國立交通大學

資訊工程系

碩 士 論 文

運用具有變化關聯性之FR向量以分析訊號變化活動

Switching activity analysis with FR-vector considering transition correlation

研 究 生：夏文忠

指導教授：陳昌居　博士

中 華 民 國 九 十 三年 六 月

Switching activity analysis with FR-vector considering transition correlation

研 究 生：夏文忠　　　　Student：Wen-Chung Shia

指導教授：陳昌居　　　　Advisor：Chang-Jiu Chen

國 立 交 通 大 學

資 訊 工 程 學 系

碩 士 論 文

A Thesis

Submitted to Department of Computer Science and Information Engineering

College of Electrical Engineering and Computer Science

National Chiao Tung University

in

partial Fulfillment of the Requirements

for the Degree of Master

in

Computer Science and Information Engineering

June 2004

Hsinchu, Taiwan, Republic of China

中華民國九十三年六月

授權書

(博碩士論文)

本授權書所授權之論文為本人在＿＿國立交通＿＿大學(學院)＿資訊工程＿系所
＿＿＿＿＿＿＿組＿九十三＿學年度第＿二＿學期取得＿碩士＿士學位之論文。

論文名稱：＿＿運用具有變化關聯性之 FR 向量以分析訊號變化活動＿＿

1.☐同意　　☐不同意

本人具有著作財產權之論文全文資料，授予行政院國家科學委員會科學技術資料中心、國家圖書館及本人畢業學校圖書館，得不限地域、時間與次數以微縮、光碟或數位化等各種方式重製後散布發行或上載網路。

本論文為本人向經濟部智慧財產局申請專利的附件之一，請將全文資料延後兩年後再公開。(請註明文號:　　　　　　　　　　　　)

2.☐同意　　☐不同意

本人具有著作財產權之論文全文資料，授予教育部指定送繳之圖書館及本人畢業學校圖書館，為學術研究之目的以各種方法重製，或為上述目的再授權他人以各種方法重製，不限地域與時間，惟每人以一份為限。

上述授權內容均無須訂立讓與及授權契約書。依本授權之發行權為非專屬性發行權利。依本授權所為之收錄、重製、發行及學術研發利用均為無償。上述同意與不同意之欄位若未鉤選，本人同意視同授權。

指導教授姓名：　　　陳昌居

研究生簽名：　　　　　　　　　　　　　　　學號：9117567
（親筆正楷）　　　　　　　　　　　　　　　（務必填寫）

日期：民國　九十三年　六月　三十日

---

1. 本授權書請以黑筆撰寫並影印裝訂於書名頁之次頁。
2. 授權第一項者，所繳的論文本將由註冊組彙總寄交國科會科學技術資料中心。
3. 本授權書已於民國 85 年 4 月 10 日送請內政部著作權委員會（現為經濟部智慧財產局）修正定稿。
4. 本案依據教育部國家圖書館 85.4.19 台(85)圖編字第 712 號函辦理。

運用具有變化關聯性之FR向量以分析訊號變化活動

研究生：夏文忠　　　　　　　　　　指導教授：陳昌居

國立交通大學 資訊工程學系

摘要

為達到低耗能電路設計之目的，估算在閘層時的能量消耗是主要的關鍵之一。在互補金屬氧化物導體之組合電路中，估算能量消耗可以藉由量測訊號變換的次數來達成。在本論文中，我們提出結合 FR-Vector 和“布林逼近法”的新方法。這個方法使用機率的觀念，以時間分割加入時間關聯性的考量，甚至更進一步用泰勒展開式的逼近法以處理電路間的空間關聯性，然後用少量的資料就可以模擬出危障、延遲以及重聚合電路的影響。我們將我們的方法、FR-Vector 及 Cadence NC-VHD 對十八個 MCNC 的標竿電路做模擬，並比較所得的結果。得到的結果顯示我們的方法遠比原 FR-Vector 減低了每一個邏輯閘上的平均加權誤差值，可從 6.09 個百分點降到 2.77 個百分點。同時，更大幅降低了誤差的尖峰值－由 23.74 個百分點降到 13.24 個百分點。最後，我們的方法不僅改進了原使用 FR-Vector 時會產生的誤差，執行時間也幾乎與原 FR-Vector 相近。比起使用 NC-VHDL 模擬電路所花費的時間，差距更可到達三十多倍。實際上，相對於對我們的方法而言，因為電路大小對 NC-VHDL 的影響更為明顯，所以在實際的電路中，增加的速度更將不止於三十倍(實際電路通常會比我們所測的標竿電路為大)。

Switching activity analysis with FR-vector considering transition correlation

Student : Wen-Chung Shia Advisor : Dr. Chang-Jiu Chen
Department of Computer Science and Information Engineering

National Chiao-Tung University

Abstract

In the research of the low-power circuit design, one of the key issues is to estimate the power dissipation in a gate-level implementation. In CMOS combinational circuit, the switching activity measurement is an approach to the power dissipation estimation. In this thesis, we propose a new method to combine the FR-Vector model and the concept of Boolean approximation method. We use probabilistic method that time to divide a clock into several time units, and exploit Taylor expansion, then glitches and reconvergent circuit effect can be handled easily. For performance evaluation, we make comparisons among the proposed method, the FR-Vector, and Cadence NC-VHDL in 18 MCNC benchmark circuits. The comparison results show that our method decreasing the weight error percentage (the error percentage in each gate) very much－ from 6.09% to 2.77%. Meanwhile, it by far decreases the peak value of error percentage － from 23.74% to 13.24%. Finally, our method not only improves the error caused by the FR-Vector, but also the time it spends is closed to the FR-Vector. Comparing to the simulation of NC-VHDL in the time spending, out method could speed up to 33 times. Moreover, in real circuits, the speed-up would be more than 33 times. It is because the variable “gate count” affects time complexity in NC-VHDL much than the time complexity in our method, and in real circuits, there is usually a larger gate count than our benchmark.

Acknowledgment

完成這篇論文要感謝的人很多，首先感謝指導老師陳昌居老師在提供穩定的環境與資源讓我能有所成長，也感謝老師在這兩年給予我的協助。特別感謝黃年時學長，在完成這篇論文的期間一起討論以及引導，也感激同實驗室的夥伴們曾經提供的協助以及生活上的點點滴滴。最重要的是感謝我的家人與親友給我的支持，讓我能夠穩定的求學以及研究，尤其是父親的苦心栽培。

Contents

摘要 I

ABSTRACT II

ACKNOWLEDGMENT III

CHAPTER 1 INTRODUCTION 1

CHAPTER 2 RELATED WORK 5

2.1 BASIC DEFINITION IN SWITCHING ACTIVITIES ESTIMATION 5

2.2 PREVIOUS RESEARCH OF SWITCHING ACTIVITIES ESTIMATION 6

2.3 FR-VECTOR 9

2.3.1 Signal Modeling with FR-vector *9*

2.3.2 FR-Vector Logic Composition *10*

2.3.3 FR-Matrix *11*

2.3.4 Non-Zero Delay FR-matrix *14*

2.4 BOOLEAN APPROXIMATION METHOD (BAM) 15

2.4.1 Basic Assumptions *15*

2.4.2 Signal probability *16*

2.4.3 Algorithm of BAM *21*

2.4.4 Switching Activities *21*

2.4.5 Example *22*

2.5 Real-Delay Boolean Function (RDBF) **24**

CHAPTER 3 FR-VECTOR WITH BAM **27**

3.1 Review and Integration **27**

3.2 Basic Definition and assumption **28**

3.3 Probabilities Propagation **31**

3.3.1 Propagation of the Cofactor Transition Relation *31*

3.3.2 Propagation of the FR-Matrix *34*

3.4 An Example of FR-Vector with BAM **39**

3.5 Reconvergent Circuit in Limit Depth **41**

3.6 Non-Zero Delay Model **42**

3.7 Implementation **47**

CHAPTER 4 EXPERIMENTAL RESULTS **51**

CHAPTER 5 CONCLUSIONS AND FUTURE WORKS **57**

REFERENCE: **59**

List of Figure

Figure 1-1: A simple reconvergent circuit..4

Figure 2-1: 2-input AND gate with n primary inputs..17

Figure 2-2: General case in the circuit..20

Figure 2-3: A test circuit. 22

Figure 2-4: A logic circuit with non-zero delay gates...25

Figure 3-1: An example of supply set of the reconvergent circuit. 30

Figure 3-2: A reconvergent circuit and the FR-Matrix of its input node x. 34

Figure 3-3: A reconvergent circuit with non-unit delay gate 45

Figure 3-4: The pseudo-code of this paper 47

List of Table

Table 1-1: The switching state of a 2-bit counter..4

Table 2-1: the logic composition tables for basic logic function in FR-vector ... 11

Table 2-2: (a) Signal sampling. (b) Its FR-matrix..12

Table 2-3: An example of FRM composition: Input FRM 1, (b) Input FRM 2, and (c) Output FRM. 13

Table 3-1: The composition table for cofactor transition relation. 33

Table 3-2: The transition probabilities of primary inputs of the circuit in example 3.4 40

Table 4-1: Switching activity comparisons between NC-VHDL logic simulator, FRM model and FR-Vector with BAM based on MCNC benchmark circuits with no glitch input. 52

Table 4-2: The CPU Time for NC-VHDL simulation..54

Table 4-3: The CPU Time for FR-Vector simulation..54

Table 4-4: The CPU Time for FR-Vector with BAM simulation........................54

Table 4-5: The comparison with FR-Vector and with NC-VHDL. 55

Table 4-6: Comparison among different technique. 56

Chapter 1 Introduction

In circuit design, the decreasing of the feature size leads to the increasing of chip density. While the operating frequency growing rapidly and the feature size getting smaller, these factors make the power consumption to be taken into account seriously. Especially, with the rapid, strong demand development in market sectors such as wireless application, laptop and portable medical devices, it makes the power consumption to be one of the most critical topics in digital system design [1]. Time-to-Market requirement could be achieved by low power design techniques and power estimation methodology. With the aid of power estimation function of CAD tools, it can help designers to meet the power specification earlier in designing phase, and further reduce redesign cost at the same time.

Power consumption in a CMOS circuit can be classified into following three categories:

1. Static leakage power;
2. Short-circuit power;
3. Dynamic power.

In a well-designed circuit, if we don't consider the special case such as the power lost when portable device is in a dormant state, the total power dissipation cost by the dynamic power consumption of the nodes, which arises due to the charging and discharging of the parasitic capacitance during switching, will by far exceed the first two factors. In [2], the average power consumption at a gate is given by $P(x) = 0.5V_{dd}^2 f_{clk} C_{load} sw(x)$, where V_{dd} is the supply voltage, f_{clk} is the clock frequency, C_{load} is load capacitance, and $sw(x)$ is the switching activity of the output node $x[2]$, so we could know $sw(x)$ is a very important factor in power estimation.

In power estimation, both speed and accuracy are the most important factors, but obviously, they conflict with each other. The easiest and most direct method of how to estimating power consumption is to simulate the operation of the whole circuit. However, though this method has the most accurate outcome, it also takes too much time. Today, the techniques of power estimation could be divided into two categories: dynamic (statistic) and static (probabilistic) [1] [2]. Dynamic techniques explicitly simulate the circuit under a "typical" input stream. These techniques provide a high level of accuracy but it also takes a very high run time which is because the required number of simulation vectors is usually large. Static techniques would calculate the input patterns first, and use a probability to represent a signal state, and then it propagates or calculates these probabilities from the primary inputs to the output of the whole circuit. In final, it uses these probabilities to get the number of switching. These techniques could provide fast measurement without losing accuracy too much.

Gate delay is an important factor about the accuracy in estimations, but it's very hard to consider the delays in a circuit. For this reason, many papers tried to ignore the impact caused by gate delays [1] [3] [6][9][10][15].

- **Zero delay models**: all the gate delays of the circuit are taken as zero.
- **Non-zero delay model**: each gate delay of the circuit is a positive number or zero. We suppose that non-zero delay model with inertial mode in this paper.

The most important drawback to activities analysis when without considering gate delay is glitch ignoring. However, in non-zero delay model, glitches could happen from the difference of arrival time between two (or more) input signals, and these glitches would cause additional 20% power estimation in average. In some special cases, like specific adder, the additional consuming power could be up to 70% [3].

In additional to glitches, another important factor to power estimation techniques is correlations between signals. There might be two kinds of correlations between signals: **temporal dependence** and **spatial dependence.**

- **Temporal dependency:** Signals may be temporally dependent; in other words, the next value of a signal may depend on its current or previous values.
- **Spatial dependency:** Spatial dependency comes from two aspects:
  - ✓ **Structure dependency is** due to reconvergent fanout in the circuit;
  - ✓ **Input dependency** is that spatial and/or temporal correlations among the input signals which result from the actual input sequence applied to the target circuit.

The input of a counter is an example of dependencies between inputs; **Table 1-1** shows the transition of a two-bit counter. In input C_2, the next state is depended on the current state, this is a kind of temporal dependency, and in input C_1, its next state is not just depended on the current state, it would also depend on C_2, this is a kind of spatial dependency.

Figure 1-1shows another kind of spatial dependency. In this circuit, the logic value of node *m* and *n* are not dependent due to they are diverge node from node *b*. *m* and *n* convergent at node *z* for a while. For this reason, we should process the correlation between node *m* and node *n* to analyze the switching activities in node z precisely.

The aim of this thesis is to propose a modified FR-vector [5]. FR-vector is a model for power consumption which uses probability technique. It could operate under multiple input switching, non-zero gate delay and even glitches happen in a combinational circuit. We combine the advantages of FR-vector

Current state		Next state	
C_1	C_2	C_1	C_2
1	1	0	0
0	0	0	1
0	1	1	0
1	0	1	1

Table 1-1: The switching state of a 2-bit counter.

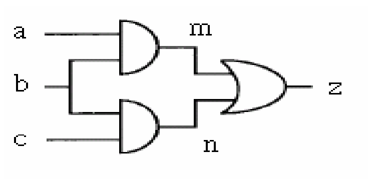


Figure 1-1: A simple reconvergent circuit.

and the Boolean Approximation Method (BAM) [6], which is a data structure to process signal correlation ,then propose a new modified FR-vector — FR-vector with BAM.

The rest of this thesis is organized as follows. Chapter 2 introduces some estimation methods and reviews FR-vector and BAM. Chapter 3 describes new modified FR-model. Chapter 4 shows some experimental results. Chapter 5 gives our conclusions and the directions for future development.

Chapter 2 Related Work

Low power circuit is a more and more important research today. As mentioned in the first chapter, the switching activity is a very important factor in power estimation, and we will pay attention to it in the thesis. In this chapter, a picture about some researches of switching activity analysis will be given; we will also introduce some techniques and definitions needed later. Section 2.1 gives some basic definitions used in switching activity analysis. Then we will give short review of representative researches in section 2.2 . Section 2.3 introduces FR-vector. Section 2.4 describes Boolean approximation Method (BAM). Finally, the Real Delay Boolean Function (RDBF) is introduced in section 2.5 .

2.1 Basic Definition in Switching Activities Estimation

Before discussing the techniques of switching activity estimation, we will introduce some terms or some basic definitions which would be frequently used in this thesis. It is the most basic term in switching activity analysis and all calculations in these techniques is based on the physical signal of circuits.

- **Physical signal** is an electrical signal that appears in an input line.

Then, we could get signal probability immediately:

Definition 2.1 (Signal probability): *For any signal line x of the circuit, the probability of x is logic high is denoted as $P(x)$, and called* ***signal probability.***

Though signal probabilities could show the states of circuits, we need to get the information about how signal changes in additional. Thus we use switching probabilities to replace signal probabilities.

- **Signal switching (**an alternate name is **switching activity)** is the sequence of continuous signal states: "ab", where $a,b \in (0,1)$, a and b represent the current and the latter state of continuous states respectively.

Definition 2.2 (Switching probability): *For any signal line x of the circuit, the switching probability of x is defined as* $P[(x_t = j \cap x_{t-1} = i)]$, and is denoted by $P(x_{i \to j})$, $i, j \in (0,1)$.

As we have mentioned in chapter 1, glitches will cause additional power consumption. Actually, the technique that doesn't consider glitches is unrealistic in usual.

Definition 2.3 (Glitch): *The **glitch** is (an) unexpected transition(s) which occur(s) when the signal switches at the time that is not required. A **static hazard** exists if a signal remains constant after twice (or more) switches during a clock cycle, so there are **static 0 hazards,** and **static 1 hazard**. In opposite side, a **dynamic hazard** exists if a signal changes is not the same at the start and the end of a clock cycle：there are **dynamic 1 hazard** (start with logic low and end with logic high) and **dynamic 0 hazard** (start with logic high and end with logic low).*

2.2 Previous Research of Switching Activities Estimation

The simplest and the most intuitive method to estimate the switching activity is logic simulation [7] [8]. Logic simulation is the most accurate method but it is also the slowest one. However, most of the actual combinational circuits are so complicated that it is not practical to simulate them immediately.

It has been mentioned by many papers about how to use probability to analyze switching activity. However, their accuracy is not all the same due to the detail about how they implemented. We could classify the key factors which affect the accuracy into if they process the correlation inside the circuit and if they process the glitch power. The transitions between circuits are often correlated to each others, for example, the reconvergent circuits will complex the switching activity. Further more, the input patterns might hide some correlation inside itself. In another aspect, if we consider for the gate delay in circuits, there are always some unanticipated transition happens, and cause to additional power consumed.

Farid N. Najm [2] has introduced the techniques about switching activity analysis in detail. He notes two statistic methods, and categorizes probabilistic techniques into following five classes: A) Signal probability, B) Probabilistic Simulation CREST, C) Transition Density DENSIM, D) using Binary Decision Diagram BDD, and E) Correlation coefficient. In final, he discussed how to extend from combinational circuits to sequential circuits.

We could say that correlation coefficient is the most accurate probabilistic method today. In early years, [4] introduces this concept to take care of the signal correlation between circuit nodes. It uses the correlation coefficient to represent the correlation between the probability two signal probabilities multiplied immediately and the probabilities of two signal are logic high in real, that is: $P(AB) = P(A)P(B)C_{A,B}$, where $P(AB)$ is the probability of $(A \cap B)$ is true (signal A and signal B both are logic high), and $C_{A,B}$ is correlation coefficient between node A and node B. Hence, we could get $C_{A,B} = \frac{P(AB)}{P(A)P(B)}$. This technique will passes this coefficient through the whole circuit to calculate signal probabilities under correlated effect. Someone extended this concept to the transition of signals[1]: $TC_{ij,kl}^{xy} = \frac{p(x_{i \to k} \cap y_{j \to l})}{p(x_{i \to k})p(y_{j \to l})}$, where $p(x_{i \to k} \cap y_{j \to l})$ is the probability of signal A changed from i to k and signal B changed from j to l at the same time, and

$TC^{xy}_{ij,kl}$ is transition correlation coefficient between A changed from i to k and signal B changed from j to l at the same time, this technique also uses OBDD to transmit transition correlation coefficient and switching probability through circuits.

To pass correlation coefficient through the whole circuit would cost a lot of time, so “Limited depth” and “Limited nodes” was proposed in [9] [10]. They told that if the reconvergent circuit in a circuit exceeds a constant number of gate levels, the signals between circuits could be taken as independent without considering correlation. Because the larger the gate level is, the weaker the correlation between circuits. We should not sacrifice too much time to a negligible correlation.

Another disadvantage of correlation coefficient is when the circuit grows lager and lager; it is unrealistic to build OBDD of the whole circuit. This is because it would consume too much memory. There is another technique called Boolean approximation method (BAM) [6] [11], it uses BAM which is a data structure to process correlation between circuit. BAM could be propagated through circuit by logic characteristic without using OBDD. In this way, BAM could save memory to process larger circuit and make the speed up. BAM uses Taylor expansion to calculate the difference between probability two probabilities multiplied immediately and probabilities in real.

Most probabilistic methods could not be very accurate for they do not process glitch. In 1999, G-vector was proposed in [12]. Though this model doesn’t discuss switching activity, it gives a way to process about the generation, transmit and elimination of glitch. FR-vector comes from G-vector. It also uses probability to estimate switching activity, and it further discusses the situation about non-zero delay, multiple-input change. There is another similar method called “tagged probabilistic simulation” [13]. Though FR-vector could process glitches, it ignores signal correlations.

2.3 FR-Vector

Though FR-vector does not process signal correlations, it is useful to analyze glitching activity. Most papers which talks about how to process glitches would become very complex for the reason that there would be a lot of complex mathematic equations. FR-vector makes glitch processing become easier by taking the calculation of switching probabilities as the calculation of signal probabilities.

2.3.1 Signal Modeling with FR-vector

There are some basic terms used in FR-vector.

- **0/1 sampling:** is the method that samples a physical signal node N times in a clock cycle. By 0/1 sampling, we can get a sequence of length N consist of 0 and 1, which is a representation of the waveform of the physical signal.
- **0/1 sampled signal** is the 0/1 sequence representation of a physical signal.
- Each internal that we sample one time is called a **frame**. There is an assumption that the sampling rate is so fast that there is at most one or one glitch in a frame.
- A frame value of 0/1 sample signal is named **state**.

And in FR-vector, **signal switch** is:

- Signal switching (an alternate name is Switching activity) is the sequence "01" or "10" part of a 0/1 sampled signal.

FR-vector keeps the information of glitches before and after the gate in circuits, and simulates the glitch propagation, elimination, and generation very well. It models the switching activities as falling and rising, and marked as "F" and "R" for the signal rising and falling.

Definition 2.4 (FR-vector): *An FR-vector is a set of* $\{x \mid x \in [0|1][0|1|F|R]^{N-1}\}$*; 0 is the non-glitch 0 state, 1 is the non-glitch 1 state,* F *is the falling signal state,* R *is the rising signal state, and N is the number of frames in a input vector.*

It assumes the first frame is the earliest frame, and it is a non-zero delay model in inertial mode. By these assumptions, there will be at most one switching in a frame (In inertial mode, if a glitch appears shorter than a gate delay, it would be elimination, and in FR-vector, a frame is the smallest gate delay of the circuit.).

The rule of obtaining a FR-vector from a 0/1 sampled signal is as follows:

Extract FR-vector v **from a 0/1 sampled signal** S

- Base case: The state of the initial frame in v is equal to the initial state of S .
- General case:
  - If the state of S_i = the state of S_{i-1} then $v_i = S_i$;
  - If the state of S_i = 1 and the state of S_{i-1} =0, then $v_i = R$;
  - If the state of S_i = 0 and the state of S_{i-1} =1, then $v_i = F$.

The rule of getting a FR-vector from a 0/1-sampled signal mentioned above means that the frame value of FR-vector is represented by the state change from the last frame to current one. Assuming that the initial frame is stable from the final frame in the preceding input vector, than the base case of the rule is obtained.

2.3.2 FR-Vector Logic Composition

FR-vector could be used in logic composition according to the corresponding logic composition table. A logic composition table is a table shows what a FR-vector would be given if two FR-vectors of different inputs propagate through a specific logic gate. **Table 2-1** shows the logic composition table in FR-vector.

AND	0	1
0	0	0
1	0	1
R	0	*R*
F	0	*F*

(a) Composition tab

Table 2-1: the logic composition tables for basic logic function in FR-vector

The item in these tables is get from the meaning of the corresponding input frame values. For example, the meaning of *R* and 0 is “01” AND “00”, and it will get 0 at the output, for 0 represents “00”.

2.3.3 FR-Matrix

It is unrealistic to calculate all input pattern in a circuit. For this FR-vector must be transformed into another form, called **FR-matrix**.

Definition 2.5 (FR-matrix): *A FR-matrix is a* $4 \times N$ *matrix. Each column in the matrix represents a frame with the same index, and the entries in the 4 rows will be used to represent the occurrence probability of 0, 1, R, and F in that frame, respectively.*

NAND	0	1
0	1	1
1	1	0
R	1	*F*
F	1	*R*

To obtain an input represented by FR-matrix, first, we should sample the input signal line over multiple clock cycles and obtain an FR-vector for each cycle. Second, counts the number of instances of each distinct FR-vector in the input line. Third, for

(c) Composition tab

each frame, compute the occurrence probabilities of each 0, 1, *F* and *R*. Then we get a FR-matrix. There is an example about how to get FR-matrix from input sampling.

Table 2-2: (a) Signal sampling. (b) Its FR-matrix

Example 2.1 First, we sample the signals that appear in an input line over one hundred cycles, and obtain 100 corresponding FR-vectors. In **Table 2-2**(a) suppose we get six distinct FR-vectors with the number of instances of the state for each vector. Then we can determine the corresponding FR-matrix by computing the probability of the state in each frame from this table. In this example, there are 80 instances that the first frame is in 0-state (30 in the first FR-vector, 20 in the second FR-vector, and so on). And the total number of instances is 100, so the occurrence probability of 0-state in the first frame is 0.8. **Table 2-2**(b) shows the corresponding FR-matrix.

Theorem 2.1: *The expected number of signal switching activities in a FR-matrix for a signal line is simplified to the value in the rows of R and F, called the weight of a FR-matrix. When a FR-matrix is obtained from the set of FR-vectors which appeared in a signal line, the weight of the FR-matrix is exactly the average weight of each*

FR-vector in the signal line.

From Theorem 2.1, we could know the expected number of signal switching activities from a FR-matrix is considered as the total number of signal switching activities over the total number of FR-vectors. From **Table 2-2**(a), the average estimated number of signal switching activities from FR-vectors is (0 + 1 + 1 + 0 + 1)*0.3 + (0 + 1 + 0 + 1 + 0)*0.2 + (0 + 0 + 1 + 1 + 1)*0.25 + (0 + 0 + 1 + 1 + 0)*0.1 + (0 + 1 + 1 + 1 + 0)*0.1 + (0 + 0 + 0 + 1 + 0)*0.05 = 2.6, and the weight of the corresponding FR-matrix (**Table 2-2**(b)) is 0.5 + 0.35 + 0.15 + 0.55 + 0.1 + 0.4 + 0.55 = 2.6.

FR-matrix also could be used in composition according logic composition table. **Table 2-3** is an example of two input FRMs and the corresponding outputs FRM of a OR gate.

Table 2-3: An example of FRM composition: Input FRM 1, (b) Input FRM 2, and (c) Output FRM.

2.3.4 Non-Zero Delay FR-matrix

FR-matrix can group the frame states according to the evaluation dependencies for a given combinational circuit. And we could get the following Theorem:

Theorem 2.2 : *Let N be the frame size,* δ *be the logic gate delay. Let* $FRM_{out}^{\delta}(RFM_{in1}, RFM_{in2})$ *be the output FRM derived from a logic composition with two input FRMs,* FRM_{in1} *and* FRM_{in2}*, for a logic gate with delay* δ*. Let* $RFM_{out}^{0}(RFM_{in1}, RFM_{in2})$ *be the output FRM derived from a logic operation with two input FRMs,* RFM_{in1} *and* RFM_{in2}*, for a zero delay gate. Then*

$$RFM_{out}^{\delta} = [RFM_{out}^{0}]^{\delta},$$

where $[RFM]^{\delta}$ *denotes the right-shifted FRM for* δ *times. In other words, the* i_{th} $(i \le N - \delta)$ *frame in* RFM_{out}^{0} *appears in* $(i+\delta)_{th}$ *frame in* RFM_{out}^{δ}*. And the* j_{th} *frame* $(1 \le j \le \delta)$ *in* RFM_{out}^{δ} *will be filled with the first frame in* RFM_{out}^{0}*.*

And an FRM is also a statistical average of the signal states over multiple clock cycles. We also can apply the following theorem only when the multi-cycle operations are allowed.

Theorem 2.3 : *Let N be the frame size, r be the gate delay. Let* $RFM_{out}^{r}(RFM_{in1}, RFM_{in2})$ *be the output FRM from a logic composition of two input FRMs,* RFM_{in1} *and* RFM_{in2} *for a gate with delay r. Let* $RFM_{out}^{0}(RFM_{in1}, RFM_{in2})$ *be the output FRM from a logic operation with two input FRMs,* RFM_{in1} *and* RFM_{in2}*, for the same gate with a zero delay. Then*

$$RFM_{out}^{r} = [RFM_{out}^{0}]^{r},$$

where $[RFM]^{r}$ *denotes the rotated FRM for r times. In other words, the* i_{th} *state in* RFM_{out}^{0} *appears in* $(((i+r-1)\%N)+1)_{th}$ *state in* RFM_{out}^{r}.

In this thesis, we will take FR-vector as basic model for its ability to process glitches. In additional, it simplifies the calculation in switching activities. The most obvious disadvantage of FR-vector is that it doesn't consider correlations between signals in circuits. In next section, we describe a technique that takes care of correlations.

2.4 Boolean Approximation Method (BAM)

BAM is proposed by Taku Uchino [6]. It is an incremental approach taking into account the first order signal correlation effects by using the Tayler expansion technique to ensure the accuracy. Cofactor probabilities with respect to the primary inputs play a role of the differential coefficients in the Tayler expansion. These cofactor probabilities are calculated incrementally as well as the signal probabilities and switching activities at each circuit node. BAM is able to handle large circuits since the probabilities are calculated incrementally without constructing global BDDs.

2.4.1 Basic Assumptions

There are two assumptions should be set before BAM is used:

- Mutual independence of the primary inputs.
- Invariance of probabilities under time translation

The first assumption implies that the probability of the product of primary inputs can be decomposed into the product of the probabilities at each primary input, that is:

$P(x_i^{\delta_i}(t_k)x_j^{\delta_j}(t_{k+1})) = P(x_i^{\delta_i}(t_k))P(x_j^{\delta_j}(t_{k+1}))$, where $\delta_x \in (1,0)$ is the transition at node x, and $P(x_i^{\delta_i}(t_k))$ denotes the probability of the transition of node x_i is δ_i at time t_k.

The second assumption implies that the probability does not depend on the position of the origin of the time axis, that is:

$P(x_i^{\delta_i}(t_k)x_j^{\delta_j}(t_{k+1})) = P(x_i^{\delta_i}(t_k+t))P(x_j^{\delta_j}(t_{k+1}+t))$ for arbitrary t.

Theorem 2.4 : *In the zero delay model, the switching activity of a logic signal* $x(t)$, *to be denoted by* $P_{sw}(x)$, *is given by:*

$$P_{sw}(x) = 2(P(x)\times(1-P(x)),$$

Proof: *Let* $P(x_{0\to1})$ *is the probability of signal x(t) transition from 0 to 1,* $P(x_{1\to0})$ *is the probability of signal x(t) transition from 1 to 0.* $P(x(\tau)\overline{x(0)})$ *is the probability that* x *is 0 at time 0 and is 1 at time* τ, $P(\overline{x(\tau)}x(0))$ *is the probability that* x *is 1 at time 0 and is 0 at time* τ *.(i.e. the probability of x(t) switching from one state to another). The 1 and 0 mean steady state ONE and state ZERO respectively. Then:*

$$\begin{aligned} P_{sw}(x) &= P(x_{0\to1}) + P(x_{1\to0}) \\ &= 2P(x(\tau)\overline{x(0)}) \\ &= 2P(\overline{x(\tau)}x(0)) \end{aligned}$$

From the assumption "Invariance of probabilities under time translation", we get

$P(x(\tau)\overline{x(0)}) = P(\overline{x(\tau)}x(0)) = P(x)\times(1-P(x))$.

Thus, $P_{sw}(x) = 2(P(x)\times(1-P(x))$.

2.4.2 Signal probability

We will describe how BAM is used with signal probabilities calculation. There are following some definitions in BAM.

Definition 2.6 (Cofactor): *If* $Y = f(x_1, x_2, \ldots, x_n)$ *is Boolean function, the **cofactor** of* *the gate Y is defined as:* $Y[x_i^\alpha]$ $\quad (i=1,2,\ldots,n;\alpha=0,1)$,

where $Y[x_i^\alpha]$ *is derived in the following Shannon expansion:* $Y = x_i^0 Y[x_i^0] + x_i^1 Y[x_i^1]$.

Definition 2.7 (Cofactor Probability): *If* $Y = f(x_1, x_2, \ldots, x_n)$ *is Boolean function, the* ***cofactor probability*** *of the gate* Y *is defined as :* $P(Y[x_i^\alpha])$ $\quad (i=1,2,\ldots,n;\alpha=0,1)$.

Definition 2.8 (BAM data structure): *For any node in a circuit, the BAM data structure of this node is its signal probability and cofactor probabilities .*

By BAM data structure, we could use the first order of Taylor expansion to get the difference between $P(A\cap B)$ and $P(A)P(B)$.

2.4.2.1 2-Input AND Gate Case

We take a 2-input AND gate for example to explain how they are implemented. Suppose A and B are the inputs and Z is the output of a 2-input AND gate in **Figure 2-1**, thus we have $Z = AB$. The difference between $P(AB)$ and $P(A)P(B)$ represents the signal correlation effects. We can get

$$P(AB) = P(Z) \approx P(A)P(B) + \sum_{i=1}^{n} P(x_i)P(\overline{x_i}) \times \left(P(A[x_i^0]) - P(A[x_i^1])\right) \times \left(P(B[x_i^0]) - P(B[x_i^1])\right)$$

and $P(AB[x_i^\alpha]) = P(Z[x_i^\alpha]) \approx P(A[x_i^\alpha])P(B[x_i^\alpha])$. *(Equation 2-1 and Equation 2-2)*

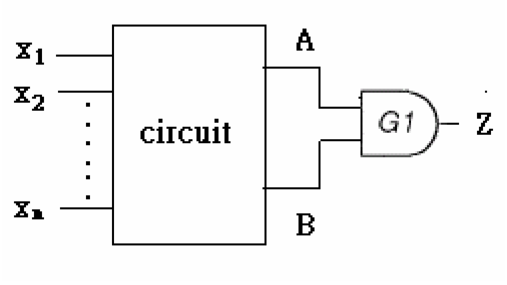


Figure 2-1: 2-input AND gate with n primary inputs.

In the equation

$$P(AB) = P(A)P(B) + \sum_{i=1}^{n} P(x_i)P(\overline{x_i}) \times \left(P(A[x_i^0]) - P(A[x_i^1])\right) \times \left(P(B[x_i^0]) - P(B[x_i^1])\right),$$

the first term of right-hand side of the equation is obtained by considering A and B are mutually independent. The second term is considered as a sum of the correction terms to the first term. These correction term pull-back real number calculation to Boolean algebraic calculation. In fact, for example, these correction terms replace $P(x_i)P(\overline{x_i})$ contained in $P(A)P(B)$ by 0 in accordance with the fact $x_i\overline{x_i} = 0$ in the Boolean algebra. These operations correct the difference of the multiplication law between Boolean algebra and real number, e.g. $x_i x_i = x_i$ ($x_i\overline{x_i} = 0$) and $P(x_i)P(x_i) \neq P(x_i)\left(P(x_i)P(\overline{x_i}) \neq 0\right)$.

To proof the equation mentioned above, suppose A and B are logic functions of mutually independent Boolean variables $x_1, x_2, \ldots, x_n$ .The can be expressed formally in the form of sum of product, for example

$$A = \sum_{\alpha_1=0}^{1} \cdots \sum_{\alpha_n=0}^{1} x_1^{\alpha_1} \cdots x_n^{\alpha_n} A[x^{\vec{\alpha}}],$$

where BAM have defined $\vec{\alpha} \equiv (\alpha_1, \ldots, \alpha_n)$ and $A[x^{\vec{\alpha}}]$ is either 0 or 1 for each $\vec{\alpha}$. The similar equation also holds for B as

$$B = \sum_{\beta_1=0}^{1} \cdots \sum_{\beta_n=0}^{1} x_1^{\beta_1} \cdots x_n^{\beta_n} B[x^{\vec{\beta}}],$$

The probability that the logic value of A equals to 1 and B equals to 1 are obtained as follows:

$$P(A) = \sum_{\alpha_1=0}^{1} \cdots \sum_{\alpha_n=0}^{1} P(x_1^{\alpha_1}) \cdots P(x_n^{\alpha_n}) A[x^{\vec{\alpha}}],$$

$$P(B) = \sum_{\beta_1=0}^{1} \cdots \sum_{\beta_n=0}^{1} P(x_1^{\beta_1}) \cdots P(x_n^{\beta_n}) B[x^{\vec{\beta}}].$$

The product of $P(A)$ and $P(B)$ can be expressed as follows: $P(A)P(B) = F(\eta)$

where $\eta_i^{\alpha\beta} \equiv P(x_i^{\alpha})P(x_i^{\beta})$ $(i = 1, \ldots, n; \alpha, \beta = 0, 1)$,

$$F(\xi) \equiv \sum_{\bar{\alpha}} \sum_{\bar{\beta}} \xi^{\alpha_1\beta_1} \cdots \xi^{\alpha_n\beta_n} A[x^{\bar{\alpha}}]B[x^{\bar{\beta}}]$$

On the other hands, since the form of sum of products for the Boolean product of A and B is such that

$$AB \equiv \sum_{\bar{\alpha}} \sum_{\bar{\beta}} x_1^{\alpha_1} x_1^{\beta_1} \cdots x_n^{\alpha_n} x_n^{\beta_n} A[x^{\bar{\alpha}}]B[x^{\bar{\beta}}]$$

The probability that the logic value of AB equals to 1 can be expressed as follows:

$$P(AB) = F(\chi)$$

where $\chi_i^{\alpha\beta} \equiv P(x_i^\alpha x_i^\beta) \quad (i = 1,\dots,n; \alpha, \beta = 0,1)$

Note that $P(AB)$ and $P(A)P(B)$ have been expressed by a single equation F .Therefore, $P(AB) - P(A)P(B)$ can be approximated by the first-order terms of the Taylor expansion of F :

$$P(AB) - P(A)P(B) = F(\chi) - F(\eta) \approx \sum_{i=1}^{n} \sum_{\alpha=0}^{1} \sum_{\beta=0}^{1} (\chi_i^{\alpha\beta} - \eta_i^{\alpha\beta}) \frac{\partial F(\eta)}{\partial \xi_i^{\alpha\beta}}$$

It's easy to show that

$$\chi_i^{\alpha\beta} - \eta_i^{\alpha\beta} = (-1)^{\alpha+\beta} P(x_i)P(\overline{x_i}) \quad (i = 1,\dots,n; \alpha, \beta = 0,1)$$

and $\dfrac{\partial F(\eta)}{\partial \xi_i^{\alpha\beta}} = P(A[x_i^\alpha])P(B[x_i^\beta])$, where $[x_i^\alpha]$ and $[x_i^\beta]$ are the cofactors of the Shannon expansions of A and B around x_i respectively.

As a result, we obtain the first equation of BAM:

$$P(AB) = P(A)P(B) + \sum_{i=1}^{n} P(x_i)P(\overline{x_i}) \times \left(P(A[x_i^0]) - P(A[x_i^1])\right) \times \left(P(B[x_i^0]) - P(B[x_i^1])\right)$$

By taking the 0th-order Taylor expansion of the difference between $P(AB[x_i^\varepsilon])$ and $P(A[x_i^\alpha])P(B[x_i^\alpha])$, then we could obtain:

$$P(AB[x_i^\alpha]) \approx P(A[x_i^\alpha])P(B[x_i^\alpha]) \quad (i = 1,\dots,n; \alpha = 0,1) .$$

According to the first equation of BAM, the necessary information at the input node, for example A, is a set of signal probability $P(A)$ and cofactor probabilities $P(A[x_i^\alpha])(i = 0,1,\dots n; \alpha = 0,1)$ which is BAM data structure in definition 2.8.

2.4.2.2 General Case

This subsection will present the application of BAM to a general logic gate such as OR, XOR, or more complicated functions.

Suppose the gate under consideration has m inputs, $A_1,\cdots,A_m$, and Z is one of the outputs of the gate. **Figure 2-2** is the illustration of such gate.

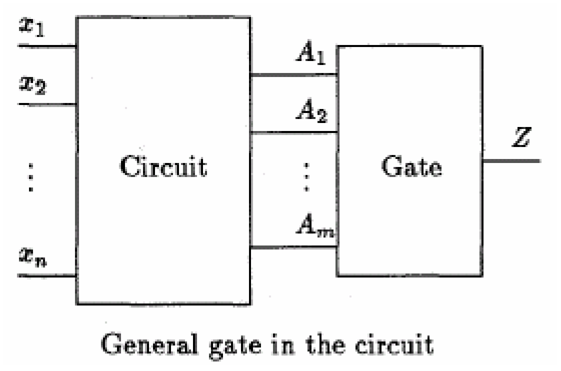


Figure 2-2: General case in the circuit

Shannon expansion of Z around A_1:

$$Z = \overline{A_1}Z[\overline{A_1}] + A_1 Z[A_1]$$

gives the relation between Z and A_1. Since $\overline{A_1}Z[\overline{A_1}](A_1 Z[A_1])$ is the product of two logic functions $\overline{A_1}$ and $Z[\overline{A_1}]$ (A_1 and $Z[A_1]$), **Equation 2-1** can be applied to approximate the signal probability of Z such that

$$P(Z) \simeq P(\overline{A_1}) * P(Z[\overline{A_1}]) + P(A_1) * P(Z[A_1])$$

The Cofactor probabilities at node Z are obtained in the same manner as follows:

$$P(Z[x_i^\alpha]) \simeq P(\overline{A_1[x_i^\alpha]}) * P(Z[\overline{A_1}][x_i^\alpha]) + P(A_1[x_i^\alpha]) * P(Z[A_1][x_i^\alpha]) \quad (i = 1, \cdots, n; \alpha = 0,1)$$

According to equation 2-13 and 2-14, the BAM data structure at node Z is obtain from those for $\overline{A_1}$, A_1, $Z[\overline{A_1}]$, and $Z[A_1]$. The BAM data structure for $\overline{A_1}$ is easily obtained from that for A_1, while those for $Z[\overline{A_1}]$ and $Z[A_1]$ are obtained by recursive Shannon expansion around $A_2, \cdots A_n$.

Equations 2-1 and 2-2 imply that the approximate signal and cofactor probabilities for the gate output Z are obtained from those for the gate inputs immediately without knowledge about the previous circuit.

2.4.3 Algorithm of BAM

The signal probabilities at all circuit nodes can be calculated by means of the following algorithm:

I. Set BAM data structures for the primary inputs.

II. Extract a gate such that BAM data structures for its inputs are already calculated but those for its outputs are nod yet calculated. If such gate does not exist, then exit.

III. Calculate the BAM data structures for the gate outputs from those for the gate inputs by using **Equations 2-1 and 2-2**

IV. Go to II.

2.4.4 Switching Activities

BAM also could be used to the switching activity calculation. But it is more complicated than to the signal the temporal correlation of the identical primary input. Assume A, B, C and D are logic function of primary inputs. The first fundamental equation is as follows:

$$P\left(A(\tau)B(0)C(\tau)D(0)\right) \simeq$$
$$P(A(\tau)B(0))P(C(\tau)D(0)) + \sum_{i=1}^{n} \sum_{(\alpha\alpha')<(\beta\beta')} p_i^{\alpha\alpha'} p_i^{\beta\beta'} \times (X_i^{\alpha\alpha'} - X_i^{\beta\beta'}) \times (Y_i^{\alpha\alpha'} - Y_i^{\beta\beta'})$$

(Equation 2-3)

where $(00) \overset{def}{=} 0, (01) \overset{def}{=} 1, (10) \overset{def}{=} 2, (11) \overset{def}{=} 3,$

$p_i^{\alpha\beta} \overset{def}{=} P(x_i^{\alpha}(\tau)x_i^{\beta}(0)),$

and $X_i^{\alpha\beta} \overset{def}{=} P(A[x_i^{\alpha}](\tau)B[x_i^{\beta}](0)),$ for and $\alpha, \beta = 0,1.$

$Y_i^{\alpha\beta} \overset{def}{=} P(C[x_i^{\alpha}](\tau)D[x_i^{\beta}](0)),$

The Second fundamental equations are as follows:

$$P\left(AC[x_i^{\alpha}](\tau)BD[x_i^{\beta}](0)\right) \simeq P(A[x_i^{\alpha}](\tau))P(B[x_i^{\beta}](0)) \times P(C[x_i^{\alpha}](\tau))P(D[x_i^{\beta}](0)),$$

$(i = 1, \cdots, n; \alpha, \beta = 0,1)$ (Equation 2-4)

2.4.5 Example

In this section, we take the circuit in **Figure 2-3** for an example to see how BAM is implemented to calculate switching activities at each node level by level:

Example 2.2

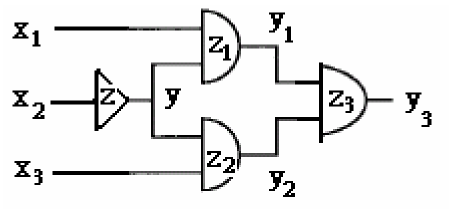


Figure 2-3: A test circuit.

There are some basic values of the circuit in **Figure 2-3**:

$$P(x_1) = 0.5; P(x_2) = 0.5; P(x_3) = 0.5. \qquad P_{sw}(x_1) = 0.1; P_{sw}(x_2) = 0.2; P_{sw}(x_3) = 0.1.$$

Then we could use this information to do calculation.

<u>**Level 0:**</u> The BAM data structures for the primary inputs are set as follows:

i. If $i = j$ then

$$P\left(x_i[x_j^{\alpha}](\tau)x_i[x_j^{\beta}](0)\right)=1, \text{ for } \alpha \text{ and } \beta = 1.$$

$$P\left(x_i[x_j^{\alpha}](\tau)x_i[x_j^{\beta}](0)\right)=0, \text{else.}$$

ii. If $i \neq j$ then

$$P\left(x_i[x_j^{\alpha}](\tau)x_i[x_j^{\beta}](0)\right)=P\left(x_i(\tau)x_i(0)\right)=P(x_i)-0.5\times P_{sw}(x_i)$$

(because x_i and x_j are mutually independent) $(i=1,\cdots,n;\alpha,\beta=0,1)$

e.g.

$$P\left(x_1[x_1^0](\tau)x_1[x_1^0](0)\right)=0, P\left(x_1[x_1^1](\tau)x_1[x_1^1](0)\right)=1,$$
$$P\left(x_1[x_2^0](\tau)x_1[x_2^0](0)\right)=P\left(x_1(\tau)x_1(0)\right)=0.5-0.5\cdot 0.1=0.45$$

<u>Level 1:</u> The BAM data structure for node y is obtained from those at level 0 as

$$P_{sw}(y)=P_{sw}(x_2)=0.2$$

$$P\left(y[x_i^{\alpha}](\tau)y[x_i^{\beta}](0)\right)=P\left(x_2[x_i^{\alpha}](\tau)x_2[x_i^{\beta}](0)\right),(i=1,\cdots,n;\alpha,\beta=0,1).$$

e.g. $P\left(y[x_2^0](\tau)y[x_2^0](0)\right)=P\left(x_2[x_2^0](\tau)x_2[x_2^0](0)\right)=0$

<u>Level 2:</u> The BAM data structure for node y_1 is obtained from those at level 0 and level 1 as follows:

$$P_{sw}(y_1)=2\cdot\left(P\left(\overline{x_1(\tau)}x_1(0)y(\tau)y(0)\right)+P\left(\overline{x_1(\tau)}x_1(0)\overline{y(\tau)}y(0)\right)+P\left(x_1(\tau)x_1(0)\overline{y(\tau)}y(0)\right)\right)=0.14$$

$$P\left(\overline{x_1(\tau)}x_1(0)y(\tau)y(0)\right)$$
$$=P\left(\overline{x_1(\tau)}x_1(0)\right)P\left(y(\tau)y(0)\right)+\sum_{i=1}^{3}\sum_{(\alpha\alpha')<(\beta\beta')}P\left(x_i^{\alpha}(\tau)x_i^{\alpha'}(0)\right)P\left(x_i^{\beta}(\tau)x_i^{\beta'}(0)\right)$$
$$\times\left(P\left(\overline{x_1}[x_i^{\alpha}](\tau)x_1[x_i^{\alpha'}](0)\right)-P\left(\overline{x_1}[x_i^{\beta}](\tau)x_1[x_i^{\beta'}](0)\right)\right)\times\left(P\left(y[x_i^{\alpha}](\tau)y[x_i^{\alpha'}](0)\right)-P\left(y[x_i^{\beta}](\tau)y[x_i^{\beta'}](0)\right)\right)$$

The calculation of $P\left(\overline{x_1(\tau)}x_1(0)\overline{y(\tau)}y(0)\right), P\left(x_1(\tau)x_1(0)\overline{y(\tau)}y(0)\right)$ is similarly.

$$P\left(y_1[x_i^{\alpha}](\tau)y_1[x_i^{\beta}](0)\right)\approx P\left(x_1[x_i^{\alpha}](\tau)x_1[x_i^{\beta}](0)\right)\cdot P\left(y[x_i^{\alpha}](\tau)y[x_i^{\beta}](0)\right)$$

$(i = 1, \cdots, n; \alpha, \beta = 0,1)$

and BAM data structure for node y_2 is obtained similarly.

Level 3: The BAM data structure for node y_3 is obtained from those at level 2 as follows:

$$P_{sw}(y_3) = 2 \cdot \left(P\left(\overline{y_1(\tau)} y_1(0) y_2(\tau) y_2(0)\right) + P\left(\overline{y_1(\tau)} y_1(0) \overline{y_2(\tau)} y_2(0)\right) + P\left(y_1(\tau) y_1(0) \overline{y_2(\tau)} y_2(0)\right)\right) = 0.088$$

$$\begin{aligned}
& P\left(\overline{y_1(\tau)} y_1(0) y_2(\tau) y_2(0)\right) \\
& = P\left(\overline{y_1(\tau)} y_1(0)\right) P\left(y_2(\tau) y_2(0)\right) + \sum_{i=1}^{3} \sum_{(\alpha\alpha')<(\beta\beta')} P\left(x_i^{\alpha}(\tau) x_i^{\alpha'}(0)\right) P\left(x_i^{\beta}(\tau) x_i^{\beta'}(0)\right) \\
& \times \left(P\left(\overline{y_1}[x_i^{\alpha}](\tau) y_1[x_i^{\alpha'}](0)\right) - P\left(\overline{y_1}[x_i^{\beta}](\tau) y_1[x_i^{\beta'}](0)\right)\right) \times \left(P\left(y_2[x_i^{\alpha}](\tau) y_2[x_i^{\alpha'}](0)\right) - P\left(y_2[x_i^{\beta}](\tau) y_2[x_i^{\beta'}](0)\right)\right)
\end{aligned}$$

The calculation of $P\left(\overline{y_1(\tau)} y_1(0) \overline{y_2(\tau)} y_2(0)\right), P\left(y_1(\tau) y_1(0) \overline{y_2(\tau)} y_2(0)\right)$ is similarly.

$$P\left(y_3[x_i^{\alpha}](\tau) y_3[x_i^{\beta}](0)\right) \approx P\left(y_1[x_i^{\alpha}](\tau) y_1[x_i^{\beta}](0)\right) \cdot P\left(y_2[x_i^{\alpha}](\tau) y_2[x_i^{\beta}](0)\right)$$

$(i = 1, \cdots, n; \alpha, \beta = 0,1)$

In this example BAM gives the exact signal probability for node y_3. The exact signal probability at node y_3 can be derived by:

$$P_{sw}(y_3) = P_{sw}\left((x_1 x_2)(x_2 x_3)\right) = P_{sw}(x_1 x_2 x_2 x_3) = 0.088$$

BAM could estimate the switching activities at all nodes in a combinational logic circuit with relative accuracy and without constructing global BDDs. It uses the concept of Taylor expansion to get the first order signal dependence effects due to reconvergent fan-out nodes are taken into account. Further more, it take the temporal correlation among the primary input in estimating switching activities, but ignore spatial correlation.

2.5 Real-Delay Boolean Function (RDBF)

Considering for glitches generation again, a glitch is generated at the output of a gate, if the following conditions are met:

i. The *necessary condition*, which requires the difference of the arrival times of the input signals to be greater than the inertial delay of the gate.

ii. The *sufficient condition*, which requires the appropriate transitions of the input signals(s) to switch the gate output.

Consequently, the timer parameter plays a critical role in the power consumption estimation. Hence, the exact description of a logic circuit should include not only its logic signals. Furthermore, since the glitch generation is strongly dependent on time, a modified Boolean function — *real delay Boolean function (RDBF)*, which describes the logic and timing behavior of each signal, is needed.

Example 2.3 We assume the logic circuit of **Figure 2-4**, where the gate delays are multiples of reference delay unit d .

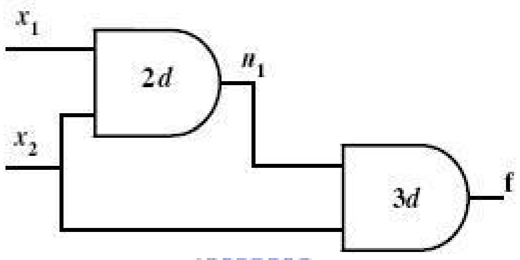


Figure 2-4: A logic circuit with non-zero delay gates.

The logic behavior of the node f can be described in time domain by the following RDBF:

$$f = F(x_1, x_2, t)$$

$$= x_1(t-5d)x_2(t-5d)x_2(t-3d) \quad \textit{(Equation 2-1)}$$

The signal f may switch at two time instances, i.e. $t_1^f = 3d$ and $t_2^f = 5d$. More specifically, the transition of the signal f at t_1, $t_1^f = 3d$, depends on the transitions of the primary inputs x_1 and x_2 at time points $t_1^{x_1} = -2d, t_1^{x_2} = -2d$, and $t_1^{x_2} = 0$. The corresponding Boolean function is

$$f_1 = x_1(-2d)x_2(-2d)x_2(0) .$$

The transition of f at $t_2^f = 5d$ depends on the transition of the signal x_1 and x_2 at $t_2^{x_1} = 0$ and $t_2^{x_2} = 0$, while the corresponding Boolean function is

$f_2 = x_1(0)x_2(0)x_2(2d)$.

Thus, the behavior of the signal f is described in time domain by the corresponding RDBF. Moreover, the RDBF of f is reduced to ordinary Boolean function f_1 and f_2, whose variables are the logic values of the input signals at specific time instances. Also, since node f may perform transitions at $t_1^f = 3d$ and $t_2^f = 5d$, evaluation the transition activity of function f_1, f_2 the transition activity of the RBDF is also evaluated.

For us, we take FR-Vector to process glitches, and combine it with BAM to consider the correlations. In original BAM, it must be operated in a zero-delay model. .For this, we need to add the delay information into BAM by real delay Boolean function. We would discuss it in detail in next chapter.

Chapter 3 FR-Vector with BAM

3.1 Review and Integration

The goal of this paper is to propose a new technique which could analyze switching activity, and further more, signal correlations and glitches processing are included. In section 2.3 , we introduce FR-vector. It could be used in a non-zero delay model and it could also process glitch activity. In additional, while most papers use complex equations to calculate switching probability, FR-vector simplifies the calculations of switching probability as the calculations signal probability. As we can see, though there are many advantages in FR-vector, the signal correlations are not included in FR-vector.

The other techniques we talked in section 2.4 is Boolean approximation method, it uses Taylor expansion to approximate the difference between two probabilities $P(A)$ and $P(B)$ multiplied immediately and the real probabilities as they intersected－ $P(A \cap B)$. BAM indeed is a data structure, which could be used to calculate the difference mentioned above. BAMs are used to pass through the circuit and calculate the signal (or switching) probability of nodes in the whole circuit. In BAM, it solves the correlations due to reconvergent circuit by the concept of Taylor expansion. Another advantage of BAM is it doesn’t need to construct OBDD of the circuit, so it is faster than many other techniques which processing reconvergent circuits. Unfortunately, the delay model use in BAM is zero-delay model. Without delay information of the circuit, the activity of glitches would be ignored too.

The ability of glitch processing in FR-Vector, and the ability of signal correlations processing in BAM both are what we want. So, in this thesis, we try to combine these two techniques as a new model, and call it as FR-Vector with BAM.

3.2 Basic Definition and assumption

FR-Vector wit BAM is a *non-zero delay* model in *inertial mode*. It has an assumption:

- Mutual independency of the primary inputs.

This assumption is gotten from the original first assumption in BAM, and we eliminate the second assumption to fit with FR-Vector. The inertial model is the basic condition in FR-Vector.

As in FR-Vector, we will sample a signal line S n times in a clock cycle, then divide a clock cycle into n states (state $= 0,1$). Then, using the sampled rule to get *FR-Vector* (n frames a vector, each frame $= 0, F, R, 1$). From FR-Vector, we could get *FR-Matrix* ($4\times N$ matrix. Each column in the matrix represents a frame with the same index, and the entries in the 4 rows will be used to represent the occurrence probability of 0, 1, R, and F in that frame, respectively.). Following is the explanation of the meaning of FR-Matrix in FR-Vector with BAM:

Definition 3.1 (Switching signal):*In FR-Vector with BAM, the* ith *frame,* $x_{t=i}$, *in FR-Vector is called the switching signal in time* i. $(0 < i < n-1)$

Definition 3.2 (Switching probability): *In FR-Vector with BAM, the* ith *frame in FR-Matrix is called the switching probability in time* i, *and denoted as* $P(x_{t=i})$, $(x = 0, F, R, 1 \quad ; \quad 0 < i < n-1)$.

In FR-Vector with BAM, we also used a data structure to carry the information of correlation among the circuit. Recall the cofactor probability used in section 2-4, we also have cofactor transition relation in FR-Vector with BAM. The differences are:

I. We add the concept of “frame” as a basic time unit in place of “clock”.

II. The BAM data structures in FR-Vector only exist in nodes which are in reconvergent part of the circuit.

Definition 3.3 (Cofactor Transition Relation): *If* $Y = f(x_1, x_2, \ldots, x_n)$ *is Boolean function, and we having* m *frames in a clock cycle. The cofactor transition relation of it are defined as:*

$$\begin{aligned}
P(Y_f^0[x_i^{\alpha\beta}]) &= P\left(\overline{Y_f}[x_i^{\alpha}](t)\overline{Y_f}[x_i^{\beta}](t-1)\right) \\
P(Y_f^F[x_i^{\alpha\beta}]) &= P\left(\overline{Y_f}[x_i^{\alpha}](t)Y_f[x_i^{\beta}](t-1)\right) \\
P(Y_f^R[x_i^{\alpha\beta}]) &= P\left(Y_f[x_i^{\alpha}](t)\overline{Y_f}[x_i^{\beta}](t-1)\right) \\
P(Y_f^1[x_i^{\alpha\beta}]) &= P\left(Y_f[x_i^{\alpha}](t)Y_f[x_i^{\beta}](t-1)\right)
\end{aligned} \quad (i = 1, 2, \ldots, n\,;\, \alpha, \beta = 0,1\,;\, 0 < f < m-1)$$

which means the probabilities of cofactors remains 0, transition from1 to 0, transition from 1 to 0, and remains 1 at frames f *respectively.*

Definition 3.3 (Reconvergent circuit): *A reconvergent circuit is a circuit which convergent in a specific node and some signal lines in this circuit has forked before its convergent node.*

Definition 3.4 (Supply set of the reconvergent circuit): *A supply set* S_X *of a reconvergent circuit X is a set of nodes* $\{s_1, s_2, \ldots, s_n\}$, *which* s_i *is the input of reconvergent circuit X.*

There is an example of supply set:

Example 3.1 There is a circuit likes **Figure 3-1**:

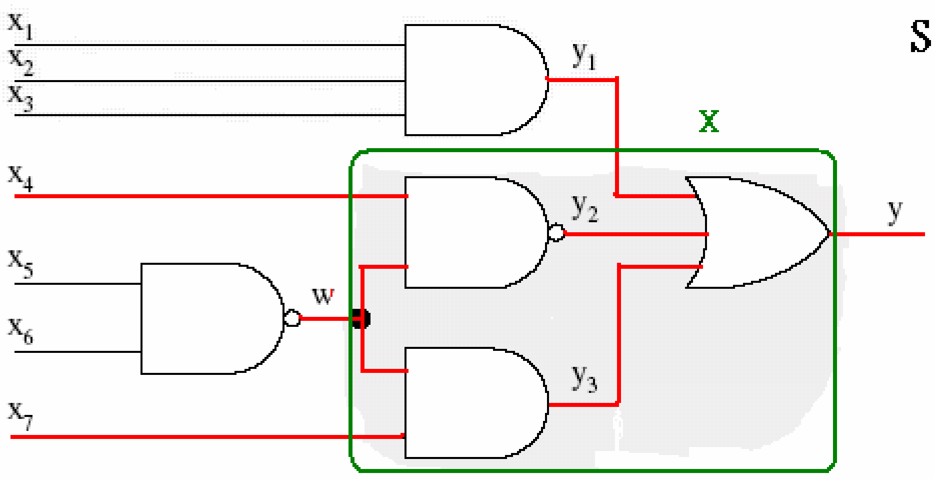


Figure 3-1: An example of supply set of the reconvergent circuit: In reconvergent circuit X , the supply set of X is $\{x_{4,}w, x_7, y_2\}$.

In **figure 3-1**, there is a reconvergent circuit $X = \{w, x_4, x_7, y, y_2, y_3\}$ in circuit S , and the supply set of X is $\{x_{4,}w, x_7, y_2\}$.

Definition 3.5 (Cofactor Transition Relation Matrix): *A* ***cofactor transition relation matrix*** *is a three dimension matrix of node Y . (A $4 \times f \times n$ matrix, n is the number of nodes of the reconvergent circuit which node A is belonged to, and f is the number of frames.) Each item in the matrix denotes the cofactor transition relation $P(Y_t^{\gamma}[x_i^{\delta}])$ of node Y where $\delta, \gamma = 0, F, R, 1$; $0 < t \leq f-1$; and $0 < i \leq n$.*

Definition 3.6 (The BAM data structure of FR-Vector): *For any node belonged to a reconvergent in a circuit, the FR-Matrix and the cofactor transition matrix of this node is called its* ***BAM data structure of FR-Vector****.*

Because of the assumption — "Mutual independency among primary input", each pair of nodes which are not in the reconvergent circuit or they are in the supply set of a reconvergent circuit, would be taken as independent. By this, we could get:

Theorem 3.1 : *The BAM data structures of a node x_i , $x_i \in S$ and $S = \{x_1, x_2, \ldots, x_n\}$ is a supply set of the reconvergent circuit, could be gotten as:*

(i) *If* $i = j$, $P(x_i^{\delta}[x_j^{\gamma}]) = 1$, *when* $\delta = \gamma$,

$$P(x_i^{\delta}[x_j^{\gamma}]) = 0 \text{, } when \ \ \delta \neq \gamma .$$

(ii) *If* $i \neq j$, $P(x_i^{\delta}[x_j^{\gamma}]) = P(x_i^{\delta})$.

$$(\ 0 < i \leq n-1 \ ; \ \delta, \gamma = 0, F, R, 1 \ )$$

Now, we know how to sample the signal as FR-Vector and get FR-Matrix of primary inputs. In further more, we can use theorem 3.1 to get their BAM data structures if it is in reconvergent circuit. In next section, we talk about how to propagate these information gate by gate, and from primary inputs to the output of the circuit.

3.3 Probabilities Propagation

In convenient to introduce FR-Vector with BAM, we start in zero-delay model, we will extend FR-Vector with BAM to non-zero delay model in section 3.6.

In order to analyze the probabilities at each node in the circuit, we divide the circuit into two parts:

- General circuit, and
- Reconvergent circuit.

In general circuit, we do not use BAM data structure. Because of the assumption “Mutual independence among the primary input”, if there is no reconvergent circuit, there is no correlation among circuit too. For this reason, it is redundant to calculate and propagate the BAM data structure in these circuits. Indeed, in non-reconvergent circuit, even if we still use the BAM data structure in it, it would do nothing. We leave the explanation of this later in section 3.3.2. Without BAM data structures, our model is simplify to original FR-Vector, so we could propagate FR-Matrix as it propagates in original FR-Vector.

In the other case, reconvergent circuit, we will set up BAM data structure using theorem 3.1 in the supply set of the reconvergent circuit. Then we need to propagate these BAM data structure to calculate the other nodes which is not in supply set in the reconvergent circuit, including cofactor transition relation and FR-Matrix.

3.3.1 Propagation of the Cofactor Transition Relation

Before talking about the propagation of the cofactor transition relation, we should discuss the real significance in it. The cofactor comes from Shannon expansion. Suppose the gate under consideration has m inputs, $A_1,\cdots A_m$, and Z is one of the outputs of the gate. Shannon expansion of Z around A_1 is

$$Z = \overline{A_1}Z[\overline{A_1}] + A_1 Z[A_1],$$

and thus

$$P(Z) = P(\overline{A_1})P(Z[\overline{A_1}]) + P(A_1)P(Z[A_1]) .$$

We could say that a cofactor $P(Z[x_i])$ is the probability of Z is logic high when x_i is logic high. As the same concept, we could get:

Lemma 3.1 *If there is a gate whose inputs are $(x_1, x_2, \ldots, x_n)$ and one of its output is Z, we take the cofactor transition relation $P(Z_\delta[x_i^\alpha])$ as the probability of transition δ $(\delta = 0, F, R, 1)$ happens at output Z when the transition of its input x_i is α $(\alpha = 0, F, R, 1)$.*

Theorem 3.2 : *If there is a gate whose inputs are A and B, the circuit it belonged to has the primary input set $\{x_1, x_2, \ldots, x_n\}$, and a clock cycle is divided into m frames, then we have:*

$$\begin{aligned} &P(A_f^\alpha[x_i^\delta] \cap B_f^\beta[x_i^\delta]) \\ &= P\left(A_f B_f[x_i^{\delta_\tau}](\tau) A_f B_f[x_i^{\delta_0}](0)\right) \\ &\simeq P(A_f[x_i^{\delta_\tau}](\tau) A_f[x_i^{\delta_0}](0)) \times P(B_f[x_i^{\delta_\tau}](\tau) B_f[x_i^{\delta_0}](0)) , \\ &= P(A_f^\alpha[x_i^\delta]) P(B_f^\beta[x_i^\delta]) \end{aligned} \qquad \textbf{(Equation 3-1)}$$

$$(i = 1, \cdots, n;\ \delta = 0, F, R, 1;\ \delta_\tau, \delta_0 = 0, 1;\ f = 1, \cdots, m)$$

Proof:

The proof of this theorem is similar to **equation 2-4** in **section 2.4**, we get the **equation 3-1** in the same way as how we get **equation 2-4.** The difference is we add the concept of “frame”.

By lemma 3.1, we could use the logic composition table (**Table 3-1**) as FR-Vector used to propagate FR-matrix to propagate the cofactor transition relation.

To explain how to use these tables, we take an example as below:

Example 3.2 There is an AND gate whose inputs are a, b and output is c ,the primary inputs is $(x_1, x_2, \ldots, x_n)$, and there are m frames in a clock cycle. Thus, according to **table 3.1(a)** we could get

$$P(c_f^0[x_i^\alpha]) = P(a_f^0[x_i^\alpha])\times P(b_f^0[x_i^\alpha]) + P(a_f^0[x_i^\alpha])\times P(b_f^F[x_i^\alpha]) + P(a_f^0[x_i^\alpha])\times P(b_f^R[x_i^\alpha]) +$$
$$P(a_f^0[x_i^\alpha])\times P(b_f^1[x_i^\alpha]) + P(a_f^1[x_i^\alpha])\times P(b_f^0[x_i^\alpha]) + P(a_f^F[x_i^\alpha])\times P(b_f^0[x_i^\alpha])$$
$$+P(a_f^R[x_i^\alpha])\times P(b_f^0[x_i^\alpha]) + P(a_f^R[x_i^\alpha])\times P(b_f^F[x_i^\alpha]) + P(a_f^F[x_i^\alpha])\times P(b_f^R[x_i^\alpha])$$

$$P(c_f^F[x_i^\alpha]) = P(a_f^1[x_i^\alpha])\times P(b_f^F[x_i^\alpha]) + P(a_F[x_i^\alpha])\times P(b_f^F[x_i^\alpha]) + P(a_f^F[x_i^\alpha])\times P(b_f^1[x_i^\alpha])$$

\

Table 3-1: The composition table for cofactor transition relation.

$$P(c_f^R[x_i^\alpha]) = P(a_f^1[x_i^\alpha])\times P(b_f^R[x_i^\alpha]) + P(a_f^R[x_i^\alpha])\times P(b_f^R[x_i^\alpha]) + P(a_f^R[x_i^\alpha])\times P(b_f^1[x_i^\alpha])$$

and $P(c_f^1[x_i^\alpha]) = P(a_f^1[x_i^\alpha])\times P(b_f^1[x_i^\alpha])$.

$(0 < i < n-1; \quad \alpha = 0, F, R, 1; \quad 1 < f < m-1)$

For example, if we want to get the probability of transition F happens at output c when the transition of node x_i is α, that is $P(c_f^F[x_i^\alpha])$. We should know $c_f^F[x_i^\alpha]$ will happen at time $a_f^1[x_i^\alpha]\cap b_f^F[x_i^\alpha]$, $a_f^F[x_i^\alpha]\cap b_f^F[x_i^\alpha]$, and $a_f^F[x_i^\alpha]\cap b_f^1[x_i^\alpha]$ (where $A_f^\delta[x_i^\alpha]$ means node transition δ happens at node A at the time the transition of x_i is α in time frame

AND 0
0 0
1 0
R 0
F 0

(a) Composition

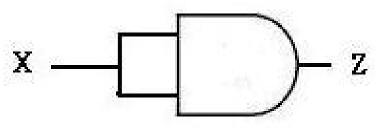
f). So we could get

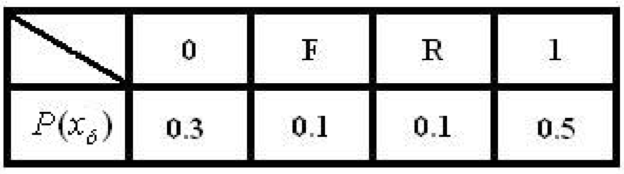
$$\begin{aligned}
&P(c_F[x_i^\alpha]) = P(c_f^R[x_i^\alpha]) \\
&= P(a_f^1[x_i^\alpha] \cap b_f^R[x_i^\alpha]) + P(a_f^R[x_i^\alpha] \cap b_f^R[x_i^\alpha]) + P(a_f^R[x_i^\alpha] \cap b_f^1[x_i^\alpha])
\end{aligned}$$

3.3.2 Propagation of the FR-Matrix

In FR-Vector with BAM, we also use logic composition table to propagate the FR-Matrix, but the difference is when the node is in a reconvergent part of the circuit. That is, in the original FR-Vector, we get the FR-Matrix of an output from its immediate inputs. It comes from two items in each matrix of the input multiplied immediately, and without considering signal correlation due to reconvergent circuit. Though there may be distance between two probabilities multiplied immediately and the real probability it will be, the original FR-Vector ignores this. Take a simple example to see how the reconvergent circuit affects the transition probability:

Example 3.3 **Figure 3-2**is a simple reconvergent with input node x, and output node z. The right table is the FR-Matrix of node x.





	0	F	R	1
$P(x_\delta)$	0.3	0.1	0.1	0.5

Figure 3-2: ***A reconvergent circuit and the FR-Matrix of its input node x.***

In this circuit, the signal will diverge from node x, and reconvergent at node z. If we use the original FR-Vector to calculate the transition probability of node z as its transition is 1, it will be $P(z^1) = P(x^1) \times P(x^1) = 0.5 \times 0.5 = 0.25$. Indeed, the real probability is $P(z^1) = P(x^1) = 0.5$.

As mentioned before, considering for the correlation due to reconvergent, we use

BAM to handle it. Replace the calculation— $P(A^{\alpha} \cap B^{\beta}) = P(A^{\alpha})P(B^{\beta})$ in the original FR-Vector (that is, multiplying two items in different FR-Matrices), we have:

Theorem 3.3 : *At the convergent node of a reconvergent circuit, if there is a gate whose inputs are A and B, the circuit it belonged to has the primary input set $\{x_1, x_2, \ldots, x_n\}$, and a clock cycle is divided into m frames, then we have:*

$$\begin{aligned} P(A_f^{\delta} \cap B_f^{\gamma}) &= P\left(A_f^{\delta_\tau}(\tau) B_f^{\gamma_\tau}(0) A_f^{\delta_0}(\tau) B_f^{\gamma_0}(0)\right) \\ &\simeq P(A_f^{\delta})P(B_f^{\gamma}) + \sum_{i=1}^{n} \sum_{\alpha<\beta} p_{i\,f}^{\alpha} p_{i\,f}^{\beta} \times (X_{i\,f}^{\alpha} - X_{i\,f}^{\beta}) \times (Y_{i\,f}^{\alpha} - Y_{i\,f}^{\beta}) \end{aligned} \quad , \qquad \textbf{(Equation 3-2)}$$

$$\text{where } (0) \overset{def}{=} 0, (F) \overset{def}{=} 1, (R) \overset{def}{=} 2, (1) \overset{def}{=} 3, \qquad \text{and } \begin{aligned} p_{i\,f}^{\alpha} &\overset{def}{=} P(x_{i\,f}^{\alpha}) \\ X_{i\,f}^{\alpha} &\overset{def}{=} P(A_f^{\delta}[x_{i\,f}^{\alpha}]), \\ Y_{i\,f}^{\alpha} &\overset{def}{=} P(B_f^{\gamma}[x_{i\,f}^{\alpha}]) \end{aligned}$$

$$(\delta, \gamma, \alpha, \beta = 0, F, R, 1;\ f = 0, \cdots m;\ \delta_\tau, \delta_0, \gamma_\tau, \gamma_0 = 0, 1;\ i = 1, \cdots, n.)$$

Proof:

The proof of this theorem is similar to **equation 2-3** in **section 2.4**, we get the **equation 3-2** in the same way as how we get **equation 2-3.** The difference is we add the concept of "frame".

In equation 3.2, the second item of right-hand side

$\sum_{i=1}^{n} \sum_{\alpha<\beta} p_{i\,f}^{\alpha} p_{i\,f}^{\beta} \times (X_{i\,f}^{\alpha} - X_{i\,f}^{\beta}) \times (Y_{i\,f}^{\alpha} - Y_{i\,f}^{\beta})$ is the difference we used to modify the original equation in FR-Vector.

Recall the example 3.3, if we used FR-Vector with BAM to calculate $P(z^1)$, it would be

$$\begin{aligned} P(Z^1) &= P(x^1 \cap x^1) \\ &\simeq P(x^1)P(x^1) + \sum_{x^1} \sum_{\alpha<\beta} p_i^{\alpha} p_i^{\beta} \times (X_i^{\alpha} - X_i^{\beta}) \times (Y_i^{\alpha} - Y_i^{\beta}) \\ &= (P(x^1))^2 + \sum_{\alpha<\beta} P(x^{\alpha})P(x^{\beta}) \times (P(x^1[x^{\alpha}]) - P(x^1[x^{\beta}])) \times (P(x^1[x^{\alpha}]) - P(x^1[x^{\beta}])) \\ &= (P(x^1))^2 + \sum_{(\alpha,\beta)=(0,1),(1,2),(1,3)} P(x^{\alpha})P(x^{\beta}) \times (P(x^1[x^{\alpha}]) - P(x^1[x^{\beta}])) \times (P(x^1[x^{\alpha}]) - P(x^1[x^{\beta}])) \end{aligned}$$

$$
\begin{aligned}
&= (P(x^1))^2 + P(x^0)P(x^1)\times(P(x^1[x^0]) - P(x^1[x^1]))\times(P(x^1[x^0]) - P(x^1[x^1])) \\
&+ P(x^1)P(x^2)\times(P(x^1[x^1]) - P(x^1[x^2]))\times(P(x^1[x^1]) - P(x^1[x^2])) \\
&+ P(x^1)P(x^3)\times(P(x^1[x^1]) - P(x^1[x^3]))\times(P(x^1[x^1]) - P(x^1[x^3])) \\
&= (P(x^1))^2 + P(x^0)P(x^1) + P(x^1)P(x^2) + P(x^1)P(x^3) \\
&= 0.25 + 0.03 + 0.03 + 0.05 = 0.36
\end{aligned}
$$

In the beginning of this section, we have mentioned:

Theorem 3.4 : The process of *BAM data structures in non-reconvergent will do nothing in the propagation of switching probability.*

Proof:

For each gate in non-reconvergent circuit, we divide them into two cases:

(i) The inputs of the gate is primary input:

$$
\begin{aligned}
&(X_{if}^{\alpha} - X_{if}^{\beta})\times(Y_{if}^{\alpha} - Y_{if}^{\beta}) \\
&= (P(A_f^{\delta}[x_{if}^{\alpha}]) - P(A_f^{\delta}[x_{if}^{\beta}]))\times(P(B_f^{\gamma}[x_{if}^{\alpha}]) - P(B_f^{\gamma}[x_{if}^{\beta}]))
\end{aligned}
$$
, where this equation is

extract form equation 3.2.

From theorem 3.1 and $A \neq B$, there must be one of

$\{(X_{if}^{\alpha} - X_{if}^{\beta}), (Y_{if}^{\alpha} - Y_{if}^{\beta})\}$ would be 0,

(for there must be $A \neq x_i$ or $B \neq x_i$)

Thus, the difference equation $\sum_{i=1}^{n}\sum_{\alpha<\beta} p_{if}^{\alpha} p_{if}^{\beta} \times (X_{if}^{\alpha} - X_{if}^{\beta})\times(Y_{if}^{\alpha} - Y_{if}^{\beta})$ will be 0.

(ii) The input of the gate is not primary input:

Because the gate is in a non-reconvergent circuit, the transition of a node X would be independent of the primary input x_i.

Thus we can get $P(A_f^{\delta}[x_{if}^{\alpha}]) = P(A_f^{\delta}[x_{if}^{\beta}])$, and $B_f^{\gamma}[x_{if}^{\alpha}]) = P(B_f^{\gamma}[x_{if}^{\beta}])$,

Then,
$$
\begin{aligned}
&(X_{if}^{\alpha} - X_{if}^{\beta})\times(Y_{if}^{\alpha} - Y_{if}^{\beta}) \\
&= (P(A_f^{\delta}[x_{if}^{\alpha}]) - P(A_f^{\delta}[x_{if}^{\beta}]))\times(P(B_f^{\gamma}[x_{if}^{\alpha}]) - P(B_f^{\gamma}[x_{if}^{\beta}])) \\
&= (0)\times(0) \\
&= 0
\end{aligned}
$$

Thus, the difference equation $\sum_{i=1}^{n}\sum_{\alpha<\beta} p_{i_f}^{\alpha} p_{i_f}^{\beta} \times (X_{i_f}^{\alpha} - X_{i_f}^{\beta}) \times (Y_{i_f}^{\alpha} - Y_{i_f}^{\beta})$ will be 0. From (i) and (ii), we could get the conclusion that, *BAM data structures in non-reconvergent will do nothing in the propagation of switching probability*

Beside we do not propagate the switching probability in non-reconvergent; we only calculate the switching probability at the node where reconvergent circuit is converged for there is only this node would be affected by reconvergent correlation. Thus, we use FR-Vector among the whole circuit, and use BAM data structures in the reconvergent part of the circuit and propagate them gate by gate until the convergent node, *but*

Lemma 3.2 *We only use BAM data structure to calculate the switching probability at convergent node of a reconvergent circuit.*

Again, we discuss how these BAM data structures solve the correlations caused by reconvergent circuit. Let's survey the proof of equation 3-2 firstly.

A part of proof of equation 3-2:

If $P(x_f^{\delta})$ is the probability of the transition δ happens in the f th frame of x, then let

$$P(A_f^{\delta}) \times P(B_f^{\gamma}) = F(\eta),$$

where

$$\eta_i^{\alpha\beta} \overset{def}{=} P(x_{i\ f}^{\alpha})P(x_{i\ f}^{\beta})$$
$$(i = 1,\cdots,n;\ \alpha,\beta,\gamma,\beta = 0,1,2,3;\ 0 \equiv 00, 1 \equiv 01, 2 \equiv 10, 3 \equiv 11)$$

and

$$F(\xi) \overset{def}{=} \sum_{\bar{\alpha}}\sum_{\bar{\beta}} \xi_1^{\alpha_1\beta_1} \cdots \xi_n^{\alpha_n\beta_n} A[x^{\bar{\alpha}}{}_f]B[x^{\bar{\beta}}{}_f]$$

$(\bar{\alpha} \overset{def}{=} (\alpha_{1f},\cdots,\alpha_{nf}))$, and $A[x_{i\ f}^{\bar{\alpha}}]$ is either 0 or 1 for each $\bar{\alpha}$).

The Boolean product of A_f^{δ} *and* B_f^{γ} is such that

$$A_f^\delta B_f^\gamma = \sum_{\bar{\alpha}} \sum_{\bar{\beta}} x_1^{\alpha_1\beta_1}{}_f \cdots x_n^{\alpha_n\beta_n}{}_f \, A[x^{\bar{\alpha}}{}_f] B[x^{\bar{\beta}}{}_f] \,,$$

the probability that the logic value of $A_f^\delta B_f^\gamma$ equals to 1 can be expressed as follows:

$$P(A_f^\delta B_f^\gamma) = F(\chi)$$

where $\chi_i^{\alpha\beta} \overset{def}{=} P(x_i^\alpha{}_f x_i^\beta{}_f)$ $(i = 1, \cdots n;\ \alpha, \beta = 0, 1, 2, 3)$.

Note that $P(A_f^\delta B_f^\gamma)$ and $P(A_f^\delta)P(B_f^\gamma)$ have been expressed by a single function F .

Therefore, $P(A_f^\delta B_f^\gamma) - P(A_f^\delta)P(B_f^\gamma)$ can be approximated by the first-order terms of the Taylor expansion of F :

$$\begin{aligned} P(A_f^\delta B_f^\gamma) - P(A_f^\delta)P(B_f^\gamma) &= F(\chi) - F(\eta) \\ &\simeq \sum_{i=1}^{n} \sum_{\alpha=0}^{3} \sum_{\beta=0}^{3} (\chi_i^{\alpha\beta} - \eta_i^{\alpha\beta}) \frac{\partial F}{\partial \xi_i^{\alpha\beta}}(\eta) \,, \end{aligned}$$

where $\frac{\partial F}{\partial \xi_i^{\alpha\beta}}(\eta) = P(A[x_i^\alpha{}_f])P(B[x_i^\beta{}_f])$, and $A[x_i^\alpha{}_f]$ and $B[x_i^\beta{}_f]$ are the cofactor translation relation of the Shannon expansions of A and B around x_i respectively.

Thus,

$$(\chi_i^{\alpha\beta} - \eta_i^{\alpha\beta}) \frac{\partial F}{\partial \xi_i^{\alpha\beta}}(\eta) = \left(P(A_f^\delta B_f^\gamma) - P(A_f^\delta)P(B_f^\gamma)\right) \times P(A[x_i^\alpha{}_f])P(B[x_i^\beta{}_f]) \,.$$

(Equation 3-3)

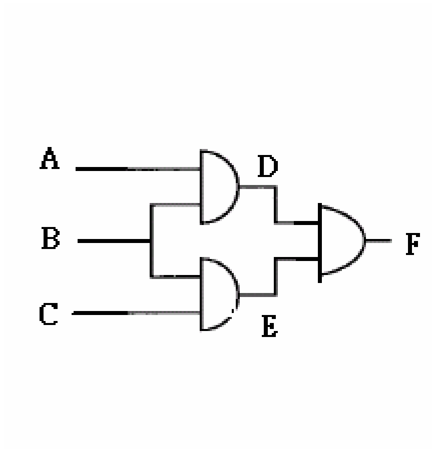
From the right-hand side of (Equation 3-3, we could get the physical meaning of equation 3-2. For the equation 3-3 means the effect do to transition at node A and node B caused by input variables $x_i^\alpha{}_f$ and $x_i^\beta{}_f$. We could see $\sum_{\alpha=0}^{3} \sum_{\beta=0}^{3} (\chi_i^{\alpha\beta} - \eta_i^{\alpha\beta}) \frac{\partial F}{\partial \xi_i^{\alpha\beta}}(\eta)$ is the summation of the effect caused by all transition at node $x_{i\,f}$, and $\sum_{i=1}^{n} \sum_{\alpha=0}^{3} \sum_{\beta=0}^{3} (\chi_i^{\alpha\beta} - \eta_i^{\alpha\beta}) \frac{\partial F}{\partial \xi_i^{\alpha\beta}}(\eta)$ is the summation of all effect caused by input variables. Thus equation 3-2 means the equation $P(A_f^\delta B_f^\gamma)$ is approximating to $P(A_f^\delta)P(B_f^\gamma)$ add the effect caused by input variables. Further more, for the cofactor is generated by the characteristic of circuit and its structure. The effect calculated by equation 3-2 could catch the effect due to different types of circuit structure, thus the reconvergent circuit.

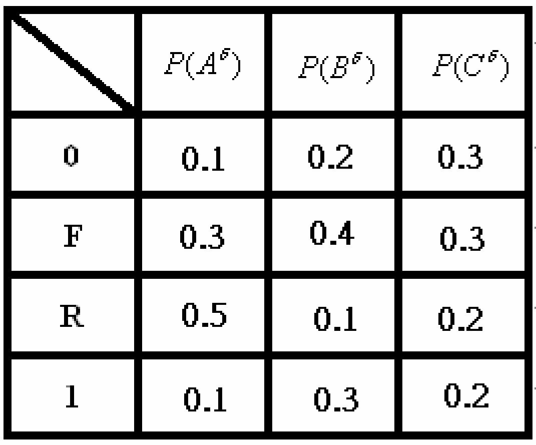
A node’s cofactor transition relation represents the meaning of how each primary input affect the transition in node itself. Thus, in reconvergent circuits, we propagate this information to the immediate inputs of the convergent node, and use equation 3-2 to correct the switching probability in convergent node. Why this equation could do correction. It’s because when we use the logic composition table and FR-Matrices of inputs to compute the FR-Matrix of output node, equation 3.2 could correct the differences between the probability of two event happened in the same time (what we need) and the probability of two probabilities of event multiplied immediately(two terms in logic composition table multiplied immediately). The correction is gotten from the effect due to primary inputs which has been recorded in cofactor transition relation as we have explained above.

3.4 An Example of FR-Vector with BAM

In previous section, we have a tiny attempt in FR-Vector in BAM. In this section, we try to use a simple and more complete example to show the propagation of the FR-Vector with BAM in reconvergent circuit.

Example 3.4 There is a simple reconvergent circuit, and **Table 3-2**is its switching probabilities in each input nodes.





	$P(A^\delta)$	$P(B^\delta)$	$P(C^\delta)$
0	0.1	0.2	0.3
F	0.3	0.4	0.3
R	0.5	0.1	0.2
1	0.1	0.3	0.2

Table 3-2: The transition probabilities of primary inputs of the circuit in example 3.4

We calculate the BAM data structure at each node level by level as follows:

<u>Level 0</u> The BAM data structure for the primary inputs are set as follows:

$P(X^\delta)$ is the information we sampled.

If $(X=Y)$, $P(X^\delta[Y^\gamma]) = \begin{cases} 1,(\delta=\gamma) \\ 0,(\delta\neq\gamma) \end{cases}$, $(X,Y=A,B,C;\quad \delta=0,F,R,1)$

else $P(X^\delta[Y^\gamma]) = P(X^\delta)$.

e.g. $P(A^1[A^1]) = 1, P(A^1[A^0]) = 0,$

$P(A^F[B^R]) = P(A^F) = 0.3$

<u>Level 1</u> The BAM data structure for the node D and node E are set as follows:

According to the logic composition table, we could get the FR-Matrix of node D from FR-Matrices of node A and node B.

$$
\begin{aligned}
&P(D^0) = P(A^0)\times P(B^0) + P(A^0)\times P(B^F) + P(A^0)\times P(B^R) + P(A^0)\times P(B^1) \\
&+P(A^1)\times P(B^0) + P(A^F)\times P(B^0) + P(A^R)\times P(B^0) + P(A^R)\times P(B^F) + P(A^F)\times P(B^R) \\
&= 0.1\times(1) + 0.2\times(1-0.1) + 0.5\times 0.4 + 0.3\times 0.1 \\
&= 0.51 \\
&P(D^F) = P(A^1)\times P(B^F) + P(A^F)\times P(B^F) + P(A^F)\times P(B^1) \\
&= 0.1\times 0.4 + 0.3\times 0.4 + 0.3\times 0.3 \\
&= 0.25 \\
&P(D^R) = P(A^1)\times P(B^R) + P(A^R)\times P(B^R) + P(A^R)\times P(B^1) \\
&= 0.1\times 0.1 + 0.5\times 0.1 + 0.5\times 0.3 \\
&= 0.21
\end{aligned}
$$

$P(D^1) = P(A^1) \times P(B^1) = 0.1 \times 0.3 = 0.03$

The cofactor transition relation of D could be gotten from logic composition table too.

e.g.

$P(D^1[A^1]) = P(A^1[A^1]]) \times P(B^1[A^1]]) = 1 \times 0.3 = 0.3$

$$\begin{aligned}
&P(D^R[A^1]) = P(A^1[A^1]) \times P(B^R[A^1]) + P(A^R[A^1]) \times P(B^R[A^1]) + P(A^R[A^1]) \times P(B^1[A^1]) \\
&= 1 \times 0.1 + 0 \times 0.1 + 0 \times 0.3 \\
&= 0.1
\end{aligned}$$

We could get others cofactor transition relation by the same way.

BAM data structure for node E is obtained similarly.

<u>Level 2</u> In the last level of the reconvergent, we do not propagate the whole BAM data structure again, in place of, we only calculate the switching propagate of the convergent node F using BAM data structure of node in the former level.

We also use logic composition table in the calculation of switching propagation, but without multiplying two switching immediately we use equation 3.6 to calculate.

e.g.

$$\begin{aligned}
&P(F^1) \\
&\simeq P(D^1) \times P(E^1) + \sum_{X=A,B,C} \sum_{\alpha<\beta} P(X^\alpha)P(X^\beta) \times (P(D^1[X^\alpha]) - P(D^1[X^\beta])) \times (P(E^1[X^\alpha]) - P(E^1[X^\beta])) \\
&= 0.03 \times 0.06 + \sum_{\alpha<\beta} P(X^\alpha)P(X^\beta) \times (P(D^1[B^\alpha]) - P(D^1[B^\beta])) \times (P(E^1[B^\alpha]) - P(E^1[B^\beta])) \\
&= 0.0018 + 0.0012 \\
&= 0.003
\end{aligned}$$

Others switching probabilities of node F could be obtained similarly.

3.5 Reconvergent Circuit in Limit Depth

It has been discussed ([1], [9], and [11]) that the correlation among the circuit would become weaker as the depth of the reconvergent circuit becomes deeper. Let's survey the equation 3.6 $P(A^{\delta})P(B^{\gamma})+\sum_{i=1}^{n}\sum_{\alpha<\beta} p_i^{\alpha} p_i^{\beta} \times (X_i^{\alpha}-X_i^{\beta})\times(Y_i^{\alpha}-Y_i^{\beta})$.

Terms $(X_i^{\alpha}-X_i^{\beta}) = P(A^{\delta}[x_i^{\alpha}]) - P(A^{\alpha}[x_i^{\beta}])$ represent the distance between the probability of the transition at node A when the transition of its primary input x_i is α and the probability of the transition at node A when the transition of its primary input x_i is β. Thus, when the depth of the reconvergent circuit is getting deeper, the correlation among the circuit becomes weaker too. When the correlation among the circuit becomes weaker, the distance $(X_i^{\alpha}-X_i^{\beta})$ will become smaller too, so dose $(Y_i^{\alpha}-Y_i^{\beta})$. Then the second item of equation 3.6 will becomes smaller and smaller. We get:

Lemma 3.3 *Let ℓ be the depth of the reconvergent circuit, when $\ell \to \infty$, then*

$$P(A^{\delta})P(B^{\gamma})+\sum_{i=1}^{n}\sum_{\alpha<\beta} P(x_i^{\alpha})P(x_i^{\beta})\times(P(A^{\delta}[x_i^{\alpha}])-P(A^{\delta}[x_i^{\beta}]))\times(P(B^{\gamma}[x_i^{\alpha}])-P(B^{\gamma}[x_i^{\beta}]))$$

will be approach to $P(A^{\delta})P(B^{\gamma})$. In actual, when the depth of the reconvergent circuit is larger enough, even if it is not approach to ∞, the equation above will degrade to two probabilities multiplied immediately.

By lemma 3.3, we could set a limit depth $\ell_{\lim}$ as we needed, and we only process the reconvergent circuit when its depth is less than the limit depth $\ell_{\lim}$.

3.6 Non-Zero Delay Model

Extending FR-Vector with BAM to the non-zero delay model, we divide the whole model into three parts:

- The propagation of switching probabilities in general case;
- The propagation of cofactor transition relation in reconvergent circuit without including the node it convergent;

- And the calculation of switching probability of the convergent node of the reconvergent circuit.

In first part, the propagation of switching probabilities in general case, it has already been defined in original FR-Vector. We could extend it into non-zero delay model with theorem 2.2 and theorem 2.3. Further more, we use the same concept to extend the second part, the propagation of cofactor transition relation in reconvergent circuit, into non-zero delay model. That is:

Theorem 3.5 : *Let N be the frame size, δ be the logic gate delay in terms of frames. Let CTR_{out}^{δ} (CTR_{in1}, CTR_{in2}) be the output correlation transition relation derived from a logic composition with two input correlation transition relations, CTR_{in1} and CTR_{in2}, for a logic gate with delay δ. Let CTR_{out}^{0} (CTR_{in1}, CTR_{in2}) be the output correlation transition relation derived from a logic operation with two input correlation transition relations, CTR_{in1} and CTR_{in2}, for a zero delay gate. Then*

$$CTR_{out}^{\delta} = [CTR_{out}^{0}]^{\delta},$$

where $[CTR]^{s}$ denotes the right-shifted correlation transition relation for s frames. In other words, the i_{th} ($i \le N-\delta$) state in CTR_{out}^{0} appears in $(i+\delta)_{th}$ state in CTR_{out}^{δ}. And the j_{th} frame $(1 \le j \le \delta)$ in CTR_{out}^{δ} will be filled with the first state in CTR_{out}^{0}.

The following theorem applied only when the multi-cycle operations are allowed.

Theorem 3.6 : *Let N be the frame size, r be the gate delay. Let CTR_{out}^{r} (CTR_{in1}, CTR_{in2}) be the output correlation transition relation from a logic* *composition of two input output correlation transition relation s,* CTR_{in1} *and* RFM_{in2} *for a gate with delay r. Let* $CTR^{0}_{out}\,(CTR_{in1}, CTR_{in2})$ *be the output correlation transition relation from a logic operation with two input correlation transition relations,* CTR_{in1} *and* CTR_{in2}*, for the same gate with a zero delay. Then*

$$CTR^{r}_{out} = [CTR^{0}_{out}]^{r},$$

where $[CTR]^{r}$ *denotes the rotated correlation transition relation for r frames. In other words, the* i_{th} *state in* CTR^{0}_{out} *appears in* $(((i+r-1)\%N)+1)_{th}$ *state in* CTR^{r}_{out} ●

As considering for the calculation of switching probability in a convergent node of the reconvergent circuit, we need to add the concept of real-delay Boolean function we discuss in section 2.5 . In equation 3.2

$$P(A^{\delta})P(B^{\gamma}) + \sum_{i=1}^{n}\sum_{\alpha<\beta} p_i^{\alpha} p_i^{\beta} \times (X_i^{\alpha} - X_i^{\beta}) \times (Y_i^{\alpha} - Y_i^{\beta}),$$

the scope of the first $\sum$, $(1 < i \le n)$, means that assumes the convergent node we are computing is Z, and $f(Z) = (x_1, x_2, \ldots, x_n)$ where x_i is the node in the supply set of this reconvergent circuit. Above equation considers the effect causes by each x_i will do to the transition at node Z. In zero-delay model, we could use this equation immediately as $f(Z_f) = (x_{1f}, x_{2f}, \ldots, x_{nf})$ for X_f denote the Boolean function of the output node Z in the fth frame, but in non-zero delay model, for the reason of switching probabilities of each x_i would be divide into a number of frames, and so do the switching probabilities of output node Z, the Boolean function would be rewrite as real-delay Boolean function, and the equation will become as below:

Theorem 3.7 : *In non-zero delay model, if we divide a clock cycle into* m *frames, and there is a convergent node* Z *whose input is* A *and* B*, then*

$$P(A_f^{\delta} \cap B_f^{\gamma})$$
$$\simeq P(A_f^{\delta})P(B_f^{\gamma}) +$$
$$\sum_{X \in rdbf(Z_f)} \sum_{\alpha<\beta} P(X^{\alpha})P(X^{\beta}) \times (P(A^{\delta}[X^{\alpha}]) - P(A^{\delta}[X^{\beta}])) \times (P(B^{\gamma}[X^{\alpha}]) - P(B^{\gamma}[X^{\beta}]))$$

(Equation 3-4)

where $rdbf(Z_t)$ *is the real-delay Boolean function of the convergent node* Z *in time* f *, and* $(0) \overset{def}{=} 0, (F) \overset{def}{=} 1, (R) \overset{def}{=} 2, (1) \overset{def}{=} 3,\ \ (\delta, \gamma, \alpha, \beta = 0, F, R, 1;\ f = 0, \cdots m.)$.

Indeed, the RDBF we use there is modified RDBF for there is only m frames in a clock cycle, but if **multi-cycle operation** is allowed, RDBF could not be the representation of supply set. Thus,

Definition 3.7 (Modified RDBF): *In FR-Vector with BAM, if a clock cycle is divided into m frames, and there is a convergent node* Z *and a support set* $\{x_1, x_2, \ldots, x_n\}$*, then the Modified RDBF of convergent node* Z *is a RDBF whose scope of the time t is* $1 < t \le m$*, and the variable in Modified RDBF is* $x_{i;f}$*, where* $1 < i \le n$ *and* $0 < f \le m-1$*. If it exists a variable* $x_{i;f}$ *whose* $f > m$ *,the variable* $x_{i;f}$ *would be reduce to* $x_{i;f'}$*,where* $f' = f \bmod m$.

Example 3.5 If we divide a clock cycle into 4 frames, then:

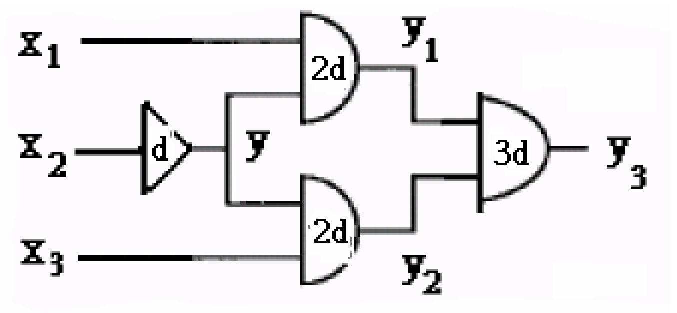


Figure 3-3: ***A reconvergent circuit with non-unit delay gate***

The original RDBF of node y_3 in **Figure 3-3** should be

$f(y_3) = F(x_1, x_2, x_3, d) = x_1(t-5d)x_2(t-6d)x_3(t-5d)$

In modified RDBF

$x_1(t-5d)$ will reduce as $x_{1;5d \bmod 4} = x_{1;1}$,

$x_2(t-6d)$ will reduce as $x_{2;6d \bmod 4} = x_{1;2}$,

$x_3(t-5d)$ will reduce as $x_{3;5d \bmod 4} = x_{3;1}$.

Thus, in our FR-Vector with BAM, for 4 frames a clock cycle, the modified RDBF of node y_3 is:

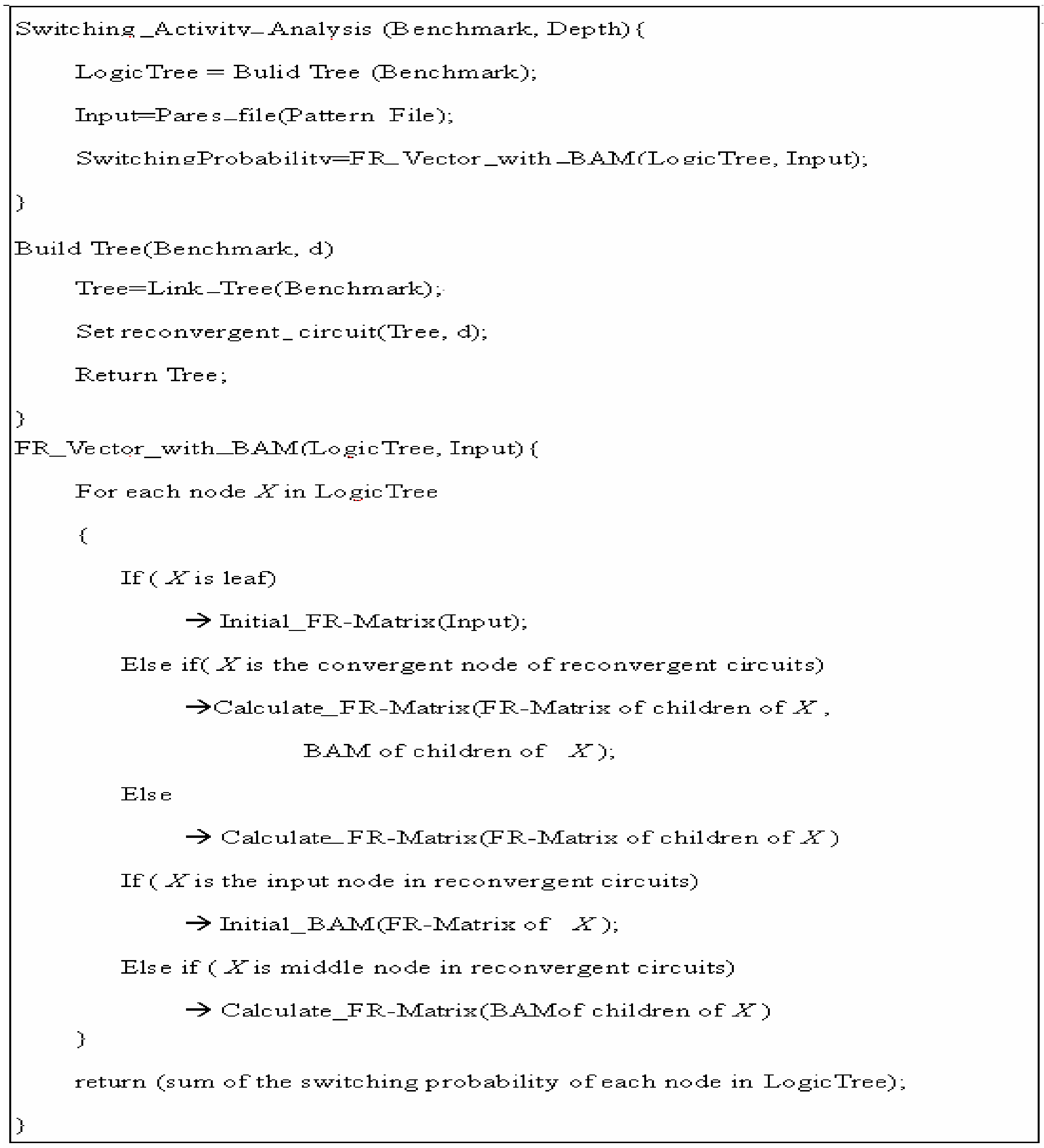
$$f^m(y_3) = f^m(x_1(t-5d)x_2(t-6d)x_3(t-5d))$$
$$= x_{1;1}x_{2;2}x_{3;1}$$

With theorem 3.7 and definition 3.7, we have:

Theorem 3.8 : *If there is a reconvergent circuit whose convergent node is Z. Through logic composition table, we can use BAM data structures of the immediate input nodes of Z and assisted with modified real-delay Boolean function (RDBF) and the concept of Taylor Expansion to calculate the switching probabilities of the convergent node Z.*

From Theorem 2.2, 2.3, and Theorem 3.8, 3.5, and 3.6, we could expand the FR-Vector with BAM as non-zero delay model.

3.7 Implementation



```
Switching_Activity_Analysis (Benchmark, Depth){
     LogicTree = Bulid Tree (Benchmark);
     Input=Pares_file(Pattern File);
     SwitchingProbability=FR_Vector_with_BAM(LogicTree, Input);
}
Build Tree(Benchmark, d)
     Tree=Link_Tree(Benchmark);
     Set reconvergent_circuit(Tree, d);
     Return Tree;
}
FR_Vector_with_BAM(LogicTree, Input){
     For each node X in LogicTree
     {
         If ( X is leaf)
               → Initial_FR-Matrix(Input);
         Else if( X is the convergent node of reconvergent circuits)
               →Calculate_FR-Matrix(FR-Matrix of children of X ,
                          BAM of children of  X );
         Else
               → Calculate_FR-Matrix(FR-Matrix of children of X )
         If ( X is the input node in reconvergent circuits)
               → Initial_BAM(FR-Matrix of  X );
         Else if ( X is middle node in reconvergent circuits)
               → Calculate_FR-Matrix(BAMof children of X )
     }
     return (sum of the switching probability of each node in LogicTree);
}
```

Figure 3-4: ***The pseudo-code of this paper***

First, we use the circuit with pairwise-inputs gates to build a logic tree of the circuit, involving linking each node together and setting reconvergent circuits. Then we parse the input pattern file whose data is produced in random. In final, we use FR-vector with BAM to propagate and calculate the switching probabilities in each node of the circuit. **Figure 3-4** is the pseudo code of the implementation.

The time complexity of *LinkTree*() will be $O(n)$, where n is the number of nodes in the circuit, and the time complexity of $Set\_reconvergent\_circuit()$ will be by far smaller than $FR\_Vector\_with\_BAM()$ because of the limit depth $Depth$ would limit it calculation. Thus, the time complexity of the whole program would be determined by $Parse\_file()$ and $FR\_Vetcor\_wit\_BAM()$, and

$$T\left(Parse\_file()\right) = \mathrm{O}(I \times f \times clk), \qquad \textit{(Equation 3-5)}$$

where I is the number of the input nodes, f is the number of frames in a clock cycle, and clk is the number of clock cycles.

In $FR\_Vector\_with\_BAM()$, each $Initial\_FR\_Matrix()$ needs $4 \times 4 \times f$ times calculations, where 4×4 is the state of composition of transition in node and it's input node (that is, the 4 state of δ multiply the 4 state of γ in $X^{\delta}[Y^{\gamma}]$), and $Initial\_FR\_Matrix()$ will be calculate at each input node, so

$$T\left(\text{All } Initial\_FR\_Matrix()\right) = \mathrm{O}(4 \times 4 \times f \times I) = \mathrm{O}(f \times I) \qquad \textit{(Equation 3-6)}$$

And as considering for $Initial\_BAM()$, for every reconvergent circuit, we should initial the BAM of every supply node in the reconvergent circuit. For each initialization, it needs $4 \times 4 \times f \times I^{i}_{con}$ times calculation, where I^{i}_{con} is the inputs number of the $i-th$ reconvergent circuits, so

$$T\left(\text{All } Initial\_BAM()\right) = \mathrm{O}(\sum_{i=1}^{c} I^{i}_{con} \times 4 \times 4 \times f \times I^{i}_{con}) = O(f \times \sum_{i=1}^{c} {I^{i}_{con}}^{2}),$$

where c is the number of reconvergent circuit in the circuits.

The input number of each reconvergent circuit will be bounded in $m \times \ell - 1$, where m is the maximum input number of gates, and ℓ is the depth we set(ℓ will be usually small, usually 3 to 5), so

$$T\left(\text{All } Initial\_BAM()\right) = (f \times \sum_{i=1}^{c} {I^{i}_{con}}^{2}) = \mathrm{O}(f \times c \times m^{2} \times \ell^{2}). \qquad \textit{(Equation 3-7)}$$

To $Calculate\_BAM()$, for every reconvergent circuit, we should calculate the cofactor transition relation of every node in the reconvergent circuit, beside supply and convergent node. For each of these nodes, it needs $4\times 4\times f\times I^{i}_{con}$ times calculation, so

$$T\left(\text{All } Calculate\_BAM()\right)=\mathrm{O}(\sum_{i=1}^{c} n^{i}_{con}\, 4\times 4\times f\times I^{i}_{con})=\mathrm{O}(f\times\sum_{i=1}^{c} n^{i}_{con}\times I^{i}_{con}),$$

n^{i}_{con} is the number of nodes in the $i-th$ reconvergent circuits beside supply and convergent nodes.

For n^{i}_{con} will be bounded in $\mathrm{O}(m\times\ell)$, so

$$T\left(\text{All } Calculate\_BAM()\right)=\mathrm{O}(f\times\sum_{i=1}^{c} n^{i}_{con}\times I^{i}_{con})=\mathrm{O}(f\times c\times m^{2}\times\ell^{2}),$$

(Equation 3-8)

$Calculate\_FR\_Matrix()$ in non-convergent node needs $4\times 4\times f$ times calculation, and it will be calculated in all node of the circuit beside convergent node:

$$\begin{aligned}&T(\text{All } Calculate\_FR\_Matrix() \text{ in non-convergent node})\\&=\mathrm{O}(4\times 4\times f\times n)\\&=\mathrm{O}(f\times n)\end{aligned}$$, *(Equation 3-9)*

By equation 3.7, it takes $f\times I^{i}_{rdbf}\times 7$ times calculations in $Calculate\_FR\_Matrix()$ of the convergent node, where I^{i}_{rdbf} is the variable number in the RDBF of the $i-th$ reconvergent circuit, and 7 for $\sum_{\alpha<\beta}$ in equation 3.7. Thus,

$$T(\text{All } Calculate\_FR\_Matrix() \text{ in convergent node}) = \mathrm{O}(\sum_{i=1}^{c} f\times I^{i}_{rdbf}\times 7)=\mathrm{O}(f\times\sum_{i=1}^{c} I^{i}_{rdbf})$$

For the reason that each node in supply set of a reconvergent circuit, it could produce ℓ variables in the RBDF it belonged, so I^{i}_{rdbf} will be bounded in $I^{i}_{con}\times\ell$, so

$$\begin{aligned}&T(\text{All } Calculate\_FR\_Matrix() \text{ in convergent node})\\&=\mathrm{O}(f\times\sum_{i=1}^{c} I^{i}_{rdbf})=\mathrm{O}(f\times\sum_{i=1}^{c} I^{i}_{con}\times\ell)=\mathrm{O}(f\times\ell\times\sum_{i=1}^{c} I^{i}_{con})\\&=\mathrm{O}(f\times c\times\ell^{2})\end{aligned}$$ *(Equation 3-10)*

From equation 3.9-3.12, we get the time complexity of $FR\_Vector\_with\_BAM()$ is

$$\mathrm{O}(f \times I + f \times c \times m^2 \times \ell^2 + f \times n) = \mathrm{O}(f \times n + f \times c \times m^2 \times \ell^2)$$

From above, we could get:

Lemma 3.4 *Using FR-Vector with BAM to analyze switching activities, the time complexity would be* $\mathrm{O}(f \times n + f \times c \times m^2 \times \ell^2)$. *Further more, if we take the time which used to sampling signals into account, the time complexity will be* $\mathrm{O}(f \times n + f \times c \times m^2 \times \ell^2 + I \times f \times clk)$, *where f represent a clock is divided into f frames, n is the number of nodes in the circuits, c is the number of reconvergent circuits, m is the maximum input number of gates in the circuit,* ℓ *is the limit depth we set, I is the input number of the circuit, and clk is the number of clock we sampling signals.*

This time complexity shows when the gate counts grows larger, $f \times n$ will become larger and larger too, then we'll get $\mathrm{O}(f \times n + f \times c \times \ell^2) = \mathrm{O}(f \times n)$. However if we take the time of sampling signals into account, as the number of sampling clock grows lager (maybe ten of thousands or more), this time complexity will become $\mathrm{O}(I \times f \times clk)$.

Chapter 4 Experimental Results

The proposed switching activities analyzing method is implemented by JAVA language, for its flexibility to various environment. To prove the accuracy of the proposed method, we compare our method with simulation results and FR-Vector. The former one could tell us the real switching number which would be gotten according to the given input pattern and benchmark, and the latter shows how much we could improve the accuracy than the original FR-Vector. We get the simulation results from the commercial tool, Cadence NC-VHDL 5.0, as a counterpart. All of these experiments are executed in a Linux-x86 computer. We simulate experiments based on 18 combinational circuits from MCNC [14].

In the first experiment, we tested on 18 MCNC benchmark circuits shown in the first column of Table 4.1. We generate 1000 random vectors for all inputs in each benchmark circuits, the frame number of a clock cycle is 20, and every gate delay is 1 frame. The second column in **Table 4-1** is the simulated result gotten by NCVHDL , the third to the fourth column is the analysis result from the FR-Vector and FR-Vector with BAM, where the estimated numbers of switching activities in FR-Vector are shown in the third column; the forth column is defined as the error percentage of FR-Vector over NC-VHDL; the estimated numbers of switching activities in our method are shown in the fifth column, and the last column is defined as the error percentage of our method over NC-VHDL . The row "avg" represent the average of the error percentage, and "W-avg" is weighted average which is gotten by equation $\left(\sum_{i=1}^{n} E_i \times G_i\right)/n$, where E_i is the error percentage of circuit i in experiment, and G_i is the gate count of circuit i, and n is the number of circuits in experiment. "W-avg" shows the average error percentage in gate. Because in normal circuit there would be no glitch input, thus we control the input generator and make every input pattern to have at most one switching in a clock cycle.

circuit	#switching (logic sim.)	#switching (FR)	FRM error(%)	#switching (F-B)	F-B error(%)
b1	11509	10783	6.31	10504	8.73
C17	3768	3718	1.33	3656	2.97
c8	151074	158345	4.81	151838	0.51
cht	75046	82600	10.07	76205	1.54
cm138a	7676	7441	3.06	7441	3.06
cm150a	55311	61587	11.35	56395	1.96
cm152a	14287	14452	1.15	14597	2.17
cm162a	29718	30385	2.24	29947	0.77
cm163a	30432	30378	0.18	29505	3.04
cm42a	8723	8743	0.23	7829	10.23
cm82a	15801	14541	7.97	14021	11.23
cm85a	25555	26558	3.92	24953	2.36
cmb	17230	21321	23.74	19511	13.24
count	53781	51343	4.53	51321	4.57
cu	31490	31532	0.13	31033	1.45
pm1	32168	34464	7.14	33206	3.23
sct	98575	103872	5.37	100568	2.02
tcon	27165	28830	6.13	27461	1.09
avg			5.54		4.12
W-avg		6.09		2.78	

Table 4-1: Switching activity comparisons between NC-VHDL logic simulator, FRM model and FR-Vector with BAM based on MCNC benchmark circuits with no glitch input.

In first experiment, we could see that FR-vector improve the accuracy of most circuits. Additionally, in original FR-Vector, the maximum percentage of error happens in "*cmb*", which could up to 23.74%. In the same circuit, FR-Vector with BAM improves the accuracy greatly to 13.24 %, and it is the highest one in FR-Vector with BAM too. As for averages of error percentage, it also decreases from 5.54 to 4.17. Further more, if we consider the "weight-averages" which shows the average error percentage in each gate, they further decrease from 6.09 to 2.78.

Lemma 5.1 *The error percentage of FR-Vector over the real simulation is*

$$|error| = \left| \frac{sw_{FR} + C_{BAM} - sw_{real}}{sw_{real}} \right| ,$$

where sw_{FR} *and* sw_{real} *represent the switching number estimated by FR-Vector and the real switching number,* C_{BAM} *is the value that BAM correct.*

Indeed, considering for signal correlation, we should have:

Lemma 5.2 $sw_{FR} - d_{FR} + C_{BAM} - d_{BAM} = sw_{real}$*, where* d_{FR} *and* d_{BAM} *is the distance causes by signal correlation.*

In FR-Vector with BAM, we wish we could get closer to $d_{FR} = C_{BAM}$. Thus, if we could ignore d_{BAM} , we could get $sw_{FR} - d_{FR} + C_{BAM} = sw_{real}$. However, in $d_{FR} = d_{con} + d_{input}$ where d_{con} is the distance caused by convergent and d_{input} is the distance caused by input spatial correlation, BAM could only solve d_{con}, and ignore d_{input} . Even in BAM itself, d_{BAM} would arise for input spatial correlation too. In some case, as small circuit for it low distance from input to output, the effect of d_{input} would be more serious. We could see this phenomenon happens in circuit — "*b1*","*cm42a*", and "*cm82a*". Moreover, $sw_{FR} \cong sw_{real}$ might happen as d_{con} approach to d_{input} occasionally. Thus, by lemma 5.2 we could know as $sw_{FR} \cong sw_{real}$, the switching number analyzed by FR-Vector with BAM will be $sw_{FR} + C_{BAM} = sw_{real} + C_{BAM} \neq sw_{real}$. We could see this phenomenon happens in circuit —”*c17*”,“*cm152*”,”*cm163a*”,”*cm42a*”, and “*cu*”.

In the third experiment, we examine the most four complex circuits with 100000 input vectors. The derived CPU time for NC-VHDL simulation, FR-Vector and our methods are listed in **Table 4-2** to **Table 4-4**, respectively. **Table 4-5** shows the speed-ups of our estimation method against the NC-VHDL simulation. We could see that the time cost by FR-Vector with BAM is close to FR-Vector but it is by far faster than the simulate result gotten from NC-VHDL.

NC-VHDL(seconds)						
	1	2	3	4	5	average
c8	62.50	58.30	55.50	56.80	60.40	58.70
cht	30.50	30.30	30.40	30.00	29.90	30.22
count	23.10	23.90	23.60	23.50	22.90	23.40
sct	33.00	32.10	32.60	32.20	32.00	32.38

Table 4-2: The CPU Time for NC-VHDL simulation

FR-Vector(seconds)						
	1	2	3	4	5	average
c8	1.515	1.437	1.480	1.533	1.521	1.497
cht	1.763	1.703	1.699	1.736	1.740	1.728
count	1.701	1.667	1.724	1.652	1.687	1.686
sct	1.142	1.166	1.162	1.193	1.113	1.155

Table 4-3: The CPU Time for FR-Vector simulation

FR-Vector with BAM(seconds)						
	1	2	3	4	5	average
c8	1.783	1.750	1.708	1.687	1.773	1.740
cht	1.935	1.955	1.925	1.912	1.947	1.935
count	1.873	1.806	1.865	1.822	1.781	1.829
sct	1.328	1.349	1.341	1.347	1.347	1.342

Table 4-4: The CPU Time for FR-Vector with BAM simulation

	Inputs	Outputs	Gates	FR	FR-BAM	NC-VHDL	Speed-up (NC-VHDL/ FR-BAM)
c8	28	18	316	1.50	1.740	58.70	33.736
cht	31	23	163	1.73	1.935	30.22	15.618
count	31	16	112	1.69	1.829	23.40	12.794
sct	19	15	199	1.16	1.342	32.38	24.128

Table 4-5: The comparison with FR-Vector and with NC-VHDL.

Finally, we compare FR-Vector and FR-Vector with BAM with other techniques. These techniques include the most basic and earliest technique, “signal probability” [16], and the other technique is “transition density” [17]. However, these two techniques doesn’t process unexpected transition, glitch. To recover glitches, we use G-Vector [12] to find these unexpected transitions in circuit, and add these glitches with switch number computed by “signal probability” or “transition density” to represent the estimate value. The result is listed in **Table 4-6**, the experimental result of “signal probability” is in the 7th and the 8th column, and the last two columns is the result of “transition density”. From this experiment, we could see the switch number gotten from “signal probability” and “transition density” is very unaccurat comparing to FR-Vector and our method. This is because “signal probability” try to use signal probability to compute switching numbers in circuit, however the switching activity in circuit could be very different in the same signal probability. “Transition density” adds the information about how many transition in a time unit to each gate in the circuit, and this information is gotten from the signal probability and transition density of the immediate input of the gate itself. Though it indeed improves the error percentage, it still based on signal probability. In FR-Vector and our method, we use switching probability to analyze switching activity. This conforms to real situation substantially, and this is one of the reasons why we take FR-Vector as out basic model.

circuit	#switching (logic sim.)	#switching (FRM)	error (FRM)%	#switching (F.B)	error (F.B)%	#switching (Sig.+G)	error (Sig.+G)%	#switching (T.D+G)	error (T.D+G)%
b1	11509	10783	6.31	10504	8.73	8095	29.78	11611	0.89
C17	3768	3718	1.33	3656	2.97	1212	67.83	1887	49.92
c8	151074	158345	4.81	151838	0.51	103180	31.7	177586	17.55
cht	75046	82600	10.07	76205	1.54	52521	30.01	85200	13.53
cm138a	7676	7441	3.06	7441	3.06	5161	32.76	7917	3.14
cm150a	55311	61587	11.35	56395	1.96	32426	41.38	66267	19.81
cm152a	14287	14452	1.15	14597	2.17	10250	28.26	16190	13.32
cm162a	29718	30385	2.24	29947	0.77	19848	33.21	38668	30.12
cm163a	30432	30378	0.18	29505	3.04	15716	48.36	28186	7.38
cm42a	8723	8743	0.23	7829	10.23	8493	2.64	12044	38.07
cm82a	15801	14541	7.97	14021	11.23	9475	40.04	15392	2.59
cm85a	25555	26558	3.92	24953	2.36	18192	28.81	31554	23.47
cmb	17230	21321	23.74	19511	13.24	22019	27.79	35722	107.32
count	53781	51343	4.53	51321	4.57	44798	16.7	72563	34.92
cu	31490	31532	0.13	31033	1.45	30055	4.2	44133	40.15
pm1	32168	34464	7.14	33206	3.23	28665	10.89	41735	29.74
sct	98575	103872	5.37	100568	2.02	70435	28.55	108871	10.44
tcon	27165	28830	6.13	27461	1.09	24979	8.05	33446	23.12
avg			5.54		4.12		28.39		25.86
w-avg			6.09		2.77		27.21		23.94

Table 4-6: Comparison among different technique.

Chapter 5 Conclusions and Future Works

In this research, we propose modified FR-Vector model, named FR-Vector with BAM which combined the advantage of FR-Vector — glitch processing in non-zero delay model, and the advantage of Boolean approximation method — the ability of processing correlation among convergent circuit. This model could analyze the switching activities in combinational circuit in probabilistic way with few data.

We have solved the correlation due to temporal dependence (using switching probability) and spatial correlation due to internal spatial dependence (using BAM to solve reconvergent circuit). Though it still has input spatial correlations, we have successively improved the error percentage in each gate from 5.05 to 2.27 and from 6.09 to 2.78 which represents input pattern with glitches and without glitches respectively. The peak value of error percentage also decreases from 23.74 to 13.24 %. In time issue, for we use the concept of Taylor expansion to approximate the spatial correlation, we do not need to build OBDD of the whole circuit which adopted in most techniques. Thus, the time spend by FR-Vector with BAM is closed to the original FR-Vector, and compare to the simulation time of NC-VHDL, it could up to 33.73 times faster. Indeed, the larger the circuit is or the more input patterns there is, we will get the more speed-up for the time is depend on the size of circuits.

As mentioned before, there is still something which could be improved in FR-Vector with BAM, that is, the spatial dependency among inputs. Thought in larger circuit (may be long depth or a big number of gates), the input correlations could only do little affect. But in special case, such as small circuits or circuits with specific input patterns, it still would be a serious problem. For circuits without specific input patterns, we could simulate them in various input patterns to eliminate the error caused by input spatial correlations, but for circuits with specific input patterns, we could only find out some way to process the dependency for controlling the error.

Another point could be improved is the way represent signal transition or the way we propagate the BAM data structure. In probabilistic calculation, the most correct consequence would appear when the probabilities in each state are balance. However, in FR-Vector, the probability of transition F and R would be by far smaller than the probability of transition 1 and 0 in no doubt, and result in an inaccuracy. So we should find a new way to represent the transition probabilities, or we should find a method that could calculate diverge values of probabilities without losing accuracy.

Reference:

[1] Radu Marculescu, Diana Marculescu, and Massoud Pedram, “Probabilistic Model of Dependencies During Switching Activity Analysis”, *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, Volume: 17, Issue: 2, Feb. 1998 Pages:73 – 83.

[2] Farid N. Najm, Member, IEEE,“A survey of Power Estimation Techniques in VLSI Circuit,” *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on* , Volume: 2, Issue: 4, Dec. 1994 Pages:446 – 455.

[3] A. Shen, A. Ghosh, S. Devadas, and K. Keutzer, “On average power dissipation and random pattern testability of CMOS combinational logic networks,” *Computer-Aided Design, 1992. ICCAD-92. Digest of Technical Papers, 1992 IEEE/ACM International Conference on*, 8-12 Nov. 1992 Pages: 402 – 407.

[4] S. Ercolani, M. Favalli, M. Damiani, P. Olivo, and B. Ricco, “Testability Measures in Pseudorandom Testing,” *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, Volume: 11 , Issue: 6, June 1992 Pages:794 – 800.

[5] Landy Huang, Zheng-Lun Lin, Chan-Shian Huang, and Chang-Jin Chen, “FR-vector A new Model for Switching Activity Analysis”, *The 14th VLSI Design/ CAD Symposium, Session P2-14.*

[6] T. Uchino, F. Minami, T. Mitsuhashi, and N. Goto, “Switching Activity Analysis using Boolean Approximation Method,” *Computer-Aided Design, 1995. ICCAD-95. Digest of Technical Papers, 1995 IEEE/ACM International Conference on*, 5-9 Nov. 1995 Pages: 20 – 25.

[7] S. M. Kang, “Accurate Simulation of Power Dissipation in VLSI Circuits”, *Solid-State Circuits, IEEE Journal of*, Volume: 21, Issue: 5 , Oct 1986 Pages:889 – 891.

[8] Ashok K. Murugavel and N. Ranganathan, “Petri Net Modeling of Gate and Interconnect Delays for Power Estimation”, *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, Volume: 11, Issue: 5 , Oct. 2003 Pages:921 – 927.

[9] Jose C. Costa, Jose C. Monteiro, and Srinivas Devadas, “Switching Activity Estimation using Limited Depth Reconvergent Path Analysis”, *Low Power Electronics and Design, 1997. Proceedings, 1997 International Symposium on*, 18-20 Aug. 1997 Pages:184 – 189.

[10] S. Theoharis , G. Theodoridis , D. Soudris, and C. Goutis, A. Thanailakis "A fast and accurate delay dependent method for switching estimation of large combinational circuits," *Computers and Digital Techniques, IEE Proceedings*-, Volume: 147, Issue: 6, Nov. 2000 Pages:444 – 450.

[11] Jer Min Jou, Shung-Chih Chen, and Chih-Liang Wang, **"**Fast delay-dependent power estimation of large combinational circuits" *Circuits and Systems, 1998. ISCAS '98. Proceedings of the 1998 IEEE International Symposium on,* Volume: 6, 31 May-3 June 1998 Pages: 53 - 56 vol.6.

[12] Ki-Seok Chung, Tae-Whah Kim, and C.L Lin, "G-vector: a new model for glitch analysis," *ASIC/SOC Conference, 1999. Proceedings. Twelfth Annual IEEE International*, 15-18 Sept. 1999 Pages:159 – 162

[13] C. Ding, C. Tsui, and M. Pedram, "Gate-level power estimation using tagged probabilistic simulation", *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, Volume: 17 , Issue: 11 , Nov. 1998 Pages:1099 – 1107.

[14] MCNC, http://www.cbl.ncsu.edu

[15] Tan-Li Chou and Roy. K, "Estimation of Circuit Activity Considering Signal Correlations and Simultaneous Switching", *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on,* Volume: 15, Issue: 10 , Oct. 1996 Pages:1257 - 1265.

[16] K. P. Parker and E. J. McCluskey, "Probabilistic treatment of general combinational networks," *IEEE Transactions Comput. on*, Volume: C-24, June 1975 Pages: 668-670.

[17] F. Najm, "Transition density, a stochastic measure of activity in digital circuits," in *28th ACM/IEEE Design Automation Conference*, San Francisco, CA , June 17-21, 1991 Pages:644-649.