Case Study: A Tool Centric Approach for Fault Avoidance in Microchip Designs

Clemente Izurieta
Department of Computer Science
Colorado State University
Fort Collins, Colorado, USA
cizuriet@cs.colostate.edu

***Abstract*— Achieving reliability in fault tolerant systems requires both avoidance and redundancy. This study focuses on avoidance as it pertains to the design of microchips. The lifecycle of a microchip is centered on the design of every block that makes up the hierarchy of the design. Early in the lifecycle the emphasis is on obtaining clean and fault free blocks for every CAD (Computer Aided Design) tool that the block is subjected to. A block is either a schematic or the artwork of a given functional circuit. As a block evolves, a regression of already released CAD tools is run to obtain predetermined levels of quality. Without running these regression suites, developers risk delaying the appearances of faults; which in the field of microchip development can prove extremely costly, especially when entering the manufacturing stages where the costs of making masks and test chips can be prohibitive. This study uses traditional data mining techniques to follow the evolution of a number of legacy blocks in a microchip and track the number of failures encountered per block and per tool over a period of time.**

I. INTRODUCTION

According to Avizienis [1], *"the increasing complexity of software and VLSI chip logic emphasized the impossibility of removing all design faults prior to operation,"* however, this ever increasing complexity is also forcing design teams to focus more closely on fault avoidance techniques. Avoiding and planning to avoid failures, i.e. fault-avoidance –*"is the most important aspect of fault tolerance."* [7] Today, most design centers from companies such as National Semiconductor, AMD, Intel, Hewlett Packard, IBM, etc., that focus on full custom chip design, are actively focusing on fault avoidance and quality of designs by subjecting them to various CAD tools that analyze noise levels, power consumption, circuit analysis, electrical rule violations, and design rule violations to name a few. In a study done by Wadsack [8], various design verification and testing methods were evaluated during the development of four AT&T processors. Wadsack's study focused on the fault coverage achieved by design verification tools, physical tools, and diagnostic tools. In this case study we focus on the evolution of the block data itself rather than the tools. The data from various CAD tools allows us to track progress made in any block of a microchip over a period of time. When a given milestone in the lifecycle of the microchip is reached, we analyze the progress of a block by comparing its statistics within all the CAD tools. Traditional data mining techniques are used to extract block information. Because milestone data is not available from commercial companies, the sampling of the data at discrete periods of time will be used. The analysis of these statistics can help pin point areas for improvement, or areas of concern. The trends of the data over a period of time also help us to determine if we can accept or refute our hypotheses. Data collected over a period of time provides "experience," which can also be used for prediction purposes in future designs of microchips. In a recent study by Schroter et al. [6] for example, the authors use past experience to help with predicting the fault-proness of java plug-ins *"by learning from history which design decisions correlated with failures in the past."*

This research aims to provide an insight into the fault avoidance techniques of microchip design, where the author empirically analyzes hierarchical chip block data over a period of time. According to Niessen [4], *"Hierarchical design methods permit the creation of a new generation of CAD programs that can both give a designer better support and can be much more efficient than the present generation of tools"*. The following hypothesis is tested:

H_0: All CAD tool failure rates of a microchip block appear to correlate over a period of time.

It is expected that the hypothesis will not be rejected.

II. PROCEDURE AND METHODOLOGY

Our approach to the case study is to first investigate various CAD tools. A brief description of each CAD tool will be given to help the reader understand why such a tool helps with fault avoidance. The tools themselves will be from the design verification and physical spaces. Diagnostic tools are not generally available, as they are much more involved and focus on root causing faults at much later stages of the chip lifecycle. The case study has the following steps:

1. Obtain traditional (legacy) data of old hierarchical blocks from microchips. Data will be collected from commercial companies if available, or open software

organizations such as the OpenAccess coalition [5]. Data will be sanitized if necessary.

2. Develop a framework in some scripting programming language such as Perl, Tcl, or Ksh that allows for plug and play of CAD tools.
3. Use data mining methods for every CAD tool.
4. Plot data over a period of time using graphing software.
5. Analyze the data over some period of time. I will use statistical software and manual techniques to perform various curve fitting models to identify the growth rates of faults.
6. Identify the threats to the validity of the study. Construct, content, internal, and external threats will be identified.
7. Analyze H_0. Refute or accept the hypothesis.

III. CAD Tools

There are too numerous CAD tools to run on a block. I limit this study to five tools from the design verification and physical space that cover different aspects of the design over the lifecycle of the chip. The following section gives a brief overview of each of the tools studied.

The first tool that we study is a noise analysis tool. Noise analysis primarily encompasses the understanding of coupling effects between signal lines in a microchip. *"Coupled noise analysis has become a critical issue for deep-submicron, high performance design,"* [9] and using one such tool is critical to this study. Noise coupling causes glitches in signals and lines, which turns out to be one of the biggest electrical failures of microchips. Failures occur due to a number of reasons; the closeness of signal lines, the length of signal lines, and strength of the outputs of the gates that drive the signals. The data gathered by these tools typically comes from the schematics of the microchip, and from characterizations of fates and signals. There are typically aggressor and victim lines that need to be accounted for. Noise analysis tools try to tag the victim lines when the glitches observed on them exceeds a voltage threshold. This is an indication to the designer to reduce capacitance via a number of different ways in order to avoid faults later in the lifecycle. Because this is beyond the scope of this study, we do not pursue it any further.

Circuit analysis CAD tools also play a significant part. Their role is to analyze a circuit and separate out clear components for further analysis. This is necessary because today's designs are made of millions (in some cases billions) of transistors which would make searching algorithms useless without this separation of components. Most companies use circuit analysis tools. Their goal is to identify parts (circuits) such as latches, ram cells, dynamic circuits, and classify them as such. Once classified, a separate kind of tool can be used to determine if the component will actually work given the values specified for the circuit. The latter is achieved with the aid of spice simulations. CAD tools in this space can flag certain circuits as unrecognizable, and in most cases this may be due to the placement of the wrong gates or connections between gates which should not be there. Clearly the need to recognize every circuit before proceeding to silicon is imperative in order to avoid costly rework. Tools in this space are typically run in the same time period as noise analysis tools.

Finally, in the design space we also investigate a clock verification tool. These tools examine the clock networks of a chip and make sure that all circuits do not suffer from clock delays, or that clock skews are kept in check.

It is important to note that all tools work on designs that are hierarchical to begin with, but during the analysis phase, these hierarchies are "flattened", in order to make more efficient and accurate use of the tool. In some cases during the flattening stages, certain blocks are not available and are instead synthesized.

In the physical space, a design rule checker (DRC) tool is studied. This was the only tool that I had access to. Design rules clearly and unambiguously specify the geometries of the microchip. The rules are used by designers when creating the artwork (layout) of the chip. Design rules are guarded heavily by different companies and they tend to change as more mature designs are produced. Figure 1 is an example of the layout of a chip.

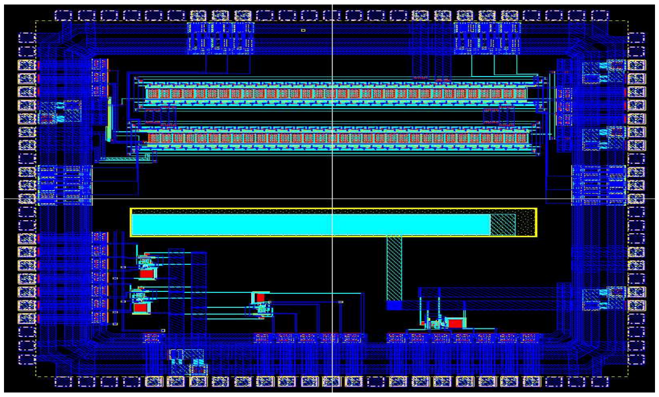
Figure 1. Layout of a Generic Microchip Block

All design errors must be eliminated in order to reduce faults when masks for the chip must be produced during the manufacturing stages. Interestingly, an additional and common way to avoid faults during manufacturing is to allow for some "void" silicon to exist in the artwork. In essence, this is a form of silicon redundancy. This silicon real estate can be used during manufacturing to create fixes to design errors.

To summarize, this study looks at four design verification tools; namely Noise Analyzer (NA), Circuit Analysis 1 (CA_1) , Circuit Analysis 2 (CA_2), and the Clock Analyzer (ClkA). In the physical space only one physical design tool is available; the Design Rule Checker (DRC).

IV. Measurement Software

The framework for this project was developed using the Perl programming language. Code for the various modules can be found in the appendices. Figure 2 shows a high level view of the architecture.

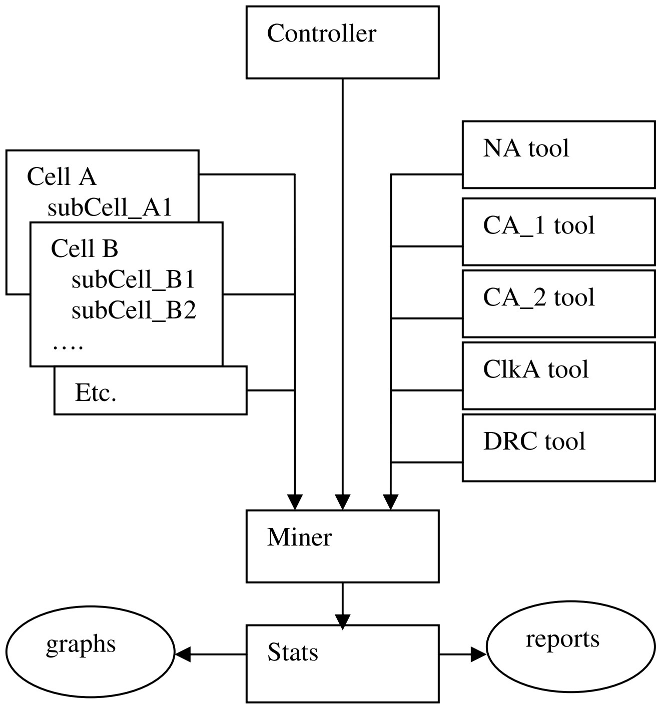

Figure 2. Framework for Statistics gathering per cell and CAD tools.

The Controller is responsible for setting up global policies and variables that are used throughout the execution of the program. The controller also sets up and registers the various tool modules. Each tool module is responsible for mining the statistics of the cells by looking for the outputs in each of the files left behind after running a CAD tool. The Cell files are each hierarchical in that they are composed of a number of subcells. The depth of the hierarchies is much larger than the data available. I was only able to expand the blocks to just the first level of hierarchy. The miner component is responsible for calling the callback functions of each tool and after collecting the data, calling the Stats module to create the reports in the form of ASCII files. The graphs were created manually by importing them into MS Excel.

V. Analysis

In this section I analyze the curves and data for the various tools and blocks obtained. Each tool was run on every cell and subcell where data was available. Data was available generally for a period of at most five months and as little as two days, while some subcells did not have any data available at all. The data for each cell was sanitized by removing superfluous and unnecessary information from the output files. The interesting data for purposes of this study are the number of subcells that the tool is run on, and the total number of passes, or non-failures per cell. The data is tracked over a period of time to see if trends tend to appear in terms of the number of failures per cell and subcell.

In order to make progress as a microchip is developed one would ideally like to see the number of failures decrease as times goes by. In other words, we would like to see the curve that describes the total number of blocks that the tool is run on to come closer together to the curve that describes the total number of non-failures. To illustrate this point lets look at a representative subcell. Figure 3 displays the data after running the clock analyzer tool ClkA for subcell A1 of cell A.

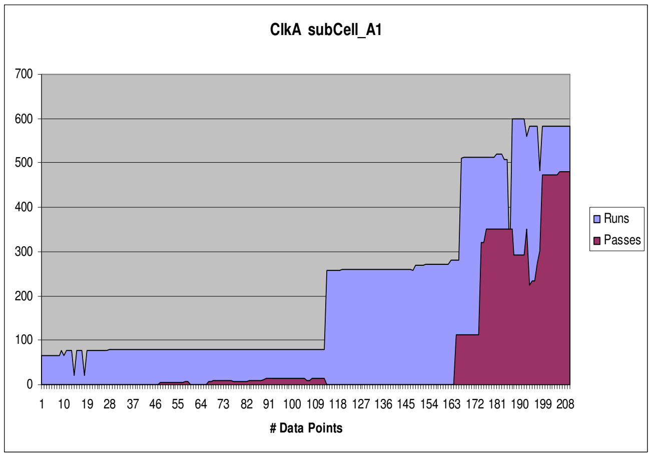

Figure 3. Results for tool ClkA on subcell A1

We can clearly see that as time increases, the number of passes gets closer to the total number of cells that the tool was run on. It is also interesting to note that the total number of cells that a tool is run on can also decrease. This phenomenon is due to the redesigning of some cells. When the functionality of a cell changes, new subcells are either added or removed from the cell, thus changing the total number of blocks. This can be observed in Figure 3 in the latter stages of the graphs. Figure 4 illustrates this phenomenon more clearly. In this figure you can clearly see the total number of cells decrease as time increases, but you can also observe that the two curves appear to start to come together towards the latter stages of the lifecycle of this cell.

We find that not all cells reduced the total number of non-failures over time. Some cells showed evidence of decay, or increased failure rates. Figure 5 is an example where we can clearly see the number of failures increase (number of passes go down) relative to the total number of cells as time progresses. In other words, we can see that the two curves appear to grow apart.

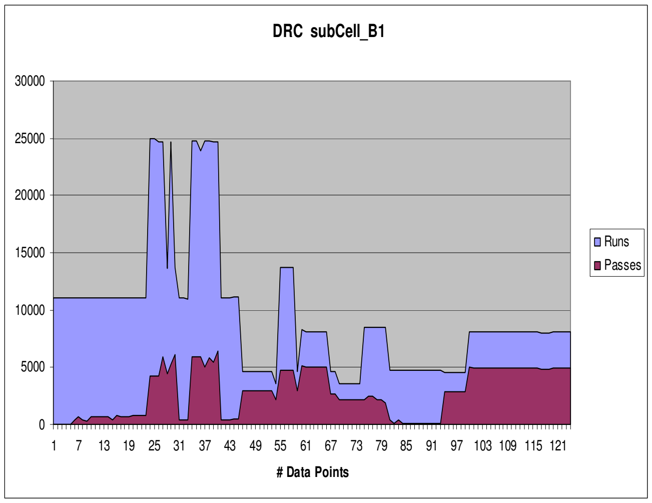

Figure 4. Results for tool DRC on subcell B1

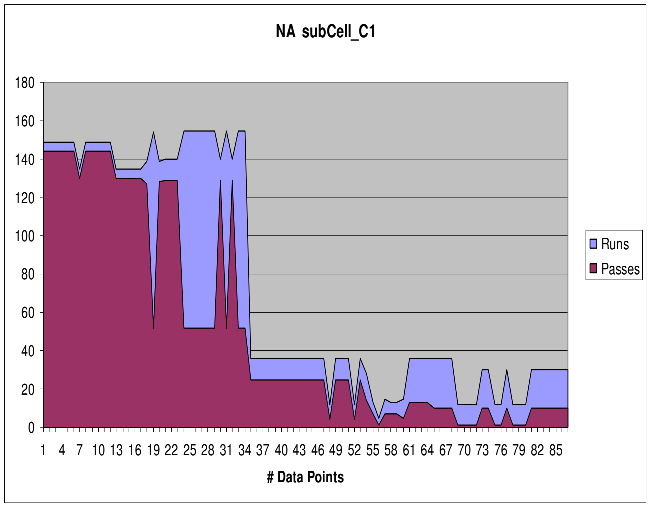

Figure 5. Results for tool NA on subcell C1

In some cases, the data clearly shows that a tool has not been run on a given cell, but when a tool starts to be run on that cell, the number of failures is non-existent. This is not the norm, and usually only occurs in cells of small size. Figure 6 is an example of such phenomenon. This type of phenomenon is observed across all tools, but usually only when the number of cells in the hierarchy is in the order of size ~ 10.

It is important to note that some graphs show *"glitches"* in the data. Glitches are observed when you have steep drops in totals. These glitches are due to events such as the loss of network connectivity, tools not being available, cells being locked and left that way overnight while engineers work on them, etc. Figure 7 shows some of these glitches.

In order to evaluate $\boldsymbol{H_0}$ "All CAD tool failure rates of a microchip block appear to correlate over a period of time", we need to observe what happens to every cell per tool over a period of time. Table 1 breaks down the data per tool and per cell. It summarizes trends observed for each.

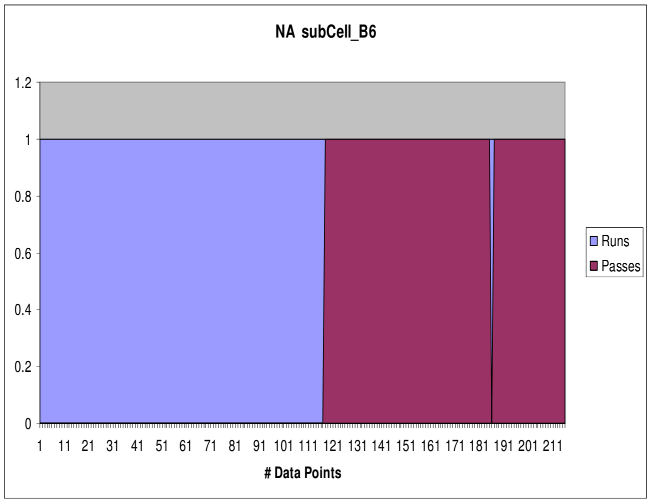

Figure 6. Results for tool NA on subcell B6

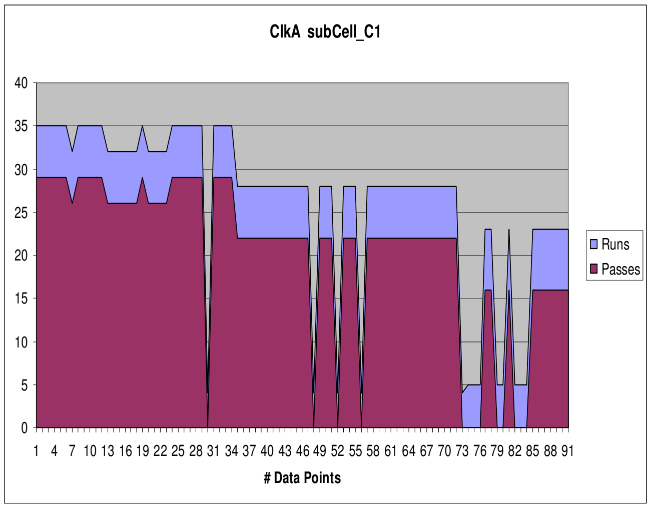

Figure 7. Results for tool ClkA on subcell C1

For every tool we calculate the total number of cells and subcells where the number of failures is decreasing over time as a percentage of the total number of cells or subcells available. In other words, the general trend is examined. If the two curves from our graphs tend to come together in the latter stages of the cell, then this counts as a cell whose failures decrease over time, if the two curves exhibit the opposite behavior then this counts as a cell whose failures increase over time. Finally, it is expected that for some cells, no noticeable trend will be identified; thus, a special count for these cells is also kept.

Table 1 displays the findings. Note that the total number of subcells studied per tool is 20. By analyzing this table we can clearly see that the total number of failures for every tool studied appears to decrease over time. Table 2, gives additional insight by breaking up the data not only by tool, but also by sub cells.

	Total # of subCells where the # failures **decrease** over time	Total # of subCells where the # failures **increase** over time	No noticeable changes in trend observed
CA_1	13	0	7
CA_2	10	2	8
ClkA	7	0	13
DRC	15	1	4
NA	15	2	3

Table 1. Summary of trends observed per tool. The total number of subcells studied per tool is 20.

	Total # of subCells where the # failures **decrease** over time	Total # of subCells where the # failures **increase** over time	No noticeable changes in trend observed
CA_1			
Cell A	5/5		
Cell B	3/6		3/6
Cell C	3/3		
Cell D	2/5		3/5
Cell E			1/1
CA_2			
Cell A	4/5		1/5
Cell B	2/6	1/6	3/6
Cell C	2/3	1/3	
Cell D	2/5		3/5
Cell E			1/1
ClkA			
Cell A	4/5		1/5
Cell B	2/6		4/6
Cell C	1/3		2/3
Cell D			5/5
Cell E			1/1
DRC			
Cell A	5/5		
Cell B	4/6	1/6	1/6
Cell C	3/3		
Cell D	3/5		2/5
Cell E			1/1
NA			
Cell A	5/5		
Cell B	6/6		
Cell C		1/3	2/3
Cell D	3/5	1/5	1/5
Cell E	1/1		

Table 2. Trends observed per cell and per tool

The entries display the number of subcells belonging to that category (x) out of a possible total (y), i.e. x/y. Out of a possible 20 subcells studied per tool an average of 12 subcells show decreases in failures, 1 shows an average of increases in failures, and an average of 7 show no apparent trend.

By looking at each separate tool, we can also observe a similar trend. Tools DRC and NA show the strongest evidence of failure reduction over time, while tool ClkA shows slower progress toward failure reduction, but it nevertheless appears to be moving in the correct direction.

Even though all tools clearly show a trend where failures are decreasing, Table 2 allows us to see the trends observed per cell rather than just by tool. Cell A clearly shows the strongest evidence to support the removal of failures as the cell block matures through the lifecycle of the microchip, regardless of which tool is run on the cell. Cell A has an average of 4.6 out of a possible 5 subcells where the trends point to a decrease in failures.

Some cells, such as E show no noticeable changes in all but tool NA, which shows hat the failures are decreasing. Also, it is important to note that some cells, such as B, showed an increase in failures over time for tools CA_2 and DRC, and cell C also showed an increase in failures over time for tools CA_2 and NA. This data points should allow designers to concentrate on these particular cells.

Based on the information mined from the tools there is clear evidence to support $\boldsymbol{H_0}$, that "All CAD tool failure rates of a microchip block appear to correlate over a period of time". We found that as the process of developing a microchip moves through the lifecycle, the number of failures appears to decrease regardless of which of the five CAD tools is used.

Additional evidence of this trend can be observed in Figure 8, where we display for each tool, the total number of passing cells in the Y-axis, versus each cell that is broken up by the total number of subcells in that cell. We can observe that tools CA_1, CA_2, DRC, and NA have a stronger correlation to each other than ClkA.

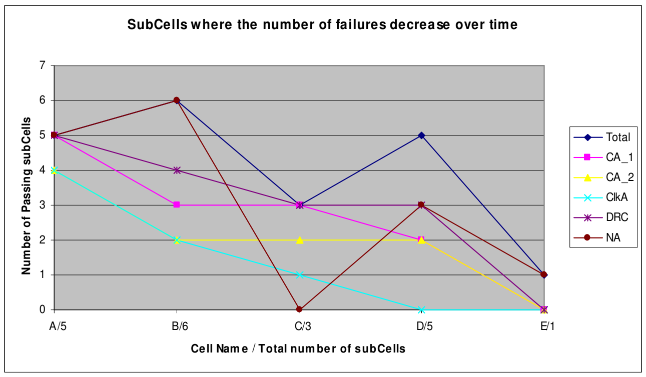

Figure 8. Trends

VI. Threats To Validity

In this section we analyze, if possible, the threats to the validity of the study. There are four types of threats to the validity of any empirical study. They are construct, content, internal, and external threats [2].

Construct validity refers to the meaningfulness of measurements, and to validate this you must show that the measurements are consistent with an empirical relation system. An empirical relation system is an intuitive ordering of the data in terms of the attributes of interest. Clearly the dependent variable in this case study is the number of passes or non-failures observed over a period of time for each of the five different CAD tools studied. The number of passes is also clearly related to the level of quality of a cell at some given point in the lifecycle. The construct validity of this study can be enhanced by finding other measurements, such as effort spent per cell and per CAD tool over a period of time.

Content validity refers to the adequate representation of the content. In order for this study to capture the notion of the quality of a block, additional measures beyond counting the number of failures is necessary, however, no additional data is necessary in order to evaluate whether multiple CAD tools tend to behave similarly in terms of finding failures at different checkpoints in the lifecycle of a microchip. One can argue that the sample size of this data is rather small, and the study could benefit from not only additional data, but also from studying additional tools. Additionally, the cells studied are small in terms of the number of subcells or the number of transistors that make up these cells, and one must also trust that the legacy data used by this case study is not flawed. Of particular interest are the *"glitches"* in the data. Glitches are anomalies that are not directly related to the quality of the cells themselves, but rather, to external events such loss of network connectivity, errors in the source control that house the various cells, unavailability of a CAD tool for any number of reasons, etc. Finally, not all the cells studied had the same date ranges available for study for each of the CAD tools. In fact, as little as just two days of available data points were available for some cells.

Internal validity focuses on the cause and effect relationships. In this study one can try to determine whether decreased number of failures is directly related to the milestones in the lifecycle of the block. The data shows that the number of failures encountered is correlated according to CAD tool, and also one can infer that temporal precedence does exist, in fact it is necessary, as the lifecycle of a microchip matures. In other words, more failures exist at an earlier rather than later stage of the lifecycle.

Finally, external validity refers to the ability to generalize results. Based on the small number of tools examined, and the small date ranges available to only twenty subcells examined, one can not generalize from this data. The study does however provide another data point that shows that failure prevention is necessary as a microchip matures.

VII. Conclusion

The goal of this case study was to analyze cells of a microchip using various CAD tools in order to help us evaluate the following hypothesis,

***H_0*:** All CAD tool failure rates of a microchip block appear to correlate over a period of time.

Significant evidence was found to support ***H_0***. This suggests that most CAD tools are correlated and are thus dependent on the microchip data. As block data is repaired other CAD tools tend to pick up on some of the fixes. Similarly, many efforts do proceed in parallel, with many CAD tools executed concurrently.

References

[1] A. Avizienis, *"Toward Systematic Design of Fault-Tolerant Systems,"* IEEE Computer, pp. 51-58, April 1997.

[2] Bieman J.M, Straw G., Wang H., Munger P.W., Alexander R. *"Design Patterns and Change Proness: An Examination of Five Evolving Systems,"* IEEE-CS 9th International Software Metrics Symposium, Metrics 2003, September 2003.

[3] CurveExpert Statistical software. http://curveexpert.webshop.biz

[4] C. Niessen, *"Hierarchical design methodologies and tools for VLSI chips,"* Proceedings of the IEEE, pp. 66-75, Vol. 71, Issue 1, Jan 1983.

[5] OpenAccess coalition. http://www.si2.org

[6] A. Schroter, T. Zimmermann, and A Zeller, *"Predicting Component Failures at Design Time,"* Proceedings of the 5th ACM-IEEE International Symposium on Empirical Software Engineering, pp. 18-27, Sept 2006.

[7] A. Somani, N. H. Vaidya, *"Understanding Fault Tolerance and Reliability,"* IEEE Computer, Guest Editor's Introduction, pp. 45-50, April 1997.

[8] R.L. Wadsack, *"Design Verification and Testing of the WE32100 CPUs,"* IEEE Design and Test, pp. 66-75, August 1984.

[9] V. Zolotov, A. Grinshpon, A. Dasgupta, S. Sirichotiyakul, B. Orshav, C. Oh, D. Blaauw, G. Braca, R. Levy, *"ClariNet: A Noise Analysis Tool for Deep Submicron Design,"* IEEE Computer Society, 37th Conference on Design Automation, DAC '00, pp. 233-238.