M.Tech.(Computer Science) Dissertation Series

Multi Cycle Droop Fault in Combinational Circuits

a dissertation submitted in partial fulfillment of the requirements for the M.Tech. (Computer Science) degree of the Indian Statistical Institute

By

Onkar N. Tiwari

Roll No. : CS0603

July 9^{th} 2008

under the supervision of

Dr. Susmita Sur-Kolay

Advanced Computing and Microelectronics Unit

Indian Statistical Institute

203, B.T. Road

Kolkata-700108

Indian Statistical Institute

Certificate of Approval

This is to certify that the thesis entitled **Multi Cycle Droop Fault in Combinatorial Circuits** submitted by Onkar N. Tiwari towards partial fulfillment for the degree of M. Tech. in Computer Science at Indian Statistical Institute, Kolkata, embodies the work done under my supervision.

Dr. Susmita Sur-Kolay
ACMU, ISI, Kolkata

Date:

Acknowledgment

Working on this topic has been a constant source of inspiration and a matter of immense joy for me. I present this topic with a hope that future works in this field will get a glimpse of problems in this area. Doing this project has been the most fulfilling work of my academic carrier as it has not only taught me about the finer details of this topic but also to work in a cooperative environment full of helping hands. I take pride in admitting that there are some people who have guided me and helped me through my entire course of work and without whose help I may not have been able to present this work before you. I express my gratitude to **Dr. Susmita Sur-Kolay**, Advanced Computing and Microelectronics Unit, ISI, Kolkata for her constant guidance and inspiration throughout this project. I sincerely extend my gratitude to **Prof. Bhargab B. Bhattacharya** and all faculty of ACMU for there motivation and cooperation to complete this work. I would like to thank **Mr. Abhijit Jas**, Intel Corp., Austin who has advised on possible ways to move ahead in this work. I would like to heartily thank **Mr. Debasis Mitra**, WBUT, Kolkata who has been a constant helping hand throughout this project. I would also like to express my gratitude to **Mr. Ashish Nigam** whose work in this field has been an inspiration for me.

Finally I would like to thank all of my colleagues, class mates, friends, family members and staff of ISI, Kolkata for there support and motivation to complete this project.

Onkar N. Tiwari

Abstract

The scale of integration in VLSI chips is soaring, thereby the power consumption per unit area and also per unit length of power supply lines is increasing phenomenally. As the logic gates in physical proximity switch simultaneously, a noticeable power drop, known as droop, occurs at the power via nearest to these gates due to high frequency of operation and inductive effects of the narrow grid lines. Further, this drop propagates along the power supply lines, exponentially decaying in both time and the distance from the origin of droop, and thus causing a power drop even a few cycles later at some distant via. This may result in timing faults at few logic gates at that via. These faults are termed as *multi cycle droop faults (MDF)*. Modeling of these faults and understanding their behavior is yet to be explored. In this dissertation, a new model of multi cycle droop faults is first proposed at gate level granularity. But for the sake of efficiency of test generation, this is approximated by a model at via level granularity. Finally, a simple ATPG based procedure to generate test patterns for these faults is presented along with preliminary results for few ISCAS85 benchmark circuits.

Contents

Chapter 1

Introduction

The scale of integration in *very large scale integration (VLSI)* chips is soaring because of the progress in technology. Today more than a billion transistors can be fabricated on a VLSI chip with operating frequency nearly 10 GHz (and even this scale is increasing day by day) thereby the power consumption per unit area and also per unit length of power supply lines is increasing phenomenally. The increase in power consumption density in VLSI chips along with decreasing threshold and power supply voltages has resulted in greater power supply noise and therefore different types of faults in the operation of these chips. One such fault is the Droop Fault. Droop faults originate due to power supply noise termed as *power supply droop* or simply *droop*. This dissertation report has addressed a more specific kind of droop called *mid frequency power droop (MFPD)* and *multi cycle droop fault (MDF)* which originates from MFPD.

1.1 Power Supply Droop

Today's VLSI chips, having very large number of components and high frequency of operation, tend to draw considerable amount of power during their operation. Therefore large transients in supply current can happen while the chip is in operation. These transients specially happen when there is a sudden activity in the chip. For example when chip is going from idle mode to active mode. This flow of large transient current causes a temporal and spatial voltage variation

in the power supply grid (Fig. 1.1). Such a variation in voltage at some particular location of power supply grid is termed as *power supply droop* or simply *droop*.

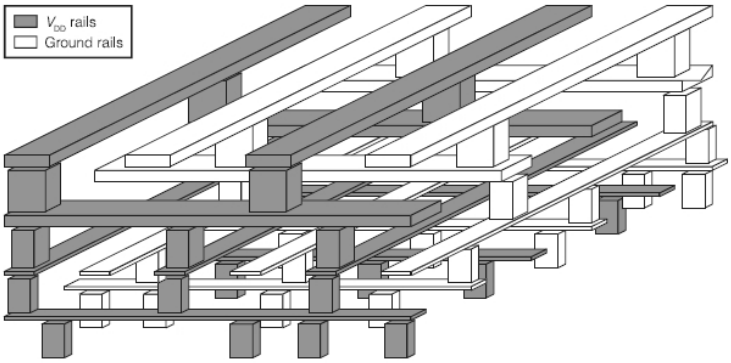

Figure 1.1: A typical 4 layer power supply grid with V_{DD} rails (shaded) and ground rails (white). Vias are the interconnect between two rails in consecutive layers. : courtesy [4]

Droop is broadly classified into three types according to their locality in both space and time.

1. *Low Frequency Power Droop (LFPD)*: This affects the whole grid after a few clock cycles.
2. *Mid Frequency Power Droop (MFPD)*: Localized to a small portion of power supply grid and effective for a few clock cycles.
3. *High Frequency Power Droop (HFPD)*: Highly localized and effective in the same cycle in which it is originated.

These types are discussed below.

1.1.1 Low frequency Power Droop (LFPD)

A schematic of power supply to the VLSI chips has been shown in Fig. 1.2. In deep sub-micron regime large currents from *voltage regulator module (VRM)* flow through the power supply grid of chips delivering current to the transistors which are connected to the grid through vias. Fig. 1.1 shows a typical 4 layer power supply grid in these chips with vias at the cross points of lines in two consecutive layers.

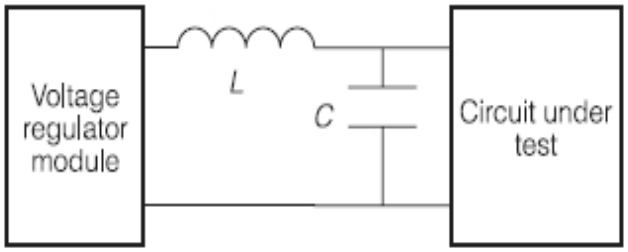

Figure 1.2: Circuit under test (CUT) connected to a voltage regulator model, including capacitance C and the interconnect's parasitic inductance L. : courtesy [4]

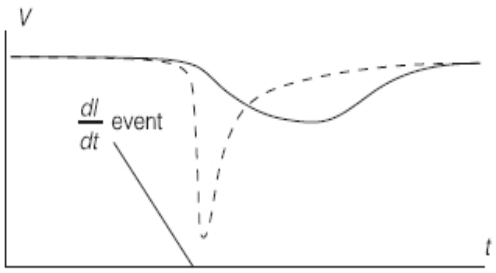

Figure 1.3: Voltage on the circuit under test (CUT) after a dI/dt event, without capacitor (dashed curve) and with capacitor (solid curve). : courtesy [4]

Fig. 1.2 illustrates the cause of LFPD. The circuit under test (CUT) is connected to the VRM. L denotes the parasitic inductance of the interconnect. Let the CUT demands a high current due to sudden increase in activity (as discussed before). Let us call this a dI/dt event. dI/dt event denotes an increased demand of dI amount of current in dt amount of time. Thus after a dI/dt event the power supply voltage V_{DD} to CUT will decrease by an amount $L(dI/dt)$. This amount of decrease in voltage is very high even for moderate values of current demanded ($\approx 10^2 A$ with $V_{DD} \approx 1V$) in a few clock cycles (1-10) in a high frequency ($\approx GHz$) range of operation of a circuit with the value of L well below 1nH.

The effect of such a dI/dt event on voltage supplied by VRM is shown in Fig. 1.3. Dashed curve shows the voltage difference between the connecting points of VRM and CUT in absence of the capacitor C. We can see a sharp decrease in the voltage when dI/dt event happens. This takes some time for VRM to recuperate and come to the original voltage level. The solid curve shows the effect on the voltage in presence of C. This is the real situation. The capacitor is

charged during the power on of VRM. When a dI/dt event happens, the fully charged capacitor C makes for the deficiency of current and in the process discharges itself. After several clock periods there is a situation when capacitor is discharged but VRM is not yet ready to deliver the full supply. This situation is shown as a dip albeit smaller in magnitude on the solid curve. We can see that this dip occurs after several clock cycles from the dI/dt event. This reduction in V_{DD}, which is felt by entire device (CUT), can slow the switching time of the gates in CUT. Such a transient reduction in power supply is known as LFPD.

When LFPD occurring in a circuit is sufficiently large in magnitude then it may be the case that certain gates have large enough delay causing them to fail to switch their state properly within one clock period. This gets revealed as a *stuck-at-fault (STF)*. Such a fault is known as *low frequency droop fault (LDF)*.

1.1.2 Mid frequency Power Droop (MFPD)

In contrast to LFPD, which affects V_{DD} over the entire package, MFPD is a localized phenomenon. MFPD affects only a small area of the chip and its effect last a few clock cycles only. We again see Fig. 1.1 in which a 4 layer power supply grid is shown with vias. Vias are junctions which deliver power to gates fabricated on the chip. In deep sub-micron regime typically 5-6 gates are connected to each via. We concentrate on a small segment of a power supply line typically containing 1-3 vias. When gates drawing power from this segment (i.e connected to the vias present in that segment) switch simultaneously in a considerable number within a clock period, a sudden increase in demand of power happens. This situation produces a negative spike in the supply current at that segment. Deficiency in current produced at that segment quickly tries to dissipate in time drawing current from the nearby region of the power supply grid. However in the today's VLSI chips where the transistor density is very high along with the high frequency of operation the on-die inductive effects get excited with a quick change in the current flowing through some segment of power supply grid as in this case of a sudden decrease in current supply. On-die inductive effects are comparatively smaller and highly localized in the deep sub-micron regime and decay very quickly. Such transient inductive effects along with a background of RC (with resistance R and decoupling capacitance (decap)

C) behavior shown by the chip plays an important role in the decay process of the current spike.

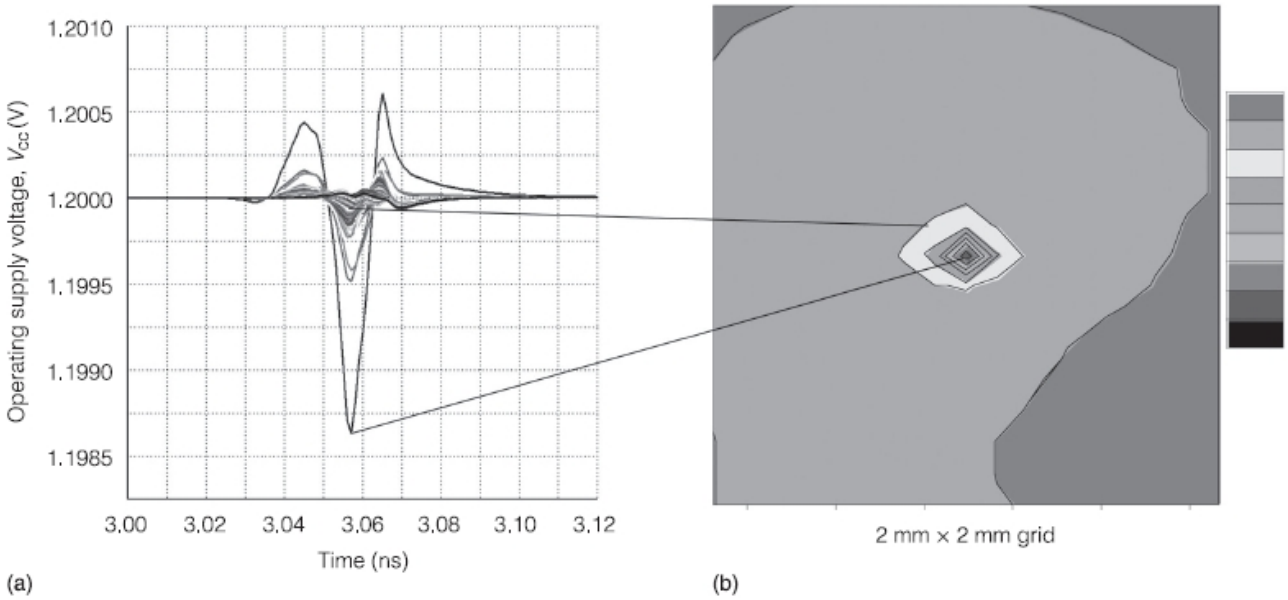

Figure 1.4: A fast current spike *(a)* injected into a power grid can excite localized inductive effects *(b)*, which quickly dissipates in time and space and becomes RC-only effects. (V_{cc} : nominal supply voltage). : courtesy [1]

Fig. 1.4 shows dissipation and the locality of such a current spike injected into a power grid segment. We can see an exponentially decreasing wave like phenomenon while the deficiency in voltage dissipates. This dissipation of deficiency takes a few clock cycles and also travels outwards from the point at which the spike was introduced. This is because the current deficiency is being quickly filled by the nearby power segments.

Transient inductive effects as discussed above are added by the action of decaps present in the circuit. Decaps act both globally and locally. Globally they correspond to the main resonance frequency of excitation currents while at higher frequencies they produce more localized effects. Fig. 1.5 shows the effect of such decaps. When a current spike (high frequency) is introduced the local effects of decaps are more effective. This effect along with the on-die inductive effect tries to slow down the recovery of current supply to the global resonant frequency current for a few clock cycle. Thus a deficient current supply can be felt at the same segment even after some clock cycles.

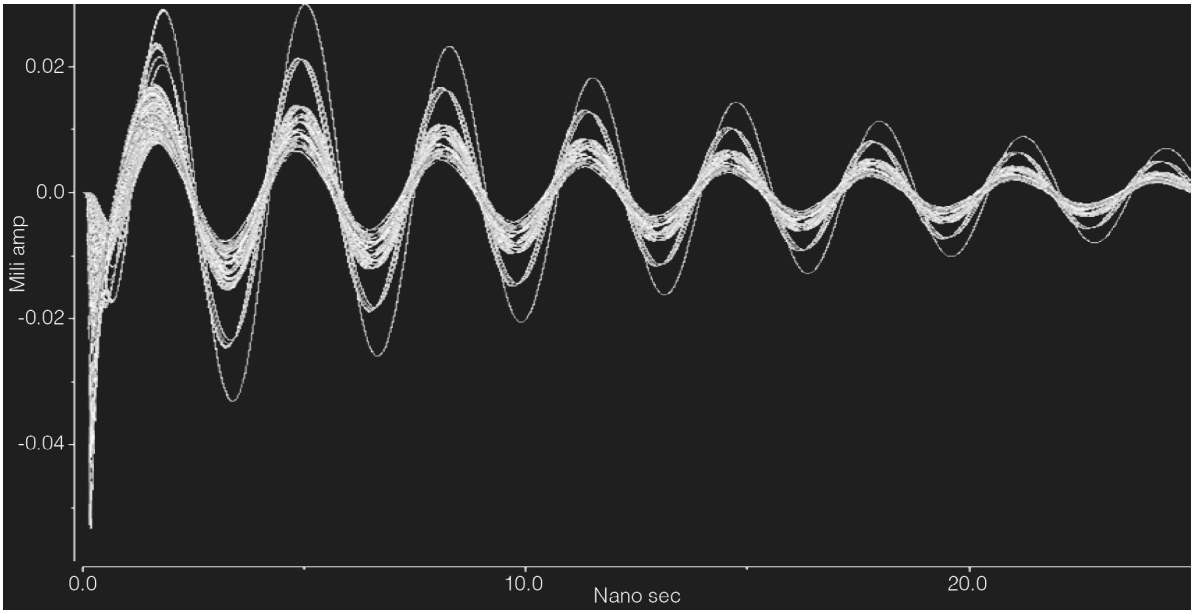

Figure 1.5: Transient current response of the nonuniform decaps. Initially, there are various frequencies greater than the global resonant frequency, but eventually all currents respond at the main resonant global frequency. : courtesy [1]

Fig. 1.6 shows the locality of mid frequency and low frequency currents on the chip when a current spike is introduced to a power segment. This shows that the deficiency of current supply is also felt by the nearby power segments though less in magnitude.

The above discussion shows that the droop produced by simultaneous switching of gates in physical proximity travels along the power grid and can be felt at some nearby positions after a few clock cycles but with less magnitude. Also the droop is felt at the same position for a few clock cycles. Such a droop is termed as *mid frequency power droop (MFPD))*. Again, as in the case of LFPD, MFPD also slows down the switching of the gates drawing power from the via where it is present. If the delay in switching is large enough for a clock period, which depends on the magnitude of the MFPD present at that via, then the gates may fail to switch within that clock period. This results in the faulty operation of the circuit. This is known as *mid frequency droop fault* or *multi cycle droop fault (MDF)*. It can be noted that LDF is also a multi cycle droop fault but for our purpose we denote *mid frequency droop fault* as *multi cycle droop fault (MDF)*.

Apart from this behavior there is also another difference between LFPD and MFPD. LFPD

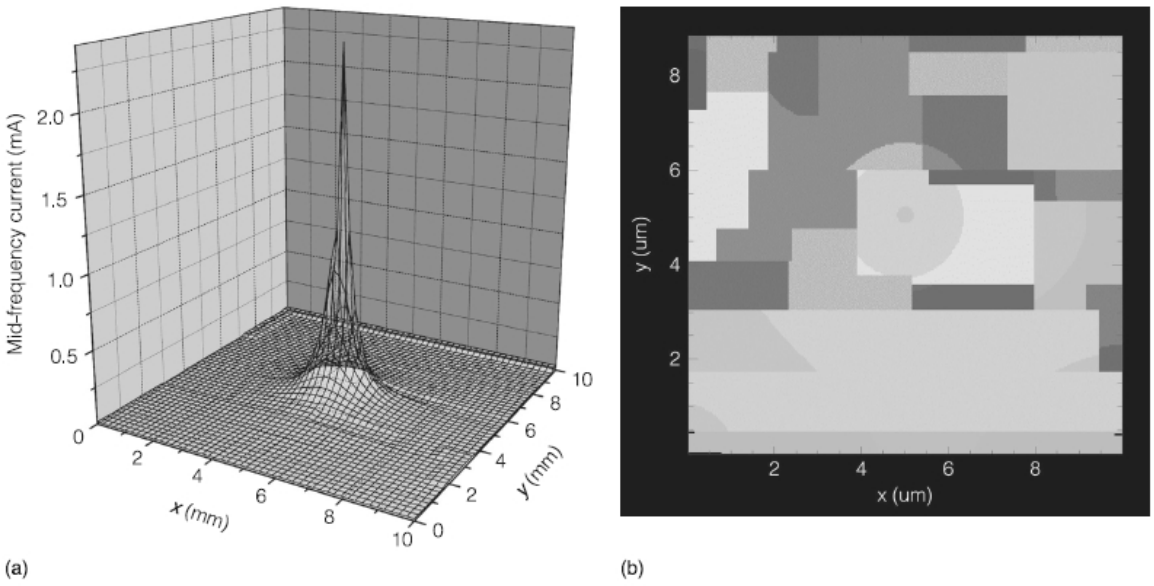

Figure 1.6: Locality of mid frequency (a) and low frequency (b) currents present in the first and second dips, respectively of the current amplitude curves in Fig. 1.5 : courtesy [1].

has highest magnitude after some time from the time of sudden increase in current demand (dI/dt event) i.e it increases and then decreases in time after the event as shown in Fig. 1.3. While in case of MFPD the highest magnitude of droop happens just when the sudden increase in demand of current happens. It decreases both spatially and temporally and eventually disappears. This is also one of the reasons why we termed *mid frequency droop fault* as *multi cycle droop fault (MDF).*

1.1.3 High frequency Power Droop (HFPD)

HFPD can be considered as a special case of MFPD. The cause of HFPD is same as MFPD. This classification is done only to capture the effect of droop and fault caused by it within only one clock cycle and at the same via where the sudden activity happens.The magnitude of MFPD within the first clock cycle and at the same location where it originates is highest. Therefore in VLSI chips with moderate transistor density often MFPD is of only one clock cycle length affecting the same locality where it originates. Thus notion of HFPD helps in formulating the faults due to MFPD within the same clock cycle.

Fig. 1.4 also depicts how a current spike injected in a circuit typically dissipates in space

and time. If the injected current spike consists of only very high frequency components then it dissipates very quickly (within one clock period) as depicted in Fig. 1.5. In this figure we can see that the high frequency components present in the first dip quickly dissipate within a few nano seconds. For VLSI chips with operating frequency of the order of GHz dissipation often happens within one clock cycle. Also such a droop is highly localized. Such droop is termed as *High frequency Power Droop (HFPD)*.

The faulty operation of the circuit occurring due to HFPD is termed *high frequency droop fault (HDF)* or *single cycle droop fault (SDF)*. The reason for the origin of the fault is same as that of fault due to MFPD or LFPD.

1.2 Previous Works

This field is still nascent as there are a few works reported in recent time. This is largely due to the fact that droop faults are being seen in chips with very high density of transistors with a very high frequency of operation. Advances in fabrication techniques promise production of such chips in large scale in the near future. But still we are on the verge of entering the domain where these chips will come into large scale use. Present day chips are experiencing a little effect due to droop.

The problem of *single cycle droop fault (SDF)* in combinatorial circuits was addressed in the work of Mitra et. al. in [2]. There SDF is modeled as *slow to rise (STR)* fault taking into account the switching of gates 0→1. The other way transitions have a similar modeling. The SDF gets revealed as *stuck-at-0 (s-a-0)* fault at the victim gates. A test vector pair is found which excites the HFPD on some victim gates and then propagates the effect of *s-a-0* fault on every victim gate at some primary output.

SDF in full scan circuits has been covered in [3] by Mitra et.al.. A similar modeling for full scan circuits as with combinatorial circuits has been done in this work. A methodology for test pattern generation in *launch on control (LOC)* environment in full scan circuits has been proposed and an *automatic test pattern generator (ATPG)* has been implemented for it.

While in [4] Ilia Polian et. al. have addressed the faults due to LFPD and HFPD both but not MFPD per se. They propose a method for ATPG for exciting the worst case droop by

combining the effects of LFPD and HFPD both and detecting these faults.

This thesis concentrates on MFPD and presents a model for it in VLSI chips. Here a model for MDF has been presented along with a prototype ATPG for exciting and detecting the MDF. This thesis presents the significance of MDF in the VLSI chips particularly the future chips.

1.3 Scope of this thesis

This thesis mainly concentrates on *mid frequency power droop (MFPD)* and the fault generated due to it i.e. *multi cycle droop fault (MDF)*. To study the behavior of MDF and to generate the test pattern for it we first need to model MFPD and MDF. The modeling of MFPD and MDF and test pattern generation for MDF is the main subject matter of this thesis.

This thesis presents a cumulative model for MFPD. Which assumes that MFPD at any place in circuit is accumulated by the excitation (sudden activity) of the nearby gates. MFPD, if sufficient enough, produces MDF at some gates connected to the target via where MFPD is present. The nature of a test sequence to excite and test MDF is presented in this work. Also the nature of an *automatic test pattern generator (ATPG)* for MDF has been discussed.

A test sequence is needed to excite the MFPD at some target via and then to propagate the effect of MDF to some primary output. A prototype algorithm for generation of such test sequence is also presented in this thesis. A tool called *multi cycle droop fault test generator (MDFTG)* has been implemented using the prototype algorithm and experiments are done on ISCAS85 bench mark circuits. These experiments demonstrate the significance of MDF in high performance VLSI chips.

1.4 Organization of this thesis

The rest of the chapters cover the modeling and test pattern generation for MDF. Chapter 2 builds the theoretical ground to understand MDF. This chapter presents the cumulative model for MFPD and then discusses how MDF occurs using this model. The nature of a test sequence for MDF has been discussed in chapter 2. This chapter also discusses the nature and the requirements for an ATPG for MDF.

Chapter 3 discusses about some practical aspects of MDF in today's chips. This chapter presents a prototype algorithm for test pattern generation for MDF. This prototype algorithm is a restricted version of the ATPG discussed in chapter 2. A tool namely *multi cycle droop fault test generator (MDFTG)* has been implemented based on this prototype algorithm. Chapter 3 also presents the experiments done using MDFTG and there results showing the significance of these faults.

Chapter 4 contains the details of the implementation of MDFTG. While chapter 5 presents the overall conclusion and the scope of future works in the field of test pattern generation for MDF.

Chapter 2

Model for Mid Frequency Droop Fault (MDF)

In order to model MDF, first we have to arrive at a model of MFPD. Here we first propose a model for MFPD which is essentially cumulative in nature. Origin of MFPD can be at a power segment (or via) as discussed in section 1.1.2. MFPD travels along the power grid to nearby power segments (or vias). Thus if we observe a droop at a via which is affected by MFPD then it is easy to see that this droop is present there due to nearby vias which are origins of MFPD. Thus a via can experience an accumulated MFPD in which each component comes from nearby vias which have produced an MFPD in an earlier clock period. Here we assume that total MFPD at any via T is the sum of MFPD's (present at T) from every via nearby. Hence the name of the model is *cumulative model for MFPD*. This model is presented here assuming gate level granularity i.e assuming that switching of only one gate is sufficient enough to produce a droop. But for practical reasons and for present day high performance chips this model can be easily modified to assume mid level granularity i.e assuming a group of gates instead of single gates. These groups can be formed from the gates which are in physical proximity and drawing power from same via or power segment. Next section presents the cumulative model in a formal manner.

2.1 Cumulative Model for Mid Frequency Power Droop (MFPD)

The proposed model is illustrated with an example. Fig. 2.1 shows a typical top-view schematic of power supply grid lines, orthogonal in alternate metal layers, with vias at the intersections. In the figure X and Y axes are also shown with the origin at any arbitrary via in the power grid so that all the vias which we consider for our discussion are in first quadrant. Each via is denoted as $V^{x,y}$ where x and y are the coordinates of the via with respect to the axes. Each via usually supplies power to five or six logic gates (in deep sub micron regime). Each gate connected to via $V^{x,y}$ is denoted as $g_i^{x,y}$ where x, y is the location of the via and i $(1 \leq i \leq n)$ is the gate number. Let us for sake of easy formulation assume that $n = 6$ for all vias i.e. 6 gates are connected to every via. This figure will serve as the example for explaining the model. Without loss of generality, let us assume only 0→1 transitions. The other way transitions can be handled similarly. Since, due to the presence of considerable amount of MFPD at any gate g a failure of switching of g happens, therefore the fault generated due to MFPD at any gate g can be effectively assumed as *stuck-at-0 (s-a-0)* fault at g.

Now to study the effect of droop we first choose a via called *target* via. A target via T is shown in Fig. 2.2, on which the effect of MFPD is to be observed. In our case we can choose via $V^{3,2}$ from Fig. 2.1 as the target via.

$$V^{3,2} \text{ is the target via } T.$$

Some of the gates connected to the target via T on which the effect of MFPD is likely to reveal itself as a *s-a-0* fault (while they try to switch 0→1), form a list called *victim list* [2]. Thus *victim list* contains all those gates which are vulnerable to switching failure if sufficient amount of droop is present on via T. We denote *victim list* by L. For our example we can take L as a subset of gates connected to via T. Let

$$L = \{g_1^{3,2}, g_2^{3,2}, g_4^{3,2}, g_5^{3,2}\}$$

Target via T experiences a droop originated from vias in its vicinity. Let some of the gates connected to a via $V^{x,y}$ in the vicinity of T simultaneously switch their state 0→1 within

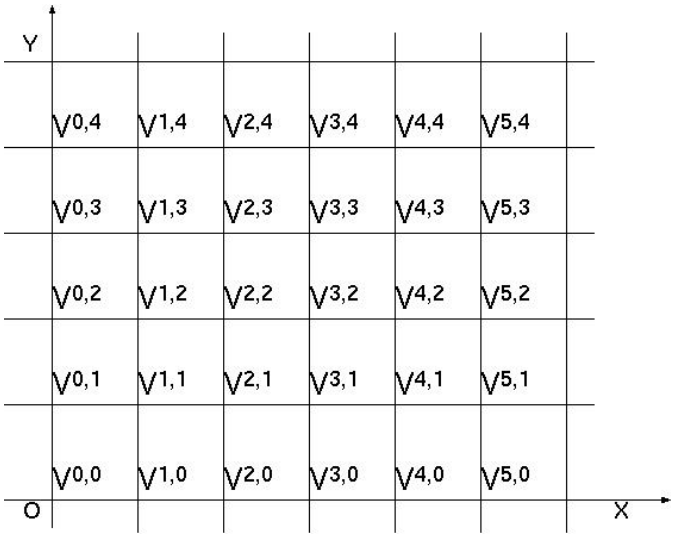

Figure 2.1: Top view of power supply grid with lines showing the metal lines which are orthogonal in alternate layers, vias are at the cross section of two orthogonal lines. X and Y axes are shown and each via is denoted as $V^{x,y}$ where x, y are the coordinates of the via with respect to the axes.

the clock cycle t. The droop formed by such simultaneous transition of states of gates starts traveling along the power grid in all directions (as shown in Fig. 1.4) ever decreasing in amplitude. Thus reaching a nearby via after some time. Let such droop produced at via $V^{x,y}$ first reaches via T after i clock cycles (i.e. at clock cycle $t + i$). Then we call $V^{x,y}$ is of type V_i. Such a droop affects via T not only within clock cycle $t + i$ but also within clock period $t + i + 1$, $t + i + 2$ and so on with ever decreasing amplitude and eventually dies out (Fig. 1.5).

Let us call a via V in the vicinity of via T of type V_i if the droop originated at via V, due to simultaneous 0→1 transition of the gates connected to V, reaches T after i clock periods.

Fact: *A via V_i which produces a droop at via T after i clock periods also produces a droop of smaller intensity after j $(j > i)$ clock cycles at T.*

This can be verified on referring to the figures Fig. 1.4 and Fig. 1.5.

Fact: *The location and the shape of boundary of the region around via T hosting vias of type V_0 depends upon the local transistor density and frequency of operation of the chip along with other factors. This is also true for the other via types.*

This can be verified by figures Fig. 1.4 and Fig. 1.6.

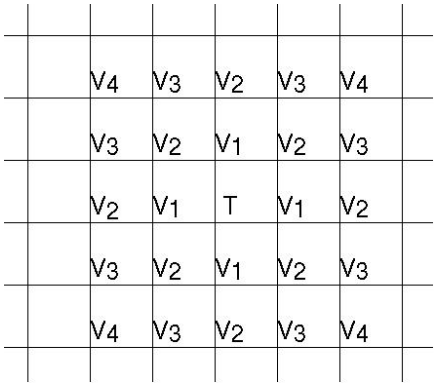

Figure 2.2: Power supply grid (top view) with vias at intersection points which are marked as T or V_i depending on their distance from target via T (note : via of type T is also of type V_0 and any via of type V_i is also of type V_j where $j > i$).

It can be easily seen that via T is itself of type V_0 since the droop produced at via T within clock cycle t affects the switching of gates connected to via T within clock cycle t itself also. Some vias nearest to T are also candidates of being typed V_0 since they are so near that droop originated at them within clock cycle t reaches at via T within t itself. This is because of the above stated fact. But for the sake of clarity it is shown in Fig. 2.2 that only via T is of type V_0. Thus

via $V^{3,2}(T)$ in Fig. 2.1 is of type V_0.

The vias denoted by V_1 should be nearest to T but farther than that of type V_0. This can be verified from Fig. 1.4. V_1 vias host gates at which simultaneous 0→1 transition (in a sizable number) produces a droop, which propagates along the power line ever decreasing in amplitude and reaches via T after one clock cycle. Similarly, droop produced at V_2 vias reaches T after two clock cycles, and so on. Fig. 2.2 shows the vias along with their types.

Note : *In Fig. 2.2 the via T is also of type V_0 and any via of type V_i is also of type V_j where $j > i$.*

This follows from the above fact. However it is not shown in the Fig. 2.2 for the sake of clarity.

Coming back to our example the Table 2.1 shows the type of some vias around $V^{3,2}$.

Table 2.1: Types of certain vias in Fig. 2.1

Type	***Via***
T, V_0	$V^{3,2}$
V_1	$V^{3,2}, V^{3,1}, V^{4,2}, V^{3,3}, V^{2,2}$
V_2	$V^{3,2}, V^{3,1}, V^{4,2}, V^{3,3}, V^{2,2}, V^{3,0}, V^{4,1}, V^{5,2}, V^{4,3}, V^{3,4}, V^{2,3}, V^{1,2}, V^{2,1}$

It can be noted that the droop originated at distant vias from the target via have less effect on the target via. So after a suitable distance (depending upon the chip) the droop effect can be safely assumed to be negligible. Similarly the droop effect decreases with time too. Thus we consider types V_i of vias with only small values of i and in a small neighborhood of target via T.

2.1.1 Accumulating the droop

Suppose some of the victim gates connected to via T have to switch $0 \rightarrow 1$ at clock period (say) t, and there was simultaneous switching of some of the gates at via type V_0 during clock period t, simultaneous switching of the gates connected to via type V_1 during clock period $t-1$, and so on. Due to the linear superposition of the droop inherited by via T up to the t^{th} clock period, reduced supply current is likely to cause a substantial delay in switching of victims connected to via T. This gets manifested as a *s-a-0* fault at some of these victim gates of T. In other words the droops from via type V_i gets accumulated at via T. Thus, this model is cumulative in the sense that it assumes that the total droop at the target via T is the sum of all the droops (present at T) caused by every gate (which has made a transition) in its vicinity at any of the previous clock cycles.

2.1.2 Formal model

The above discussion leads us to a few definitions as follows:

Let we call *aggression (excitation)* a event when a group of gates switch simultaneously $0 \rightarrow 1$.

An *aggressor* $\alpha_i(T)$ is a gate in the vicinity of T which causes a droop at via T after i

clock cycles. For example in Fig. 2.1 the gate $g_2^{3,1}$ is an aggressor of type $\alpha_1(T)$, since this is connected to a via of type V_1. A gate connected to a via of type V_i is an aggressor of type $\alpha_i(T)$.

Fact : *An aggressor $\alpha_i(T)$ is also of type $\alpha_j(T)$ for $j > i$.*

This is because of the same reason as a via of type V_i is also of type V_j for $j > i$.

The Table 2.2 shows some of the gates with their types.

Table 2.2: Types of certain (aggressor) gates in Fig. 2.1

Type	***Gate***
α_0	$g_1^{3,2}$,$g_3^{3,2}$,$g_4^{3,2}$,$g_6^{3,2}$
α_1	$g_1^{3,2}$,$g_2^{3,1}$,$g_1^{4,2}$,$g_6^{3,3}$
α_2	$g_1^{3,2}$,$g_3^{3,1}$,$g_4^{3,2}$,$g_6^{4,1}$

Let D be the least amount of droop at via T which can cause switching delay, in the gates in the victim list L for via T, large enough to be perceived as *s-a-0* fault. Let $\delta_i(g,T)$ denote the amount of droop caused at via T by an aggressor $\alpha_i(T)$ $(= g)$, then $\epsilon_i(g,T) = \delta_i(g,T)/D$ is the effectiveness of this droop caused by g on T.

$\lambda_i(T)$ denotes an aggressor list (set of gates) causing a droop at T after i clock periods. Thus $\lambda_i(T)$ is a set of gates such that each gate $g \in \lambda_i(T)$ is an aggressor of type $\alpha_i(T)$. The total effectiveness of $\lambda_i(T)$ is sum of effectiveness $(\epsilon_i(g,T))$ of each gate $g \in \lambda_i(T)$ or 0 if the list is empty. The effectiveness of $\lambda_i(T)$, denoted by $\xi(\lambda_i(T))$, is the worst case droop caused by the simultaneous transition of gates (aggression or excitation) belonging to it at via T. An exhaustive aggressor list, denoted by $\lambda_i^e(T)$, is the set of all gates in the vicinity of T which are of type $\alpha_i(T)$ for any $0 \leq i$. Therefore, any $\lambda_i(T)$ is a subset of $\lambda_i^e(T)$. For example (from Fig. 2.2):

$\lambda_0^e(T) = \{g_1^{3,2}, g_2^{3,2}, g_3^{3,2}, g_4^{3,2}, g_5^{3,2}, g_6^{3,2}\}$

While a $\lambda_0(T) = \{g_1^{3,2}, g_3^{3,2}, g_4^{3,2}, g_6^{3,2}\}$, which is a subset of $\lambda_0^e(T)$.

The effectiveness of $\lambda_0(T)$ is $\xi(\lambda_0(T)) = \epsilon_i(g_1^{3,2},T) + \epsilon_i(g_3^{3,2},T) + \epsilon_i(g_4^{3,2},T) + \epsilon_i(g_6^{3,2},T)$.

An *M_Aggression* for a target via T consists of aggressor lists which switched during any of the M previous clock cycles is defined as:

$M\_Ag(T) = \{\lambda_i(T) \mid 0 \leq i \leq M \text{ and } \lambda_M(T) \text{ is not empty}\}$

The total effectiveness of $M\_Ag(T)$, denoted by $\xi(M\_Ag(T))$, is the sum of the effectiveness of all the aggressor lists belonging to it. Once again this effectiveness is the worst case droop at the target via T produced by the aggression (excitation) of each $\lambda_i(T)$ belonging to it in their respective previous clock cycle.

An *M_Aggression* which includes all the gates producing a multi cycle droop at the target via T is called an *exhaustive M_Aggression* ($M\_Ag^e(T)$). In other words it is the set of exhaustive aggressor lists $\lambda_i^e(T)$ for all $0 \geq i \leq M$.

Example:

$4\_Ag^e(T) = \{\lambda_0^e(T), \lambda_1^e(T), \lambda_2^e(T), \lambda_3^e(T), \lambda_4^e(T)\}$

while a $4\_Ag(T) = \{\lambda_0(T), \lambda_1(T), \lambda_3(T), \lambda_4(T)\}$

The effectiveness of $4\_Ag(T)$ is $\xi(4\_Ag(T)) = \xi(\lambda_0(T) + \xi(\lambda_1(T) + \xi(\lambda_3(T) + \xi(\lambda_4(T)$.

2.2 Occurrence of Multi Cycle Droop Fault (MDF)

An *M_Aggression* is the set of gates which is a likely candidate to produce a droop and thus a *s-a-0* fault (if sufficient enough) on target via T. We can easily see that if each gate in an *M_Aggression* has an aggression in their respective previous clock cycle (depending on its aggressor type), then a droop appears at via T whose effectiveness is the effectiveness of this *M_Aggression* and this phenomenon is called the *aggression (excitation)* in the *M_Aggression*.

For example see the $4\_Aggression(4\_Ag(T))$ of the above example. Aggression (excitation) in *4_Aggression* means the following:

- All gates in $\lambda_4(T)$ were having a logical value '0' within clock cycle $t-5$ and have a logical value '1' within clock cycle $t-4$.
- All gates in $\lambda_3(T)$ were having a logical value '0' within clock cycle $t-4$ and have a logical value '1' within clock cycle $t-3$.
- All gates in $\lambda_1(T)$ were having a logical value '0' within clock cycle $t-2$ and have a logical value '1' within clock cycle $t-1$.

- All gates in $\lambda_0(T)$ were having a logical value '0' within clock cycle $t-1$ and have a logical value '1' within clock cycle t.

Aggression in this $4\_Aggression$ produces a droop at via T at clock cycle t. The effectiveness of this droop is $\xi(4\_Ag(T))$. If $\xi(4\_Ag(T)$ is more than or equal to 1 then the total droop at T exceeds D, i.e., the minimum amount of droop needed to produce such a switching delay that some of the gates connected to via T show *s-a-0* fault. Thus an MDF at any gate connected to via T occurs if there is an aggression in an $M\_Ag(T)$ whose effectiveness is more than or equal to 1.

Fact : *If an $M\_Aggression$ for target via T has a total effectiveness greater than or equal to 1, then only it is a likely candidate to produce an MDF at any victim of T.*

It is evident from our formulation of effectiveness.

2.2.1 Testing Multi Cycle Droop Fault (MDF) due to $M\_Aggression$

For generation of a MDF corresponding to a given $M\_Aggression$ on a given target via T, we need to produce aggression (excitation) in the given $M\_Aggression$ so that the worst case droop is produced on via T. MDF due to such droop at some of the gates connected to T is then propagated to some of the primary output of the circuit to be observed.

Mid Frequency Power Droop (MFPD) excitation

For the given $M\_Aggression$ ($M\_Ag(T)$) and a victim list V, the test pattern (TV) which excites droop (MFPD) of effectiveness $\xi(M\_Ag(T))$ at T at clock period t ($t > M$) will do the following:

When TV is applied to the circuit, the gates in the list $\lambda_i(T) \in M\_Ag(T)$, for each $0 \leq i \leq M$, undergo a $0 \rightarrow 1$ transition within clock period $t - i$.

Multi Cycle Droop Fault (MDF) excitation

The above section tells the necessary condition for producing the droop (MFPD) at the target via T produced by $M\_Ag(T)$. But to produce MDF we need to also force the transition in gates of victim list V. This can be done if the following properties satisfy:

The gates in victim list V are set to '0' within clock period $t-1$ and are set to '1' in clock period t.

This condition along with the MFPD excitation condition is the condition for MDF excitation as we can see that the droop is produced at clock period t and the gates in victim list are also forced to make 0→1 transition. If the droop will be sufficient ($\xi(M\_Ag(T)) \geq 1$) then the gates in victim list will fail to switch and thus MDF will occur.

Multi Cycle Droop Fault (MDF) propagation

We can see that at clock period $t-1$ all the gates in victim list V are set to '0' while within clock period t they try to set to '1'. But due to droop (MFPD) at via T they will remain at '0'. This may make the fault redundant (since multiple gates are stuck at '0'). Thus in essence the MDF propagation should detect (multiple) *s-a-0* fault at every gate in victim list V. Therefore to propagate the MDF, at a target via T, to primary outputs we need to apply a test vector at clock period t which should satisfy the following property:

all gates in the victim list are set to '1' (this comes from MDF excitation property) and this is a test vector for *s-a-0* fault at all the gates of victim list.

Test sequence

we can sum up all the above properties of a test sequence as below.

For a target via T along with its victim list V, an $M\_Aggression$ $M\_Ag(T)$ with a suitable value of M, a sequence of $M+2$ test vectors (TV) is required to detect an MDF at T. Fig. 2.3 shows the properties of such a test sequence. Such a sequence should have the following properties:

1. $TV[M+1]$ sets the gates in $\lambda_M(T)$ to '0'.

2. $TV[i]$ sets the gates in $\lambda_i(T)$ to '1' (if it is not empty) and simultaneously sets gates in $\lambda_{i-1}(T)$ to '0' (if it is not empty), or a random test vector (if both $\lambda_i(T)$ and $\lambda_{i-1}(T)$ are empty and $i \neq 1$), for all $1 \leq i \leq M$.

3. $TV[0]$ sets the gates in $\lambda_0(T)$ to '1', and also is a test vector to detect a *s-a-0* fault at all

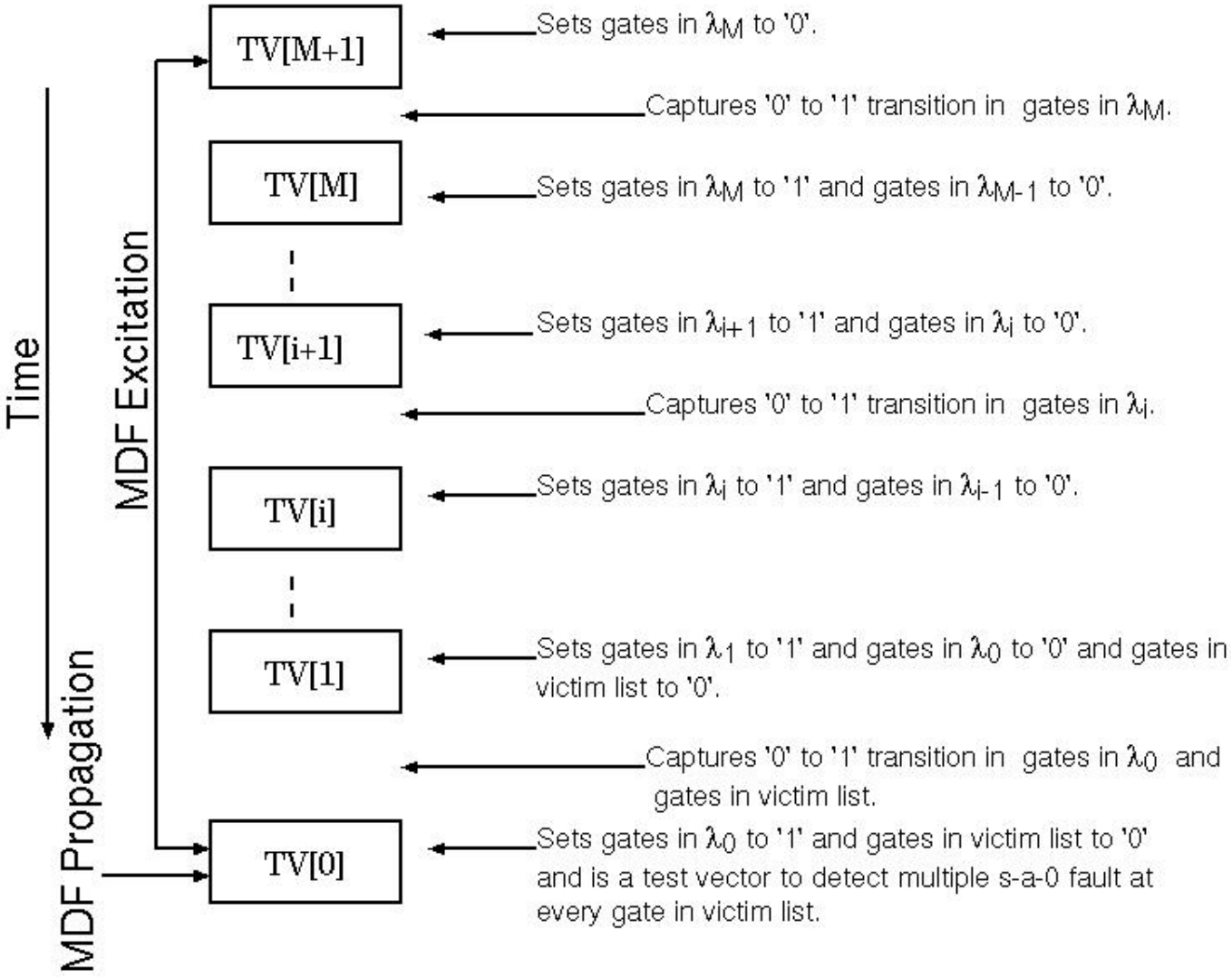

Figure 2.3: Properties of a test sequence corresponding to an $M\_Aggression$. Figure shows the flow of time in which the test sequence should be applied.

of the victim gates at via T and $TV[1]$ sets all victim gates to '0' (along with satisfying 2 above).

Application of test sequence : If we want to observe the MDF at a clock period t ($t > M$), then the test vector $TV[i]$ should be applied at the clock period $t - i$, for all i. Note that the index of clock cycles and the test vectors begin at $M + 1$ and decrease by 1 to 0.

If a TV can be found as above, then $M\_Ag(T)$ is said to be a corresponding $M\_Aggression$ for TV.

An MDF generated by the aggression of a given $M\_Aggression$ ($M\_Ag(T)$) is said to be *testable (excitable)* if and only if there exists a test sequence as above corresponding to $M\_Ag(T)$.

2.3 Test Pattern Generation for Multi Cycle Droop Fault (MDF)

In order to generate a test pattern to detect an MDF at any via we need to first excite the MDF at that via and then propagate the fault to some primary output of the circuit.

The subsequent sections specify what a test pattern generator should do to test MDF for a given circuit. This is elaborated by MDF excitation process and MDF propagation process.

2.3.1 Multi Cycle Droop Fault (MDF) excitation

Generating a test pattern requires first to excite the fault at some target via. But since we don't know a suitable $M\_Aggression$ therefore we start with the exhaustive $M\_Aggression$ ($M\_Ag^e(T)$) for a given target via T and subsequently refine it to get an excitable $M\_Aggression$. Refinement is necessary since aggression in an $M\_Aggression$ requires simultaneous aggression in all the gates in each $\lambda_i(T)$ in their respective cycle which may not be possible due to logical constraints. Therefore the test pattern generation process should refine $M\_Ag_e(T)$ to a suitable $M\_Aggression$ ($N\_Ag(T)$ for $N \leq M$) which is excitable having maximal effectiveness. Moreover the requirement of having 0→1 transition in at least one of the gates in victim list

while applying $TV[0]$ requires to refine the victim list also (due to same reason). Therefore the excitation process is a simultaneous procedure of finding a test sequence and refining the exhaustive *M_Aggression* and the victim list.

Given an exhaustive *M_Aggression* list $M\_Ag_e(T)$ and a victim list V, the test sequence (TV) which excites MDF at T at clock period t $(t > M)$ will do the following:

When TV is applied to the circuit, the gates in the list $\lambda_i(T)$, for each $0 \leq i \leq M$, undergo a 0→1 transition within clock period $t - i$. The gates in victim list are set to '0' by applying $TV[1]$ (at clock period $t - 1$) and are set to '1' by applying TV[0] (at clock period t).

Since all the gates in list $\lambda_i(T)$ may not undergo the specified transition together (due to logical constraints), the excitation process must excite the maximal list of gates (a subset of $\lambda_i(T)$) which can have the specified transition together and have the maximum achievable total effectiveness. This provides us with a refined *M_Aggression* $(N\_Ag(T))$ where $N \leq M$) corresponding to the test sequence TV which can produce a maximal droop at T. Similarly a refined victim list V', for target via T, is also obtained where each gate $g \in V'$ is set to '0' when $TV[1]$ is applied and is set to '1' when $TV[0]$ applied. Clearly the test sequence then contains $N + 2$ test vectors (input vectors to the circuit).

We are interested in the refined *M_Agression* which has the effectiveness greater than or equal to 1 (then only this list will produce MDF at via T). Therefore an MDF is said to be unexcitable if there is no test sequence TV which has a corresponding refined aggression $(N\_Ag(T)$) with effectiveness greater than or equal to 1 and has the corresponding refined victim list V' non empty.

2.3.2 Multi Cycle Droop Fault (MDF) propagation

Propagation of the MDF at a target via T to primary outputs needs the test sequence TV to have the test vector $TV[0]$ such that it is a test vector for the *s-a-0* fault at all the gates of the refined victim list V'.

An MDF is redundant if there is no test vector which propagates the MDF to some primary output.

2.3.3 Automatic Test Pattern Generation (ATPG) for MDF

As far as excitation of MDF is concerned, the ATPG for MDF should find a test sequence for a given exhaustive $M\_Aggression$ ($M\_Ag_e(T)$, where M is suitably large) for a target via T, which excites the aggression. In the process it may be necessary to refine the aggression (due to logical constraints as discussed above). Thus the ATPG for MDF should be flexible in the sense that in the process of finding a test sequence for excitation of a given $M\_Ag_e(T)$, it may settle down (refine) to an aggression $N\_Ag(T)$ ($N \leq M$) which has the highest effectiveness possible and yet excitable.

It is easy to see that the classical ATPG's for STF developed so far cannot find these type of test vectors directly, since these excite the given faults without giving any priority to the excitation of the gates (as in the case of MDF where effectiveness of each aggressor gate can act as its weight or priority, with higher priority associated with higher weights). Thus ATPG's for STF either excite the fault at given gates or fail to do so. On the other hand, an ATPG for MDF should find a maximally effective excitable set of gates which can make transitions together in their respective clock period, from the given exhaustive aggression depending on their priority.

For MDF, an ATPG does the following:

- It first finds the exhaustive aggression $M\_Ag_e(T)$ on target via T and victim list for T. These lists are made using external data coming from the layout information of the chip using static timing analysis and RLC network simulation of the chip [2].

- It finds a test sequence TV which excites and propagates the MDF at T. While trying to find this test sequence the ATPG determines a refined aggression $N\_Ag(T)$ ($N \leq M$) along with a refined victim list V' and simultaneously the corresponding test sequence.

- If the test sequence (TV) found above has the effectiveness of its corresponding refined aggression ($N\_Ag(T)$) less than 1 or V' is empty, then the ATPG declares via T safe from MDF. Otherwise, TV is a desired test sequence for MDF at via T.

In the next chapter, we utilize the cumulative model for MDF to develop an algorithm for test pattern generation.

Chapter 3

A Prototype Algorithm for detection of MDF

It is evident from the previous discussion that we need a new kind of ATPG to generate test patterns for MDF. Here a prototype ATPG is proposed which is a much more restricted form of the required ATPG. This ATPG we call *multi cycle droop fault test generator (MDFTG)*. Before presenting the algorithm for MDFTG we can see some practical considerations which may be useful in doing experiments with MDFTG. By restricted we mean that MDFTG does not refine each aggressor list in a provided $M\_Aggression$ by weeding out gates in the list rather it weeds out lists itself to arrive at a refined $M\_Aggression$. Even this refinement is done following some rules which are inherently greedy in nature.

3.1 Practical Considerations

We can see in Fig. 1.4 that the amplitude of droop quickly diminishes with time and distance. So we can assume the effectiveness of a droop caused by a gate far (distance is more than a threshold) from a target via T to be negligibly small. Same is the case for the droops caused after N clock periods where $N > M$ (a threshold). Thus we can safely choose value of M to be small (3 to 4 cycles for a working frequency of chip to be of the order of GHz). Also we can easily leave out all gates outside a disk of a threshold radius around T.

A gate on its own produces a little effective droop on a target via T so we can take a group of gates (more specifically a group of gates connected to a via of type V_i) as an aggressor $\alpha_i(T)$ with the effectiveness being the sum of individual effectiveness of each gate in the group and add them to the list $\lambda_i(T)$.

3.2 Proposed scheme for generation of test sequence for MDF in combinatorial circuits using classical ATPG's for *stuck-at-fault*

As we have seen earlier in order to generate a test pattern we must refine a given $M\_Aggression$ to reach to a maximally effective $M\_Aggression$. But here we refine the given $M\_Aggression$ by weeding out the aggressor lists rather than gates. This refinement is done in this algorithm using some greedy rules.

3.2.1 Rules to refine $M\_Aggression$

Let the target via be T and the associated victim list be L. Let the given $M\_Aggression$ is $M\_Ag(T)$.

First we try to see that all gates in list L are such that multiple *s-a-0* faults at those gates is redundant or not. If it is redundant then there is no need to proceed ahead since the MDF at via T would then be redundant. Next as we can see in Fig. 1.6, the intensity of droop caused by an aggressor list of type $\lambda_i(T)$ is much more than that of type $\lambda_j(T)$ where $i < j$. This suggests that in case of contradiction between both lists we must first try to retain $\lambda_i(T)$ and if failed then try to retain $\lambda_j(T)$ in the refined $M\_Aggression$. Considering these we can devise some rules as follows:

1. First find that multiple *s-a-0* fault at the gates in the victim list is redundant or not. If redundant then stop generating test sequence.

2. Next find that the gates in victim list can be simultaneously set to '0' or not. If not then stop generating test sequence.

3. In order to capture the transition in an aggressor list $\lambda_i(T)$ give preference to the list with lower value of i (i.e. with more effectiveness).

4. Stop generating test sequence whenever the total effectiveness of all the list whose transition has been captured exceeds 1 and output the test sequence.

5. If at any point of time the remaining effectiveness to be captured is too little such that there is no hope of capturing a total effectiveness of more than or equal to 1, then stop generating test sequence.

The next section presents the algorithm for MDFTG following these rules.

3.2.2 Algorithm for Multi Cycle droop Fault Generator (MDFTG)

To generate a test sequence for a target via T, first make some tentative $M\_Aggressions$ for T where M is the threshold clock cycles and using vias only nearby distance from T, also make a victim list V. These lists are made using external data coming from the layout information of the chip using static timing analysis and RLC network simulation of the chip [2]. Now for an $M\_Aggression$ $M\_Ag(T)$ apply the below algorithm.

ALGORITHM : MDFTG

INPUT : A circuit C, a victim list V, an $M\_Aggression$ $M\_Ag(T)$.
OUTPUT : If successful then a test sequence TV and number of test vectors in it else failure.
START

- **Step 0 :** Let effectiveness covered $E = 0$, hope $H = \sum_{\lambda_i(T) \in M\_Ag(T)} \xi(\lambda_i(T))$, and effectiveness at stake $S = 0$.

  If $H < 1$, then stop and output failure.

- **Step 1 :** (for test vector $TV[0]$)

  Find a test vector which detects a multiple *s-a-0* fault at victim gates and also simultaneously sets the gates in $\lambda_0(T)$ to '1' (if $\lambda_0(T)$ is not empty). If such a test vector is

not possible then find a test vector which just detects the multiple *s-a-0* fault at victim gates. If this test vector is also empty then stop processing this *M_Aggression* else we have found $TV[0]$.

$$H = H - \xi(\lambda_0(T))$$

If $TV[0]$ sets gates in $\lambda_0(T)$ to '1',

$$\text{then } S = \xi(\lambda_0(T)).$$

If $(E + S + H) < 1$, then stop and output failure.

- **Step 2 :** (for test vector $TV[1]$)

  Find a test vector which sets the gates in $\lambda_0(T)$ to '0' (if it is not empty and if $TV[0]$ sets gates in $\lambda_0(T)$ to '1') and simultaneously sets gates in $\lambda_1(T)$ to '1' (if it is not empty) and also simultaneously sets the gates in V to '0'. If this test vector is empty, then find a test vector which simultaneously sets the gates in V and $\lambda_0(T)$ to '0' (if it is not empty and if $TV[0]$ sets gates in $\lambda_0(T)$ to '1'). If this test vector is also empty, then find a test vector which simultaneously sets the gates in V to '0' and gates in $\lambda_1(T)$ to '1' (if it is not empty). If this test vector is also empty, then stop processing this *M_Aggression* else we have found $TV[1]$.

  $$H = H - \xi(\lambda_1(T))$$

  If $TV[1]$ sets gates in $\lambda_0(T)$ to '0' and $TV[0]$ sets gates in $\lambda_0(T)$ to '1' (this means the transition of $\lambda_0(T)$ has been captured), then

  $$E = E + \xi(\lambda_0(T)).$$

  If $E \geq 1$, then stop and output the test sequence and 1.

  If $TV[1]$ sets gates in $\lambda_1(T)$ to '1', then

$$S = \xi(\lambda_1(T)).$$

If $(E + S + H) < 1$, then stop and output failure.

- **Step 3 :** (for test vector $TV[i]$, $2 \leq i \leq M$)

  For $2 \leq i \leq M$ **do,**

  If both $\lambda_i(T)$ and $\lambda_{i-1}(T)$ are empty, then $TV[i]$ is a random test vector else find a test vector which sets the gates in $\lambda_{i-1}(T)$ to '0' (if it is not empty and if $TV[i-1]$ sets gates in $\lambda_{i-1}(T)$ to '1') and simultaneously sets gates in $\lambda_i(T)$ to '1' (if it is not empty). If this test vector is empty and $TV[i-1]$ sets gates in $\lambda_{i-1}(T)$ to '1', then find a test vector which just sets gates in $\lambda_{i-1}(T)$ to '0'. If this test vector is also empty, then find a test vector which just sets gates in $\lambda_i(T)$ to '1' (if it is not empty). If this test vector is also empty, then find a random test vector.

  $$H = H - \xi(\lambda_i(T))$$

  If $TV[i]$ sets gates in $\lambda_{i-1}(T)$ to '0' and $TV[i-1]$ sets gates in $\lambda_{i-1}(T)$ to '1' (this means the transition of $\lambda_{i-1}(T)$ has been captured), then

  $$E = E + \xi(\lambda_{i-1}(T)).$$

  If $E \geq 1$, then stop and output the test sequence and i.

  If $TV[i]$ sets gates in $\lambda_i(T)$ to '1', then

  $$S = \xi(\lambda_i(T)).$$

  If $(E + S + H) < 1$, then stop and output failure.

  done.

- **Step 4 :** (for $TV[M+1]$)

  If $TV[M]$ sets gates in $\lambda_M(T)$ to '1', then find a test vector which sets $\lambda_M(T)$ to '0'. If this test vector is possible, then

  $$E = E + \xi(\lambda_M(T)).$$

Else if test vector above is not possible (empty), then output failure.

STOP

This procedure is repeated for each $M\_Aggression$ in the tentative list of aggressions until a test sequence is found or the list is exhausted. If no test sequence is found for any $M\_Aggression$ then via T is declared to be safe from an MDF.

For the purpose of finding test vectors in each step, any ATPG or SAT solver available in practice can be used along with the circuit modifications similar to ones used in [2]. For this purpose a reduced victim list V^R [2] is used instead of the victim list V. Reduced victim list, by definition, is a subset of victim list V in which no two gates are connected by a signal path in the circuit.

3.2.3 Comparison of MDFTG with ATPG for MDF

This method detects an MDF at a target via T with a probability depending on the procedure to choose the tentative aggressions for via T. If we choose a higher number of tentative aggressions then the probability for the MDF at via T to be detected is generally higher. But since each step in the method above uses an ATPG for STF, it takes a lot of time to complete the test sequence generation. So we aim at keeping the number of tentative lists small. Thus there is a trade off between accuracy and test pattern generation time.

The ATPG for MDF discussed in chapter 2 detects the MDF at via T with higher probability, because the exhaustive aggression does not include the gates which have negligible contributions to the MFPD at via T. Further, the excitation process searches for the maximum effective aggression (refined aggression) by weeding out gates while finding the test sequence for it.

3.3 Experiments

A working ATPG for combinatorial circuits is implemented in C using the algorithm presented above. This ATPG is called MDFTG. In absence of actual chip data the vias are made using the ISCAS85 bench mark circuits. The $M\_Aggressions$ and victim lists are chosen from those

vias. Two different methods have been employed to generate *M_Aggressions*. These are *(a) logical neighbors of seed* and *(b) random selection*, where seed is any gate randomly chosen from the circuit. The generated *M_Aggressions* are then fed to MDFTG along with circuit and victim lists and the response of MDFTG is observed.

It must be noted that for the purpose of implementation of MDFTG we have taken the reduced victim list V^R instead of a victim list V. Since the gates in list V^R don't have a path between them. This is important for the circuit modification as discussed in next chapter.

The experiments are discussed below.

3.3.1 Experiment #1 (Logical neighbors method)

As the name suggests this method makes the *M_Aggression* using the logically connected gates nearest to a seed gate. Thus the formed *M_Aggressions* are strongly logically correlated. In this method we form an *M_Aggressions* using following steps:

- **Step 0 :** Form a gate list G from the given circuit. Choose a value of M randomly between 1 to 4.
- **Step 1 : While G is not empty, do**

  Choose a random seed gate g from list G.

  Find the nearest logically connected gates from the gate g going in both directions (forward and backward) up to 2 levels. Call this set T'.

  If T' has more than 4 gates, then do

  Form a set of 5-6 gates from gates in T'. Call this set T. These are the gates which are connected to the target via.

  Form a victim list V of 1-4 gates from the set T choosing randomly.

  Find the set of gates logically connected to gate g going in both directions (forward and backward) up to M+2 levels starting from 3rd level. Form an *M_Aggression* from this set in such a manner that the gates nearer to g fall in $\lambda_i(T)$ with smaller value of i.

  Remove gates in T from the list G.

else

Remove gates in T' from the list G.

end if.

done.

In this manner a list of $M\_Aggressions$ is formed. These $M\_Ag(T)'s$ are fed to MDFTG and the number of testable $M\_Ag(T)'s$ are recorded as shown in the Table 3.1 below.

Table 3.1: Experiment #1 (Program run on HP XW8400 Workstation and $1 \leq M \leq 4$)

Circuit	**# M_Ag(T)**	**# testable M_Ag(T)**	**# redundant M_Ag(T)**	**# unexcitable M_Ag(T)**	**time(s)**
c432.bench	6	2	2	2	1
c499.bench	7	2	2	3	1
c880.bench	27	1	21	5	1
c1355.bench	48	0	34	14	3
c1908.bench	74	0	55	19	4
c3540.bench	111	2	76	33	20
c5315.bench	184	0	130	54	76
c6288.bench	210	0	170	40	100
c7552.bench	269	1	193	75	241

3.3.2 Experiment #2 (Random selection method)

In this method the $M\_Aggressions$ are formed choosing random gates from the circuit. Thus the formed $M\_Aggressions$ are uncorrelated. In this method we form an $M\_Aggression$ using following steps:

- **Step 0 :** Form a gate list G from the given circuit. Choose a value of M randomly between 1 to 4.
- **Step 1 : While G has more than 4 gates, do**

Form a set of 5-6 gates choosing randomly from list G. Call this set T. These are the gates which are connected to the target via.

Form a victim list V of 1-4 gates from the set T choosing randomly.

Remove gates in T from the list G.

Form an $M\_Aggression$ choosing gates randomly from list G.

done.

In this manner a list of $M\_Aggressions$ is formed. These $M\_Ag(T)'s$ are fed to MDFTG and the number of testable $M\_Ag(T)'s$ are recorded as shown in the Table 3.2 below.

Table 3.2: Experiment #2 (Program run on HP XW8400 Workstation and $1 \leq M \leq 4$)

Circuit	**# M_Ag(T)**	**# testable M_Ag(T)**	**# redundant M_Ag(T)**	**# unexcitable M_Ag(T)**	**time(s)**
c432.bench	26	4	4	18	2
c499.bench	33	3	8	22	2
c880.bench	63	34	11	18	7
c1355.bench	90	10	18	62	67
c1908.bench	146	51	22	73	301
c3540.bench	278	50	77	151	2096
c5315.bench	384	35	27	322	1835
c6288.bench	402	27	112	263	13164
c7552.bench	585	27	37	521	5522

3.4 Inferences on experimental results

We can observe that both the methods employed for the experiments are actually implementing two extreme cases to choose the $M\_Aggressions$. In actual practice the real VLSI chips has gates (connected to the same via) which are neither strongly logically correlated nor completely uncorrelated. Thus the experiments performed above are either much more constrained or

completely free in choosing *M_Aggressions*. More over the MDFTG tool developed is also much constrained since it refines the given *M_Aggressions* by weeding out aggressor lists instead of gates. In view of this case we can observe few points as below:

- Appearance of testable *M_Aggressions* suggest that even when the *M_Aggressions* are chosen from logically connected neighborhood from target via T they are capable of producing the droop and hence MDF at the victims at via T. Though they are much less likely candidates to produce MDF. This is because the gates in these *M_Aggressions* are strongly logically correlated.

- Also we can observe that a large proportion of *M_Aggressions* chosen randomly are testable. Thus the *M_Aggressions*, whose gates are less logically correlated as in random selection mode above, are more likely candidates to produce MDF.

- The time consumption shown in tables shows that it is a very time consuming process. This is due to the fact that each step of MDFTG modifies the circuit in some suitable manner and runs ATPG ATALANTA to find the test vectors. Which itself is very time consuming.

Thus we can conclude that in real VLSI chips a sizable number of vias may be prone to MFD. With increasing transistor density and frequency of operation, MFD can become a potential hazard to the operation of high performance chips in near future. The next chapter presents the details of implementation of a test pattern generator for MDF faults.

Chapter 4

Implementation of Multi Cycle Droop Fault Generator (MDFTG)

The tool MDFTG is implemented in C. To generate test vectors at various steps the ATPG ATALANTA has been used. The implementation of MDFTG is similar to the tool DFTG developed by Ashish Nigam [5]. In tool MDFTG some functions of the tool DFTG has been used. The data structure used to store circuit is similar to the data structure used in DFTG [5]. The functions which have been written for MDFTG are in files starting with character 'M' (ex. *M_testGen.c*). The details of the functions implemented and used from the tool DFTG are given in subsequent sections.

4.1 Reading and writing to files

For the purpose of reading and storing the given ISCAS85 bench mark circuit files and also to make fault files (to be given to atalanta as input) the functions from tool DFTG [5] were used. Some of the important functions are listed below:

- **readCircuitFile() :** This function reads a given ISCAS85 benchmark circuit file and stores the circuit in the data structure *circuitInfo*.

  This function is implemented in DFTG and directly used in MDFTG.

- **writeCircuitFile() :** This function writes a given ISCAS85 benchmark circuit file stored

in data structure *circuitInfo* to the output file.

This function is implemented in DFTG and directly used in MDFTG.

- **read_m_AggressionFile() :** This function reads a given *M_Aggression* and stores this in the data strucure *m_Aggression*.

  This function is implemented in MDFTG.

- **constructFaultFile() :** This function generates a fault file used as an input to the ATPG ATALANTA.

  This function is implemented in DFTG and directly used in MDFTG.

4.2 Functions to modify circuit

While finding the test vectors for different conditions in the algorithm MDFTG, we need to modify the circuit in such a way that by finding the test vector for a specific fault in the modified circuit using ATALANTA we get the required test vector. The modifications used in MDFTG are similar to ones used in DFTG and can be found in [5]. The functions used to modify the circuit in desired way are given below:

- **constructConeSubCircuit() :** This function constructs the cone sub circuit of a given circuit using a set of gates as the starting points and backtracking to some primary input of the original circuit.

  This function is implemented in DFTG and directly used in MDFTG.

- **NOR_AND_NAND_modification() :** This function modifies a given circuit. The inputs to this function are two sets of gates (say A and B) and a circuit C to be modified. Let

$$A = \{a_1, a_2, .., a_n\}$$
$$B = \{b_1, b_2, .., b_m\}$$

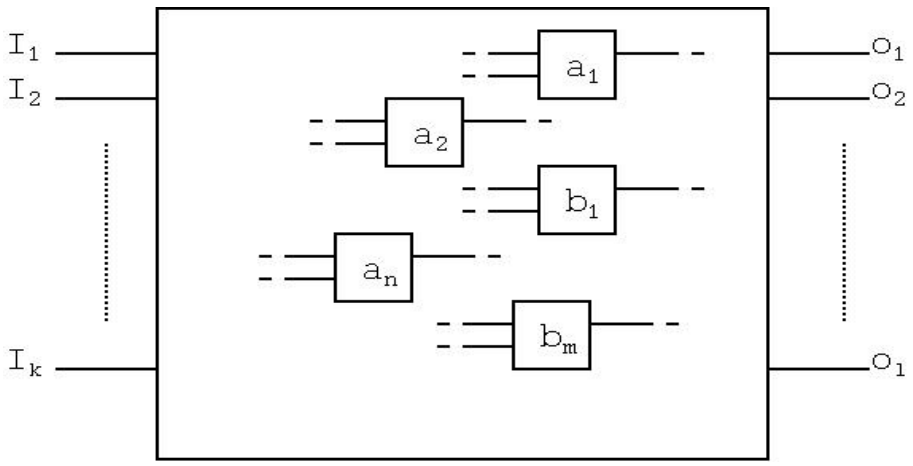

Figure 4.1: The circuit C showing the gates belonging to set A and B.

The following figures show the working of this function. Fig. 4.1 shows the combinatorial circuit C and the gates belonging to sets A and B. This function takes these as input and first finds the cone sub circuit C' of C starting with gates in both the sets A and B and backtracking to some primary input. This is done by internally calling *construct-ConeSubcircuit()*. Fig. 4.2 shows the cone sub circuit C' of the gates in set A and B.

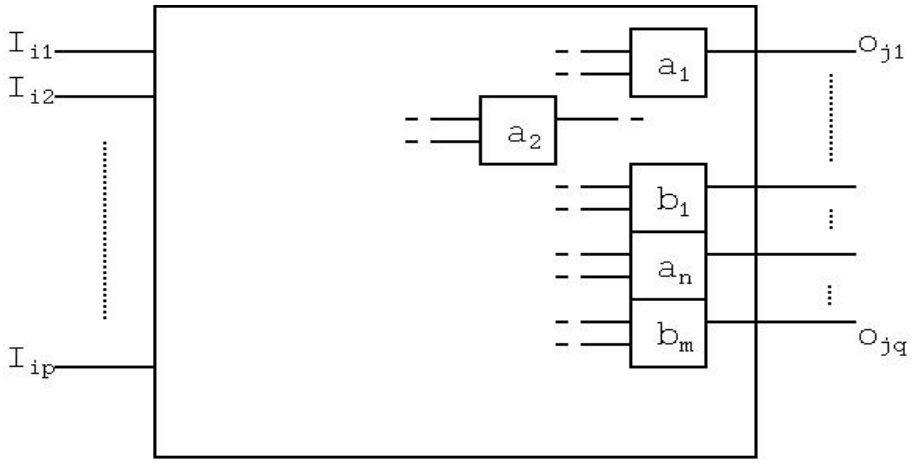

Figure 4.2: The cone sub circuit C' of circuit C in Fig. 4.1 corresponding to the gates in set A and set B. Note that not all of these gates have their output as the primary output of circuit C'.

We can see that the primary output of C' are the outputs of some of the gates in set A

and B. Not all the gates of these set always form the primary output of C' since some of the gates in these set may be on the path from some primary input to some other gate of these sets.

After finding the cone sub circuit C', it is modified to the circuit C'' as shown in Fig. 4.3. The outputs of the gates belonging to set A are input to a NOR gate and outputs of the gates in set B are input to an AND gate. The outputs of these NOR and AND gates are input to a NAND gate and the output of this NAND gate is the primary output of circuit C''.

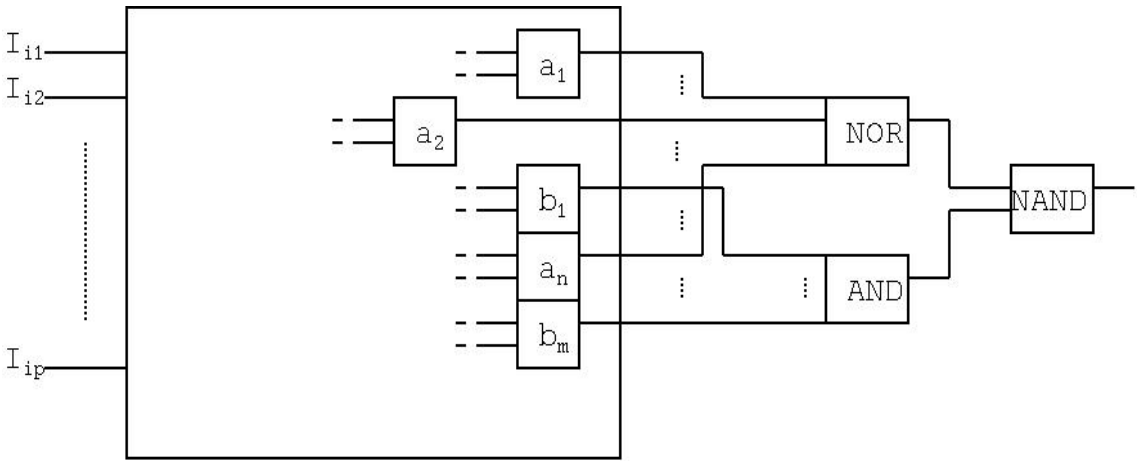

Figure 4.3: The modified circuit C'' (output of *NOR_AND_NAND_modification()*).

This circuit modification is important since the *s-a-1* fault at the primary output of C'' has a test vector which sets this primary output to '0' and thus sets the gates in set A to '0' and the gates in set B to '1'. By using ATALANTA we can find this test vector.

- **modify() :** This function modifies the circuit for *step 1* of the MDFTG algorithm. This modification is similar to the one by function *NOR_AND_NAND_modification()*. This function is used in implementing *step 0* of the algorithm MDFTG.

  This function is implemented in DFTG and directly used in MDFTG.

4.3 Other functions

- **findReducedVictimList() :** This function finds a maximal subset V^R of a given victim list (list of gates) V, which contains gates such that no two gates of V^R are on the same signal path from a primary input to a primary output of the given circuit. It is important to find this reduced victim list V^R since the function *modify()* modifies the given circuit in such a manner that if reduced victim list V^R is not used instead of victim list V in modification, then feed back loops will be introduced in the resulting circuit. Thus combinatorial nature of the circuit will be lost. Details of this can be found in [5].

  This function is implemented in DFTG and directly used in MDFTG.

- **test_aggression() :** This function is the main function implementing the steps of algorithm MDFTG.

Chapter 5

Conclusions

This thesis presents a new model for multi cycle droop faults to capture the excitation of these faults. The nature of the test pattern and the ATPG for these faults is also presented. On the basis of experiments and the model following conclusions can be drawn :

- Experiments show that even if we chose $M\_Ag(T)$'s from logically nearer gates (Experiment # 1) there can be some MDF which are non redundant and hence testable. Also random selection (Experiment # 2) has many testable $M\_Ag(T)$'s. These methods are two extreme cases to choose an aggression. In practice an $M\_Ag(T)$ made from a real chip would be less constrained than that in Experiment # 1 and less random than that in Experiment # 2. So there is a considerable chance of MDF occurring in a real high performance chip specially in future chips.

- The time taken by tool MDFTG to generate test pattern is very high since at each step it modifies the circuit and uses ATPG ATALANTA to find a required test vector while generating a test sequence for a given $M\_Aggression$. This can be reduced if we implement the ATPG discussed in chapter 2. There we generate the test sequence while refining the exhaustive $M\_Aggression$. But implementation of this ATPG is yet to be explored.

5.1 Future works

This thesis presents a theoretical model for MDF and also presents a guideline for an ATPG for it. But only a restricted version of this ATPG has been implemented. It remains to find a suitable algorithm (preferably applicable in industrial applications) for the ATPG for MDF.

In order to build an ATPG for MDF one of the main challenge is to refine a given $M\_Aggression$ to a maximally effective yet excitable $M\_Aggression$ by weeding out gates causing problem (i.e. making it unexcitable). For this two different approaches can be adopted. These are:

- One approach can be to generate an incompatibility graph where gates (belonging to a $\lambda_i(T)$) which can not be excited (i.e. made to switch $0 \rightarrow 1$) within the same clock cycle are connected by edges. The gates has weights equal to their effectiveness. In this manner the incompatibility graph for the given $M\_Aggression$ can be made. Now this graph can be analyzed and a maximally effective $M\_Aggression$ can be formed. This may give a refined $M\_Aggression$ which has a very high chance of being excitable. Now we can implement the tool MDFTG to find the test sequence of this refined $M\_Aggression$.

- The other approach can be to formulate the problem to find refined $M\_Aggression$ in a logical predicate (Conjunctive Normal Form (CNF)) involving the given circuit, gates of the given $M\_Aggression$, their effectiveness and the time of their required switching. This predicate is equivalent to a circuit whose satisfiability we have to find. Any SAT solver present in literature will give the required test sequence as a solution to this predicate's satisfiability question. But the main difficulty in this approach is to incorporate the effectiveness of each gate in the predicate and formulating in such a manner that (preferably a single run of) SAT solver gives the refined $M\_Aggression$ along with the test sequence corresponding to it.

Appendix A

Manual for tool MDFTG

This Manual contains all available options in the tool. MDFTG is the tool for generating test sequence for finding out faults due to mid frequency power droop in the combinational circuits.

SYNOPSIS : mdftg [options]

For on line helps available, Type "mdftg -help"

The options to run the tool MDFTG are given below. If no option is provided then default is used.

- Option '$-help$' : For HELP

  command is "mdftg $-help$"

- Option '$-c$' : To provide a circuit file

  command is "mdftg $-c$ [circuit file]"

  Ex. mdftg $-c$ c432.bench

- Option '$-a$' : For test sequence generation for a given M_Aggression

  command is "mdftg $-a$ [aggressor file] $-c$ [circuit file]"

  Ex. mdftg $-a$ m_aggression.1 $-c$ c432.bench

- Option '$-l$' : To put the test sequence in a log file

  command is "mdftg $-a$ [aggressor file] $-c$ [circuit file] $-l$ [logfile]"

If no -l option is provided then 'stdout' is default logfile.

Ex. mdftg $-a$ m_aggression.1 $-c$ c432.bench $-l$ result.log

- Option '$-n$' : For experiment using nearest neighbour method

  command is "mdftg $-e$ [circuit file] $-n$"

  If no option is provided then '$-n$' is default option.

  Ex. mdftg $-e$ c432.bench $-n$

- Option '$-r$' : For experiment using random selection method

  command is "mdftg $-e$ [circuit file] $-r$"

  If no option is provided then '$-n$' is default option.

  Ex. mdftg $-e$ c432.bench $-r$

- Option '$-M$' : To fix the value of M in experiment

  command is "mdftg $-e$ [circuit file] [$-r$ or $-n$] $-M$ [value of M]"

  Default value of M is 4.

  Ex. mdftg $-e$ c432.bench $-n$ $-M$ 6

- Option '$-v$' : To fix the no. of gates in each via in an experiment

  command is "mdftg $-e$ [circuit file] [$-r$ or $-n$] $-v$ [no. of gates in via]"

  Default value is 6.

  Ex. mdftg $-e$ c432.bench $-n$ $-v$ 4

Bibliography

[1] S. Pant, E. Chiprout, D. Blaauw, *Power Grid Physics and implications for CAD*, Proc. of the 43^{rd} Annual Conference on Design Automation, ACM IEEE Automation Conference, Session 13, pp. 199-204, 2006.

[2] D. Mitra, S. Bhattacharjee, S. Sur-Kolay, B. B. Bhattacharya, S. T. Zachariah and S. Kundu, *Test Pattern Generation for Droop Faults*, Proc. of the 19^{th} International Conference on VLSI Design, pp. 343-348, 2006.

[3] D. Mitra, A. Nigam, S. K. Dey, S. Sur-Kolay, B. B. Bhattacharya, *Testing Droop Faults in Full Scan Circuits*, Proc. of the VLSI Design and Test Symposium (VDAT 2007), pp. 185-195, 2007.

[4] I. Polian, A. Czutro, B. Becker, S. Kundu, *Power Droop Testing*, IEEE Design & Test, Volume 24, ISSUE 3, pp. 276-284, May 2007.

[5] A. Nigam, *Test Pattern Generation for Timing Faults due to Power Supply Droop in Full Scan Circuits*, M. Tech. (Computer Science) Dissertation Series, Indian Statistical Institute, 2006.

[6] C. Tirumurti, S. Sur-Kolay, S. Kundu, Y. S. Chang, *A Modeling Approach for Addressing Power Supply Switching Noise Related Failures of Integrated Circuits*, Proc. Design, Automation and Test in Europe Conference and Exhibition Volume II (DATE'04), pp 21078, 2004.

[7] M. Abramovici, M. A. Breure, A. D. Friedman, *Digital Systems Testing and testable Design*, Piscataway, New Jersey: IEEE Press, 1990.

[8] *http://service.felk.cvut.cz/.*

[9] *http://www.princeton.edu/ chaff/zchaff.html.*