Parametric Yield Modeling and Simulations of FPGA Circuits Considering Within-Die Delay Variations

PETE SEDCOLE and PETER Y. K. CHEUNG
Imperial College London

Variations in the semiconductor fabrication process result in differences in parameters between transistors on the same die, a problem exacerbated by lithographic scaling. Field-Programmable Gate Arrays may be able to compensate for within-die delay variability, by judicious use of reconfigurability. This article presents two strategies for compensating within-die stochastic delay variability by using reconfiguration: reconfiguring the entire FPGA, and relocating sub-circuits within an FPGA. Analytical models for the theoretical bounds on the achievable gains are derived for both strategies and compared to models for worst-case design as well as statistical static timing analysis (SSTA). All models are validated by comparison to circuit-level Monte Carlo simulations. It is demonstrated that significant improvements in circuit yield and timing are possible using SSTA alone, and these improvements can be enhanced by employing reconfiguration-based techniques.

Categories and Subject Descriptors: B.6.1 [Logic Design]: Design Styles—Logic arrays; B.7.1 [Integrated Circuits]: Types and Design Styles—Advanced technologies

General Terms: Theory

Additional Key Words and Phrases: Delay, FPGA, modeling, process variation, reconfiguration, statistical theory, within-die variability, yield

ACM Reference Format:
1. INTRODUCTION

Variations in process parameters during semiconductor fabrication are manifested in the variability of the performance of the resulting integrated circuits. Historically, performance parameters have varied from wafer to wafer or lot to lot. At-speed testing techniques combined with speed-binning has been employed to partially compensate for variations in propagation delay between dice. In deep-submicron technology nodes, variations in transistor and wire parameters within the same die are expected to become significant [Nassif 2000; Visweswariah 2003]. The parametric difference between two nominally identical features on the same die is partly stochastic and partly correlated, with the correlation depending on physical locality. Importantly, several sources of stochastic variation are intrinsic to the materials used in fabrication [Asenov et al. 2002; 2003]. Stochastic variability cannot therefore be eliminated by improving the fabrication process, and is in fact predicted to increase relative to other sources of variability [Nassif 2000].

Like other high-performance integrated circuits, Field-Programmable Gate Arrays (FPGAs) are affected by parametric variability. However, their reconfigurability gives FPGAs two distinct advantages over ASIC solutions. Firstly, the actual performance of each FPGA can be measured and characterized by configuring the device with Built-In Self-Test (BIST) circuits. Secondly, in theory it is possible to compensate for, or even make use of, the variations in performance by adapting the application circuit based on the measured parameters of the target FPGA (see, for example, Katsuki et al. [2005]).

There are a number of ways in which a circuit could be made adaptive to within-FPGA variations in performance. The approach taken has a significant impact on the development of parametric test techniques, circuit design methods and tools. It is crucially important to quantify the performance improvement a given approach is expected to provide.

The contributions of this article are:

(1) we propose two generalised reconfiguration-based strategies for variation-adaptive circuits in FPGAs (in Section 3),
(2) the derivation of analytical models based on the statistical theory underlying each strategy, as well as statistical static timing analysis and worst-case design (Section 4), which describe the theoretical limits of each approach,
(3) the validation of the theory from VPR-based Monte Carlo simulations, and comparisons of the various techniques using the models (Section 5).

This article comprises an extension of the work we reported in Sedcole and Cheung [2007]. In comparison to our earlier work, the simulations presented in this article use VPR [Betz and Rose 1997] and are therefore more realistic. Furthermore, in order to apply the model to benchmark circuits, Section 5 includes a method to determine the key model parameter from actual circuits.
2. BACKGROUND

The manufacture of high-performance digital integrated circuits requires rigorous control over many process variables, each of which influences propagation delay to a different extent [Nassif 2000]. Deviations from nominal values in process variables can be systematic or stochastic [Cao et al. 2002; Kim et al. 2003]. The effect can be localized to a few transistors, a die, a wafer, or an entire lot. Systematic variations induce a shift in circuit parameters, sources of which include, for example, mask errors due to inaccuracies in the process model, lithographic off-axis focusing errors, and reticle stepper alignment errors. Stochastic variations cause circuit parameters to increase in spread, and comes from sources such as vibrations during lithography, wafer unevenness, and nonuniformity in resist thickness.

Importantly, some sources of stochastic variability are not caused by imperfections in the fabrication process but are the result of the discrete or granular nature of materials at nanometer scales. Such sources of variation are termed intrinsic and include line-edge roughness, random discrete dopants, and oxide thickness fluctuations [Asenov et al. 2002; 2003]. As these sources cannot be corrected by improving the process, they must be compensated for by new devices or novel circuit and system level design techniques.

Within-die variability exacerbates verification complexity. The conventional approach to verifying circuit designs in the presence of variation uses static timing analysis coupled with corner-case or Monte Carlo simulations. This is feasible if parameters are constant over the entire circuit. In the extreme case, where every transistor and wire in the design has parameters which vary independently, the dimensionality of the parameter space becomes too large for simulation-based methods to be practical.

Statistical static timing analysis (SSTA) is a promising new approach to timing analysis which incorporates the effects of within-die variations [Chang et al. 2005; Visweswariah et al. 2004]. The analysis can consider complete end-to-end signal paths or propagate statistically described delays block-by-block through the circuit. Lin, Hutton and Le recently applied SSTA to enhance placement and routing techniques in FPGAs [Lin et al. 2006].

Some research has been reported on reducing variability in FPGA architectures. Nabaa et al. [2006] describe a self-characterizing and adaptive FPGA which compensates for variability using body-biasing. Wong et al. [2005] determine yields using SPICE and numerical methods, and use the information to investigate the effect of LUT and cluster sizes. Matsumoto et al. [2007] proposed simultaneously with our earlier work [Sedcole and Cheung 2007] one of the techniques in this article (using multiple configurations) and presented results of VPR simulations. Their simulations only investigated configurations with different routing, and not placement. The work presented in this article differs in that it is based on analytical models of yield, and covers two reconfiguration-based and as well as static techniques. We also present comprehensive simulations to verify the models using VPR.
Delay variability testing is similar to delay fault detection, which is performed using at-speed tests. In FPGAs, at-speed testing can be performed by Built-In Self-Test circuits, which can be design-independent [Abramovici and Stroud 2003; Girard et al. 2004] or design-specific [Harris et al. 2001; Kraśniewski 2003]. Published work describing at-speed testing for within-FPGA variability includes Li et al. [2004] and Katsuki et al. [2006]. Li et al. used an array of ring oscillators to detect process variations in commercially available FPGAs from Xilinx. Katsuki et al. made measurements on a custom-designed 90nm LUT array, and also describe a yield enhancement scheme, whereby placement is optimised based on the measured LUT variations [Katsuki et al. 2005].

3. STRATEGIES
This section describes two strategies for variability-adaptive design in FPGAs. The focus is on delay variations. For reference, a typical signal path in an FPGA is shown in Figure 1. A path is composed of a number of elements, which can be considered to be individual basic features (such as LUTs, interconnect switches or wire segments) or groups of basic features. The propagation delay of the path is the sum of the element delays.\(^1\)

For completeness, “worst-case” timing analysis is considered first. In this case, parametric yield is a manufacturing issue for the FPGA vendor. The remaining techniques are based on achieving a design-specific yield determined by the end-user. It is outside the scope of this article to determine how these variability-adaptive strategies might be executed in practice. It is necessary for the end-user to be able to perform at-speed testing of the design in the FPGA, most likely using BIST techniques to avoid the expense of automatic test equipment. Moreover, it is assumed that BIST will indicate whether a given path or circuit passes or fails the at-speed test. It is improbable that path

\(^1\)Provided the signal path does not include chains of unbuffered wire segments, which would result in a more complex path delay function. In practice, FPGA routing is buffered, so this is a valid assumption.

delays would be quantified, much less delays of individual LUTs and wires, using BIST.

3.1 Worst-Case timing

The simplest timing strategy is to assign an upper bound on the value of the delay for each path element in the die. These “worst-case” values must take into account all sources of variability: within-die and between dice. In speed-binned FPGAs, the delay values can be determined during at-speed testing. If there is little within-die variability, testing is straightforward: each die can be characterized by measuring the delay of a localized test structure. However, if within-die variation is a significant part of overall delay variability, the test coverage must be either exhaustive, or at least sufficiently comprehensive to enable statistically reliable bounds on the slowest element in the die.

Using worst-case timing, all parametric yield issues are the responsibility of the FPGA vendor; end-user designs are guaranteed to operate correctly at-speed. The designed delay for a signal path is the sum of the worst-case delays of the individual elements. Under stochastic variation, it is improbable that all elements in a path will exhibit near worst-case delays, making the design overly conservative.

3.2 Statistical Static Timing Analysis

Research into statistical static timing analysis applied to FPGA designs has only recently begun [Lin et al. 2006]. It is instructive to examine the use of SSTA in FPGAs, particularly as a comparison for the reconfiguration-based techniques described below.

Statistical static timing analysis improves on worst-case timing by taking into account the probability that each path element has a given delay. The delay of a complete path can therefore be statistically described. Conventional STA will identify a single path in a given circuit implementation as the critical path, having the least slack. However, when implemented in FPGAs with significant within-die variation, the critical path may differ from device to device based on the variation in each die. Using SSTA, a circuit can be designed such that required timing is achieved at a given yield, taking into consideration all paths, not just the path with the least nominal slack.

SSTA can be path-based or block-based. A path-based scheme examines complete end-to-end signal paths separately, making it highly accurate but computationally expensive. Block-based schemes are faster and resemble conventional static timing analysers, in that maximum delay values are propagated through leaf nodes of a path in parallel. This is less accurate as the maximum of statistically described values can in general only be estimated.

Theoretically, using SSTA in FPGAs requires at-speed testing of all end-user products. The tests would need to be specific to the end-user designs. Testing could be neglected by selecting a sufficiently high design yield, such that the risk of not testing is acceptably low, a decision that would be application and market dependent. This strategy, although not using the full benefit of SSTA, nevertheless would outperform worst-case timing, and would be more
amenable to in-field upgrades. Ultimately, statistical static timing analysis enables a trade-off between product parametric yield and speed.

3.3 Multiple Configurations

We now examine the class of strategies which makes use of the reconfigurability of the FPGA, the first of which is predicated on the use of multiple implementations of the same circuit design. This approach was simultaneously proposed by Matsumoto et al. [2007] and ourselves [Sedcole and Cheung 2007]. Statistically, a given implementation of a circuit has a certain probability of passing at-speed testing when configured on an FPGA. If several implementations of the circuit are generated, then there is an increased probability that at least one of the implementations will meet at-speed requirements.

In this context, a circuit implementation is stored as a configuration bitstream. All configurations are functionally identical, and could be generated from the same netlist if the placement and routing of the netlist differs between configurations. Ideally, each configuration uses a different set of resources for the critical path (or near-critical paths); if resource usage is highly correlated between configurations the effectiveness of this technique is diminished.

This strategy requires a specific at-speed test for each configuration. Tests are run one by one in a given FPGA until a configuration is found which passes or all configuration options are exhausted. In the later case, the FPGA is failed.

Multiple configurations adds a degree of freedom (the number of configurations) to the design space, in addition to parametric yield and speed. The design will therefore theoretically outperform statistical static timing analysis.

The approach has limitations. Several circuit configurations and test configurations must be generated and stored. Storage can be particularly problematic if the strategy is implemented online in an embedded system. Design constraints will generally preclude completely uncorrelated configurations.

3.4 Region Relocation

The second strategy which exploits reconfiguration involves reconfiguring and relocating parts of a complete circuit. The premise is similar to the multiple configuration case: different implementations of the same circuit increase the probability that there exists one implementation that passes at-speed testing. In this case, instead of completely reconfiguring the FPGA, different configurations are created by partitioning the circuit into modules and then assembling the modules in different ways.

Different approaches to this strategy are illustrated in Figure 2. Fundamentally, the circuit design must be sufficiently modular that critical (or near-critical) paths are encapsulated in module blocks. Moreover, the design must support some degree of relocation of the modular blocks. This can include, for example, swapping the location of modules in the FPGA, or shifting modules into unused areas.

With appropriate constraints on the circuit design, it is possible to store the implemented circuit modules as partial bitstreams, and perform module relocation using dynamic reconfiguration [Sedcole et al. 2006]. A relocatable
at-speed test configuration is required for each module. Compared with the multiple configuration case, the amount of bitstream storage required is reduced, and the implementation of the circuit needs to be generated only once.

The approach has some limitations. The strategy would be most suitable for large systems comprising distinctly separate IP blocks connected by a system bus or on-chip network, since such systems are designed in an inherently modular way. Implementing relocatable modular circuits increases the complexity of system design, in particular in the connectivity between modules. An important constraint on this strategy is that the connections between the module blocks cannot become the critical path for the system, since there would then be no advantage in relocating the modules. Moreover, while there are many ways to assemble circuit modules to form different implementations of the system, the implementations are clearly not all independent. The space of potential solutions is therefore large and not trivial to search.

4. MODELING AND ANALYSIS

In the preceding section, several broad strategies for variability aware design were described. Before pursuing an implementation of any particular strategy, it is expedient to determine quantitatively the benefits the approach will provide. This section presents an analysis of each of the strategies of Section 3. It is emphasized that the objective of the analysis is to determine theoretical bounds on the relative yield or speed improvement of each approach given ideal conditions. It is not intended that the theory presented below should be used unaltered to predict the yield or speed of an actual implementation of a given strategy for a particular circuit, since practical considerations will invariably degrade the achieved improvements. Moreover, a number of
simplifying assumptions are made, which may not be applicable to actual implementations, but nevertheless do not compromise the relative comparisons between the strategies provided they are applied consistently.

In the models that follow, die-to-die variation is ignored as it can be accounted for by speed-binning. It is assumed that any within-die variation present is stochastic in nature; correlated within-die variation is negligible. This assumption is perhaps surprising given that measurements in 90nm technology have shown correlated variations can be significant [Sedcole and Cheung 2006]. However, there are indications that at smaller geometries correlated variability is becoming less significant [Zhao et al. 2007]. More importantly, the assumption is necessary for the analysis to be conservative: given that devices are speed-binned based on their slowest attributes, any correlated variation would cause parts of the FPGA to operate faster than the minimum specified by the speed-bin. The pathological case that the analysis must account for is where there is no correlated variation, and all parts of the device operate equally slowly, except for stochastic variation. For experimental results verifying this assumption, see Section 5.4.

4.1 Notation

Some of the notation used in the following analysis is listed in Table I. Other notation will be introduced as necessary. The error function, erf, has the usual definition:

\[ \text{erf}(z) \equiv \frac{2}{\sqrt{\pi}} \int_0^z e^{-x^2} dx. \]

The complementary error function, erfc, is defined as \( \text{erfc}(z) \equiv 1 - \text{erf}(z) \).

4.2 Worst-Case Timing

An FPGA has many types of primitives which may form elemental parts of a signal path, such as LUTs, wires, interconnect switch points, multipliers, and so on. For timing modeling, a path element is composed of the smallest interesting segment of a signal path, such as a LUT together with the input and output wiring. Parametric yield modeling can be applied to one or more different elemental types. Assume there are \( K \) types of element of interest for parametric yield in an FPGA, \( L_k \) elements of type \( k \), and the delay through
any given type $k$ element is normally distributed: $N(\mu_k, \sigma_k)$. The cumulative probability distribution for the delay of a type $k$ element is:

$$D_k(d_k) = \frac{1}{2} + \frac{1}{2} \text{erf} \left( \frac{d_k - \mu_k}{\sigma_k \sqrt{2}} \right).$$

(1)

In this equation the variable $d_k$ is the target delay for elements of type $k$. The parametric yield of an element type, $Y_k$, is an order statistic that depends on $L_k$, the total number of elements of that type in the FPGA. The manufacturing parametric yield of the FPGA is:

$$Y = \prod_{k=1}^{K} Y_k = \prod_{k=1}^{K} \left[ D_k(d_k) \right]^{L_k}. \quad (2)$$

We will assume here that the yield of each elemental type is balanced, such that $Y_k = Y \frac{1}{K}$. Physically, this means that a given FPGA has an equal chance of failing to meet the manufacturing timing requirements due to a slow LUT, a slow multiplier, or a slow routing switch. This is not a necessary assumption, as it is possible to bias the yield towards a particular resource, such as LUTs. Nevertheless, it makes comparisons of the strategies more meaningful.

Given the assumption of balanced elemental yields, the designed-for delay of a signal path of $N$ elements is:

$$T = \sum_{i=1}^{N} d_i = \sum_{i=1}^{N} \left[ \mu_i + \sigma_i \sqrt{2} \text{erf}^{-1} \left( 2Y \frac{1}{N} - 1 \right) \right].$$

(3)

where the $i$th element has mean delay $\mu_i$ and variance $\sigma_i^2$, dependent on its type. Considering the specific case where parametric yield is applied to LUTs only (for the sake of comparison) for an FPGA with $L$ LUTs, the relative target delay for a given yield $Y$ is:

$$\frac{T - \mu_\pi}{\sigma_\pi} = \sqrt{2N} \text{erf}^{-1} \left( 2Y \frac{1}{L} - 1 \right).$$

(4)

4.3 Statistical Static Timing Analysis

An ideal path-based statistical static timing analyser is examined here, which is assumed to be able to accurately predict the actual path delays. Thus, throughout the rest of the article, SSTA is synonymous with actual path delay.

For a single path $\pi$ in a given circuit implementation, with mean delay $\mu_\pi$ and variance $\sigma_\pi^2$, the yield of the path will be:

$$Y_\pi = D_\pi(T) = \frac{1}{2} + \frac{1}{2} \text{erf} \left( \frac{T - \mu_\pi}{\sigma_\pi \sqrt{2}} \right).$$

(5)

In general, a circuit implementation will have a number of paths that will contribute to yield loss. The impact each path has on the die yield is related to the path delay mean and variance. For simplicity, assume that $P$ of the paths have sufficiently little delay slack such that they impact on yield; these we will label “near-critical” paths. The remaining paths in the circuit have a negligible effect on yield. Moreover, assume each near-critical path has the same mean
delay and variance, and therefore the same yield. To be consistent, the same
approximation will be made when analysing the other strategies. The yield of
the circuit is the product of the yields of the paths:
\[ Y = D_\pi (T)^P. \]  (6)

The relative target delay for a given yield is:
\[ \frac{T - \mu_\pi}{\sigma_\pi} = \sqrt{2} \text{erf}^{-1} \left( 2Y^{\frac{1}{2}} - 1 \right). \]  (7)

This assumes that all the \( P \) paths are independent and separate. If the
near-critical paths share segments (because of signal path divergence or con-
vergence) the yield will be higher than (6), as the effective number of paths
(\( P_{\text{eff}} \)) is lower. In the limit, \( P_{\text{eff}} \to 1 \) as the correlation between paths
approaches unity and all critical paths converge to a single path.

4.4 Multiple Configurations
To be consistent with the SSTA evaluation, it is again assumed that a circuit
has \( P \) near-critical paths, each with mean delay \( \mu_\pi \) and variance \( \sigma_\pi^2 \). The yield
of an individual path is given by (5). For a single configuration, the die yield is
given by (6).

When multiple independent configurations are available, and the fastest
configuration is chosen for each FPGA through at-speed testing, the circuit
yield is:
\[ Y = 1 - \left[ 1 - D_\pi (T)^P \right]^C. \]  (8)

To derive this, note that for an FPGA to fail, it must fail at-speed tests for
all \( C \) configurations. The probability that it fails for a single configuration is
\( 1 - D_\pi (T)^P \), and therefore to fail in all configurations \( \left[ 1 - D_\pi (T)^P \right]^C \).

The relative target delay for a given yield is:
\[ \frac{T - \mu_\pi}{\sigma_\pi} = \sqrt{2} \text{erf}^{-1} \left[ 2 \left( 1 - (1 - Y)^{\frac{1}{2}} \right)^{\frac{1}{2}} - 1 \right]. \]  (9)

From (9) it is possible to determine how many independent configurations
are required to achieve a required yield \( Y \), given a target path delay of \( T \):
\[ C \geq \frac{\ln (1 - Y)}{\ln \left( 1 - \left( \frac{u + 1}{2} \right)^{\frac{1}{2}} \right)}, \]  (10)
where
\[ u = \text{erf} \left( \frac{T - \mu_\pi}{\sigma_\pi \sqrt{2}} \right). \]  (11)

It should be emphasized that this technique, and the analysis, are dependent
on independency of configurations. Configurations are independent if the
near-critical paths in each configuration use different resources. In practice,
this will not be the case. Correlations between configurations have the effect of
reducing the effective value of \( C \). The limiting case is where each configuration
uses exactly the same resources, and the effective value of \( C \) is unity.
4.5 Region Relocation

The strategy of subdividing the circuit into many separate modular regions that can be assembled in different ways is next considered. Assume that the modularization of a circuit creates $R$ identical regions and $R$ subcircuit modules, each of which can be assigned to any of the $R$ regions. Clearly, there are $R!$ possible permutations for placing the subcircuits. The yield of this strategy is the probability of finding at least one assignment within the $R!$ implementations where all subcircuits function (that is, pass at-speed testing). However, unlike the multiple configuration scheme, the implementations are not independent.

Assume that the circuit is subdivided evenly into $R$ subcircuits such that each subcircuit has $P$ critical paths. This we will term a balanced division. Unbalanced divisions will be examined later.

**Theorem 4.5.1.** Given a balanced subdivision of a circuit with $P$ near-critical paths into $R$ subcircuit modules, and considering all possible assignments of modules to regions, the yield of the system can be approximated by:

$$ Y \approx (1 - (1 - q)^R)^{2R}, $$

where

$$ q = D_\pi(T)^\frac{P}{R} = \left[ \frac{1}{2} + \frac{1}{2} \text{erf} \left( \frac{T - \mu_\pi}{\sigma_\pi \sqrt{2}} \right) \right]^{\frac{P}{R}}. $$

Note that $q$ is the yield of an individual subcircuit module assigned to a single region.

**Proof.** The subdivision of the circuit is balanced. Therefore, it is reasonable to assume that any given pairing of a subcircuit and a region will have a fixed probability of functioning, given by $q$ in (13). The probability that a given subcircuit does not function in a given region is $(1 - q)$.

The yield of the region relocation scheme can be derived by determining the probability that of the $R!$ possible assignments of subcircuits to regions, no combination can be found where all subcircuits function. We denote this event (that no combination works) as $E$. It is conjectured that $E$ mostly occurs due to one of two scenarios: either there a subcircuit that does not function in any region, or there is a region in which none of the subcircuits function. This does not cover all causes of $E$, such as there being two modules that function only in the same one region. Nevertheless, the probability of such cases occurring are sufficiently small that they may be ignored.

For a given subcircuit module $m_i$, the probability that it does not function in any region (event $F_{m_i}$) is:

$$ P(F_{m_i}) \equiv P(m_i \text{ cannot be placed}) = (1 - q)^R. $$

Since there are $R$ modules, the probability that there is a subcircuit which cannot be placed is:

$$ P(F_m) = P \left( \bigcup_i F_{m_i} \right) = 1 - (1 - (1 - q)^R)^R. $$
Similarly, over all $R$ regions, the probability there exists a region in which no subcircuit works is:

$$P(F_r) = P\left( \bigcup_i F_{r_i} \right) = 1 - (1 - q)^R. \tag{16}$$

where $F_{r_i}$ is the event that no subcircuit functions in region $r_i$. The yield is therefore:

$$Y = 1 - P(E) \tag{17}$$

$$\approx 1 - (P(F_m \cup F_r)) \tag{18}$$

$$\approx 1 - (P(F_m) + P(F_r) - P(F_m)P(F_r)). \tag{19}$$

which can be reduced to (12) by substitution of (15) and (16). Note that events $F_r$ and $F_m$ are only weakly dependent for large $R$, and so $P(F_m \cap F_r) \approx P(F_m)P(F_r)$. \hfill \Box

The relative target delay given a yield $Y$ is:

$$\frac{T - \mu}{\sigma} = \sqrt{2} \text{erf}^{-1}\left[2 \left(1 - \left(1 - \frac{Y^{\frac{1}{2}}}{2}\right)^{\frac{1}{2}}\right)^{\frac{1}{2}} - 1\right]. \tag{20}$$

If the subdivision of the circuit is unbalanced, the yield will be reduced. The limiting case is where all $P$ near-critical paths are allocated to a single subcircuit. There are then $R$ different placements of this subcircuit, while the placement of the remaining subcircuits is irrelevant. The yield is then similar to the multiple configuration case:

$$Y_{\text{unbalanced}} = 1 - \left[1 - D_{\pi}(T)^{Pr}R\right]. \tag{21}$$

In practice, it may not be reasonable to have all regions and modules the same size. An alternative may be to divide the regions and modules into two sizes: large and small. Large modules would be placed in large regions, and small modules in small regions. More generally, consider the case where there are $S$ differently sized regions. There are $R_i$ regions of size $i$ and $P_i$ critical paths in the corresponding $R_i$ modules. The overall timing yield can be calculated from the product of each of the $S$ independent yields:

$$Y = \prod_i Y_i \tag{22}$$

$$Y_i \approx (1 - (1 - q_i)^{R_i})^{2R_i} \tag{23}$$

$$q_i = D_{\pi}(T)^{\frac{P_i}{R_i}} = \left[\frac{1}{2} + \frac{1}{2} \text{erf}\left(\frac{T - \mu_{\pi}}{\sigma_{\pi} \sqrt{2}}\right)\right]^{\frac{P_i}{R_i}}. \tag{24}$$

For the experimental results in this article, we choose to assume balanced regions of equal size.

Table II. Summary of the Analysis

<table>
<thead>
<tr>
<th>Strategy</th>
<th>Yield Expression</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSTA</td>
<td>( Y = D_\pi(T)^P )</td>
</tr>
<tr>
<td>Mult. conf.</td>
<td>( Y = 1 - \left[ 1 - D_\pi(T)^P \right]^C )</td>
</tr>
<tr>
<td>Region reloc.</td>
<td>( Y \approx \left[ 1 - (1 - D_\pi(T)^P)^R \right]^{2R} )</td>
</tr>
</tbody>
</table>

where \( D_\pi(T) = \frac{1}{2} + \frac{1}{2} \text{erf} \left( \frac{T - \mu_\pi}{\sigma_\pi \sqrt{2}} \right) \)

4.6 Summary

The derived expressions for the yields of the different strategies are summarized in Table II.

5. EXPERIMENTS

The theory presented in the previous section uses simple characteristic parameters of a circuit to estimate the performance and yield of each strategy. It is assumed that the circuit has a number of independent critical paths \( P \), each of which have the same nominal delay and variance. This section describes the methodology used to apply the theory to realistic circuits, and experiments performed using VPR [Betz and Rose 1997]. The objectives of the experiments are twofold. The first is to validate the theoretical results derived in the previous section. Moreover, it is of interest to examine and compare the relative yield enhancement offered by the alternative strategies under different conditions.

5.1 Methodology

A number of Monte Carlo simulations have been performed using the VPR software tool. The software has only been modified to gather more information on the placement and routing than is provided by default. In particular, details of the specific routing resources for each path in the circuit are extracted. There are three types of delay elements in the abstract architecture employed by VPR: (1) LUTs, (2) global routing, and (3) local routing. Delays in global routing are due to wire segments and switches connecting wire segments. Fully buffered switches have been used in the simulations so that the delays can be isolated. Local routing delays occur from the input pin of a logic block to the input pin of a LUT.

For this work, the deterministic timing figures produced by VPR are used as a reference, and are chosen to be nominal (zero variation) delays. Random delay variations for every element type were generated for a set of 1000 different devices. The timing for a worst-case design implementation is found by adding guard band values to the delays of every LUT, wire segment and local route and recalculating the signal delays of all paths in the given circuit. The device yield is found by comparing the guard band values with the generated elemental variations.

The multiple configuration strategy requires many implementations of the same circuit, with each implementation having the same timing. Unfortunately, VPR is designed to produce “best effort” timing results, rather than timing driven by constraints. This results in a significant variation in the nominal
timing of the critical path produced by the tool depending on the starting conditions and random seed values. In contrast, a timing-constraint driven place and route tool, when given different starting conditions and random seed values, would produce designs with very similar critical path delay. To overcome this limitation of VPR and generate implementations that more closely match the output of a constraint-driven tool, we produce many more implementations of the circuit than are needed (in this case 40), initializing VPR with a different random seed to generate each implementation. From these we select a subset that have the closest nominal critical path timing. Each implementation of the circuit uses the same pin constraints, as would be required in a real scenario.

To simulate the region relocation strategy, we assume a large circuit made from several copies of one MCNC benchmark circuit. Each copy corresponds to one region. The approach artificially ensures that the circuits in each region are all balanced. Moreover, it does not take into account any communication that may be required between regions. However, it is sufficient as an approximation at this level of examination. The circuit implementations from the multiple configuration investigation are reused for this part of the study.

The theoretical predictions for the yield of the Statistical Static Timing Analysis, multiple configuration and region relocation strategies all use a parameter $P$ for the number of independent critical paths in the circuit. An estimation of $P$, which we will call the effective value, $P_{\text{eff}}$, is calculated using principal components analysis. The procedure is as follows.

1. From the VPR implementation of a circuit, the critical path is identified using the worst-case timing analysis. The nominal timing of this path is used for $\mu_{\pi}$, and the variance of the path delay $\sigma^2_{\pi}$ calculated by summing the delay variances of the path elements.

2. Paths with nominal timing slower than a threshold of the critical path nominal timing are identified.

3. A correlation matrix $V$ is constructed, where each entry $V_{ij}$ in the matrix is the correlation in delay between two paths $\pi_i$ and $\pi_j$:

$$V_{ij} = \text{cor}(\pi_i, \pi_j) = \frac{\sum_{k \in S} \sigma^2_k}{\sigma_{\pi_i} \sigma_{\pi_j}}.$$  (25)

Here, $S$ is the set of path elements that the two paths have in common.

4. The eigenvalues of $V$ are calculated, and the scree test [Cattell 1966] applied to estimate the equivalent number of independent paths.

Although this method is somewhat imprecise, it provides a sufficient approximation for $P_{\text{eff}}$.

5.2 VPR Results

Here, the results of the VPR experiments are described. In all the experiments, the delay through any particular path element is normally distributed. The delays through each type of path element (LUTs, local routing and global routing) have standard deviations of 5%. All results use an FPGA model with parameters as summarized in Table III. The device size and routing channel
width was set to the minimum required to reliably place and route each circuit. The input/output blocks were made dense enough to avoid the device size being influenced by I/O constraints. Technology attributes were specified to ensure that routing delays for any particular path were comparable to logic delays, to match the characteristics of recent FPGAs.

To start with, results from a single circuit (alu4) from the well-known MCNC benchmark set are examined. Ten implementations (placement and routing) with similar timing were created using VPR. The eigenvalues of the equivalent independent critical paths (which we call eigen-paths) as found by principal components analysis are plotted in Figure 3. It can be seen that although the circuit has several thousand signal paths, very few are close to the critical path delay. Moreover, the equivalent number of independent paths is less than 5. Applying the scree test a value of 2 is chosen for $P_{eff}$ in this case.

The timing yield using the worst-case design strategy is plotted for all ten implementations of the alu4 circuit in Figure 4. The timing is normalised to
the nominal delay of the design, which would be achieved if there were no variability in the delays. It is important to emphasise that the curves are for the manufacturing yield: if the FPGA vendor chooses decrease the threshold for the slowest acceptable elemental delay, thereby discarding more devices, the guaranteed worst-case timing will improve, but not by much. Even though circuit implementations with similar timing have been selected, there is still a spread of about ±3.4% in the circuit timing. Note that the theoretically predicted yield matches the yield measured from the simulations.

Using high-quality statistical timing analysis the actual delays of the implemented circuits can be predicted. It is assumed that SSTA is employed to predict the speed of a design after it has been placed and routed. Different results would be obtained by using SSTA during the place and route process as this would alter the circuit implementation. The yields of all ten implementations for the actual delays are plotted in Figure 5, as well as the yields estimated by the theory. Here the theoretical results are sensitive to the value of the nominal critical path delay, \( \mu_\pi \). Nevertheless, the theory predicts the actual results well, with a mean absolute error in the delay prediction for the 85% yield point of 0.23%.

The multiple configuration strategy is examined next. From the Monte Carlo simulations, for each device the fastest of the ten implementations is chosen. The results of this are plotted in Figure 6. To generate theoretical predictions a single value for the nominal critical path delay \( \mu_\pi \) is needed.
However, this value changes between implementations of the circuit, by up to ±1.8%. Using the two extreme values, bounds can be calculated for the yield, as are shown on the graph. Using the midpoint of the two bounds as an estimation, the difference between actual and predicted performance at the 85% yield point is 0.94%.

In order to simulate the region relocation strategy, we assume a large design is constructed by assembling nine copies of a single benchmark circuit, arranged in a three by three grid. A different implementation of the circuit is required for each copy; nine of the implementations from the multiple configuration experiment are reused for that purpose here. A small adjustment is made to the timing of each implementation to remove the variance introduced by VPR. This is necessary to avoid the overall limiting delay of the large circuit being dominated by the slowest of the nine implementations, which would then skew the results.

Figure 7 shows the outcome of this experiment. The yield of the initial assembled circuit and the improved yield from using region relocation are both plotted. The initial circuit yield is predicted as previously by assuming an ideal statistical static timing analyser. Since the assembled circuit is nine repetitions of the original benchmark, it could be assumed that the number of equivalent critical paths $P_{\text{eff}}$ is the sum of the individual critical path values from the individual circuits. However, with a higher number of critical paths, any given path must be closer to being critical to have a significant probability...
of being critical. The actual number of eigen-paths for the assembled large design is estimated to be 9.

The upper bound and lower bounds for the improvement in yield and performance, as shown in Figure 7, are found using (12) and (21) respectively. The yield curve from the Monte Carlo simulations falls between these two extremes, as predicted. As before, using the midpoint of the two bounds to estimate the actual improvement gives an error of 0.85% at the 85% yield point.

Having examined in detail the results for a single circuit from the MCNC benchmark set (alu4), it is instructive to compare the results across the set. The three experiments were performed on 15 MCNC circuits; the results are listed in Table IV. The table lists two properties of the benchmarks; the nominal delay of the deterministic critical path, $T_c$, and the equivalent number of independent critical paths, $P_{eff}$.

For each circuit, the relative improvement in the maximum operating frequency $f_{max}$ is given at a yield point of 85% using each of the strategies. The Statistical Static Timing Analysis (SSTA) case is compared with worst-case (W.C.) design. This experiment compares the means of the actual and predicted results. The improvements using multiple configuration (M.C.) and region relocation (R.R.) strategies are given relative to using actual (SSTA) timing. The theoretical value is calculated from the midpoint of the bounds. It should

Fig. 6. The timing yield achieved by choosing the best alu4 implementation for each given FPGA. The bounds of the original circuit yields are shown by the solid lines, while the bounds of the multiple configuration case are given by the dashed lines.
Fig. 7. The timing yield of the region relocation experiment, using the alu4 circuit replicated 9 times.

Table IV. Characteristics of the MCNC Benchmark Circuits and Results of Experiments

<table>
<thead>
<tr>
<th>Benchmark circuit</th>
<th>Nominal paths</th>
<th>Critical improvement in $f_{\text{max}}$ (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Simulation</td>
<td>Theory</td>
</tr>
<tr>
<td>alu4</td>
<td>2</td>
<td>26.8</td>
</tr>
<tr>
<td>apex2</td>
<td>3</td>
<td>26.1</td>
</tr>
<tr>
<td>apex4</td>
<td>2</td>
<td>25.7</td>
</tr>
<tr>
<td>bigkey</td>
<td>3</td>
<td>27.9</td>
</tr>
<tr>
<td>des</td>
<td>2</td>
<td>25.9</td>
</tr>
<tr>
<td>dsip</td>
<td>3</td>
<td>24.5</td>
</tr>
<tr>
<td>ex1010</td>
<td>3</td>
<td>26.5</td>
</tr>
<tr>
<td>frisc</td>
<td>2</td>
<td>29.0</td>
</tr>
<tr>
<td>misex3</td>
<td>2</td>
<td>25.2</td>
</tr>
<tr>
<td>pdc</td>
<td>2</td>
<td>26.8</td>
</tr>
<tr>
<td>s38584.1</td>
<td>3</td>
<td>30.6</td>
</tr>
<tr>
<td>seq</td>
<td>3</td>
<td>27.3</td>
</tr>
<tr>
<td>spla</td>
<td>2</td>
<td>26.8</td>
</tr>
</tbody>
</table>

be emphasised that the multiple configuration case cannot be compared here with the region relocation strategy because in the later scenario the implemented circuit is nine times larger than the original benchmark circuit.

In general the predictions give reasonably accurate estimations of the actual performance measured from the simulations. Moreover, while statistical
Fig. 8. The yields using actual timing for six of the MCNC benchmark circuits. These simulated results show that timing is less affected by variability in circuits that have longer critical paths.

timing analysis gives a significant gain in performance, the gain from using reconfiguration is incremental in these results.

5.3 Exploration

Performing experiments using benchmark circuits give credence to the predictions of the probabilistic analysis. However, it is also of interest to examine the impact that different parameters have on the outcome of the strategies, as well as making a comparison of statistical timing analysis with the two reconfiguration based approaches. For these purposes the theoretical model is invaluable.

From the details listed in Table IV it appears that the improvements in performance are indifferent to the length of the critical path in the circuit. This is counter-intuitive, as one might expect that variability would have a greater impact on circuits which have short critical paths; with longer critical paths the variations from each element in the path would “average out”. If the actual yields of different circuits are plotted, as in Figure 8, it can be seen that path length does have an effect: greater variation is present in circuits with shorter critical paths, resulting in lower yields. However, the magnitude of this effect is outweighed in the VPR-based experiments by the variation between circuit implementations.

A noticeable feature of the VPR-based experiments is the low number of independent critical paths $P_{\text{eff}}$ for all circuits. No circuit has a $P_{\text{eff}}$ value higher
than 3. This is partly due to the small size of the benchmark circuits. However, it also likely to be an artifact caused by the place and route algorithms in VPR, which have relatively simple strategies to minimise the critical path delay. Commercial place and route tools, which are capable of producing constraint-based implementations, tend to be much more aggressive and thereby create a so-called timing-wall; a large number of paths with timing close to critical.

Figure 9 plots the simulated and theoretical yield curves for circuits with different numbers of critical paths $P_{eff}$. It is clear that a higher number of critical paths results in a slower circuit at a given yield. For the remaining investigations in this section, a circuit implementation with 50 critical paths is assumed.

Figure 10 compares the benefits of the two reconfiguration-based strategies at high yield values. In Figure 10, the timing achieved by using between 2 and 10 configurations, and between 2 and 9 regions, are plotted and compared with SSTA. It can be seen that increasing the number of configurations provides rapidly diminishing returns. By comparison, subdividing the circuit into more regions provides greater gains in performance.

Another interpretation of this is given in Figure 11, which shows the minimum number of configurations or the number regions required to meet a given yield and timing target. The number of configurations required to meet a
high yield target increases dramatically as the timing is made more aggressive. This is significant, as an increase in the number of configurations leads to a commensurate increase in the storage required for configuration data. Increasing the number of regions a circuit is divided into has a minimal effect on the storage required for configuration data. However, subdividing a circuit into relocatable modules has several non-trivial practical issues to overcome, not least connecting the subcircuits together after they have been relocated so that they can communicate.

A final comparison is given in Figure 12, where the performance of all four strategies is presented for a circuit with $P = 50$ in a device with 10000 LUTs. In this graph, the yield curves are plotted for 10 configurations and 9 regions. The timing degradation due to worst-case design is severe. However, SSTA can compensate for most of this if the quality of the SSTA tool is sufficiently high. The reconfiguration techniques provide incremental improvements beyond this improvement.

5.4 Correlated Variability and Speed-Binning

The analysis and experimental results thus far all use the assumption that delay variability is entirely stochastic and uncorrelated. We now briefly examine the effect that spatially correlated variation would have on the results.
Fig. 11. A comparison of the number of configurations or the number of regions required to meet a yield and timing target. The curves are circuit timing targets, expressed in the offset from the nominal critical path delay ($\mu_\pi$) by a number of standard deviations ($\sigma_\pi$).

This necessarily incorporates speed-binning, which is conventionally used when process variation causes a set of dice to exhibit differences in the mean delay.

For this experiment, the MCNC benchmark circuit alu4 was used. Correlated variations were added to the stochastic variations in all elemental delays. A simple two-dimensional quadratic model was used to represent correlated/global variations:

$$
\delta(x, y) = Ax^2 + By^2 + Cx + Dy + E,
$$

(26)

where each coefficient is a normally distributed random variable and $\delta(x, y)$ is the relative change in delay of an element at location $(x, y)$ in the device. For this experiment, $\delta$ has a large range of $(-0.55, 0.53)$ in order to emphasise the effects of these variations.

The objective of speed-binning is to group together devices with similar delays. Different strategies for defining the limits of the bins and allocating devices to bins may be defined. Conventionally, a worst-case timing strategy is followed: a device is assigned to a bin based on the delay of its slowest element. Although it is theoretically possible to measure the delay of every element in a given FPGA before allocating it to a speed-bin, it is assumed here that this is impractical. Instead, we assume a speed-binning decision is made considering...
the worst-case correlated variation in each device only. This can be estimated with relatively few measurements.

The result of the speed-binning strategy for the experimental simulation is depicted in Figure 13. The yield of the worst-case correlated delay is plotted, and the devices are allocated to one of three speed bins based on yield points (chosen arbitrarily to give bin ratios of approximately 1:2:3) of 17%, 50% and 99%. The slowest 1% of devices are discarded in this case.

The resulting worst-case timing yields for each of the three bins are plotted in Figure 14. These include the stochastic variations in addition to the correlated variations in delay, and therefore the worst-case path delay is larger than the boundary point of the bin. The theoretical lower bounds of the delay-yield curves can be obtained using the theory presented in Section 4.2 and assuming each device exhibits pathological correlated variability. The pathological case is where the correlated delay of all devices in a given bin is the same and equal to the speed-bin boundary. The lower bound is then given by (4), renormalized to the bin boundary.

For example, the devices in the speed bin 1 of Figure 13 have worst-case correlated delays of 1.062 times nominal or better. The lower bound of the delay yield is given by (4), assuming that the mean path delay $\mu_\pi$ is 1.062 times the nominal designed delay.
Thus, the effect of spatially correlated variation, to a first approximation, is to cause a shift the mean nominal delay. By including this shift, the analysis presented in Section 4 can be used as a conservative estimate of the true yield. Importantly, the comparison between the different timing strategies is unaffected. It may be noted that the level of spatially correlated delay variation used here in this illustrative experiment is deliberately high, and the effect would be less pronounced in reality. Moreover, as mentioned in the introduction, the relative significance of stochastic variability is predicted to increase [Nassif 2000]. Nevertheless, future work may incorporate spatial correlations into the models to enhance the accuracy of estimated yields.

6. CONCLUSIONS AND FUTURE WORK

Within-die delay variation will become increasingly significant in future technology nodes. This article presented two strategies for compensating for variability by exploiting the reconfigurability of FPGAs. The techniques involve reconfiguring the entire FPGA and relocating subcircuits to different regions within the FPGA. Using probability theory, the yield of each approach was modeled and compared with statistical static timing analysis and worst-case design, demonstrating the benefits of reconfiguration-based techniques. The theoretical analysis has been shown to be valid with VPR-based
experiments. The results show that the present technique of timing, which assumes worst-case variation for all path elements, dramatically over-estimates the actual circuit delays. Using statistical static timing analysis compensates significantly for this pessimism. Exploiting reconfiguration further improves the achievable timing performance and yield of circuits implemented in FPGAs. Although the incremental improvement of using reconfiguration is not as significant as the initial gain by using SSTA, using reconfiguration will become more valuable as process variability increases.

The analysis presented in this article provides a foundation from which to explore delay variability adaptive design in FPGAs. Further work is required to investigate the practical feasibility of the proposed approaches, including examining the limitations of subdividing a large circuit into relocatable modular blocks and the implementation of a communication infrastructure to support the module relocation. The effectiveness of the reconfiguration approaches may be altered by incorporating statistical static timing analysis into the optimisation of circuit design. Moreover, it would be instructive to remove some of the more conservative assumptions from the analysis by incorporating the effects of die-to-die variability within a speed grade and, by extension, correlated within-die variability into the models.

ACKNOWLEDGMENT

Thanks to Dr. R. Sedcole for advice and suggestions on statistical theory.

REFERENCES


Received May 2007; revised September 2007, January 2008; accepted April 2008