Variability and Energy Awareness: A Microarchitecture-Level Perspective*

Diana Marculescu, Emil Talpes
Dept. of Electrical and Computer Engineering
Carnegie Mellon University
Pittsburgh, PA 15213
Email: {dianam, etalpes}@ece.cmu.edu

ABSTRACT

This paper proposes microarchitecture-level models for Within Die (WID) process and system parameter variability that can be included in the design of high-performance processors. Since decisions taken at microarchitecture level have the largest impact on both performance and power, on one hand, and global variability effect, on the other hand, models and associated metrics are needed for their joint characterization and analysis. To assess how these variations affect or are affected by microarchitecture decisions, we propose a joint performance, power and variability metric that is able to distinguish among various design choices. As a design-driver for the modeling methodology, we consider a clustered high-performance processor implementation, along with its Globally Asynchronous, Locally Synchronous (GALS) counterpart. Results show that, when comparing the baseline, synchronous and its GALS counterpart, microarchitecture-driven impact of process variability translates into 2-10% faster local clocks for the GALS case, while when taking into account the effect of on-chip temperature variability, local clocks can be 8-18% faster. If, in addition, voltage scaling (DVS) is employed, the GALS architecture with DVS is 26% better in terms of the joint quality metric employing energy, performance, and variability.

Categories and Subject Descriptors: I.6 [Simulation and Modeling]: Modeling methodologies; B.8.2 [Performance and reliability]: performance analysis and design aids.

General terms: design, performance.

Keywords: variability, power consumption, GALS design.

1 INTRODUCTION

Driven by aggressive technology scaling, sub-wavelength litography is causing increased variability in process technology parameters. In addition, due to increased power density and stressed thermal envelope, system parameter variability (e.g., temperature and voltage variation) increases as well. Such process and system parameter variations can manifest themselves across a single die (within-die or WID, e.g., in the case of temperature or voltage variations) or across several dies (die-to-die or D2D, e.g., in the case of clock speed and leakage power variations). In addition, variations can be systematic or random, static or dynamic in nature. Irrespective of their source or manifestation, variability poses a major challenge in designing high-performance processors or complex single chip systems. First, due to increased parameter variation, reliability of logic and memory available on chip decreases significantly. Second, aggressive techniques for increasing performance (e.g., deep pipelines and shallow logic depths) or decreasing power (e.g., multiple V_t or V_{dd} designs) have the side effect of increasing the number of critical paths or decreasing the logic depth, which in turn affects negatively variability and overall performance.

The problem of analyzing system and process parameter variability has gained only recently attention and has been addressed from the perspective of leakage and performance analysis of deep-submicron designs [4]. In addition, techniques that deal with reducing variability have been proposed at lower levels of abstraction (circuit level): e.g., the use of adaptive body biasing and adaptive voltage supply for increasing highest frequency bins, and thus for reducing clock speed variability [14]. However, many decisions taken at microarchitecture level affect design variability, while also significantly impacting overall performance and power profile of the design. This is especially true for high performance processors where complex logic is dedicated to increasing performance and extracting more parallelism (e.g., issue and out-of-order logic, branch or value prediction, trace caches), or where clustered architectures are used for running non-critical traces at the lowest speed necessary to meet performance targets. Such techniques increase the number of critical paths, thus impacting negatively variability, while deep pipelining and the push for high clock speeds decreases logic depth and has an undesirable impact on design variability. Models and associated metrics that can assess early various trade-offs and pinpoint to cost-effective or complexity aware solutions become thus indispensable for computer architects and designers.

To address this problem, this paper proposes *models and metrics* for *variability at microarchitecture level*, that can provide an early guide to various choices available in the design process. The models are *statistical* in nature for characterizing both process and system parameter variability and target *high-performance processors* with various microarchitectural organizations. We also propose metrics that combine die area, power consumption and expected mean clock speed value, with performance (or instructions per cycle - IPC). To assess the usefulness of the models and metrics, fully synchronous and Globally Asynchronous, Locally Synchronous (GALS) high-performance processors are compared and contrasted.

1.1 Related Work

The problem of process and system parameter variability and its impact on performance or leakage energy has started to gain attention in the past few years. Specifically, Eisele *et al.* [7] have shown how random and systematic within-die variations affect overall performance of low power designs. Later on, Bowman *et al.* [4] have developed statistical models for both WID and D2D variations and their correlation with microarchitecture-driven parameters such as number of independent critical paths and logic depth. Models are validated against real microprocessor chips fabricated in 0.25um and 0.13um process technology and found to be within 3% error. Borkar *et al.* [3] described how microarchitecture decisions affect variability and advocate for the introduction of a probabilistic optimization metric involving maximum allowable clock speed FMAX, energy and total die area. They also suggest the use of adaptive body biasing for reducing the effect of WID and D2D variations on microprocessor speed and leakage current, topic initially addressed by Tschanz *et al.* [14].

Energy aware design solutions based on multiple clock domain or GALS designs have been addressed by Semeraro *et al.* [11], Iyer *et al.* [8] and analyzed by Talpes *et al.* [13] in terms of trade-offs among inter-domain communication mechanisms, granularity of clock

* This research was supported in part by Semiconductor Research Corporation contracts no. 2004-HJ-1189 and 2005-HJ-1314.

DAC 2005, June 13–17, 2005, Anaheim, California, USA.
Copyright 2005 ACM 1-59593-058-2/05/0006…$5.00.

domains and voltage scaling algorithms. In [13], the arbiter-based FIFO mixed clock communication scheme is found to have the best overall performance/power operating point, while the use of a threshold-based voltage scaling algorithm with inter-domain dependency information is found to perform better than other proposed scaling techniques.

Thermal considerations in high-performance processors have been first addressed by Skadron *et al.* [12] where thermal management at microarchitecture-level has been addressed. It is found that up to 15% temperature difference can be found in high-end processors, with significant impact on on-chip performance and reliability.

1.2 Paper Contribution

As described in [3], microarchitecture decisions have a significant impact on WID variability due to gate length variations, in conjunction with number of independent critical paths and logic depth. All these factors impact overall performance and its variability: indeed, with increased number of independent critical paths, or decreased logic depths, WID gate length variation induced variability of performance increases, by decreasing the mean of the allowable maximum clock speed (FMAX). This paper targets exactly this problem, by making the following two main contributions:

- *First*, we propose microarchitecture-level statistical models that can assess how various decisions taken at this level affect both design variability and overall power/performance metrics. We also propose a probabilistic quality metric Q that includes the maximum clock frequency distribution, as well as energy, performance and die area, for characterizing various designs.
- *Second*, based on the developed statistical models, we contrast and compare two design drivers: (1) a fully synchronous baseline high performance processor; and (2) its multiple clock domain or GALS counterpart, with and without voltage scaling. Results show that including WID variations due to gate length and temperature variability allows detailed analysis of local clock speeds, which can be 8-18% faster than the fully synchronous version. In addition, adding DVS capabilities reduces the overall quality metric Q by 26% when compared to the synchronous baseline, thus showing that multiple clock designs with voltage scaling are better not only in terms of *power* and *performance*, but also in terms of *variability.*

The rest of the paper is structured as follows: Section 2 introduces the basis of the variation models used, while Section 3 describes our approach for modeling variability at microarchitecture level. The experimental setup and results are included in Section 4 and Section 5, with the paper being concluded with final remarks and directions for future work in Section 6.

2 PRELIMINARIES

In this paper, the effect of decisions taken at microarchitecture-level on variability, and in turn, its impact on overall product performance is analyzed. To this end, we are mainly concerned with impact of microarchitecture decisions on maximum clock frequency (FMAX) and overall performance distribution. For the purpose of our microarchitecture modeling, we rely on a previously derived FMAX distribution model [4], as summarized next.

Based on statistical circuit simulation of representative speed-limiting path delays, [4] determines the individual contributions of D2D and WID variations on the path delay distribution. The model has been validated by direct evaluation against real FMAX distributions measured for two generations of microprocessor products (0.25um and 0.13um) and found to be within less than 3% error. Specifically, the critical path delay density functions resulting from D2D and WID parameter fluctuations are modeled as normal distributions with mean $T_{cp,\,nom}$ (nominal mean critical path delay, assumed equal to the longest path delay) and standard deviations $\sigma_{D2D - T_{cp,\,nom}}$ and $\sigma_{WID - T_{cp,\,nom}}$, respectively.

As described in [4], WID fluctuations determine the mean of the maximum critical path delay distribution, while D2D fluctuations determine the variance (due to this reason, in this paper, we are only concerned with WID variations due to gate length and temperature).

Indeed, the maximum critical path delay can be expressed as:

$$T_{cp,\,max} = T_{cp,\,nom} + \Delta T_{D2D} + \Delta T_{WID} \quad (1)$$

where the distributions for the variations due to D2D and WID variations (ΔT_{D2D}, ΔT_{WID}) are normally distributed with mean 0 and standard deviation $\sigma_{D2D - T_{cp,\,nom}}$. Thus, the effective global distribution for overall variability was shown to be:

$$f_{\Delta T_{WID}}(t) = N_{cp} f_{WID - T_{cp,\,nom}}(t - T_{cp,\,nom}) \cdot (F_{WID - T_{cp,\,nom}}(t - T_{cp,\,nom}))^{N_{cp} - 1} \quad (2)$$

where $f_{WID - T_{cp,\,nom}}$ is the normal distribution for WID variations and $F_{WID - T_{cp,\,nom}}(t) = \int_0^t f_{WID - T_{cp,\,nom}}(\tau) d\tau$ is the corresponding cumulative probability that a critical path has a delay less than t. As seen in (2), the WID distribution heavily depends on the total number of independent critical paths for the entire chip N_{cp}. With larger number of critical paths, the mean value of the maximum critical path delay increases [4]. Intuitively, as the number of critical paths goes up, the probability that at least one of them will be affected by process-induced variability is higher, thus moving the maximum critical path delay up. On the other hand, the standard deviation goes down with increased values of N_{cp}, thus making the spread of the overall critical path delay a more predominant function of $\sigma_{D2D - T_{cp,\,nom}}$, or the D2D induced variations.

Another factor that affects the FMAX distribution is the logic depth (or number of gates) per critical path n_{cp}. In conjunction with a random or systematic WID variation model, the impact of the logic depth n_{cp} on the critical path distribution is different. While in the case of systematic WID variations, all gates on a path are affected in the same way, in the case of completely random fluctuations, they are expected to have an averaged effect on the overall critical path delay distribution [7]. Bowman *et al.* find that real measurements put the number of FMAX standard deviations somewhere in between the random and systematic case.

By direct analysis of (1)-(2) and as shown before, we note that both N_{cp} and n_{cp} are a direct function of the microarchitecture-level decisions usually associated with either increasing performance, or decreasing power consumption. Indeed, decisions such as superpipelining imply a decrease in the logic depth per pipeline stage (n_{cp}), as well as a likely increase in the number of independent critical paths (N_{cp}) due to an increase in the overall number of logic paths. At the same time, any microarchitecture enhancements performed with the goal of increasing performance or decreasing power imply increased complexity or added transistor count, which translates to a proportionally larger number of critical paths. A special case is the one of GALS architectures that have been described as possible power saving solutions [11][8]. As it will be seen in the sequel, multiple, locally clocked domains, may reduce the impact of process variability and allow for higher local clock speeds and overall performance, while also enabling the use of power savings techniques.

To identify the interplay between performance enhancing and power saving features, on one hand, and maximum clock speed distribution, on the other hand, we propose a *microarchitecture-level* modeling methodology for process-induced variability. Such models can be used in conjunction with power and performance evaluation of various microarchitecture features to identify the potential impact on FMAX distribution, and thus, the effect on overall performance distribution.

As a design driver for this methodology we consider the case of high performance processors, relying on a superscalar, out-of-order architecture, which are characterized by aggressive clock speed scaling and are more likely to have the FMAX distribution severely affected by various microarchitecture decisions.

3 VARIABILITY MODELING AT MICROARCHITECTURE LEVEL

In this section, we detail our microarchitecture model of variability, including the impact of various microarchitecture features on N_{cp} or n_{cp}, and thus, on overall performance distribution.

To this end, we consider as a design driver for the analysis in this paper, the case of a typical superscalar, out-of-order processor, as shown in Figure 1. Two cases will be considered: (a) the case of a synchronous baseline architecture using a partitioned issue window, depending on the type of operation performed: integer, floating point, or memory operation (Figure 1(a)) - typical for architectures characterizing Alpha processors; and (b) the GALS counterpart, as it was also described in related work [11][8] (Figure 1(b)). Since the models in equations (1)-(2) have been shown to be extremely close to measured data for real processors manufactured in 0.25um and 0.13um process technology, we will use (a) above for validating our own microarchitecture models for variability. We will then use the models to assess how a power saving architecture organization, such as one using GALS design, with and without voltage scaling, affects the design described in Figure 1(b) in terms of its variability and maximum clock speed.

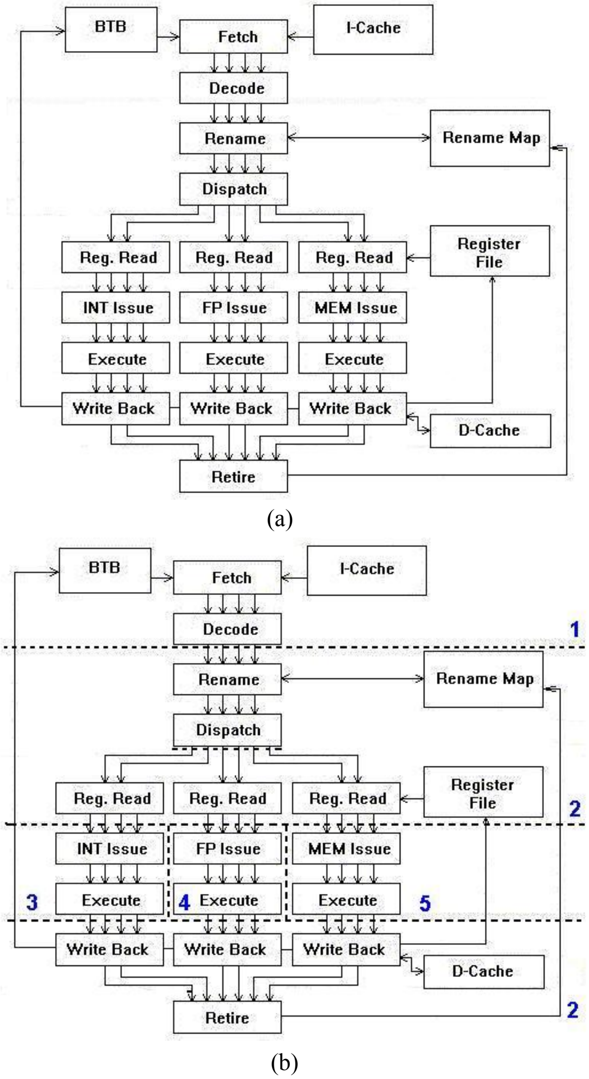
While both a four-clock domain and a five-clock domain GALS architecture have been proposed [11][8], we have chosen the five-clock domain implementation, as shown in Figure 1(b) as a design point that strikes the balance between target performance and partitioning criteria driven by physical constraints. Specifically, the register file is typically hotter than other on-chip modules, and thus, assigning it to a separate clock domain than the front-end (Fetch/Decode) will allow temperature-driven variability to be lower. In addition, when employing voltage scaling mechanisms for the back-end (i.e., the Integer, Floating Point, or Memory domains), previous work has shown how inter-clock-domain communication traffic can be used to decide when and if a certain clock domain can have its voltage/speed scaled down. An attack-decay [11] and threshold-based [8] voltage scaling mechanism have been proposed, while more recent work [13] has shown that adding inter-clock domain dependency information reduces the overall performance hit and improves energy efficiency. In our GALS architecture, if DVS is employed, we rely on the best performing algorithm, as described in [13] (threshold-based with dependency). Finally, multiple communication mechanisms are possible for inter-clock domain interfacing. We assume an arbiter-based, mixed-clock FIFO communication mechanism [11].

To this end, we assume that the number of critical paths is proportional to the total device count of the design. While this may not be a valid assumption when specific architecture enhancements are considered (e.g., various predictors, trace or execution caches) which are typically on the overall critical path of the design, it can be applied when analyzing the same architecture at different technology points or for comparing the synchronous design against its GALS counterpart.

We note that not all microarchitecture structures can be easily modeled by a generic critical path model comprised of two-input NAND gates with a fanout of three, as described in [4]. Examples include array structures that are part of the issue window or rename table structures. Such structures are affected mostly by *wire delay variations*, as they are structures whose latency is dominated by wires (bitlines or wordlines). However, as gate length is among the most difficult device parameters to control, assuming the same generic critical path model for all on-chip structures, without considering other deice and circuit parameters (such as supply voltage V_{dd}, oxide thickness t_{ox}, etc.) will provide a *lower bound* on the expected variability of a given design.

3.1 N_{cp}, n_{cp}, and impact on global design quality metrics

While previous work has shown how total number of independent critical paths N_{cp} and logic depth n_{cp} impact overall design variability, the validation has been done for a single design point. We show in this section how, based on complexity measures characterizing the design, one can determine not only which design decisions offer the best overall performance (defined as $IPC \cdot f_{clk} = \frac{IPC}{T_{cycle}}$), but also which configurations are likely to change the mean value and standard deviation of the maximum allowable clock speed FMAX. Thus, a joint metric that characterizes the distribution of the overall performance (or its reciprocal, time needed to commit a single instruction) is necessary. We define such a metric by first characterizing the maximum time needed to commit an instruction through $T_{instr,max} = T_{cp,max} \cdot CPI$ where $T_{cp,max}$ is the maximum critical path delay as in equation (1). For a given microarchitecture, both *CPI* and the distribution of the maximum critical path delay are affected, thus prompting us to consider this metric for characterizing the quality of a given design.



(a)

(b)

Figure 1. The design driver: A superscalar, out-of-order processor architecture. Two cases considered: (a) A synchronous architecture; (b) A multiple clock domain architecture [11][8].

At the same time, as suggested in [3], a possible probabilistic design optimization metric is one which involves the maximum clock speed distribution FMAX, total energy and die area: $\frac{FMAX^{\alpha}}{Energy^{\beta} \cdot DieArea^{\gamma}}$ where α, β, γ are weighting factors which can be used to tune the optimization objective toward various goals (i.e., higher performance, lower power, higher yield or lower die area). Since in the case of high-end processors clock speed is only one of the terms in the performance equation, we propose the following metric to characterize overall design quality:

$$Q = Energy \cdot DieArea \cdot T_{instr,\,max} \qquad (3)$$
$$= Energy \cdot DieArea \cdot T_{cp,\,max} \cdot CPI$$

where for the $DieArea$ we can use the total device count as a proxy. The quality metric Q gives equal weight to performance (via $T_{cp,\,max} \cdot CPI$), energy, and yield (via $DieArea$). A good design from the perspective of energy, performance and yield will most likely minimize this metric. We note that, since $T_{cp,\,max}$ is characterized by the distribution shown in equations (1)-(2), the quality metric will also be distributed similarly, up to some constant factor. An observation is in order here: energy is the sum of both switching and static energy, with static energy also being heavily characterized by deviations due to D2D V_t variability. However, for the purpose of this study, we consider only *gate length variations*, due to WID-induced variability and do not include the effect of D2D variations on leakage. Including the leakage energy variations in the definition of quality metric Q above would imply an additional convolution of the distribution for $T_{cp,\,max}$ and the one for energy ($Energy$).

We can also include the effect of nonuniform die temperature on the maximum clock frequency. In the case of the GALS architecture, one can assign nominal clock speeds locally, depending on the temperature profile. As it has been shown before [12], high-performance processors can have up to 15% temperature variation across different modules. Hotter modules dictate the global clock speed in case of a fully synchronous design, but clock domain temperature only is responsible for the speed of the local clock. For computing the relative nominal speeds of the various clock domains as a function of the temperature, we use the model proposed in [2]:

$$f_{clk} \propto \frac{1}{\theta \cdot n_{cp}} \qquad (4)$$

where θ is the temperature and n_{cp} is the logic depth (that we assume identical across the different pipeline stages, given that the baseline was a balanced pipelined implementation). As it will be shown, "cooler" domains benefit from a faster locally generated clock, which overall translates into a potentially faster design.

Finally, to determine the maximum critical path distribution for each clock domain, we estimate the number of independent critical paths per domain as being proportional to the device count per domain: $N_{cp,\,domain} = \frac{N_{device,\,domain}}{N_{device}} \cdot N_{cp}$. As die area, as well as the total number of independent critical paths are a part of the Q metric above and a function of the overall device count, we present in the following details on estimating the total device count.

3.2 Total device count estimation

As described in Section 3.1, to get a sufficiently accurate estimate for the total number of independent critical paths N_{cp}, we need to rely on the total device count for the design. While the number of devices presented here assumes specific structures, the trends observed are likely to be reproduced using a more accurate model:

- *Rename table:* We assume that the baseline processor uses a MIPS-like [10] rename mechanism, where each physical register can hold the value for any ISA-defined register. We model a RAM-based version that uses the same number of entries as the architected register file. Each entry holds the identifier of the correct physical register, and it needs to be either read or written when the register is renamed. Thus, the number of RAM cells needed for the baseline processor is: $\log_2((N_{PhysicalRegs}))N_{ArchitectedRegs}$.
- *Branch predictor:* Assuming a global G-share predictor [9] with $N_{BpredSize}$ entries, the predictor table (assumed to be implemented with RAM cells) needs only two bits per entry, while the target buffer has the same number of entries, but 128 bits per entry (64 bits each for the destination address and tag). Thus, the total number of cells is estimated as: $N_{BpredSize} \cdot 2 + N_{BpredSize} \cdot (64 + 64)$.
- *Instruction Window* and *Load Store Queue*: Assuming a similar issue window organization as in [10], we consider that 128 bits would be enough to store the required information for each entry of the issue window. If the issue window size is $N_{WindowSize}$, the total number of latch cells needed for this structure is $N_{WindowSize} \cdot 128$. Two other pieces of logic are needed: the *select* logic responsible for sending instructions ready to issue to the appropriate functional units; and the *wake-up* logic, responsible for notifying dependent instructions that their operands become available. For the selection mechanism, we assume a tree implementation with eight requests per block. Each block takes 111 gates, if implemented with standard two-input cells, and the total number of blocks in an 8-ary tree with $N_{WindowSize}$ inputs is $N_{WindowSize} \cdot \frac{1}{8} \cdot \frac{1 - 1/N_{WindowSize}}{1-1/8}$. This logic needs to be duplicated for each issued instruction, thus the total number of gates for these request/grant blocks is: $((N_{WindowSize} - 1)/7 \cdot 111)N_{IssueWidth}$. $\approx N_{IssueWidth} \cdot N_{WindowSize} \cdot 16$

  For the *wake-up* mechanism, each entry compares the input registers with all tags coming from the functional units. Thus, each entry will use two comparators for each wake-up line and enough logic XOR gates to accommodate the total number of physical registers, for a total number of gates given by $2N_{PhysicalRegs}N_{IssueWidth}N_{WindowSize}$.
- *Register File:* The register file is modeled as a RAM-based structure with $N_{PhysicalRegs}$ lines and 64-bits wide entries, for a total of $N_{PhysicalRegs} \cdot 64$ cells.
- *Caches:* For the cache structures, the model assumes a RAM cell implementation with a total number of cells given by $N_{Sets} \cdot Associativity \cdot LineSize$.
- *Functional units:* We have considered that an integer 64-bit ALU requires about 10K static logic gates. For the floating point ALU, we estimated roughly a double number of devices (20K).

4 EXPERIMENTAL SETUP

We have implemented models for the quality metric Q on top of a cycle-accurate simulator similar to the one described in [11][8]. As opposed to SimpleScalar, it uses normal pipeline registers, separate Instruction Windows for each of the three execution clusters and a Retire Buffer. The register renaming mechanism chosen is similar to the one used by the MIPS R10000 processor. We have also moved the execution from Decode (as it is done in SimpleScalar) to the Execute stage, to better reflect the behavior of the pipeline.

In order to model a multiple clock-domain environment without any global synchronization point, we have developed an event-driven simulation engine. This event-driven simulation engine allows for any mixture of clocks running at different speeds and with different starting phases. We have used the Wattch framework [5] to include power models in our simulator. These power models (including the ones for the asynchronous communication) are integrated in both baseline and GALS versions to provide energy statistics. We have assumed that unused modules are clock-gated and are subject to leakage power. We have used the methodology proposed by Butts and Sohi [6] to estimate leakage energy. This model is based on using the total number of devices in conjunction with their type and a normalized leakage current to determine overall leakage power. The normalized leakage current per device was estimated as in [1].

In addition to modeling the switched capacitance of memories and buses, we also included models for the global clock grid and local clock grids corresponding to each of the synchronous domains. The area and metal density for each clock grid are the ones published for the Alpha 21264 processor. The parameters for the microarchitecture under consideration are presented in Table 1. In our experiments, we have used integer and floating-point benchmarks from both SPEC95 and SPEC2000 suites: {*ijpeg, gcc, gzip, vpr, mesa, equake, parser, vortex, bzip2, turb3d*}. For each benchmark, we have determined the performance (or IPC) and energy by skipping the first 500 million instructions and then performing detailed simulation for another 50 million instructions. For the GALS case, since the local clock signals are randomly staggered, simulations in this case were run three times, averaging the results. In the case where Dynamic Voltage Scaling (DVS) is used, we have used the best performing algorithm (as far as energy-delay product is concerned), that is, the threshold based scaling algorithm, with dependency information [13]. The thresholds (i.e., the issue queue occupancy) for scaling the local speed/voltage for each clock domain are given in Table 1, in addition to available speed and voltage levels. For the GALS architecture (with and without DVS), we have assumed an arbiter-based asynchronous FIFO communication. In this case, we can assume that the active clock edge in the producer signals the moment when data is available for reading (a consumer cycle can start). Thus, a subsequent active edge in the consumer can be accepted as a valid request, the setup time being already observed during the producer cycle.

Table 1: Microarchitecture settings

Parameter	Value
Pipeline	16 stages, 4 way out-of-order
Instruction Window	64 entries - 32 Int, 16 FP, 16 Mem
Load / Store Queue	32 entries
I-Cache	32K, 2 way set-associative, 1 cycle hit time, LRU replacement
D-Cache	32K, 4 way set-associative, 2 cycles hit time, LRU replacement
L2 Cache	Unified, 256K, 4 way set-associative, LRU replacement, access time 10 cycles
Memory access time	100 cycles
Functional Units	4 Integer ALUs, 2 Integer MUL/DIV 2 Memory ports 2 FP Adders, 1 FP MUL/DIV
Branch Prediction	G-share, 11 bits history, 2048 entries
Technology	0.13 um technology (high speed) V_{dd} = 1.8V, V_t = 0.2V
Normalized leakage current per device [1]	80 nA
Clock Speed / V_{dd}	250MHz - 1000MHz, 0.7V - 1.8V
DVS - Thresholds	Integer - 9, 12; Memory - 9, 12; FP - 6, 9
Frequency levels for the threshold - based DVS algorithm	Integer - High 1GHz, Low 750MHz FP - High 1GHz, Low 250MHz Memory - High 1GHz, Low 500MHz

As in [3], we assume that the gate length deviation due to WID variations varies little with the technology point, being around 10-12% for 0.13um, 0.09um and 0.065um if critical dimension (CD) control is employed. We have included the effect of gate length variability on overall performance by also including the estimated number of independent critical paths N_{cp} in the baseline, synchronous architecture (Figure 1(a)), and in each clock domain (as in Figure 1(b)). As in [4], we have used as an estimate for the synchronous baseline $N_{cp} = 100$. For the GALS case, in each clock domain, the number of independent critical paths is considered proportional to the number of devices per clock domain and estimated as in Section 3.2. In both synchronous and GALS cases, we have considered an equal effect of random and systematic WID variations (as this is how the model in [4] was validated).

5 EXPERIMENTAL RESULTS

To assess the impact of the proposed microarchitecture models, we have compared and contrasted a synchronous baseline implementation versus its GALS counterpart, with and without considering thermal considerations or DVS. We show in Figure 2 the normalized probability density for the maximum critical path delay due to WID variation (as in equations (1)-(2)) for the synchronous baseline and each clock domain of the GALS version (shown in Figure 1). In this case, all clock domains are assumed to run at the same nominal speed as the baseline. As it can be seen, due to a different (and reduced) number of independent critical paths per domain, the GALS version can be anywhere between 2% (Fetch/Decode, which has the largest device count among all domains) to 12% (Rename/Dispatch) faster than the synchronous baseline, the exact speed-up depending on the workload. The number of devices per domain is estimated as in Section 3.2 and is shown in Table 2.

If, in addition, thermal effects are considered to increase the nominal clock speed of locally synchronous domains (as in equation (4)), we get the results shown in Figure 3. Using the thermal model from [12] and equation (4), we show in Table 2 the relative speed-up for each clock domain with respect to the synchronous baseline clock speed. Since the nominal maximum critical path delay changes as well, the resulting distribution is different, as shown in Figure 3. In this case, local clock domains can be 8% (Memory) to 18% (Floating Point) faster than the synchronous case.

Table 2: Device count and relative speed-up based on temperature distribution for various local clock domains

Clock domain	Number of devices [%]	Speed-up factor
Fetch/Decode	50.65	1.11
Rename/Dispatch	1.22	1.00
Integer	11.05	1.09
Floating Point	8.59	1.15
Memory	28.47	1.06

Assuming that, in the most conservative case, the overall effective speed is given by the slowest clock domain (Fetch/Decode for Figure 2) and Memory (for Figure 3), we consider the Q metric defined as in (3), while also including the effect of performance (defined as IPC or CPI) and energy. We have simulated the set of ten Spec95 and Spec2000 benchmarks from Section 4 to obtain mean CPI and energy values to be used in equation (3) and combined them with the maximum critical path delay distribution of the slowest clock domain in the case of GALS architecture. We denote by “GALS” the architecture in which all clock domains run at the same speed as the baseline, “GALS-T” the case in which thermal considerations are included, while “GALS-T-DVS” the case where, in addition, voltage scaling is enabled for the back-end of the pipeline (Integer, Memory and Floating Point domains). To estimate die area in all four cases, we have used the device count as in Section 3.2 (for the GALS case, the added cost of mixed-clock asynchronous FIFOs is also included and accounts for about 2% of the total device count). As it can be seen in Figure 4, the mean value for Q is 2% (GALS), 9% (GALS-T) and 26% (GALS-T-DVS) better than the synchronous case.

We note that results presented here are conservative, in the sense that: (1) local speeds are likely to translate into faster overall GALS designs (we have used the slowest clock domain for determining the overall maximum critical path distribution); and (2) including the impact of other process and system parameter variability (e.g., leakage or voltage) as well as wire delay variability is likely to increase the gap between the fully synchronous and GALS implementations.

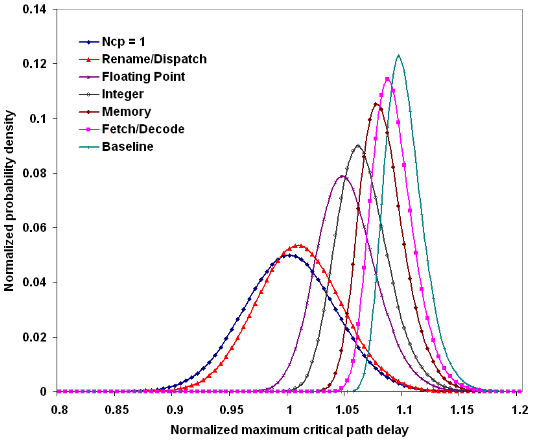


Figure 2. Probability density for the maximum critical path delay for the synchronous baseline and each clock domain of the GALS version without considering on chip thermal distribution. Results are normalized with respect to the single gate case ($N_{cp} = 1$).

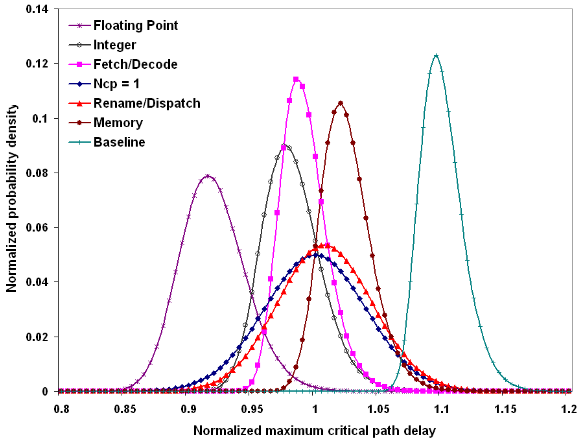


Figure 3. Probability density for the maximum critical path delay for the synchronous baseline and each clock domain of the GALS version when considering on chip thermal distribution. Results are normalized with respect to the single gate case ($N_{cp} = 1$).

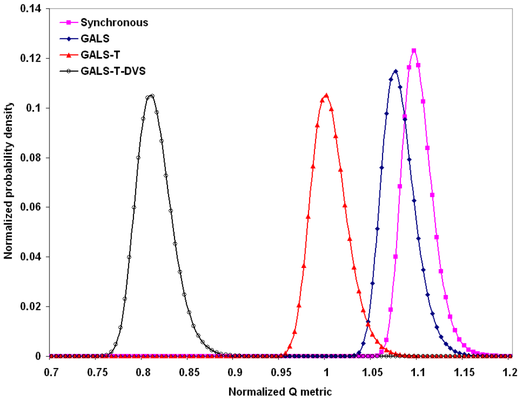


Figure 4. Probability density of the Q metric for the synchronous baseline and the GALS version with and without thermal considerations or DVS.

6 CONCLUSION

This paper has proposed a microarchitecture-drive variability modeling methodology which can be used to contrast and compare various energy aware design paradigms and their impact on overall design performance variability. As a design driver, the case of fully synchronous and multiple clock domain processors has been considered and impact of microarchitecture decisions on clock speed variability has been assessed. Results show that, when including the impact of gate length variability, a GALS design can provide a 2-12% speed-up, while also considering on-die thermal distribution provides a 8-18% speed-up when compared to the synchronous baseline. The use of voltage scaling provides 26% better combined power, performance and decrease in variability, thus increasing the likelihood of designs falling into a certain power/performance envelope. Future work includes addressing the impact of other variability factors, such as leakage current, voltage, and wire delay.

7 REFERENCES

[1] E. Acar, A. Devgan, R. Rao, Y. Liu, H. Su, S. Nassif, J. Burns, "Leakage and Leakage Sensitivity Computation for Combinational Circuits," in Proc. ACM/IEEE Intl. Symp. on Low Power Electronics and Design, pp. 96-99, Aug. 2003.

[2] A. Basu, S. Lin, V. Wason, A. Mehrotra, and K. Banerjee, "Simultaneous Optimization of Supply and Threshold Voltages for Low-Power and High-Performance Circuits in the Leakage Dominant Era," in Proc. ACM/IEEE Design Automation Conference, June 2004.

[3] S. Borkar, T. Karnik, V. De, "Design and Reliability Challenges in Nanometer Technologies," in Proc. ACM/IEEE Design Automation Conf., June 2004.

[4] K.A. Bowman, S.G. Duvall, J.M. Meindl, "Impact of Die-to-Die and WIthin-Die Parameter Fluctuations on the Maximum Clock Frequency Distribution for Gigascale Integration," in IEEE Journal of Solid-State Circuits, vol.37, no.2, Feb.2002.

[5] D. Brooks, V. Tiwari, and M. Martonosi, "Wattch: A Framework for Architectural-Level Power Analysis and Optimizations", in Proc. ACM Intl. Symp. on Computer Architecture, June 2000.

[6] J. A. Butts, and G. S. Sohi, "A Static Power Model for Architects," in Proc. Intl. Symp. on Microarchitecture, pp. 191-201, Dec. 2000.

[7] M. Eisele, J. Berthold, D. Schmidt-Landsiedel, R. Mahnkopf, "The Impact of Intra-Die Device Parameter Variations on Path Delays and on the Design for Yield of ow Voltage Digital Circuits," in Proc. ACM/IEEE Intl. Symp. on Low Power Electronics and Design, Aug. 1996.

[8] A. Iyer and D. Marculescu, "Power efficiency of Multiple Clock, Multiple Voltage Cores", in Proc. IEEE/ACM Intl. Conf. on Computer-Aided Design, San Jose, CA, Nov. 2002.

[9] S. McFarling, "Combining branch predictors", Technical Report DEC WRL Technical Note TN-36, DEC Western Research Laboratory, 1993.

[10] S. Palacharla, N. P. Jouppi, and J. E. Smith, "Complexity-effective superscalar processors," in Proc. ACM Intl. Symp. on Computer Architecture, June 1997.

[11] G. Semeraro, D.H. Albonesi, S.G. Dropsho, G. Magklis, S. Dwarkadas, and M.L. Scott, "Dynamic Frequency and Voltage Control for a Multiple Clock Domain Microarchitecture," in Proc. ACM Intl Symp. on Microarchitecture, Nov. 2002.

[12] K. Skadron, M. R. Stan, W. Huang, S. Velusamy, K. Sankaranarayanan, and D. Tarjan, "Temperature-Aware Microarchitecture," in Proc. ACM Intl. Symp. on Computer Architecture, June 2003.

[13] E. Talpes and D. Marculescu, "A Critical Analysis of Application-Adaptive Multiple Clock Processors," in Proc. ACM/IEEE Intl. Symp. on Low Power Electronics and Design, Aug. 2003.

[14] J. Tschanz, J.T Kao, S.G. Narendra, R. Nair, D.A. Antoniadis, A.P. Chandrakasan, V. De, "Adaptive Body Bias for Reducing Impacts of Die-to-Die and Within-Die Parameter Variations on Microprocessor Frequency and Leakage," in IEEE Journal of Solid-State Circuits, vol.37, no.11, Nov. 2002.