Impact of Die-to-Die and Within-Die Parameter Variations on the Throughput Distribution of Multi-Core Processors

Keith A. Bowman, Alaa R. Alameldeen, Srikanth T. Srinivasan, and Chris B. Wilkerson

Microprocessor Technology Lab, Intel Corporation, Hillsboro, OR

keith.a.bowman@intel.com

Abstract

A statistical performance simulator is developed to explore the impact of die-to-die (D2D) and within-die (WID) parameter variations on the distributions of maximum clock frequency (FMAX) and throughput for multi-core processors in a future 22nm technology. The simulator integrates a compact analytical throughput model, which captures the key dependencies of multi-core processors, into a statistical simulation framework that models the effects of D2D and WID parameter variations on critical path delays across a die. *The salient contributions from this paper are: (1) Product-level variation analysis for multi-core processors must focus on throughput, rather than just FMAX, and (2) Multi-core processors are inherently more variation tolerant than single-core processors due to the larger impact of memory latency and bandwidth on overall throughput.* To elucidate these two points, multi-core and single-core processors have a similar chip-level FMAX distribution (mean degradation of 9% and standard deviation of 5%) for multi-threaded applications. In contrast to single-core processors, memory latency and bandwidth constraints significantly limit the throughput dependency on FMAX in multi-core processors, thus reducing the throughput mean degradation and standard deviation by 50%. Since single-threaded applications running on a multi-core processor can execute on the fastest core, mean FMAX and throughput gains of 4% are achieved from the nominal design target.

Categories & Subject Description:

B.8.0 [Performance and Reliability]: General

General Terms: Design, performance, reliability

Keywords: Multi-core, FMAX distribution, throughput distribution, parameter fluctuations, parameter variations

1. Introduction

Power consumption is the number one challenge in today's high-performance microprocessor designs [1]-[3]. Arguably, the second most significant design challenge is managing the variability in device and circuit parameters [3]-[6]. As process technology scales to 65nm and beyond, the impact of variations on the maximum clock frequency (FMAX) and power of a microprocessor is expected to worsen. Parameter variations can be classified into two categories: Die-to-die (D2D) and Within-die (WID). D2D variations, resulting from lot-to-lot, wafer-to-wafer, and a portion of the within-wafer variations, affect all transistors and interconnects on a die equally. Conversely, WID variations consisting of random and systematic components induce different electrical characteristics across a die [7]. A random-WID parameter variation fluctuates randomly and independently from device to device (i.e., device-to-device correlation is zero). A systematic-WID parameter variation results from a repeatable and governing principle, where the device-to-device correlation is empirically determined as a function of the distance between the devices. Although systematic-WID variations exhibit a correlated behavior, the profile of these variations can randomly change from die to die. From a designer's perspective, systematic-WID variations behave as continuous and smooth correlated random-WID variations [7]-[9].

Multi-core processors are emerging as a power-efficient approach to designing high-performance microprocessors. In contrast to large single-core processors, multi-core processors employ a number of less complex cores on a die, where the number of cores and core complexity is a key design trade-off. Multi-core processors can achieve better performance on highly parallel multi-threaded applications by executing the threads across the cores while operating at a lower clock frequency and lower power.

In designing high-performance microprocessors, the importance of accurately estimating the impact of parameter variations on product-level performance directly relates to the overall revenue of a company. An overestimation increases design complexity, possibly leading to an increase in design time, an increase in die size, rejection of otherwise good design options, and even missed market windows [7]. Conversely, an underestimation can compromise product performance and overall yield as well as increase the silicon debug time [7]. In summary, overestimating variations impacts the design effort, and underestimating variations impacts the manufacturing effort.

Previous works evaluated the impact of D2D and WID parameter variations on the FMAX distribution of single-core processors [4], [10], [11]. In addition, the effect of parameter variations on the product-level power distribution has been studied for single-core processors [10], [11]. The impact of parameter variations on power, where leakage is the dominant variation component, does not fundamentally change from single-core to multi-core processors. A multi-core processor may enable much finer granularity in placing portions of the chip into a sleep state. When all transistors on the chip are in an operational mode, however, the effect of D2D and WID parameter variations on the leakage is expected to be similar between single-core and multi-core processors, assuming an equal number of transistors, V_{DD}, etc.

This work explores the impact of D2D and WID parameter variations on FMAX and throughput distributions of multi-core processors through a newly developed statistical performance simulator. A key component of the simulator is a compact analytical throughput model, enabling accurate projections of multi-core throughput for parallel applications while retaining runtime efficiency. The throughput metric represents the actual microprocessor performance, thus providing an architecture-level perspective of device and circuit parameter variability.

2. Statistical Performance Simulator

A statistical performance simulator is developed to explore the impact of D2D and WID parameter variations on the FMAX and throughput distributions for multi-core processors. In Fig. 1, a flowchart illustrates the simulation methodology. First, statistical SPICE-equivalent [12] circuit simulations are performed in a 65nm process technology [13] on speed-limiting paths in a high-performance microprocessor to generate individual critical path delay distributions for separate contributions of D2D and WID variations. The primary inputs to the statistical circuit simulator include: (i) D2D and WID transistor and interconnect parameter variation models, (ii) process file, and (iii) critical path netlists.

Since D2D variations affect transistors and interconnects equally, the D2D standard deviation to mean ratio for the critical path delay is similar from path to path. Systematic-WID variations affect transistors and interconnects equally for a path contained within a small region, and random-WID variations average across the number of logic stages in a path [4], [14]. These effects result in systematic-WID variations dominating the WID critical path delay variability [4], where the WID standard deviation to mean ratio for the critical path delay is similar from path to path. To demonstrate these effects, the path delay standard deviation to mean ratio is plotted in Fig. 2 across a variety of path types for separate contributions of D2D and WID variations. Path delay sensitivities to either D2D or WID variations appear relatively consistent. Although the normalized WID path delay variability can differ from path to path when comparing transistor-delay dominated paths with wiring RC-delay dominated paths, the vast majority of critical paths in a microprocessor are transistor-delay dominated. This directly results from traditional microprocessor design methodologies that ensure FMAX scaling as the process matures [15]. If the FMAX is limited by RC delay, then the FMAX would not benefit from transistor process improvements. As an example of these design methodologies, the RC-delay portion of a path is intentionally penalized during timing analysis by increasing the interconnect resistance by 30% or more [15]. Furthermore, additional pipelining and repeater insertion is applied to long distance paths to reduce the effect of wiring RC delay.

Using the critical path delay statistics, D2D and WID delay distributions of a typical critical path are constructed by averaging the standard deviation to mean ratios across the simulated critical paths for D2D and WID variations separately. Since systematic-WID variations contain a spatial correlation, critical path delays have a distance dependent correlation. Assuming each critical path traverses a fixed distance in a 65nm technology, simulations are performed with paths separated in x and y directions to extract the spatial critical path delay correlation. The typical critical path delay statistics are inputs to the statistical performance simulator.

In determining the number of critical paths per core, the path delay correlation provides guidance. In today's single-core microprocessors, 100's to 1000's of critical paths are uniformly spread across the die. For critical paths located in close proximity, the paths are highly correlated such that the distribution of one critical path is a close approximation of the maximum critical path delay distribution for the collection of critical paths. Based on the extracted path delay correlation statistics, a highly correlated region can be approximated to an area of 0.02mm^2 (i.e., 141µm x 141µm). Assuming the critical paths are uniformly spread across the core, the number of critical paths per core is determined by dividing the core area by the highly correlated region of 0.02mm^2. The path delay correlation for critical paths in separate highly correlated regions is modeled. The path delay correlations are a salient feature of this simulation approach, enabling the core FMAX correlations to be captured spatially across the die.

The statistical performance simulator performs 1000's of Monte Carlo runs per multi-core design. For each run, a delay sample is generated for each critical path. The slowest critical path for each core determines the FMAX for the core. All cores in the multi-core processor are assumed to have the same global clock frequency. For highly parallel multi-threaded (MT) applications, where all cores execute instructions, the slowest core FMAX determines the FMAX of the microprocessor. For single-threaded (ST) applications, the fastest core determines the FMAX. The corresponding MT and ST FMAX values are inputs to the multi-core throughput model, which is described in Section 3, to calculate the corresponding MT and ST throughput values. After performing 1000's of Monte Carlo runs, the simulator generates FMAX and throughput distributions for MT and ST applications.

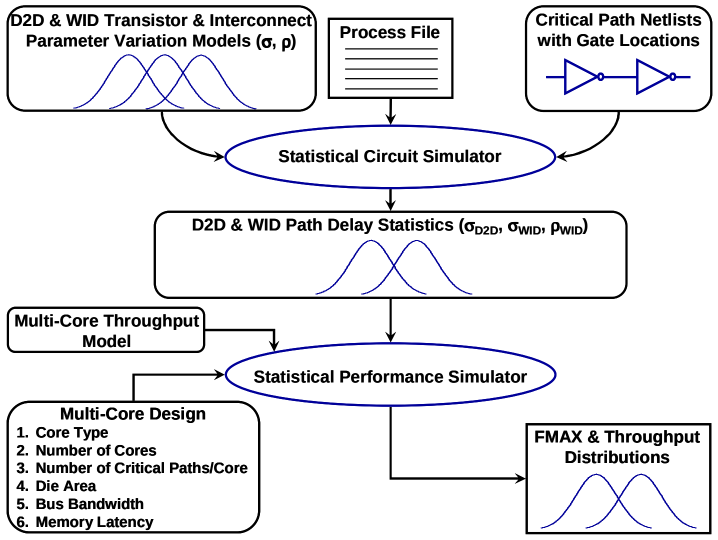

D2D & WID Transistor & Interconnect Parameter Variation Models (σ, ρ)
Process File
Critical Path Netlists with Gate Locations
Statistical Circuit Simulator
D2D & WID Path Delay Statistics (σ_{D2D}, σ_{WID}, ρ_{WID})
Multi-Core Throughput Model
Statistical Performance Simulator
Multi-Core Design
1. Core Type
2. Number of Cores
3. Number of Critical Paths/Core
4. Die Area
5. Bus Bandwidth
6. Memory Latency
FMAX & Throughput Distributions

Fig. 1: Flowchart describing the statistical simulation methodology for generating FMAX and throughput distributions for multi-core processors.

Path Type	Description
INVGATE	20-stage, fan-out of 3, inverter chain
INVFIX	10-stage, inverter chain with interconnect load
NAND2	16-stage, fan-out of 3, 2-input NAND gate path
NAND3	12-stage, fan-out of 3, 3-input NAND gate path
NOR2	12-stage, fan-out of 3, 2-input NOR gate path
NOR3	8-stage, fan-out of 3, 3-input NOR gate path
MUX	22-stage, inverter & mux combination path

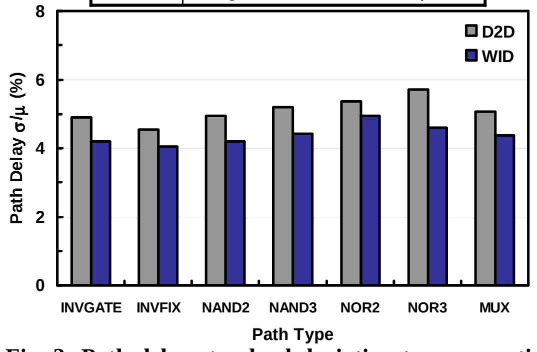


Fig. 2: Path delay standard deviation to mean ratio for D2D and WID variations versus path type.

3. Multi-Core Throughput Model

A key component of the statistical performance simulator is a tightly integrated multi-core throughput model. Since 1000's of Monte Carlo runs are performed per multi-core design, a computationally efficient analytical model is selected to enable accurate projections of average performance for a suite of Recognition, Mining, and Synthesis (RMS) applications [16]. The multi-core processor die area (A_{die}) is separated into three parts:

$$A_{die} = A_{core} + A_{L2(N)} + A_{I/O} \quad (1)$$

A_{core} is the total core area, where each core is assumed to contain private L1 instruction and data caches. $A_{L2(N)}$ is the total L2 cache area with N cores sharing the cache. $A_{I/O}$ is the area reserved for external memory controllers and I/O pads. For a given $A_{L2(N)}$ and area of a 1 MB cache (A_{1MB}), which is determined by process technology, the L2 cache size in units of MB is calculated as:

$$S_{L2(N)} = A_{L2(N)} / A_{1MB} \quad (2)$$

For a given workload, the cycles per instruction (CPI) of a single in-order blocking core is modeled as:

$$CPI(1) = CPI_{com} + M_{rate}(S_{L2(1)})L_{miss}(F_{clk}) \quad (3)$$

CPI_{com}, the computation component of CPI, is the core CPI with an infinite L2 cache. CPI_{com} is independent of the processor clock frequency (F_{clk}). $M_{rate}(S_{L2(1)})$, the miss rate, is the number of misses per instruction for a cache size of $S_{L2(1)}$. $L_{miss}(F_{clk})$, the miss penalty, is the average number of cycles per L2 cache miss. $L_{miss}(F_{clk})$ is a function of F_{clk}. The product of $M_{rate}(S_{L2(1)})$ and $L_{miss}(F_{clk})$ represents the memory latency and memory bandwidth components of CPI. $S_{L2(1)}$ is the effective L2 cache size for one core. If the cores do not share code or data in the cache, then the average cache size per core is 1/Nth of the entire L2 cache size ($S_{L2(1)}$=$S_{L2(N)}$/N). For applications that share code or data, the working set size is adjusted by the average number (N_{share}) of cores that share an L2 cache line, where $N_{share}(N)$ is a function of N. The average cache size for a single core is calculated as [17]:

$$S_{L2(1)} = S_{L2(N)} / \left(N - N_{share}(N) + 1\right) \quad (4)$$

To project the miss rate for caches of different sizes, the square root rule of thumb is applied, which models the cache miss rate as:

$$M_{rate}(S_{L2(1)}) = M_{rate}(1MB) \Big/ \sqrt{S_{L2(1)}} \quad (5)$$

For some applications, the square-root model in (5) is less accurate than the working set model, where the miss rate remains constant as cache size increases until the working set fits in the cache; subsequently the miss rate sharply falls off. Based on simulations across a variety of applications, the square root model provides the most accurate approximation of the average miss rate.

To model instructions per cycle (IPC) for the multi-core system, the effects of limited off-chip memory bandwidth is captured by separating $L_{miss}(F_{clk})$ into two components:

$$L_{miss}(F_{clk}) = \frac{L_{mem}(F_{clk})}{N_{pr}} + L_{link}(F_{clk}) \quad (6)$$

$L_{mem}(F_{clk})$, the off-chip DRAM memory latency, is calculated as the average number of cycles spent in the DRAM array to obtain data, where there is no distinction between hits to an open page versus a closed page or hits to the same row versus a different row in the DRAM array. In modeling out-of-order non-blocking cores that exploit memory-level parallelism (MLP), $L_{mem}(F_{clk})$ is divided by the average number (N_{pr}) of parallel memory requests since each request on average blocks the processor for a fraction of the total memory latency [18]. For in-order blocking cores, N_{pr}=1. $L_{link}(F_{clk})$, the total link latency, includes the latency of the physical off-chip link and the queuing latency (e.g., Miss status handling registers (MSHRs), bus queues). $L_{link}(F_{clk})$ is calculated as the average number of cycles for an off-chip memory access. $L_{link}(F_{clk})$ is separated into two components as:

$$L_{link}(F_{clk}) = L_s(F_{clk}) + L_q(F_{clk}) \quad (7)$$

The parameters $L_s(F_{clk})$ and $L_q(F_{clk})$ are the service and queuing latencies per cache miss, respectively. $L_s(F_{clk})$ is the physical off-chip link latency for data to traverse across the link from the processor to the DRAM chip and back, where no transmission errors are assumed. $L_q(F_{clk})$ is computed as the mean queuing latency. Defining λ as memory requests per cycle, λ is computed as λ=IPC(N)×$M_{rate}(S_{L2(1)})$. Using Little's law, the link utilization (U) is defined as U=λ$L_s(F_{clk})$. Assuming the physical off-chip link to memory represents an M/D/1 queue (Markovian arrival rate of requests with a deterministic service time and an infinite number of request sources), the response latency is calculated as:

$$L_{link}(F_{clk}) = L_s(F_{clk}) + \frac{UL_s(F_{clk})}{2(1-U)} = L_s(F_{clk}) + \frac{\lambda\left(L_s(F_{clk})\right)^2}{2\left(1-\lambda L_s(F_{clk})\right)} \quad (8)$$

The IPC for the entire multi-core system is calculated from (3), (6), and (8) as N/CPI(1) [17]:

$$IPC(N) = \frac{N}{CPI_{com} + M_{rate}(S_{L2(1)})\left(\frac{L_{mem}(F_{clk})}{N_{pr}} + L_s(F_{clk}) + \frac{\lambda\left(L_s(F_{clk})\right)^2}{2\left(1-\lambda L_s(F_{clk})\right)}\right)} \quad (9)$$

Since λ is a function of IPC(N), IPC(N) in (9) reduces to a quadratic equation, where the roots of the equation result in an explicit IPC(N) expression. The $L_{mem}(F_{clk})$ and $L_s(F_{clk})$ dependencies on F_{clk} are simply modeled as $L_{mem}(F_{clk})$= $L_{mem}(F_{clk,nom})$×F_{clk}/$F_{clk,nom}$ and $L_s(F_{clk})$=$L_s(F_{clk,nom})$×F_{clk}/$F_{clk,nom}$, where $F_{clk,nom}$ is the nominal processor clock frequency. Assuming all N cores have the same clock frequency, the overall throughput (TP) in instructions per second for a multi-core processor is calculated as IPC(N)×F_{clk}:

$$TP(N) = \frac{N}{\frac{CPI_{com}}{F_{clk}} + \frac{CPI_{mem,lat}(F_{clk})}{F_{clk}} + \frac{CPI_{mem,bw}(F_{clk})}{F_{clk}}} \quad (10)$$

$CPI_{mem,lat}(F_{clk})$/F_{clk} and $CPI_{mem,bw}(F_{clk})$/F_{clk} are the memory latency and bandwidth components of throughput, which are modeled as:

$$\frac{CPI_{mem,lat}(F_{clk})}{F_{clk}} = M_{rate}(S_{L2(1)})\frac{L_{mem}(F_{clk,nom})}{F_{clk,nom}N_{pr}} \quad (11)$$

$$\frac{CPI_{mem,bw}(F_{clk})}{F_{clk}} = M_{rate}(S_{L2(1)})\frac{L_s(F_{clk,nom})}{F_{clk,nom}}\frac{1-\frac{1}{2}\left(\frac{F_{clk}\lambda L_s(F_{clk,nom})}{F_{clk,nom}}\right)}{1-\left(\frac{F_{clk}\lambda L_s(F_{clk,nom})}{F_{clk,nom}}\right)} \quad (12)$$

Additional assumptions are applied to trade-off accuracy for runtime efficiency: (i) benchmark applications are perfectly parallelizable (i.e., the serial portion of a parallel program is not accurately modeled); (ii) average benchmark performance is an appropriate metric for evaluating general trends; and (iii) inter-thread interactions and operating system cooperation are ignored.

In validating the analytical model in (9) for ST applications, a comparison of total system IPC is performed with an industrial cycle-accurate simulator for different core types and cache sizes. Although the model primarily targets the performance of highly parallel MT applications, the analytical model is easily modified for ST applications, where one core has access to the entire L2 cache, by adjusting the miss rate from $M_{rate}(S_{L2(1)})$ to $M_{rate}(S_{L2(N)})$. In the comparison, IPC is evaluated across a set of 460 workloads, consisting of server, multi-media, games, SPEC2K, and office productivity applications, where the average error is 4%.

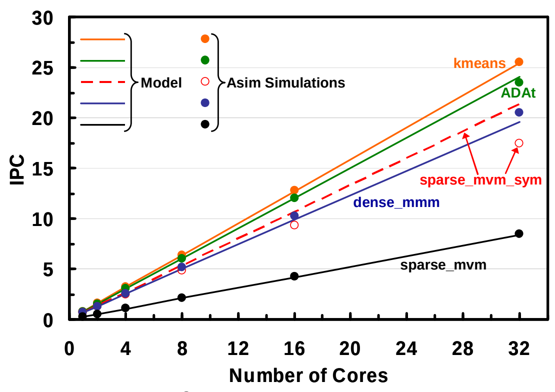


Fig. 3: Comparison of IPC model projections in (9) with Asim [19] for a variety of RMS benchmarks versus number of cores.

In validating the model for MT applications, the analytical model of total system IPC in (9) is compared with Asim simulations [19] in Fig. 3 for a variety of RMS benchmarks across the number of cores contained in the multi-core processor. These RMS benchmarks focus on the basic building blocks of matrix-oriented data manipulation and calculations that are increasingly being utilized to model and process complex systems [16]. The benchmarks include: (i) ADAt, (ii) dense_mmm, (iii) kmeans, (iv) sparse_mvm, and (v) sparse_mvm_sym. The Asim simulator [19] evaluates each workload while capturing the effects of multiple cores, shared L2 cache, and the interconnect network between the L2 cache and off-chip DRAM memory. The simulator models a 2-wide in-order core. The comparison in Fig. 3 includes a 32 MB L2 cache, 128 byte cache line size, and 200 cycle memory latency. The only workload specific inputs to the analytical model are: $M_{rate}(S_{L2(1)})$ and CPI_{com}. For three of the benchmarks (ADAt, dense_mmm, and kmeans), the square-root cache miss rate model in (5) is applied. For the other two benchmarks (sparse_mvm and sparse_mvm_sym), the working set model is used to estimate the cache miss rate. For the ADAt, dense_mmm, kmeans, and sparse_mvm benchmarks, the analytical model agrees closely with Asim simulations, where the worst-case error is less than 5%. The sparse_mvm_sym benchmark contains large sections of serial execution, leading to a worst-case model error of 22%. Although the model is less accurate for MT applications with large portions of serial execution, the multi-core throughput model agrees well with Asim simulations for both MT applications with large sections of parallel execution and ST applications.

4. Results

In exploring the impact of D2D and WID parameter variations on the FMAX and throughput of multi-core processors, three separate multi-core processors are evaluated. These three processors contain either small, medium, or large cores to investigate a range of multi-core design options. In addition, a traditional single-core processor, containing a monolithic core, is simulated as a baseline comparison. The small, medium, and large cores are based on the Intel® Pentium® P54C (in-order), the Intel® Pentium® III (out-of-order), and the Intel® CORE™2-Duo® (advanced out-of-order) microprocessors, respectively. In Fig. 4, the product introduction technology generation, core area, average F_{clk}, normalized average SpecInt throughput, cache size, V_{DD}, and core power for each core type are summarized based on historical data [20]-[23]. Note that the core area excludes the L2 cache area.

Using the historical data in Fig. 4 and traditional scaling trends, the area, nominal F_{clk}, V_{DD}, and power for the three cores in Fig. 4 are extrapolated from the core introductory technology node to a 22nm technology in Fig. 5. The monolithic core in Fig. 5 is based on historical trends of single-core microarchitecture performance improvements to scale the single-core performance of a 65nm microprocessor to 22nm. Key throughput model parameters (e.g., CPI_{com}, $M_{rate}(S_{L2(1)})$, N_{pr}) are generated from an industrial cycle-accurate simulator for each of the four core designs. System-level parameters represent an aggressive target for laptop and desktop microprocessor products at the 22nm technology node, including: (i) cache density of 0.5MB/mm^2, (ii) memory latency of 40ns and memory bandwidth of 50GB/s, (iii) die area allocated to cores and cache of 70mm^2 (excluding I/O), and (iv) power budget of 100W.

Since there is uncertainty in forecasting device and interconnect variability for the 22nm generation, especially since the device structure is still being defined, an optimistic assumption is made by applying the normalized D2D and WID critical path delay statistics from simulations performed on an industrial 65nm process technology [13]. In bounding the behavior of all four core types in Fig. 5, the two extreme core types (small and monolithic) are initially compared. Since the effect of systematic-WID parameter variations on processor FMAX and throughput depends on the die area allocated to the cores, the multi-core and single-core processors are designed with an equivalent A_{core} of 35mm^2, corresponding to 70 small cores for the multi-core processor and one monolithic core for the single-core processor. Both processors contain an L2 cache area of 35mm^2. In evaluating the multi-core processor, two modes of operation are considered: (i) MT applications, where all cores execute instructions while sharing the L2 cache, and (ii) ST applications, where only one core executes instructions while having access to the entire L2 cache.

In Fig. 6(a), the normalized FMAX distributions are plotted for the multi-core processor with small cores and the monolithic-core processor. The multi-core processor contains two separate FMAX distributions that correspond to MT and ST applications. The x-axis FMAX values are normalized to the nominal clock frequency for each separate processor (e.g., 3GHz for the multi-core processor and 4GHz for the monolithic-core processor). Statistical simulation results indicate that multi-core and single-core processors both have an FMAX mean degradation of 9% and an FMAX standard deviation of 5% for MT applications. Previous work revealed that WID variations primarily impact the FMAX mean and D2D variations primarily impact the FMAX variance [4]. From this perspective, the FMAX distributions for the two separate processors are equally affected by both D2D and WID parameter variations. As discussed in Sections 2 and 3, all cores in the multi-core processor are assumed to have the same global clock

Core Type	Technology (nm)	Core Area (mm^2)	Average F_{clk} (MHz)	Normalized Average SpecInt TP	Cache Size (KB)	V_{DD} (V)	Core Power (W)
Small	500	148	120	1	1000	3.30	12
Med	180	60	730	10	256	1.70	22
Large	65	35	2250	70	4000	1.33	35

Fig. 4: Historical data for small, medium, and large cores.

Core Type	Core Area (mm^2)	Nominal F_{clk} (GHz)	V_{DD} (V)	Core Power (W)
Small	0.5	3.0	0.90	1.3
Med	1.5	4.0	0.90	3.5
Large	5.0	4.0	0.90	9.6
Monolithic	35.0	4.0	0.90	75.0

Fig. 5: Core projections for a 22nm technology generation.

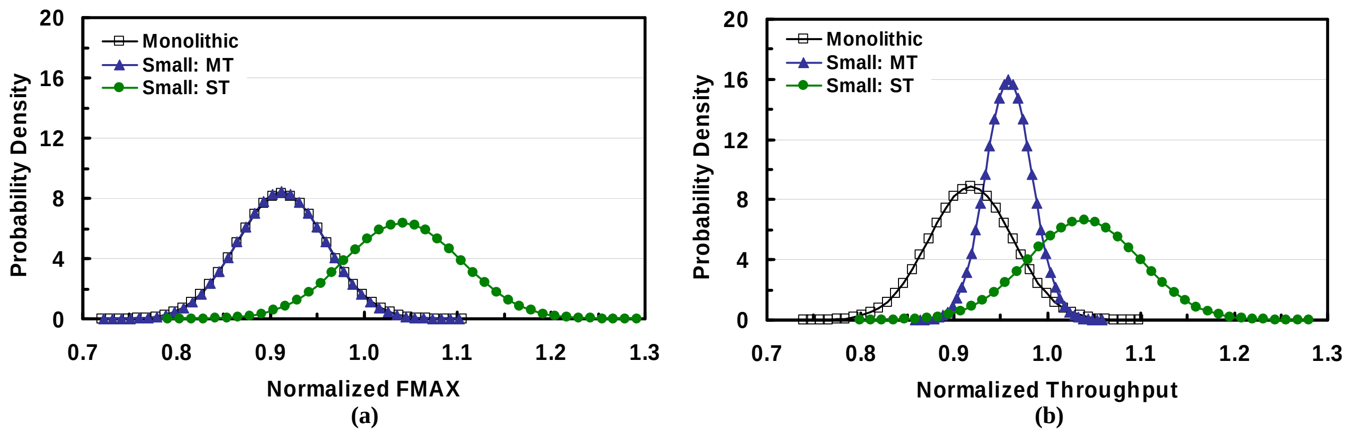


Fig. 6: (a) FMAX and (b) throughput distributions for the multi-core processor with small cores and the monolithic-core processor for multi-threaded (MT) and single-threaded (ST) applications.

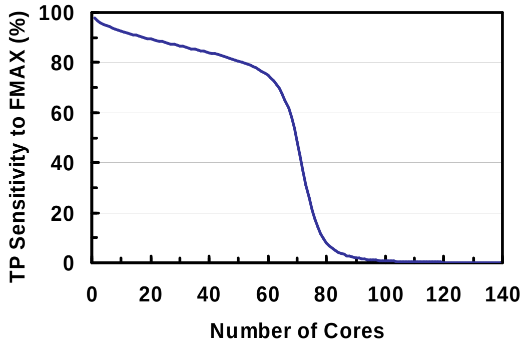


Fig. 7: Normalized throughput (TP) sensitivity to FMAX versus number of cores.

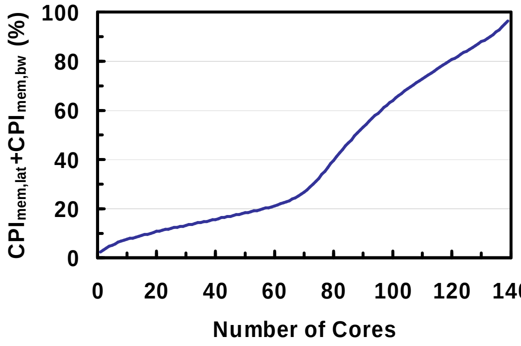


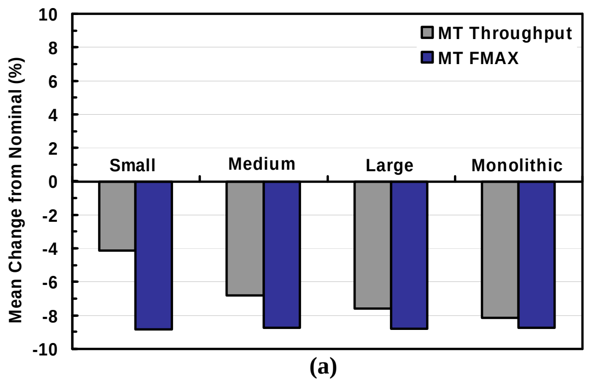
Fig. 8: Contribution of memory latency ($CPI_{mem,lat}$) and bandwidth ($CPI_{mem.bw}$) to total CPI versus number of cores.

frequency. Since both processors have an equal core area which effectively corresponds to a similar number of critical paths affected by systematic-WID variations, the maximum critical path delay distribution is similar between the two processors for MT applications, leading to an equal percentage of mean FMAX degradation. Since ST applications running on a multi-core processor can execute on the fastest core per die, simulation results indicate a 4% mean FMAX gain from the nominal design target.

In Fig. 6(b), the normalized throughput distributions are plotted for the corresponding FMAX distributions in Fig. 6(a). Similar to Fig. 6(a), the x-axis throughput values are normalized to the nominal throughput for each separate processor and separate application. Comparing the distributions in Fig. 6(a) and Fig. 6(b), the throughput distribution of the monolithic-core processor closely tracks the corresponding FMAX distribution. *The throughput distribution of the multi-core processor for MT applications, however, exhibits a 50% reduction in both the mean degradation and standard deviation as compared to the corresponding FMAX distribution*.

To provide physical insight for this drastic reduction in throughput variability, the normalized throughput sensitivity to FMAX for the multi-core processor is plotted versus the number of cores in Fig. 7. The throughput sensitivity to FMAX gradually decreases as the number of cores increases from 1 to 60, and then exponentially reduces as the number of cores increases beyond 60. As described in Section 3, the overall throughput contains three main components: (i) computation, (ii) memory latency, and (iii) memory bandwidth. Observing (10)-(12), the computation component depends linearly on F_{clk} while the memory latency and memory bandwidth components are largely insensitive to F_{clk}. In Fig. 8, the contribution of memory latency and memory bandwidth to the total CPI is plotted versus the number of cores in a multi-core processor. Initially, for only a few cores, the computation component of CPI dominates the total CPI, resulting in a throughput sensitivity to FMAX of unity. In this region, the throughput distribution closely tracks the FMAX distribution as illustrated in Fig. 6 for the monolithic-core processor. Thus, an analysis of FMAX is sufficient for analyzing product-level variability in single-core processors. As the number of cores increases, the L2 cache size allocated to each core reduces, thereby increasing the cache miss rate per core as described in (5). This directly increases the contributions of memory latency and bandwidth to CPI as modeled in (11)-(12) and illustrated in Fig. 8. *Therefore, multi-core processors are inherently more variation tolerant than single-core processors due to the larger impact of memory latency and bandwidth on overall throughput, which reduces the throughput sensitivity to FMAX. Furthermore, this result highlights the importance of evaluating the throughput metric, rather than just FMAX, for product-level variation analysis of multi-core processors*.

Since the computation component of multi-core throughput dominates for ST applications, the throughput distribution closely follows the FMAX distribution as illustrated in Fig. 6. In Fig. 9, the mean FMAX and throughput change from nominal is plotted versus the processors with the four core types summarized in Fig. 5 for MT and ST applications. For each simulation, the area dedicated to cores is 35mm^2 and the L2 cache area is 35mm^2, corresponding to one monolithic core, 7 large cores, 24 medium cores, and 70 small cores. From Fig. 9, the variation tolerance in throughput for MT applications improves as the smaller cores are



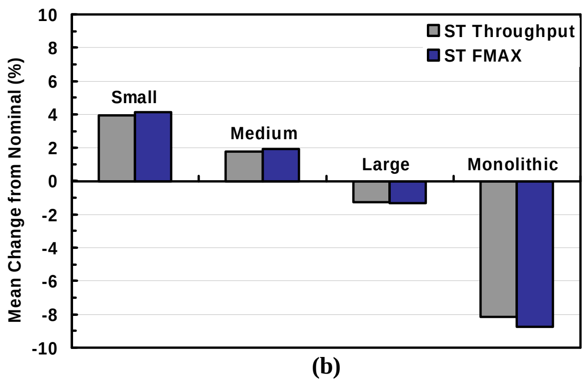


Fig. 9: Mean throughput and FMAX change from nominal versus processors with various core types for (a) multi-threaded (MT) and (b) single-threaded (ST) applications.

used, which directly corresponds to the reduced throughput sensitivity to FMAX as described in Figs. 7 and 8. Note that the reduction in the throughput standard deviation follows a similar trend. For ST applications, the relative mean FMAX and throughput also improves as smaller cores are used, resulting from the relationship between the systematic-WID critical path delay correlation and the number of spatially separated cores. In contrast to the MT applications, the throughput standard deviation for ST applications is primarily unaffected by the core type.

5. Conclusion

A statistical performance simulator is developed to examine the impact of die-to-die (D2D) and within-die (WID) parameter variations on the distributions of maximum clock frequency (FMAX) and throughput for multi-core processors in a future 22nm technology. The simulator integrates an analytical throughput model for multi-core processors into a statistical simulation framework that models the effects of D2D and WID parameter variations on critical path delays across a die. The analytical throughput model captures the key dependencies of a multi-core system as compared with rigorous cycle-accurate simulators for both single-threaded (ST) and multi-threaded (MT) applications. Statistical simulation results indicate that both multi-core and single-core processors with an equivalent total core area have an FMAX mean degradation of 9% and an FMAX standard deviation of 5% for MT applications. Although the throughput distribution of the single-core processor closely tracks the corresponding FMAX distribution, the throughput distribution of the multi-core processor exhibits a 50% reduction in both mean degradation and standard deviation as compared to the corresponding FMAX distribution. This variation tolerance for the multi-core processor results from the large influence of memory latency and memory bandwidth on overall throughput, which reduces the throughput sensitivity to FMAX. Therefore, multi-core processors are inherently more tolerant to parameter variations than single-core processors. Moreover, this result highlights the importance of examining the throughput metric, rather than just FMAX, for product-level variation analysis of multi-core processors. Since ST applications running on a multi-core processor can execute on the fastest core, mean FMAX and throughput gains of 4% are obtained from the nominal design target. These large improvements in the throughput distribution for both MT and ST applications reveal that multi-core processors could drastically reduce the product design and process development complexities due to parameter variations as compared to single-core processors, enabling faster time to market for high-performance microprocessor products.

6. References

[1] P. Gelsinger, "Microprocessors for the New Millennium: Challenges, Opportunities, and New Frontiers," in *IEEE ISSCC*, Feb. 2001, pp. 22-25.

[2] M. Horowitz and W. Dally, "How Scaling Will Change Processor Architecture," in *IEEE ISSCC*, Feb. 2004, pp. 132-133.

[3] M. Horowitz, et al., "Scaling, Power, and the Future of CMOS," in *IEEE IEDM*, Dec. 2005, pp. 11-17.

[4] K. Bowman, S. Duvall, and J. Meindl, "Impact of Die-to-Die and Within-Die Parameter Fluctuations on the Maximum Clock Frequency Distribution for Gigascale Integration," *IEEE JSSC*, pp. 183-190, Feb. 2002.

[5] *International Technology Roadmap for Semiconductors (ITRS)*, Semiconductor Industry Association, www.itrs.net, 2006.

[6] S. Borkar, et al., "Parameter Variations and Impact on Circuits and Microarchitecture," in *40th DAC Proc.*, June 2003, pp. 338-342.

[7] S. Duvall, "Statistical Circuit Modeling and Optimization," in *5th Intl. Workshop Statistical Metrology*, June 2000, pp. 56–63.

[8] H. Masuda, et al., "Challenge: Variability Characterization and Modeling for 65- to 90-nm Processes," in *IEEE CICC*, Sept. 2005, pp. 593-600.

[9] S. Samaan, "The Impact of Device Parameter Variations on the Frequency and Performance of Microprocessor Circuits," in *IEEE ISSCC Microprocessor Circuit Design Forum*, Feb. 2004.

[10] J. Tschanz, et al., "Adaptive Body Bias for Reducing Impacts of Die-to-Die and Within-Die Parameter Variations on Microprocessor Frequency and Leakage," *IEEE JSSC*, pp. 1396-1402, Nov. 2002.

[11] Y. Abulafia and A. Kornfeld, "Estimation of FMAX and ISB in Microprocessors," *IEEE Trans. VLSI Syst.*, pp. 1205–1209, Oct. 2005.

[12] *HSPICE User's Manual*, Meta-Software Inc., Mar. 1995.

[13] P. Bai, et al., "A 65nm Logic Technology Featuring 35nm Gate Lengths, Enhanced Channel Strain, 8 Cu Interconnect Layers, Low-k ILD and 0.57 μm^2 SRAM Cell," in *IEEE IEDM*, Dec. 2004, pp. 657-660.

[14] M. Eisele, et al., "The Impact of Intra-Die Device Parameter Variations on Path Delays and on the Design for Yield of Low Voltage Digital Circuits," *IEEE Trans. VLSI Syst.*, pp. 360–368, Dec. 1997.

[15] J. Schutz and C. Webb, "A Scalable X86 CPU Design for 90nm Process," in *IEEE ISSCC*, Feb. 2004, pp. 62-63.

[16] P. Dubey, "A Platform 2015 Model: Recognition, Mining and Synthesis Moves Computers to the Era of Tera," Intel Corp., Feb. 2005, download.intel.com/technology/computing/archinnov/platform2015/download/RMS.pdf.

[17] A. Alameldeen, *Using Compression to Improve Chip Multiprocessor Performance*, PhD Thesis, University of Wisconsin, 2006.

[18] T. Karkhanis and J. Smith, "A First-Order Superscalar Processor Model," in *Proc. of 31st Annual Intl. Symp. Comp. Arch.*, June 2004, pp. 338-349.

[19] J. Emer, et al., "ASIM: A Performance Model Framework," *IEEE Computer*, pp. 68-76, Feb. 2002.

[20] *Microprocessor Quick Reference Guide*, Intel Corp., www.intel.com/pressroom/kits/quickrefyr.htm.

[21] *SPEC Benchmarks*, Systems Performance Evaluation Corp., www.spec.org.

[22] J. Doweck, "Inside the CORE™ Microarchitecture," in *HotChips*, Aug. 2006.

[23] S. Rusu, et al., "A Dual-Core Multi-Threaded Xeon Processor with 16MB L3 Cache," in *IEEE ISSCC*, Feb. 2006, pp. 102-103.