PLACE IN RETURN BOX to remove this checkout from your record.
TO AVOID FINES return on or before date due.
MAY BE RECALLED with earlier due date if requested.

DATE DUE	DATE DUE	DATE DUE
NOV 1 4 2005 0 5 2 2 0 5		
MAR [illegible]		
MAR 0 5 2008		
1 1 1 1 0 7		

A SELF-STRUCTURING ANTENNA PROTOTYPE

By

Bradley Thomas Perry

A THESIS

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

MASTER OF SCIENCE

Electrical and Computer Engineering

2002

ABSTRACT

A SELF-STRUCTURING ANTENNA PROTOTYPE

By

Bradley Thomas Perry

This thesis presents a class of antennas, called self-structuring antennas, from an experimental perspective. The self-structuring antenna (SSA) is a concept that involves creating a wideband antenna that can automatically adjust its electrical shape in response to changes in its electromagnetic environment. This is done through the use of simple on/off switches employing binary search algorithms to find optimal configurations for the antenna.

Topics covered include a general overview of the self-structuring antenna to familiarize the reader with the concepts. A brief history of the SSA and some practical uses are also covered. Next, a prototype SSA is discussed. This includes the building and testing of the antenna, including the application of a genetic algorithm (GA) to the control of the prototype. Testing of this prototype for UHF and VHF television reception follows, providing a proof of concept for both the SSA and GA.

Once the general framework of the self-structuring antenna has been laid down through the above work, consideration is given to the occurrence of switch malfunctions. Switch failures and their consequences are considered for single switches, as well as groups of switches. Finally, conclusions drawn from this research are discussed.

For My Family

ACKNOWLEDGMENTS

Throughout the course of this research, there have been many people who have helped me to keep my sanity. Thanks to everyone who helped me out along the way.

First, I would like to thank my professors who helped me to find my way through graduate work. Thanks to Dr. Edward Rothwell for being my major professor and teacher, to Dr. Leo Kempel for helping me to decide to keep on the educational track, and Dr. Dennis Nyquist for the instruction and example.

Thanks to friend and mentor, Dr Mike Havrilla, for taking time to teach me things that I would not have learned otherwise. Your dedication to EM is nothing short of admirable.

I would also like to thank friend and fellow graduate student Tony Lambert for his help in C++ programming for the switch failure analysis presented in this thesis. Without your help I wouldn't have been able to complete this analysis in such an efficient manner.

Thanks is also in order for Dr. Christopher Coleman for help with LaTeX and for the style file that made my life easier while writing this thesis.

Many thanks to Professor Percy Pierre for supporting my studies through the graduate office fellowship. Additional thanks are in order for Dr. Lou Nagy and Dr. Mark Krage of Delphi Research Labs for their support of SSA research at MSU.

Most of all I would like to thank my wife Beckie and the rest of my family for their support through all of my studies. Your constant recognition and encouragement is greatly appreciated.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

KEY TO SYMBOLS AND ABBREVIATIONS

AGC: Automatic Gain Control

CPLD: Complex Programmable Logic Device

EA: Evolutionary Algorithm

GA: Genetic Algorithm

PCB: Printed Circuit Board

SSA: Self-Structuring Antenna

SWR: Standing Wave Ratio

UHF: Ultra High Frequency

VHF: Very High Frequency

CHAPTER 1

INTRODUCTION

This thesis introduces and investigates a new class of antennas called "Self-Structuring Antennas" or SSAs. The SSA is a new concept developed by Dr Edward Rothwell at Michigan State University. These antennas are designed to change their electrical shapes in response to changes in their electromagnetic environment. This opens the door for many uses of the SSA, especially in mobile applications, where the electromagnetic environment is constantly shifting.

An overview of Self-Structuring Antennas is given in Chapter 2, including a description and basic operation of the SSA. Also included in this chapter is a brief history of the SSA and some possible applications of the technology.

Chapter 3 covers the construction and operational testing of the first printed circuit SSA built at Michigan State University.

The use of Genetic Algorithms, or GAs, for control of the SSA is covered in Chapter 4. Also included is testing done using the prototype SSA. This testing served as a proof of concept for the use of GAs in control of the SSA and has important implications for future control through evolutionary algorithms.

With applications in mobile communications, where replacement of switches may be difficult, it is important to understand how the SSA will respond when a switch, or group of switches, malfunctions. For this reason, a study of the effects of switch failures on the performance of the SSA was conducted. The results of this study are the subject of Chapter 5.

Chapter 6 serves as a conclusion to this thesis. Contributions to this research are discussed, along with future work needed to further the understanding of this new technology.

CHAPTER 2

SELF-STRUCTURING ANTENNA CONCEPTS AND IDEAS

Traditional antennas face many unseen challenges when placed in their end application. This is especially evident in automobile applications where rapid changes in a car's direction and the addition of aftermarket products can greatly affect performance of the antenna. Primarily this is due to the fixed geometrical configurations of traditional antennas. These antennas require costly measurement and testing in order to perform well in a particular environment. The result is an antenna that requires substantial engineering costs for a design that is very specific in application and whose performance can degrade substantially due to changes in environment. From these limitations, the self-structuring antenna has evolved.

The self-structuring antenna is a concept that involves creating a wideband antenna that can automatically adjust its electrical shape in response to changes in its electromagnetic environment. A brief history of the SSA is given in Section 2.1. Concepts behind the operation of the SSA are the subject of Section 2.2. Finally, some practical applications of the SSA are given in Section 2.3.

2.1 Origination of the Self-Structuring Antenna

The self-structuring antenna was invented by Dr. Edward Rothwell at Michigan State University in 1997. Several prototypes were constructed and testing showed that the SSA had promising characteristics. Following these experiments, Dr. Rothwell applied for a U.S. patent with the aid of Michigan State University. U.S. patent number 6,175,723 was issued to Michigan State University in January of 2001. [1]

Research on the SSA began to accelerate in 1998-1999 when Dr. John Ross began intensive numerical simulations. Dr. Christopher Coleman began research on the SSA in 1999, culminating in a Ph.D. dissertation in 2002 [1]. Delphi Research Labs got involved in SSA research in 2000 and began funding the research at Michigan State in 2001. The first conference papers were presented at the 2000 IEEE AP-S / URSI conference in Salt Lake City, Utah [2],[3]. A paper on this subject has also been accepted for publication by the IEEE Antennas and Propagations magazine [4].

The author became involved in SSA research as an undergraduate through an electrical engineering capstone project in the fall of 2000. Subsequently, he continued research on the SSA as a graduate student of Dr. Rothwell. Several more conference papers regarding the SSA were presented at the 2001 IEEE AP-S / URSI conference in Boston, Massachusetts [5],[6], as well as at the 2002 IEEE AP-S / URSI conference in San Antonio, Texas [7],[8].

Research on the self-structuring antenna continues at Michigan State University, John Ross and Associates, and Delphi Research Labs.

2.2 Overview of the Self-Structuring Antenna System

The self-structuring antenna system consists of several components. These include hardware and software used to control the states of the antenna, as well as the antenna itself. Each component plays an important role in the overall performance of the SSA. Figure 2.1 shows a general block diagram of the SSA system. The main components of the SSA are shown, including the antenna skeleton, or template, a receiver, and a microprocessor based controller. This diagram represents the minimal components necessary for the SSA to work as intended. Occasionally, additional control or measurement hardware is added when necessary.

The following sections discuss the components of the SSA in greater detail. Section 2.2.1 discusses the SSA template, focusing on the current versions of the SSA that have been constructed using printed circuit boards (PCBs). Microcontrollers, computers, and PCB control boards employing field programmable gate arrays (FPGAs) have been used in controlling various versions of the SSA. The different control methods for the SSA are the subject of Section 2.2.2.

2.2.1 Self-Structuring Antenna Skeleton

The self-structuring antenna skeleton, or template, is the primary structure for receiving or transmitting electromagnetic energy in the SSA system. The SSA skeleton consists of radiating elements, switches, a feed network, and headers for connection to the control hardware. In general, the radiating elements can be wires, patches, or other radiating bodies interconnected by controllable switches [1]. The printed circuit SSAs built during this research have used traces on the top side of a PCB as their intended radiating elements. Figure 2.2 is the top of a typical SSA skeleton showing the radiating elements, as well as other hardware. The antenna lines on the SSA are interconnected using electromagnetic relays, which are shown as rectangles in Figure 2.2. Control of the relays is accomplished through connections made to the headers located on the antenna skeleton. The wiring between the relays and control network are routed on the bottom side of the SSA template. Each SSA built in this manner also required the use of driver hardware to supply the current necessary for switching of the relays. In some cases the drivers were located on the SSA skeleton, in others they were located on a separate control board. Further discussion is left for Section 2.2.2. Finally, the feed networks for the printed circuit SSAs built to date have consisted of standard 4:1 wideband television baluns. These have been used as the feed network because the applications of the current SSAs are to mobile television reception [5]. The end application for future generations of the SSA will need to be considered in designing feed networks.

2.2.2 Control of the Self-Structuring Antenna

A microprocessor in the form of a microcontroller or computer is used to control the self-structuring antenna. In each case a receiver is employed to measure the performance of the SSA in a given configuration. The information attained by the receiver can be in the form of signal strength, input impedance, or any other appropriate measurable quantity [1]. This information is then used by the microprocessor to make decisions on the future states of the SSA. The decision making process generally employs the use of a smart algorithm, such as a genetic algorithm as discussed in Chapter 4.

Specific details on one control system used for the SSA can be found in Section 3.1.2.

2.3 Practical Applications of Self-Structuring Antennas

Current research activity on the self-structuring antenna is focused on an application to mobile communications. Particularly, the research reported in this thesis focuses on television reception in a changing environment, with application to automobiles. Other practical applications to mobile communications include multitasking antennas to do the jobs that currently require multiple antennas in automobiles, as well as adaptable antennas for cellular phones and hand-held radios.

The applications to which the self-structuring antenna may be applied are numerous, many of which probably haven't even been conceived. This is the beauty of the SSA – the engineering costs of design are low, while the opportunities for use are high.

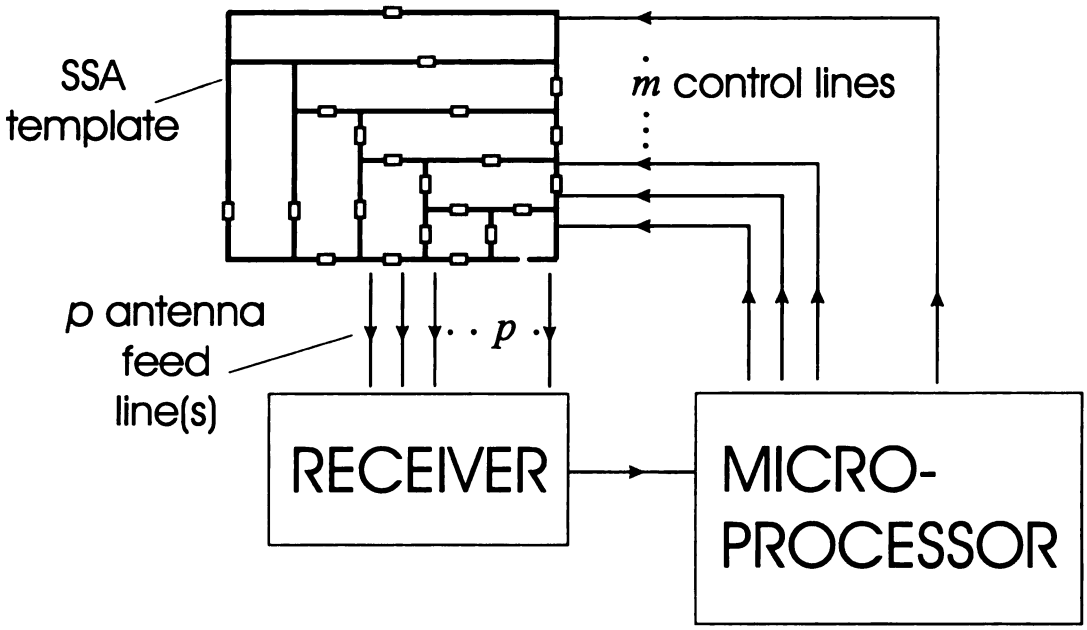

Figure 2.1. Self-Structuring Antenna System (Reprinted with permission [1])

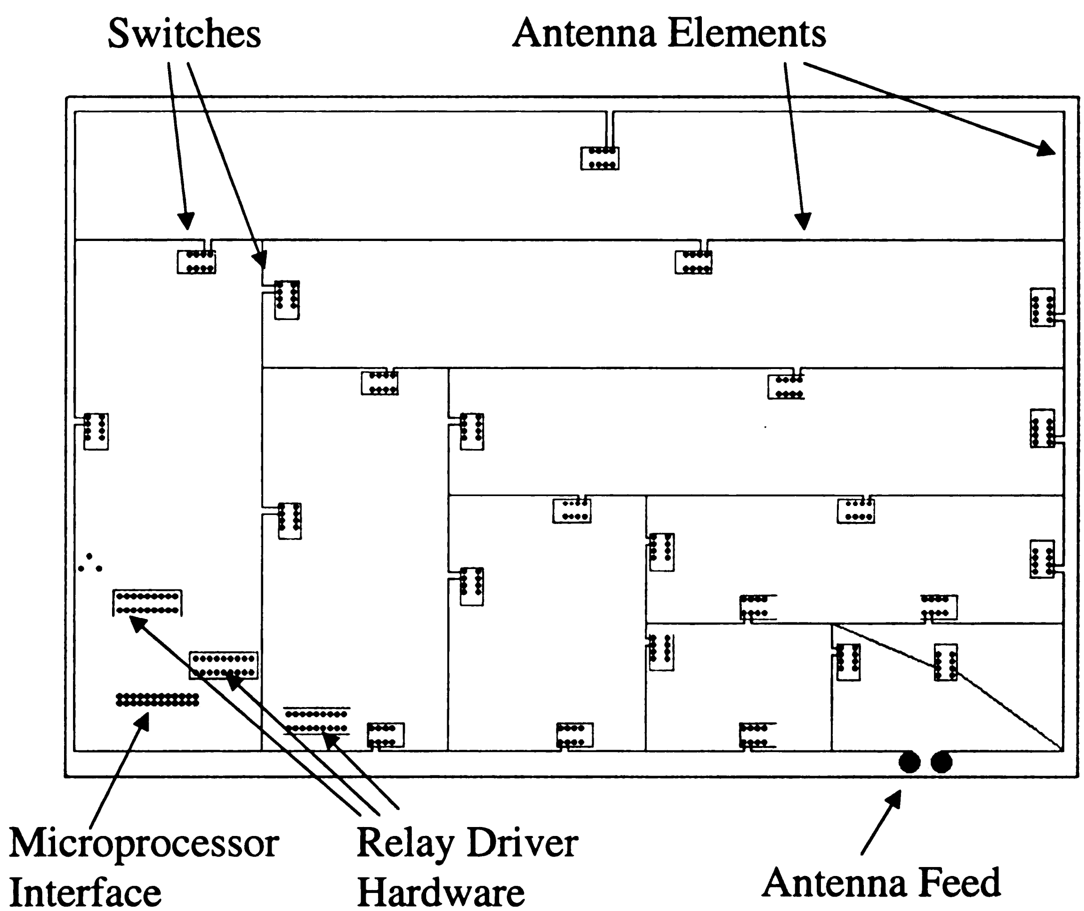

Figure 2.2. A Typical Self-Structuring Antenna Skeleton

CHAPTER 3

30 SWITCH PROTOTYPE CONSTRUCTION

Design and testing of the self-structuring antenna in printed circuit board (PCB) form began with a 30 switch prototype. This board, along with subsequent versions of the antenna were created using ORCAD Capture and Layout software (www.orcad.com). They were then manufactured through Advanced Circuits Corporation (www.4pcb.com). The creation of circuit schematics and board layouts for the self-structuring antenna are covered in Section 3.1 along with production of the SSA prototype. Section 3.2 covers the functional testing of the first printed circuit board SSA.

3.1 Layout of the Self-Structuring Antenna

Layout and construction of the self-structuring antenna began with printed circuit board layouts created through the use of ORCAD Capture and Layout software. The circuit schematics for the antenna and control board were created using the Capture software; these are included as Figure 3.1 and Figure 3.2, respectively. Once the circuits were laid out using ORCAD Capture, netlists were created to tell the computer how components should be connected. Next, footprints were created for the various components to be placed on the circuit board. Footprints are a representation of the connection of components to the circuit board and are used in connecting the traces of the PCB. Finally, the components were placed in their positions within the Layout software. Details regarding the layout and schematics of the antenna and control boards for the 30 switch prototype are the subject of Sections 3.1.1 and 3.1.2. Section 3.1.3 explains the production process for the self-structuring antenna prototype.

3.1.1 Antenna Board

The antenna skeleton laid out for this prototype includes 30 electromechanical relays, connections for a television balun, and several headers, as shown in Figure 3.3. Driver and control hardware were placed on a separate circuit board as described in the next section. Placement of the relays on this version of the SSA included some relays to section off large portions of the board and some to create antenna elements of various shapes and sizes. It is important to note that when sections of the board are cut off from the feed structure, they still can play a part in reception through coupling. In other words, even though a portion of the board is cut off from the rest, it is still present and still effects the performance of the antenna. Also coupling into the antenna and effecting the performance are the control lines that are routed on the backside of the PCB. Studies as to the effect of the presence of the control lines on reception have not been considered; it is only noted that they will in fact play some role. The completed antenna board layout for the 30 switch SSA is included as Figure 3.4.

3.1.2 Control Board

The 30 switch SSA prototype required the use of a control board in order to reset the latching relays used in the design. Included on this board are two complex programmable logic devices (CPLDs) used to invert the control signals and reset the relays, as well as drivers needed to provide current for turning the relays on and off. The code used to program the CPLDs has been included as Appendix D. As in the case of the antenna board, the control board was laid out using ORCAD Capture and imported into ORCAD Layout for routing of traces on the PCB. The completed circuit board layout is shown in Figure 3.5.

3.1.3 Production of the SSA Prototype

Once the antenna and control board layouts were completed, the printed circuit boards needed to be fabricated and components needed to be attached.

The control board was fabricated at Michigan State University through the ECE Shop. This was done using the Quick Circuit T-Tech Prototyping Machine, which etches the traces onto a double sided copper board. Once the etching of the board was complete, components were attached through hand soldering. The completed control board is shown in Figure 3.6.

The antenna board was fabricated through Advanced Circuits Corporation (www.4pcb.com) because of their ability to remove excess copper from the PCB. This was necessary for proper operation of the antenna. Again, once the board was etched the components were attached through hand soldering. The completed antenna board is shown in Figure 3.7.

3.2 Functionality Testing of the Self-Structuring Antenna

Once the SSA prototype was built, it was important to verify that it would work as expected. This was done in various stages, first through testing of the control board for proper functionality, then through testing of the entire system. Testing of the control board is covered in Section 3.2.1. Testing of the entire system, with emphasis on functionality of the antenna skeleton, is left for Section 3.2.2.

3.2.1 Control Board Testing

Testing of the control board included the construction of a test circuit to verify proper functionality. The test circuit was built on a protoboard using LEDs and 250Ω resistors, chosen to match the resistance of the relays used on the antenna board. A schematic of this test circuit is included as Figure 3.8. Headers identical to those placed on the antenna board were connected to the protoboard and LEDs and resistors were wired in series to simulate the switching of the relays. Signals were applied to the control lines to turn the LEDs on and off. This testing proved successful and the control board worked as expected.

3.2.2 Antenna Skeleton Testing

Testing of the antenna skeleton was a matter of connecting the entire SSA system together, as explained in Chapter 2. Once the system was connected together, signals were sent to the antenna skeleton to turn on certain relays that were subsequently checked for functionality using a digital multimeter. Once all of the relays had been checked, the SSA prototype was ready to be tested in an application. This was done using a genetic algorithm and tests were run to measure the SSA's performance for the reception of UHF and VHF television signals. The genetic algorithm and its testing are the subjects of Chapter 4.

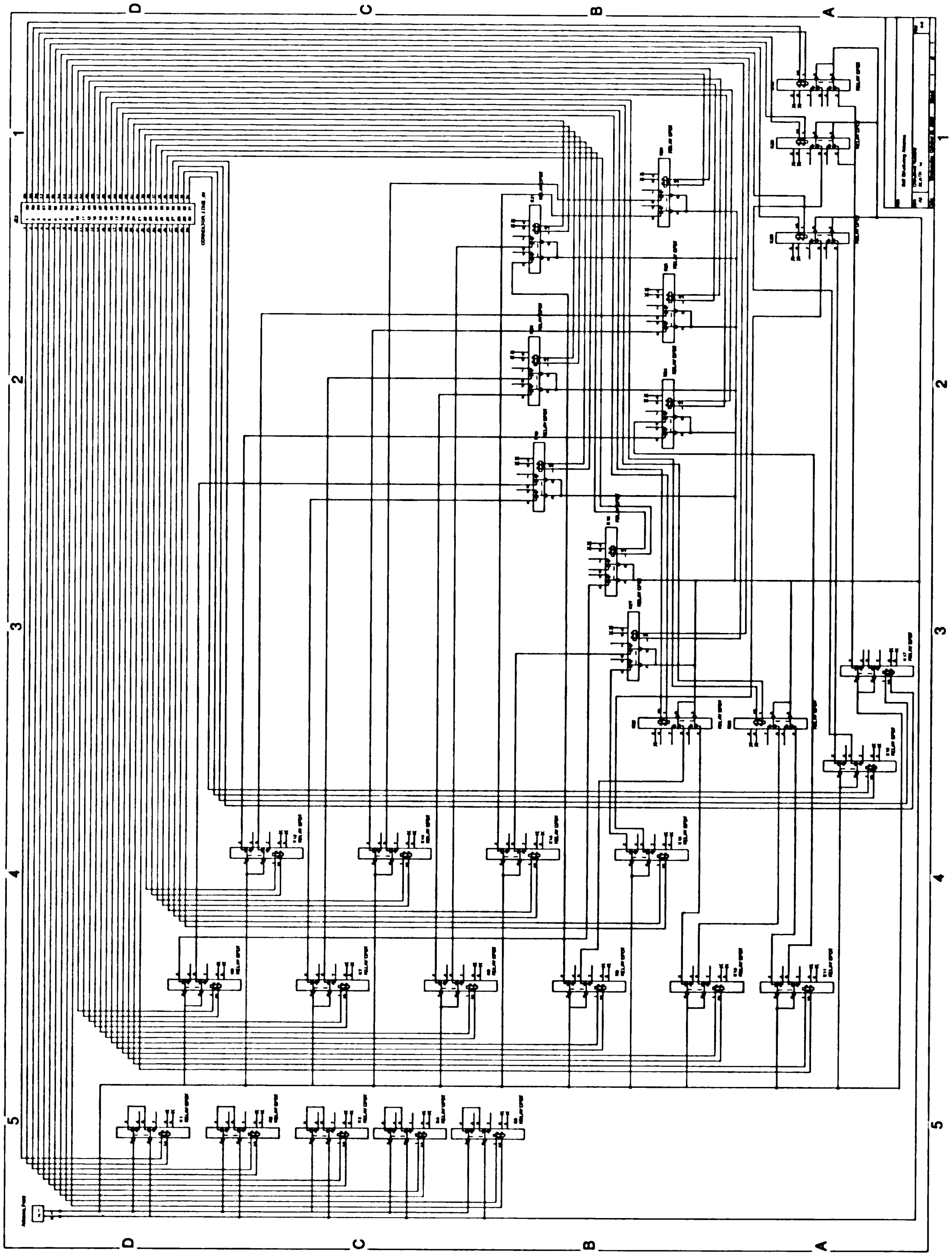

Figure 3.1. Antenna Board Schematic

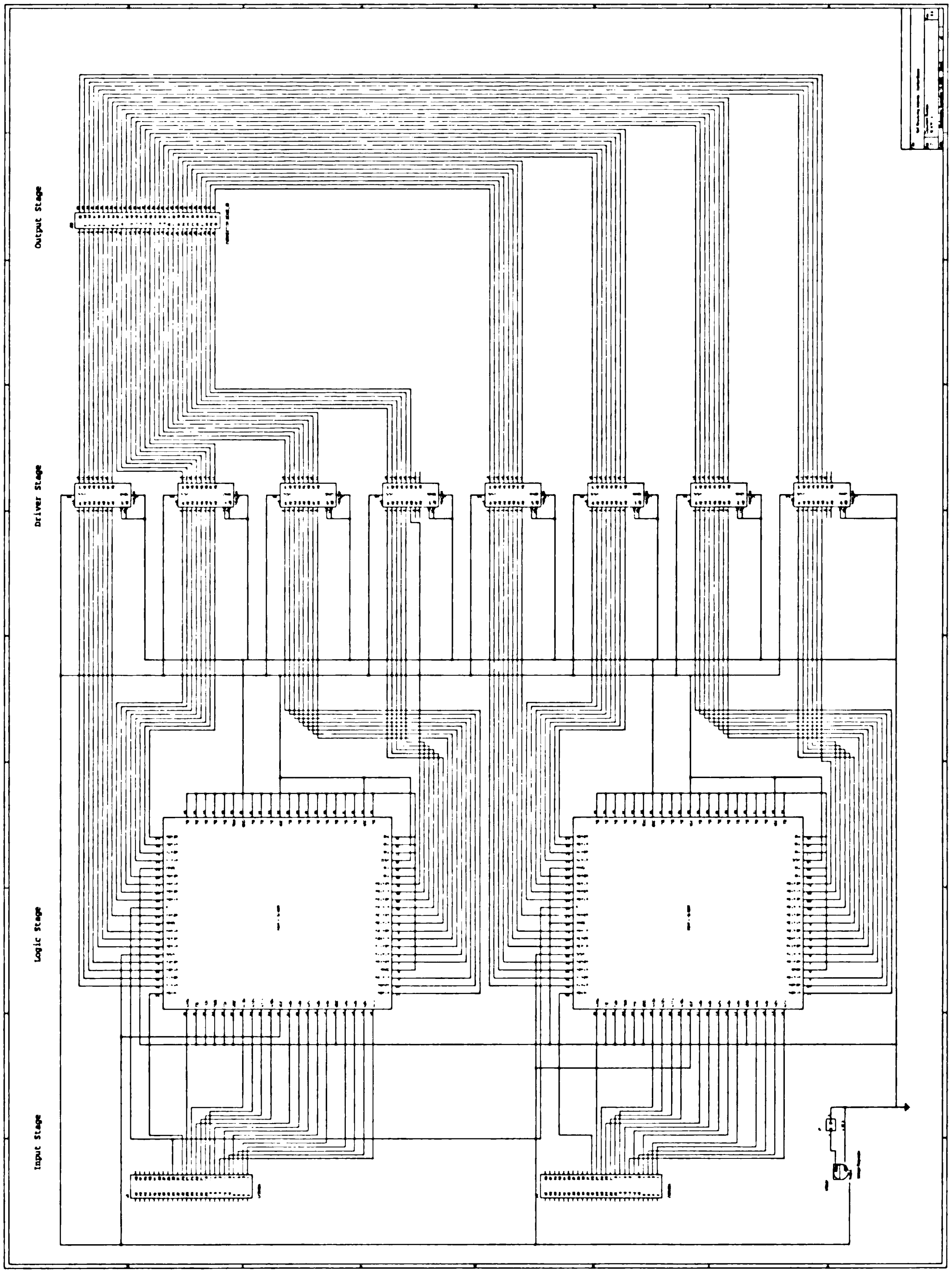

Figure 3.2. Control Board Schematic

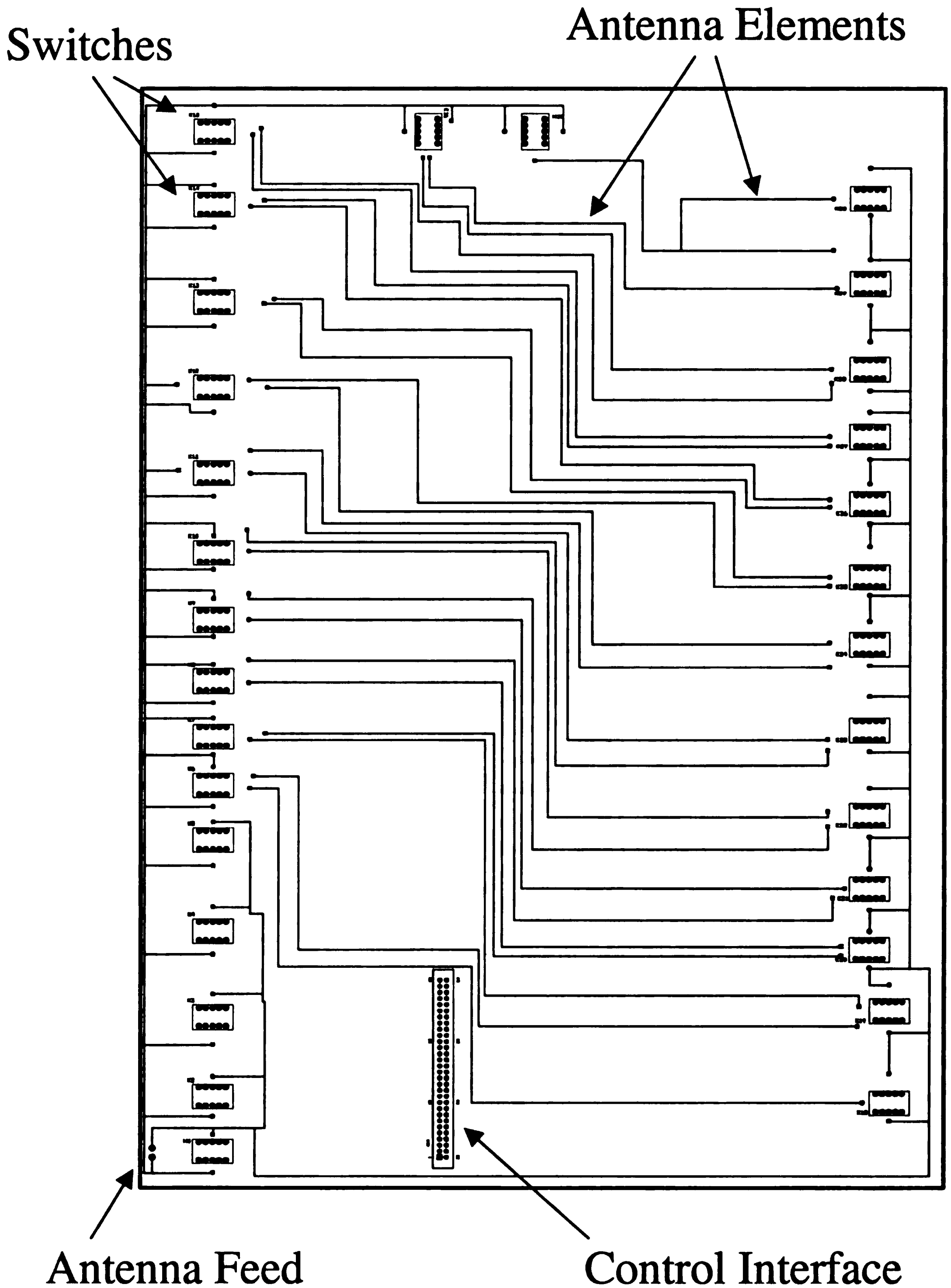

Figure 3.3. Self-Structuring Antenna Skeleton

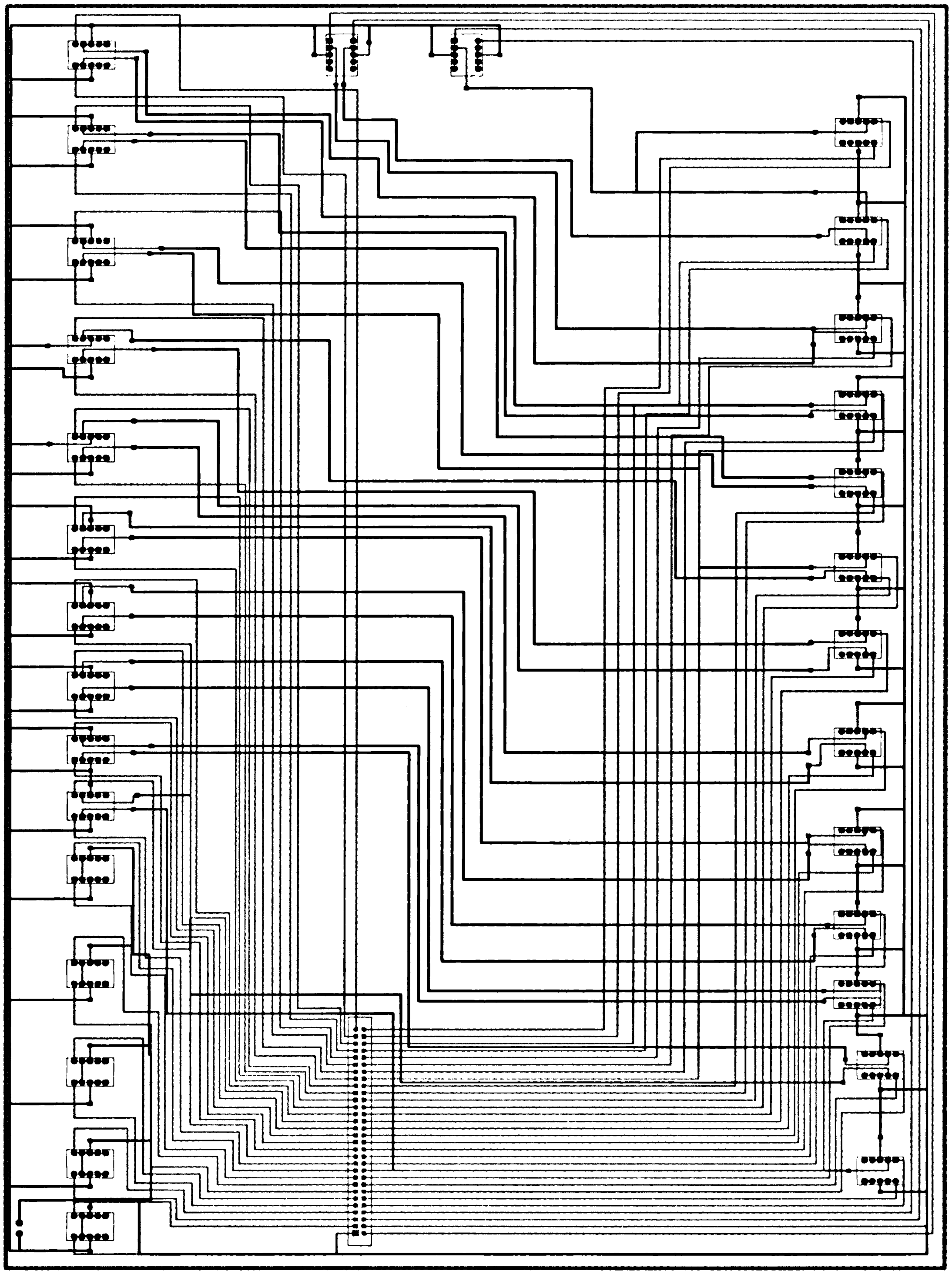

Figure 3.4. Antenna Board Layout

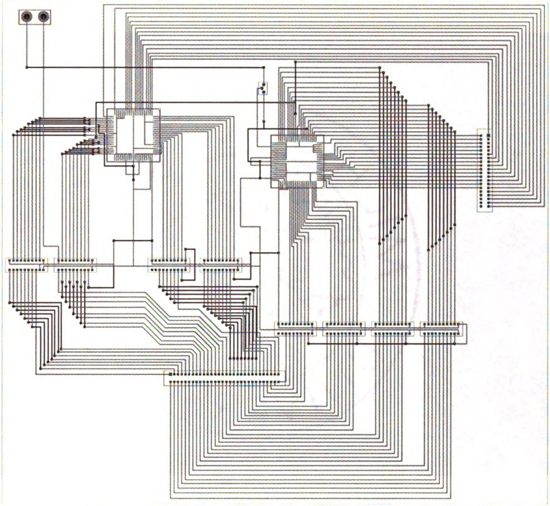

Figure 3.5. Control Board Layout

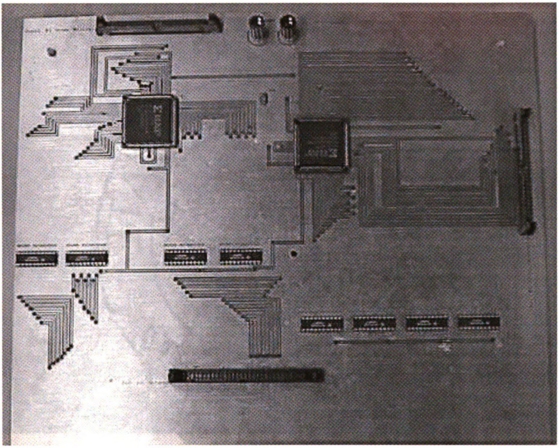

Figure 3.6. Completed Control Board

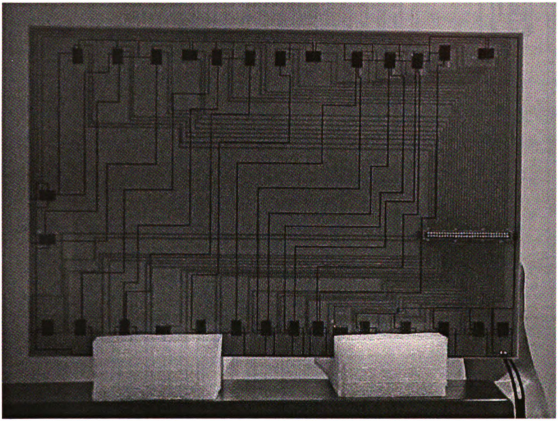

Figure 3.7. Completed 30-switch Self-Structuring Antenna

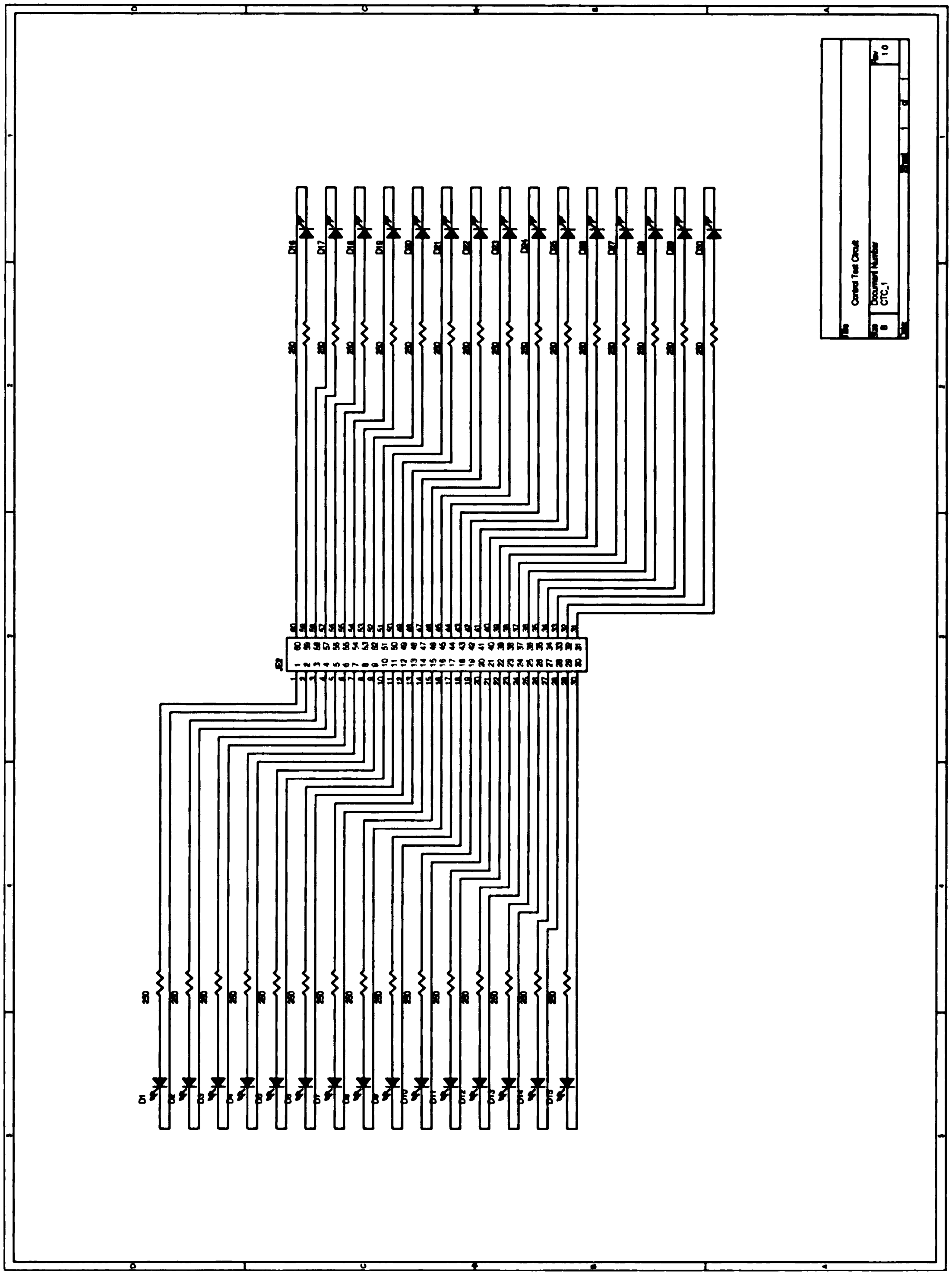

Figure 3.8. Control Board Test Circuit Schematic

CHAPTER 4

GENETIC ALGORITHM FOR THE SSA

The self-structuring antenna prototypes built during this project consist of 24-30 electromagnetic relays used to control restructuring of the antenna skeleton. With n relays, the number of electrical configurations available to the antenna is 2^n, which is between 1.67×10^7 and 1.07×10^9 combinations. Due to the large number of configurations available to the self-structuring antenna, the use of a search algorithm to handle the choice of configurations becomes essential. The use of simple on/off switches in the design of the self-structuring antenna makes a binary search algorithm, such as a genetic algorithm, well suited for this task. The genetic algorithm used to control the SSA is included as Appendix A.

4.1 Genetic Algorithm Concepts

Evolutionary algorithms, such as the genetic algorithm, use principles modeled after nature to find a best fit solution to a given problem. This section will discuss the principles behind the genetic algorithm, as well as application and specialization to the self-structuring antenna.

4.1.1 General Concepts

Genetic Algorithms, or GA's, consist primarily of three components modeled after the natural selection process: preselection, selection, and crossover or breeding. In addition to these processes, the makeup of individuals comprising the population needs to be specified. These individuals, commonly known as chromosomes, are built from a string of ones and zeros. In the case of the self-structuring antenna, the bit-string that makes up the chromosomes represents the states of the switches, ones being switches in the 'on' position and zeros in the 'off' position. Each of the above components play an important role in the functionality of the algorithm and will be covered in more detail in the following subsections.

4.1.1.1 Initialization

The first step in the use of a GA is creation of an original population of chromosomes. The preferable method of filling this population is through a random process. The original population was filled in a random manner in the case of the SSA. Once this original population has been created, the GA is ready to be initiated.

4.1.1.2 Preselection

The preselection process involves choosing which individuals in a given population will be allowed to breed in creating the next generation. The genetic algorithm for the SSA uses the voltage present at the input to the automatic gain control (AGC) of the television to evaluate the fitness of a given configuration. Consequently, individuals in the population with the highest levels of fitness contribute more to the next population than those with lower fitness levels. In other words, the strongest individuals are allowed to breed more than other individuals in order to produce better offspring. The fitness function used in the GA for the SSA was defined as

$$f(v) = \frac{0.4}{(v - v_{ideal})^2 - 0.4} \tag{4.1}$$

where v_{ideal} is the ideal voltage at the input to the AGC of the television based on measurements made using a dipole antenna attached to the television. v is the measured voltage at the input to the AGC using the self-structuring antenna in a given configuration. The above fitness function was chosen to give a value of unity for what was determined to be a perfect picture on the television, while a value of 0.9 represented a good picture. The value of the fitness function was then used to fill a breeding pool based on the relative fitness of an individual compared to the average fitness of the population. If the population fails to fill the breeding pool completely, random selection is used to bring the number of individuals up to the size of the population. This completes the preselection process and the selection process begins.

4.1.1.3 Selection

The selection process involves randomly choosing chromosome pairs from the breeding pool created in the preselection process to breed, or crossover, in order to create the next generation of combinations. This process is accomplished through stochastic remainder sampling without replacement, allowing all members of the breeding pool to create offspring [9]. Throughout this process, individuals are randomly selected from the breeding pool to form a mating pair. Once an individual has been chosen to be one half of a mating pair, that individual is removed from the breeding pool and the pool size decreases. The process is repeated until there are no members left in the breeding pool, having all been paired off for the crossover, or breeding, process.

4.1.1.4 Crossover and Mutation

The breeding process consists of mating pairs chosen in the selection process undergoing a crossover operation. This operation consists of two parents being split at a random location along the chromosome, and recombining with a probability of P_{cross}. A common value chosen for P_{cross} is in the range of 0.6-0.8; the chosen value for the SSA's GA was 0.7. This means that the split chromosomes will recombine with each other 70% of the time, giving offspring chromosomes made up of some information from one parent and some from the other. The rest of the time the chromosomes will not crossover, giving offspring identical to the parents. Once this process is complete, there is a new population, identical in size to the original, with offspring made up of the parents of the previous generation.

After the breeding process, the new population is subjected to a mutation process in which single bits can be randomly changed with a probability of P_{mut}. The common practice is a choice of $P_{mut} = \frac{1}{K}$, where K is the population size. This was the chosen value of P_{mut} for the SSA. Once this mutation process is complete, the new population takes the place of the old and the algorithm is repeated, starting with the preselection process.

4.1.2 Specialization to the Self-Structuring Antenna

Several modifications were made to the standard genetic algorithm in order to speed convergence of the population to a global maximum. The first of these specializations was the use of a selective linear scaling of the fitness given by [9]

$$\tilde{F} = aF + b \tag{4.2}$$

where F is the fitness of a given configuration and $\tilde{F}_{avg} = F_{avg}$. If the maximum fitness, F_{max}, was less than 1.5 times the average fitness, F_{avg}, then no scaling was done. This was the case of having no individuals dominating the population. When a given configuration appeared to dominate the population, scaling was performed to give $\tilde{F}_{max} = c_{mult}F_{avg}$, where c_{mult} is the scaling factor desired for the best population member. In order to fulfill the above relations, a and b were chosen as

$$a = \frac{(c_{mult} - 1)F_{avg}}{F_{max} - F_{avg}} \tag{4.3}$$

$$b = (1 - a)F_{avg} \tag{4.4}$$

If $\tilde{F}_{min} = aF_{min} + b < 0$, then a and b were recomputed such that $\tilde{F}_{min} = 0$ and $\tilde{F}_{avg} = F_{avg}$. Relations which give this behavior are

$$a = \frac{F_{avg}}{F_{avg} - F_{min}} \tag{4.5}$$

$$b = -aF_{min} \tag{4.6}$$

Scaling the fitness of the population in this manner helped to avoid having the population driven to a local maximum, rather than a global one.

The second specialization of the GA to the self-structuring antenna was the addition of a limited amount of elitism, or survival of the fittest, which insured that the best states were not lost due to crossover or mutation. This addition gave more favorable results, since the best states from several generations were able to produce offspring in subsequent generations. Another feature of the genetic algorithm that is unique to the SSA is the use of a single bit-string to make up a full chromosome. Generally, a genetic algorithm uses several bit-strings pieced together to make up a chromosome, allowing only changes in a few of the variables during crossover. Having only one variable to optimize, that being the configuration of the antenna skeleton, the GA for the SSA is inherently simpler, providing for a faster convergence. The modifications made to the standard GA allow for better performance in the intended application.

4.2 Television Measurements

Measurements were made on the 30-switch self-structuring antenna prototype using a genetic algorithm for control of switch states. The relays on the antenna template were controlled using a Multitrax microcontroller board (Control Technology, www.controltrax.com). This board, based on the Hitachi HD64180 microcontroller, has 48 lines of digital I/O, 8 channels of 10-bit A/D conversion, 64kB of RAM and 32kB of EPROM memory, and a ROM-based multitasking BASIC run-time compiler (MT-BASIC). The digital I/O lines could not supply enough current to drive the relays directly, so they were used in conjunction with 74AC541N driver ICs. The genetic algorithm was written using MT-Basic and has been included as Appendix A. The purpose of measurements made using the 30-switch prototype with genetic algorithm control described below were for proof of concept. This testing shows the self-structuring antenna's ability to adapt itself in order to find optimum reception in a given environment through the implementation of a genetic algorithm.

Section 4.2.1 describes the environment under which testing took place. Sections 4.2.2 and 4.2.3 give results of testing done for channels in the UHF and VHF bands. Specifically, channel 23 (524-530 MHz) was used for UHF testing, while channel 10 (192-198 MHz) was used for VHF testing.

4.2.1 Testing Environment

Testing performed on the 30-switch SSA prototype took place outside of the Engineering Research Complex on the campus of Michigan State University. The antenna was placed on top of the television used for testing. Figure 4.1 shows the test setup for the SSA placed horizontally atop the television, while Figure 4.2 displays a vertical orientation. In both cases, the television was placed on a wooden table along with power supplies, a computer and monitor, a control board, and the microcontroller. The computer was used to collect data as tests were run, and to provide an interface to the microcontroller. The use of latching relays made the control board containing drivers and logic hardware necessary, since the relays needed to be released after each combination and required large amounts of current for switching. For more information on the control board, see Section 3.1.2.

A baseline for what constitutes a good received signal was provided through the use of a standard television dipole antenna, or "rabbit ears", using the voltage present at the automatic gain control of the television. This voltage was then used to set the ideal voltage in the objective function within the genetic algorithm as mentioned in 4.1. Using this function we were able to quantify how clear the television picture would be when placed in a given configuration and consequently choose the best parent configurations for the next generation.

The testing performed here included looking at two local channels, one in the Ultra High Frequency (UHF) band and one in the Very High Frequency band (VHF). For

UHF testing channel 23 (524-530 MHz) was used and for VHF, channel 10 (192-198 MHz) was used. In both cases, tests were run using both a vertical and horizontal orientation for the SSA, as mentioned earlier. The results were then compared to improvements seen with a typical genetic algorithm. For a typical GA the average value of the fitness increases in a fairly smooth fashion, then saturates at a steady value. Evolution in maximum fitness for a typical GA often proceeds in "punctuated equilibria", with sudden improvements interspersed between periods in which fitness remains relatively unchanged [10]. To get an idea of what a good signal is, consider Figure 4.3, giving a fitness of 0.5. This figure shows what is considered to be a bad picture. The television has 'snow' on the screen and is generally an unpleasant viewing experience. An improvement is seen in Figure 4.4, where the fitness is 0.8 and the 'snow' has decreased. The best pictures are seen between 0.9 and 1.0, shown in Figure 4.5 and Figure 4.6. These pictures are crisp and free of 'snow', which is what one wants to see when watching television. Further detail on this testing is given in the following sections.

4.2.2 UHF Test Results

Testing was performed to evaluate the effectiveness of the SSA and its genetic algorithm control for the frequency band known as Ultra High Frequency, or UHF. Experiments were performed for both a horizontal and vertical orientations of the SSA.

Results for the horizontal orientation are given in Figure 4.7. Comparing these results to the typical case described above, the genetic algorithm, as well as the SSA perform as desired. Specifically, there is a fairly smooth increase in the average fitness of the population to a fitness of about 0.7, while the maximum fitness increases quickly to a steady value of about 0.9 and up. Note that there are variations in the maximum fitness with generation, suggesting that the best configuration is changing from generation to generation. This is a result of the fitness scaling explained above.

The results for UHF testing using a horizontal orientation were the most promising, telling us several things. First, the orientation of the antenna skeleton is important. This was to be expected, since most U.S. television stations broadcast using a horizontal polarization, with the exception of some VHF stations who broadcast with a circular polarization. Second, the SSA is of a good dimension to receive signals at around 500 MHz. At this frequency, the dimensions of the SSA used in testing are about 0.8λ by 0.53λ, where λ is the wavelength of the received signal.

Vertical orientation test results are given in Figure 4.8. These results reinforce the conclusion given above, that the orientation of the SSA is important. Looking at Figure 4.8, one can see that the performance of the SSA is generally random in average, maximum, and minimum fitness. The performance is also of a generally worse quality than in the horizontal case, with average values around 0.5-0.6 and maximum values of 0.7-0.9. The ineffectiveness of the GA here is due to the lack of good combinations with the SSA oriented vertically.

4.2.3 VHF Test Results

Testing for the VHF band show that the SSA template used for these experiments was too small to effectively receive the signals in this frequency range. The size of the antenna at around 195 MHz is about 0.3λ by 0.2λ. The fitness levels seen in VHF testing were generally lower, giving steady state values averaging about 0.5 for horizontal orientation, with around 0.75 fitness levels for maximum values, as seen in Figure 4.9. Figure 4.10 shows that vertical orientation again gives decreased performance, with average fitness levels around 0.4 and maximum fitness levels around 0.6. The lower fitness levels seen in the vertical case, as well as the slightly more random features, lead one to believe that the levels seen in the VHF case are indeed due to the size of the antenna and not necessarily due to a circular polarization of the received signal.

4.3 Conclusions

The use of a genetic algorithm and the testing done here provides a proof of concept for the self-structuring antenna. It shows us that the antenna is capable of delivering good performance under certain conditions. Specifically, it shows that the current template design is effective for UHF television reception under a horizontal orientation. This testing gives some insight into the necessary orientation and size of the antenna. As seen in the VHF testing, the antenna is currently undersized for the frequency band in which it is expected to perform. These are all considerations that need to be taken into account for future antenna templates in order to create an antenna that is both diverse and effective.

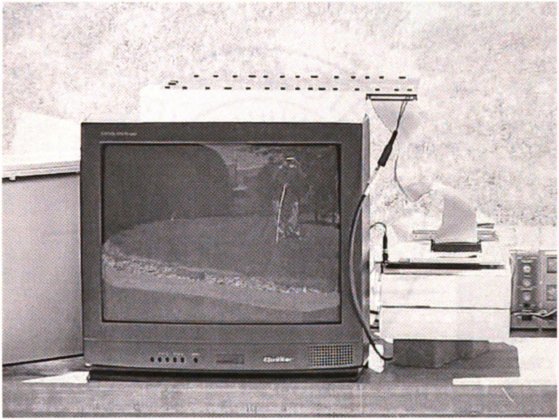

Figure 4.1. Test Setup with horizontal placement of the SSA.

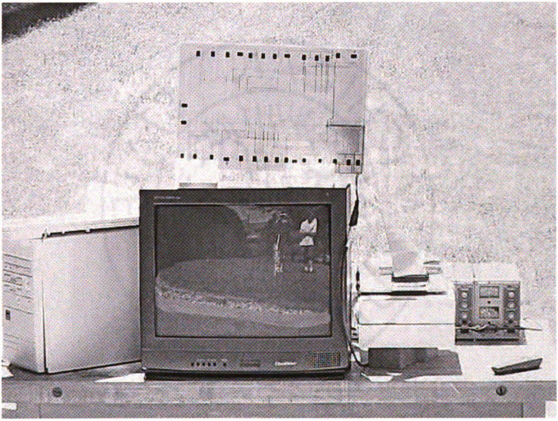

Figure 4.2. Test Setup with vertical placement of the SSA.

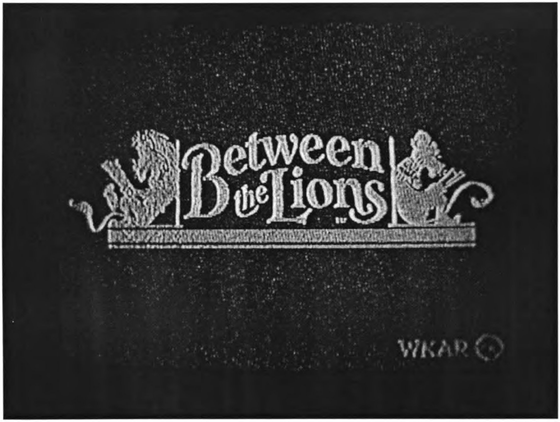

Figure 4.3. Television Reception at a Fitness of 0.5.

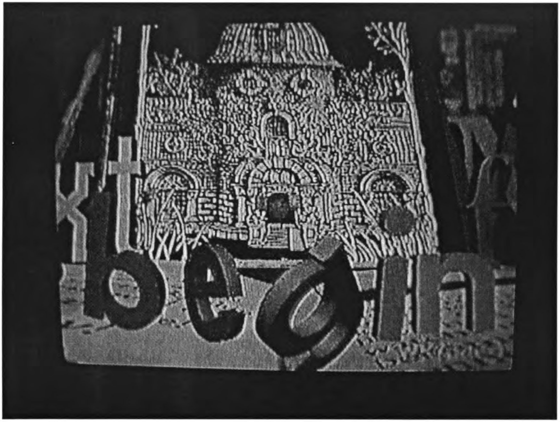

Figure 4.4. Television Reception at a Fitness of 0.8.

Figure 4.5. Television Reception at a Fitness of 0.9.

Figure 4.6. Television Reception at a Fitness of 1.0.

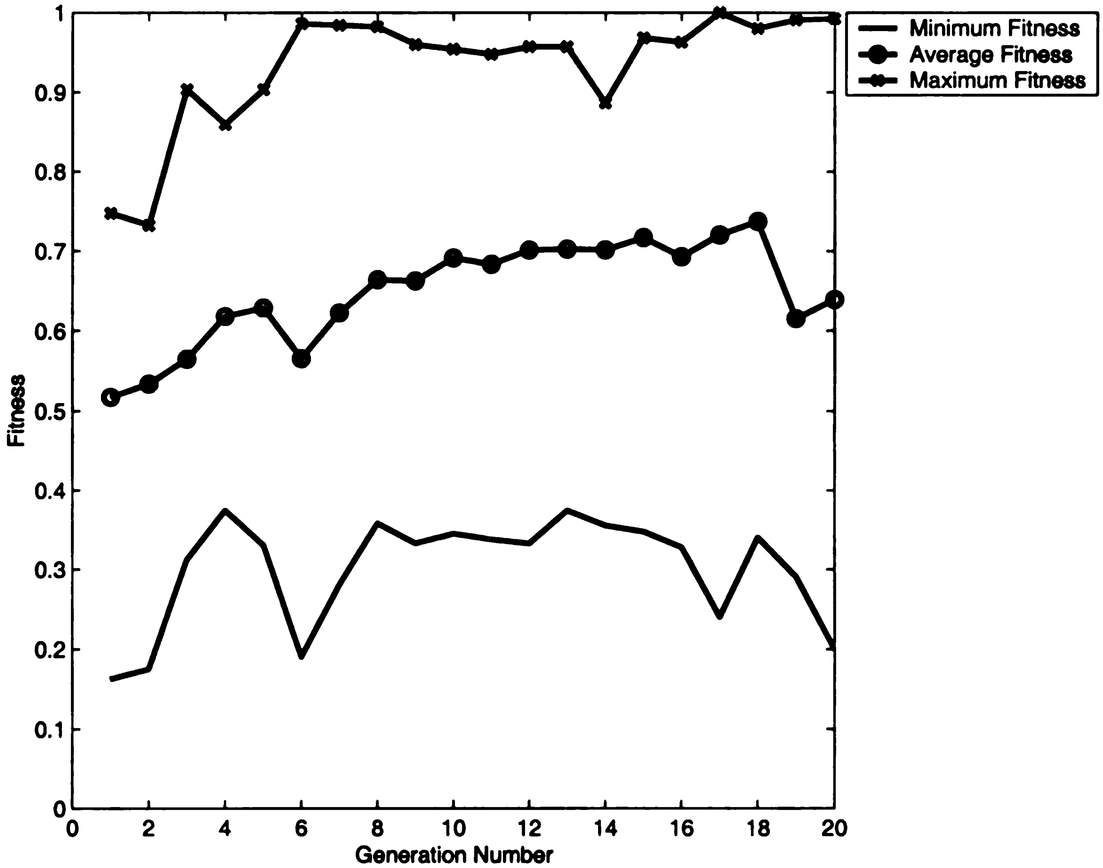

Figure 4.7. UHF Test Results with Horizontal Placement of the SSA.

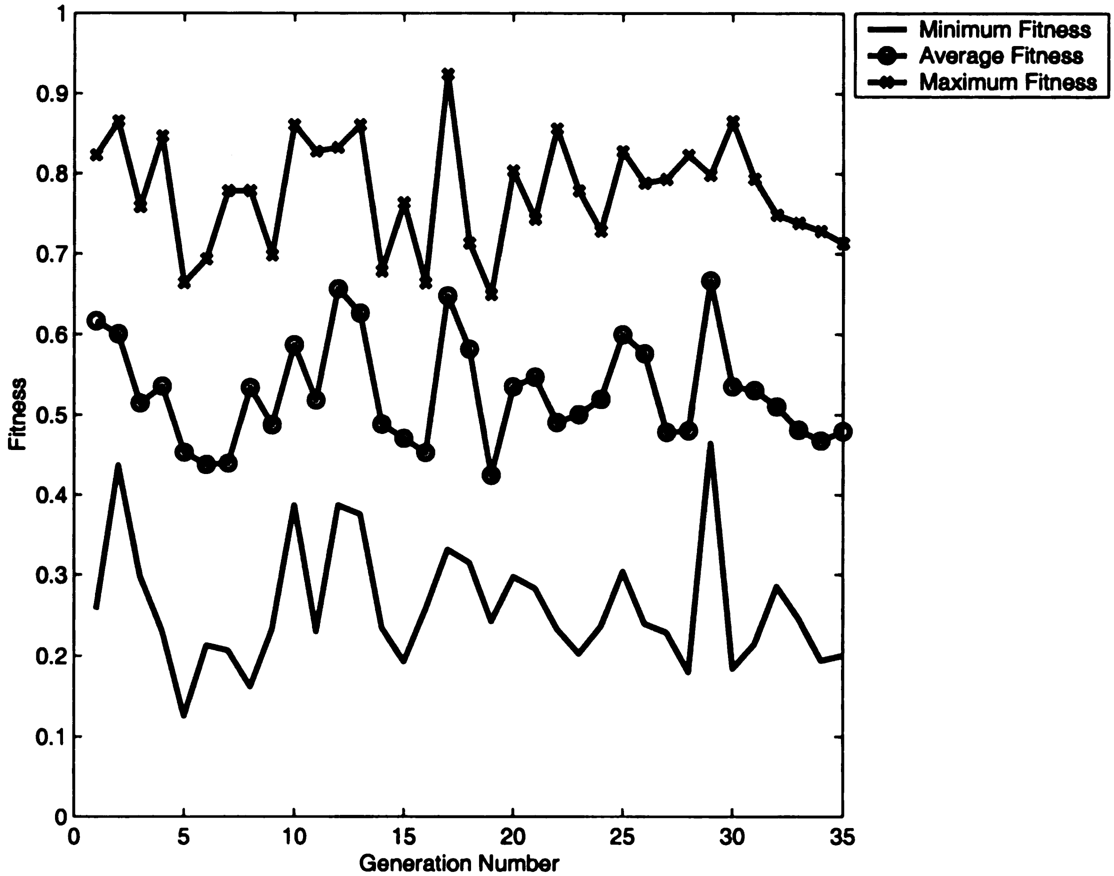

Figure 4.8. UHF Test Results with Vertical Placement of the SSA.

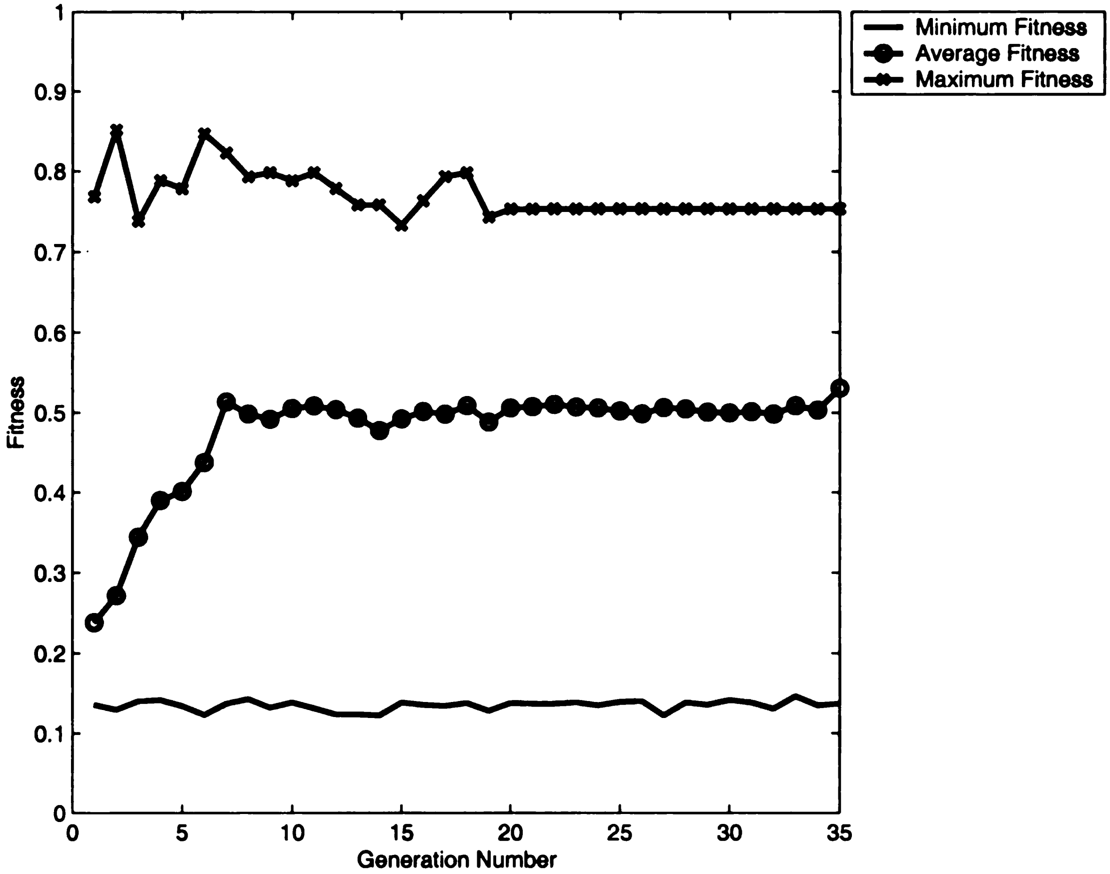

Figure 4.9. VHF Test Results with Horizontal Placement of the SSA.

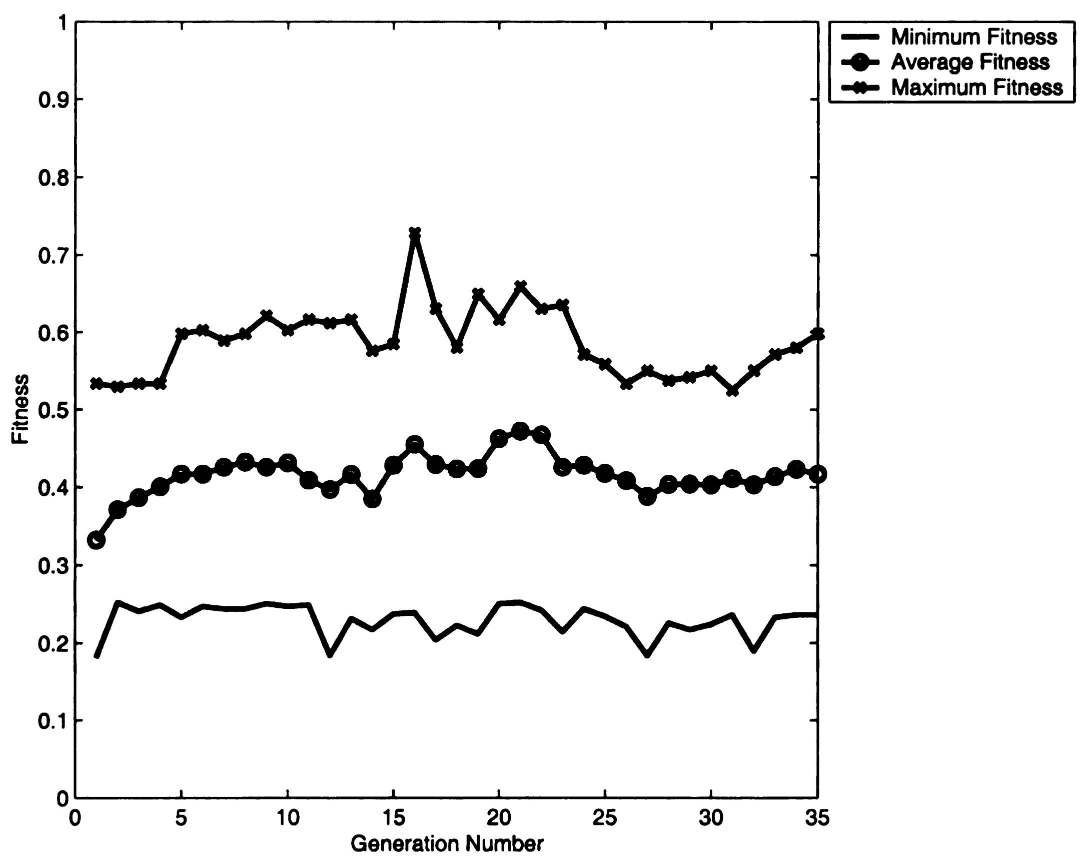

Figure 4.10. VHF Test Results with Vertical Placement of the SSA.

CHAPTER 5

SWITCH FAILURE ANALYSIS

Many times antennas are subject to harsh environmental conditions, where their physical degradation, alteration, or misuse results in decreased electrical performance. Self-Structuring Antennas should be designed to operate under these difficult conditions and respond to changes both in their environment and in their own physical structure without loss of effectiveness. Since these antennas are to be used in many consumer applications where replacement and repair may be difficult, it is important to consider the effects of switch failure on the performance of the antenna.

The successful operation of the SSA is dependant on the large number of available states in which the antenna performs acceptably. It is important that the antenna template is designed in such a way that the failure of a switch, or a group of switches, causes only minimal degradation in the performance of the antenna. That is, the performance of the antenna does not fall below an acceptable level under switch failure. For this reason, a study of the effect of switch failure on the SSA was performed.

Section 5.1 gives an overview of test procedures and the environment in which the switch failure analysis was conducted. Results of single switch failures and their consequences are covered in Section 5.2. Multiple switch failures are the topic of Section 5.3.

5.1 Testing Environment and Procedure

Experiments were performed to measure the standing wave ratio (SWR) of the Self-Structuring Antenna using the HP-8510 Network Analyzer. Figure 5.1 shows the environment under which switch failure was analyzed. It can be seen in Figure 5.1 that the antenna was placed horizontally atop a styrofoam pedestal. The antenna switches were controlled using a FlashTrax microcontroller board which was placed on a neighboring pedestal. This board, based on the Hitachi HD64180 microcontroller, has 24 lines of digital I/O, 8 channels of 10-bit A/D conversion, 64k of flash RAM memory, and a ROM-based optimized BASIC interpreter. The antenna feed was connected to the network analyzer through a standard 4:1 wideband television balun. Measured standing wave ratios for each switch combination were logged on a nearby computer for future analysis. The BASIC code used to control the network analyzer is included in Appendix B. The BASIC code used by the microcontroller to set the antenna states is given in Appendix C.

The procedure for testing the SSA involved sending 10000 random states to the antenna and measuring the resulting standing wave ratio. The switch combinations sent to the antenna were statistically random and were consistent throughout the experiments. That is, the states were sent to the SSA from a file of 10000 random numbers that remained the same for all tests. This procedure was carried out for operating frequencies in the range of 50 to 450 MHz, taken in 50 MHz steps.

The results of this testing were analyzed using a C++ program called switch.cpp. This program was used to compare the SWR measurements for failures of individual switches, as well as groups of switches. A baseline was established for comparison using all 10000 switch combinations when no switches failed. The resultant figures are explained in Sections 5.2 and 5.3.

5.2 Single Switch Failures

In order to get a fundamental understanding of the consequences of a switch failure, the malfunctioning of a single switch is considered. A switch failure occurs when any switch on the antenna template is permanently placed in either an "on" or "off" position. To quantify the effect of a failed switch, the following function was defined:

$$frac(f) = \frac{N_{GS}}{N_{AS}} \tag{5.1}$$

where N_{GS} is the number of good states defined as those switch combinations which give SWR ≤ 2.0, and N_{AS} is the number of available states. Available states are defined as switch combinations that the antenna is still able to be placed into. With this function, switches which give minimal, intermediate, or large degradation in the performance of the antenna are placed into groups. It should be noted that the number of available states is roughly cut in half with each switch failure. Therefore, a switch failure reduces the total number of good states and causes a degradation in performance, although its effect may be minimal. Figure 5.2 shows the location of switches on the antenna template.

5.2.1 Minimal Degradation under Single Switch Failures

Switches that exhibit minimal degradation in SSA performance under failure are those that only cause the efficiency of the control algorithm to deteriorate. The algorithm's ability to find good states depends on the large number of available good states. That is, the more total good states available to the algorithm, the faster it can optimize the population. This means that when the total number of states is cut in half the algorithm becomes less efficient, even if the number of good states is also cut in half. Switch failures showing minimal degradation represent the case of the number of good states being cut in half when the total number of states is also cut in half. For this case, the frac(f) function remains unchanged from the fully functional case over the entire frequency band (50 to 450 MHz). Switches that exhibit this behavior are generally located away from the antenna feed, controlling larger sections of the antenna template. These switches are numbers 1 (Figure 5.4), 3 (Figure 5.6), 5 (Figure 5.8), 7 (Figure 5.10), 10 (Figure 5.13), 11 (Figure 5.14), 15 (Figure 5.18), 19 (Figure 5.22), 21 (Figure 5.24), and 23 (Figure 5.26) in Figure 5.2. The figures listed above show cases in which the frac(f) function follows the baseline case, regardless of whether a switch fails in the "on" or "off" position. Figures showing this behavior represent switch failures causing minimal degradation.

5.2.2 Intermediate Degradation under Single Switch Failure

Switches exhibiting intermediate levels of performance degradation involve a deterioration in the efficiency of the control algorithm, as well as small changes in the frac(f) function for discrete frequencies. Switch 8 (Figure 5.11) is an example of a switch failure which causes intermediate levels of degradation. Figure 5.11 shows a noticeable change in the frac(f) function at both 200 and 400 MHz. This behavior also occurs in switches 9 (Figure 5.12), 12 (Figure 5.15), and 17(Figure 5.20) at differing frequencies in each case. It is interesting to note that these switches lie in the area of the board bounding minimal and large degradation areas, as shown in Figure 5.3. These switches show better behavior than those which control shorter elements, but worse behavior than those controlling longer ones. This suggests a relationship between the electrical length of the antenna elements and the performance of the SSA under switch failure. No conclusions have been drawn as to why certain switches in this group show changes in performance at differing frequencies.

5.2.3 Large Degradation under Single Switch Failure

Switches which show large levels of change in performance lower the efficiency of the algorithm, as well as changing the frac(f) function large amounts at various frequencies. Figure 5.5 shows a very pronounced case of this type of failure. This figure represents a failure of switch 2 of Figure 5.2. It can be seen in this figure that the performance of the antenna is changed dramatically when switch 2 fails. At low frequencies there is a vast degradation in performance if the switch fails in the "on" position, while performance seems to improve when the switch fails "off". The opposite is true at higher frequencies. It is the unpredictability of the state in which a switch may fail, as well as the wideband application of the SSA, that causes this type of switch failure to be unacceptable. Other switches which cause large performance changes are 4 (Figure 5.7), 6 (Figure 5.9), 13 (Figure 5.16), 14 (Figure 5.17), 16 (Figure 5.19), 18 (Figure 5.21), 20 (Figure 5.23), 22 (Figure 5.25), and 24 (Figure 5.27). Note that all of these switches lie close to the antenna feed and control the configuration of short antenna elements, again suggesting a link between electrical length and performance.

It should be emphasized that failures showing large performance degradation are unacceptable to the design of the SSA template, since they lower the total number of good states in an unpredictable fashion. These failures need to be accounted for through redundancy in the layout of the SSA template in order to avoid complete failure of the antenna in its end application.

5.3 Multiple Switch Failures

In order to obtain a better understanding of the effects of switch failure, the possibility of multiple failures was explored. Multiple failures are ones in which more than one switch gets permanently placed in either the "on" or "off" position. This means that there are four possible ways in which a given pair of switches can fail. This can be seen in Figure 5.28, with both switches failing on, both failing off, switch 7 failing on and 11 off, and finally switch 7 off and 11 on. Also included in this and the rest of the Figures in this section is the control case of no switch failures. With four ways in which a pair of switches can fail, and 24 total switches, 1044 possible failures exist. These failures were grouped into the following categories, shown in Table 5.1: minimal degradation , intermediate degradation, large degradation due to one switch, and large degradation due to both switches. Figure 5.28 shows an example of switches exhibiting minimal degradation in antenna performance. Figure 5.29 is an example of intermediate levels of degradation, while Figure 5.30 and Figure 5.31 are examples of large degradation due to one and both switches, respectively. These examples are representative of each type of performance degradation seen during multiple switch failure analysis.

	First Switch Failed																						
Second Switch Failed	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
2	3																						
3	1	3																					
4	3	4	3																				
5	1	3	1	3																			
6	2	4	3	4	2																		
7	1	3	1	3	1	3																	
8	2	4	2	4	2	4	2																
9	2	4	2	4	2	4	2	2															
10	1	3	1	3	1	3	1	2	2														
11	1	3	1	3	1	3	1	2	2	1													
12	2	4	2	4	2	4	2	3/4	4	4	2												
13	3	4	3	4	3	4	2	3	3	3	3	4											
14	3	4	3	4	3	4	3	4	3	3	3	4	4										
15	1	3/4	1	3	1	3	1	2	2	2	1	2	3/4	4									
16	3	4	3	4	3	4	3	4	4	3	3	4	4	4	3								
17	2	4	2	4	2	4	2	3	2	4	2	4	4	4	2	4							
18	3	4	3	4	3	4	3	3	4	4	3	4	4	4	3	4	4						
19	1	3/4	1	3	1	3	1	2	2	1	1	2	3	3	1	3	2	3					
20	3	4	3	4	3	4	3	3	4	4	3	4	4	4	3	4	4	4	3				
21	1	3	1	3	1	3	1	2	1	1	1	2	3	3	1	3	2	3	1	3			
22	3	4	3/4	4	3	4	3	4	4	4	3	4	4	4	3	4	4	4	3	4	3		
23	1	3	1	3	1	3	1	2	2	1	1	2	3	3	1	3	2	3	1	3	1	3	
24	3	4	3	4	3	4	3	4	3	3	3	4	4	4	3	4	4	4	3	4	3	4	3

Under Single Failure
= Minimal Change
= Intermediate Change
= Large Change

1 = Minimal change in fraction of good states
2 = Intermediate change in fraction of good states
3 = Large change due to single switch failure only
4 = Large change due to multiple switch failures

Table 5.1. Multiple Failures

5.3.1 Minimal Degradation under Multiple Switch Failures

Minimal degradation is regarded as having no appreciable change in the frac(f) function over the entire frequency band (50-450 MHz), as defined in Section 5.2. With multiple switches failing, combinations which showed this property were primarily those in which both switches showed minimal degradation under single switch failure. This held true for most combinations of switches behaving in this fashion, although some combinations did result in intermediate amounts of degradation when one or more of the switches were near the minimal-intermediate boundary shown in Figure 5.3.

5.3.2 Intermediate Degradation under Multiple Switch Failures

Intermediate degradation involves similar behavior to the above case, with some slight deviations at discrete frequencies. Figure 5.29 shows this behavior for switch combinations involving switches 7 and 12. Note that individually, one of these switches was in the minimal degradation group, while the other was in the intermediate group. This tends to be the case for multiple switch failures resulting in intermediate levels of degradation, although in some cases a large change was noted when switches were near the boundaries of minimal-intermediate, or intermediate-large change under single switch failures, as shown in Figure 5.3.

5.3.3 Large Degradation under Multiple Switch Failures

Switch combinations that involved large degradation in performance of the SSA were split into two groups in order to get a better understanding of the interaction between switches. Combinations that exhibited large change independent of one of the failing switches made up one group, while those that varied greatly due to both switches made up a second group. Figure 5.30 shows a switch combination that belongs to the first group. We can see that a large degradation occurs, but it is independent of the position of one of the switches, specifically switch 22. In contrast, Figure 5.31 shows a combination that belongs to the second group. In this case, the degradation that occurs is dependent on both of the switches failing. The last group represents the worst possible case in terms of switch failures. The design of the SSA template needs to be reconsidered in order to avoid failures of this type, since they are unpredictable in their consequences.

Generally, switch combinations that produced large amounts of degradation independent of one switch involved both a switch showing minimal degradation, and one showing large degradation under single failures. This is what one might expect to happen. Multiple failures that caused the largest amounts of performance degradation due to both switches were also predictable. These were switches which caused large amounts of degradation under single failures. Figure 5.31 is an example of this second case, involving switches 22(Figure 5.25) and 24(Figure 5.27).

5.4 Conclusions

The current SSA template has various shortcomings that can be readily seen through this switch failure analysis. First, at low frequencies, the antenna does not perform well. This is true when no switches fail, due to the relatively small size of the template in comparison to a wavelength at low frequency. In addition, switches controlling short antenna elements cause a considerable loss in performance of the antenna under both single and multiple failure, especially at higher frequencies. This results from a lack of redundancy for the shorter antenna elements, i.e., longer elements can be built from shorter ones, but the inverse is not true. Realization of these shortcomings can be used in future generations of template layouts in order to create a more robust design that will withstand failure of one or more switches.

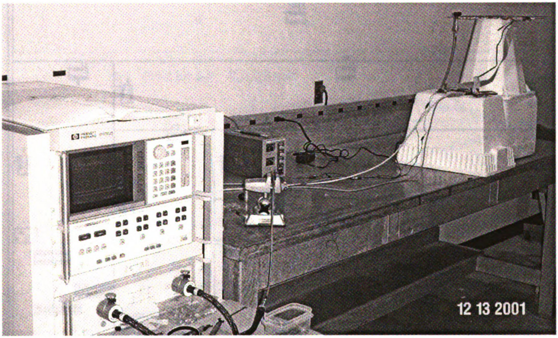

Figure 5.1. Laboratory Test Setup.

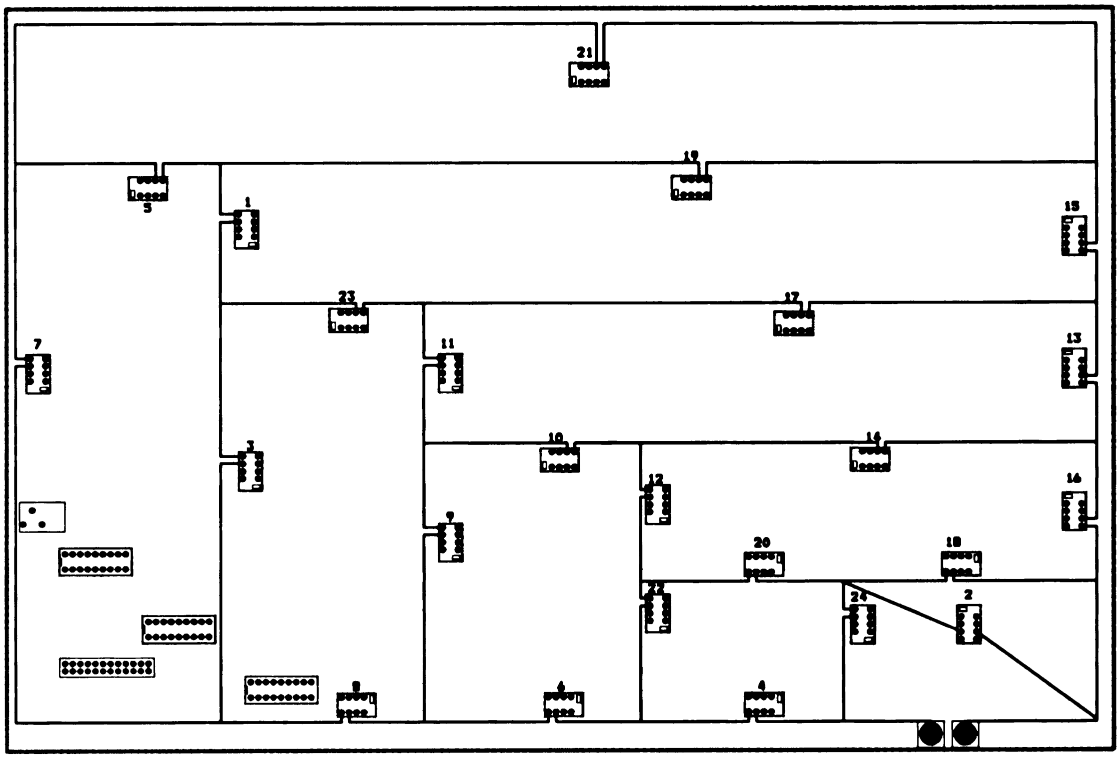

Figure 5.2. Self Structuring Antenna Template.

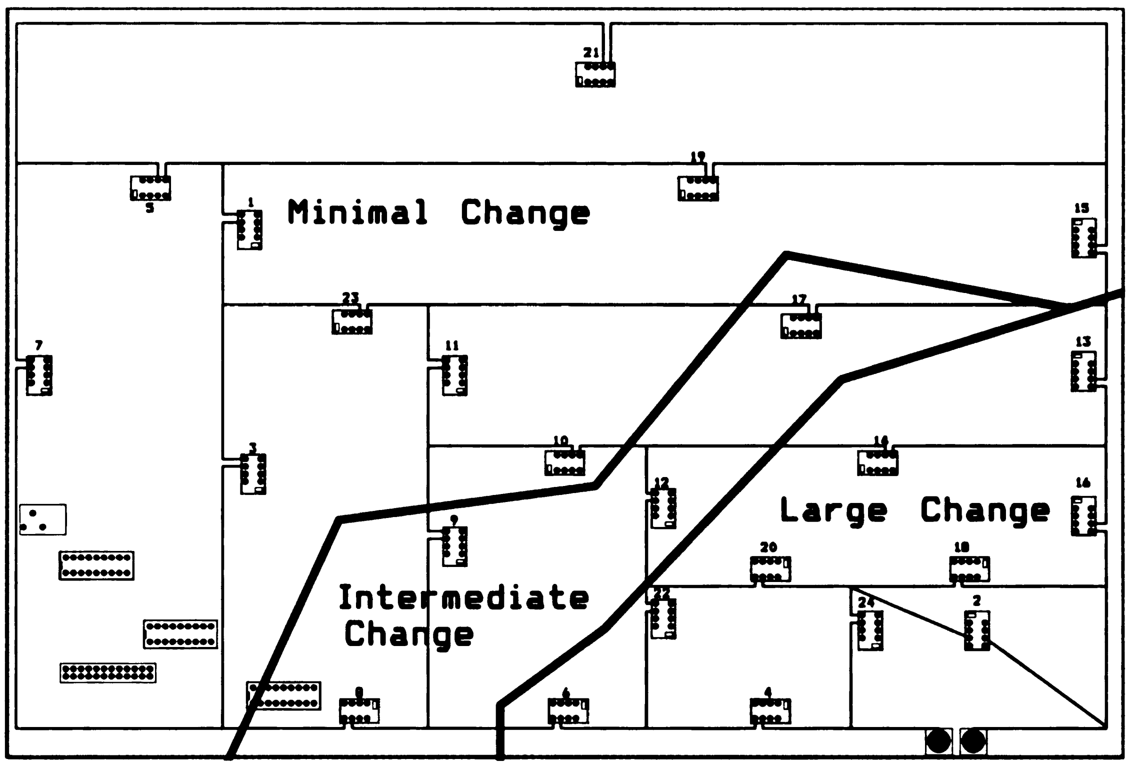

Figure 5.3. Regions of the SSA Exhibiting Different Levels of Degradation under Single Switch Failures.

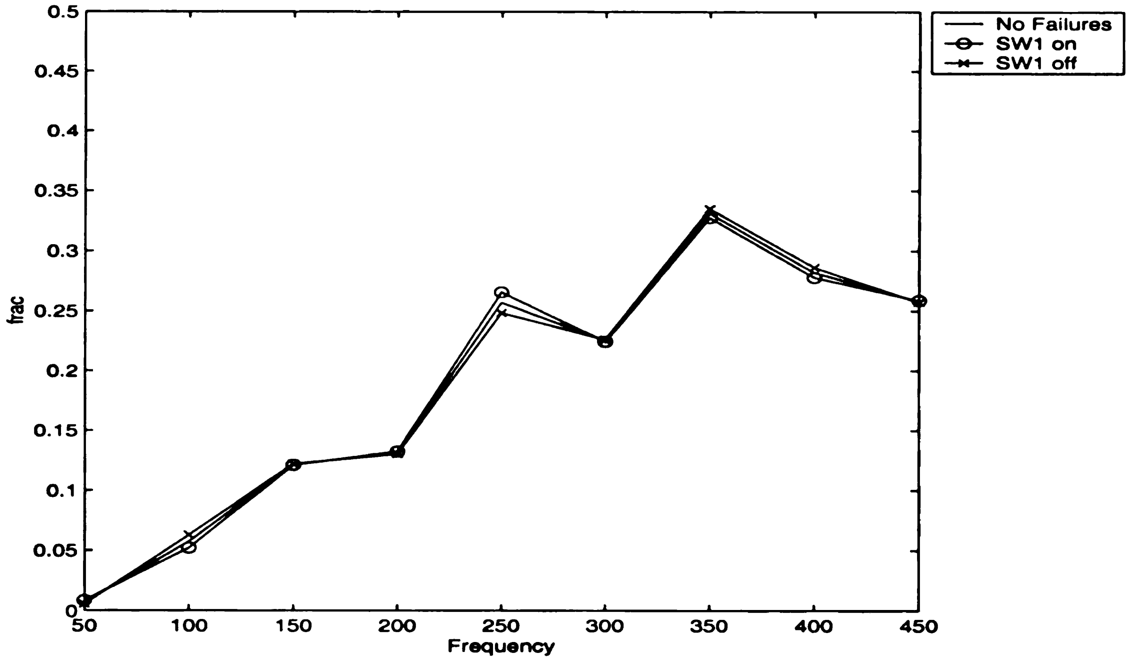

Figure 5.4. Single Failure of Switch 1.

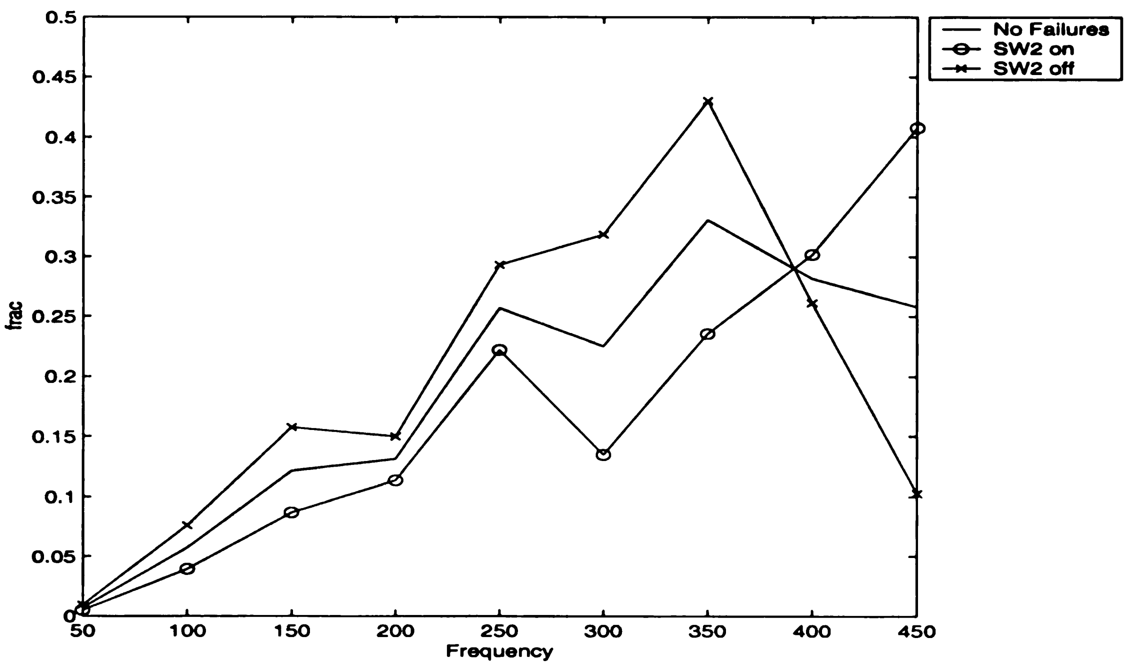

Figure 5.5. Single Failure of Switch 2.

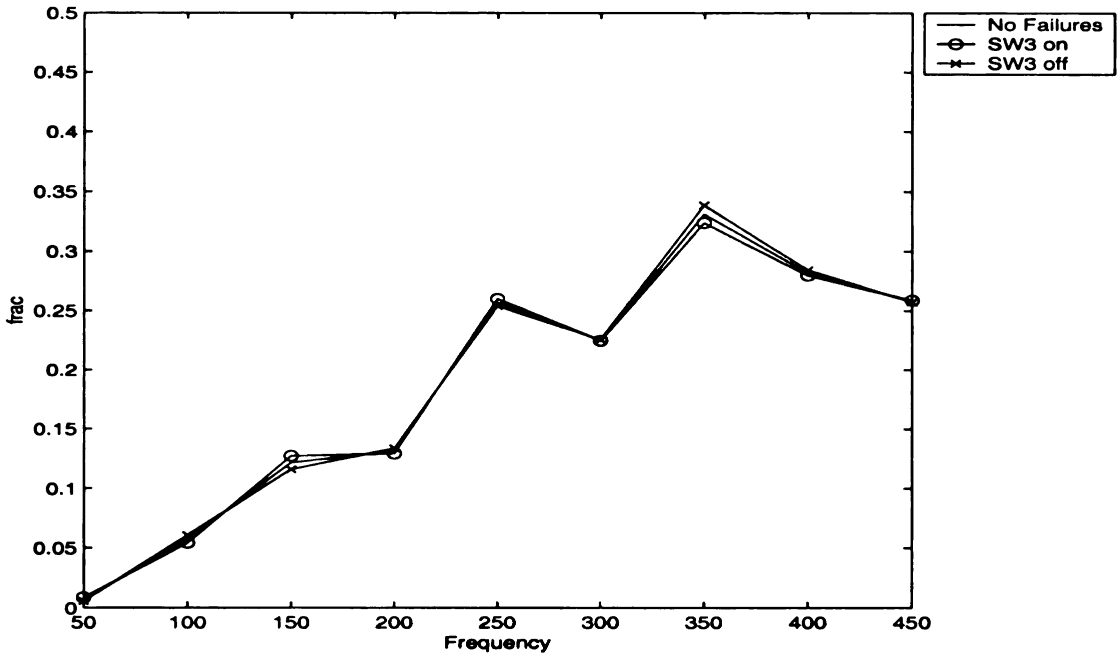

Figure 5.6. Single Failure of Switch 3.

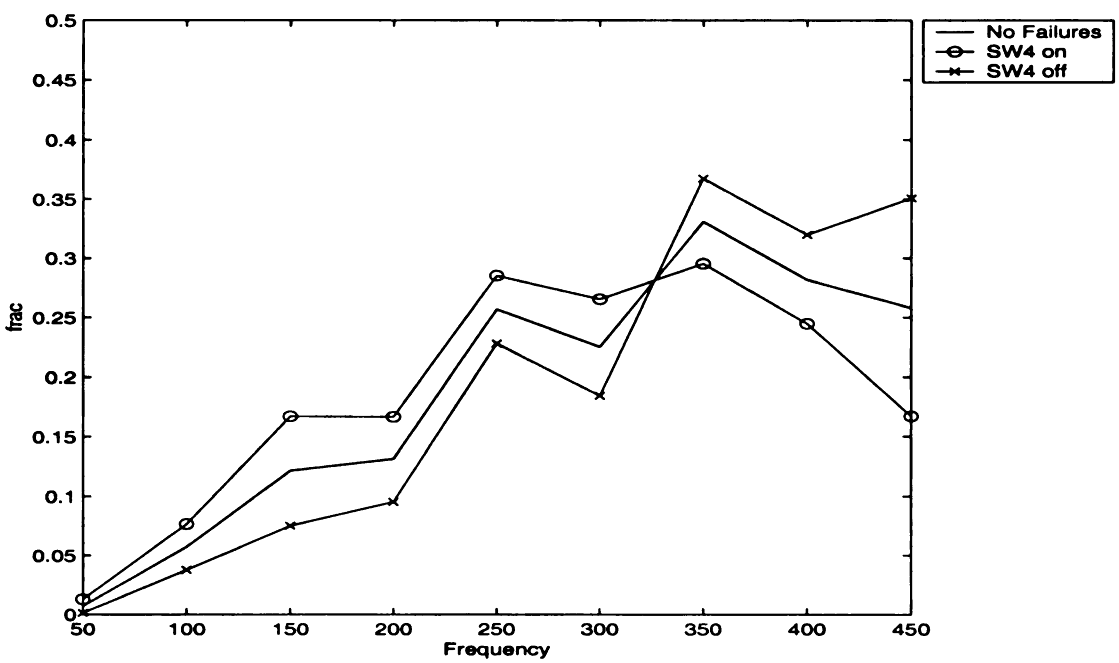

Figure 5.7. Single Failure of Switch 4.

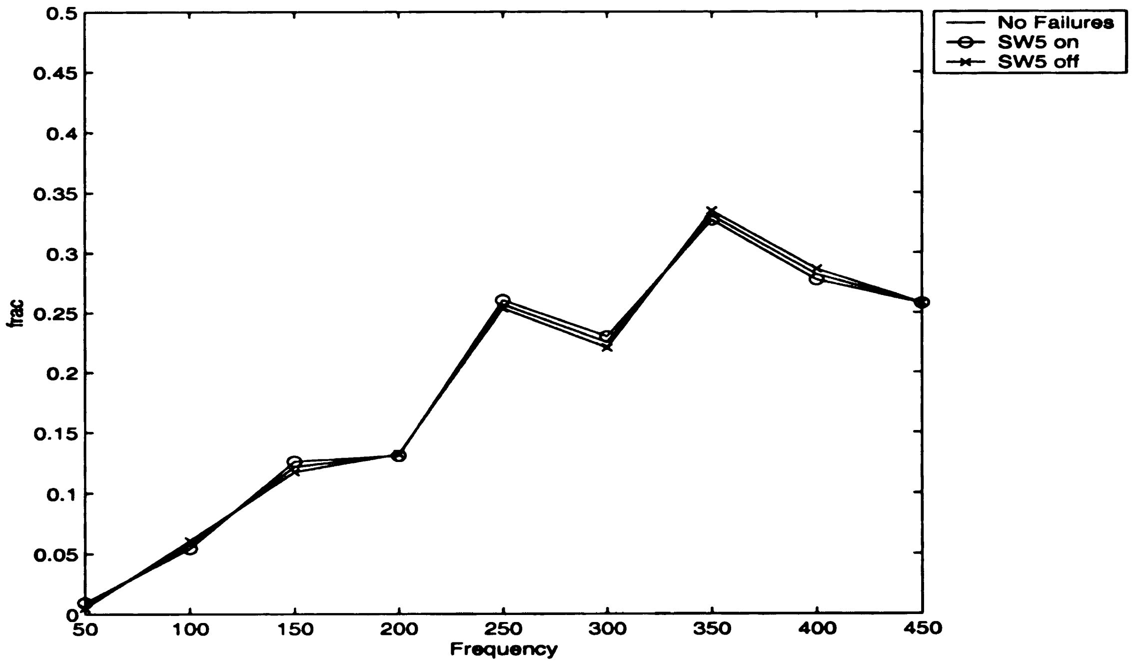

Figure 5.8. Single Failure of Switch 5.

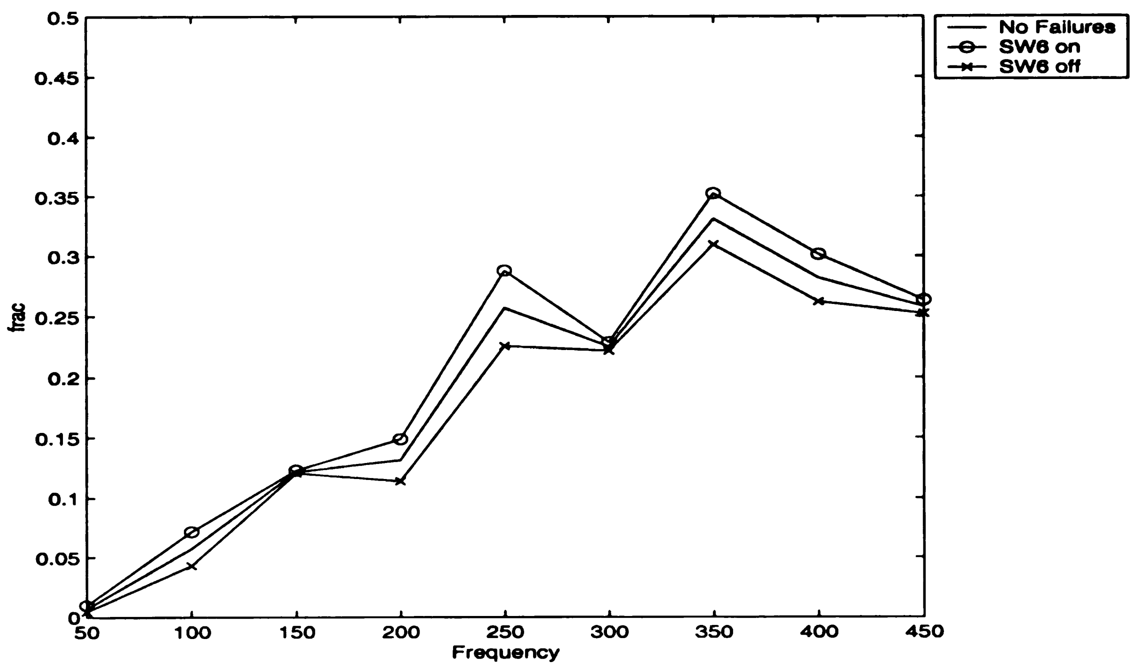

Figure 5.9. Single Failure of Switch 6.

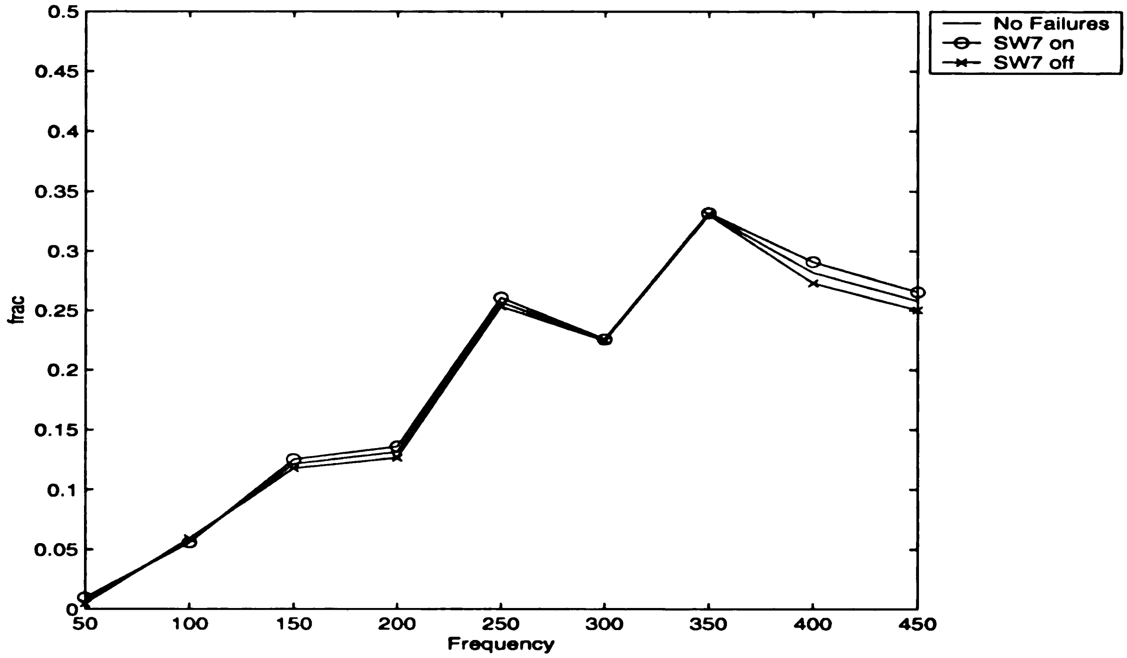

Figure 5.10. Single Failure of Switch 7.

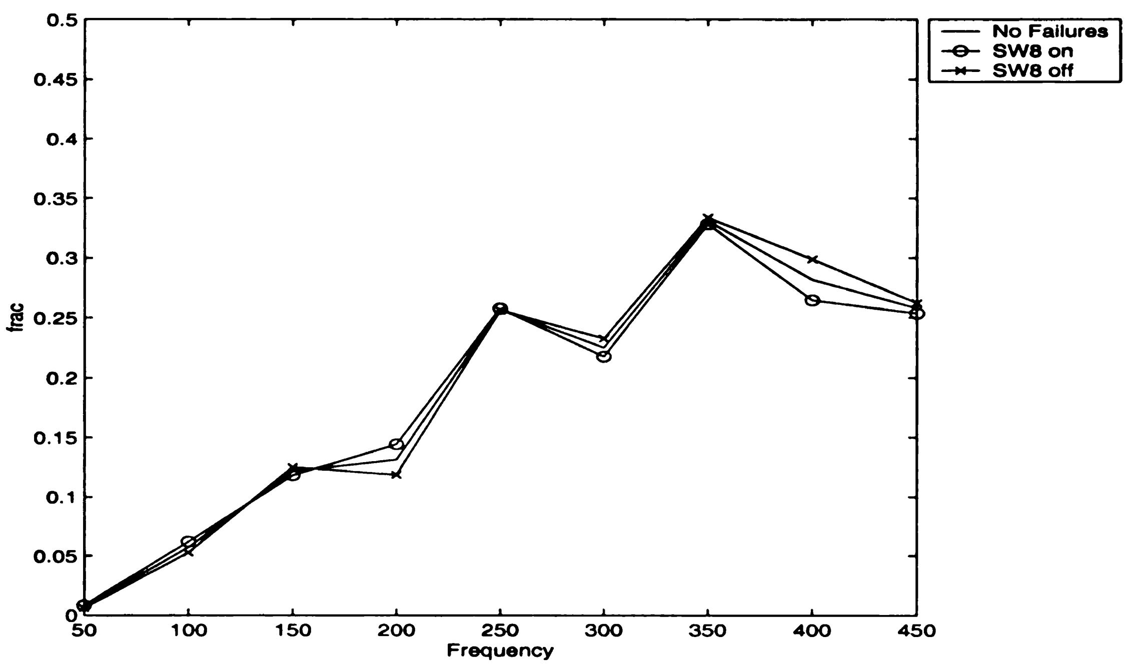

Figure 5.11. Single Failure of Switch 8.

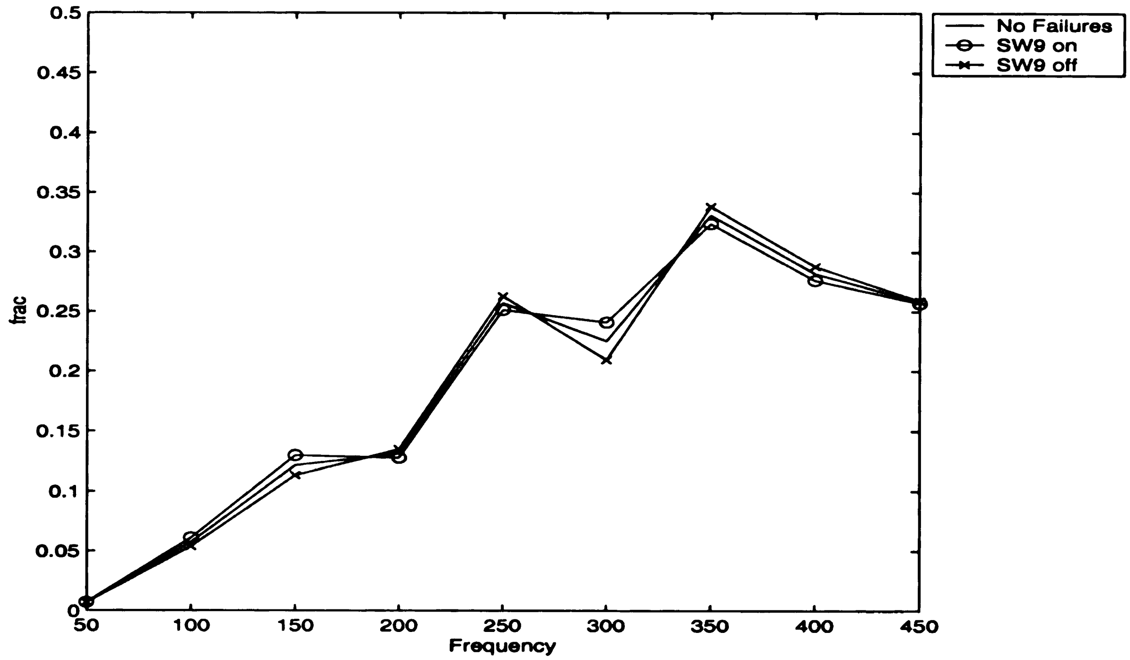

Figure 5.12. Single Failure of Switch 9.

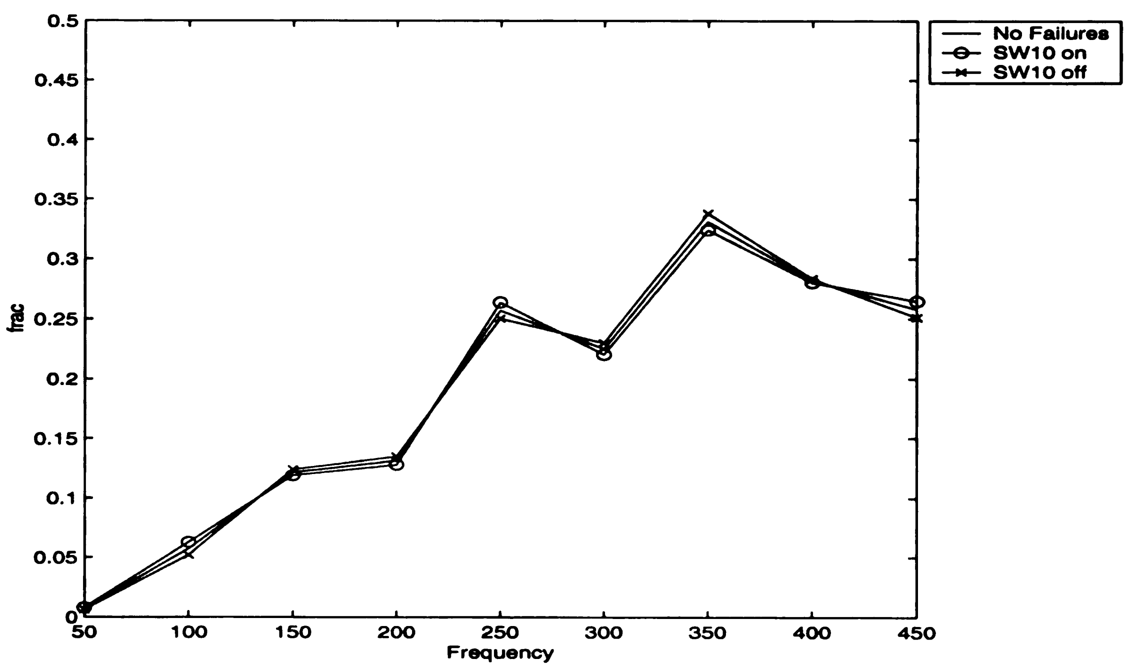

Figure 5.13. Single Failure of Switch 10.

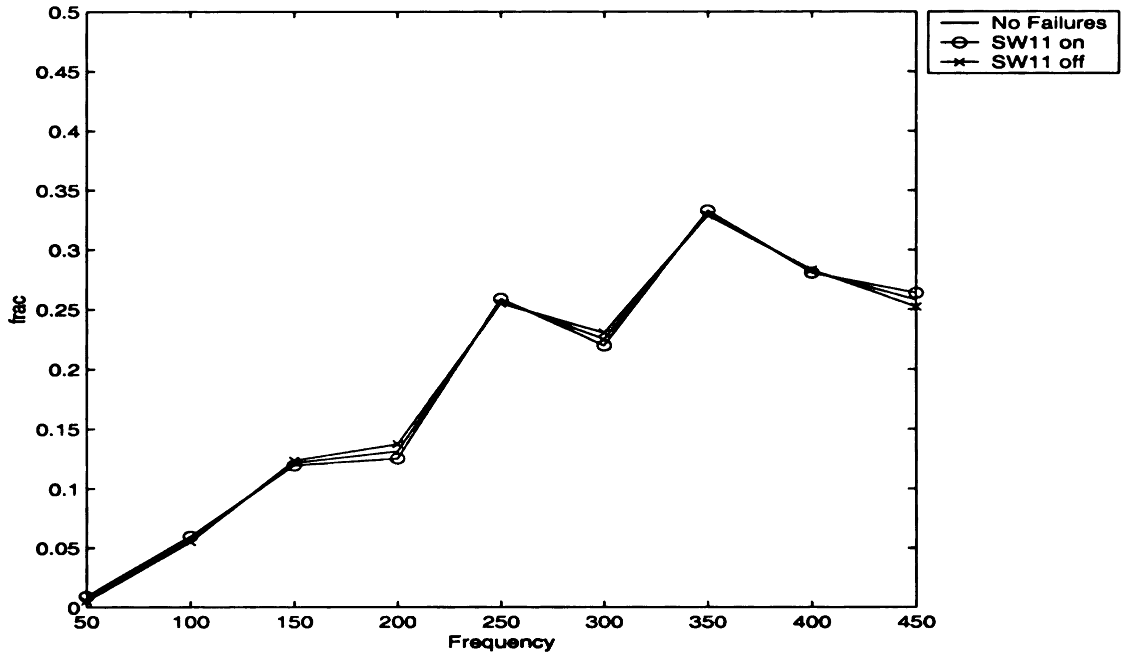

Figure 5.14. Single Failure of Switch 11.

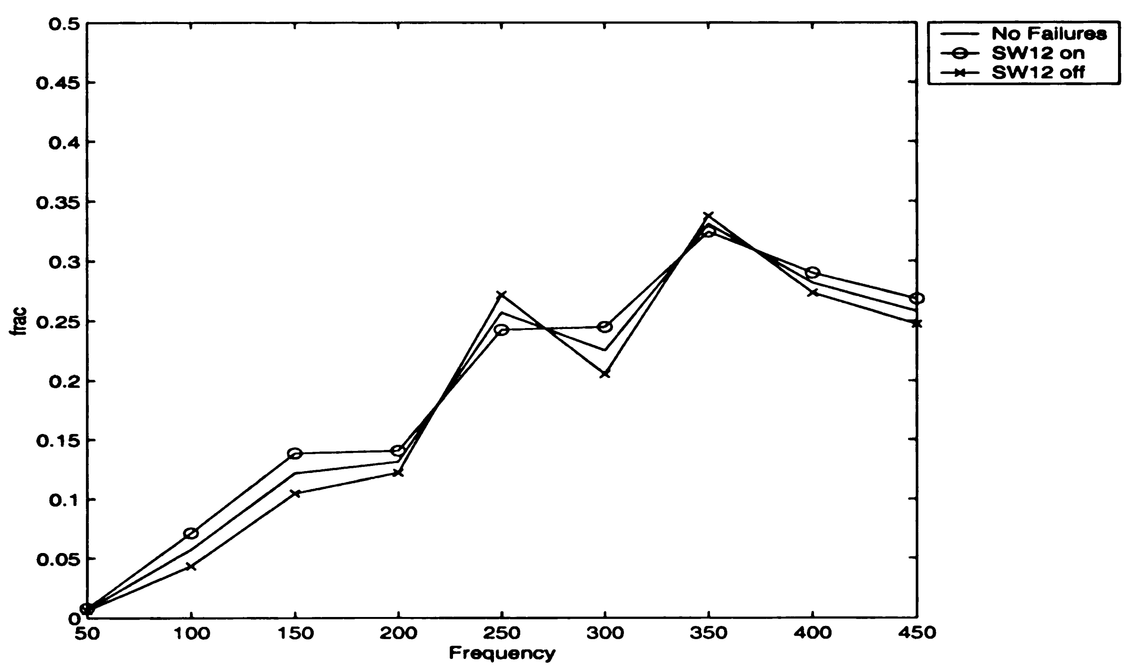

Figure 5.15. Single Failure of Switch 12.

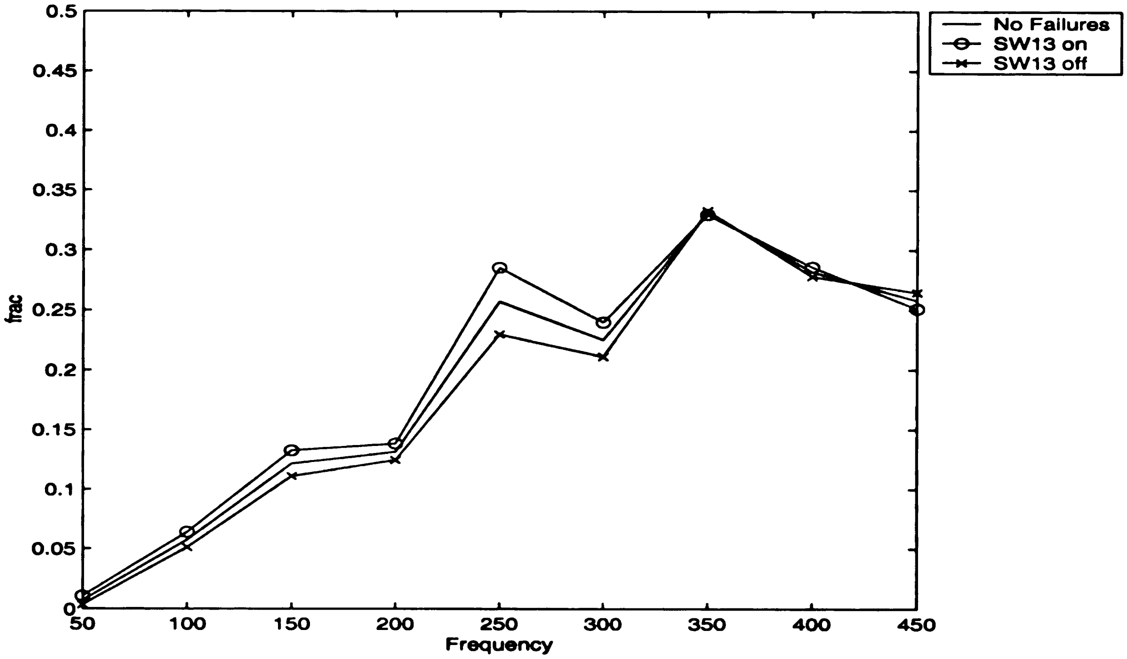

Figure 5.16. Single Failure of Switch 13.

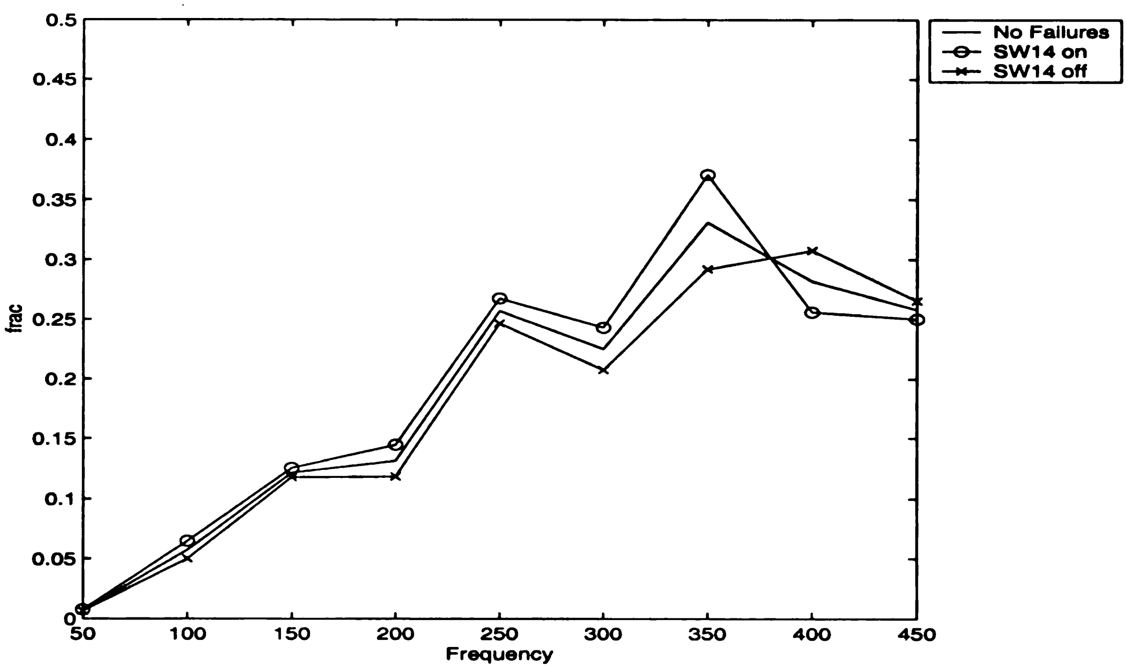

Figure 5.17. Single Failure of Switch 14.

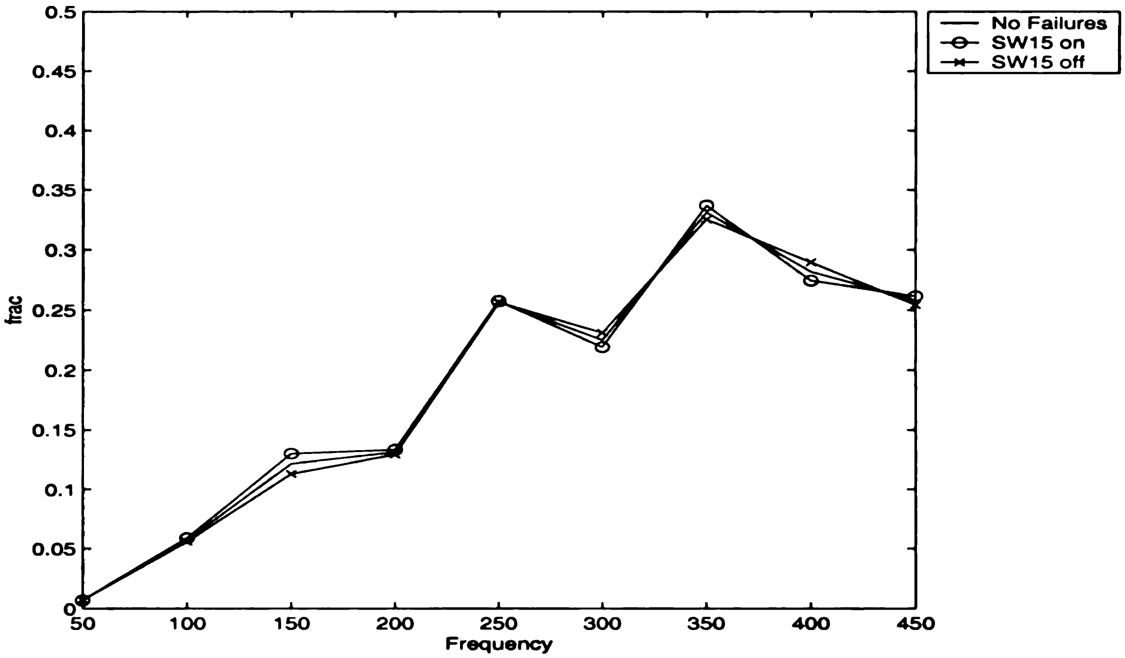

Figure 5.18. Single Failure of Switch 15.

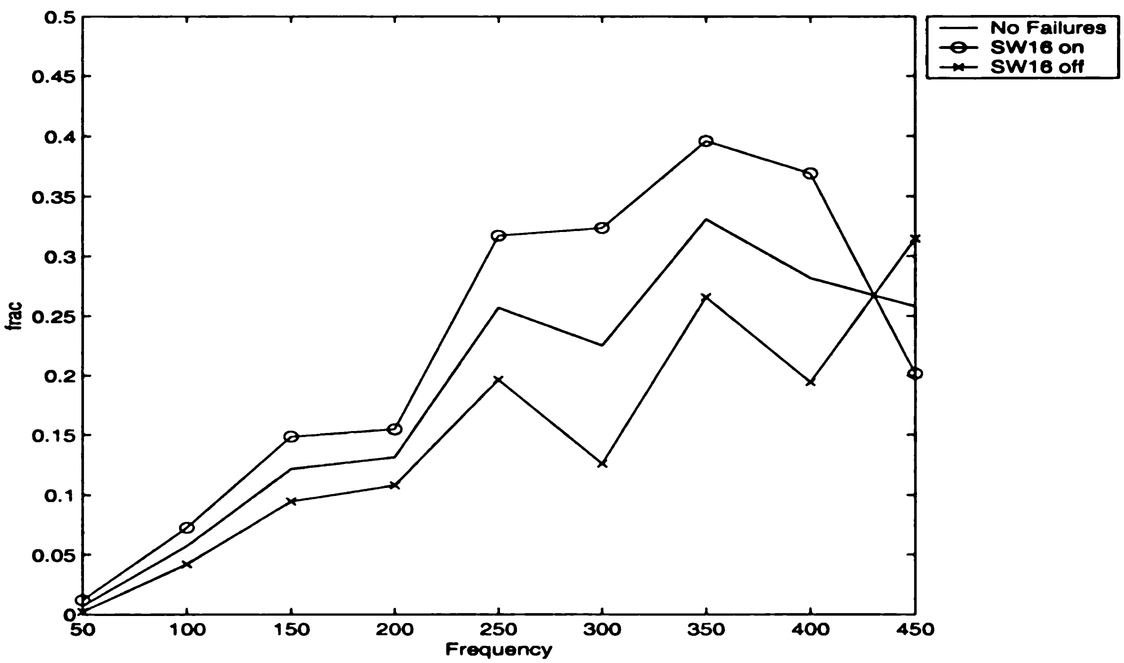

Figure 5.19. Single Failure of Switch 16.

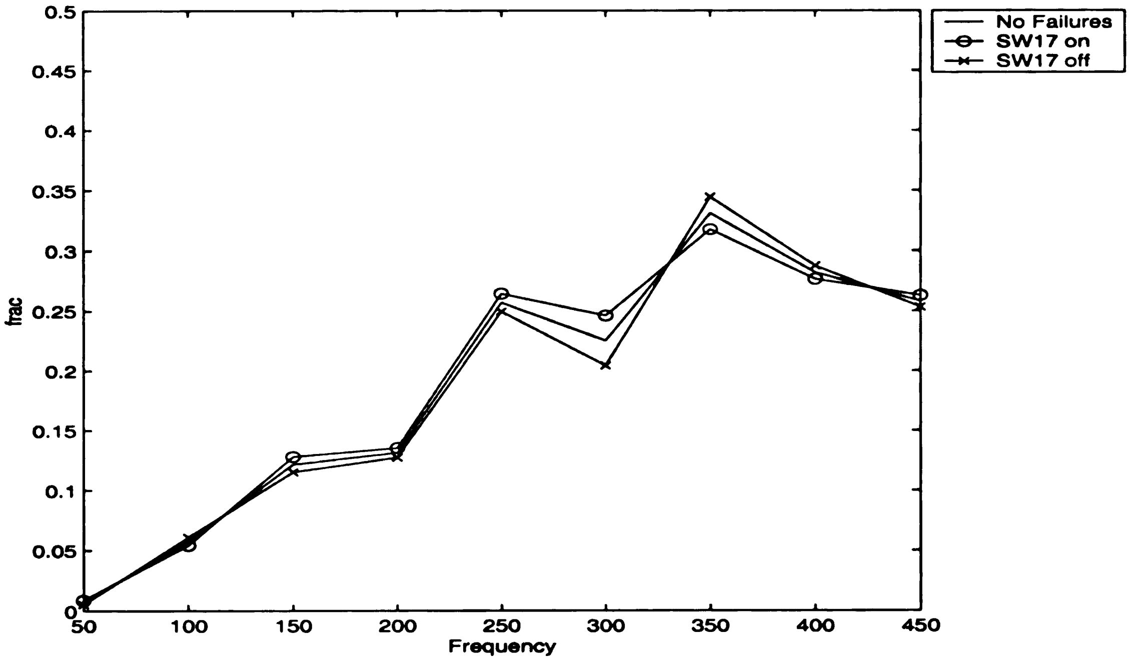

Figure 5.20. Single Failure of Switch 17.

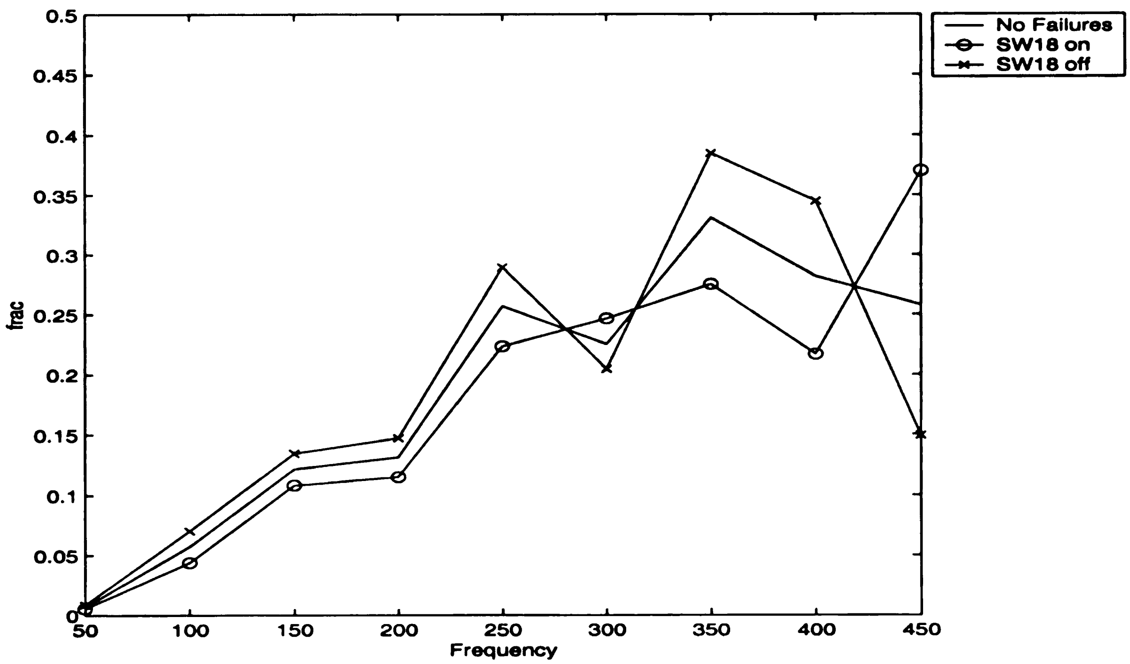

Figure 5.21. Single Failure of Switch 18.

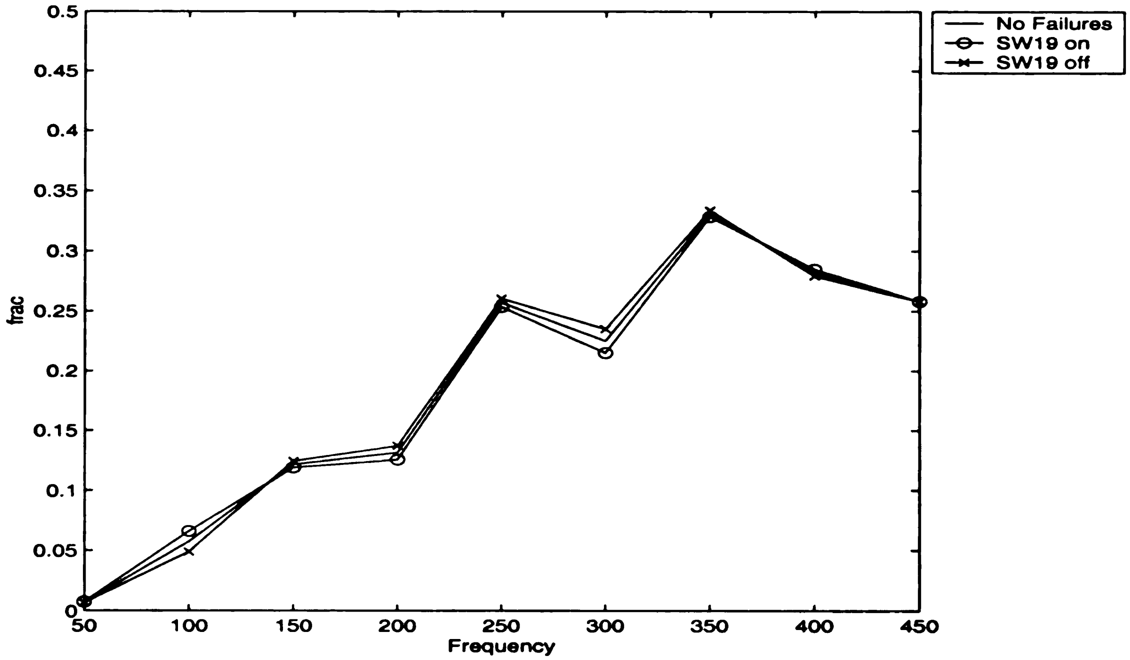

Figure 5.22. Single Failure of Switch 19.

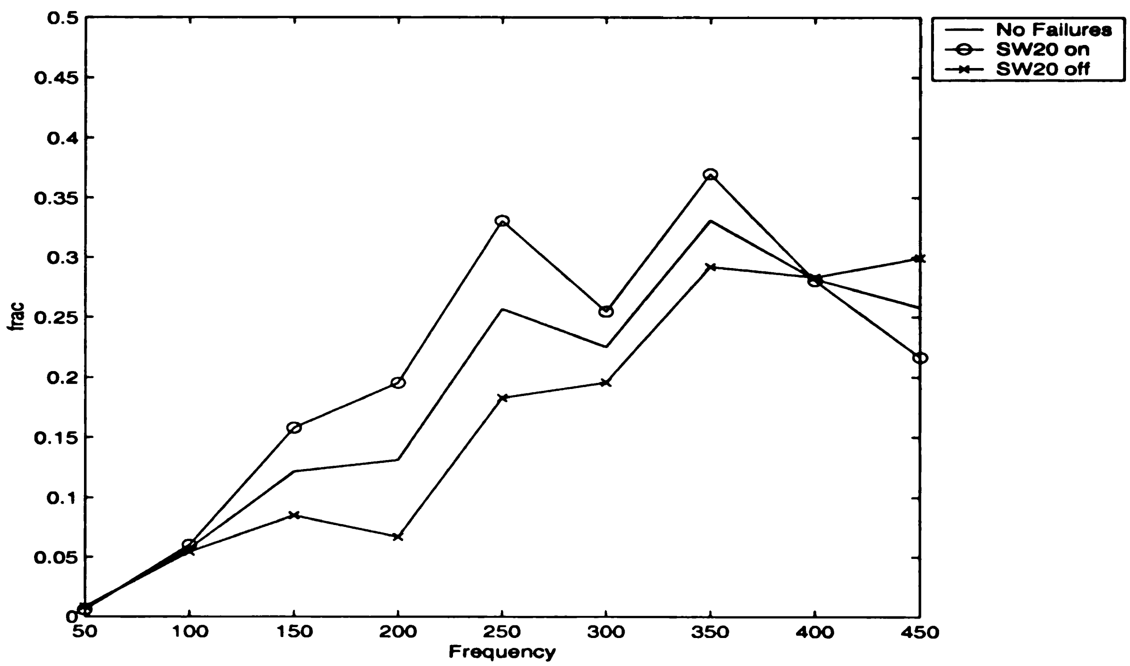

Figure 5.23. Single Failure of Switch 20.

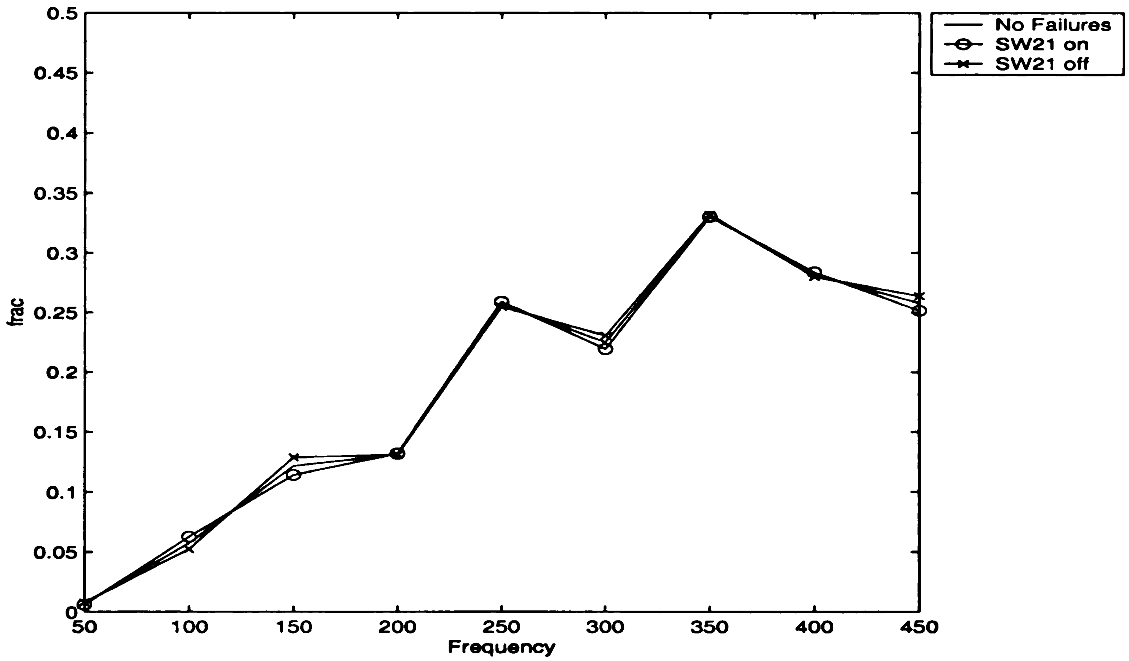

Figure 5.24. Single Failure of Switch 21.

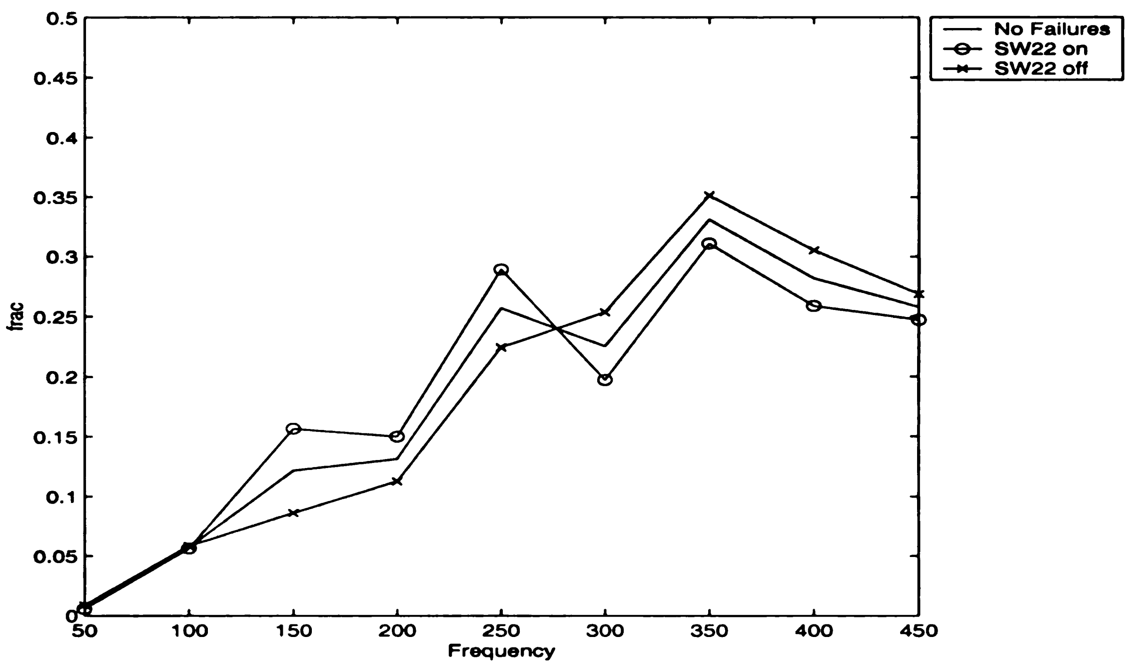

Figure 5.25. Single Failure of Switch 22.

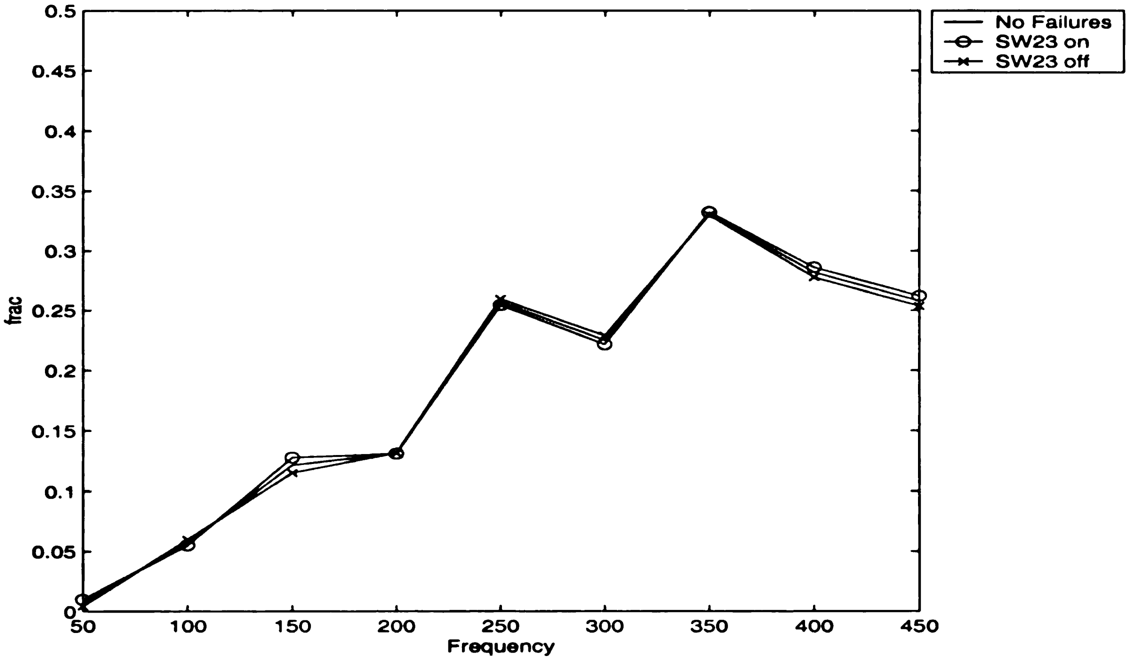

Figure 5.26. Single Failure of Switch 23.

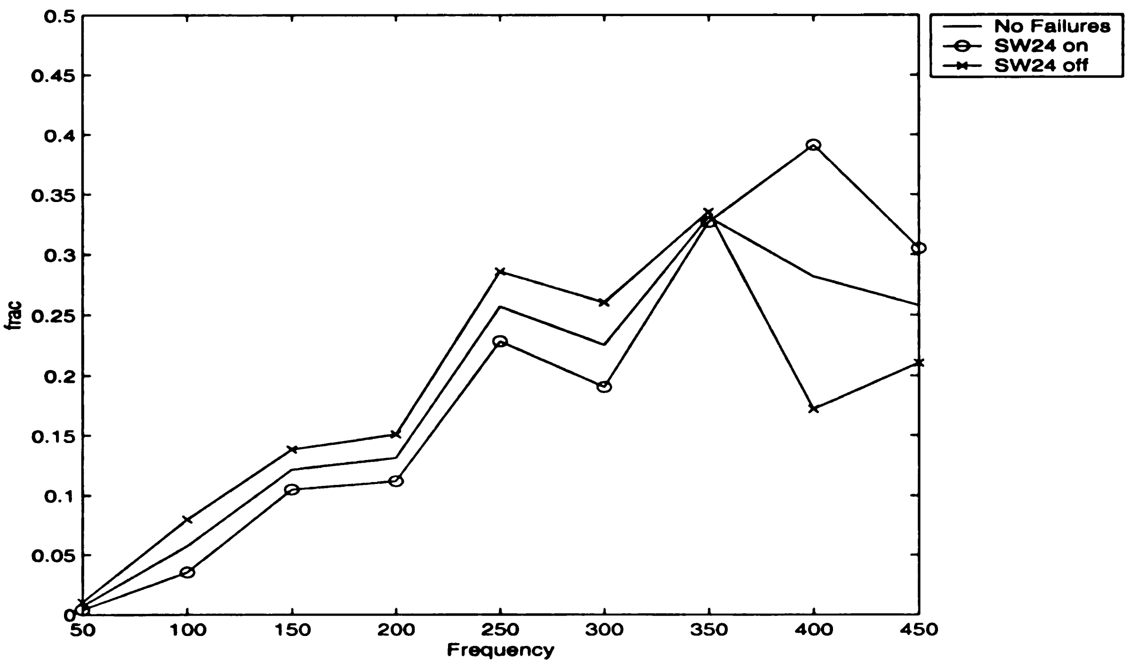

Figure 5.27. Single Failure of Switch 24.

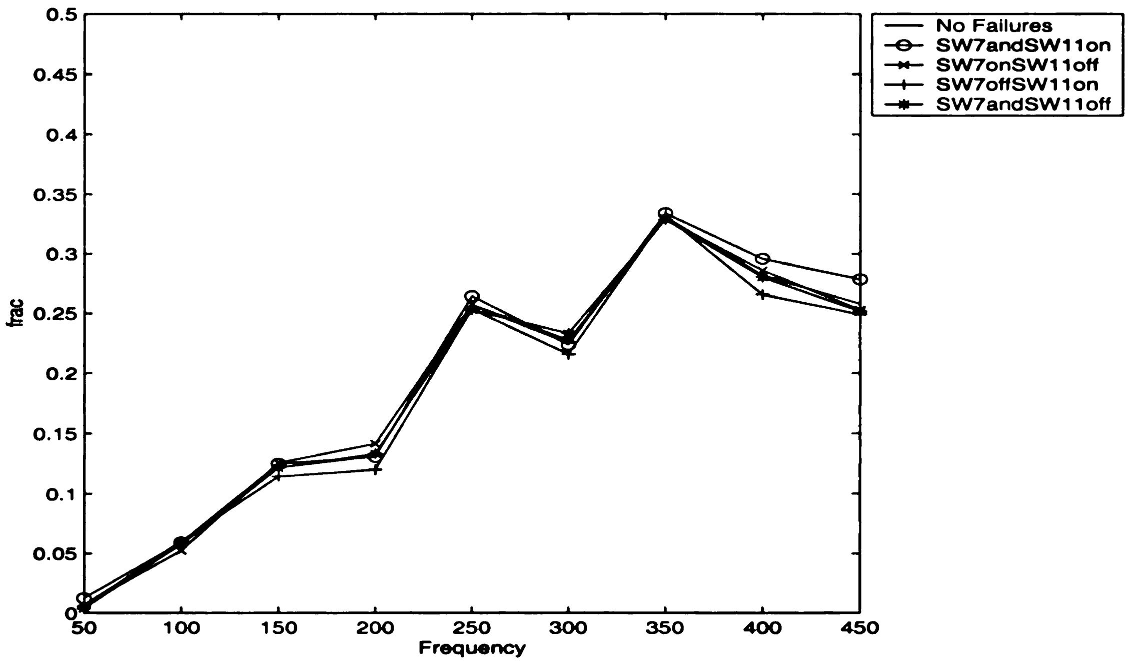

Figure 5.28. Multiple Failures Showing Minimal Degradation

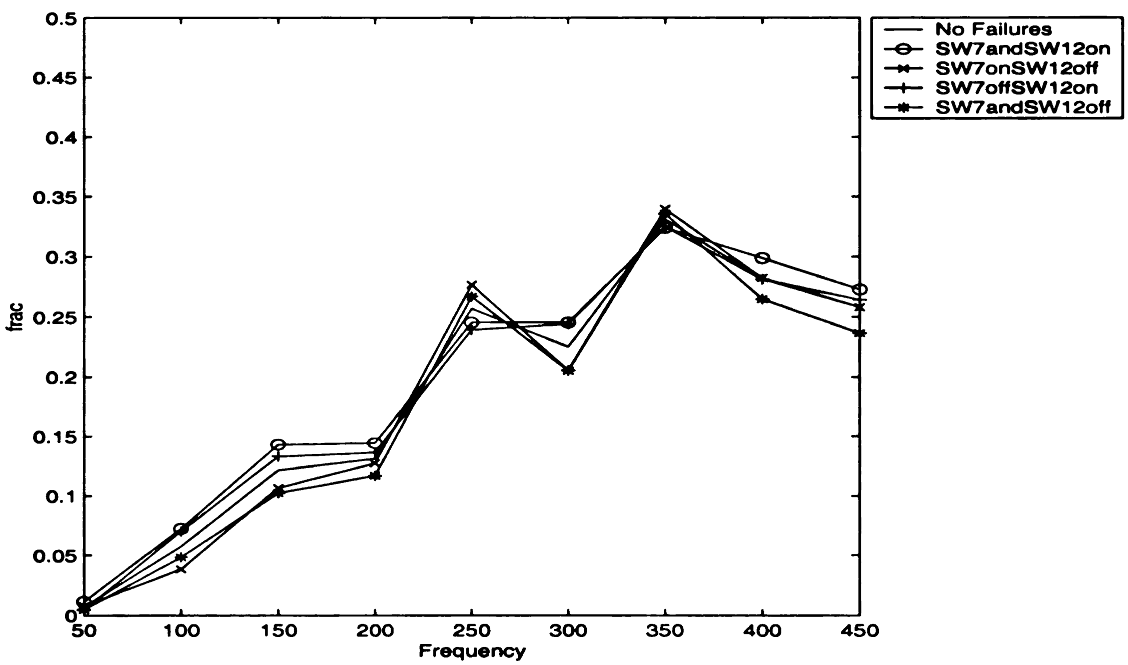

Figure 5.29. Multiple Failures Showing Intermediate Degradation

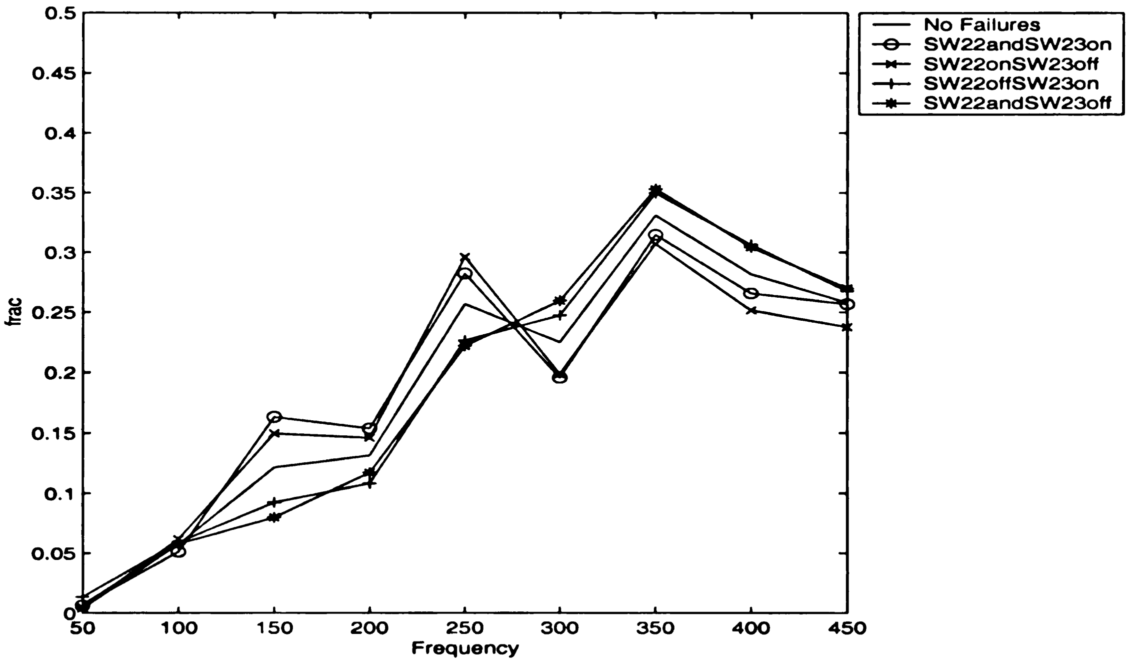

Figure 5.30. Large Degradation due to One Switch under Multiple Failures

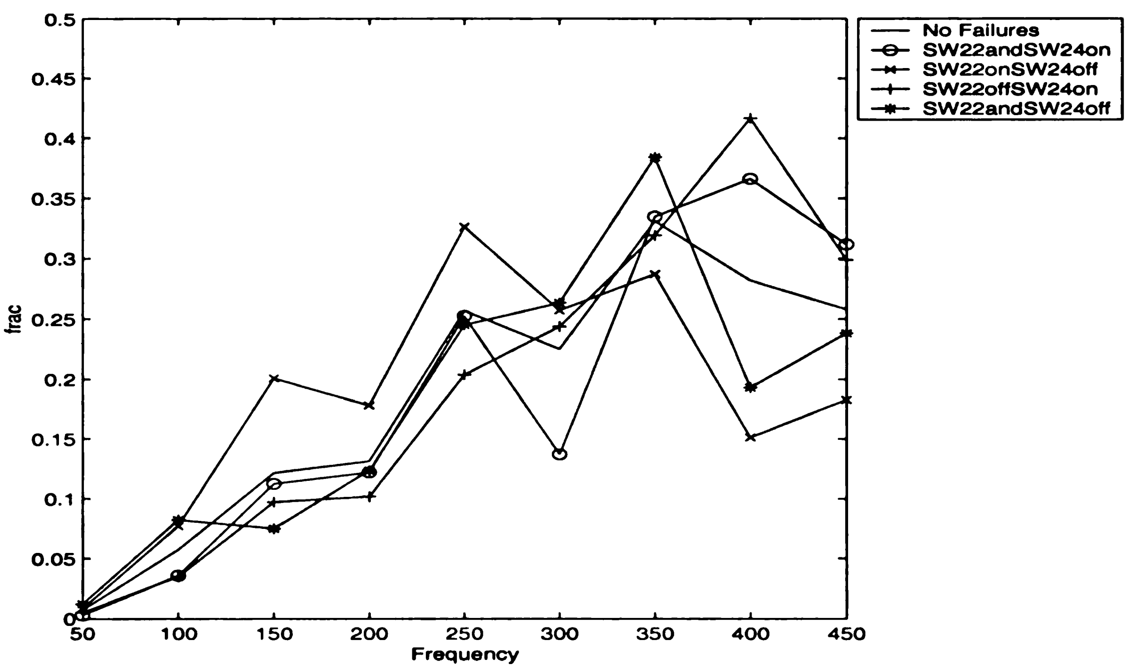

Figure 5.31. Large Degradation due to Two Switches under Multiple Failures

CHAPTER 6

CONCLUSIONS

This thesis has presented an experimental perspective of a new class of antennas known as self-structuring antennas, or SSAs. General concepts regarding the SSA system, design, and testing process were covered. Several conclusions can be drawn from the research done in constructing this thesis.

First, a prototype SSA was constructed using a printed circuit board for the skeleton of the system. This prototype was built and tested, providing some insight into the operation of the SSA. Specifically, the conclusions that were drawn from this prototype were the necessity of consideration of orientation and size in the design of the SSA. These factors are to be used in the design of future generations of the self-structuring antenna. Also, the use of a genetic algorithm as an efficient control for the SSA was shown using this prototype. This has implications in the use of smart algorithms for control of the SSA in future applications, giving reason to test the effectiveness of other algorithms, such as ant colony optimization or simulated annealing.

Switch failure analysis performed during this research shone some light on several shortcomings of the SSA. First, the relatively small size of the self-structuring antenna prototypes figured directly into the results of testing at low frequencies. It was shown that the antenna performed poorly at these frequencies with or without switch failure. In addition, testing showed that switches controlling short antenna elements created a substantial degradation in performance under switch failure, especially at high frequencies. A conclusion drawn from this was that the antenna did not have the necessary redundancy to counteract the failure of a switch controlling a short antenna element. Perhaps more clearly, short elements can build longer ones, but the inverse is not true.

The shortcomings in the SSA designs that were exposed during this research give some insight into how future generations should be built. These allow the designer to take into account the application of the self-structuring antenna in deciding the necessary orientation and size of the template. One can also take into account the need for redundancy in the short antenna elements on a given template. All of these conclusions can be put together to create a good framework from which future SSA templates can be constructed.

APPENDICES

APPENDIX A

MT-BASIC CODE: GENETIC ALGORITHM FOR THE SELF-STRUCTURING ANTENNA

```
5 'This program works as the genetic algorithm for control of the SSA
6 'Authors: Brad Perry and Dr. Ed Rothwell, Michigan State University

10 STRING FULL$(32), F1$(30), F$, MSB$, LSB$, MSB2$, LSB2$
11 STRING BX$, Q$, MATE1$(30), MATE2$(30), XX$(30)
12 STRING OLDPOP$(30,30), NEWPOP$(30,30), OFF1$(30), OFF2$(30)
13 STRING A$(30), A1$(30), A2$(30), B$(30)
14 STRING B1$(30), B2$(30), C$(30), D$(30)
15 STRING MX$(30), TEMP$(30), MUT$, MAX$(30), BREEDPL$(30,60)

20 INTEGER LS2B, MS2B, MS1B, LS1B
21 INTEGER STRINT, QQ, SUM, GEN, II
22 INTEGER ISEED, N, M, X, Y, ZX, ZZ, IRND, I1, I2
23 INTEGER POPSIZ, I, J, K, IMUT, AA, BB, JJ, KK, PP, P1, P

30 REAL SEED, XRND, TFIT, TMAX
31 REAL RDG, VOLTS, CH(10), CC, V1
32 REAL FIT(30), OBJ, CMULT, DELTA, AF, BF, HH
33 REAL PCROSS, PMUT, FRAC(30), RR
34 REAL SUMFIT, FMIN, FMAX, AVG, EE
35 REAL GOODFIT

50 POPSIZ = 30 'BE SURE TO CHANGE SIZE OF STRING TO MATCH
51 PCROSS = 0.7: PMUT = 1./POPSIZ
52 CH(1) = 3.05 'CHANNEL ARRAY TO BE USED WITH KEYPAD
53 GEN = 1
54 CMULT = 1.5: GOODFIT = 0.9

1020 'This is the seed for a random number generator
1021 SEED = .45342

1022 DEF XRND:'Randomly generates a number between 0 and 1
```

```
1023 SEED = SEED * 997. : ISEED = SEED : SEED = SEED - ISEED
1024 XRND = SEED
1025 FNEND

1026 DEF IRND(I1,I2):
1027 'Randomly selects an integer between I1 and I2
1027 IRND = I1 + XRND * (I2 - I1 +1)
1028 FNEND

1040 'This function converts a binary string
1041 'to an equivalent 8-bit integer

1044 DEF STRINT(BX$)
1045 QQ = 128: SUM = 0
1046 FOR X= 1 TO 8
1047 Q$ = MID$(BX$,X,1)
1048 IF Q$ = "1" THEN SUM = SUM + QQ
1049 QQ = QQ/2
1050 NEXT X
1051 STRINT = SUM
1052 FNEND

1200 'The next two functions perform crossover of two chromosomes
1220 DEF C$ (A$,B$)
1221 IF N = 30 THEN C$ = A$:GOTO 1226
1222 IF N = 0 THEN C$ = A$:GOTO 1226
1224 A1$ = MID$(A$,1,N)
1225 B2$ = MID$(B$,N+1,M)
1226 C$ = CONCAT$(A1$,B2$)
1227 FNEND

1230 DEF D$ (A$,B$)
1231 IF N = 30 THEN D$ = B$:GOTO 1236
1232 IF N = 0 THEN D$ = B$:GOTO 1236
1234 A2$ = MID$(A$,N+1,M)
1235 B1$ = MID$(B$,1,N)
1236 D$ = CONCAT$(B1$,A2$)
```

```
1237 FNEND

1240 DEF OBJ(V1)
1241 CC = CH(1)
1242 OBJ = 0.4 / ((V1-CC)*(V1-CC)+0.4)
1243 FNEND

1263 '**************************************************

1264 'Initializing the population
1265 FOR I = 1 TO POPSIZ
1266 NEWPOP$(I) = ""
1267 FOR J = 1 TO 30
1268 IF XRND < 0.5 THEN NEWPOP$(I) = CONCAT$(NEWPOP$(I),"0"): GOTO 1270
1269 NEWPOP$(I) = CONCAT$(NEWPOP$(I),"1")
1270 NEXT J
1275 NEXT I
1280 FOR K = 1 TO 30
1285 OLDPOP$(K) = NEWPOP$(K)
1290 NEXT K

1400 INITLCD 2,2,4

1415 'The following commands initialize U14 and U15 as output ports
1416 OUT 179, 128
1417 OUT 195, 128

1500 START 1,5
1505 START 2
1515 GOTO 1515

2999 '****************************************************
3000 'The following task reads and interprets the AGC Voltage level
3100 TASK 1
3101 'The following commands initialize the LCD display
3103 DEVICE 6
3110 RDG = ADC(4)
```

```
3120 VOLTS = (RDG*5.)/1024.
3200 EXIT

3999 '*****************************************************
4000 'Task 2 evaluates a 30 bit binary string & outputs it
4011 TASK 2
4020 FOR Y = 1 TO POPSIZ
4030 F1$ =  OLDPOP$(Y)
4040 '32 bit strng to be broken into 4 8-bit strings:
4041 FULL$ = CONCAT$("00",F1$)

4050 MSB$ = MID$(FULL$,1,8)
4060 MSB2$ = MID$(FULL$,9,8)
4070 LSB2$ = MID$(FULL$,17,8)
4080 LSB$ = MID$(FULL$,25,8)

4135 'Using a function defined above,
4136 'converting the strings to integers
4141 LS1B = STRINT(LSB$)
4150 LS2B = STRINT(LSB2$)
4160 MS2B = STRINT(MSB2$)
4170 MS1B = STRINT(MSB$)

4187 'The following commands reset and set
4188 'relays based on above integers
4189 OUT 177, PEEK(ADR(LS1B))
4190 OUT 178, PEEK(ADR(LS2B))
4200 OUT 193, PEEK(ADR(MS2B))
4210 OUT 194, PEEK(ADR(MS1B))

4213 'The following commands send information to the LCD display
4214 DEVICE 6
4215 OUTLCD 192
4216 PRINT "                               ";
4217 OUTLCD 192
4220 PRINT "J5B ";
4225 OUTLCD 196
```

```
4230 PRINT PEEK(ADR(LS1B));
4233 OUTLCD 199
4234 PRINT " J5C ";
4235 OUTLCD 204
4240 PRINT PEEK(ADR(LS2B));
4243 OUTLCD 207
4244 PRINT " J7B ";
4245 OUTLCD 212
4250 PRINT PEEK(ADR(MS2B));
4253 OUTLCD 215
4254 PRINT " J7C ";
4255 OUTLCD 220
4260 PRINT PEEK(ADR(MS1B));
4270 OUTLCD 222
4271 PRINT "          ";
4272 FIT(Y) = OBJ(VOLTS)
4275 TMAX = 0
4276 TFIT = OBJ(VOLTS)
4277 OUTLCD 128
4278 PRINT TFIT;
4279 IF TFIT > TMAX THEN TMAX = TFIT
4280 IF TFIT >= GOODFIT THEN GOTO 4276
4281 FIT(Y) = TMAX
4380 IF Y = POPSIZ THEN GOSUB 5000
4381 IF Y = POPSIZ THEN GOSUB 5300
4900 OUTLCD 154
4910 PRINT "   ";Y;" ";
4999 NEXT Y

5000 'Subroutine that performs preselection, selection and crossover
5002 '*********************************************************

5003 'COMPUTE AND DISPLAY STATISTICS
5004 SUMFIT = FIT(1)
5005 FMIN = FIT(1)
5006 FMAX = FIT(1)
5007 MAX$ = OLDPOP$(1)
```

```
5008 FOR JJ = 2 TO 30
5009 EE = FIT(JJ)
5010 SUMFIT = SUMFIT + EE
5011 IF EE > FMAX THEN FMAX = EE: MAX$ = OLDPOP$(JJ)
5012 IF EE < FMIN THEN FMIN = EE
5013 NEXT JJ
5014 AVG = SUMFIT/30.
5015 START 3

5020 'FITNESS SCALING ALGORITHM
5021 IF FMAX < 1.5 * AVG THEN GOTO 5100
5025 DELTA = FMAX - AVG: IF DELTA <= 0.0 THEN DELTA = 0.00001
5030 AF = (CMULT - 1.0)* AVG / DELTA
5035 BF = (1.0 - AF)* AVG
5040 HH = AF * FMIN + BF
5050 IF HH > 0.0 THEN GOTO 5080
5060 DELTA = AVG - FMIN: IF DELTA <= 0.0 THEN DELTA = 0.00001
5070 AF = AVG / DELTA: BF = -AF * FMIN
5080 FOR II = 1 TO POPSIZ
5085 FIT(II) = AF* FIT(II) + BF
5090 IF FIT(II) <= 0.0 THEN FIT(II) = 0.00001
5098 NEXT II

5100 'BEGIN PRESELECTION
5101 KK = 0
5102 FOR P = 1 TO POPSIZ
5103 IF FIT(P) >= (1.2*AVG) THEN FIT(P) = 2. * FIT(P)
5104 RR = FIT(P) / AVG : PP = RR
5105 FRAC(P) = RR - PP
5106 IF PP <= 0 THEN GOTO 5109
5107 FOR P1 = 1 TO PP: KK = KK + 1
5108 BREEDPL$(KK) = OLDPOP$(P): NEXT P1
5109 NEXT P

5110 P = 0
5111 IF KK >= POPSIZ THEN GOTO 5190
5112 P = P + 1: IF P > POPSIZ THEN P = 1
```

```
5113 IF FRAC(P) < 0 THEN GOTO 5111
5114 IF XRND > FRAC(P) THEN GOTO 5111
5115 KK = KK + 1 : BREEDPL$(KK) = OLDPOP$(P)
5116 FRAC(P) = FRAC(P) - 1.
5117 GOTO 5111
5150 'END PRESELECTION

5180 'BEGIN SELECTION AND CROSSOVER
5190 FOR AA = 1 TO (POPSIZ/2)
5191 N = IRND(1,30)
5192 M = 30 - N

5193 ZX = IRND(1,POPSIZ)
5194 MATE1$ = BREEDPL$(ZX)
5195 BREEDPL$(ZX) = BREEDPL$(POPSIZ)
5196 POPSIZ = POPSIZ - 1

5197 ZZ = IRND(1,POPSIZ)
5198 MATE2$ = BREEDPL$(ZZ)
5199 BREEDPL$(ZZ) = BREEDPL$(POPSIZ)
5200 POPSIZ = POPSIZ - 1

5205 IF XRND > PCROSS THEN OFF1$ = MATE1$: OFF2$ = MATE2$: GOTO 5250

5215 OFF1$ = C$(MATE1$,MATE2$)
5216 OFF2$ = D$(MATE1$,MATE2$)

5250 NEWPOP$(AA) = OFF1$
5260 NEWPOP$(AA+15) = OFF2$
5262 NEXT AA
5263 POPSIZ = 30
5264 'END SELECTION AND CROSSOVER

5265 'Update the population
5270 FOR K = 1 TO POPSIZ
5280 OLDPOP$(K) = NEWPOP$(K)
5290 NEXT K
```

```
5291 GEN = GEN + 1
5295 RETURN

5300 'Subroutine that performs mutation of chromosomes
5301 FOR BB = 1 TO POPSIZ
5302 MX$ = OLDPOP$(BB)
5310 IF XRND > PMUT THEN GOTO 5395
5320 IMUT = IRND (1,30)
5330 IF MID$(MX$,IMUT,1) = "0" THEN MUT$ = "1": GOTO 5350
5340 MUT$ = "0"
5350 IF IMUT = 1 THEN MX$ = CONCAT$(MUT$,MID$(MX$,2,29)):GOTO 5390
5360 IF IMUT = 30 THEN MX$ = CONCAT$(MID$(MX$,1,29),MUT$):GOTO 5390
5370 TEMP$ = CONCAT$(MID$(MX$,1,IMUT-1),MUT$)
5380 MX$ = CONCAT$(TEMP$,MID$(MX$,IMUT+1,30-IMUT))
5382 OLDPOP$(BB) = MX$
5383 OUTLCD 138
5384 PRINT "                  ";
5385 OUTLCD 138
5390 PRINT "Mutate: ";IMUT;
5395 NEXT BB
5398 Y = 0
5400 RETURN
5499 EXIT

5999 'Task 3 outputs statistics computed in task 2
6000 TASK 3
6014 PRINT "GENERATION #: ";GEN; " FMIN: ";FMIN
6015 PRINT "  FMAX: ";FMAX;"  AVG= ";AVG
6017 CANCEL 3
6018 EXIT
```

APPENDIX B

BASIC CODE: NETWORK ANALYZER CONTROL

Professional BASIC code for controlling the HP 8510C network analyzer via HPIB, and for communicating with the FlashTrax controller via serial connection. Sets all 24 switches to random states as determined by random number read from file listr.txt.

```
Program: SSA_ED2.BAS
DECLARE SUB RunTest (isc&, device&)
DECLARE SUB Calibrate (isc&, device&)
DECLARE SUB ErrorTrap(device&)
DECLARE SUB FreqList (isc&, device&)
DECLARE SUB GetInfo ()
DECLARE SUB greeting ()
DECLARE SUB hp8510setup (isc&, device&)
DECLARE SUB init (isc&, device&)
DECLARE SUB Ioouts(device&, cmd\$)
DECLARE SUB MeasImpedance (isc&, device&, real.Z!, im.Z!)
DECLARE SUB MeasSWR (isc&, device&, swr!)
DECLARE SUB ReadyToMeasure (isc&, device&)
DECLARE SUB SpecifyParameters (isc&, device&)
DECLARE SUB waithere (t!)
DECLARE SUB randm ()

COMMON SHARED isc&, device& COMMON SHARED F
COMMON SHARED nstates%
COMMON SHARED twait
COMMON SHARED row%, col%
COMMON SHARED maxc%, maxr%, maxv% 'for download data

REM $INCLUDE: 'bigsetup'

' This program performs automated measurements of
' self-structuring antenna
' using randomly-generated states.
' The measurements are performed using an HP 8510 Network Analyzer.
```

```
'  Version of 15 September 2001

'  HP 8510 measurement implementation written by Chris Coleman from
'  HP 8720 measurement routines

'  HP 8720 initialization and error checking subroutines written
'  by John E. Ross III (taken from nam3.bas)

'  Modification for use with FlashTrax controller (www.multitrax.com)
'  by Ed Rothwell

REM Num% = RegisterFonts%("TMSRB.FON")
REM Num% = LoadFont%("N1/N2/N3/N4/N5/N6")

OPEN "debug.txt" FOR OUTPUT AS #10

twait = .03

' row% and col% are variables used by the DrawSSA and DrasSSAGrid
' subroutines.
row% = 4
col% = 1

' Define the address of the HPIB interface (isc&) and
' the HP 8510 (device&) isc& = 7 device& = isc& * 100 + 16

' These variables are used for downloading data via HPIB
maxc% = 10
maxr% = 20
maxv% = 4096

'-----Start here-------

'set up serial communications with controller

OPEN "com2:2400,n,8,1,cd0,ds0,cs0,op0,rs,rb2048" FOR OUTPUT AS #1
```

```
CALL greeting
CALL GetInfo
CALL hp8510setup(isc&, device&)
CALL RunTest(isc&, device&)

'-----------------------

'       Put the HP8510 into continual measurement mode when completed
'        cmd$ = "cont;"
'        CALL Ioouts(device&, cmd$): CALL ErrorTrap(device&)

CLOSE #1 CLOSE #10 CLS

END

SUB Calibrate (isc&, device&)

    CLS
    PRINT "Calibration Options for HP 8510 Network Analyzer:"
    PRINT "         0: Recall calibration from register 5 (default)"
    PRINT "         1: Fully manual calibration using front panel"
    INPUT "Choice: "; caltype$

SELECT CASE caltype$

    CASE "", "0"
        PRINT "Recalling calibration coefficients from register 5"
        cmd$ = "reca5;"
        CALL Ioouts(device&, cmd$): CALL ErrorTrap(device&)
    CASE "1"
        PRINT
        PRINT
        PRINT
        PRINT "Press LOCAL on HP 8510, calibrate S11 1-port,"
        PRINT "and press any key when finished"
        WHILE INKEY$ = ""
```

```
        WEND

END SELECT
        CLS
        PRINT "Calibration complete."
        INPUT "Save coefficients to register 5? (Y/N) (default=N): "
        INPUT calcoef$

SELECT CASE calcoef$

        CASE "", "n", "N"

        CASE "y", "Y"

                ' Save the result in register 5.
                cmd$ = "save5;"
                CALL Ioouts(device&, cmd$): CALL ErrorTrap(device&)

END SELECT

END SUB

SUB ErrorTrap (device&)

' This subroutine checks for errors
' Check for BASIC or HPIB errors

        GOSUB errorhp

' Check for 8510C errors by checking status of 8510C Error que

chkque:

' Read the status byte of the 8510C analyzer using the serial poll.

        CALL iospoll(device&, stat%): GOSUB errorhp
```

```
        LOCATE 23, 5: PRINT "ERROR ... STATUS = "; stat%;

        SLEEP (1)

' If bit three of the status byte is set,
' read the error queue summary list.

    IF (stat% MOD 16) > 7 THEN

' Tell 8510C to send out oldest error message in queue.

    cmd$ = "outperro;"
    CALL Ioouts(device&, cmd$): GOSUB errorhp

' Receive error messages.

    length% = 50
    errdata$ = SPACE$(length%)
    CALL ioenters(device&, errdata$, length%, actual%): GOSUB errorhp

'  Extract the error number from the string read in.

    errnum% = VAL(LEFT$(errdata$, 5))

'  Initialize the string counter to begin after the error number.

    i% = 9

'  Initialize the error message string.

    errid$ = ""

'  Extract the error message from the string one character at a time.

    DO UNTIL MID$(errdata$, i%, 1) = CHR$(34)
        errid$ = errid$ + MID$(errdata$, i%, 1)
        i% = i% + 1
```

```
    LOOP

'  Display error message on status line.

    LOCATE 23, 5: PRINT "HP 8510C ERROR "; errnum%; ": "; errid$;

'  Beep to warn operator.

    BEEP

'  Pause and then recheck status byte.

    SLEEP 1
    GOTO chkque
    END IF

errorhp:  IF PCIB.ERR <> NOERR THEN ERROR PCIB.BASERR

END SUB

SUB FreqList (isc&, device&)

        cmd$ = "editlist;"
        CALL Ioouts(device&, cmd$): CALL ErrorTrap(device&)

        cmd$ = "sadd;"
        CALL Ioouts(device&, cmd$): CALL ErrorTrap(device&)

        singlefreq! = F * 1!

        cmd$ = "cwfreq" + LTRIM$(STR$(singlefreq!) + "MHz") + ";"
        CALL Ioouts(device&, cmd$): CALL ErrorTrap(device&)

        cmd$ = "sdon;"
        CALL Ioouts(device&, cmd$): CALL ErrorTrap(device&)

        cmd$ = "lisfreq"
```

```
        CALL Ioouts(device&, cmd$): CALL ErrorTrap(device&)

        cmd$ = "sseg1;"
        CALL Ioouts(device&, cmd$): CALL ErrorTrap(device&)


END SUB

SUB GetInfo

CLS

        INPUT "Enter frequency (MHz) to be investigated"; F
        INPUT "Enter number of antenna states to check"; nstates%

END SUB

SUB greeting

CLS

PRINT "Welcome to the automated self-structuring antenna testing system"
PRINT
PRINT "Press any key to continue"

WHILE INKEY$ = ""
WEND

END SUB

SUB hp8510setup (isc&, device&)

    CLS
    PRINT "Initializing HP 8510 Network Analyzer"
    PRINT
    PRINT
    CALL init(isc&, device&)
```

```
'   PRINT "Got past Init.  Press any key to continue"
'   WHILE INKEY$ = ""
'   WEND

    CALL SpecifyParameters(isc&, device&)
'   PRINT "Got past SpecifyParameters.  Press any key to continue"
'   WHILE INKEY$ = ""
'   WEND

    CALL FreqList(isc&, device&)
'   PRINT "Got past Freqlist.  Press any key to continue"
'   WHILE INKEY$ = ""
'   WEND

    CALL Calibrate(isc&, device&)
'   PRINT "Got past Calibrate.  Press any key to continue"
'   WHILE INKEY$ = ""
'   WEND

    CALL ReadyToMeasure(isc&, device&)
'   PRINT "Got past Readytomeasure"
    PRINT "HP 8510 now set up.  Press any key to continue"
    WHILE INKEY$ = ""
    WEND

END SUB

SUB init (isc&, device&)

' This subroutine initializes the HPIB and HP 8510C Network Analyzer.

' Reset the HPIB interface

      CALL ioreset(isc&)

' Define a system time-out of 10 seconds
```

```
        time! = 10
        CALL iotimeout(isc&, time!): CALL ErrorTrap(device&)

'  Abort any HPIB transfers

        CALL ioabort(isc&): CALL ErrorTrap(device&)

'  Clear the HPIB interface

        CALL ioclear(isc&): CALL ErrorTrap(device&)

'  Disable End-Or-Identify mode for transferring data

        CALL ioeoi(isc&, 0): CALL ErrorTrap(device&)

' Preset the 8510C

        CALL Ioouts(device&, "pres;"): CALL ErrorTrap(device&)

' Enable HPIB debug mode for 8510C

        CALL Ioouts(device&, "debuon;"): CALL ErrorTrap(device&)

END SUB

SUB Ioouts (device&, cmd$)

'  This subroutine outputs a string command to the HPIB device

        length% = LEN(cmd$)
        CALL iooutputs(device&, cmd$, length%)

END SUB

SUB MeasImpedance (isc&, device&, real.Z!, im.Z!)
```

```
' This is the important measurement
        cmd$ = "sing;"
        CALL Ioouts(device&, cmd$): CALL ErrorTrap(device&)

        DIM impedance(9)
        max.elem% = 10
        act.elem% = 0
        DIM frqd1!(1 TO 2, 0 TO 1), xr(0), xi(0)
        STATIC frqd1!

        t0! = TIMER
        cmd$ = "outpmark;"
        CALL Ioouts(device&, cmd$): CALL ErrorTrap(device&)
        CALL ioenter(device&, reply.1!, reply.2!, stim!)
        CALL ErrorTrap(device&)
        t! = TIMER - t0!
        real.Z! = reply.1!
        im.Z! = reply.2!
        PRINT
        PRINT
        PRINT "Impedance = "; real.Z!; " + j"; im.Z!; " Ohms"
        PRINT "Transfer time = "; t!; " seconds"
        PRINT "Stimulus = "; stim!
        PRINT
        PRINT "press any key to continue"
        WHILE INKEY$ = ""
        WEND

END SUB

SUB MeasSWR (isc&, device&, swr!)

' This is the important measurement

        cmd$ = "sing"
        CALL Ioouts(device&, cmd$): CALL ErrorTrap(device&)
```

```
        cmd$ = "outpmark"
        CALL Ioouts(device&, cmd$): CALL ErrorTrap(device&)
        CALL ioenter(device&, reply!): CALL ErrorTrap(device&)
        swr! = reply!

END SUB

SUB ReadyToMeasure (isc&, device&)

'        cmd$ = "lisfreq"
'        CALL Ioouts(device&, cmd$): CALL ErrorTrap(device&)
'        cmd$ = "sseg1;"
'        CALL Ioouts(device&, cmd$): CALL ErrorTrap(device&)
'        cmd$ = "reca5;"
'        CALL Ioouts(device&, cmd$): CALL ErrorTrap(device&)
'        cmd$ = "swet?;"
'        CALL Ioouts(device&, cmd$): CALL ErrorTrap(device&)
'        CALL ioenter(device&, reply!): CALL ErrorTrap(device&)
'        PRINT
'        PRINT "Sweep time after single seg= "; reply!
         cmd$ = "mark1 " + LTRIM$(STR$(F * 1!) + "MHz") + ";"
         CALL Ioouts(device&, cmd$): CALL ErrorTrap(device&)
'        cmd$ = "smic;"
'        CALL Ioouts(device&, cmd$): CALL ErrorTrap(device&)
'        CALL MeasImpedance(isc&, device&, real.Z!, im.Z!)
'        CALL MeasMagPhase(isc&, device&, mag!, phase!)
'        PRINT "press any key to continue"
'        WHILE INKEY$ = ""
'        WEND

         cmd$ = "swr;"
         CALL Ioouts(device&, cmd$): CALL ErrorTrap(device&)

'        CALL MeasSWR(isc&, device&, swr!)
'        PRINT "SWR = "; swr!

END SUB
```

```
SUB RunTest (isc&, device&)

CLS

'RANDOMIZE TIMER

OPEN "result.dat" FOR OUTPUT AS #3
OPEN "listr.txt" FOR INPUT AS #4

'listr contains a file of random bytes stored as integers
'listr has no repeated 24-bit strings

' Write Header to files "result.dat" and "test.dat"

PRINT #3, "This is an output file for test of the SSA"
PRINT #3, "using a random setting of states."
PRINT #3, "Date test was performed: "; DATE$, "Time of test: "; TIME$
PRINT #3, "f = "; F; "MHz"
PRINT #3, "no. states="; nstates%
PRINT #3, "Using file listr.txt"

'Begin test BEEP PRINT :

PRINT
PRINT "Ready to perform test."
PRINT "Attach antenna to test port."
PRINT
PRINT "Press any key to continue"
WHILE INKEY$ = ""
WEND
BEEP

CLS

FOR i% = 1 TO nstates%
```

```
'Generate random byte for bank 1
p1% = 0
kk% = 1
FOR j% = 1 TO 8
  INPUT #4, iin%
  IF iin% = 1 THEN
    p1% = p1% + kk%
    PRINT #1, CHR$(1);
  ELSE
    PRINT #1, CHR$(0);
  END IF
  kk% = kk% * 2
  CALL waithere(.03)
NEXT j%
'Generate random byte for bank 2
p2% = 0
kk% = 1
FOR j% = 1 TO 8
  INPUT #4, iin%
  IF iin% = 1 THEN
    p2% = p2% + kk%
    PRINT #1, CHR$(1);
  ELSE
    PRINT #1, CHR$(0);
  END IF
  kk% = kk% * 2
CALL waithere(.03)
NEXT j%
'Generate random byte for bank 3
p3% = 0
kk% = 1
FOR j% = 1 TO 8
  INPUT #4, iin%
  IF iin% = 1 THEN
    p3% = p3% + kk%
    PRINT #1, CHR$(1);
  ELSE
```

```
    PRINT #1, CHR$(0);
  END IF
  kk% = kk% * 2
CALL waithere(.03)
NEXT j%


' Measure SWR
' and calculate reflection (p) and transmission (s) coefficient
  CALL MeasSWR(isc&, device&, swr!)
  p! = (swr! - 1) / (swr! + 1)
  s! = 1 - p!
  refl! = p!
  IF s! < 0 THEN
    s! = .01
    refl = 1 - s!
  END IF

  PRINT #3, i%, p1%, p2%, p3%, swr!

'write results to screen
  LOCATE 1, 1
  PRINT "Bank1="; p1%; "Bank2="; p2%; " Bank3="; p3%

  LOCATE 4, 64
  PRINT "num ="; i%
  LOCATE 5, 64
  LOCATE 2, 25
  PRINT "                      "
  LOCATE 2, 25
  PRINT "s   ="; s!
  LOCATE 3, 25
  PRINT "                       "
  LOCATE 1, 52
  PRINT "swr="; swr!

NEXT i%
```

```
LOCATE 6, 12
PRINT "Test Complete. Press any key to continue"
WHILE INKEY$ = ""
WEND
BEEP
CLOSE #3


END SUB

SUB SpecifyParameters (isc&, device&)

' Define start frequency
        startfreq! = F * 1!
        cmd$ = "star" + LTRIM$(STR$(startfreq!) + "MHz") + ";"
        CALL Ioouts(device&, cmd$): CALL ErrorTrap(device&)

' Define stop frequency
        stopfreq! = 1.5
        cmd$ = "stop" + LTRIM$(STR$(stopfreq!) + "GHz") + ";"
        CALL Ioouts(device&, cmd$): CALL ErrorTrap(device&)

' Define the number of points
        nop% = 3
        cmd$ = "poin" + LTRIM$(STR$(nop%)) + ";"
        CALL Ioouts(device&, cmd$): CALL ErrorTrap(device&)

' Turn on Averaging
        cmd$ = "averon128" + ";"
        CALL Ioouts(device&, cmd$): CALL ErrorTrap(device&)

'' Define the number of averages
'        nav% = 4
'        cmd$ = "opc?;averfact" + LTRIM$(STR$(nav%)) + ";"
'        CALL Ioouts(device&, cmd$): CALL ErrorTrap(device&)
```

```
END SUB

SUB waitforsrq (issc&, dev&)
STATIC SHARED PCIB.ERR, PCIB.BASERR, NOERR

           srq% = 1
checkstat: CALL iostatus(issc&, srq%, status%)
           IF PCIB.ERR <> NOERR THEN ERROR PCIB.BASERR
           IF status% = 0 THEN GOTO checkstat
           CALL iospoll(dev&, response%)
           IF PCIB.ERR <> NOERR THEN ERROR PCIB.BASERR
           IF (response% AND 68) <> 68 THEN GOTO checkstat

END SUB

SUB waithere (t)

'Pauses the program for t seconds '  t can be less than 1

t1 = TIMER WHILE TIMER - t1 < t WEND

END SUB
```

APPENDIX C

BASIC CODE: SSA CONTROL FOR SWITCH FAILURE ANALYSIS

Program: swrprog2.txt

```
200 GOSUB 5000
210 GOSUB 5100
220 GOSUB 5200
300 LPRINT"SSA SWR Measurement Program"
400 GOSUB 5100
401 IF ABUFFER<8 THEN GOTO 401
402 p1=0:k=1:for i=1 to 8
403 p=aread:If p=1 then p1=p1+k
404 k=k*2:next i
405 IF ABUFFER<8 THEN GOTO 405
406 p2=0:k=1:for i=1 to 8
407 p=aread:If p=1 then p2=p2+k
408 k=k*2:next i
409 IF ABUFFER<8 THEN GOTO 409
410 p3=0:k=1:for i=1 to 8
411 p=aread:If p=1 then p3=p3+k
412 k=k*2:next i
420 'Set i/o pins
422 OUT 192,P1:OUT 193,P2:OUT 194,P3
425 GOSUB 5050
426 LPRINT " B1=";P1;" B2=";P2;" B3=";P3
440 GOTO 400
500 END

5000 'Initialize LCD display
5010 OUT 224,56:OUT 224,56
5020 OUT 224,6:OUT 224,12
5030 RETURN 5050 'Clear LCD display
5060 OUT 224,1
5070 RETURN

5100 'Initialize auxilliary com port
5110 ACONFIG 2N81
```

```
5120 RETURN

5200 'Initialize i/o pins and set all pins low
5210 OUT 195,128
5211 OUT 193,0:OUT 192,0:OUT 194,0 5220 RETURN
```

APPENDIX D

CPLD CODE: CONTROL ALGORITHM FOR THE RESETTING OF LATCHING RELAYS

Program for the resetting of latching relays used on the SSA control board. Written in VHDL by Matt Freel as a part of an ECE capstone design.

```
entity rectrl is
    port(   a: in std_logic;
        b: in std_logic;
        s: in std_logic;
        o1: out std_logic;
        o2: out std_logic);
end rectrl;

architecture rectrl_behave of rectrl is

begin
    o1<=(s and a)or(not s and b);
    02<=(not s and a)or s and b);
end rectrl_behave;

entity cont is
    port(   i: in STD_LOGIC_VECTOR (1 to 15);
        o1: out STD_LOGIC_VECTOR (1 to 15);
        o2: out STD_LOGIC VECTOR (1 to 15);
        s: in STD_LOGIC );
end cont;

architechture cont_arch of cont is component rectrl is
    port (  a: in std_logic;
        b: in std_logic;
        s: in std_logic;
        o1: out std_logic;
        o2: out std_logic);
end component;
```

```
signal zero: std_logic;

begin
r1: rectrl port map(i(1), zero, s, o1(1), o2(1));
r2: rectrl port map(i(2), zero, s, o1(2), o2(2));
r3: rectrl port map(i(3), zero, s, o1(3), o2(3));
r4: rectrl port map(i(4), zero, s, o1(4), o2(4));
r5: rectrl port map(i(5), zero, s, o1(5), o2(5));
r6: rectrl port map(i(6), zero, s, o1(6), o2(6));
r7: rectrl port map(i(7), zero, s, o1(7), o2(7));
r8: rectrl port map(i(8), zero, s, o1(8), o2(8));
r9: rectrl port map(i(9), zero, s, o1(9), o2(9));
r10: rectrl port map(i(10), zero, s, o1(10), o2(10));
r11: rectrl port map(i(11), zero, s, o1(11), o2(11));
r12: rectrl port map(i(12), zero, s, o1(12), o2(12));
r13: rectrl port map(i(13), zero, s, o1(13), o2(13));
r14: rectrl port map(i(14), zero, s, o1(14), o2(14));
r15: rectrl port map(i(15), zero, s, o1(15), o2(15));
end cont_arch;
```

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] C.M. Coleman, *Self-Structuring Antennas*, Ph.D. Dissertation, Michigan State University, East Lansing, MI, 2002.

[2] C.M. Coleman, E.J. Rothwell, and J.E. Ross, "Self-structuring antennas," IEEE AP-S International Symposium and URSI Radio Science Meeting, Salt Lake City, Utah, July 16-21, 2000.

[3] J.E. Ross, E.J. Rothwell, C.M. Coleman, and L.L. Nagy, "Numerical simulation of self-structuring antennas based on a genetic algorithm optimization scheme," IEEE AP-S International Symposium and URSI Radio Science Meeting, Salt Lake City, Utah, July 16-21, 2000.

[4] C.M. Coleman, E.J. Rothwell, J.E. Ross, and L.L. Nagy, "Self-Structuring Antennas," *IEEE Antennas and Propagation Magazine,*, to appear, June 2002.

[5] B.T. Perry, C.M. Coleman, B.F. Basch, E.J. Rothwell, J.E. Ross, and L.L. Nagy, "Self-Structuring Antenna for Television Reception," IEEE AP-S International Symposium and URSI Radio Science Meeting, Boston, Massachusetts, July 8-13, 2001.

[6] C.M. Coleman, E.J. Rothwell, J.E. Ross, and L.L. Nagy, "Application of Two-Level Evolutionary Algorithms to Self-Structuring Antennas," IEEE AP-S International Symposium and URSI Radio Science Meeting, Boston, Massachusetts, July 8-13, 2001.

[7] C. Coleman, B. Perry, E. Rothwell, L. Kempel, J.E. Ross, and L. Nagy, "A Study of Simple Self-Structuring Antenna Templates," IEEE AP-S International Symposium and URSI Radio Science Meeting, San Antonio, TX, June 16-21, 2002.

[8] B. Perry, C. Coleman, E. Rothwell, L. Kempel, J. Ross, and L. Nagy, "Effect of Switch Failure on the Performance of a Self-Structuring Antenna," IEEE AP-S International Symposium and URSI Radio Science Meeting, San Antonio, TX, June 16-21, 2002.

[9] D.E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*. Reading, Mass.: Addison-Wesley, 1989.

[10] K. Mehrotra, C.K. Mohan, S. Ranka, *Elements of Artificial Neural Networks*. Cambridge, Mass: The MIT Press, 1997.

General References

[11] W. Medenhall, *Introduction to Probability and Statistics*, Duxbury Press, North Scituate, MA, 1975.

[12] J.H. Holland, *Adaptation in Natural and Artificial Systems.* Ann Arbor: The University of Michigan Press, 1975.

[13] R. Haupt, "An Introduction to Genetic Algorithms for Electromagnetics," *IEEE Transactions on Antennas and Propagation*, vol. 17, pp. 7-15, April 1995.

[14] D.S. Weile and E. Michielssen, "Geneic Algorithm Optimization Applied to Electromagnetics: A Review," *IEEE Transactions on Antennas and Propagation*, vol. 45, pp. 343-353, March 1997.

[15] J.M. Johnson and Y. Rahmat-Samii, "Genetic Algorithms in Engineering Electromagnetics," *IEEE Antennas and Propagation Magazine*, vol. 39, pp. 7-25, August 1997.

[16] Y. Rahmat-Samii and E. Michielssen, eds., *Electromagnetic Optimization by Genetic Algorithms.* New York: John Wiley & Sons, 1999.

[17] E. Bonabeau, M. Dorigo, and G. Theraulaz, *Swarm Intelligence: From Natural to Artificial Systems.* New York: Oxford University Press, 1999.