physical design work has not started and high-level floorplanning is still ongoing. The subsequent design phase estimates wire lengths using the Manhattan distance (the shortest distance possible) and predefined assumptions for wiring layer and width, and spacing

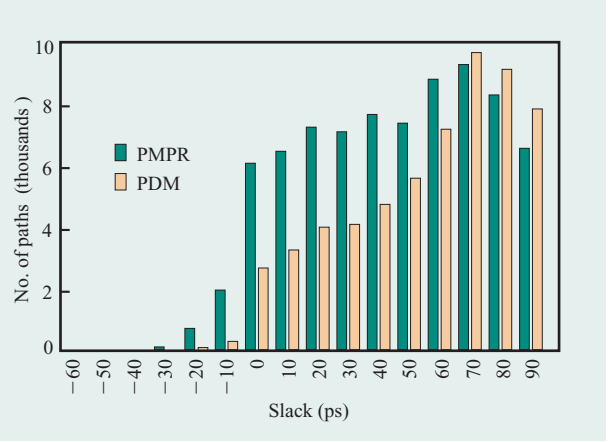


**Figure 1**

Dual-processor chip timing histogram comparing 2D PMPR-based chip timing and 3D PDM-based chip timing methods.

of the connections. As the integration work progresses, actual routing solutions are incorporated using real wires, where they exist in the design, and estimated wire positions and widths for nets that are not yet routed. To model the detailed wire characteristics in this mixed mode, a two-dimensional (2D) database is used that contains pi models, poles, and residues (PMPR) values for a net. Once all nets are routed, the final timing phase utilizes a three-dimensional (3D) fully extracted wire model based on a parasitic data model (PDM) database that contains netlist information for $R$, $C$, $L$ (self-inductance), and $M$ (mutual inductance) for each net. In addition, noise uplifts (i.e., timing adjustments based on the simulated effects of coupled noise) are fed back into the timing model.

Three-dimensional extraction works only for the physical wires that are not open or shorted. (The process of 3D extraction includes wires in the metal layers above and below the subject wire, in addition to adjacent wires, when calculating the parasitic coupling.) Both shorts and open circuits usually arise from mismatches between the actual layout of the current cell and the simplified layout model for the cell used at the other levels of the physical hierarchy, which enables design work to proceed in parallel at all hierarchy levels. The design is not in a state that allows performing 3D wire extraction on all wires until just before the design release. Because the 2D wire modeling makes use of the actual width and metal layer of a route, and assumes a minimum spacing from neighboring tracks as well as fully populated wiring levels below, it tends to be more conservative in its modeling compared with the more accurate 3D extraction. **Figure 1** shows a comparison of 2D- and 3D-based chip timing, using a histogram of paths with a slack (timing margin) less than 100 ps.

With the System z9 project, we introduced an additional level of quality to the project flow as indicated above. We developed a methodology that enables the timing convergence process to benefit much earlier in the project by using 3D extracted data for all of the wires that could be extracted successfully, and we merged PMPR-based wiring data for all others. This new methodology enabled the design team to focus on the most critical paths and to reduce the wire delay percentage of these paths in order to improve the sort capability of the design. As is customary in the field of chip design and testing, chips are sorted for performance on the basis of various measurements.

### *Design approach to minimize power*

For the System z9, one of the biggest challenges, in addition to achieving the cycle-time target, was to meet the power and thermal targets. Therefore, the design initially targeted the use of a relatively low $V_{dd}$ and did not consider aggressive line centering (i.e., polysilicon-gate width-process tailoring) to keep both dynamic and static power dissipation at a minimum.

With the technology map from the 130-nm to the 90-nm SOI process, we projected a significant increase in device leakage currents. In addition to the increase in subthreshold leakage, the gate leakage also increased drastically. While the subthreshold current contributes to the static leakage power only during the period in which a transistor is switched off, the gate leakage occurs in both states of a transistor.

The subthreshold leakage depends exponentially on the transistor threshold voltage $V_t$. Thus, by fully exploiting IBM triple-$V_t$ devices, we had a way to very effectively reduce dc power. The System z990 processor used high-$V_t$ n-FET devices only in the array SRAM cells. All peripheral array circuits and all logic circuits in the System z990 used standard threshold voltages, except for approximately 6.5% of the logic transistor gate widths, which had to be implemented in low $V_t$ to improve the most critical paths. Reducing low-$V_t$ usage and introducing high-$V_t$ devices for noncritical logic was a key design goal for reducing power consumption in the System z9. Initially, all devices in the SRAM cell were switched to high $V_t$, as were the scan portions of the logic latches and the early-mode timing-delay books. The standard logic library and all repeaters were extended with a high-$V_t$ version in order to provide a choice of three $V_t$ versions for each book. **Table 3** compares the device threshold-voltage usage for the different devices of the System z990 and the System z9 CP chip. (The design width is given in meters; the scale factors used to obtain

**Table 3** Comparison of low-, nominal-, and high-$V_t$ device usage in dual-processor chip for System z9 and System z990. The design width in meters is the sum of the device widths of all of the transistors on the chip. Percentages represent the portion of the total device width for that device type.

| | *Low $V_t$* | *Standard $V_t$* | *High $V_t$ Outside SRAM* | *Standard $V_t$ SRAM cells* | *High $V_t$ SRAM cells* |
|---|---|---|---|---|---|
| System z990 (CMOS 9SG) (%) | 5 | 78 | 0 | 3.4 | 13.5 |
| Design width (m) | 20.8 | 317.9 | — | 12.5 | 54.9 |
| System z9 (CMOS 10S0) (%) | 3 | 66 | 14.3 | 0 | 16.5 |
| Design width (m) | 14.6 | 315.4 | 67.7 | — | 78.6 |

the wafer dimensions for 130-nm and 90-nm technologies are 0.25 and 0.175, respectively.)

The subsequent power calculation showed that the increased high-$V_t$ usage and the reduction of the numbers of low-$V_t$ devices lowered the overall leakage power dissipation by 15% at the nominal silicon process settings and by 20% at the three-sigma fast point in the process distribution. This usage also enabled a faster line centering at constant power.

Spare (i.e., unused) gates provide the opportunity to incorporate late design changes by implementing them with only metal changes which can be done after the front-end-of-line (FEOL) design has been released. Traditionally, the inputs of the spare gates (n-FET and p-FET transistor gates) have been connected to GND, resulting in constant gate leakage for the p-FET devices whose drain and source contacts were both tied to $V_{dd}$. By disconnecting the p-FET gates from the n-FET gates and connecting them to $V_{dd}$, the gate leakage of all spare devices was eliminated, resulting in an additional leakage power reduction of approximately 4%. Finally, the use of thick-oxide decoupling cells eliminated associated gate leakage while reducing decoupling capacitance by approximately 20%.

Another major contributor to the device-leakage savings shown in Table 3 is the improved design flow of custom and random logic macros (RLMs), as discussed below. With the need to achieve the desired cycle time, the measures of device sizing and $V_t$ selection are critical. Though the two have opposing effects on the power and cycle-time targets, they must be optimized in order to achieve both targets. Leakage depends directly on the size of the transistors and hence on the area of the macro. The size of the RLMs must be constrained in order to avoid disruptive changes in the floorplan and wiring of the unit and chip.

The RLM-optimization process uses high-$V_t$ devices to minimize leakage and low-$V_t$ devices to improve performance. Because leakage increases exponentially with $V_t$, the use of high $V_t$ contributes substantially to power savings and thus should be maximized. Also, because of the negative impact of low-$V_t$ devices, which increase leakage, the use of such devices should be minimized. We generally used nominal-$V_t$ devices first; high-$V_t$ devices were then used for noncritical timing paths, and low-$V_t$ devices were used for the most timing-critical paths. Paths are timing-critical if they have a negative slack (i.e., timing margin) and/or a slew (i.e., edge speed at the input to a gate) that is larger than the target slew. Therefore, we used the following design flow to optimize a placed and routed design:

**Table 4** Timing and $V_t$-optimization components. (FAR: free-area recovery; TPS: total positive slack.)

| *Optimization engine* | *Optimizes* | *Target paths* | |
|---|---|---|---|
| | | *Noncritical paths* | *Critical paths* |
| EinsTuner | Device width, area | FAR | TPS |
| $V_t$ conversion | Device $V_t$ | High $V_t$ | Low $V_t$ |

1. Convert all gates to nominal $V_t$ usage.
2. Noncritical path optimization:
   a. Device-width tuning to recover free area [i.e., free-area recovery (FAR)].
   b. High-$V_t$ conversion.
3. Critical path optimization:
   a. Device-width tuning for performance total positive slack (TPS).
   b. Low-$V_t$ conversion.
4. Re-extract and re-time the design.

**Table 4** distinguishes the essential components of this process.

The EinsTuner software tool is used for device width, and hence area, optimization [7–10]. EinsTuner is a gradient-based optimization engine for static timing. For the System z9 design, an improved version of EinsTuner with a new minimization engine [11–13], coupled with

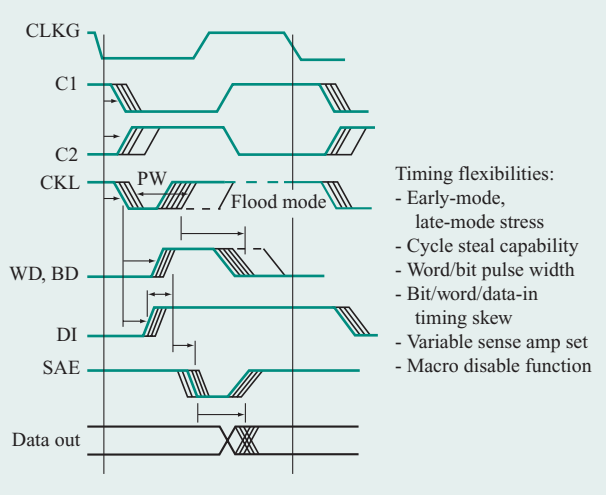


**Figure 2**

Programmable array timing control [14]. Each array macro contains extensive internal timing control to ensure hardware functionality. The input boundary latches, the bit and word decode paths, the data-in path, and the sense amplifiers are all supplied with separate flexible timing engines for delay, pulse width, and skew adjustments. (CLKG: global clock; CKL: local clock generator; PW: pulse width; WD: word decode; BD: bit decode; DI: data in; SAE: sense amplifier enable. Flood mode is used to change internal settings for early-mode or late-mode timing stressing.)

parallelization, yielded a drastic reduction in runtime. Parallelization (the ability to split a job into multiple pieces that can be executed in parallel on different processors) enabled larger macros to be processed.

EinsTuner optimizes the device width of the transistors using different optimization routines. Several constraints, such as beta ratio, pin capacitance, or area, can be applied, but EinsTuner does not change device types. (*Beta ratio* is an industry-standard term that refers to the ratio of the p-FET device width to the n-FET device width in a given CMOS logic book. The dc switching characteristics are related to beta ratio.) After optimization, the best fit book fit from the standard cell library is determined on the basis of the new transistor sizes. Because the standard cell library used for RLMs is fine-grained [14], the loss of precision is minimal. Changes in device size cause local perturbations of the placement, and such perturbations are handled in an incremental mode to attain a legal placement.

Traditional slack optimization targets only the worst paths, possibly leaving other paths with negative slack. TPS works on all critical paths in order to improve slack and slew, but keeps to the area budget by relaxing noncritical paths. Therefore, EinsTuner TPS is used for timing optimization, keeping leakage bounded.

When optimizing for area, traditional EinsTuner optimization works on all paths, possibly making some critical paths worse. FAR retains the critical paths while relaxing the noncritical paths by reducing the device widths and thus reducing leakage.

Two additional programs which complement the EinsTuner optimizations perform low-$V_t$ conversion and high-$V_t$ conversion. Both keep the area unchanged, as constrained by the available cell library books. Here, each book has an implementation in nominal $V_t$, low $V_t$, and high $V_t$, with the same physical book sizes and topology that enabled simple switching of the device type. The application programming interface (API) and the incremental timing available using the static transistor-level timer EinsTLT [15] enable both programs to obtain immediate timing results after device types are changed. Some books are converted to high-$V_t$ devices to reduce leakage power, whereas other books are converted to low-$V_t$ devices to meet the performance goals for the critical paths.

An additional program was used for RLMs and custom macros in order to identify possible device type conversions. For this purpose, EinsTLT results were parsed, and for each transistor, the slack and slew were retrieved. Transistors having a slack greater than or equal to the positive slack threshold and having a slew less than the slew threshold were listed as candidates for high-$V_t$ conversion, and transistors having a slack less than or equal to the negative slack threshold as candidates for low-$V_t$ conversion. Optionally, the device type changes could be implemented in the schematics by replacing the cell books with high-$V_t$ or low-$V_t$ cell books. Books that drive primary outputs were untouched in order to maintain slew on external paths. Here, re-extraction and re-timing were necessary, in contrast to the cases involving the high-$V_t$ and low-$V_t$ conversion programs described above.

### Basic cycle-time work

After the technology migration (from CMOS 9SG to CMOS 10S0), early timing analyses revealed two main categories of cycle-time-limiting paths. The first category involved the performance-sensitive cache arrays on both the L1 cache of the CP chip and the L2 cache and L2 directory on the system control data (SCD) chip. Programmable timing features (**Figure 2**) for these arrays had already been introduced in the System z900 processor [14]. By fully exploiting these timing controls, it was possible to balance the read and write paths of the arrays and set the cycle-time target for the whole chip.

The second category of cycle-time-limiting paths became apparent when the performance gain of the wires did not improve as much as the device speed increased. This challenge required us to add two more high-

performance wiring levels to the back end of line (BEOL) mask set in CMOS 10S0. Multiple-wide data buses (64–144 bits) that crossed long distances over the cores had to be re-engineered, and, in addition, some of the bi-stack wiring solutions inside the execution units had to be moved to better-performing metallization levels in order to support the cycle-time target. With the help of extensive PowerSPICE simulations, the most effective wiring solutions for those critical paths could be identified, and, after manually placing the necessary repowering buffers, the obtained width and level wiring constraints were passed to the wire-routing CAD tool.

Our approach included a means of floorplan tuning and logic re-partitioning to achieve a minimum of global and local wire delay in the critical paths. Two independent methods were used to address this problem. The first method is called *C3 checking*. Using PMPR values to model wire delays and slews in the early phases of the project enabled the checking of the scaled C3 value, which is a calculated, representative measure for wire *RC* delay and slew degradation. By introducing individual C3 targets for the different kinds of test-signal categories (e.g., ac, ac-test, ac-asynchronous, dc, scan) and by being sensitive to the specific characteristics of these categories, the team was able to effectively optimize the reduction of wire delays early in the design.

The second method for minimizing wire delays required us to inflate all chip, core, unit, and wire delays by a constant factor and then generate a priority list of the paths most affected by *RC* delays; such paths were candidates for manual wiring solutions.

### Learning with early hardware

In the absence of hardware-based design models, a thorough analysis of the early hardware was essential to reducing subsystem cycle time. Various performance screen ring oscillator (PSRO) circuits addressing all three $V_t$ types, small and large devices, and all basic logic-gate structures were included in the design. Measurements from these PSRO circuits were crucial in modifying the design models to more accurately reflect the actual performance ratios of the different $V_t$ types.

In addition to these measurements, we performed an extensive analysis of leakage and power with respect to performance, which was also measured using the on-chip PSROs. The evaluation of these measurements across the silicon-process window provided an early estimate of the actual leakage versus performance characteristics, which then established a baseline for potential line centering. (The phrase *silicon process window* refers to the natural, overall variation that may arise through all of the different steps in the silicon processing.) Accordingly, the design models were adjusted to reflect this baseline in the timing methodology for the final design phase.

Early logic built-in self-test (LBIST) selective signature generation was performed at the wafer and chip levels. This technique allows early identification of long delay paths prior to the system-level evaluation. Correlating these results with those obtained by system-level characterization provided a clear, comprehensive list of critical paths to be improved in the final design phase. The critical paths identified by LBIST and system-level characterization indicated that the performance sort capability was limited by a high percentage of *RC* delay in these paths. As a result, the timing corner for the final design phase was shifted to a higher voltage and a more aggressive line centering to increase the speed of devices, with the goal of exposing paths with excessive wire delay, because *RC* delays are invariant with respect to both voltage and transistor channel length. (The term *timing corner* refers to a set of conditions used for timing; these include supply voltage, temperature, and line centering.) This work was greatly assisted by an automated analysis of wire delays with respect to device delays for each timing path, uncovering a large number of high-delay paths for wires inside synthesized macros that had to be addressed. This methodology originated in the pSeries* CP design and was adapted for the zSeries* CP design.

The characterization of the early hardware was problematical because of significant systematic across-reticle linewidth variation (ARLV) observed on the hardware. The processor chips were manufactured using a two-chip reticle, and both chips had systematic across-chip linewidth variation (ACLV); however, the variation was significantly different between the chips, further complicating the characterization. In order to improve the ACLV for the production hardware, we used a single-chip reticle, and the chip and adjacent kerf structure both used an improved fill methodology to ensure a more consistent shape density across the reticle for all FEOL masks. A new method was developed to correlate hardware results with respect to PSRO data in order to remove the localized ACLV from the timing characterization; however, this methodology was not ready to be used in time to influence the final hardware design.

The IBM process-development organization assisted in all early hardware characterization and evaluation work, and our close cooperation was very beneficial in determining which issues could best be resolved in chip design and which issues could be addressed by technology process tuning.

The innovations and enhancements in the applied microprocessor design methodology were very useful. The NBTI-induced degradation of critical paths was significantly reduced, and the approach that included the NTBI effect in the design methodology returned measurable results, as discussed earlier in this paper.

Aside from reducing the sensitivity of BEOL variations on the cycle time, the applied *RC* uplift methodology also lowered the noise sensitivity of the wires with respect to timing and functional failures early in the design phase. Assisted by a timing-based algorithm and an automated tool, the introduction of high-$V_t$ devices into logic circuits yielded a noticeable usage of these devices and, along with the reduction of low-$V_t$ usage, this made an important contribution to reducing the chip static power dissipation without having an impact on the final cycle time.

Driven by our learning with early hardware, the timing methodology for the final design phase was adjusted with respect to multiple parameters, and the resulting sort capability of the production hardware was significantly improved.

## Nest design considerations

Many challenges existed in connection with making the nest chip designs meet our cycle-time target even with the slower nest cycle time. (Nest chips are system control elements that include the L2 cache and the main storage control.) Common random logic macros (RLMs) are used across all three chips and multiple instances of the same RLM on a single chip. Making the physical implementation of a single logic function work in multiple instances required new methodology for the System z9 nest designs. Innovations existed with respect to the nest I/O area, with the addition of the Elastic Interface [1]. Some of the solutions to the new challenges are described below.

### *Additional cycle-time margin*

As in past System z platforms, the nest chips run at twice the processor's chip cycle time. To ensure that the dual-core processor would be the cycle-time-limiting chip, the nest target cycle time was made faster by using one-sigma process variation. Each sigma process variation changes the cycle time by 3.5%, so the target cycle time was set to 1.93 times the processor cycle-time target. This strategy reduced the hardware characterization and tuning effort significantly, and no cycle-time-limiting paths were ever found in the nest chips. In addition, this reduced the risk associated with the process line centering and sorting by reducing the degree to which the nest chips had to be pushed to achieve higher speeds.

### *Optimized SCD chip floorplan to reduce L2 cache latency*

One of the traditional critical timing paths on the system control data (SCD) relates to the delivery of cached data to the processor chips. In order to maximize system performance, it is crucial for the nest chips to immediately dispatch the target doubleword of the requested cache access, followed by the remaining words that comprise the cache line. For the System z9, the bandwidth of each SCD chip is one doubleword per nest cycle. Two SCD chips work in tandem to distribute two doublewords every nest cycle, which equates to one doubleword for every CP cycle. In order to achieve the aggressive frequency goals while maintaining the necessary bandwidth, we made two key decisions early in the design phase. First, the doublewords were partitioned by even and odd words across the pair of SCD chips. This allowed word 0 and word 1 to be accessed simultaneously across both SCD chips on the first CP cycle. Words 2 and 3 follow on the second CP cycle. Because all four words are actually read out of the cache arrays in parallel, words 2 and 3 are afforded more time to propagate to the data port. For the second design decision, in order to address the challenge of long distances of the cache arrays from the processor data ports, we employed an innovative solution to exploit this difference in timing requirements between the two words on a given SCD chip. The communications between data ports and the processors use full-speed Elastic Interfaces [1] that are capable of implementing a late send for the second word leaving the chip. In order to take full advantage of the late send option, we significantly reorganized the floorplan of the data arrays and the I/O ports, compared with previous systems in which the input and output for a particular processor were next to each other. On the System z9, the output ports were placed along the left and right edges of the chip, as close to the horizontal centerline as possible, while the input ports were placed farther from the centerline. The location at which the data arrays were physically placed depended on whether they were storing the first word (word 0 or word 1) or the second word (word 2 or word 3). The first word arrays were placed along the bottom and top of the chip, as close to the vertical centerline as possible. The second word arrays were placed farther left and right of the vertical centerline than the first word arrays, since that data had an extra half cycle to arrive. The end result was that any first array data was able to reach any CP I/O port in less than one cycle, and any second array data to reach any CP I/O port in less than one and one half cycles. This structure eliminated the additional L2 latency that the floorplan from previous systems would have required.

### *Design approach in order to minimize power*

Approaches similar to those used for the dual-core processor chip were used to reduce power in the nest chips. **Table 5** shows the improvements in low-$V_t$ and high-$V_t$ usage for the System z9 nest chips with respect to the z990 nest chips.

### *Enhancements to macro design methodology*

RLM synthesis on the system cache controller (SCC) employed an edge-pin-based orientation on approximately 90% of the macros. Interior pins (i.e., pins located inside the physical boundary of the macro) were used on the remaining 10%. Interior pins had three main advantages in this design. When used on the more complex, large RLMs, interior pins minimized the occurrence of large *RC* delays by allowing driving and receiving books to be placed as close to their pins as possible. Because macro wires tended to reside on the lowest physical layers in the design, edge-pin-based solutions often suffered from large *RC* delays. This problem was magnified when macro wires were scenic. (The term *scenic wire* refers to a wire that meanders around the macro instead of taking the shortest path possible, known as a Manhattan route.) The use of interior pins drastically reduced the chance that a macro wire connected to a macro pin would have *RC* problems. Interior pins also resulted in optimal placement during the synthesis process by allowing the pins to be placed near their associated logic blocks instead of having to be placed at the edges of the macro. The secondary effect of this was better overall macro timing. Interior pins also removed the synthesis inefficiencies incurred from multi-use macros. Macros that are reused in a design often suffer from gross discrepancies in their pin assertions due to various geometric properties of the design, and these assertions in turn affect the synthesis results. Interior pins make a multi-use macro independent of the chip integration, thus allowing the designer to achieve consistent timing results across all instances.

A new capability, RLM cloning, was implemented in order to enable multiple physical clones of a single functional macro. This process allowed a single VHDL (hardware description language) model to be mapped to multiple physical implementations, and the process was used in several different ways.

One application permitted aspect ratios of different dimensions on different chips. In one case, a common RLM which was used across three chips originally had a square aspect ratio. On one of the chips, the square aspect ratio prevented the RLM from being placed optimally. The timing could not be achieved with the suboptimal placement. For another chip, the logic was cloned into a new physical implementation which had a short-to-wide aspect ratio. This allowed paths through this RLM to achieve cycle-time targets and reduced the use of higher and wider wire resources.

Wiring resources for a given metal level can be shared between an RLM and the chip. This sharing is managed through contracts that allocate certain wiring tracks to the RLM while the other tracks are allocated for chip routing. In numerous cases, RLMs are used on multiple chips, and in multiple instances on a single chip. Common RLMs that share the same wire layer with the chip can cause routing problems at the chip level. These routing problems can lead to scenic routes that fail to meet cycle-time targets. Cloning the RLM allows different instances to have different wiring contracts, and each wiring contract is then optimized for routability and timing.

**Table 5** Comparison of low-, nominal-, and high-$V_t$ device usage in nest chips for System z9 and System z990 (excluding arrays). (MSC: main storage controller; SCC: system cache controller; SCD: system control data.)

| | *MSC* | | *SCC* | | *SCD* | |
|---|---|---|---|---|---|---|
| | *z990* (%) | *z9* (%) | *z990* (%) | *z9* (%) | *z990* (%) | *z9* (%) |
| Nominal $V_t$ | 91 | 55 | 76 | 61 | 93 | 74 |
| Low $V_t$ | 8 | 2 | 24 | 9 | 6 | 5 |
| High $V_t$ | 1 | 43 | 0 | 30 | 1 | 21 |

The three nest chips are released to the foundry at staggered times. Several cases existed in which we found timing problems on a common RLM on chip B after it had already been released on chip A. Cloning allowed the team to fix the problem for chip B without releasing a new VHDL model to the verification process and affecting the verification schedule.

RLM pin placement is a final example of the use of cloning to improve the design. Pin placement affects synthesis and gate placements inside the RLM, RLM routing, and chip routing. With cloning, the pin placement of different instances of an RLM can be optimized for each chip placement. This reduces wrong-way wires (ones that run perpendicularly to their designated orientations) and improves chip timing.

## Hardware characterization and tuning

As we suggest throughout this paper, hardware characterization and cycle-time tuning are intense efforts to guarantee that the chip runs reliably across the silicon process window and at possible voltage and temperature conditions. During this tuning phase of the project, no serious technology or design marginalities were encountered in the hardware. Therefore, the design team could concentrate on the hardware cycle-time tuning even more than during the development of previous System z platforms. Very close cooperation within the hardware tuning team and with the logic, circuit, timing, and technology team was crucial for this very successful design effort.

### *Methodology*

We ran specific system hardware verification test programs while reducing the cycle time in steps until a

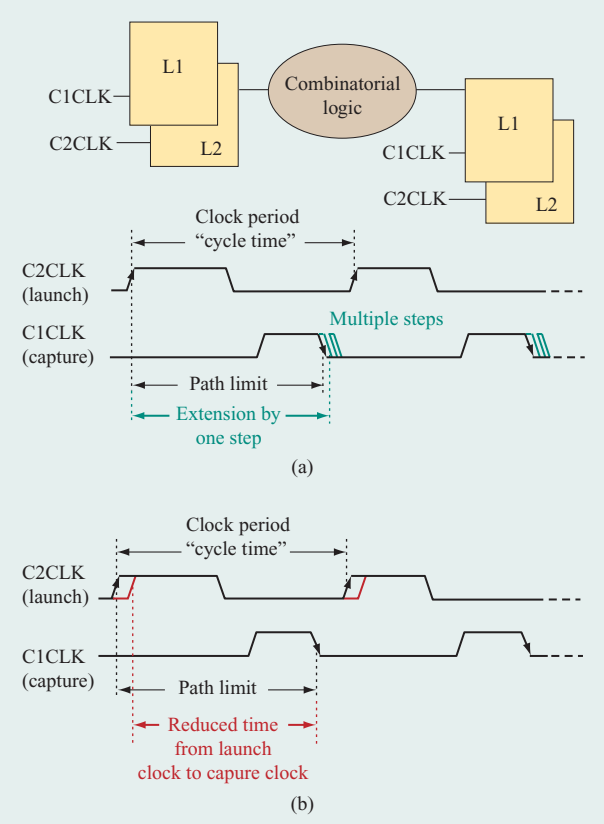


**Figure 3**

(a) Critical path alleviation. The figure also shows a typical latch-to-latch path (top) and provides a graphical definition of the term *cycle time* (bottom). (b) Critical path aggravation. [Parts (a) and (b) adapted from [17].]

logic-path timing marginality was encountered that caused a malfunction. In System z platforms, these fails correspond to turning on hardware checkers, which are key hardware features, to detect any functional problems. For the purposes of checking, the execution units are duplicated on the CP and compared in every cycle to ensure equivalent results. All other logic functions are protected by parity, illegal state, and other checking mechanisms to guarantee the superior reliability, availability, and serviceability (RAS) characteristics of mainframes [16].

In a system that runs normally, a hardware checker that indicates a malfunction triggers a recovery action. During the hardware test and characterization phase, a setup is chosen that will stop the clocks in order to prevent further instruction processing. This allows the analysis of the failing function by extracting the state of the machine and by using trace arrays, which record the executed instruction stream. On the basis of this information, potential logic paths that could fail to meet the cycle time can be identified. These are compared with the timing report to determine which paths are the most critical. If the timing models used to calculate critical paths accurately reflect the actual hardware delays, a good match between models and hardware can be found. This design effort required a very close communication with the logic, circuit, and technology design teams.

Once potentially failing paths are identified, we can start the hardware cycle-time tuning to remove them. The clock-generation circuits include a so-called clock-mode block. The mode can be set to allow some flexibility in the capturing and launching of a path (see **Figure 3**). Each macro on the chip includes one clock-mode block that controls all of its latches and registers. These are built using the master–slave concept. The master latch is clocked by the so-called c1 clock and the slave latch by the c2 clock. A rising edge of a c2 clock launches a signal via the slave output that propagates through combinatorial logic to the next master latch. This captures the signal at its input with the c1 falling edge [17]. The basic mechanism is illustrated in Figure 3.

A critical path can be extended by setting up the clock-mode block such that it moves the falling capture c1 edge out in one, two, or three steps. This extends the time to capture the signal at its data input accordingly. Each step represents about 20 picoseconds in the given technology. Figure 3(a) shows the alleviation of a critical path timing failure. (Because the capture clock is delayed, the signal has more time to propagate to the capture latch, which would alleviate a failing path that was just at the edge of the timing margin.) Once such a setup is defined, the system hardware verification test program is run again and the cycle time reduced, as described above. If indeed the previous critical path is alleviated, another path becomes marginal, and a different hardware checker turns on at a faster cycle time.

The next step is aggravating the failing path as shown in Figure 3(b), in which the c2 rising launch clock of the macro is delayed at the latch where the failing path originates. This setup reduces the time for the path to propagate, causing a malfunction at a slower cycle time. If the original checker turns on again, it can be assumed that the analysis of the failing path and the alleviation is correct.

This process is repeated until a failing timing path is found that cannot be improved by adjusting the capture or launch clocks. At this point in system testing, it is very important to stress the hardware with these clock settings at all possible voltage and temperature conditions and across the silicon process window to guarantee that no other marginality is introduced.

Early-mode margins must also be checked. In cases of an early-mode problem, a latch/register stage is skipped,

causing serious malfunctions. This can happen when there is insufficient combinatorial logic delay in a path while both c1 and c2 clocks are on. A signal, launched by the c2 clock rising edge, could propagate through the combinatorial logic and flush through its receiving latch, because both master and slave latches are active in order to capture data.

Typically, a design has provisions to compensate for clock skew in order to avoid these kinds of problems at nominal clock settings. However, to facilitate hardware-cycle tuning, additional early-mode margin has been built into the design. This margin is reduced by moving the c1 falling edge past the c2 rising edge to alleviate critical paths. Therefore, thorough testing with the maximum possible overlap between the c1 falling edge and c2 rising edge is needed at all corners to ensure that the hardware runs reliably under these conditions as well. This is especially important at higher voltages, where early-mode problems are more likely to occur.

### *Broad-based cycle-tuning approach*

After a specific clock-tuning setup is found for the most critical timing path, it becomes more difficult to find the next-longest critical path because the number of timing-equivalent critical paths increases as the cycle time is reduced. To simplify the cycle-tuning effort, we chose a broad-based approach of delaying the capture clocks (alleviating) for all of the macros on the chip at once. With this setup, the fastest possible cycle time can be determined. This bottom-up method was fairly successful and resulted in 10–15-ps cycle-time improvement over the previously removed cycle-limiting paths. It is not desirable to leave all macros on the chip in this setup because of the risk of early-mode problems as described above. On the basis of the timing report, less-critical macros were set back to nominal clock settings. To validate these settings, this set of macros was cycled down as before, and any macros that did not fail were clearly not timing-critical. The goal of this hardware cycle-time tuning phase was to find a minimum set of macros, which required moving the c1 falling capture clock edge to achieve the fastest possible speed. Until these macros were determined, it was very helpful to test the hardware with the full set of all macros alleviated to reach a maximum possible speed and to also test the hardware with the reduced set for comparison.

As mentioned earlier, more and more paths become critical at the same time, as the cycle time is pushed to the fastest limit possible. Therefore, more and more paths become sensitive to clock, noise, and voltage tolerance variations.

### *Hardware cycle-tuning efforts*

The major clock-tuning effort on the first available hardware version was to determine the top-cycle failing path, using the previously described method of alleviating and aggravating critical paths. These findings drove several corrective actions for the next and final silicon hardware version.

Once the final hardware became available for testing, all alleviates for the top paths in the early hardware were removed to find out whether the previous critical paths were all corrected and no longer limiting cycle time. The results were very encouraging because none of the improved paths were found in the top failing paths.

At this stage in the hardware tuning effort, the bottom-up method as described earlier was applied to the full extent to determine the minimum set of alleviated macros needed to provide the fastest cycle time possible. We had to make determinations with respect to cycle time very quickly to allow time for parts to be thoroughly characterized at all corners in order to guarantee reliability. All of the hardware was first tested with all macros that were tuned to find the reference for maximum speed, followed by testing with the currently available reduced set. If the reduced set was sufficient to cycle down as far as with the full set, the reduced set was used further.

With this approach, it was later also possible to add another macro to the tuning set. No retesting of already available and tuned hardware was required because the added macro was already in the set that determines the fastest cycle time possible.

Nevertheless, it is critical that the clock-tuning settings be applied to as many processor subsystem modules as possible to validate the manufacturing performance sort settings and to avoid problems in the assembly and test of manufacturing parts, as more and more parts become available.

### *Correlation of failing paths with chip speed*

The critical paths on the System z9 were expected to be localized internally to the CP chip. Traditional techniques, such as deep sorting and raising voltage, were evaluated. Since leakage power increases exponentially with both voltage and sort depth, power constraints limited these options. As a result, an aggressive effort was taken to discover and fix as many critical paths as possible within the allotted schedule time. LBIST at wafer level, together with system-level functional patterns, was used to correlate critical paths with a chip-speed metric defined as the average of the 12 on-chip PSROs.

LBIST is run at full speed on the CP chip and, by design, the test coverage is very high (see **Table 6**). High test coverage implies that almost all paths are covered by the test pattern. Several potential complications exist with respect to critical path correlation. LBIST may not test system critical paths, especially when the logic is deep, because many latches must be set to specific states to

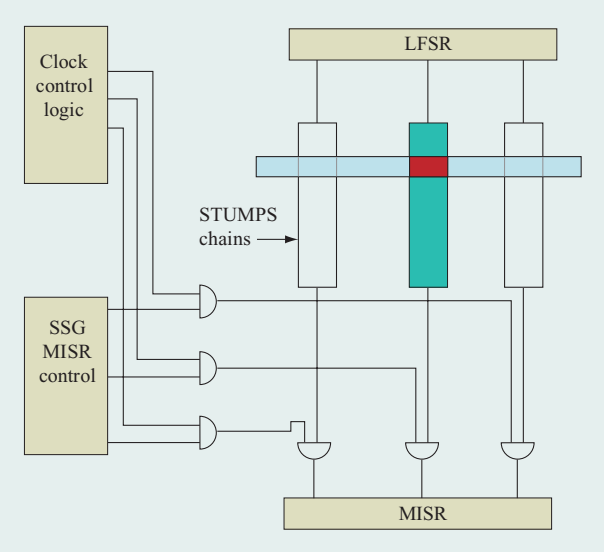


Figure 4

Selective signature generation structure. (LFSR: linear feedback shift register; MISR: multiple-input shift register; SSG: selective signature generation.)

**Table 6** CP LBIST test coverage. (1g refers to a single clock sequence (C2–C1); 2g refers to a double-clock sequence (C2–C1–C2–C1); 4K indicates 4,000 patterns.)

| *Test* | *dc* (%) | *ac* (%) |
|---|---|---|
| LBIST 1g4K | 93.5 | 86.4 |
| LBIST 2g4K | 96.5 | 92.2 |
| LBIST (all other 4K) | 98.5 | 96.0 |
| LBIST > 4K | 99.3 | 97.9 |
| LSSD | 99.8 | 99.2 |

exercise the path. LBIST may test paths that are nonfunctional and that are slower than critical paths, thus effectively masking them. The scanning and switching activity of LBIST may cause a transient power-supply drop and resultant lower performance. In practice, we were able to circumvent each of these complications. Weighted LBIST was used along with standard LBIST tests to determine which latches were the most critical. The process of weighting LBIST tends to load latches with more zeros or ones than standard LBIST, which has a 50% probability of loading with a zero. This tends to exercise paths that are more resistant to random patterns. Clock controls are available to block paths to every logic macro (several hundred macros per chip) to help remove noncritical paths. An "idle" time, added after LBIST scan initialization, decreased voltage drop during the critical multiple-input shift register (MISR) compression time.

LBIST has several powerful advantages over a functional critical-path test approach. LBIST can be run at the wafer level, weeks sooner than functional testing. For the System z9, functional testing is limited to the system level, which is more complicated than chip testing. All components must function faster than the CP chip, and non-CP system components may limit test conditions. When testing the wafer, it is possible to test dozens of chips under multiple test conditions (e.g., voltage, temperature, cycle time) in days instead of months. LBIST has broad fault-type coverage and includes transition faults. Finally, LBIST has excellent diagnostic capability and makes possible a direct readout of all failing latches.

Selective signature generation (SSG) was used to determine the failing cycle time of every latch within a specific range of the highest failing cycle time. SSG gates STUMPS data (LBIST scan channels) into the MISR according to channel and by clock. [STUMPS is a multilevel acronym that stands for Self-Test Using MISRs and Parallel SRSGs (shift register sequence generators).] In **Figure 4**, the latch data in all LBIST STUMPS chains is compressed into each MISR on every clock in a normal LBIST test. At the end of the test, the data in the MISR comprises a "signature." The signature of a "good" chip will match the simulated signature. If even one latch fails to match the expected value even once during the test, the signature at the end of the test is a mismatch, indicating that the chip is "bad." Selective signature generation modifies what is compressed into the MISR according to either channel (e.g., green shading in Figure 4), clock range (stump bit range in blue shading), or both (red shading). SSG uses a golden signature approach to identify critical-path capture latches. (The term *golden signature* is used to indicate a defect-free signature.) In the simplest SSG configuration, a single latch is compressed into the MISR for a specific LBIST test. Once this is accomplished, the signature no longer matches the simulated good signature, so a golden signature is determined by the MISR value at the end of the test under relaxed conditions. The cycle time is then decreased until the signature does not match the golden signature. The failing cycle time is then uniquely associated with that single latch. In practice, a search algorithm is used to determine all critical latches within a specified range of the slowest failing cycle time.

The output from SSG is used to generate a table of critical latches and sensitivities to voltage and PSRO delay. This list was reviewed with the design team to ensure that all latches were capturing true functional paths and not "false" LBIST-only paths. **Figure 5** shows the delay of selected latches as a function of PSRO. This

is used to determine the required clock adjustment for each latch at any potential PSRO sort point. These latch clock adjustments agreed well with the adjustments determined at functional system-level clock tuning.

SSG determined the delay of critical capture latches; however, we do not know what pattern in LBIST caused the cycle-time fail, what latch launched the fail, nor what the specific path was through the chip. Further diagnostics were done to identify the logic macro (a logic hierarchical level) that launched the data when it failed. Each failing latch was traced back to all source latches. Source latches are usually located in several macros. Each macro has independent clock controls that allow launch and capture clock timing adjustments. The source macro launch clocks were each delayed until the capture latch failed at a slower cycle time. The final list sent to the design team included a failing latch, a failing cycle time at the projected sort point, and the source macro. In most cases, using this approach, the suspected failing critical path, and the amount it must be improved, could be identified.

SSG was performed on multiple-process split chips to intentionally vary polysilicon gate length as well as n-FET and p-FET $V_t$ to determine the sensitivity of critical paths to these parameters. SSG helped identify unexpected sensitivity to high-$V_t$ circuits (used for power reduction in early-mode timing padding) as well as low-voltage sensitivity.

### *Sort point determination*

Every customer's application, environment, and needs are unique. Most large-system customers require high availability of their systems, and platform failure must be extremely infrequent. During the manufacturing process, systems are tested using conditions beyond those any customer would actually experience. These stresses involve voltage, cycle time, and temperature. The goal of our manufacturing tests is to protect the customer from various forms of degradation, system clock variation, and manufacturing variation.

Chips are sorted for performance on the basis of several tests. The sort point is defined as the slowest acceptable chip speed (chip PSRO delay in this context) to guarantee a given system performance. Determining the sort point is a delicate balance that must take into consideration manufacturing cost and the potential value to the customer. A faster sort point may contribute to a faster system, providing more value to the customer; however, this may result in a larger number of slow but otherwise good chips being discarded. Relatively low-volume (in terms of number of systems sold), high-value systems such as the System z9 should always be sorted to some extent, but there are realistic limits on the sort depth. The "width" of the sort window from the fastest

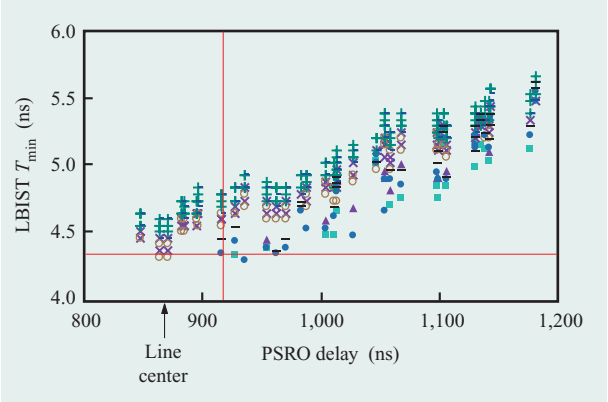


**Figure 5**

LBIST failing cycle time for specific latches, indicated by different colored symbols, as a function of average PSRO. Red crosshairs indicate target PSRO sort point and cycle time.

to the slowest allowable chips must be wide enough to produce a reliable supply of chips. Sorting must be done before chips are mounted in MCMs. In our case, sorting is done at the wafer level. The fast limit is defined by an MCM power limit and is, in effect, a power sort based on quiescent current, $I_{ddq}$, not chip speed. The upper $I_{ddq}$ limit is determined as follows: 1) An absolute MCM power is defined by the power and thermal team; 2) MCM power for many MCMs is plotted against the sum of chip wafer $I_{ddq}$ for each of the chips on the MCM to establish a total wafer $I_{ddq}$ limit; 3) $I_{ddq}$ limits are apportioned to each of the eight CPs, four SCDs, two MSCs, one SC, and one CLK chips on the MCM. In addition, two power sort bins for the CP and SCD chips exist, called fast and standard, to help limit total MCM power. Every MCM must have at least two standard (lower-power) chips of each type. **Figure 6** shows a plot of $I_{ddq}$ vs. PSRO delay for the CP chip. Effectively, the upper $I_{ddq}$ bound translates to a 770–780-ns PSRO lower bound.

The sort point may be defined using various speed metrics, such as architecture verification pattern (AVP) cycle time, LBIST cycle time, flush delay through the latch scan chain, or average PSRO delay. Functional AVPs were not available at the wafer level, and PSRO delay provided the best predictor of system cycle time. **Figure 7** shows a plot of system cycle time vs. average PSRO delay. System cycle time is determined using a large set of functional patterns developed over many years. Cycle time is decreased until a fail occurs. The fail must be traced to a CP chip. If the fail is not due to a CP chip, the system or MCM has other problems that must first be addressed, and that data point cannot be used for sort point calculation. The failing CP is then disabled

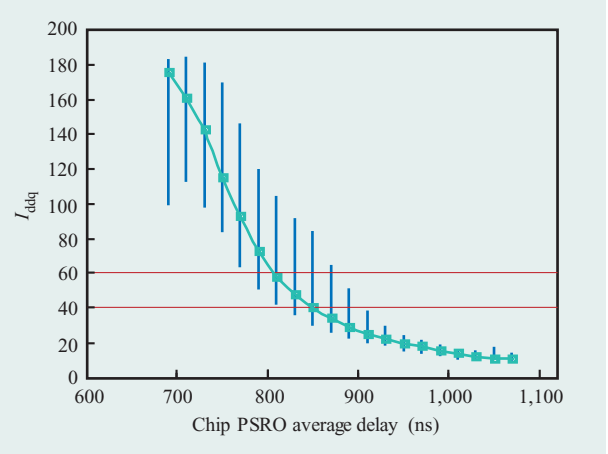


**Figure 6**

CP $I_{ddq}$ vs. PSRO delay showing fast (41–60 A) and standard (<40 A) bins. The data in the chart represents thousands of chips. The vertical bar at each data point represents a range of $I_{ddq}$ values for each 20-ns range of PSRO delay values.

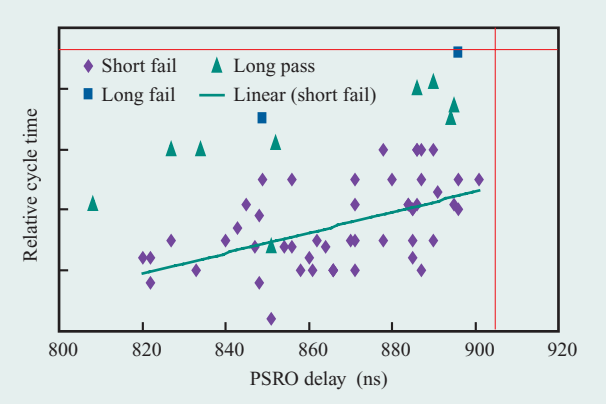


**Figure 7**

System cycle time vs. CP average PSRO delay. All points correspond to CP fails. Crosshairs indicate desired cycle time and resultant sort point.

and cycle time is decreased until the next CP fails. This process is repeated until as many CPs as possible are tested. In order to collect many data points, the dwell time at each cycle-time point is relatively short (roughly ten minutes). Extended tests are run on a small number of MCMs in order to estimate a long-run failing cycle time from a short run.

## Concluding remarks

The System z9 processor subsystem cycle time was achieved through the combined design, characterization, and tuning efforts outlined in this paper, as well as through the technology enhancements in IBM 90-nm CMOS SOI technology [18]. No single effort dominated the cycle-time gains; however, the overall focus on power and leakage enabled unexpected gains through silicon line centering, chip sorting, and increasing the processor subsystem supply voltage. This focus was, perhaps, the strongest area of collaboration between the process and chip/subsystem design teams, while the combination of lower device leakage and device-dominated cycle time brought about significant improvements in cycle time. Because we initially projected that cycle time would be limited by cooling capacity, our focus started on high-level design and continued through the building of the production platforms. Our project touched on all facets of development, process, design, and hardware evaluation at all levels.

## Acknowledgments

Many individuals have been involved in the cycle-time enhancements for the System z9 processor subsystem, from its earliest conceptual stages, through chip and subsystem design, process development, hardware evaluation and tuning, and silicon and system manufacture. Some of the key contributors are listed here: Dan Poindexter, Scott Stiffler, Phil Wu – silicon process development; H. J. Nam – model evaluation for sorting and leakage, power assessments; Ee Cho, Larry Lange, Gebhard Weber – device $V_t$ optimization algorithms and code; Don Rozales, Adil Bhanji – timing accuracy verification of PMPR-versus-PDM-based wire models; Joachim Clabes – consultation on design methodology and design-technology interlock; Howard Smith – 3DX and noise methodology; David Rude – low-power standard library books and RLM build support; Gary Van Huben – SCD floorplan and macro cloning methodology; Dave Webber – macro cloning methodology; Al Sutcliffe – evaluation and summation of the chip characterization data; Hans-Juergen Muenster, Jim Burns, Bill Landry, John Mance, and Scott Roser – implementation of the vast stream of cycle-tuning experiments; Mike Campbell – general system support and consultation, and dynamic hardware management to test and verify the hardware tuning settings across the multitudes of hardware.

Unfortunately, space does not permit us to mention all of the other individuals who contributed to the System z9 processor subsystem cycle-time achievements, but their efforts are greatly appreciated.

*Trademark, service mark, or registered trademark of International Business Machines Corporation in the United States, other countries, or both.

## References

1. D. M. Berger, J. Y. Chen, F. D. Ferraiolo, J. A. Magee, and G. A. Van Huben, "High-Speed Source-Synchronous

Interface for the IBM System z9 Processor," *IBM J. Res. & Dev.* **51**, No. 1/2, 53–64 (2007, this issue).
2. P. Mak, G. E. Strait, M. A. Blake, K. W. Kark, V. K. Papazova, A. E. Seigler, G. A. Van Huben, L. Wang, and G. C. Wellwood, "Processor Subsystem Interconnect Architecture for a Large Symmetric Multiprocessing System," *IBM J. Res. & Dev.* **48**, No. 3/4, 323–337 (2004).
3. R. M. Averill III, K. G. Barkley, M. A. Bowen, P. J. Camporese, A. H. Dansky, R. F. Hatch, D. E. Hoffman, et al., "Chip Integration Methodology for the IBM S/390* G5 and G6 Custom Microprocessors," *IBM J. Res. & Dev.* **43**, No. 5/6, 681–706 (l999).
4. N. Kimizuka, T. Yamamoto, T. Mogami, K. Yamaguchi, K. Imai, and T. Horiuchi, "Impact of Bias Temperature Instability for Direct Tunneling Ultrathin Gate Oxide on MOSFET Scaling," *Symposium on VLSI Technology, Digest of Technical Papers*, Kyoto, Japan, 1999, pp. 73–74.
5. G. Chen, M. F. Li, C. H. Ang, J. Z. Zheng, and D. L. Kwong, "Dynamic NBTI of p-MOS Transistors and Its Impact on MOSFET Scaling," *IEEE Electron Device Lett.* **23**, No. 12, 734–736 (2002).
6. D. Schroder and J. F. Babcock, "Negative Bias Temperature Instability: Road to Cross in Deep Submicron Silicon Semiconductor Manufacturing," *J. Appl. Phys.* **94**, No. 1, 11–18 (2003).
7. C. Visweswariah and A. R. Conn, "Formulation of Static Circuit Optimization with Reduced Size, Degeneracy and Redundancy by Timing Graph Manipulation," *ACM/IEEE International Conference on Computer-Aided Design (ICCAD), Digest of Technical Papers,* November 1999, pp. 244–251.
8. A. R. Conn, I. M. Elfadel, W. W. Molzen, Jr., P. R. O'Brien, P. N. Strenski, C. Visweswariah, and C. B. Whan, "Gradient-Based Optimization of Custom Circuits Using a Static-Timing Formulation," *IEEE Design Automation Conference, Digest of Technical Papers,* June 1999, pp. 452–459.
9. S. Wolpin, IBM Research Division, "Chip Checker"; see *http://domino.watson.ibm.com/comm/wwwr_thinkresearch.nsf/pages/20020625_einstuner.html*.
10. P. Restle, IBM Research Division, "EinsTuner Animations"; see *http://www.research.ibm.com/da/animation/index.html*.
11. A. Wächter, "An Interior Point Algorithm for Large-Scale Nonlinear Optimization with Applications in Process Engineering," Ph.D. Thesis, Carnegie Mellon University, Pittsburgh, PA, 2002; see *http://www.research.ibm.com/people/a/andreasw/*.
12. A. Wächter, C. Visweswariah, and A. R. Conn, "Large-Scale Nonlinear Optimization in Circuit Tuning," *Future Generation Computer Syst.* **21**, No. 8, 1251–1262 (2005).
13. "Interior Point OPTimizer," Wikipedia; see *http://en.wikipedia.org/wiki/IPOPT*.
14. B. W. Curran, Y. H. Chan, P. T. Wu, P. J. Camporese, G. A. Northrop, R. F. Hatch, L. B. Lacey, J. P. Eckhardt, D. T. Hui, and H. H. Smith, "IBM eServer* z900 High-Frequency Microprocessor Technology, Circuits, and Design Methodology," *IBM J. Res. & Dev.* **46**, No. 4/5, 631–644 (2002).
15. V. B. Rao, J. P. Soreff, T. B. Brodnax, and R. E. Mains, "EinsTLT: Transistor-Level Timing with EinsTimer," *Proceedings of the ACM/IEEE 1999 International Workshop on Timing Issues in the Specification and Synthesis of Digital Systems (Tau-99)*, Monterey, CA, March 8–9, 1999, pp. 1–6.
16. T. J. Slegel, E. Pfeffer, and J. A. Magee, "The IBM eServer z990 Microprocessor," *IBM J. Res. & Dev.* **48**, No. 3/4, 295–309 (2004).
17. R. F. Rizzolo, G. Hinkel, S. Michnowski, T. J. McPherson, and A. J. Sutcliffe, "System Performance Management for the S/390 Parallel Enterprise Server* G5," *IBM J. Res. & Dev.* **43**, No. 5/6, 651–660 (1999).
18. D. J. Poindexter, S. R. Stiffler, P. T. Wu, P. D. Agnello, T. Ivers, S. Narasimha, T. B. Faure, et al. "Optimization of Silicon Technology for the IBM System z9," *IBM J. Res. & Dev.* **51**, No. 1/2, 5–18 (2007, this issue).

*Received March 22, 2006; accepted for publication May 22, 2006; Internet publication December 31, 2006*

**Guenter Mayer** *IBM Systems and Technology Group, IBM Deutschland Entwicklung GmbH, Schoenaicherstrasse 220, 71032 Boeblingen, Germany (guenter_mayer@de.ibm.com).* Mr. Mayer is a Senior Engineer in the IBM Systems and Technology Group, Boeblingen, Germany. He joined IBM in 1994 and has worked on several generations of System z microprocessors as a key designer and technical leader. He is recognized as a global expert in the field of high-performance circuit design and microprocessor design. Mr. Mayer received a master's degree in electrical engineering from the University of Stuttgart, Germany.

**Gerhard Doettling** *IBM Systems and Technology Group, IBM Deutschland Entwicklung GmbH, Schoenaicherstrasse 220, 71032 Boeblingen, Germany (gdoettli@de.ibm.com).* Mr. Doettling is a Senior Engineer, currently working in the IBM Hardware Development Laboratory in Poughkeepsie, New York. He was responsible for the System z9 CP core logic design and the CP hardware cycle time tuning. Mr. Doettling studied electrical engineering at the University of Stuttgart, Germany, and received a master's degree in 1978. The same year, he started work at SEL Stuttgart, Germany, in digital telephone switching system design. In 1981, he joined the IBM Germany Development Laboratory in Boeblingen. Since then, Mr. Doettling has worked as a key designer and as a team and project leader on several generations of IBM servers, CMOS microprocessors, and I/O chip designs.

**Richard F. Rizzolo** *IBM Systems and Technology Group, 2455 South Road, Poughkeepsie, New York 12601 (rizzolo@us.ibm.com).* Mr. Rizzolo is a Senior Technical Staff Member and served as test team leader for several System z projects, most recently for the System z9. He also has primary responsibility for the sort and characterization methodology for MCM chips designed in Poughkeepsie, New York. He received his B.S. degree in physics from Rensselaer Polytechnic Institute in 1977 and his M.E. degree in electrical engineering from Rensselaer in 1980. Since joining IBM in 1978, Mr. Rizzolo has worked on bipolar and CMOS projects in the areas of design for testability, high-frequency design and timing analysis, diagnostics, and circuit design. He holds ten patents and has co-authored a number of papers in the field of testability and diagnostics.

**Christopher J. Berry** *IBM Systems and Technology Group, 2455 South Road, Poughkeepsie, New York 12601 (cberry@us.ibm.com).* Mr. Berry is an Advisory Engineer working in the IBM Hardware Development Laboratory. In 1999 he joined IBM at the Poughkeepsie site working for the System z Microprocessor Design group. He became a chip integrator at the beginning of 2000 and led the physical design and integration for the System z9 processor subsystem chipset. Mr. Berry received a B.S.E.E. degree from Rensselaer Polytechnic Institute in 1999.

**Sean M. Carey** *IBM Systems and Technology Group, 2455 South Road, Poughkeepsie, New York 12601 (smcarey@us.ibm.com).* Mr. Carey is a Senior Technical Staff Member with the System/390* Hardware Development Laboratory. Since joining IBM Poughkeepsie in 1988, he has worked in the hardware development area on several System/390 projects, with emphasis on timing methodology support and development. Mr. Carey received a B.S.E.E. degree from Clarkson University in 1988 and an M.S.E.E. degree from Syracuse University.

**Christopher M. Carney** *IBM Systems and Technology Group, 2455 South Road, Poughkeepsie, New York 12601 (carneycc@us.ibm.com).* Mr. Carney joined IBM in 2001 and is a Staff Engineer in the System/390 Hardware Development Laboratory. He began his career as a logic designer working on the System z990 and is currently the L2 nest timing coordinator for the next-generation mainframe. Mr. Carney received a B.S.E.E. degree from Pennsylvania State University in 1996 and an M.S.E.E. degree from Binghamton University in 2001.

**Joachim Keinert** *IBM Systems and Technology Group, IBM Deutschland Entwicklung GmbH, Schoenaicherstrasse 220, 71032 Boeblingen, Germany (keinert@de.ibm.com).* Mr. Keinert received his M.S. degree in electrical engineering from the Technical University of Stuttgart, Germany, in 1980. He joined IBM in 1979 to work on bipolar circuit design. In 1982, he began work on CMOS circuit tool development and chip-design methodologies. Since then, he has been involved in the development of design tools for all IBM CMOS mainframe processors. His work also covers innovative technologies such as FinFETs, and he holds patents in various areas. Currently, Mr. Keinert is a focal point for circuit design tools for future IBM eServer* processors.

**Peter Loeffler** *IBM Systems and Technology Group, IBM Deutschland Entwicklung GmbH, Schoenaicherstrasse 220, 71032 Boeblingen, Germany (ploeff@de.ibm.com).* Mr. Loeffler joined IBM in 1994 and worked on hardware compression logic design for CMOS microprocessors for System/390 mainframes. In 1998, he moved to the IBM facility at Poughkeepsie, New York, on a two-year assignment, joining the I- and D-cache design team to work on high-frequency design as a unit timing coordinator for the G4 System z processor. Returning to Boeblingen in 2000, he joined the System z990 translator team and was responsible for unit timing coordination. For the System z9 microprocessor chip, he was the chip/core timing leader. Mr. Loeffler received his B.S. degree in communications engineering from the University of Applied Sciences, Esslingen, Germany, in 1994.

**Walter Nop** *IBM Systems and Technology Group, 2455 South Road, Poughkeepsie, New York 12601 (wnop@us.ibm.com).* Mr. Nop is an Advisory Engineer in the IBM Hardware Development Laboratory. In 1988, he joined the IBM facility in Essex Junction, Vermont. He joined the System z microprocessor design group in Poughkeepsie in 1999 as a unit integrator and was integration leader for the System z9 microprocessor. Mr. Nop received a B.S.E.E. degree from Rochester Institute of Technology in 1999.

**Daniel E. Skooglund** *IBM Systems and Technology Group, 2455 South Road, Poughkeepsie, New York 12601 (skooglun@us.ibm.com).* Mr. Skooglund is a Senior Engineering Manager of a processor design department in the IBM Systems and Technology Group. He received a B.S.E.E. degree from Pennsylvania State University in 1982 and an M.S.E.E. degree from Syracuse University in 1988. Mr. Skooglund joined IBM in 1982 at the East Fishkill facility, where he developed and managed advanced VLSI logic and memory test systems. In 1993, he joined an applications support group in East Fishkill, where he worked with the IBM Storage Group on the development of SCSI hard disk controllers. In 1999, he joined the System z processor development group in Poughkeepsie as a design manager

responsible for back-end design methodology, chip integration, and physical design for high-frequency custom microprocessors. Mr. Skooglund was the processor subsystem design manager for the System z9; he is currently working on the next generation of eServer processors.

**Vern A. Victoria** *IBM Systems and Technology Group, 2455 South Road, Poughkeepsie, New York 12601 (vernv@us.ibm.com).* Mr. Victoria is a Senior Engineer in the Systems and Technology Group server development area. He received a B.S. degree in mathematics and physics from Kent State University in 1981. He received his M.S. degree in mathematics from Carnegie Mellon University in 1983, joining the Poughkeepsie server development group that same year as a programmer working on boolean equivalence checking and timing analysis. He later worked as an engineer performing timing analysis and optimization on emulator processor chips. He worked on early SOI evaluation chips and SiGe chips. He is currently the team leader for timing of the memory controller, system controller, and L2 cache. Mr. Victoria has received an IBM Outstanding Technical Achievement Award for S/390* G5 library development.

**Alan P. Wagstaff** *IBM Systems and Technology Group, 2455 South Road, Poughkeepsie, New York 12601 (alanpwag@us.ibm.com).* Mr. Wagstaff is a Staff Engineer in the Systems and Technology Group. He joined IBM in 2002 at the Poughkeepsie site, working on the System z9 product line as an integrator. Mr. Wagstaff received a B.S.E.E. degree in 2001 and an M.S.E.E. degree in 2002, both from Carnegie Mellon University.

**Patrick M. Williams** *IBM Systems and Technology Group, 2070 Route 52, Hopewell Junction, New York 12533 (patricw@us.ibm.com).* Mr. Williams is the manager of the Electronic Design Automation Circuits Department. In 1984, he joined IBM at the East Fishkill facility, where he developed VLSI high-speed memory test systems. In 1992, he joined the advanced CMOS microprocessor team in Poughkeepsie. He was initially part of the SRAM development team, and in 1994 joined the CAD development team in support of the System z line of microprocessors. He has held various lead roles in both circuit and integration CAD methodology and development. Mr. Williams has been involved in many aspects of CAD development in support of high-speed microprocessors, including timing analysis, noise analysis, power analysis, IR drop analysis, electromigration analysis, device and parasitic extraction, chip integration, circuit optimization, electrical circuit checking, and layout automation. He received a B.S.E.E. degree from Pennsylvania State University in 1984. Mr. Williams has several U.S. patents and publications.