On-chip Timing Uncertainty Measurements on IBM Microprocessors

R. Franch, P. Restle, N. James[1], W. Huott[2], J. Friedrich[1], R. Dixon[1], S. Weitzel[1], K. Van Goor[3], G. Salem[4]

IBM Research, Yorktown Heights, NY, [1]IBM STG, Austin, TX, [2]IBM STG, Poughkeepsie, NY, [3]IBM STG, Rochester, MN, [4]IBM STG, Burlington, VT

Abstract

Timing uncertainty in microprocessors is comprised of several sources including PLL jitter, clock distribution skew and jitter, across chip device variations, and power supply noise. The on-chip measurement macro called SKITTER (SKew+jITTER) was designed to measure timing uncertainty from all combined sources by measuring the number of logic stages that complete in a cycle. This measure of completed delay stages has proven to be a very sensitive monitor of power supply noise, which has emerged as a dominant component of timing uncertainty. This paper describes the Skitter measurement experiences of several IBM microprocessors including PPC970MP, XBOX360^{TM}, CELL Broadband Engine^{TM}, and POWER6^{TM} microprocessors running different workloads.

1. Introduction

Understanding timing uncertainty is important in order to know how to properly budget it in a microprocessor's cycle time. The main components of timing uncertainty have historically been PLL jitter, clock distribution skew and jitter, across chip line width variation (ACLV) and power supply noise effects. Often, it is treated as a simple sum of the estimates for these components and is lumped into the cycle time as a guard band. This guard band is time lost in the cycle since it is time that is taken away from doing real combinational work. Being able to directly measure timing uncertainty from all combined sources not only allows for better budgeting but also allows for learning from current designs and can highlight areas of emphasis for future designs.

The on-chip measurement macro called SKITTER (SKew+jITTER) was designed to measure timing uncertainty from all combined sources [1]. With continued scaling, power supply noise effects have become more important in determining timing uncertainty. Power supply noise results in a complex interaction in the delays of both the clock distribution paths and the logic paths. Since these effects are not independent, it makes more sense to measure the total combined effect instead of the delay variations in the clock and data paths individually. The effects from all combined sources of timing uncertainty are felt by the Skitter circuits. This results in a variation of the number of delay stages in the Skitter circuit that complete in a cycle, just as it results in a variation of the number of logic stages that complete in path in the microprocessor. This measure of completed delay stages has proven to be a very sensitive monitor of power supply noise, which has emerged as a dominant component of timing uncertainty.

Skitter circuits have been placed in the core and the nest (non-core) regions on several IBM microprocessors. This paper describes several uses of the Skitter circuits. In addition to measuring timing uncertainty, they have been used to monitor the on-chip clock duty cycle measured at the Local Clock Buffers (LCBs) at the end of the clock distribution network. Skitter circuits have also been used to detect power supply noise events and correlate those events to increased switching activity at specific cycles in the instruction stream. In this usage, an instruction stream that causes a power supply noise event is run repetitively and the Skitter circuit is read out cycle by cycle, effectively recovering the on-chip VDD waveform. This has allowed the study of power supply noise events under different instruction streams (workloads). Fixes for the power supply noise can then be evaluated using Skitter.

These measured results have raised the question of how to best test the complicated interaction between the packaging and decoupling schemes and the varied workloads that run on multi-core chips. Some suggestions for testing for worst-case scenarios of power supply noise events between cores are presented.

2. Skitter Circuits and Operating Modes

2.1 Edge Capture and Accumulate Circuit

Skitter contains a latched-tapped delay line of 129 low fan-out inverters, for the best timing resolution, with a nominal delay of 5-8 ps depending on the technology and threshold voltage. The delay line and sampling latches form an edge-capture circuit, shown in Figure 1, which has been tuned to capture rising and falling edges symmetrically, without preference. An edge is detected in the chain when consecutive inverters have the same output (either both 0's or both 1's). The sampling latches take a snapshot of the state of the inverter chain every cycle. They are regular scannable master/slave latches from the standard cell library and the delay chain inverters are made using low-Vt, regular-Vt, or high Vt devices

1-4244-1128-9/07/$25.00 © 2007 IEEE

depending on the application. Inverters of moderate width are used in the delay chain for better uniformity. The 129 stage length of the delay line can typically fit up to 3 cycles of captured edges. An input mux selects one of 4 signals to be sampled. These inputs are typically either local clocks or remote clocks sent from other Skitter macros placed across the chip. When a local clock is sent down the inverter chain, the sampling latches record the number of inverters that the clock edges travel through

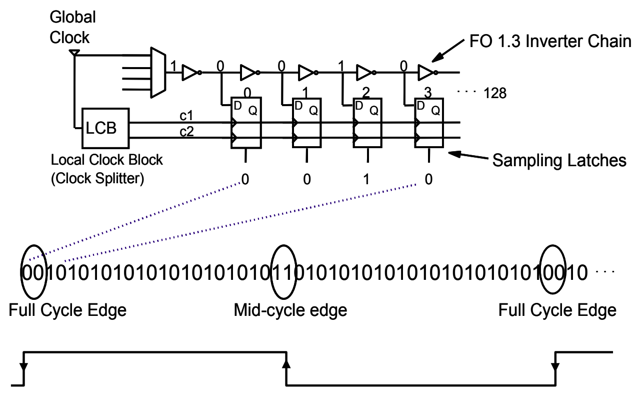


Fig 1: Basic edge-capture circuit using latch-tapped delay line

before being latched.

If the inverter delays are constant and there is no clock jitter, then the clock edges are captured in the same latches every cycle. When there is clock jitter or power supply noise, then the location of the captured edges changes. Since the inverter chain feels the same power supply noise as nearby critical paths on the chip, it responds in the same way as the paths on the chip – it slows down when VDD drops and the edges move to the left since they get through fewer inverters by the end of the cycle. In the event that clock jitter from the PLL or clock distribution causes a short cycle, a similar effect happens. The data edges traveling in a critical path have less time and make it through fewer gates at the end of the shorter cycle. In the Skitter inverter chain, the clock edges also make it through fewer inverters because of the short cycle. So the changing location of the edges in the Skitter sampling latches is a good indicator of the variations in chip timing. Skitter does not distinguish between edge movement from VDD noise or from clock jitter, but gives the bottom-line effect of the timing uncertainty from all sources.

It is easier to see a captured edge in a delay chain as a 1 against a background of 0s, so every sampling latch output is XNORed with its neighbor to obtain a 1 at the location of the detected edge as shown in Figure 2. The result of the XNOR is written into a second register called the accumulation, or "sticky" register. The latches of this register are also called buckets or "bins". By asserting the control signal called "sticky_mode", any latch(or bin) that has ever detected an edge (that has ever been written to a 1) will stay a 1. This is a very useful data-gathering mode, since this will record the worst-case excursions of the

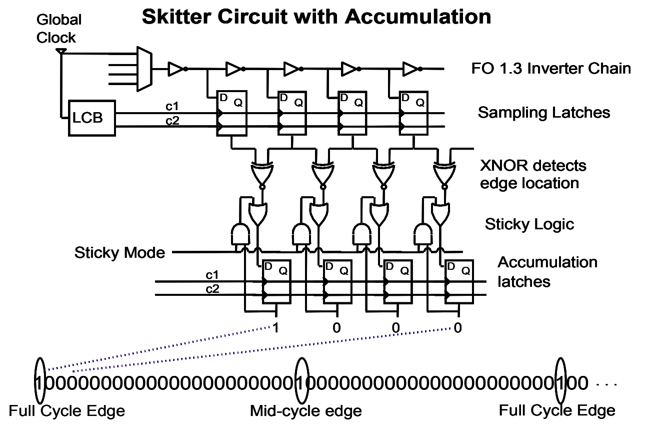


Fig 2: Basic Skitter circuit with accumulation or "sticky" register

edges as they move from bin to bin, after any number of cycles. In this way, the worst case timing uncertainty can be recorded while running any pattern. A workload could be run overnight and the extremes of the short cycle and the longest cycle that has ever occurred during the pattern will be recorded.

The default mode of Skitter operation is single-sample mode, where the sticky register is non-sticky, and new data is written into it every cycle. The earlier example of edges that have moved in a cycle that felt an 8% VDD drop compared to a nominal cycle are shown in single-sample mode and sticky mode in Fig 3. The edges traveled through fewer inverters in the low VDD cycle because the inverters slowed down. This single-sample mode is useful for duty-cycle measurements and in obtaining histograms of the timing uncertainty. In sticky-mode, the bins in between also get filled in with 1s as shown.

Another mode called hold-mode, holds the data in the sticky register constant and blocks any new data from

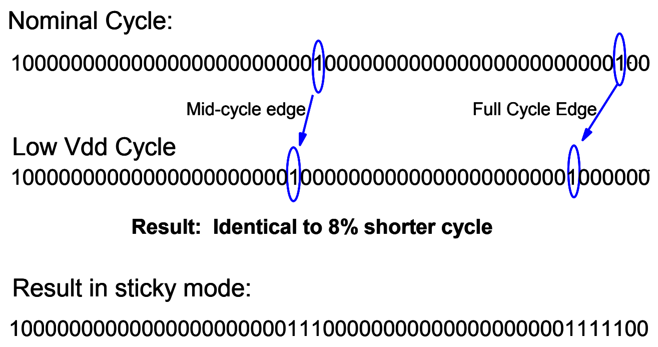


Fig 3: Edge movement due to low VDD (single-sample and sticky mode)

being written into it. This mode is used to lock in data from a particular cycle during an experiment and protecting it from being over-written prior to reading out the register. Readouts can occur in one of two ways. One way is by scanning, which means that the chip has to be put into scan mode. Another way is through the use of a service processor called SCOM, which allows for a type functional shift-out of the sticky register. This functional shifting is often preferred since it allows for readouts on-the-fly while in functional mode, without transitioning the chip into scan mode. The sticky register has to be configured with some extra logic in front of it shown in Figure 4, where the q_out of a neighboring latch can be selected for input. In this way, the contents of the register can be shifted out just like scanning, but using the regular c1,c2 functional clocks. Microprocessors that have the SCOM interface can perform readouts in a way that is extremely flexible and non-disruptive to the rest of the

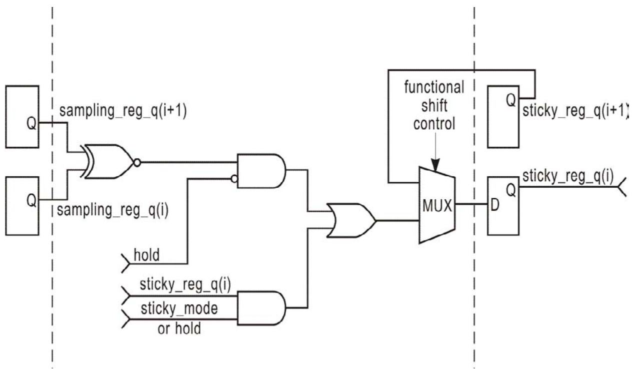


Fig 4: Added logic to enable SCOM functional shift-out

chip. Periodic readouts can be done under the program control of the service processor, or when an exception condition arises. The Skitter itself can be self-monitoring and trigger a readout if an edge is detected in a bin where it is not expected. The Skitter can notify the service processor of this condition and call for a readout.

2.2 Calibration Procedure

Just as with any measuring tool, the Skitter must be calibrated to achieve the best accuracy. The first calibration that is typically done is calibrating the inverter delay of the delay line. This is done simply by sending a clock signal of known period down the delay line and noting the bin numbers of a pair of full-cycle edges. This is best done while running a functional test pattern that generates little VDD noise. The pattern called IDLE is typically used during calibration. IDLE limits the on-chip switching activity and provides a quiet background reference for calibration. Once the bin locations of a pair of full-cycle edges are located, the difference of the bin numbers is the number of inverter delays that separate the two edges, and this number is then divided by the known period to get the delay per bin. Once this number is determined, then measurements like overall timing uncertainty can be converted from bin counts into picoseconds. This calibration needs to be done at every VDD value where measurements are taken, since the inverter delay changes with VDD.

Another type of calibration that needs to be done in order to recover the on-chip VDD waveform is measuring the sensitivity of the inverter delay to VDD. Using Skitter to recover the on-chip VDD waveform means that we have to be convinced that the timing variation that Skitter measures for this chip is due to VDD noise felt by the inverter chain and not clock jitter, since Skitter is also sensitive to jitter. Once we have proven to ourselves that VDD noise dominates the timing variation, then we can use the inverter chain's VDD sensitivity to recover the VDD waveform. Otherwise we must be satisfied with knowing only the total bottom-line timing variation from both sources.

The calibration of delay sensitivity to VDD is again done while running a quiet workload pattern like IDLE. In single-sample mode, the bin location of one of the edges is recorded, while VDD is incremented in 100 mV steps. The resulting plot of VDD vs. bin number can then be plotted as in Figure 5.

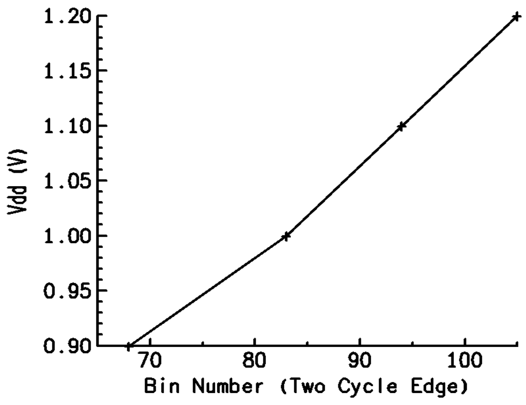


Fig 5: VDD sensitivity of inverter chain delay

When a pattern that generates VDD noise is run, the above plot can be used to convert the shift in an edge's bin number into mV of VDD noise. The plot is generally close to linear over a certain range of VDD. An alternate calibration approach is to find the slope of the above plot over the VDD range of interest. For similar hardware, the voltage sensitivity expressed as (%VDD/%Bin) is often quite reproducible, in which case it is not necessary to perform a new sensitivity calibration every time you change to another chip under test. The bin numbers will increase for faster chips (get through more inverters), but the sensitivity to VDD is similar.

Any of the edges can be used for measuring the on-chip VDD noise. Edges located further down the delay line are more sensitive to VDD variations since they pass through more inverters. Generally the one-cycle edge or the two-cycle edge gives the best results when measuring VDD noise. The one-cycle edge is the edge of most interest for measuring overall timing uncertainty. That edge is the count of the number of inverters that get completed in one cycle, which correlates to the Fmax, or the maximum operating frequency of the microprocessor when the Fmax limiting paths are one-cycle paths.

2.3 Multiple Skitters

Skitter circuits can be placed at different locations of the chip to study local variations in noise or jitter. Another application is to study skew. The most accurate method of determining skew, for example between two domains, is to place one Skitter in each clock domain, and cross-couple the two Skitters by sending the clock from within each Skitter to be measured by the other Skitter as shown in Figure 6.

If the wires between the two Skitters are delay-matched, and there is no skew, then the clock edges will be found in the same bin locations in both Skitters from symmetry. A skew of one Skitter inverter delay will cause a 2-bin difference in the edge locations. The sticky mode can then be used to find the total range of timing variation relevant for signals sent from one clock domain to the other during any desired test pattern or workload.

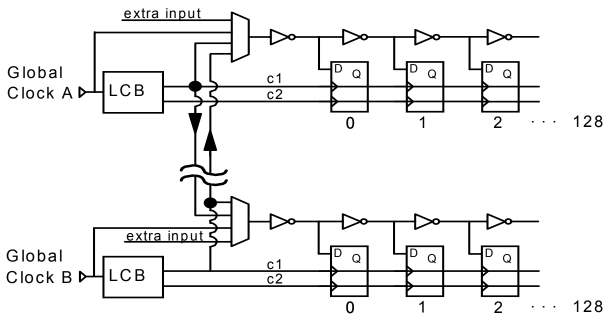


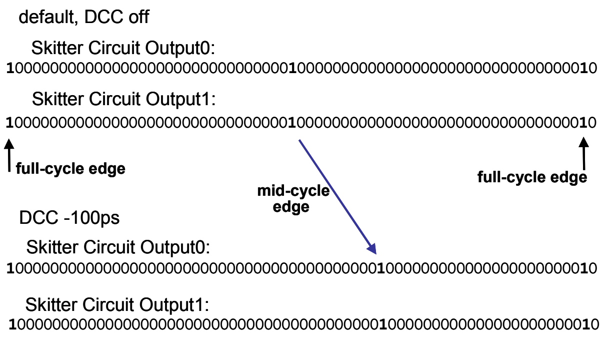
Fig 6: Cross-coupled Skitters to include clock skew effects

3. Experimental Results

3.1 CELL BE Processor

Accurate clock duty cycle measurement is important for many microprocessors. The mid-cycle edge, which is usually not used by many of the circuits on the microprocessor, can be important to the SRAM arrays which can make use of the mid-cycle edge during array operations. Also, certain dynamic circuit design styles make use of the mid-cycle edge to control circuit re-setting or pre-charge.

The CELL BE chip [2] was designed with a duty cycle correction (dcc) circuit in the PLL to allow for a wide range of duty cycle adjustment to characterize and optimize the performance of the arrays and dynamic logic. A method to characterize the dcc circuits and the duty cycle was needed that would sense the actual duty cycle at the LCBs, so that if there was any duty cycle distortion introduced by the PLL or clock distribution network, it could be included in the measurement. Two Skitters were placed on the CELL BE chip. One was located toward the corner of the chip and the other was located closer to the center of the chip. The two Skitters were several millimeters apart and therefore monitored the clock distribution at two very different points. Measured duty


```
default, DCC off
  Skitter Circuit Output0:
1000000000000000000000000000000100000000000000000000000000000010
  Skitter Circuit Output1:
1000000000000000000000000000000100000000000000000000000000000010

DCC -100ps
  Skitter Circuit Output0:
1000000000000000000000000000000000000000100000000000000000000010
  Skitter Circuit Output1:
1000000000000000000000000000000000000000100000000000000000000010
```

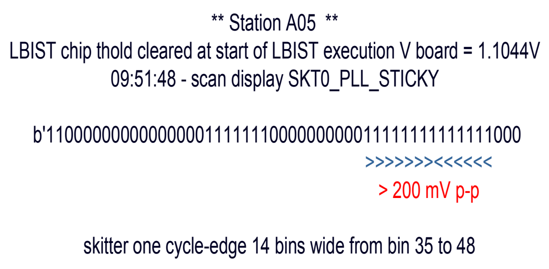
Fig 7: Measured clock duty cycle correction circuit using Skitter in single sample mode

cycle results on the two Skitters shown in Figure 7 showed nearly identical results. Both Skitters measured a 50% duty cycle when the dcc circuits were turned off (upper 2 plots) and with the dcc adjust circuit turned on, both Skitters measured the same movement (lower 2 plots) of the mid-cycle edge that the dcc circuits were designed to produce.

3.2 XBOX360 Processor

The XBOX360 processor is a 3-core PowerPC^{TM} based microprocessor used for gaming applications. Fmax testing using functional patterns represented a significant portion of the total test time. LBIST patterns were a way of speeding up Fmax testing and it was desirable to migrate the Fmax test to LBIST patterns. During the initial migration to LBIST-based patterns for Fmax, it was found that the Fmax that was measured with LBIST was significantly lower than the Fmax obtained using functional patterns. Power supply noise during LBIST was suspected. The chip was built with a total of five Skitter circuits. One Skitter was located in each core, another in the area of the Front Side Bus and one near the PLL. The Skitters were turned on during this testing and put into sticky mode in order to get an idea of the magnitude of the peak-to-peak timing uncertainty. It was discovered that some LBIST patterns produced much more timing uncertainty than functional workloads. The magnitude of the timing uncertainty was approx. 20% of the cycle. Figure 8 shows data from one Skitter in sticky mode.

The large peak-to-peak variation was attributed to VDD noise caused by the large current step that occurred when the LBIST pattern transitioned from scan mode to system mode. This event caused a peak-to-peak variation in the



** Station A05 **
LBIST chip thold cleared at start of LBIST execution V board = 1.1044V
09:51:48 - scan display SKT0_PLL_STICKY

b'11000000000000000011111110000000000111111111111111000
>>>>>>><<<<<<
> 200 mV p-p

skitter one cycle-edge 14 bins wide from bin 35 to 48

Fig 8: Measured LBIST VDD noise using Skitter in sticky mode

one-cycle edge of 14 bins. Using the calibration procedure described earlier, this amount of bin shift corresponded to a peak-to-peak VDD noise of over 200mV. As VDD collapses due to a current step event such as this one, the circuits slow down causing the chip to work at a lower frequency, consistent with the lower Fmax that was being measured during LBIST.

The large current increase that occurs when the chip transitions from scan mode to functional mode was reduced by increasing the scanning speed of the chip. The scanning speed is controllable using a set of global pins. Increasing the LBIST scanning speed increased the current that the chip draws when it scans, which in effect reduced the magnitude of the current step when the chip transitioned into the higher-current functional mode part of LBIST. This combined with some added partitioning of the LBIST pattern fixed the LBIST VDD noise issue.

The five Skitter circuits that were placed in different locations demonstrated that the noise profile was not uniform across the chip. Further, the largest voltage droop occurred coincident with the weakest area of the voltage rail decoupling, despite being furthest away from the circuitry producing the current step event.

3.3 PPC970MP Processor

The PPC970MP is a dual-core PowerPC microprocessor [3] that was built with four Skitter macros. One Skitter was placed in each core, one in the front side bus, and another in the nest. The Skitters were used for clock duty cycle measurement as well as to study power supply noise while running different workloads (instruction streams), and to measure skew between different clock meshes.

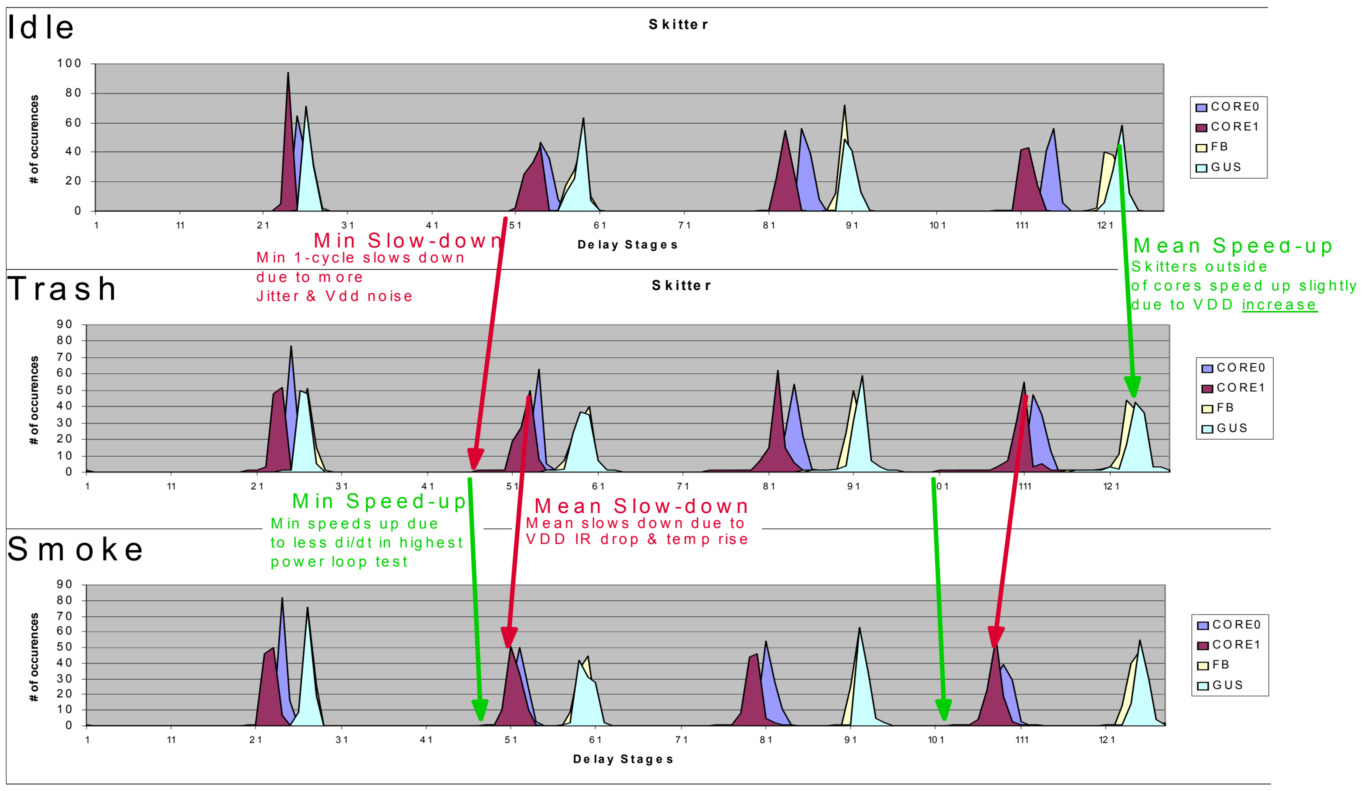


Fig 9: PPC970MP Skitters response to 3 workloads: IDLE, TRASH, SMOKE

The workloads that were selected for study are called IDLE, TRASH, and SMOKE. IDLE is a pattern that limits the amount of on-chip switching activity, and this pattern is often used as a reference since it provides a quiet, low power baseline that can be used to compare to other noisy and high-power workloads. TRASH is a pattern that generates random instructions and SMOKE is a workload that tries to maximize power consumption.

The Skitters were operated in single-sample mode and for each workload, 100 measurements were taken and the edge locations were plotted in a histogram as shown in Figure 9. In addition to the 100 single samples, the Skitters were also put into sticky mode for each workload and the variation that was measured in sticky mode is plotted on the same graph as a horizontal line on the x-axis. What was found is that each workload had its own unique "signature" on the Skitter data. In particular, TRASH produced the most variation as indicated by the sticky data (the horizontal line at the base of the histogram), even though the 100 sample histogram had only minor variation. This seems to imply that most of the time, TRASH produces low VDD noise, but if we monitor continuously using sticky mode, on rare occasions TRASH will cause some noise event that will cause a large variation in the edge locations. The sticky mode variation for TRASH was even larger than for SMOKE, the high power workload. The mean slow-down for SMOKE was as expected, since SMOKE will cause the chip to draw more current and result in a larger IR drop. The use of histograms to plot Skitter single-sample data combined with sticky mode data on the same histogram proved to be a useful way to examine the effect of different instruction streams on the timing uncertainty.

3.4 POWER6 Processor

The POWER6 is a dual core PowerPC microprocessor [4] that was built with two power grid designs for comparison. In one design, the core power grids were split from the nest (non-core) region of the chip and from each other. In the other power grid design, all power grids were connected together. Skitter was used to help compare the cycle-by-cycle timing variations from these two options [5]. To evaluate this, Skitter was used in what is called oscilloscope mode.

In oscilloscope mode, the instruction stream that causes a power supply noise event is run repetitively. The Skitter is read out, cycle by cycle, and the bins containing edges are recorded. The bin number is then plotted versus cycle number (or equivalently the bin number is converted to inverter delay using a calibration). This plot effectively reveals the on-chip VDD waveform as a function of the cycle number in the instruction stream. The VDD dips can then be correlated to cycles with large switching activity. Figure 10 shows an example of this on POWER6 with split power grids. One core (core 0) is running IDLE, the quiet pattern while the other core (core 1) is running TRASH, the random instruction pattern. The Skitter in the noisy core running TRASH measures a large edge

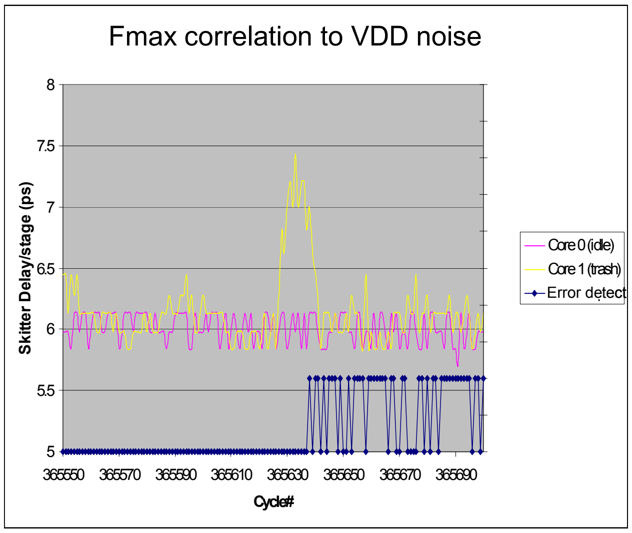


Fig 10:Skitter in o'scope mode detects cycles causing Fmax fail

variation during cycles where the switching activity suddenly jumps -- near cycle # 365630 in the instruction stream. This coincides with the detection of the fail at Fmax. The Skitters in the core running IDLE (core 0) show little edge variation, consistent with low switching activity.

By contrast, the POWER6 design with the connected power grids shows much less timing variation when running the same instruction stream. This is mainly due to the fact in the connected design, the core power grid share the "quiet" decoupling capacitance of the nest, which tends to stabilize the power grid. Figure 11 shows Skitter measurements that compare split to connected power grids. For increased resolution, multiple full-cycle edge locations were added. The noise is significantly reduced in

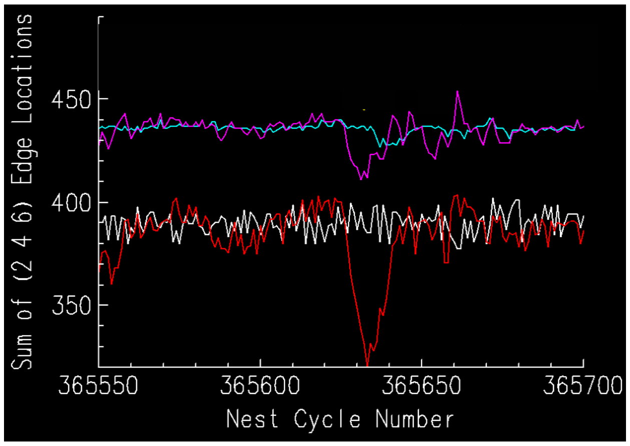


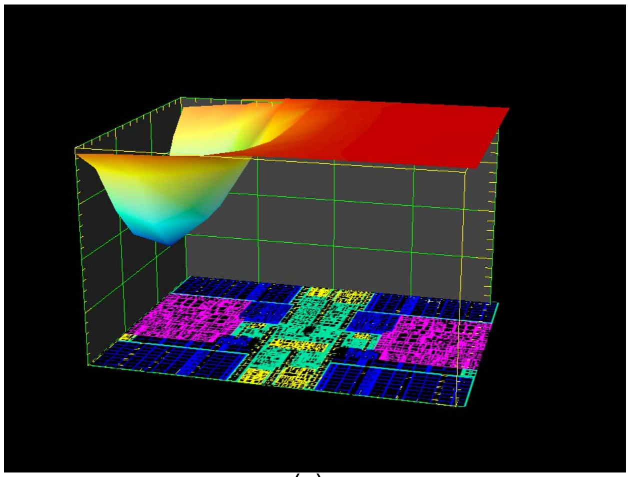
Fig 11: Measured comparison of noise on connected core power grids (top traces) and split power grids (bottom traces)

the connected power grid design.

4. Discussion

Since connected power grids seem to be better from a VDD noise standpoint, this raises the possibility of interaction of noise events from one core to another. For example, what happens if one core has a VDD dip and sends a noise "wave" over to another core. If the wave from the first core should arrive at the same time that second core generates its own VDD dip, then the two events might reinforce at the second core, resulting in a much larger dip at the receiving core.

The propagation speed of such a wave was measured on POWER6 by looking at the response of two Skitters, one in each core. The wave travels 9 mm in 4 ns. The question of how best to test for core to core VDD noise interaction will involve this wave travel time. If the receiving core is 9 mm away, then testing for the worst case noise scenario will involve delaying the starting time of the instruction stream running on the receiving core by the travel delay (4 ns in this case). In this way, the receiving core will be executing the offending cycles just when the VDD wave arrives from the sending core. Figures 12a and 12b are package simulations that show this situation. In Figure 12a, the wave is launched and in Figure 12b it arrives at the receiving core just as the receiving core generates its own VDD noise event. Clearly the worst-case noise occurs when the two VDD dips reinforce each other at the receiving core.


(a)

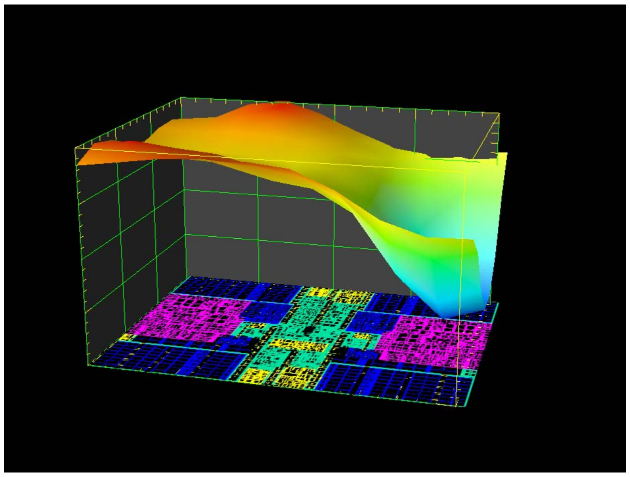

(b)

Fig 12: Simulation of VDD noise wave starting at core 1 (a) and the VDD noise wave reinforced at core 2. (b)

5. Conclusions

The on-chip Skitter circuit combined with a variety of test modes and analysis methods has been found to be valuable in the characterization of several issues on different microprocessors. The sensitivity to process variations, power supply noise, jitter, duty cycle, and skew have made Skitter useful in maximizing performance and reliability. In particular, the sensitivity of Skitter to power supply noise has made it an effective tool for recovering the on-chip VDD waveform during any workload. The measurement capability of Skitter has established it as a standard tool for testing a range of parameters on IBM chip products.

Acknowledgements

The authors would like to acknowledge Tom Strach, Brian Monwai, Steve Wilson, Tim McNamara, Tim Skergan, Steve Jones, John Reilly, Otto Wagner, Otto Torreiter, Andy Maki, Pong-Fei Lu, Tom Bucelot, and Jeff Burns for their contributions to this work. Also to be recognized are a large number of people in various test groups throughout IBM who have generated data and contributed to the growing knowledge base of Skitter measurements.

References

[1] P. Restle et al. "Timing uncertainty measurements on the Power5 microprocessor", ISSCC Digest of Technical Papers, Feb. 2004 pp 354-355

[2] D. Pham et al, "The Design and Implementation of a First-Generation CELL Processor", *ISSCC Digest of Technical Papers*, Feb. 2005, pp. 184-185.

[3] E. Cohen et al. "A 64B CPU Pair: Dual- and Single-Processor Chips", *ISSCC Digest of Technical Papers*, Feb 2006 pp 106-107

[4] J. Friedrich et al., "Design of the POWER6^{TM}", *ISSCC Digest of Technical Papers*, Feb. 2007, pp. 96-97

[5] N. James et al., "Comparison of Split versus Connected Core Supplies in the POWER6 Microprocessor", *ISSCC Digest of Technical Papers*, Feb. 2007, pp. 298-299