# Statistical Static Performance Analysis of Asynchronous Circuits Considering Process Variation

Mohsen Raji, Behnam Ghavami, Hossein Pedram
Amirkabir University of Technology, 424 Hafez Avenue, Tehran, I.R. Iran
E-mail: {mohsen.raji,ghavami,pedram}@aut.ac.ir

**Abstract** — Asynchronous logic is a hot topic due to its interesting features of power saving, low noise and robustness to parameters variations. Beside of the benefits of asynchronous design technique, the lack of automatic design and analysis tools made it hard to apply them in the new designs. Timing analysis is a necessary step in automatic design process and optimization of asynchronous circuits. On the other hand, increasing variation of process parameters of integrated circuits and more sensitivity of today's designs has increased the necessity of statistical approach to timing analysis of asynchronous circuits. So it seems to be necessary to introduce a method to the performance analysis of asynchronous circuits considering the variation in process parameters. In this paper, we present a novel method to analyze the performance of template-based asynchronous circuits statistically. Asynchronous circuit has been modeled using Variant-Timed Petri-Net. Based on this model, the probability density function of the delay of global critical cycle is calculated. The results of the experiments are compared with Mont Carlo simulation-based results and the average error is %2.8 for the mean value of the delays.

**Keywords**
Asynchronous circuits, process variation, Petri-Net model, Statistical static timing analysis.

## 1. Introduction

In asynchronous circuits, local signaling eliminates the need for global synchronization which exploits some potential advantages in comparison with synchronous ones [1][2][3][4][5]. They have shown potential specifications in low power consumption, design reuse, improved noise immunity and electromagnetic compatibility. Asynchronous circuits are more tolerant to process variations and external voltage fluctuations[1].

One of the interesting features of asynchronous circuits comparing with their clocked counterparts is better average-case performance[4][5]. However, the performance analysis of asynchronous circuits is a complicated problem and without an effective performance analysis method, one cannot easily take advantage of the properties of asynchronous systems to achieve optimal performance[6]. There are two major technical difficulties involved in the performance analysis of asynchronous systems. First, unlike clocked circuits where clock boundaries form natural partitions for logic between stages to be analyzed individually, an asynchronous circuit is inherently nonlinear, meaning there is no easy way to partition the system into independent sub-systems. The systems have to be analyzed as a whole. Second, as the functionality of the system is dependent on the concurrent events, variations in data-dependent delays mostly caused by process variation in individual components can have considerable effect on the performance of the system. As a result, performance analysis based on the average delay is not accurate. As process variations become a significant problem in deep sub-micron technology, it is really necessary to shift from deterministic timing analysis to statistical timing analysis for high-performance asynchronous circuit designs similarly to what is done in synchronous ones[7][8].

There are few works that consider probabilistic delay models in the performance analysis of asynchronous systems [6][7][10][11]. Earlier in [6], a method for analyzing the asymptotic performance of asynchronous systems considering delay variation was presented. However, it focused on modeling at system architecture level and did not consider circuit modeling issues such as delay variations due to the process variation. Recently Yahya [12] proposed a performance model for asynchronous linear-pipeline with time variable delays. But in practical designs, asynchronous pipelines can be linear or non-linear and their method cannot be applied to non-linear models.

This paper introduces a novel and efficient performance analysis method for asynchronous circuits considering process variation. In our approach a synthesized template-base asynchronous circuit is modeled as a Variant-Timed Petri-Net that captures concurrency between interactive components in decision-free[13] systems. Delay variations in component delays caused by process variation are captured in a probabilistic delay model. As the variability of process parameters is naturally random, in order to analyze the effect of the process variation, we applied a statistical approach to the proposed timing analysis. The main goal of the proposed statistical timing analysis is to calculate the probabilistic density function of a performance metric. The proposed method is implemented and tested on the benchmark circuits modeled in Variant-Timed Petri-Net and its results are compared with the Monte Carlo simulation-based results. The difference between the two methods was shown to be small, ranging from 0.6% to 5.5%.

The remaining part of the paper is organized as follows; in section 2 an overview to asynchronous circuits design is presented. Section 3 describes Variant-Timed Petri-Nets as the dominant performance analysis model. Section 4 discusses the proposed statistical static performance evaluation framework in detail while section 5 gets on with the results and analysis. In section 6 we explain about the modeling and its issues. In the last section, we conclude the paper and probable future works in this scope are proposed.

## 2. Background

This section reviews technical backgrounds on modeling necessary for the development of this paper.

## 2.1 Asynchronous Circuit Design and Performance Analysis

Asynchronous circuits rely on exchanging local requests and acknowledge signaling for the purpose of synchronization. While the first generation of asynchronous synthesis tools were mainly focused on control synthesis which make them inapplicable for practical circuits, the new generation of synthesis methods target the use of pre-designed asynchronous buffer templates[14][15].

The asynchronous synthesis process usually starts with a high level circuit specification. By applying a set of function preserving transformations to the high level specification, decomposition method[14] divides the original circuit to a set of communicating pre-defined templates. The resulted circuit then can be directly implemented down to the layout or some optimization such as sizing, threshold voltage assignment or slack-matching may be done to improve the overall performance and power of the circuit[16][17].

Performance analysis in asynchronous circuits typically starts with Petri-Net or marked graph representations, and attempts to run a simulation to infer time separation between events (TSE) in the system. The objective is to find bounds on TSEs in the presence of runtime variability. Xie et. al. [18] notes that random simulation is a familiar method for obtaining TSEs for large-scale, complex systems.

However, an effective method for focusing optimization effort on the design flow has the need of determining the performance metric of a design with using a static manner. Earlier works on static performance evaluations of asynchronous circuits rely on analytical performance analysis. Event Rule System[19], and a work on the computation of the Timed Separation of an Event[20] are good examples for such static performance estimation schemes. These techniques usually use Timed Petri-Nets for the purpose of modeling.

## 2.2 Timed Petri-Net Model

Petri-Nets are used as an elegant modeling formalism to model concurrency, synchronization, and choice in many applications including asynchronous circuit modeling[20]. A Petri-Net is a triple N = (P, T, F) where P is the finite set of places, T the finite set of transitions, and $F \subseteq (P \times T) \cup (T \times P)$ is the flow relation. The pre-set of an element $x \subseteq (P \cup T)$ is defined as $\bullet x = \{ y \in P \cup T \,|\, (y,x) \in F \}$ and its post-set is defined as $x\bullet = \{ y \in P \cup T \,|\, (x,y) \in F \}$.

A marking is a token assignment for the places and represents the state of the system. Formally, a marking is a mapping M: $P \rightarrow \{0, 1, 2,...\}$ where the number of tokens in place p under marking M is denoted by M(p). If M(p) > 0, then the place p has tokens within. All the places in our simulation system are 1-bounded which means that they maximally can contain only one token. A transition t is enabled at marking M if $M(p) \geq 1, \forall p \in \bullet t$. An enabled transition may fire eventually in a feasible asynchronous circuit specification. The firing of *t* removes one token from each place in its pre-set and inserts one token to each place in its post-set. Timed Petri-Net is a Petri-Net in which some transitions or places can be annotated with delays. In traditional deterministic performance analysis method, places were annotated with deterministic delays as process variation was not an important problem in designs. But as variations in process parameters became considerable in new designs, the designers were motivated to analyze their designs considering process variation.

## 2.3 Process Variation

One of the most important features of deep sub-micron scale CMOS technology is the increasing magnitude of variability of the key parameters which affects performance of integrated circuits. For the circuit designer's sake, the distinction is between chip-to-chip (also called inter-chip) and within-chip (also called intra-chip) variability. Intra-die variations are variations in device features that exist within a single chip, meaning that a device feature varies between different locations on the same die. Inter-chip variations are variations that occur from one die to the next, meaning that the same device on a chip has different features among different die of one wafer, from wafer to wafer, and from wafer lot to wafer lot.

Historically, intra-chip variation of parameters could be ignored safely in digital circuit design. But, the patterns of variability are changing by shrinking the scale of the circuits. For example, for 130nm CMOS technologies, the percentage of the total variation of the effective MOS channel length that can be attributed to intra-chip variation can be up to 35% [21].

Process variations can be classified into systematic and random variations. As systematic variations are deterministic in nature and are caused by the structure of a particular gate and its topological environment, the process engineer can predict the value of a variable deterministically (and thus analyze, correct, and compensate for it). Now suppose that the process engineer cannot correct the variant process parameter non-uniformity. To a circuit designer, this source of variability will appear statistical. As the placement of each die on the wafer is unspecified and cannot be utilized, the circuit designer cannot deterministically describe the values of parameter affecting each die and thus only a statistical description is possible. Therefore, considering process variation in timing analysis of a circuit requires a statistical approach to the analysis.

As a result, it is necessary to apply a performance model which supports the required statistical approach in the modeling of asynchronous circuits.

# 3. A Performance Model Using Variatn-Timed Petri-Net(VTPN) Model

Asynchronous circuits after decomposition can be considered as a set of fine grained concurrent modules each one is responsible for producing a single variable. We model the network of templates with a novel Variant-Timed Petri-Net model. The main advantage of this model is that it can be used for simulation of circuits in addition to static performance analysis. In this model, the detailed structures of the original circuit including the handshaking channels are preserved. We have developed a class of models that fully supports full buffer templates [21].

The simplest form of a full buffer is a simple buffer that only reads a value from its input and writes it to its output. This behavior can be modeled simply as shown in Figure 1. Transition tW is analogous to the write statement while place *pWa* emulates the write acknowledge. Similarly pF can be seen as the dual for read statement while *tRa* is the corresponding acknowledge. This model is very similar to FCBN model presented in [23] and the only difference is that

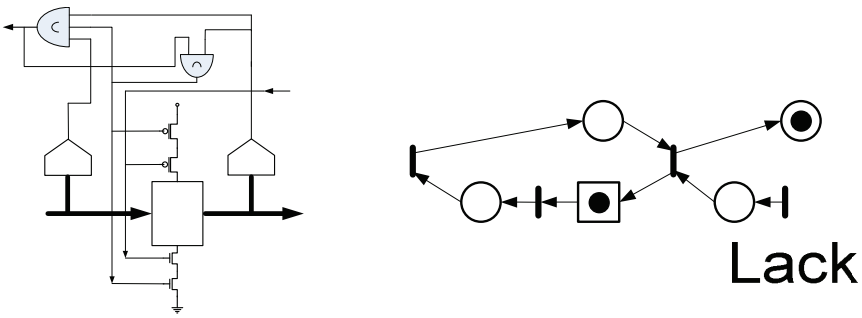


**Figure 1 :** A simple full buffer template and its corresponding Timed Petri-Nets model

we added *tRa*. The reason for this is that the used definition of the hierarchical Petri-Nets has a restriction on the input and output ports; all outputs must be transitions and all inputs must be places. This convention ensures that unwanted choices or merge constructs cannot be formed when connecting Petri-Net modules to each other.

We have considered delays on the places, therefore forward delay and backward delay can be put on *pF* and *pB*. In this model, the $d(f)$ represents the forward latency of a channel, while the corresponding $d(p)$ represents the backward latency of channel. We define these values as follows:

- The forward latency represents the delay through an empty channel (and associated cell).
- The backward latency represents the time it takes the handshaking circuitry within the neighboring cells to reset, enabling a second token to flow.

The values of these parameters are the normalized delays which are back annotated from the associated cell layout. In other words, the proposed template model exploits the normalized delays model for the sake of accurate performance estimation. Considering process variation in timing modeling necessitate applying probability distribution functions as delay models which will be more explained in further sections.

Performance of any computation modeled with a VTPN is dictated by the cycle time of the VTPN and thus the largest cycle metric. A cycle *C* in a VTPN is a sequence of places $P_1, P_2, \ldots, P_1$ connected by arcs and transitions whose the first and the last place are the same. The statistical cycle metric, (*SCM(c)*), is the statistical sum of the delays of all associated places along the cycle *C*, $d(c)$, divided by the number of tokens that reside in the cycle, $m_0(c)$, defined as:

$$SCM(c) = d(c)/m_0(c) \qquad (1)$$

The cycle time of a VTPN is defined as the largest cycle metric among all cycles in the Variant-TPN which must be computed statistically, i.e. $max(SCM(c))\forall c \in$, where *C* is the set of all cycles in the Variant-TPN.

In deterministic circuits, the cycle time is captured by the maximum cycle metric of the corresponding TPN model which can be resolved using traditional *Maximum Mean-Cycle* (MMC) algorithms (the *throughput* of the circuit is the reciprocal of this value). Karp's algorithm [24] is one of the fastest and common algorithms for this problem. Dasdan and Gupta introduced a more efficient algorithm for MMC analysis[25]. One of the advantages of MMC method is its efficiency. With the result of cycle time, we can find the bottleneck of a circuit. However, the solution is different in VTPN as the delays are modeled statistically due to considering process variation. The proposed method considers the differences in the next section.

## 4. The Proposed Statistical Performance Analysis Method

In this section, we define the modeling assumptions and our proposed SSTA method and also the required operations used in the method.

As mentioned, the delays must be modeled statistically. We model the delays as random variables with normal distributions. So each place in VTPN has a mean delay value, $\mu$, and a standard deviation, $\sigma$, which models its variation. In the proposed method, we find all the cycles of the marked graph first. After that, we calculate the statistical cycle metric of each cycle, and then, we find the maximum SCM as the critical metric and the relevant cycle will be supposed as the critical cycle. In all steps of the method, particular operations must be applied based on the model used for modeling the delays in the graph.

As each delay is modeled as a random variable with normal distribution, it is noteworthy to explain about the statistical operations first. The three operations used in our method are SUM, DIV and MAX.

### 4.1 SUM operation

The sum of two random variables with normal distribution results in a random variable with normal distribution. The two parameters of the result, i.e. the mean, $\mu$, and the standard deviation, $\sigma$, are calculated as follows:

$$C = SUM(A,B) = A + B$$
$$\sigma_C^2 = \sigma_A^2 + \sigma_B^2 + 2cov(A,B) \qquad (2)$$

### 4.2 DIV operation

In calculating the SCM of a cycle, the sum of delay values of the cycle will be divided by the number of the tokens in the cycle. As the sum of the delays modeled by normal random variable is still a normal random variable, the parameters of the division are calculated as follows:

$$\mu_{\frac{A}{n}} = \frac{\mu_A}{n}$$
$$\sigma^2_{\frac{A}{n}} = \frac{\sigma^2_A}{n^2} \qquad (3)$$

### 4.3 MAX operation

The maximum of two normal random variables does not necessarily results in a normal random variable. The MAX of two random variables with normal distribution A and B can be

approximated to another normal random variable C using the relationship proposed in [27], that is as follows:

$$
\begin{aligned}
&C = MAX(A, B)\\
&\alpha = \frac{\mu_A + \mu_B}{\beta}\\
&\beta^2 = \sigma_A^2 + \sigma_B^2 - 2\sigma_A\sigma_B\rho\\
&v_1 = (\mu_A^2+\sigma_A^2)\phi(\alpha) + (\mu_B^2+\sigma_B^2)\phi(-\alpha) + (\mu_A + \mu_B)\beta\varphi(\alpha)\\
&v_2 = \mu_A\phi(\alpha) + \mu_B\phi(-\alpha) + \beta\varphi(\alpha)\\
&\mu_C = v_2\\
&\sigma_C^2 = v_1 - \mu_C^2
\end{aligned}
\qquad (4)
$$

Here, represents the correlation coefficient between A and B , and and are the cumulative density function, CDF, and the probability density function, PDF, of a standard normal (i.e., mean 0, STD 1) distribution, respectively.

In our method, as it is supposed that process variables do not increase or decrease simultaneously and we have no assumption on this factor, the correlation coefficients are assumed to be equal to zero. Because of this and considering (5), the elements of the covariance matrices are also equal to zero.

$$cov(A, B) = \sigma_A\sigma_B\rho \qquad (5)$$

After introducing the necessary functions, it seems to be interesting to present the proposed method more perfectly:

---

Algorithm SSTA

Input: Variant-Time Petri-Net model of the Circuit

Output: Probability function of the performance metric (μ,σ)

---

1. Find all the cycles of the VT Petri-Net[26].
2. Calculate the sum of the place delays along the cycle using the SUM operation employing equation (2).
3. Divide the result of the previous step by the number of the tokens in the cycle using the DIV operation, employing equation (3).
4. Find the maximum SCM between the calculated SCMs in the previous step using the MAX operation employing equation (4).

---

As a result, we have the basic parameters of a normal random variable as a metric to evaluate the performance of the VTPN.

Here is an example. Figure 2 shows a 12-node VTPN.

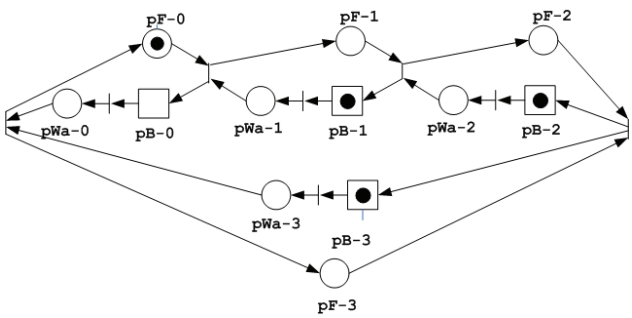


**Figure 2 :** An Example of a VTPN

As an example for the delay model, consider the mean delay value of PF-0, $\mu$, is equal to 7 and due to the process variation, the standard deviation, $\sigma$, is assumed to be equal to 0.35 unit of the delay.

Some cycles founded here are as follows:

- {PF-0, PB-0, PWa-0, PF-0}
- {PF-0, PF-1, PF-2, PB-3, PWa-3, PF-0}
- {PF-1, PB-1, PWa-1, PB-0, PWa-0, PF-0, PF-1}.

## 5. Evaluation and Experrimental Results

We have developed a Petri-Net simulation engine which supports different delay models including randomly generated delays with fixed, uniform, and normal probability distribution functions. The core simulator is based on systemC[28]. SystemC models for transition and places are developed and the tool is also able to automatically elaborate a detailed systemC model for each input Petri-Net.

The proposed SSTA method has been implemented in C++, and has been tested on a set of benchmark circuits. An asynchronous synthesis toolset (for the sake of blind review, we don't cite its name here) is employed to synthesis benchmarks. Then the developed tool automatically translates the decomposed circuits to its Variant-Timed Petri-Net equivalents. Inputs and outputs of the circuit are connected to each other in Petri-Net structure to form a closed loop system. Initially all tokens placed in input nodes. Variability of process parameters (L, Vth, and Tox) and the environmental fluctuation (Vdd) are taken into account. The 3σ values for process parameters are set at 20% of the mean. The standard deviation of Vdd is 4% of the maximum, the mean is 96% of the maximum, and the range is 84-100% of the maximum value. In the experiments, Vth, Tox and Vdd are modeled as probabilistic interval variables. The range of Vth and Tox is 80-120% of the mean. Sensitivities of parameters are from SPICE simulations for a cell library of BPTM 0.13um technology [29].

The proposed method has been run on SUN Ultra Sparc 10 workstation with 1 gigabyte of memory. The sizes of VTPNs range from 6 nodes to 56 nodes. The runtime for our benchmark ranges from 5s to 400s, depending on circuit sizes and the structure of VTPN model of circuit.

To verify the results of our statistical method, we used Monte Carlo (MC) simulation for comparison. To balance the accuracy, we chose to run 1,000 iterations for the Monte Carlo simulation. A comparison of these results with those from statistical approach is shown in Table I. For each test case, the mean and standard deviation (SD) values for both methods are listed. The results of SSTA can be seen to be close to the MC results: the average error is %2.8 for the mean value of the delays.

In Figure 3 and Figure 4, for the largest test case, named *I,* we show the plots of the PDF and CDF of the circuit delay for both SSTA and MC methods. It is observable that the curves almost match each other and the main source of the difference between them is the correlation coefficients of the process parameters which are assumed to be zero. It is notable that this can be considered as a future work on this subject in asynchronous circuits.

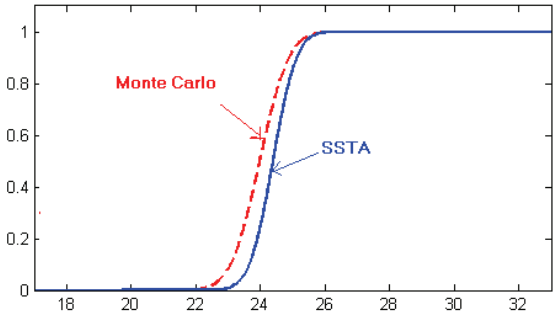


**Figure 3:** Comparison of SSTA and MC methods: CDF curves

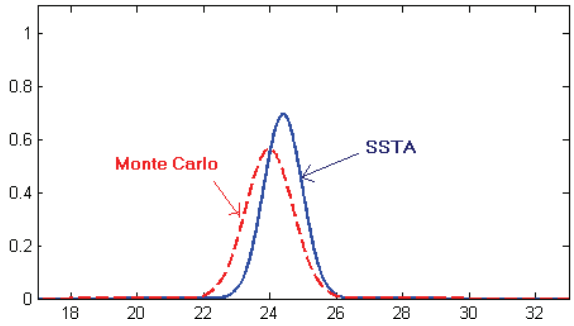


**Figure 4 :** comparison of SSTA and MC methods: PDF curves

## 6. Modeling Issues and Discussion

This section addresses two limitations of the model used in this paper: the use of a normal delay distribution, and the restriction to decision-free systems.

In order to make a system amenable to analytical methods, the delay distributions for components are currently restricted to be normal. This restriction, however, should not have a significant impact on results in practice. Only the mean and the variance of the delay have a significant impact on the result.

Another apparent drawback of our approach is the restriction to decision-free systems, as systems with complicated choice decisions and conflicts cannot be properly modeled as marked graphs. With proper modeling, however, systems with simple choices can be analyzed under our framework. For example, a choice between a slow mode and a fast mode of operation of a certain component can be modeled using a discrete delay distribution. In some cases, more complicated choices can be handled hierarchically. Extensions to handle arbitrary systems with choice in a more integrated way will be included in future works.

## 7. Conclusion and Future work

Even though asynchronous circuits are highly tolerant to process variation, it seemed to be necessary to present a method to analyze the performance of asynchronous circuits considering the variation in process parameters. This view has facilitated us to derive an efficient method to analyze system performance, and to define meaningful performance metrics for optimization. In this paper, we present a method to analyze the performance of template-based asynchronous circuits statistically. Asynchronous circuit has been modeled using Variant-Timed Petri-Net. Based on this model, the probability density function of the delay of global critical cycle is calculated. We also present a simulation tool of Variant-Timed Petri-Net and the results of the experiments are compared with Mont Carlo simulation results. We demonstrated our method via a tool. Results show that it is possible to consider the process variation and analyze the performance while there is ignorable error between the results of the proposed method and the general Monte Carlo simulation method.

We see many avenues for further investigation. Research goals in the immediate future include extensions to analyze asynchronous systems with choice, the development of performance/power optimization algorithms for asynchronous systems driven by our analysis technique, and the application of our method to a broader class of concurrent systems, such as GALS and embedded systems. In addition, considering the correlation of process parameters can be a good topic for the researchers in the asynchronous circuit designs similarly to the synchronous ones.

## 8. References

[1] C.K. Tang, C.Y. Lin, Y.C. Lu, "An Asynchronous Circuit Design with Fast Forwarding Technique at Advanced Technology Node", Proceedings of ISQED'08, IEEE Computer Society, 2008.

[2] Peter A. Beerel, "Asynchronous Circuits: An Increasingly Practical Design Solution", Proceedings of ISQED'02, IEEE Computer Society, 2002.

[3] Alain J. Martin, et al, "The Lutonium: A Sub-Nanojoule Asynchronous 8051 Microcontroller. ASYNC 2003.

[4] K.Y.Yun,P.A.Beerel,V.Vakilotojar, A.E.Dooply, J.Arceo, "A low-control-overhead asynchronous differential equation solver", In Proceedings of ASYNC, 1997.

**Table I:** Comparison results of the proposed method and Monte-Carlo simulation method

| The circuit | # of the Nodes | # of the Cycles | Worst Case | The proposed SSTA | | Monte-Carlo | | Error (SSTA-MC)/MC% | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | *Mu* ($\mu$) | *Sigma* ($\sigma$) | *Mu* ($\mu$) | *Sigma* ($\sigma$) | *Mu* ($\mu$) | *Sigma* ($\sigma$) |
| A | 6 | 17 | 14.950 | 15.448 | 0.3804 | 14.9887 | 0.5359 | 3.12 | -29.02 |
| B | 10 | 51 | 19.550 | 17.445 | 0.2478 | 16.9978 | 0.3947 | 2.63 | -37.23 |
| C | 16 | 1389 | 25.300 | 23.210 | 0.298 | 21.997 | 0.620 | 5.5 | -51 |
| D | 26 | 1864 | 33.350 | 29.982 | 0.431 | 29.300 | 0.662 | 2.3 | -34 |
| E | 35 | 7369 | 22.616 | 19.805 | 0.170 | 19.670 | 0.246 | 0.6 | -30.4 |
| F | 20 | 276 | 19.837 | 17.671 | 0.150 | 17.244 | 0.264 | 2.4 | -43 |
| G | 22 | 5605 | 23.575 | 21.374 | 0.184 | 20.522 | 0.436 | 4.1 | -51.3 |
| I | 56 | 812 | 27.600 | 24.392 | 0.575 | 23.980 | 0.708 | 1.7 | -18 |

[5] O. Garnica, J. Lanchares, R. Hermida, “Fine-grain asynchronous circuits for low-power high performance DSP implementations”, SiPS 2000.

[6] Peggy B. McGee , Steven M. Nowick , E. G. Coffman, “Efficient performance analysis of asynchronous systems based on periodicity”, Proceedings of the 3rd IEEE/ACM/IFIP international conference on Hardware/software codesign and system synthesis, 2005.

[7] C. Visweswariah et al, “First-order incremental block-based statistical timing analysis”, Proc. of DAC, 2004.

[8] Wei-Shen Wang , Vladik Kreinovich , Michael Orshansky, “Statistical timing based on incomplete probabilistic descriptions of parameter uncertainty”, Proc. Of DAC, 2006.

[9] P.B.Pang andM.Greenstreet, “Self-timed meshes are faster than synchronous”, in Proceedings of ASYNC,1997.

[10] A.Xie,S. Kim, and P.A.Beerel, “Bounding average time separationsof events in stochastictimed petrinets with choice”, In Proceedingsof ASYNC,1999.

[11] P.B.Pang andM.Greenstreet, ”Self-timed meshesare faster than synchronous”, in Proceedingsof ASYNC,1997.

[12] E. Yahya, M. Renaudin, “Performance Modeling and Analysis of Asynchronous Linear-Pipeline with Time Variable Delays”, ICECS 2007.

[13] F.Commoner,A.Holt,S.Even,and A.Pnueli, “Marked directed graphs”, Journal of Computerand System Sciences,5:511–523,1971.

[14] C G. Wong and Alain J. Martin, "High-Level Synthesis of Asynchronous Systems by Data Driven Decomposition", Proc. Of 40th DAC, Anneheim, CA, USA, June 2003.

[15] A.V. Dinh Duc, J.B.Rigaud, A.Rezzag, A.Sirianni, J.Fragoso, L.Fesquet,andM.Renaudin, “TASTCADTools:Tutorial,” Proc. Of Advanced Research in Asynchronous Circuits and Systems (ASYNC’02), 2002.

[16] Behnam Ghavami, Hossein Pedram, “Design of Dual Threshold Voltages Asynchronous Circuits”, ISLPED 2008: 185-188.

[17] Piyush Prakash, Alain J. Martin, “Slack Matching Quasi Delay-Insensitive Circuits”, ASYNC 2006: 195-204.

[18] Aiguo Xie, Sangyun Kim, and Peter A. Beerel, “Bounding average time separations of events in stochastic timed Petri nets with choice”, In ASYNC, pages 94–107, 1999.

[19] Steven M. Burns, Alian J Martin, “Performance Analysis and Optimization of Asynchronous circuits”, Advanced Research in VLSI conference, Santa Cruz, CA, March 1991.

[20] Sangyun Kim, “Pipeline Optimization for Asynchronous circuits”, PHD Thesis, University of Southern California, August 2003.

[21] Michael Orshansky, Sani R. Nassif, Duane Boning, “Design for Manufacturability and Statistical Design, A Constructive Approach”, Springer Press, 2008 pp. 11-15.

[22] Andrew Matthew Lines, “Pipelined asynchronous circuits”, Master's thesis, California Institute of Technology, Computer Science Department, 1995 CS-TR-95-21.

[23] Peter A Beerel, Nam-Hoon Kim , Andrew Lines, Mike Davies, “Slack Matching Asynchronous Designs”, Proceedings of the 12th IEEE International Symposium on Asynchronous Circuits and Systems, Washington, DC, USA,2006.

[24] R. M. Karp, “A characterization of the minimum cycle mean in a diagraph”. Discrete mathematics, 23:309–311, 1978

[25] A. Dasdan, R.K. Gupta, “Faster maximum and minimum mean cycle algorithms for system performance analysis”, IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems, Vol. 17, No. 10, 1998, pp. 889-899.

[26] Hongbo Liu, Jiaxin Wang, “A new way to enumerate cycles in graph”, Proceedings of the Advanced International Conference on Telecommunications and International Conference on Internet and Web Applications and Services (AICT/ICIW), 2006.

[27] C. E. Clark, 1961. “The Greatest of a Finite Set of Random Variable”, vol. 9. Operations Research,85–91.

[28] Braham Lane, “SystemC Language Reference Manual”, Copyright © 2003 Open SystemC Initiative ,San Jose, CA.

[29] PTM: http://www.eas.asu.edu/~ptm.