ELECTRICAL TESTING OF A CMOS BASELINE PROCESS

by

David Rodriguez

Memorandum No. UCB/ERL M94/63

30 August 1994

ELECTRICAL TESTING OF A CMOS BASELINE PROCESS

by

David Rodriguez

Memorandum No. UCB/ERL M94/63

30 August 1994

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

ELECTRICAL TESTING OF A CMOS BASELINE PROCESS

by

David Rodriguez

Memorandum No. UCB/ERL M94/63

30 August 1994

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

TABLE OF CONTENTS (cont.)

LIST OF TABLES

TABLE OF CONTENTS

LIST OF FIGURES

Acknowledgments

I would like to express my sincere gratitude to my research advisor, Professor Costas J. Spanos, for his valuable research guidance and continued support of my goals. I am also extremely grateful to Professor Jan Rabaey for his insightful comments during the review of this thesis, and to Loren Linholm for sharing his knowledge of characterizing lithography systems.

Many thanks are extended to Eric Boskin, whose patience and guidance proved invaluable from the outset of this research. Likewise, I am indebted to Crid Yu for his time and effort in sharing his knowledge of statistical analysis and characterization, particularly in the domain of lithography and etch performance. For his continued efforts in the cooperative design of the measurement software for these test structures, I am grateful to Vadim Gutnik. For answering my many FrameMaker and processing questions, I thank Sherry Lee, Tony Miranda and Shenqing Fang.

Many thanks are also extended to Raymond Chen, Sean Cunningham, Zeina Daoud, Mark Hatzilambrou, Shang-Yi Ma and Pamela Tsai. These and the rest of the persons mentioned in these acknowledgments have provided a very open and supportive environment throughout my stay here.

Finally, I would like to extend special thanks to my friends and family who have unconditionally supported me in my endeavors, in particular my wife-to-be, April Hachenburg, whose support and love has made this achievement and my happiness possible.

Chapter 1

Introduction

1.1 Background and Motivation

Over the years, the minimum feature size of typical CMOS integrated circuits has been decreasing at a rapid pace. This has lead to increasingly dense, complex circuits. The complexity of these product circuits makes them virtually useless for monitoring or debugging a fabrication process. As a result, ancillary test devices are often fabricated along with a product. These devices, or test structures, are measured in a much more timely manner to yield specific information regarding either the product circuit or the fabrication process. This information can then be used to predict circuit performance and yield, to monitor and control a process, or to provide debugging information when a process yields unacceptable results. The devices are often placed in the scribe lanes of a wafer, which are those areas between die that are later cut through to separate the die. For a more complete characterization, test structures are also included in the area between the scribe lanes.

This use of test structures is of particular interest to the Berkeley Computer-Aided Manufacturing (BCAM) group. In particular, an effort is underway to maintain a baseline process in the Berkeley Microfabrication Laboratory. This facility services research customers with a wide range of needs and recipes. As a result, careful attention must be paid to providing some form of control to the process. This control comes in the form of a baseline process run monthly. Test structures fabricated during this baseline run, along with scribe-lane test structures manufactured alongside product circuits, will be measured and the resulting data will be statistically analyzed. Analysis results will be used to determine

whether or not the process is in control, and if not what particular part of the process needs attention.

Although the primary use of the test structures will initially come in the form of monitoring the baseline process, a broader use of the test chip is expected. The structures lend themselves well to use in other BCAM areas of emphasis, such as performance and yield prediction, and modeling the manufacturability of circuit designs. As the research in these areas mature, the set of test structures proposed in this project will be available for process and device characterization.

1.2 Approach

The first step taken in designing the test structures was to survey currently used structures in both production and in research and development environments [4]-[15]. This provided an understanding of the various types of structures used, as well as the important issues in test structure design. The needs of the BCAM test chip were identified, and an appropriate set of test structures was designed and fabricated. Details of the layout and fabrication of the structures were determined with regard to the particular characterization required, along with adherence to certain standards defined for ease of probing. Furthermore, particular attention was paid to the organization of the test chip as a whole, especially with regard to the scribe lane. Software routines were written and organized within an automated probing system to further ease data gathering. The system was used to collect the characterization data, and the data was in turn verified against expected test structure results.

1.3 Thesis Organization

The work described in this document can be divided into four parts. The first part, described in Chapter 2, includes the survey of test structures and their various applications. Details of the structure designs chosen, including motivation for the structure's use

and verification results, are presented in Chapter 3. The third part, organization of the scribe lane and test chip as a whole, is described in Chapter 4. The automated probing system and accompanying software is described in Chapter 5, with a sample results and analyses following in Chapter 6. Finally, a conclusion is included in Chapter 7.

Chapter 2

Types of Electrical Test Structures

Test structures are used for a variety of purposes in the fabrication of integrated circuits. Test structures are used for device, circuit and process parameter extraction, as well as random fault and reliability testing. These five categories will be described in further detail in this chapter.

2.1 Test Structures for Device Characterization

Device parameters are those used to model devices for circuit simulation purposes. The most obvious example of such parameters are those that are used by SPICE to model transistor operation. One of two approaches may be taken when extracting device parameters. The first approach, direct parameter extraction, concentrates on designing test structures and parameter extraction routines such that a parameter can be extracted independently from the influence of other parameters. Contrary to extracting device parameters individually, the optimization method of parameter extraction involves collecting a general set of data representing all parameters. The device model's various parameters are then fitted to the measurements by an optimization routine, such that the extracted parameters can be used by the model to reproduce the measured data during simulation.

The choice of using direct extraction versus optimization is one which involves a number of trade-offs, and considerable time should be spent in studying this issue. For this reason, test structures for device parameter extraction, namely MOSFETs and capacitors,

where designed such that either method could be used. The method of parameter extraction is then left to the user.

2.2 Test Structures for Process Characterization

In semiconductor fabrication considerable emphasis is placed on the spatial uniformity, adherence to specifications, and lot to lot consistency of parameters which define a process. Monitoring process parameters such as doping concentration, contact and sheet resistance, critical dimension, oxide thickness, etc., play a key role in maintaining a process under control.

Measurements can be performed either optically or electrically. As the name suggests, optical measurements make use of optical information in extracting parameters, such as the width of a polysilicon line. Electrical measurements, on the other hand, make use of probing equipment to force a set of electrical inputs to the structure, and to measure the response. The test structure is designed such that according to basic electrical principles, the measured response can be translated to a single process parameter. Note that great care must be taken in designing the device, such that measurements depend on a single process parameter.

Although optical measurements are at times more accurate and revealing, their use is rather time consuming relative to automated, electrical measurements. As stated previously, one of the primary uses for these test structures will be to collect statistical information about a process. This necessitates a reasonably large set of measurements be performed in a short amount of time. It is for this reason that the primary emphasis of this thesis concerns electrical measurements.

2.3 Test Structures for Capturing Catastrophic Faults

The lack of uniformity in processing across a stepper field and wafer, and the impurity of the fabrication process often lead to the introduction of physical faults on a wafer. These faults come in a variety of forms, but generally result in loss of function prior to significant stressing of affected devices. Several examples of defects contributing to random faults are: breaks in metal lines due to the interaction of a contaminating particle and photoresist, shorts in metal lines caused by solid particles deposited on metal layers, oxide pinhole defects, and broken lines due to insufficient metal step coverage. Test structures were designed to electrically determine if shorting or breaking of metal lines appear on a wafer, and if so provide the approximate defect location for visual inspection, if desired.

2.4 Test Structures for Reliability Analysis

Reliability failures are those which occur after a significant amount of stress is placed on the structures of interest. This stress can come in a variety of forms, including overvoltage, current density, temperature, humidity, and radiation. The subsequent failure results from atomic motion or changes in ionic charge states [1].

As an example, consider perhaps the simplest and most common occurrence of reliability failure, electromigration. Electromigration is defined as the displacement of metal ions through a conductor resulting from the passage of direct current. This shift is caused by a modification of the normally random diffusion process to a directional one caused by charged carriers [2]. The greatest concern of electromigration is that over time sections of metal wire carrying a high current density become thinner, and eventually disconnect. A simple reliability test would then consist of stressing a long, thin metal line with a high current density over some time and checking for an open circuit.

Additional failures which may occur due to other forms of stressing include dielectric breakdown, charge injection, and corrosion [1]. In most cases, these failures can be moni-

tored by stressing structures used for device and process parameter extraction and reliability failure. For this reason, separate structures used exclusively for reliability failure analysis were not necessary, except for the case of monitoring oxide damage due to plasma etching. Therefore, random fault and reliability structures were included in the same category of test structure design in Chapter 3.

2.5 Test Structures for Circuit Characterization

Circuit parameters are those which characterize the performance of a complete integrated circuit (IC). A subset of such parameters are maximum operating frequency, average power dissipation, and drive capability. Since measuring these parameters from the product circuit is usually much too complex and time intensive, special test structures that mimic the behavior of the IC are introduced. These test structures, however, yield values which may be difficult to relate directly to some processing step. Nonetheless, they provide a reasonable estimate of the expected performance of the product circuit.

A common example of the type of test structure used to characterize an integrated circuit is the ring oscillator. Measuring the oscillation frequency of ring oscillators can provide a reasonable frequency range within which the process can be expected to provide functional product circuits.

It is obvious that test structures for circuit parameter extraction are fairly dependent on the product circuits to be characterized. Since the concern of this project was broader in scope than a specific circuit, test structures for circuit parameter extraction were not built. However, those structures designed for device parameter extraction can be used, along with circuit simulation, in order to obtain an estimate of a circuit's performance on the process of interest.

Chapter 3

Test Structure Design

This chapter provides the details concerning individual test structure design. In particular, each section will contain a general description of the test structure and its usage, along with design details and measured results. This chapter is organized according to the types of test structures defined in Chapter 2. Note, however, that as mentioned in the previous chapter test structures for characterizing ICs are not included, and structures used for reliability and catastrophic fault analysis are combined since they differ only in measurement technique.

Of particular interest to the BCAM group is characterizing a 2μm CMOS n-well process, this being the minimum size reliable device that can be produced by the CMOS baseline. Furthermore the process provides two metal layers, metal 1 and metal 2, with design rule metal linewidth and pitch of 3μm and 6μm, respectively, on each layer. Design rules for contacts require 3μm x 3μm contact cuts. All test structures described here can be easily scaled to accommodate finer features when they become available.

Common to all electrically measurable test structures designed is the array of pads used for probing. Each pad is a 100μmx100μm square of metal 2 over metal 1, with the appropriate vias. Ten pads are placed in a 2x5 array, with all pad spacings set to 100μm, in order to form the set of pads contacted with each drop of the probes onto the wafer. See Figure 1 for an example of the 2x5 pad set. The pad labels refer to the numbering scheme used by the autoprober, which is discussed further in Chapter 5. All test structures were labelled in polysilicon, metal 1 and metal 2, to aid in locating structures when using a

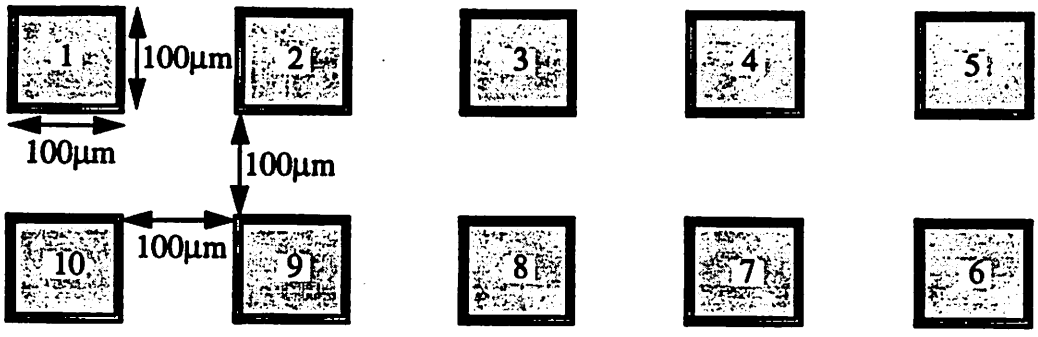


Figure 1 Configuration of standard pad set used for all electrical, DC measurements.

microscope. Labels were made as large as possible within the 100µm spacing between pads, given approximately 20µm spacing between the labels and the pads. The layers of the n-well, CMOS process are labelled as listed in Table 1. Furthermore, all numeric labels have units of microns, unless otherwise labelled. Finally, tables in this chapter which list measured data contain columns labelled "die X" and "die Y", which refer to the integer coordinates of the die within the wafer. Die configuration within the wafer will be discussed in detail in Chapter 4, but can be previewed by studying Figure 21.

Table 1 : Labels Used For Test Structure Layers

Label	layer	Label	layer
PO	polysilicon	NW	n-well
N+	n+ diffusion	M1	metal 1
P+	p+ diffusion	M2	metal 2

3.1 Test Structures for Device Characterization

3.1.1 Individual MOSFETs

The primary objective of device characterization is to extract enough physical information for modeling the electrical behavior of a transistor. Therefore, the prominent test structure in this category is the basic transistor. As mentioned previously, decisions concerning the specifics of device parameter extraction have been left for the potential users

of these structures. Nonetheless, an attempt has been made to provide a thorough set of structures to be used for that research.

Based on our 2μm CMOS process, and on device requirements for existing methods of direct parameter extraction and parameter optimization, a number of transistors were included. The transistor set consists of devices with drawn gate lengths of 1, 1.3, 1.5, 2, 3, 5, 10 and 25 μm. For each length listed, both an NMOS and a PMOS device of drawn width of 5, 10 and 50 μm were included. Each pad set can be used to individually probe three devices, each with the same gate length but a different width, resulting in a total of 16 pad sets.

With the exception of the body contact, there are no common terminals for any of these devices, in order to eliminate problems associated with parasitic leakage currents distorting results when probing a single device. The devices may be redesigned to share terminals if further research determines that this would not pose a problem. The layout of one subset of MOSFETs is shown in Figure 2.

3.1.2 4 x 4 MOSFET Arrays

Integrated circuit designers, particularly in the analog domain, often require electrically matched device parameters in a very localized area. In order to measure electrical

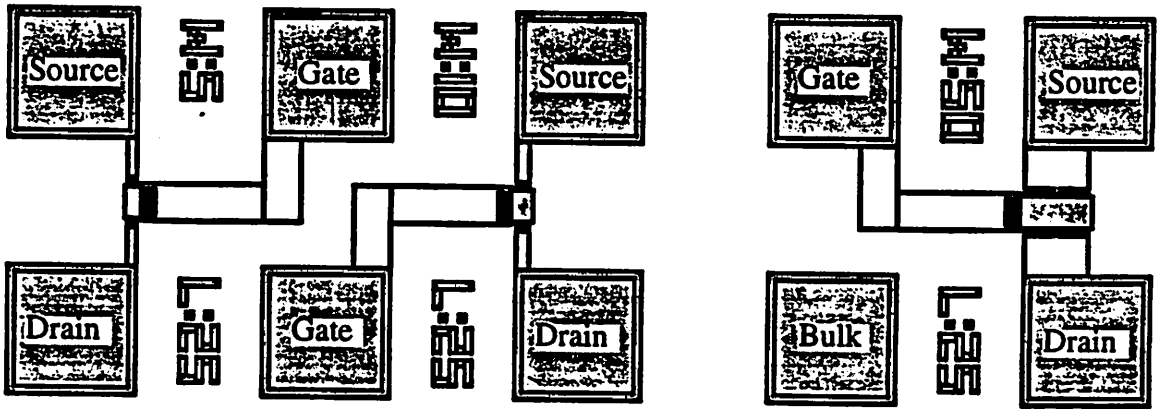


Figure 2 One subset of individually probed MOSFETs.

device mismatch, tightly coupled arrays of transistors were designed and fabricated. Both PMOS and NMOS transistors arrays have been included.

Figure 3 shows the MOSFET array in detail. Each transistor has a gate length of 2μm and a width of 10μm, with 5μm horizontal spacing and 15μm vertical spacing between source and drain regions of nearby transistors. All gates share a common lead while each of the remaining pads connects to four transistor sources or drains. Transistor drains are connected vertically via metal 2 connections, and share a common leads labelled *D1, D2, D3 and D4*, where the numbers 1 through 4 represent the array column number from left to right. Similarly, common sources are connected horizontally in metal 1, labelled *S1, S2, S3 and S4*, where the numbers 1 through 4 represent the array row number from top to bottom.

Probing a single transistor within the array is accomplished by contacting and applying inputs to the appropriate source and drain leads, while leaving the other leads floating. As a simple example, consider collecting data for an I_{DS}-V_{GS} curve on the transistor in the lower right corner of the array, at V_{DS} set to 5V. The first step would be to apply a ground to lead *S4*, and a voltage of 5V onto lead *D4*. The gate lead is appropriately swept in voltage while current readings are taken between *D4* and *S4*. All of the transistors in the array

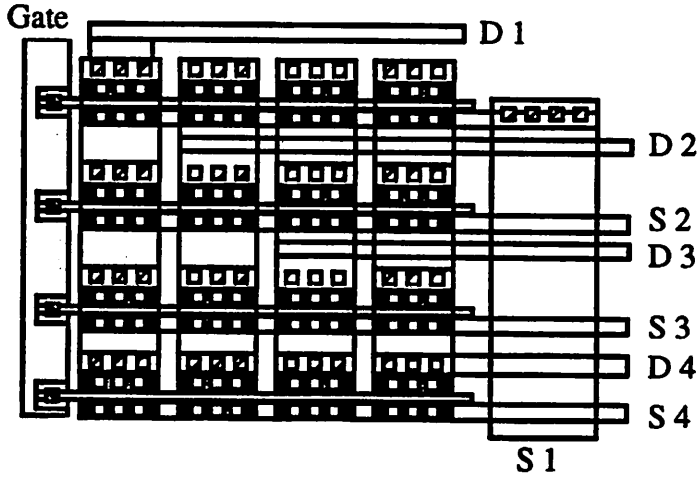


Figure 3 Close view of 4x4 array. Metal 2 connects drains in each row.

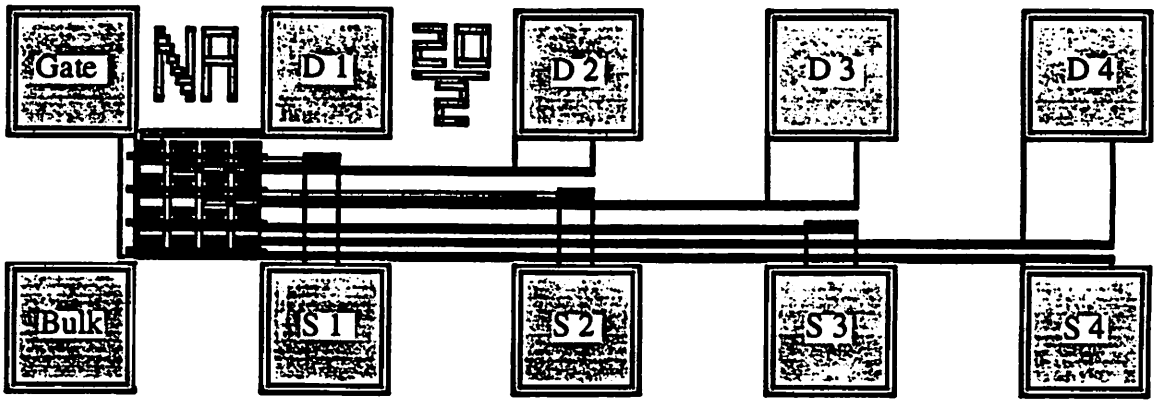


Figure 4 4 x 4 NMOS array within pad set. (W/L) = 20/2)

can be measured in a similar fashion, by simply asserting the appropriate source and drain leads.

Finally, Figure 4 shows the NMOS array within a pad set. The leads shown in Figure 3 are simply connected to pads, with an additional pad available as a contact to the substrate or well. The label "NA" indicates that the device is an NMOS array, while label "20/2" indicates that the transistors have gate lengths of 20μm and widths of 2μm.

3.1.3 Capacitors

The final set of test structures included for device parameter extraction consists of two large capacitors, illustrated in Figure 5. Capacitors have been reliably used for some time

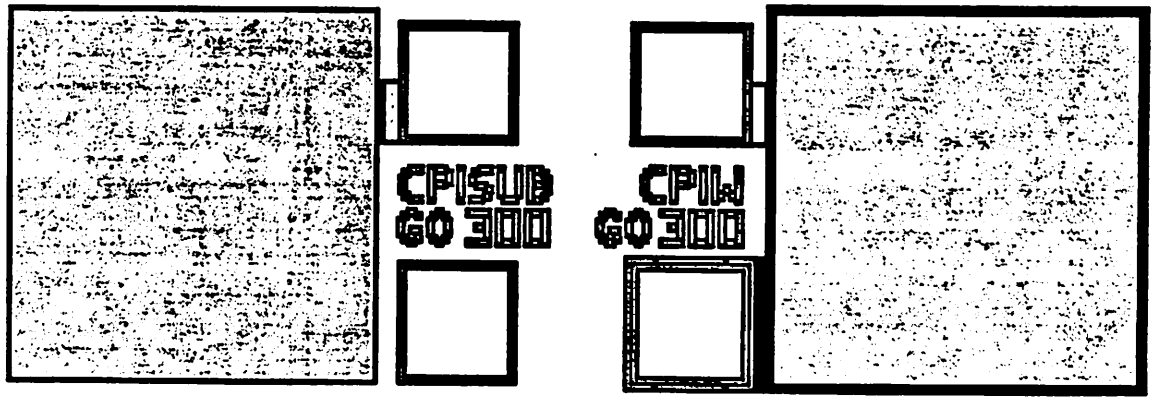


Figure 5 300μm x 300μm oxide capacitors between substrate and well.

in characterizing gate oxide. Two 300μm x 300μm capacitors have been included to perform this characterization. Frequency dependent C-V measurements can be applied to these capacitors to extract gate oxide thickness and analyze oxide integrity by monitoring trapped charge, oxide to substrate interface charge, and mobile ions in the oxide itself. These high-frequency measurement techniques fall outside the realm of the DC measurement techniques emphasized in this thesis, and as such will not be discussed here. Note that this structure has pads of 100μm x 100μm, which are spaced 100μm apart. However, this is the only structure which does not adhere to the standard 2x5 pad set.

3.2 Test Structures for Process Characterization

3.2.1 Contact Resistors

Previous studies have found that in a well-controlled CMOS process, contact resistance has a relatively large percentage variation [4]. This variation is not of great concern assuming that contact resistance is substantially less than a transistor's "on" resistance. However, sustained advances in manufacturing are resulting in smaller and thus more resistive contacts. The variation of contact resistance is then of growing concern, as contact resistance approaches that of a transistor's "on" resistance.

The basic structures used in the evaluation of contacts are contact chains and 4 terminal and 6 terminal contact resistors. Contact chains do not isolate the variability of the contacts themselves, and are therefore left for catastrophic fault analysis, to be discussed later. Contact resistors, however, are used to measure the interfacial resistance, or the resistance between the layers being contacted, as well as misalignment between the masks used to form the contact. The choice of using 4 versus 6 terminal contacts was a trade off between complexity of design and probing, and granularity of results.

For a better understanding of this trade off, consider the true, interfacial resistance, R_I, with respect to measured resistance, R_K, and a factor called flange resistance, R_F [5]. Note

in the four terminal contact resistor in Figure 6 that the voltage and current taps of the contact resistor are wider than the contact. This is necessary in order to account for misalignment between each layer and the contact cut. A simple Kelvin measurement [17] will result in a resistance higher than that of the true interfacial resistance, due to the parasitic current flow in the voltage taps. The flange resistance is a measure of this additional resistance. The following equations describe this relationship:

$$R_K = (V_1 - V_2)/I\,, \tag{1}$$

and

$$R_K = R_I + R_F\,. \tag{2}$$

The problem of extracting R_I and R_F from (2) from perfectly aligned contacts has been solved by what is known as the Thin-Film Model [5], which uses heuristics to predict the flange effect resistance. In a later study, Lieneweg and Sayah [6] propose a similar method of extracting R_I from misaligned four and six terminal contacts. The method includes procedures for estimating the actual misalignment in both horizontal, x, and vertical, y, directions. According to the study, four terminal contacts provide for the extraction of the average of two contacts' interfacial resistance, and can also calculate the sum of the x and y misalignment components. Six terminal contacts isolate the x and y misalignment components and provide a single interfacial resistance. Finally, Lieneweg and Sayah state that

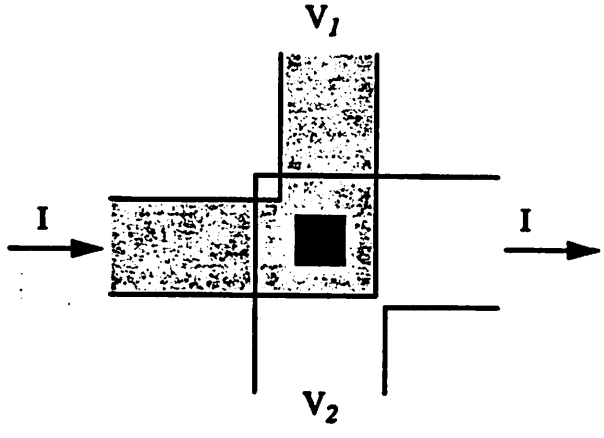


Figure 6 Four terminal contact resistor.

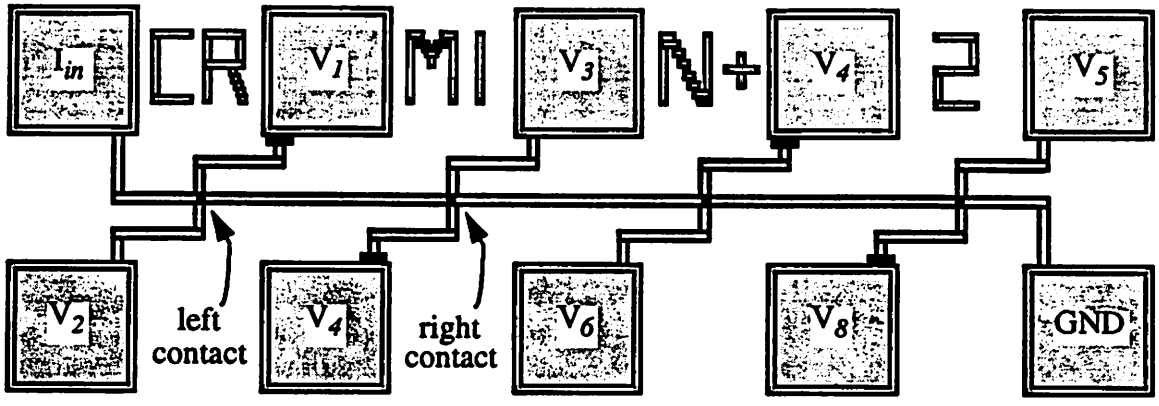


Figure 7 Cross contact resistor chain.

R_F can be treated as constant across a wafer, and R_{Kavg}, the average of a "right" and "left" contact, then reflects the variation of R_I. "Left" and "right" contacts simply differ in the position of the contact relative to the voltage taps. Figure 7 illustrates the two types of contact in a cross contact resistor chain. Note that the left contact has its n+ diffusion voltage tap extending upward, where n+ diffusion is illustrated by a darker line, and the right contact has its n+ diffusion voltage tap extending downward. Therefore, any vertical misalignment would draw the contact closer to one of the n+ diffusion voltage taps, but further from the other, causing a difference in voltage readings and therefore in resistance measurements.

A drawback of using six terminal contacts is that they cannot be chained as efficiently as four terminal contacts. Furthermore, they need an entire pad set per contact and require about twice as many measurements. Since the primary concern here is that of process control, the ability to accurately monitor contact resistance variation with four terminal contacts made it the test structure of choice. Using the four terminal contact also allowed for faster data collection by including four contacts per pad set. Figure 7 shows the final test structure design. The label "CR" indicates that the test structure is a cross-contact resistor chain. Labels "M1", "N+" and "2" indicate that contacts are between metal 1 and n-diffusion, and are 2μm on each side of the contact cut. The contact chains designed include

both 2μm and 3μm contacts between metal 1 and metal 2, n-diffusion, p-diffusion and polysilicon. Smaller contacts were also included to test the limits of the process, but were not chained in order to save area. See Figure 8 for an example of how these contacts are placed in a pad set. This smaller contact set contains 1.5μm contacts between metal 1 and n-diffusion, p-diffusion and poly, and 2.5 μm contacts between metal 1 and metal 2. The labels in Figure 8 indicate that the contact on the left is between metal 1 and polysilicon, and is 1.5μm on each side. Similarly, the contact on the right is between metal 1 and metal 2 and is 2.5μm on each side.

Taking contact resistance measurements on the cross-contact chain of Figure 7 is rather straightforward. Simply supply a current, I, which passes from pad I_{in} through the ground pad, and measure the voltage between the voltage taps of each contact.

Measurement error is introduced into equation (1) due to the resolution limit of voltage and current measurement units used during probing. Error is introduced to the contact resistance by both a voltage measurement resolution limit, ΔV, and a current measurement resolution limit, ΔI. These limits for the equipment used are described in more detail in section 6.2 of this report. Of these two resolution limits, only that of voltage measurements was found to be a significant contributor to resistance measurements, and was

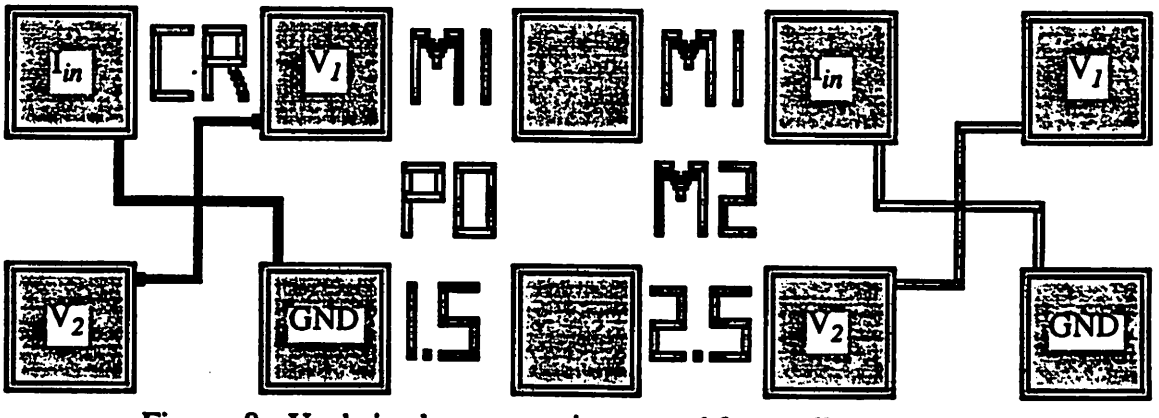


Figure 8 Unchained contact resistors used for small contact sizes.

found to be 0.1mV. The following analysis calculates the measurement error for contact resistance measurements. Using equation (1) as a basis, then:

$$\Delta R_K = \frac{\Delta V}{I}\left|\frac{\partial R_K}{\partial V}\right|, \tag{3}$$

and

$$\Delta R_K = \frac{\Delta V}{I}. \tag{4}$$

As an example, consider contact resistance measurements taken with a test current of I=3mA. Assuming that the resulting voltage measurements were approximately 1V in magnitude, then $\Delta V = (0.01\%)*1V = 0.1mV$. The expected resolution of contact resistance measurements is:

$$\Delta R_K = \frac{0.1mV}{3mA} = 0.033\Omega \quad . \tag{5}$$

Table 2 lists sample data from six die. The columns "R_{Lavg}" and "R_{Ravg}" represent the average of the two left and two right contacts, respectively, in a single n+ diffusion to metal 1cross-contact resistor chain. The columns labeled "die X" and "die Y" contain the x and y integer coordinates of the die within the wafer, and "R_{avg}" is the average of "R_{Lavg}" and "R_{Ravg}".

Table 2 : Sample Measurement Data from Metal 1 to N-Diffusion, 3μm x 3μm Cross-Contact Chains.

die X	die Y	R_{Lavg} (Ω)	R_{Ravg} (Ω)	R_{avg} (Ω)
4	5	36.98	19.59	28.28
1	5	36.97	21.39	29.18
4	7	37.97	27.98	32.98
3	2	37.98	15.89	26.93
6	4	31.98	15.19	23.58
3	4	39.97	18.69	29.33

3.2.2 Split Cross Bridge Resistors

Monitoring sheet resistance and linewidth variation are of great concern in semiconductor manufacturing. Sheet resistance is a direct reflection of the resistivity of interconnect, which can produce undesirable effects on the performance of CMOS circuits due to unwanted voltage drops and RC delay. It is also a direct measure of the doping process, which can effect a great deal of CMOS parameters, from source and drain contact resistance to threshold voltage. Linewidth variation has perhaps an even greater influence on circuit performance because it defines channel length, and therefore current drive capability of CMOS devices. Furthermore, the minimum allowable pitch of interconnect influences the overall size of a fabricated circuit, since it often dictates the area required by the routing channels in a circuit.

A single test structure can be used to measure sheet resistance, linewidth, and line pitch for a particular layer. This structure, the split-cross-bridge-resistor [8]-[10], is shown in Figure 9. The measurements are divided into three distinct sections. The cross part of the structure is used to perform the sheet resistance measurement, R_S, using the very well established van der Pauw relation [7]:

$$R_S = \frac{\pi}{\ln 2}\left(\frac{V_1 - V_2}{I_{R_s}}\right), \tag{6}$$

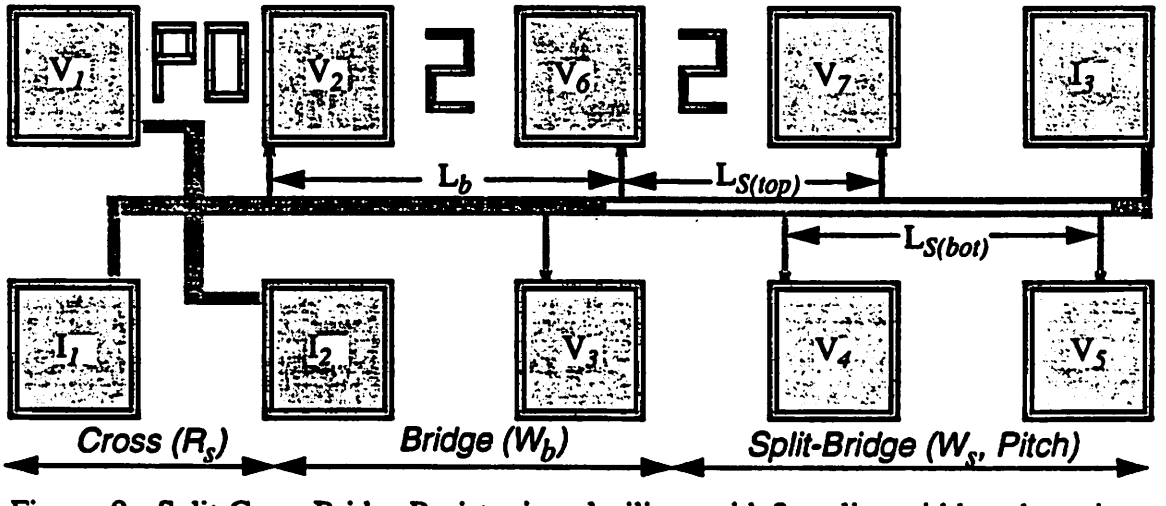


Figure 9 Split-Cross-Bridge Resistor in polysilicon with 2μm line width and spacing.

where I_R is the current flowing from pad I_1 through pad I_2. The middle part of the structure, the bridge resistor, is used to measure linewidth variation. Given the measured value of sheet resistance, the linewidth of the bridge, W_b, is calculated from the relationship:

$$W_b = \frac{R_S L_b I_b}{V_b}, \tag{7}$$

where L_b is the length between voltage pad V_2 and pad V_3 of the bridge resistor, I_b is the current flowing from pad I_1 through pad I_3, and V_b is the voltage (V_3-V_2). Note that the line length will also vary from its designed value, but the variation is negligible as compared to the considerably long drawn length. Finally, the width of the thinner, split-bridge elements are measured in a similar manner:

$$W_{S(top)} = \frac{R_S L_{S(top)} I_b}{2V_{S(top)}}, \tag{8}$$

and

$$W_{S(bot)} = \frac{R_S L_{S(bot)} I_b}{2V_{S(bot)}}, \tag{9}$$

where $W_{S(top)}$, $L_{S(top)}$ and $W_{S(bot)}$, $L_{S(bot)}$ are the widths of the split-bridges and lengths between voltage taps, of the top and bottom split-bridges, respectively. Similarly, $V_{S(top)}$ and $V_{S(bot)}$ are the voltages (V_4-V_5) and (V_6-V_7), respectively. The factor of two in the denominator is based on the assumption that current is divided equally between the top and bottom split-bridges. Finally, the spacing, S, and pitch, P, are calculated from measured data:

$$S = W_b - W_{S(top)} - W_{S(bot)}, \tag{10}$$

and

$$P = S + \frac{W_{S(top)} + W_{S(bot)}}{2}. \tag{11}$$

Obviously, the dimensions of the bridge resistors are of considerable importance when extracting parameters. Table 3 lists the split-cross-bridge resistors designed, along with specifics about the bridge resistor dimensions. Note that those cross-bridges made in diffusion, illustrated in Figure 10, do not contain a split-bridge but rather an elongated middle bridge, since pitch is not a concern for diffusion. The elongated bridge further reduces any error introduced into (7) from variation in line length.

Table 3 : Split-Cross-Bridge Resistors Included in BCAM Design

Layer	W_b (μm)	L_b (μm)	$W_{S(top,bot)}$ (μm)	$L_{S(top)}$ (μm)	$L_{S(bot)}$ (μm)	S (μm)
Polysilicon	6	219	2	204.5	247	2
Metal 1	6	218	2	206	247	2
Metal 2	9	218	3	205	247	3
n+ diffusion	4.5	647.5	N/A	N/A	N/A	N/A
p+ diffusion	2	645	N/A	N/A	N/A	N/A

Labels on the resistors, in order from left to right, refer to layer, split-bridge drawn width in microns, and split-bridge drawn spacing in microns. The drawn bridge width is simply:

$$W_{b(drawn)} = 2*W_{S(drawn)} + S_{(drawn)}, \tag{12}$$

where $W_{S(drawn)}$ is the drawn width of both the top and bottom split bridges.

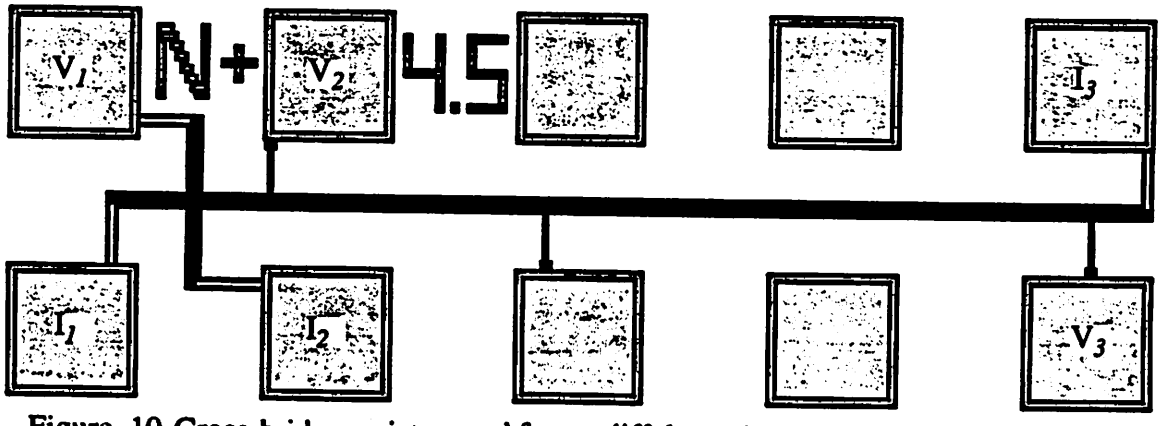


Figure 10 Cross-bridge resistor used for n+ diff. layer. P+ diff. cross-bridge is similar.

For example, the structure shown in Figure 9 is a polysilicon split-cross-bridge resistor with $W_{S(drawn)} = 2\mu m$, $S_{(drawn)} = 2\mu m$, and $W_{B(drawn)} = 6\mu m$. The labels for cross-bridges with no split consist of the layer name first, followed by the bridge width to the right. Therefore, Figure 10 is labeled as a n+ diffusion cross-bridge with drawn width of 4.5μm.

The electrical measurement technique for the cross-bridge and split-cross-bridge resistors follows the analysis described previously. First, the values necessary to calculate R_S are measured by passing a current from pad I_1 to pad I_2, while the voltages at pads V_1 and V_2 are measured. For line width calculations, the current is directed from pad I_1 to pad I_3, and voltages at pads V_2 through V_7 are measured for the split-cross-bridge pictured in Figure 9, or voltages at pads V_2 and V_3 are measured for the cross-bridge pictured in Figure 10.

Measurement error is introduced by both a voltage measurement resolution limit, ΔV, and a current measurement resolution limit, ΔI. Of these two resolution limits, only the voltage measurement was considered to be a significant contributor to resistance measurements. The following analysis calculates the measurement error for the various extracted parameters for the split-cross-bridge resistor. First, consider the sheet resistance measurement, R_S. Starting with the sheet resistance, equation (6), the error is calculated as follows:

$$\Delta R_S = \frac{\Delta V}{I_R}\left|\frac{\partial R_S}{\partial V_1}\right|, \tag{13}$$

and

$$\Delta R_S = \frac{\Delta V}{I_R}\left(\frac{\pi}{\ln 2}\right). \tag{14}$$

For the W_b measurement:

$$\Delta W_b = \Delta V\left|\frac{\partial W_b}{\partial V}\right| + \Delta R_S\left|\frac{\partial W_b}{\partial R_S}\right|, \tag{15}$$

and

$$\Delta W_b = \frac{L_b I_b}{V_b}\left(\frac{R_S}{V_b}\Delta V + \Delta R_S\right). \tag{16}$$

Similarly, for the $W_{S(top)}$ and $W_{S(bot)}$ measurements:

$$\Delta W_{S\,(top,\,bot)} = \frac{L_{S\,(top,\,bot)} I_{S\,(top,\,bot)}}{2V_{S\,(top,\,bot)}}\left(\frac{R_S}{V_{S\,(top,\,bot)}}\Delta V + \Delta R_S\right). \tag{17}$$

The form of this equation differs from that of ΔW_b by the factor of 2 in the denominator, which is due to only half of the bridge current passing through each element of the split-bridge. The error in the measurement of spacing, S, by differentiating equation (10):

$$\Delta S = \Delta W_b\left|\frac{\partial S}{\partial W_b}\right| + \Delta W_{S\,(top)}\left|\frac{\partial S}{\partial W_{S\,(top)}}\right| + \Delta W_{S\,(bot)}\left|\frac{\partial S}{\partial W_{S\,(bot)}}\right| \tag{18}$$

and

$$\Delta S = \Delta W_b + \Delta W_{S\,(top)} + \Delta W_{S\,(bot)} = \Delta W_b + 2\Delta W_{S\,(top,\,bot)}. \tag{19}$$

Finally, for the pitch measurement, P, of equation (11):

$$\Delta P = \Delta S\left|\frac{\partial P}{\partial S}\right| + \Delta W_{S\,(top)}\left|\frac{\partial P}{\partial W_{S\,(top)}}\right| + \Delta W_{S\,(bot)}\left|\frac{\partial P}{\partial W_{S\,(bot)}}\right| \tag{20}$$

and

$$\Delta P = \Delta S + \frac{1}{2}\Delta W_{S\,(top)} + \frac{1}{2}\Delta W_{S\,(bot)} = \Delta S + \Delta W_{S\,(top,\,bot)}. \tag{21}$$

Table 4 lists sample data collected from polysilicon, split-cross-bridge resistors, identical in drawn dimensions to the structure in Figure 9. The data was collected from six separate die, all on the same wafer.

Table 4 : Sample Measured Data from Polysilicon, Split-Cross-Bridge Resistor

die X	die Y	R_S (Ω/□)	W_b drawn (μm)	W_b meas. (μm)	W_s drawn (μm)	W_{sbot} meas. (μm)	W_{stop} meas. (μm)	S (μm)	P (μm)
4	5	17.9	6.0	5.53	2.0	1.80	1.79	1.94	3.74
1	5	19.1	6.0	5.88	2.0	1.87	1.88	2.13	4.01
4	7	18.0	6.0	5.79	2.0	1.86	1:88	2.06	3.92
3	2	19.8	6.0	5.89	2.0	1.87	1.87	2.15	4.02
6	4	18.3	6.0	5.82	2.0	1.85	1.86	2.10	3.96
3	4	19.0	6.0	5.75	2.0	1.85	1.85	2.06	3.91

3.2.3 Fallon Ladder

While cross-bridge resistors are used to measure line width variation, they do so only at a given drawn width. It is also desirable to assess the lithography and etching capability of a process by determining the minimum resolvable line width. Testing this with cross-bridge resistors would require a large number of structures, and therefore considerable die area. M. Fallon and A.J. Walton [11] recently proposed an electrical test structure to determine the minimum resolvable line width that can be resolved by a process, given a relatively small die area. The design of this structure, the Fallon ladder, is based on calculated step changes in resistance that occur when a line is not resolved.

Figure 11 shows the implementation of the Fallon ladders used for the BCAM test chip. Each rung of each ladder is 0.1μm smaller than the previous rung, when traversing the ladder from right to left. Furthermore, the resistance of the rails between each pair of rungs is kept constant by making all rails the same length and width. Resistance measurements are taken on the ladders by passing a current from pad I through pad GND, and measuring the voltage difference between pads V_1 and V_2. The premise of using this ladder structure is that each rung of the ladder, if resolved, will decrease the total resistance

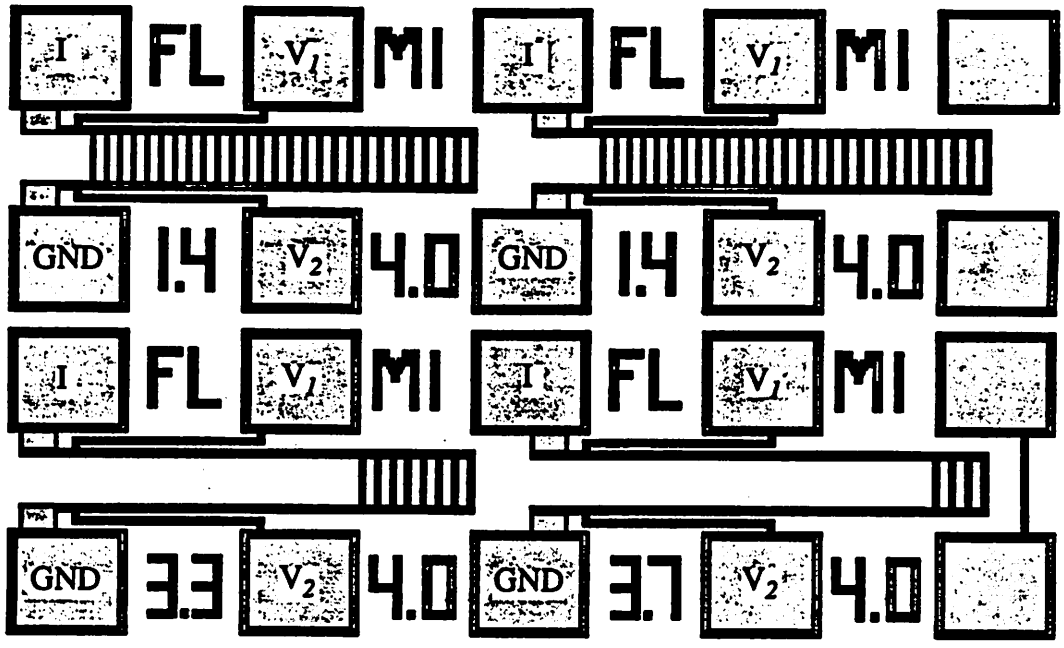


Figure 11 Fallon Ladders, used for determining minimum resolvable line width.

by some incremental amount. A linear function then exists between measured resistance and minimum resolved line width.

The bottom two ladders of Figure 11 are used to calibrate this function. The left and right ladders are drawn with minimum rung widths of 3.3μm and 3.7μm, respectively. Assuming that the process can reliably resolve at least a 3.3μm metal line, resistance measurements on both ladders provide the data necessary to determine the linear function relating minimum line width resolved and resistance measured. Note that the top two ladders have minimum rung widths of 1.4μm, although the process is only expected to reliably resolve 3μm lines. Given the calibrated function, and a resistance measurement on one of the top ladders in Figure 11, the line width of the smallest rung resolved on that ladder can now be determined.

This technique was verified on the 2μm CMOS process at Berkeley, using polysilicon ladders. Rather than the minimum set of ladders, 24 ladders were included, each with a different number of rungs. Starting with a ladder which had a minimum rung width of

2.7μm, each ladder had an additional rung which was 0.1μm narrower than the minimum rung width of the previous ladder. The ladder with the most rungs had a minimum rung width of 0.4μm. A plot of resistance measurements taken on these ladders, versus their minimum drawn rung width, should then show the linearity in the relationship. Figure 12 illustrates the results of the experiment, which show that indeed a linear relationship holds. Note that for very small rung widths, namely 0.4μm and 0.5μm, the resistance no longer drops as with the ladders with larger sized minimum rungs. This shows that 0.6μm was the last line width resolved.

Given the successful verification of the Fallon ladder on our process, the test structure as shown in Figure 11 could now be included on the test die. First, the linear function,

$$lw_{meas} = slope \times R_{meas} + b \quad , \tag{22}$$

is calibrated with the following equations:

$$slope = \frac{R_1 - R_0}{lw1_{drawn} - lw0_{drawn}} \quad , \tag{23}$$

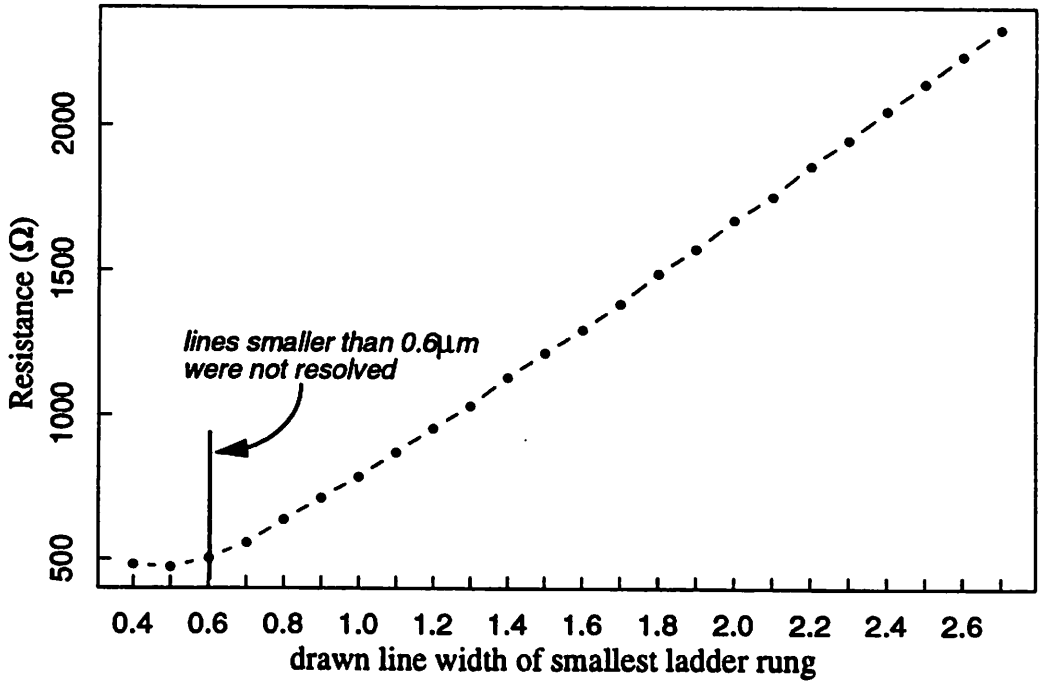


Figure 12 Results from Fallon Ladder experiment showing linear relationship.

and

$$b = R_0 - \text{slope} \times lw0_{drawn} , \tag{24}$$

where R_1 and R_0 are resistances measured from the calibration ladders, with minimum drawn rung widths of $lw1_{drawn}$ and $lw0_{drawn}$, respectively. Now, the ladder complete with all rung widths is measured, yielding the value R_{meas}, which can be used in conjunction with equations (22)-(24) to find the minimum linewidth resolved, lw_{meas}.

Measurement error again results from the limitations in voltage and current measurement. As before, the error due to the current measurement's resolution limit is considered negligible. The expected error for resistance measurements, ΔR, is first calculated by considering the following resistance equation, based on two voltage measurements:

$$R = \frac{V_{1,2}}{I} \tag{25}$$

where

$$V_{1,2} = V_1 - V_2 . \tag{26}$$

Therefore,

$$\Delta R = \frac{\Delta V}{I}\left|\frac{\partial R}{\partial V_{1,2}}\right| , \tag{27}$$

and

$$\Delta R = \frac{\Delta V}{I}(1) = \frac{\Delta V}{I} . \tag{28}$$

Now, considering equations (22)-(24), the error in linewidth measurement, Δlw_{meas}, can be calculated as follows:

$$\Delta\text{slope} = \Delta R\left|\frac{\partial \text{slope}}{\partial R_{1,0}}\right| , \tag{29}$$

where

$$R_{1,0} = R_1 - R_0 . \tag{30}$$

and

$$\Delta\text{slope} = \Delta R\left|\frac{1}{lw1_{drawn} - lw0_{drawn}}\right| . \tag{31}$$

substituting (28) into (31) and simplifying:

$$\Delta slope = \frac{\Delta V}{I\,(lw1_{drawn} - lw0_{drawn})}. \tag{32}$$

A similar analysis yields the expected error for b:

$$\Delta b = \Delta R_0 \left|\frac{\partial b}{\partial R_0}\right| + \Delta slope \left|\frac{\partial b}{\partial slope}\right|, \tag{33}$$

and

$$\Delta b = \Delta R_0 + lw0_{drawn}(\Delta slope). \tag{34}$$

The expected measurement error on minimum rung width resolved can now be calculated from (22) as follows:

$$\Delta lw_{meas} = \Delta slope \left|\frac{\partial lw_{meas}}{\partial slope}\right| + \Delta R_{meas} \left|\frac{\partial lw_{meas}}{\partial R_{meas}}\right| + \Delta b \left|\frac{\partial lw_{meas}}{\partial b}\right| \tag{35}$$

and

$$\Delta lw_{meas} = R_{meas}\Delta slope + slope\Delta R_{meas} + \Delta b. \tag{36}$$

Fallon ladders were designed for polysilicon, metal 1, and metal 2 layers. As shown in Figure 11, the set of test structures for each layer consists of two calibration ladders, occupying one pad set, and two characterization ladders, occupying another pad set. Note that in the calibration pad set there is a metal line extending between the top and bottom, rightmost pads of the set. This line has the width of the most narrow calibration rung. A continuity check should be performed on this line to ensure that the process can resolve the line widths necessary for the calibration. The labels on the top of each pad set describe the test structure, and the layer characterized. The labels along the bottom of the pad sets describe the minimum, and maximum rung widths drawn. For example, in the bottom pad set of Figure 9, the labels "FL", "M1, "3.3" and "4.0" refer to a Fallon Ladder in metal 1, with minimum rung width of 3.3μm and maximum rung width of 4.0μm. Table 5 lists the

results of resolution measurements taken on polysilicon Fallon ladders. The data was collected from six separate die, all on a single wafer.

Table 5 : Sample Measurement Data from Fabricated, Polysilicon Fallon Ladders

die X	die Y	lw0 *drawn* (μm)	R_0 (Ω)	lw1 *drawn* (μm)	R_1 (Ω)	lw2 *meas.* (μm)	R_2 (Ω)	lw3 *meas* (μm)	R_3 (Ω)
4	5	2.3	1841.9	2.7	2231.4	0.8	425.46	0.8	394.86
1	5	2.3	1865.2	2.7	2248.2	0.7	368.46	0.7	417.96
4	7	2.3	1782.8	2.7	2151.9	0.7	338.47	0.7	335.67
3	2	2.3	1924.2	2.7	2329.8	0.7	361.56	0.7	367.26
6	4	2.3	1817.0	2.7	2205.8	0.8	363.36	0.8	358.56
3	4	2.3	1914.9	2.7	2291.4	0.7	449.05	0.7	438.26

3.2.4 Self-aligned n+ Bridges

Another limiting factor in the shrinking of fabrication processes is the misalignment between the various layers of integrated circuits. The accuracy and precision of overlaying these successive patterns is often monitored optically. These structures provide early feedback in the process, but are often costly or time consuming. An electrical test structure has been included in the BCAM test chip to provide a quick, low cost, post-processing assessment of the misalignment between polysilicon and active area.

The structure is based on using the polysilicon gate of a transistor to separate the source and drain regions in a self-aligned process. The structure is designed as two, very wide transistors, with a short polysilicon gate perfectly centered over each active area region, as illustrated in Figure 13. The gate is left unconnected, and serves to create two long, thin resistors per transistor. Depending on the amount and direction of misalignment during fabrication, the resistors will vary in width, and thus in resistance.

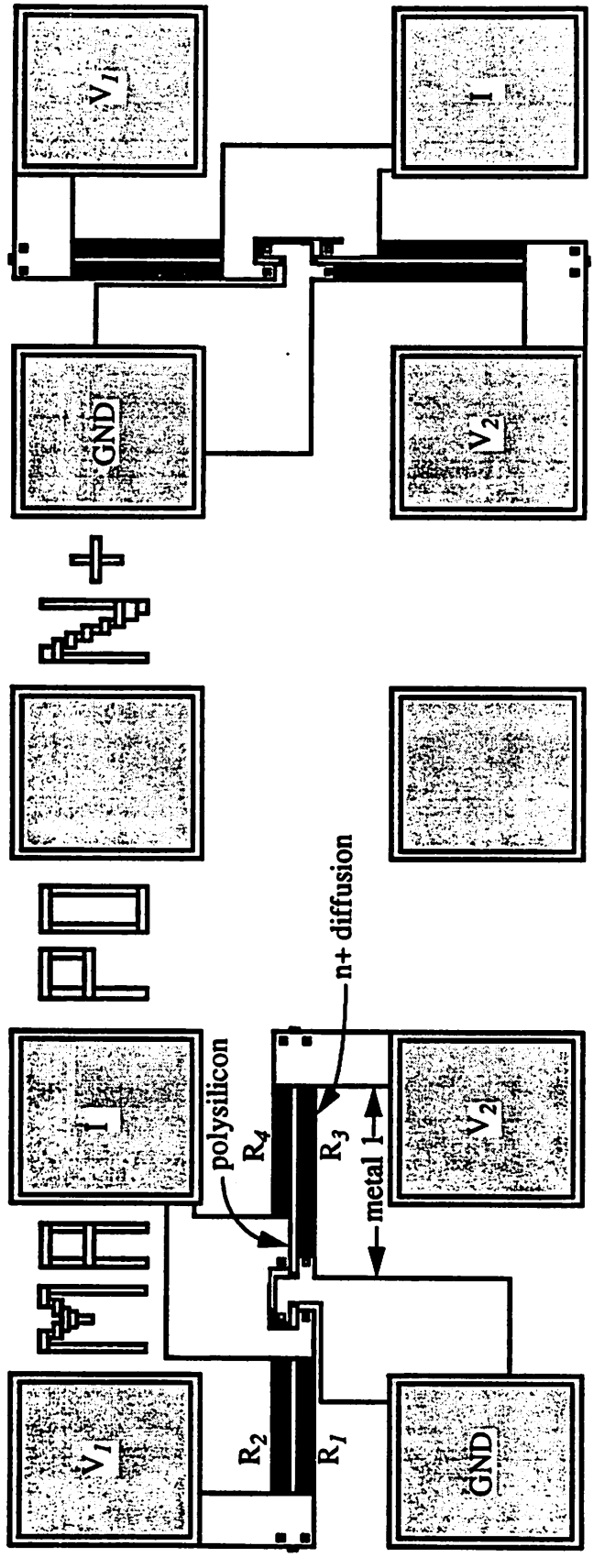


Figure 13 Self aligned n+ bridges to test misalignment between poly and nitride.

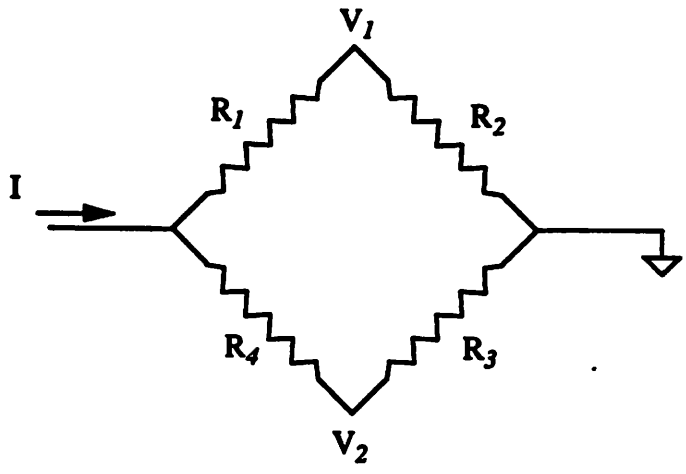


Figure 14 Model of self-aligned n+ bridge.

Four such resistors are connected as a Wheatstone bridge, illustrated in Figure 14, to determine the difference between the two values of resistance. Note that the labels in Figure 14 directly correspond to the sections of the layout referenced in Figure 13 by the identical labels.

In order to understand how the bridge structure works, consider the two devices with resistance labels in Figure 13. Due to the symmetry of the two devices, and the same degree and direction of polysilicon misalignment, resistors R_2 and R_4 will match in value, as will R_1 and R_3. Therefore, a reasonable assumption can be made that the current through each branch of the bridge will be equal. Given that assumption, the following analysis is used to determine misalignment:

$$V_1 = R_2\frac{I}{2} \qquad V_2 = R_3\frac{I}{2} \quad , \tag{37}$$

and

$$R_2 - R_3 = 2\frac{(V_1 - V_2)}{I} \; . \tag{38}$$

We also know that

$$R_2 = R_S\frac{L_2}{W_2} \qquad R_3 = R_S\frac{L_3}{W_3} \quad , \tag{39}$$

where L and W are the length and width, respectively, of the resistor. Combining these

equations results in:

$$R_2 - R_3 = R_S\left(\frac{1}{W_2} - \frac{1}{W_3}\right). \tag{40}$$

Rearranging (40) yields:

$$k = \frac{(R_2 - R_3)}{R_S L} = \frac{(W_3 - W_2)}{W_2 W_3}. \tag{41}$$

We now define $W_2 = W_{drawn} + mX$, and $W_3 = W_{drawn} - mX$, where mX is the misalignment in the upward direction as we look at the structure as shown in Figure 14. Substituting these equation into (41) results in:

$$k = \frac{-2\Delta X}{(W_{drawn} - mX)\,(W_{drawn} + mX)}. \tag{42}$$

Solving (42) for mX yields:

$$mX = \frac{1}{k} \pm \sqrt{\frac{1}{k^2} + W_{drawn}^2}, \tag{43}$$

Finally, combining (38) and (41) results in the following definition for k:

$$k = \frac{2(V_1 - V_2)}{IR_S L}. \tag{44}$$

Equations (43) and (44) can now be used to electrically measure misalignment. Note that (44) is dependent on the value of R_S, which can be extracted from a cross-bridge resistor, as described in section 3.2.2.

Measurement error again results from the limitations in voltage and current measurement. As before, the error due to the current measurement's resolution limit is considered negligible. The expected error for misalignment measurements, ΔmX, is first calculated by recalling from equation (28) the expected error in resistance measurements:

$$\Delta R = \frac{\Delta V}{I} . \tag{45}$$

Recalling from equation (41) that:

$$k = \frac{R_{2,3}}{R_S L} , \tag{46}$$

where

$$R_{2,3} \equiv R_2 - R_3 \tag{47}$$

Then,

$$\Delta k = \Delta R \left| \frac{\partial k}{\partial R_{2,3}} \right| + \Delta R_S \left| \frac{\partial k}{\partial R_S} \right| , \tag{48}$$

and

$$\Delta k = \frac{\Delta R}{R_S L} + \Delta R_S \left(\frac{R_{2,3}}{L R_S^2} \right) . \tag{49}$$

Finally, the misalignment measurement error can be derived from (28) as follows:

$$\Delta mX = \Delta k \left| \frac{\partial mX}{\partial k} \right| \tag{50}$$

and

$$\Delta mX = \Delta k \left| -\frac{1}{k^2} \pm \frac{1}{2} \left(-\frac{2}{k^3} \right) \left(\frac{1}{k^2} + W_{drawn}^2 \right)^{-\frac{1}{2}} \right| . \tag{51}$$

Simplifying (51) results in the following equation for ΔmX:

$$\Delta mX = \Delta k \left| -\frac{1}{k^2} \pm \frac{1}{k^3 \sqrt{\frac{1}{k^2} + W_{drawn}^2}} \right| . \tag{52}$$

Equations (43) and (44) were used to extract the misalignment values in the X and Y direction on six separate die, all on the same wafer. The values for R_S were extracted from cross-bridge resistors, while remaining measurements were taken on the self-aligned n+ bridges. The drawn values of L and W are 128.5μm and 9μm, respectively. Table 6 lists the results of the extraction process from six die on a single wafer.

Table 6 : Sample Extracted Results of Polysilicon-to-N+ Misalignment.

die X	die Y	V_1-V_2 (V)	i_{in} (mA)	R_S	mX (μm)	mY (μm)
4	5	-0.0103	1.00	55.5	0.117	0.094
1	5	-0.0176	1.00	55.5	0.200	0.187
4	7	0.0017	1.00	55.5	-0.019	0.101
3	2	-0.0273	1.00	55.5	0.310	0.166
6	4	-0.0102	1.00	55.5	0.116	0.063
3	4	-0.0196	1.00	56.6	0.218	0.153

3.3 Catastrophic Fault and Reliability Analysis

3.3.1 Contact Chains

Current VLSI processes require the fabrication of a great deal of contacts per die. There exists then a need to monitor the susceptibility of these contacts to random fault and reliability failures, since failure in a single contact can be catastrophic to the circuit functionality. Mitchell, Huang, and Forner [12] state four primary ways in which the electrical continuity of contacts can be interrupted: 1. Contacts are omitted during layout, 2. Contact resistance can become very large due to process variations, 3. Random defects can fall at locations during wafer processing, and 4. Contact discontinuity can occur due to a reliability failure during the operation of a circuit. The first, that of improper layout, is not a processing error, and can be virtually eliminated by the use of CAD verification tools. The next item, contact resistance variation, is of great concern in the fabrication process, and is handled by the analysis of contact resistors as described in section 3.2.1. The remaining issues about catastrophic defects and reliability failures will be addressed here, through the use of contact chains.

Contact chains are simply long serpentines of contacts connected to each other by two alternating layers of interconnect. Figure 15 shows five such contact chains within a single

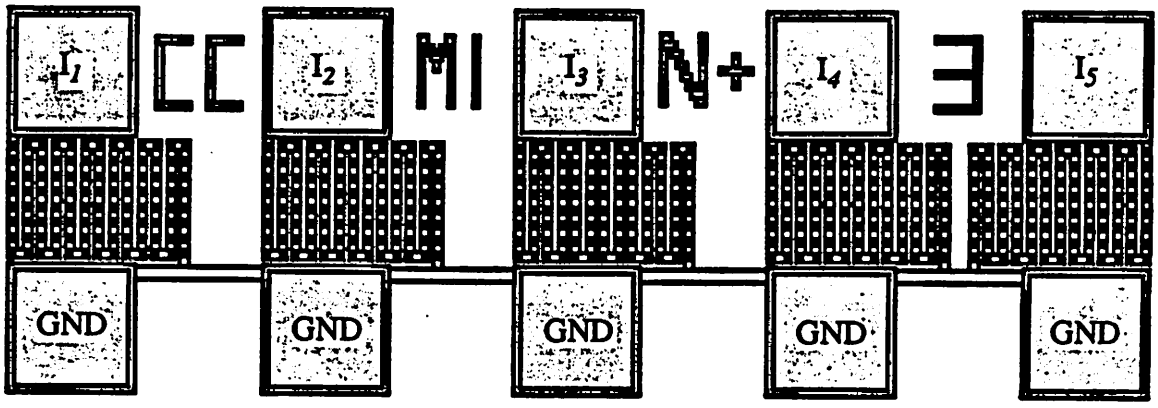


Figure 15 Metal 1 to n+ diffusion contact chains. Five chains are included per pad set.

pad set. Each of the chains consists of 104, 3μm x 3μm contacts between metal 1 and n+ diffusion. Defects are monitored by attaching a current source to one of the five current pads, and measuring the resistance between the source and ground pad. An abnormally high resistance will signal a defect in the chain. Note that metal lines connect all of the ground pads together. A continuity check is first performed on these pads to avoid incorrectly reporting defects when probes are misaligned on the pad set.

Examination of this structure reveals why it is unsuitable for measuring contact resistance variation. Consider the subset of a chain built on p-substrate, which contains two contacts and a strip of metal contacted at each end to two strips of n+ silicon. As current flows through the chain, one of the diffusion links will be at a higher voltage than the other, resulting in a leakage path through both a reverse-biased junction and a forward-biased junction between the two links. This leakage prevents accurate interfacial contact resistance measurements from being extracted from structure [13]. In addition, the resistance of diffusion and poly links can be rather high, making small changes in contact variation resistance undetectable. Finally, even if the resistance of the links was not substantially high, their variation could not be distinguished from contact resistance variation. For these reasons, contact resistors are dedicated to measuring interfacial contact resistance, while contact chains are valuable tools in monitoring contact defects.

Contact chains were designed for both 2μm x 2μm and 3μm x 3μm contacts between metal 1 and polysilicon, metal 1 and p+ diffusion, metal 1 and n+ diffusion, metal 1 and n-well, and metal 1 and metal 2. Note that all chains contain 104 contacts, with the exception of those between metal 1 and n-well, which have 54 contacts. For an example of how these test structures are labelled consider Figure 15, where the labels "CC", "M1", "N+" and "3" refer to the pad set containing contact chains of 3μm x 3μm contacts between metal 1 and n+ diffusion.

3.3.2 Comb Resistors

The lack of uniformity and the impurity of the fabrication process often lead to the introduction of physical faults on a wafer, referred to as spot defects. These defects are regions of either missing or extra material, or material with drastically changed physical characteristics, that may occur in any layer of a fabricated IC [14]. Several methods are available to monitor such defects, including in-situ particle monitors and electrical test structures. In-situ particle monitors have the advantage of short loop feedback for process control. Post-processing testing, however, alleviates the cost of having a dedicated in-situ particle monitor. A specially designed resistor structure, the comb resistor, is used to electrically monitor the density of spot defects which cause intralayer shorts in metal and polysilicon lines.

Figure 16 shows the layout of five, individually probed comb structures in a single pad set. Defects are monitored by attaching a current source to one of the five current pads, and measuring the current flowing into the ground pad. Measuring any appreciable current in the ground pad signifies a short circuit, and therefore the presence of a spot defect. Note that metal lines connect all of the ground pads together. A continuity check is first performed on these pads to ensure proper alignment before testing. Reliability analysis may also be performed on this structure, by stressing the structure with high humidity, temperature and voltage.

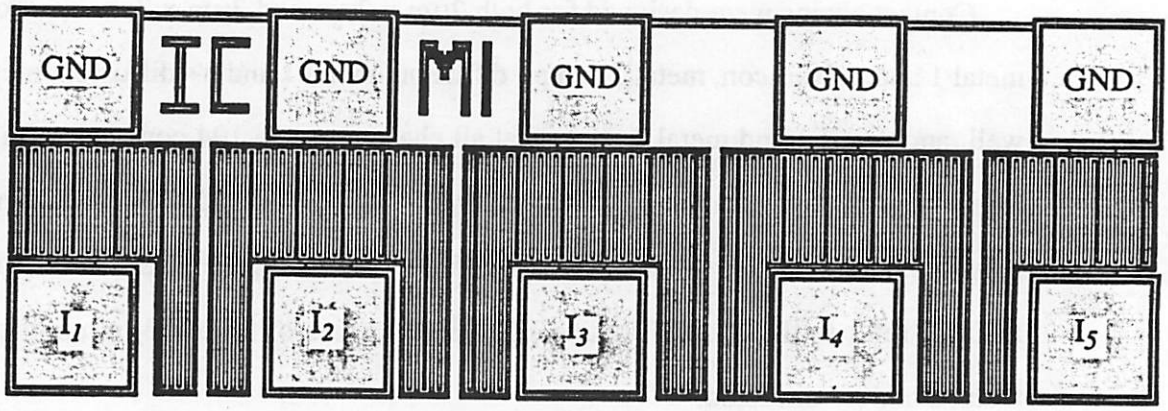


Figure 16 Comb resistor used to monitor spot defects.

Combs were designed in metal 1, metal 2 and polysilicon. The spacing between metal lines, and the width of the lines themselves are both 3μm. The spacing and width for polysilicon resistor combs are both 2μm. These spacings were chosen in order to evaluate defect sizes equal to or greater than the design rules for the process. Finally, the labels simply identify the structure as an interdigitated comb in a particular layer. For example, Figure 16 is labeled with "IC" and "M1", which corresponds to an interdigitated comb in the metal 1 layer.

3.3.3 Serpentine/Comb Resistors

Another type of spot defect involves missing material in a particular layer. Generally, this results in broken lines, which will more than likely result in a loss of functionality for the fabricated circuit. A simple structure is often used in process monitoring to evaluate the occurrence of such defects. The structure is a long serpentine of wire in the layer being characterized. The serpentine's resistance is measured, and an abnormally high resistance is interpreted as a break in the metal line. A serpentine/comb structure is simply a combination of a serpentine resistor and a comb resistor, which can be used to assess both opens and shorts in various layers.

Figure 17 illustrates the layout of a serpentine/comb structure. Defects which create broken lines are monitored by attaching a current source to pad S_1, and measuring the resistance between pads S_1 and S_2. An abnormally high resistance will signal a break, or defect, in the serpentine. Defects which create short circuits are monitored by attaching a current source to the pad labeled S_1 while leaving S_2 unconnected, and grounding pads C_1 and C_3. Measuring any appreciable current in C_1 or C_3 signifies a short circuit, and therefore the presence of a spot defect. Note that polysilicon lines connect pads C_2 and C_3 together. A continuity check is first performed on these pads to ensure proper alignment before testing.

Note that this defect monitor can measure both shorts and opens, while dedicated combs and serpentines measure only a short or an open, respectively. It then appears that the serpentine/comb combination is a preferable structure due to better area utilization. While the combination does detect both types of faults, the serpentine/comb structure requires at least four pads, while a dedicated comb or serpentine requires only two. Given that the pad set being used is a 2 x 5 set of pads, this results in only two detectable defects per pad set when using a serpentine/comb combination, while the dedicated combs and dedicated serpentines can detect up to five defects per pad set. Since the expected defect density of our fabrication process is presently undetermined, both structures have been

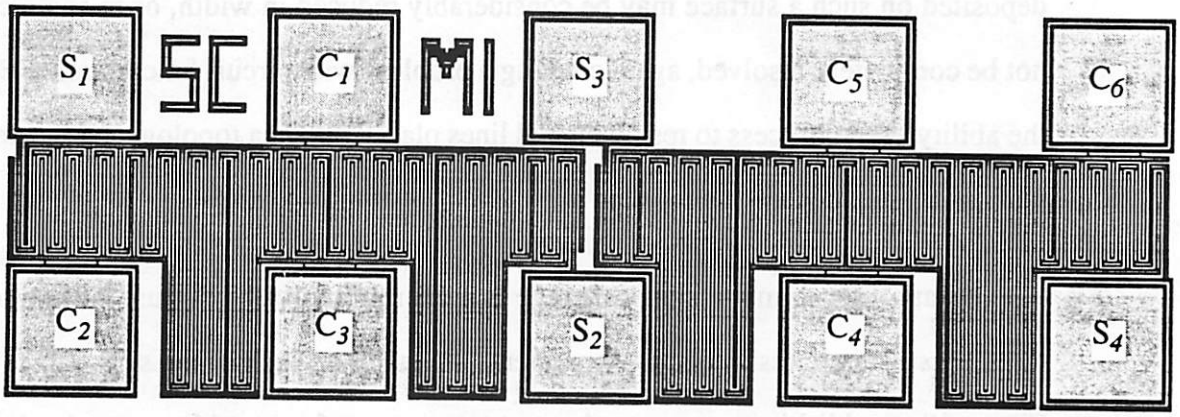


Figure 17 Serpentine/Comb structure for defect monitoring.

included in the BCAM design. If future studies determine that defect density and clustering analyses can be performed to a satisfactory degree with only two defects detectable per pad set, then serpentine/combs can be used exclusively to minimize the total die area required.

Serpentine/comb structures were designed in metal 1, metal 2 and polysilicon. The spacing between metal lines, and the width of the lines themselves are both 3μm. The spacing and width for polysilicon resistor combs is 2μm. These spacings were chosen in order to evaluate defects sizes equal to or greater than the design rules. Finally, the labels simply identify the structure as a serpentine/comb in a particular layer. For example, Figure 17 is labeled with "SC" and "M1", which corresponds to a serpentine/comb in the metal 1 layer.

3.3.4 Serpentines Over Topography

While serpentines are used to detect spot defects, they may also be used to evaluate metal step coverage. In some cases, metal lines laid over a flat surface may be resolved to an acceptable degree, but may not be acceptable when placed over topology. For example, oxide grown over a relatively large area of substrate should have a reasonably level topology. However, oxide grown over a series of polysilicon lines will develop uneven steps as the oxide conforms to the polysilicon lines which rise above the substrate. Metal lines deposited on such a surface may be considerably reduced in width, or in some cases may not be completely resolved, again creating a problem with circuit functionality. Evaluating the ability of the process to resolve metal lines placed above a topology can be determined with metal step coverage analysis.

The analysis of metal step coverage is performed through the use of two test structures. First, simple resistance measurements are taken on each of the serpentines shown in Figure 18, establishing an expected average resistance for metal lines resolved over a flat

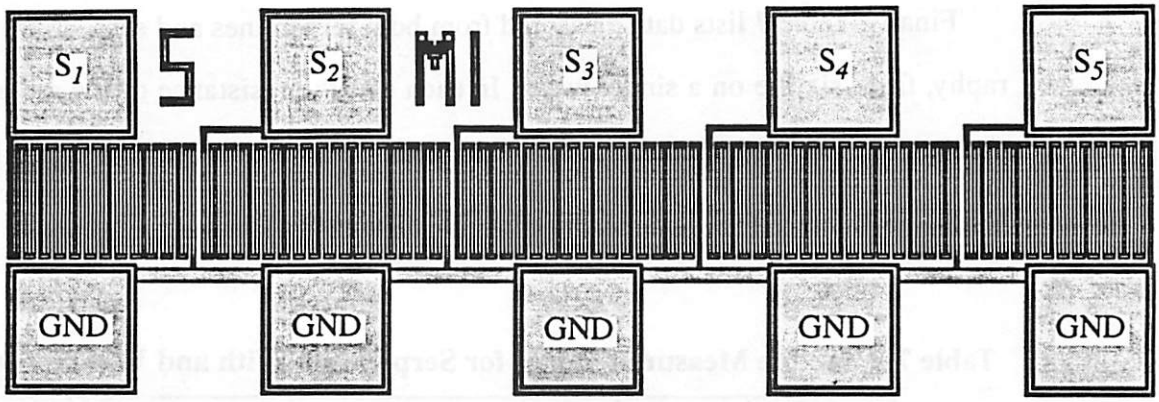


Figure 18 Metal 1 serpentine with no topography, used for metal step coverage analysis

topology. This resistance is evaluated against the same measurements taken on metal serpentines laid over a topology of many polysilicon lines, such as those shown in Figure 19. The serpentine structure is the same as that shown in Figure 18, with horizontal polysilicon lines creating the topology.

Polysilicon lines were deleted for clarity in Figure 19. The actual layout contains 23 polysilicon lines placed 2µm apart, each with a width of 2µm. The labeling is consistent with the labeling of other defect monitors. For example, the labels "S" and "M1" in Figure 18 refer to a serpentine of metal 1, while the additional label "PO" in Figure 19 refers to the metal 1 serpentine being placed over polysilicon lines.

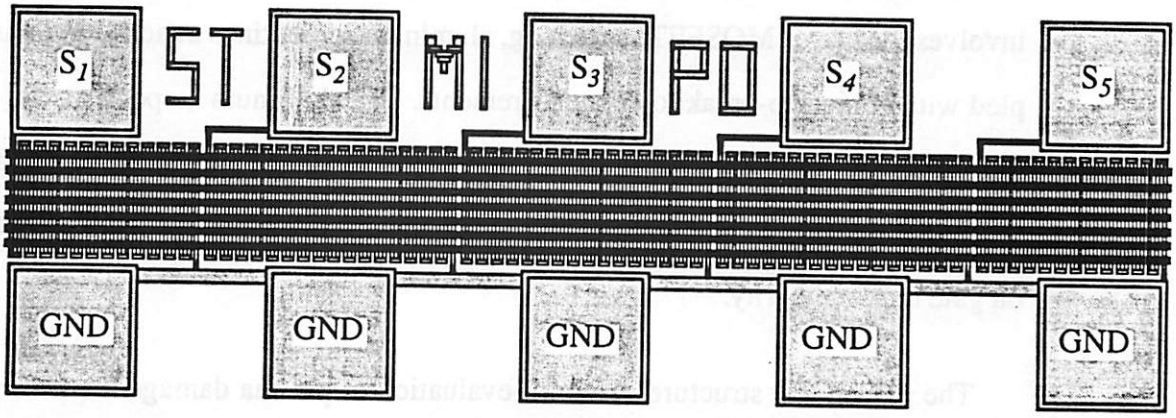


Figure 19 Metal 1 serpentine over topography.

Finally, Table 7 lists data measured from both serpentines and serpentines over topography, from six die on a single wafer. In each die, the resistance of the metal serpentine nearly doubled, with the exception of the final die in the table, in which a break in the metal line is illustrated by the extremely high resistance value.

Table 7 : Sample Measured Values for Serpentines With and Without Topography

dieX	dieY	R_{avg} (no topography) (Ω/□)	R_{avg} (with topography) (Ω/□)
4	5	0.078	0.141
1	5	0.073	0.124
4	7	0.073	0.132
3	2	0.080	0.140
6	4	0.079	0.151
3	4	0.077	2.52E+26

3.3.5 MOSFET With Antenna

The use of plasma etching has gained widespread use in the semiconductor industry. During the plasma etching process, significant charge can accumulate on the aluminum lines connected to polysilicon gates, and on the gates themselves. As a result of this charge, the gate oxide of the devices are damaged. Y. Uraoka, K. Eriguchi, T. Tamaki and K. Tsuji propose a quantitative evaluation method of plasma damage [15]. This method involves the use of MOSFETs with long, aluminum serpentines attached to their gate, coupled with charge-to-breakdown measurements. The aluminum serpentines act as antenna in picking up ions from the plasma. This accumulated charge stresses the gate. Later, charge to breakdown measurements evaluate the effect of this in-process electrical stress on gate oxide integrity.

The pair of test structures used for evaluation of plasma damage are shown in Figure 20. The device on the left is a transistor with a width of 20μm, and a gate length of 2μm.

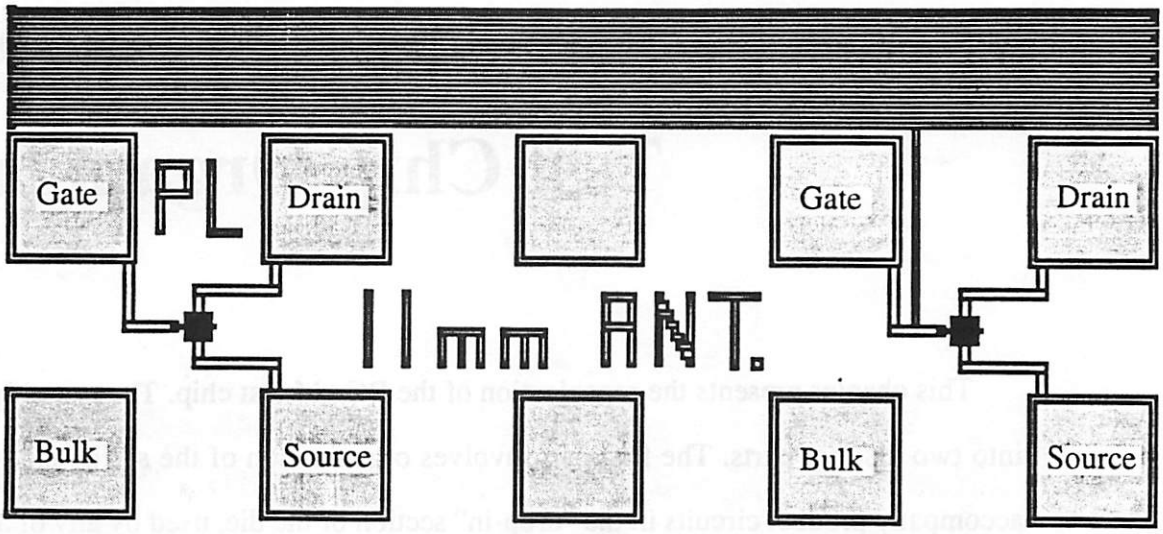


Figure 20 MOSFETs used for evaluation of gate oxide damage due to plasma process.

The device on the right is of the same size, but has an 11mm long serpentine antenna, attached to its gate. Charge to breakdown measurements are performed on both devices, and can be compared to evaluate the effect of the plasma process on the device with the antenna.

The details of charge to breakdown measurements are left for future study. Extensive testing of similar devices are presented in [15], which also provides a comprehensive evaluation method of plasma damage. The method includes photo emission analysis to detect the breakdown spot in gate oxide, which was found to occur at LOCOS edge. The devices were included here to provide preliminary results for a more comprehensive study.

Chapter 4

Test Chip Organization

This chapter presents the organization of the BCAM test chip. The test chip is divided into two distinct parts. The first part involves organization of the scribe lane which will accompany product circuits in the "drop-in" section of the die, used by any of a number of research groups using the Berkeley Microfabrication Laboratory. The scribe lane contains both structures used to characterize, monitor and debug the process, as well as tools such as alignment marks required for fabrication of the die. Aside from the scribe lane, an additional, BCAM "drop-in" die is fabricated for a more comprehensive study of the stepper field.

Test structures fabricated during this baseline run, along with scribe-lane test structures manufactured alongside product circuits, will be measured and the resulting data will be statistically analyzed. Analysis results will be used to determine whether or not the process is in control, and if not what particular part of the process needs attention.

The remainder of this chapter outlines the organization of the wafer and stepper field, and includes a description of how the scribe lane is created within the field. Additionally, the test structure subsets used for the scribe lane and drop-in area are reported, including the locations of the structures within the field.

4.1 Scribe Lane

The implementation of the scribe lane is illustrated in Figure 21. The horizontal and vertical lines represent the scribe lanes, which are the areas that are cut through during physical separation of the die for individual packaging. The scribe lanes can be used for characterization test structures prior to die separation, without a loss of area for the product circuit in the drop-in area.

Figure 22 shows how the scribe lane is created with respect to the stepper field, including dimensions of the total printed area and drop-in area. Area is used near the perimeter on the left and top sides of the field, with the drop-in completing the square. These squares are abutted in all directions to form the printed area across the entire wafer. The remainder of this section describes the structures placed in the shaded area which forms the scribe lane.

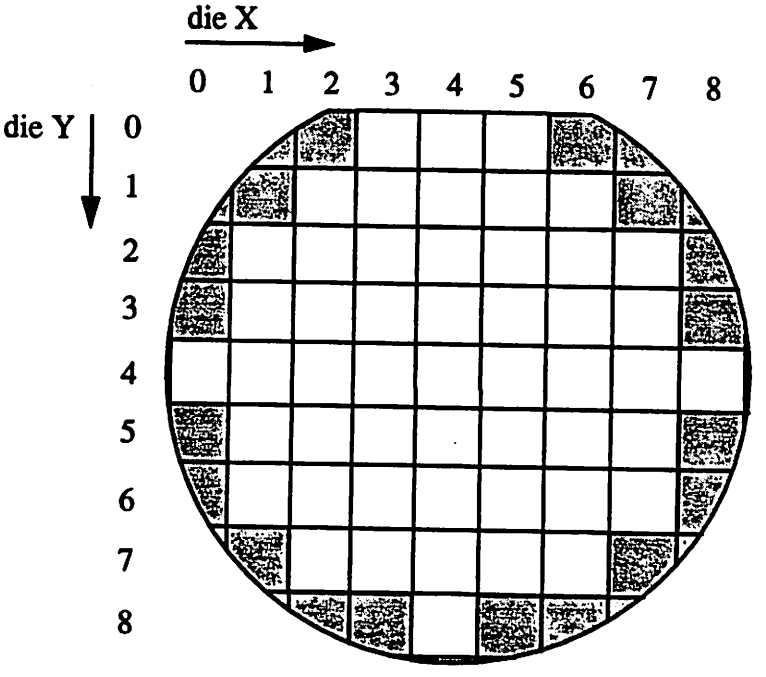


Figure 21 Die configuration and labelling of a 4 inch wafer used in the baseline process.

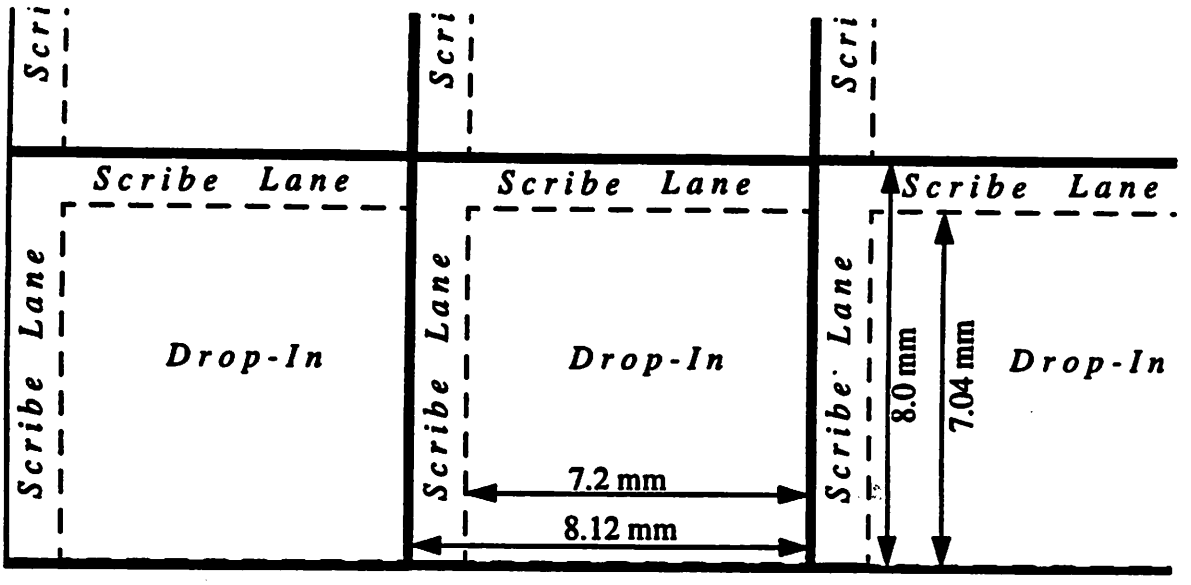


Figure 22 Configuration of scribe lane and drop-in die within the stepper field.

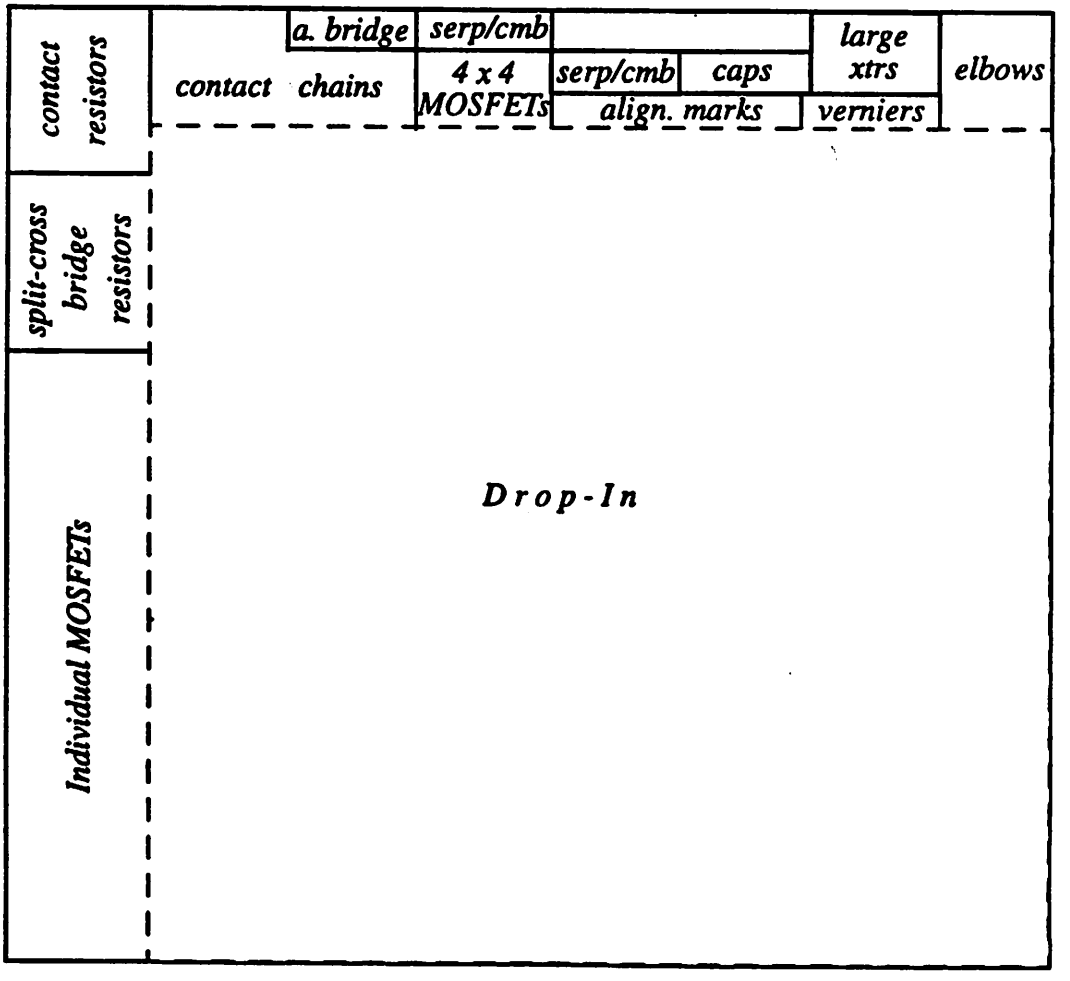


Figure 23 Arrangement of test structures within the scribe lane.

The configuration of structures in the scribe lane is shown in Figure 23. Two types of structures exist on the scribe lane. The first set of structures includes those which are necessary in order to fabricate the die, while the second set includes those used for process characterization and debugging. The former set includes marks for mask alignment, verniers to further assist in the alignment process, and elbows in various layers for optical inspection of linewidth resolution.

Of greater concern here is the set test structures used for device and process characterization and for process debugging, which in the scribe lane represents a subset of those structures described in Chapter 3. Device characterization is accomplished through the use of the individually probed MOSFETs described in section 3.1.1. The transistor set consists of devices with gate lengths 1, 1.3, 1.5, 2, 3, 5, 10 and 25 μm. For each length listed, both an NMOS and a PMOS device of width 5, 10 and 50 μm exists. Each pad set can be used to individually probe three devices, each with the same gate length but a different width, resulting in a total of 16 pad sets. These MOSFETs were placed in the lower left hand area of the scribe lane. The 300μm x 300μm capacitors mentioned in section 3.1.3 were included for characterization of thin oxide, and were placed in the upper right-hand section of the scribe lane. Finally, toward the center of the top section of the scribe, 4x4 MOSFET arrays are also included for device characterization.

The next set of test structures included in the scribe involve process characterization. The split-cross-bridge resistors described in section 3.2.2 are included to characterize sheet resistance, line width variation, and pitch. They are located just above the individually probed MOSFETs on the vertical part of the scribe. Just above those resistors are contact resistors, used to monitor contact resistance variation. These cross-contact chains have 3μm x 3μm contacts between metal 1 and polysilicon, p+ and n+ diffusion, and metal 2. The final test structure used for process characterization is the self-aligned n+

bridge, labeled "a.bridge" in Figure 23, which is used to extract the misalignment between polysilicon and active area layers.

Finally, test structures included in the scribe lane for random fault and reliability analysis include contact chains and serpentine/comb resistors, described in sections 3.3.1 and 3.3.3, respectively. These structures are located left of center, on the top portion of the scribe. The contact chains contain 3μm x 3μm contacts between metal 1 and polysilicon, p+ and n+ diffusion, nwell, and metal 2, while the two serpentine/comb structures included are those made of polysilicon and metal 1.

4.2 Drop-In Die

The entire BCAM test chip includes both the scribe lane and a drop-in die, which provides for a more comprehensive study of the stepper field. The drop-in die includes replications of all the test structures described in Chapter 3, and its layout is illustrated in Figure 24. This figure shows the instance of each test structure, replicated several times over the die. The smallest instance shown corresponds to the approximate size of a single pad set, while larger boxes are drawn to scale and contain a proportionate amount of pad sets. For example, the metal 2 Fallon ladder instance shown in the upper left corner is twice the size of the serpentine shown immediately to its right. The Fallon ladder instance can then be expected to have two pad sets, which indeed corresponds to the Fallon ladder combination shown in Figure 11.

The primary objective of the layout was to provide adequate coverage of the die in order to measure spatial variation of parameters across the stepper field. Since defect monitors are not affected by location within the field, they were placed along the bottom and right sides of the drop-in (the scribe lane already occupies the left and top sides of the field.) These defect monitors include serpentines and serpentine/comb combinations in metal 1, metal 2 and polysilicon, positioned as shown in Figure 24.

The remainder of the die is partitioned into three sections, each with the same test structures. The sections are two pad sets wide, and run vertically through the drop-in area. The first section corresponds to the shaded area of Figure 24. The remaining two sections are identical in size, and located immediately to the right of the first section. The structure placement within each section was different, in order to provide a comprehensive coverage of the die.

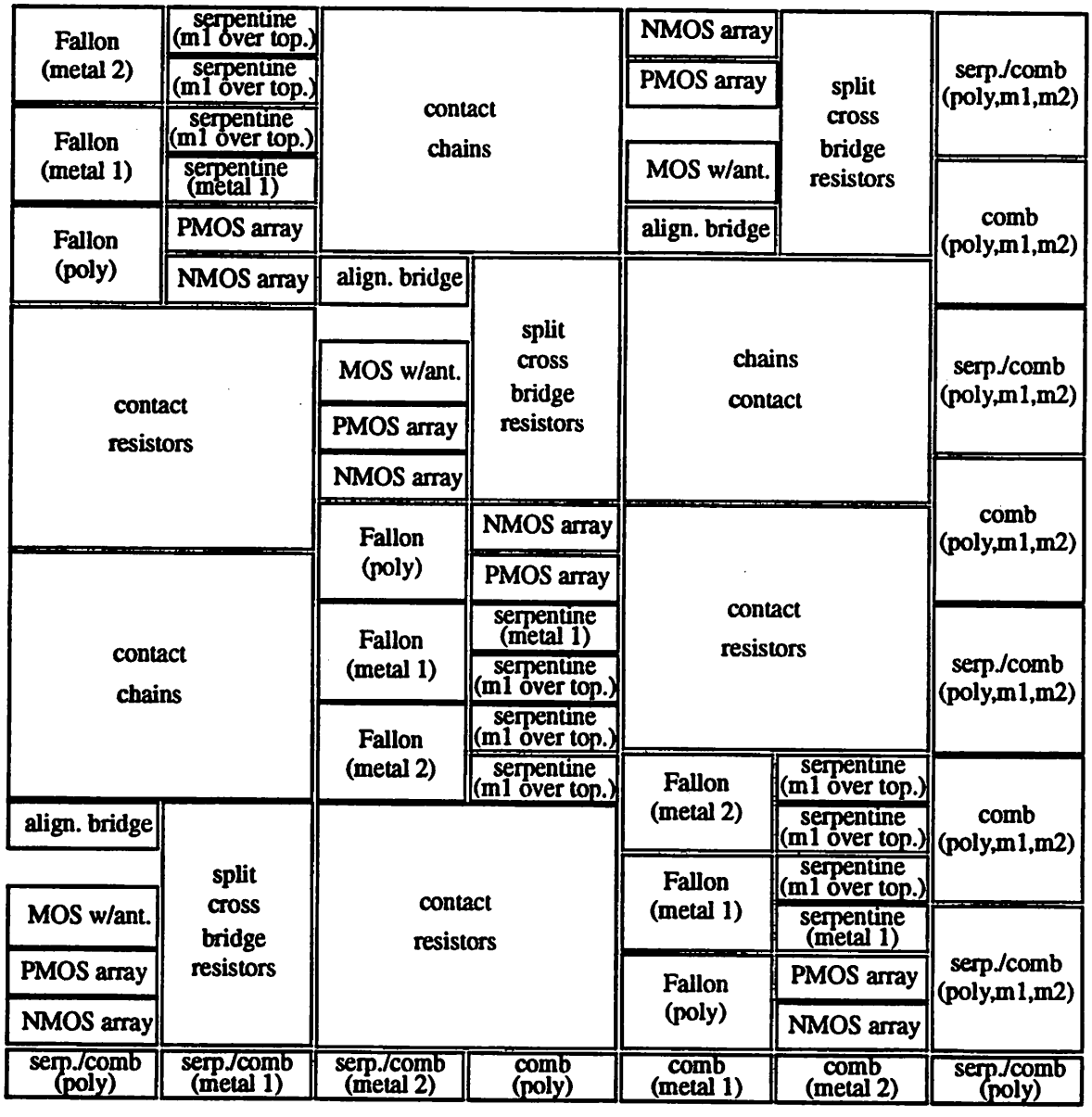


Figure 24 Configuration of test structures within the drop-in area.

4.3 Complete Test Chip

While sections 4.1 and 4.2 described the organization of test structures within the scribe lane and drop in areas, this section provides an overview of the test chip fabricated with both areas included. This test chip provides complete coverage of the stepper field, which is particularly useful in analyzing intra-die variation.

The summary of the available test structures, and the parameters and layers that they characterize, can be seen by reviewing Table 8 through Table 10. These tables include references to the detailed descriptions available in Chapter 3.

Table 8 : Test Structures for Device Characterization

Test Structure	Parameters	Device Type and Size	Ref. Section
Individual MOSFETs	SPICE Parameters/ Inter-die variation	L = 1, 1.3, 1.5, 2, 3, 5, 10, 25 μm W = 5, 10, 50 μm (both PMOS & NMOS)	3.1.1
4x4 MOSFET arrays	Intra-die variation	1 NMOS array (W/L = 20μm/2μm) 1 PMOS array (W/L = 20μm/2μm)	3.1.2
300 μm x 300 μm capacitors	t_{ox}	gate oxide	3.1.3

Table 9 : Test Structures for Process Characterization

Test Structure	Parameters	Layers Characterized	Ref. Section
4 terminal contact resistors	Contact Resistance & Misalignment	M1-M2, M1-poly, M1-n+, M1-p+ (Contact sizes: 1.5, 2, 3)	3.2.1
Split-cross-bridge resistors	R_S, linewidth, line-spacing, line-pitch	M1, M2, poly, n+, p+	3.2.2
Fallon Ladder	Minimum linewidth determination	poly, M1, M2	3.2.3
Self aligned poly-n+ bridge	Misalignment	poly-diffusion	3.2.4

Table 10 : Test Structures for Catastrophic Faults and Reliability Analysis

Test Structure	Parameters	Layers Characterized	Ref. Section
contact chains	Contact Defects	M1-n+,M1-p+,M-nwell,M1-poly,M1-M2 (Contact sizes: 2x2μm and 3x3μm)	3.3.1
comb structures	Defect Monitoring (shorts only)	poly, M1, M2	3.3.2
serpentine/comb resistors	Defect Monitoring (shorts & opens)	poly, M1, M2	3.3.3
serpentines over topography	Metal Step Coverage	M1, M1 over poly	3.3.4
Capacitors/MOSFETS (from Table 8)	Dielectric Breakdown	gate oxide	3.1
MOSFET w/"antenna"	Gate Oxide Damage from Plasma Process	gate oxide	3.3.5

Figure 25 shows the test structures within the entire scribe line and drop in area, including the details within the test structure instances in Figure 24. Special note should be taken regarding pad set placement. Pad sets were placed on a grid pattern for ease of probing, with a spacing of 20μm between pad sets. Each 2x5 pad set has a horizontal dimension of 900μm, and a vertical dimension of 300μm. Therefore, moving one pad set in a horizontal direction requires a step of 920μm, and moving one pad set in a vertical direction requires a 320μm step. The only exception here involves the MOSFET with antenna, since the antenna uses area above the pad set. In order to maintain the grid spacing, a structure placed above a MOSFET with antenna is spaced such that a vertical jump between the two structures is twice the normal step, or 640μm. Figure 25 shows the distance required to move to each test structure, relative to the shaded structure in the lower left corner of the die. This structure is referred to as the "home device" for probing purposes, and indicates the location where the probes should be placed at the outset of probing. Details concerning probing will be further discussed in Chapter 5.

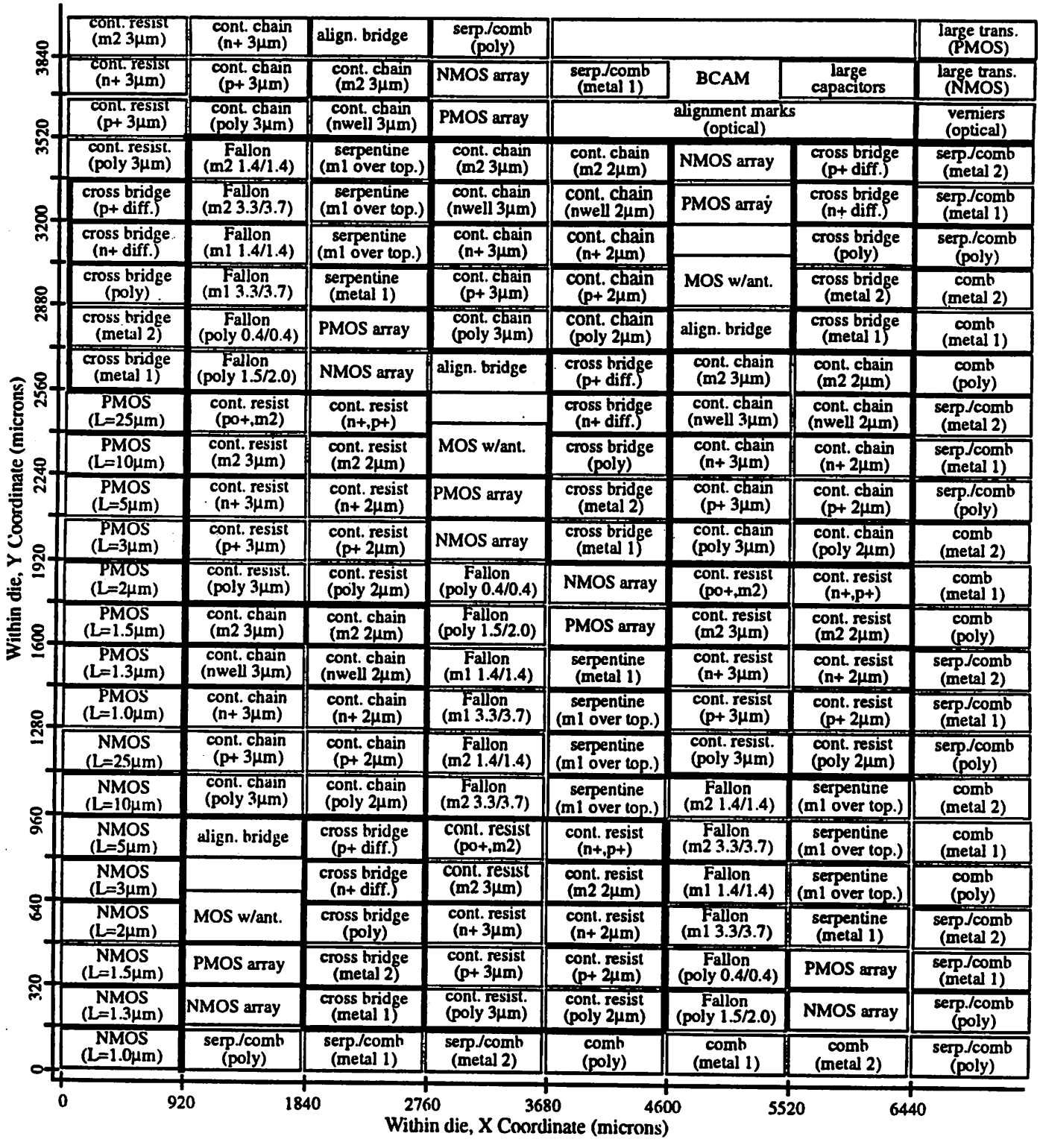


Figure 25 Complete scribe lane and drop-in area, showing location of structures, in microns.

Chapter 5

Automated Testing System

5.1 Introduction

In order to provide an efficient means of collecting large amounts of data for monitoring the baseline process, an automated testing system was developed in conjunction with the BCAM test chip. This system, referred to herein as the autoprober, provides a means of operating probing hardware from a Unix workstation, through a software interface. The user may simply utilize a set of existing measurement subroutines by configuring two text files, or may add additional subroutines to the current library. Each subroutine is designed to perform a specific set of measurements.

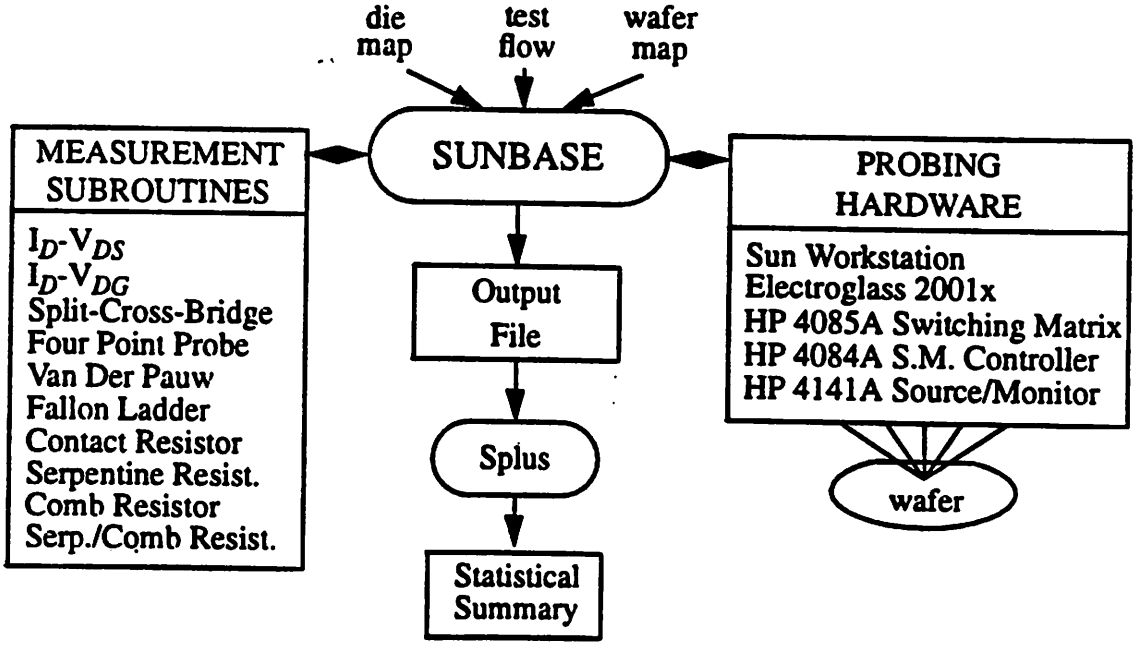


Figure 26 Hardware and software configuration of autoprobing system.

A diagram of the autoprobing system is shown in Figure 26. The shell of the autoprober, a program called Sunbase, was developed by Vadim Gutnik of the Berkeley Microfabrication Laboratory. This shell serves as an interpreter of the text files, which direct the movement of the x-y wafer stage and define the measurement routines to be used. The measurement routines define the appropriate voltage and current sources and monitors to be attached to the probes by the switching matrix, collect the results, and perform parameter extraction by applying the analyses described in Chapter 3. The subroutines then output the data to a text file, which includes the name of the structure, position of the die, position of the structure within the die, and measurement and extracted results. An example of this text file will be illustrated in section 5.3. Communication between the probing hardware and the Unix workstation is directed by Sunbase.

5.2 Using Sunbase

This section provides an overview of the steps required to use Sunbase for measuring test structures. A detailed explanation of the automated testing system, including specifics on using the hardware, adding subroutines, and understanding the Sunbase code, can be found in Chapter 8.18 of the Microfabrication Laboratory Manual [16]. Only excerpts of this chapter are included in this section. The complete chapter is included as Appendix I of this thesis.

Specifying a set of measurements on a wafer is performed by using two user-defined text files, die.map and prober.text. The file die.map contains specifics about the test structures on the die being used, while prober.text is used to specify the tests required by the user. Details concerning these files, and a sample Sunbase run will be presented in the remainder of this chapter. Sunbase should be run from the directory "~eglas" on the machine "lead", an Argon client in the Device Characterization Laboratory. The files "die.map" and "prober.text" must also be placed in the "~eglas" directory. These files are described in the following two sections.

5.2.1 "die.map"

The file "die.map" contains specifics about the test structures on the die being probed, including the name of the test structure, its location within the die, and the configuration of the pads used to probe it. The format of a test structure description in "die.map" is as follows:

```
Struct_name x,y terminal1 ... terminalx parameter1 ... parameterx
```

Struct_name is a simple, unique, alphanumeric name given to the structure by the user, while x and y correspond to the horizontal and vertical coordinates, respectively, of the structure within the die. The x and y coordinates are with respect to the origin x_0=0 and y_0=0. Prior to initiating Sunbase, the user must place the probes on the structure defined with x=0 and y=0 coordinates, herein referred to as the home device. The suggested home device is the pad set in the scribe lane containing individually probed transistors of gate lengths of 1μm. This pad set is in the lower, left corner of the scribe lane, as illustrated by the shaded device in Figure 25. Using this structure as the home device results in positive x and y coordinates for all test structures. All coordinates in "die.map" and the output files are listed in microns. The items labeled pad1, pad2, and so on, communicate to the measurement subroutine which pads correspond to particular terminals of the test structure. Finally, parameters such as designed lengths and widths or layer names can also be passed to the measurement subroutines. As an example, consider the generic test structure description of the split-cross-bridge resistor:

```
scbr1 x,y i1 i2 i3 v1 v2 v3 v4 v5 v6 v7 Lb Wb LS(top) LS(bot) WS layer_nm
```

Where the variables i1 through v7 refer to the labels of the pads in Figure 9, and L_b through W_s refer to the dimensions of the split-cross-bridge in microns, as listed in Table

4. Pad numbers are used to identify which pad in the 2 x 5 pad array is being referenced. Pads are numbered from 1 to 10, starting with the upper leftmost pad in the array and proceeding clockwise, as illustrated in Figure 1. With this fact in mind and referring to both Figure 9 and Table 4, a polysilicon split-cross-bridge resistor at location x=320μm and y=640μm, with respect to the home device, would be defined in the file die.map as follows:

```
scbrPO 320,640 10 9 5 1 2 8 7 6 3 4 219.0 6.0 204.5 247.0 2.0 poly
```

The software measurement routine SCBR, to be discussed later, will parse this line and use the information appropriately. The generic formats for test structure descriptions currently programmed into Sunbase are as follows:

```
mosfetN x,y drain gate source bulk
4ptprb x,y Iin gnd v1 v2
conrM1po3 x,y Iin gnd
fallonPO1 x,y Iin gnd v1 v2 min._rung_width
scbr1 x,y i1 i2 i3 v1 v2 v3 v4 v5 v6 v7 L_b W_b L_s(top) L_s(bot) W_s layer_nm
serpM1 x,y
cchainPO2 x,y
```

The pad names listed above correspond directly to those shown in the test structure layouts in Chapter 3. Note that the names used above are examples. It is suggested that the names be descriptive, such as the device name conrM1po3 indicating the third metal 1 to polysilicon contact resistor structure on the die.

Finally, lines beginning with an asterisk and blank lines are ignored by Sunbase. Furthermore, "die.map" must contain the line "@home 0,0", which serves to send the probes back to the origin of the die when probing of each die is complete. A complete example of a "die.map" file is included later, in section 5.3.

5.2.2 "prober.text"

The file "prober.text" is used to specify the various die to be probed on the wafer, and the specific measurements to be taken on those die. The format of "prober.text" is as follows:

```
0 0 0 0 0 0 0 0 0
0 0 1 1 1 1 0 0 0
0 1 1 1 1 1 1 0 0
1 1 1 1 1 1 1 1 0
1 1 1 1 1 1 1 1 0
1 1 1 1 x 1 1 1 0
1 1 1 1 1 1 1 1 0
0 1 1 1 1 1 1 0 0
0 0 1 1 1 1 0 0 0


Routine_name
structure_name1
structure_name2
.
```

The 9x9 array of ones and zeros above represent the mapping of die on the wafer, with a "1" indicating that the die is to be measured, and a "0" indicating that it is not to be measured. The "x" indicates the first die probed, and the die on which the probes must initially be placed. Note also that the probes should be placed on the user defined home device specified in "die.map". The "x" should be placed somewhere near the center of the wafer to alleviate probe to wafer misalignment errors. The above example would probe all the die on a wafer, but by replacing the ones with zeros the array can be changed to measure only several, or even a single die. The line "Routine_name", chosen by the user from the existing set of routines names listed in Table 11, defines the measurement routine to be used. The routine will take measurements on the test structures named in die map as "structure_name1" and "structure_name2". The period after the structure names indicates the end of the parameter list being sent to the measurement routine. There is no limit on

the amount of devices which can appear in the parameter list, nor is there a limit to the number or order of measurement routines included in the "prober.text".

Section 6.1 discusses the measurement routines currently included in the autoprober system. As a simple example of how a structure measurement can be defined in "prober.text", consider the following lin:

```
SCBR
scbrPO1
scbrn+2
```

These lines define a split-cross-bridge measurement (SCBR) to be performed on polysilicon and n+ diffusion split-cross-bridge resistors (scbrPO1 and scbrn+2), which must be defined in the file "die.map". Assuming that the wafer map illustrated at the beginning of this section is used, the system will first probe the die marked with an "x", and then will step through the five die marked with 1's. On each die, the same polysilicon and n+ diffusion cross-bridges will be probed.

5.2.3 Measurement Subroutines

Table 11 lists the measurement routines currently available for the autoprober. The table includes the name of the measurement, the case-sensitive name of the routine to be used in "prober.text", and the output, which always appears in table form. An explanation of these routines, and the parameters required by them, follow in this section.

The first two routines shown in Table 11 provide I-V characteristics for MOSFETs. Along with the structure name, these routines require additional parameters in order to define the sweep. These values, VGSstart, VGSstop, VGSstep, VDSstart, VDSstop, and VDSstep, are passed along with the device name, as shown in this example:

Table 11 : Currently Available Measurement Subroutines for Autoprober.

Measurement	Routine name	Author	Output
I_d-V_{ds} curve	IdVds	V. Gutnik	Id-Vds characteristic
I_d-V_g curve	IdVg	V. Gutnik	I_d-V_g characteristic
Four Point Probe	4ptprb	V. Gutnik	Resistance
Van Der Pauw	VDP	V. Gutnik	Sheet resistance (R_S)
Split-Cross-Bridge	SCBR	D. Rodriguez	R_S, ΔW, Spacing, Pitch
Fallon Ladder	Fallon	D. Rodriguez	Ladder resistance, min. linewidth resolved
Contact Resistance	Conr	D. Rodriguez	Contact resistances (left, right, avg.)
Comb Defect	Comb	D. Rodriguez	5 Binary result showing shorts (defects)
Serpentine Resistance	Serp	D. Rodriguez	Resistances for 5 serpentines in pad set
Serpentine/Comb Defect	SerpComb	D. Rodriguez	2 Binary result showing opens/shorts (defects)

```
IdVds
+VDSstart=0.1
+VDSstop=2
mosfet1
.
```

As is the case with all of these subroutines, the output will appear in tab delimited, table format.

The four point probe routine simply applies a current between two terminals of a structure, measures the voltage at another two terminals, and outputs the resistance. The van der Pauw routine performs the same measurement, but applies the van der Pauw resistance

factor $\frac{\pi}{\ln 2}$ shown in equation (6), in order to output the sheet resistance R_S. The only argument necessary for these subroutines is the name of the structure as defined in "die.map".

The split-cross-bridge subroutine performs the measurements described in section 3.2.2, and outputs the sheet resistance, drawn line widths, extracted variation of line widths and extracted spacing and pitch. This subroutine requires only structure names, as defined in "die.map".

The Fallon ladder subroutine also requires only test structure names, but requires four per measurement, in a particular order. Recalling from section 3.2.3, the process of extracting minimum line width resolved involves first calibrating the measurement with two Fallon ladders. These two devices must be listed first, followed by the two ladders to be characterized. For example, the following example measures two calibration ladders first, followed by two ladders from which the minimum line width resolved will be extracted:

```
Fallon
fallonPOcal1
fallonPOcal2
fallonPO1
fallonPO2
.
```

The resulting output will list the resistances of all four ladders, accompanied by the minimum line width resolved, calculated by the equations (22)-(24) in section 3.2.3.

The contact resistance subroutine passes a current through the cross contact chain shown in Figure 7, and measures the voltages at each contact. The output lists the average resistance values for the two left and two right contacts, along with the average of all contacts. Again, the only parameter required by the subroutine is the name of the contact resistors, as defined in "die.map".

The subroutine used for measuring shorts between comb structures follows the procedure described in 3.3.2. Defects are monitored by attaching a current source to one of the five current pads, and measuring the current flowing into the ground pad. Measuring any appreciable current in the ground pad signifies a short circuit, and therefore the presence of a spot defect. The subroutine outputs a 1 if a short was found, and a 0 otherwise. This is repeated for all five combs of the pad set, so the output shows five binary values. The only parameter required by the subroutine is the name of the combs, as defined in "die.map".

The serpentine subroutine is for measuring resistance of serpentines, of serpentines over topography, and of contact chains. In all cases, the routine measures the resistance of each of five structures in a pad set, and outputs the values accordingly. The only parameters required by the subroutine are the names of the test structures.

The final routine available extracts defect information from serpentine/comb structures. The first test checks the continuity of the serpentine, and outputs the result as a "1" if open circuited, or a "0" otherwise. Similarly, another column in the table lists a "1" if a short occurred between a comb and the nearby serpentine, or a "0" otherwise. Since two serpentine/comb structures are contained in each pad set, this data is repeated twice per measurement. Once again, the only parameter required by the subroutine is the name of the serpentine/comb.

Although these are the only routines currently available with the autoprober, additional routines can be written if desired. The subroutines are written in "C" language, with subroutine calls to Sunbase providing control of the autoprober. The process of adding subroutines to the system is described in detail in Chapter 8.18 of the Microfabrication Laboratory Manual [16], which is included in Appendix I of this thesis. The code for measurement subroutines named in this chapter have been included in Appendix II of this thesis.

5.3 Sample Run

This section outlines a simple run of the autoprober, so that the entire process of using the autoprober can be viewed. For this example, we shall take measurements on a split-cross-bridge resistor and a polysilicon Fallon Ladder. The entire contents of the file "die.map" is as follows:

```
@home 0,0
m1 0,0 1 2 9 5
scbrPO 2760,640 10 9 5 1 2 8 7 6 3 4 219.0 6.0 204.5 247.0 2.0 poly
scbrn+ 0,0 10 9 5 1 2 6 8 6 8 6 647.5 4.5 429.0 429.0 2.25 n+
fallonPOcal1    920,1920 1 10 2 9 2.3
fallonPOcal2    920,1920 3 8 4 7 2.7
fallonPO1       920,2240 1 10 2 9 0.4
fallonPO2       920,2240 3 8 4 7 0.4
```

Now, the following "prober.text" file will probe the above structures on six die, placed as shown:

```
0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0
0 0 0 1 0 0 0 0 0
0 0 0 0 0 0 0 0 0
0 0 0 1 0 0 1 0 0
0 1 0 0 x 0 0 0 0
0 0 0 0 0 0 0 0 0
0 0 0 0 1 0 0 0 0
0 0 0 0 0 0 0 0 0

SCBR
scbrPO
scbrn+
.

Fallon
fallonPOcal1
fallonPOcal2
fallonPO1
fallonPO2
.
```

The probes should now be placed on the home device, on the die corresponding to the one marked with an "x" above. In this example, the home device is the n+ diffusion split-cross bridge, named "scbrn+" in "die.map". Sunbase can be executed by simply typing "sun-

base" from the Unix shell prompt. When finished, a file named "output.text" will contain the results. The data in the output file is in an intermediate format listed in the order that structures were probed, and can now be sorted into tables according the types of measurements taken by running a script on "output.text", creating the file "final.out". The script is run by typing "postproc output.text" at the Unix shell prompt.

The file "final.out" now contains the following:

dieX	dieY	name	Rs	WbDrawn	DeltaWb	WsDrawn	DelWsbot	DelWstop	S	P
4	5	scbrPO	17.86	6	0.47	2	0.205	0.206	1.945	3.739
1	5	scbrPO	19.13	6	0.116	2	0.132	0.117	2.133	4.009
4	7	scbrPO	18.00	6	0.211	2	0.144	0.124	2.058	3.924
3	2	scbrPO	19.61	6	0.105	2	0.128	0.130	2.152	4.024
6	4	scbrPO	18.34	6	0.182	2	0.147	0.138	2.103	3.960
3	4	scbrPO	19.01	6	0.247	2	0.154	0.153	2.060	3.907

dieX	dieY	name	lw0(d)	R0	lw1(d)	R1	lw2-(d)	lw2(m)	R2	lw3(d)	lw3(m)	R3
4	5	fallonPO1	2.3	1841	2.7	2231	0.4	.8	425.5	0.4	.8	394.9
1	5	fallonPO2	2.3	1865	2.7	2248	0.4	.7	368.5	0.4	.8	418.0
4	7	fallonPO2	2.3	1782	2.7	2152	0.4	.7	338.5	0.4	.7	336.0
3	2	fallonPO2	2.3	1924	2.7	2330	0.4	.8	361.6	0.4	.8	367.3
6	4	fallonPO2	2.3	1817	2.7	2206	0.4	.8	363.4	0.4	.8	358.6
3	4	fallonPO2	2.3	1914	2.7	2291	0.4	.7	449.1	0.4	.7	438.3

The table columns are tab delimited, which is useful for importing the tables into various statistical packages for further analysis. The column labels for the results correspond to the characterization parameters described in Chapter 3. For example, the split-cross-bridge results table list columns for WsDrawn, DelWsbot, and DelWstop. Referring to section 3.2.2 reveals that these values refer to the drawn width of the split-bridge and the variation of the bottom and top split-bridge resistors, respectively.

Again, this section was intended to provide only an overview of the steps required to use Sunbase for measuring test structures. A detailed explanation of the automated testing

system, including specifics on using the hardware, adding subroutines, and understanding the Sunbase code, can be found in Chapter 8.18 of the Microfabrication Laboratory Manual [16]. This reference is included as Appendix I of this report.

Chapter 6

Sample Results and Analyses

6.1 Introduction

As illustrated in Figure 26, the final use of the set of test structures presented in this report is to produce a statistical summary. Although further study by potential users will dictate the type of statistical summary required, some examples are presented in this chapter. In particular, data extracted from scribe lane test structures will illustrate how they can be used to provide information about the process and about the measurement techniques used.

The remainder of this chapter describes the statistical analyses performed. A resolution analysis for voltage and current measurements has already been presented in Chapter 3. Here we use a repeatability test of sheet resistance in order to also estimate the standard deviation of voltage measurements. Additionally, scatter plots and wafer contour maps are used to understand the correlation between sheet resistance and linewidth variation on a split-cross-bridge resistor.

6.2 Resolution Tests

In performing electrical measurements using the autoprober, some degree of round-off measurement error can be expected due to resolution limits of the voltage and current measurement units. Repeatability tests were performed in order to also estimate the standard deviation of the measurement.

The tests were performed by measuring a polysilicon split-cross-bridge resistor 100 times with the same input current values, and recording the measured voltage and current values from the cross-bridge.

The resolution of voltage measurements is $\Delta V=0.1mV$. However, our experiments show that the standard deviation, σ_V, is not constant. A dependence was found to exist between the standard deviation of the measurement, and the value being measured. As the voltage being measured increased, the voltage measurements' standard deviation became worse, as is listed in Table 12. The data for Table 12 was collected by probing a bridge resistor 100 times, given the same input current for each measurement. Each time voltage measurements were performed at the various positions along the bridge resistor, thus yielding a range of voltages each time the bridge was probed. The "Voltage Measured"

Table 12 : Percent Error as a Function of Voltage Measured, 100 Replications

Average Voltage Measured (V)	$\hat{\sigma}$ as % of the average
0.4368	0.0090
0.5955	0.0099
0.8712	0.0111
0.9680	0.0100
1.2739	0.0100
1.3834	0.0073

column in Table 12 then represents the mean of values measured for a particular point along the bridge, while the "$\hat{\sigma}$" column represents the estimated standard deviation divided by the average. As an example, the trend plot for the voltages at a point along the bridge is shown in Figure 27, and has a mean of 0.4368 and a $\hat{\sigma}$ of 0.0090%. These values were listed in the first row of Table 12. The percent error was relatively constant for the

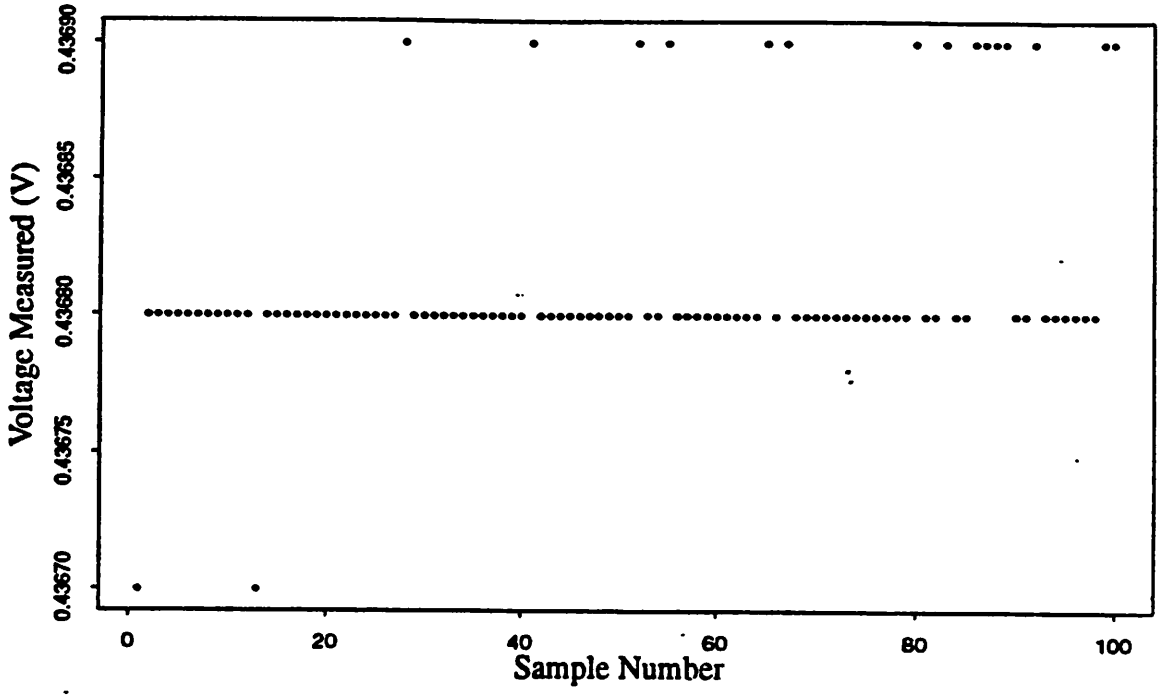


Figure 27 Trend plot showing voltage resolution for a measured voltage of 0.43680 V.

range of voltages measured, and remained at about 0.01% of the voltage measured. Therefore, this is the assumed percent error for all probing of voltages in the 0 to 1.5V range.

Recall from Chapter 3 that error analyses were performed for characterization routines. These error analyses can be verified experimentally, as the following example for R_S measurement error illustrates. A test current I_R of 8.0mA was used for the resistance measurement on a polysilicon split-cross-bridge resistor, which resulted in a voltage measurements of approximately 1.38V. We will first calculate the resolution for sheet resistance measurements, ΔR_S, from equation (14) given the voltage measurement resolution of ΔV=0.1mV:

$$\Delta R_S = \frac{\Delta V}{I_R}\left(\frac{\pi}{\ln 2}\right) = \frac{0.1\text{mV}}{8.0\text{mA}}\left(\frac{\pi}{\ln 2}\right) = 0.057 \ \Omega/\square. \tag{53}$$

We can also estimate the standard deviation of sheet resistance measurements, σ_{Rs}. Given that the percent error of voltage measurements is 0.01%, the expected error is calculated as follows:

$$\hat{\sigma}_v = 0.01\% * 1.38V = 0.138mV, \tag{54}$$

and

$$\hat{\sigma}_{Rs} = \frac{\sigma V}{I_R}\left(\frac{\pi}{\ln 2}\right) = \frac{0.138mV}{8.003mA}\left(\frac{\pi}{\ln 2}\right) = 0.078\ \Omega/\square. \tag{55}$$

These values were verified by performing repeatability tests on the cross part of a polysilicon split-cross-bridge resistor, with same test current of I_R = 8.0mA, used in the calculations above. The same structure was probed 300 times, resulting in the sheet resistance values plotted in Figure 28. The average of these sheet resistance values is 17.01Ω/□ with a $\hat{\sigma}_{Rs}$ of 0.05. Given these values, and the values in Figure 28, the predicted ΔR_S of 0.06Ω/□ and $\hat{\sigma}_{Rs}$ of 0.078Ω/□ seem to provide reasonable estimates of expected measurement error, therefore verifying the calculations of equations (14) and (55).

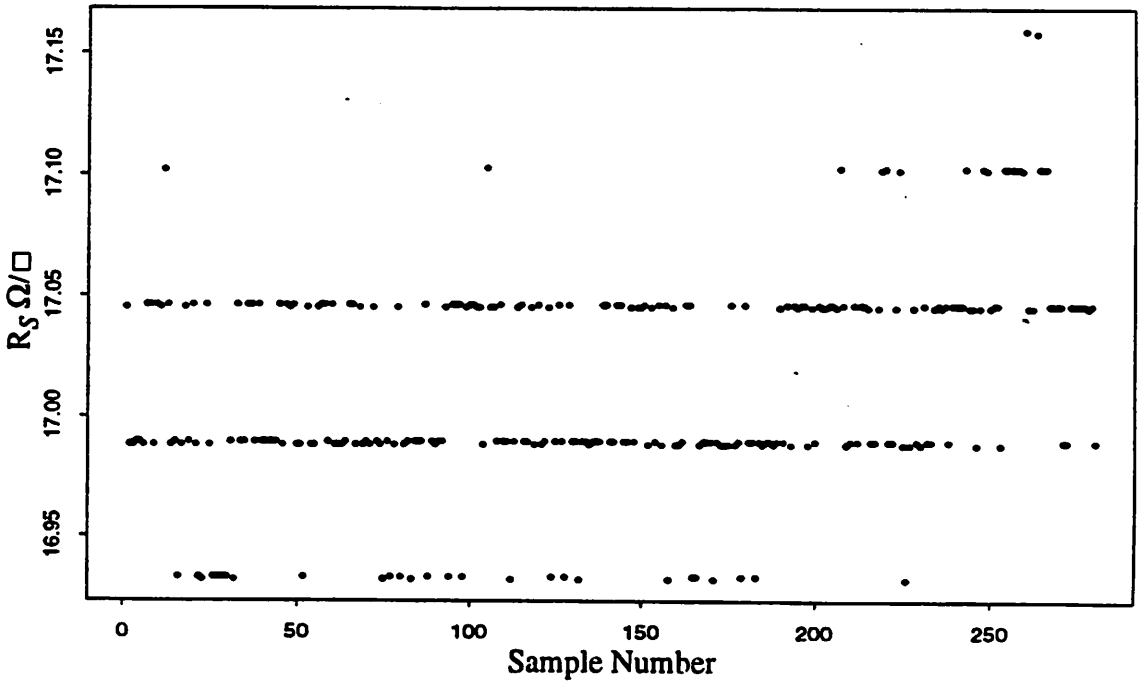


Figure 28 Trend plot showing 300 measured R_S results a single polysilicon cross-bridge

6.3 Analysis of the Split-Cross-Bridge Resistor

The autoprober was used to characterize the sheet resistance and linewidth variation of polysilicon lines. Data was extracted from the scribe lanes of two wafers from the same lot. A polysilicon split-cross-bridge resistor was probed on each of 52 die for each wafer. The polysilicon split-cross-bridge resistors used are identical to those illustrated in Figure 9, and described in Table 3. The data from the autoprober was imported into a statistical software analysis package, S-plus, and analyzed.

An unexpected correlation was found to exist between the sheet resistance and linewidth variation for wafer 1 of the lot. A scatter plot of the two parameters is illustrated in Figure 29, which shows that a moderate correlation exists. The correlation coefficient was calculated to be -0.63, a significant value for 52 samples.

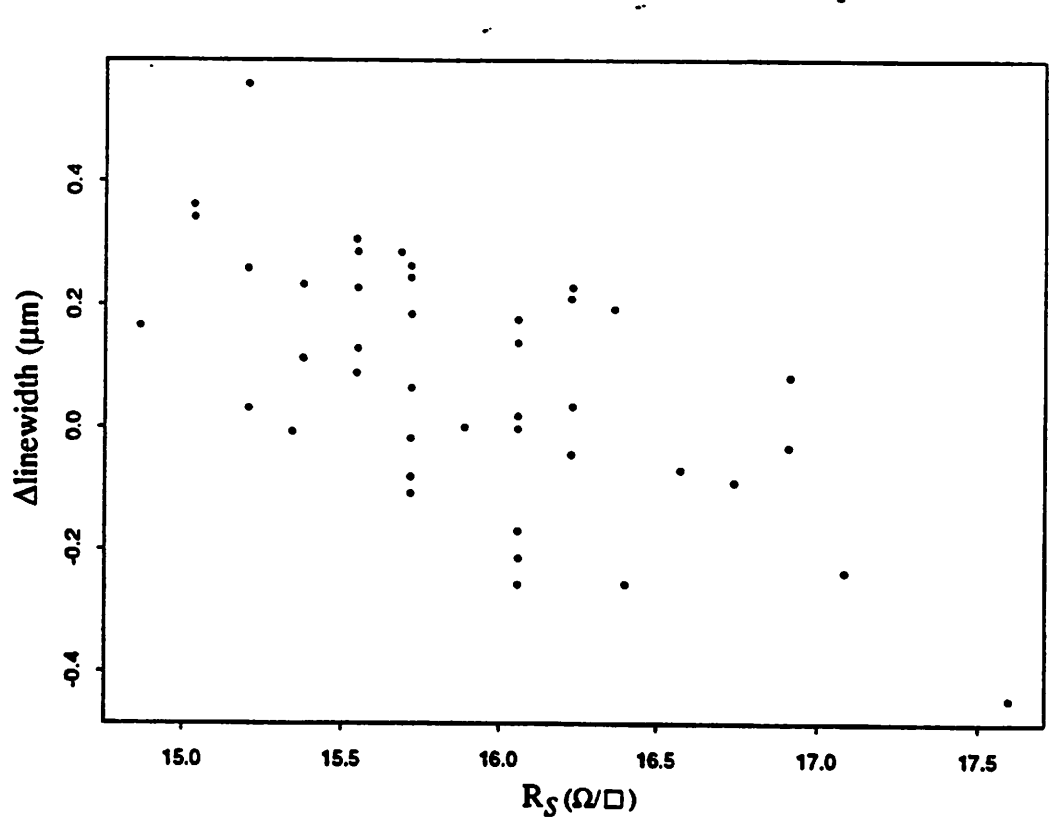


Figure 29 Scatter plot of Δlinewidth versus R_S for wafer 1. (ρ =-0.63)

In order to gain greater insight into this correlation, wafer contour maps were created for the two parameters, as illustrated in Figure 30 and Figure 31. Points marked by the symbol "⊗" denote measurements which did not produce results upon probing, indicating a catastrophic failure for that structure. Note that the correlation is also evident from these maps, since each has similar contours. It is also evident from these maps that some processing error occurred during fabrication of the wafer, since there is a region near the right edge of the wafer where the sheet resistance and linewidth change significantly. The sheet resistance, for example, is approximately 16 Ω/□ throughout most of the wafer, and drops down to 14 Ω/□ in a region constituting a relatively small area of the wafer. Similarly, the linewidth variation is highest at this point. Since the linewidth variation is defined as the measured minus the drawn width, this indicates wider lines in the region of lower sheet resistance.

A likely contributor to this correlation is the thickness of the polysilicon lines, as increasing the thickness of polysilicon line reduces its sheet resistance, while also increasing the fabricated linewidth above the drawn linewidth. Consider the illustration in Figure 32, which is an exaggerated example of increased polysilicon thickness on two lines of the same drawn width. Since the slope of the line edges are considered independent of the poly thickness, the thicker line 2 will result in a larger measured linewidth than for line 1. Furthermore, the thicker line results in a lower current density for current passing through line 2, and therefore a lower sheet resistance.

The same wafer contour maps were created for a second wafer of the same lot, and are illustrated in Figure 33 and Figure 34. Note that changes in sheet resistance are not as significant as those from wafer 1. The correlation between R_S and Δlinewidth dropped from 0.63 to 0.44, indicating that there is indeed a contribution from the process to the correlation between these parameters.

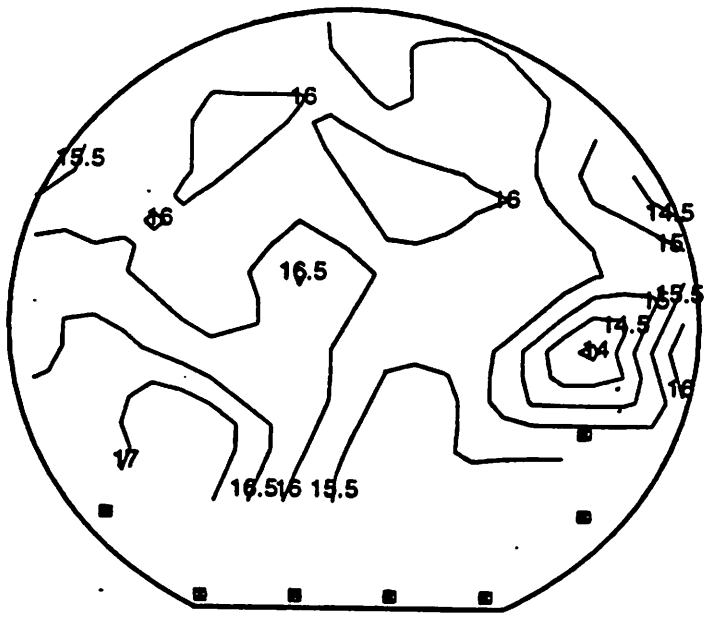


Figure 30 Wafer contour map of sheet resistance values on wafer 1.

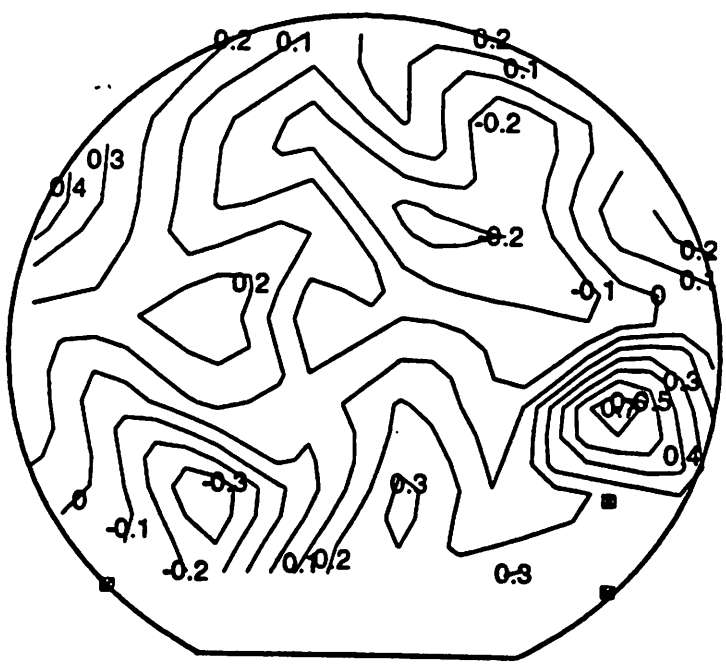


Figure 31 Wafer contour map of Δlinewidth values on wafer 1.

6.4 Conclusion

This chapter presented an example of the types of statistical analyses which can be performed on the BCAM test structures. Many other types of analyses provide considerable insight into both the process and the test structures. For example, statistical process control charts can be used to monitor a process, and generate alarms when process parameters drift beyond control limits. Additional analyses, such as the contour map example of section 6.3, can provide insight into the processing error which caused the shift in parameters. Statistical methods may also be used to provide parameter characterization for both the process and devices. These and other analysis methods will be the subject of future studies.

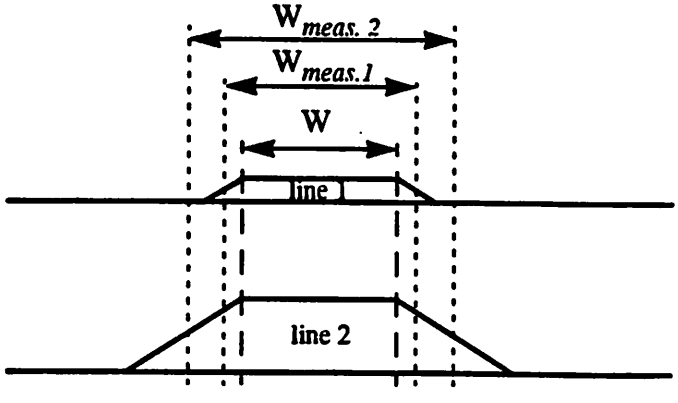


Figure 32 Illustration showing that increased poly line thickness increases linewidth.

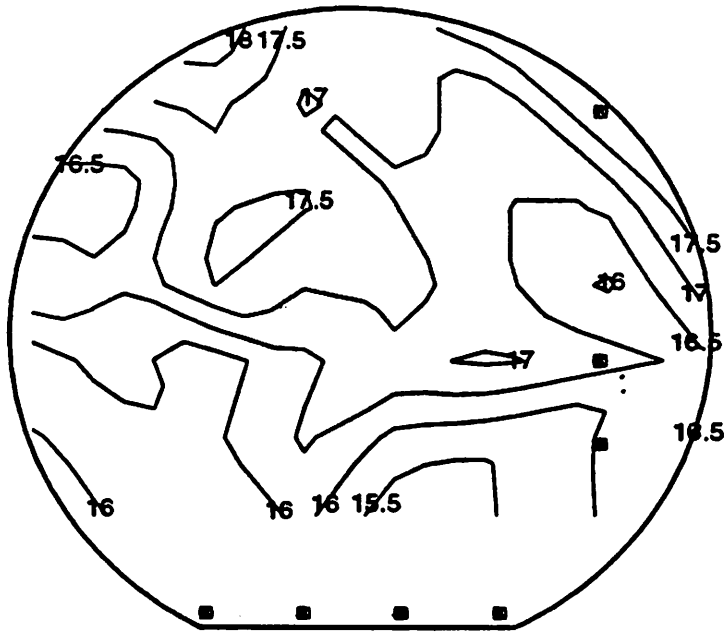


Figure 33 Wafer contour map of sheet resistance values on wafer 2.

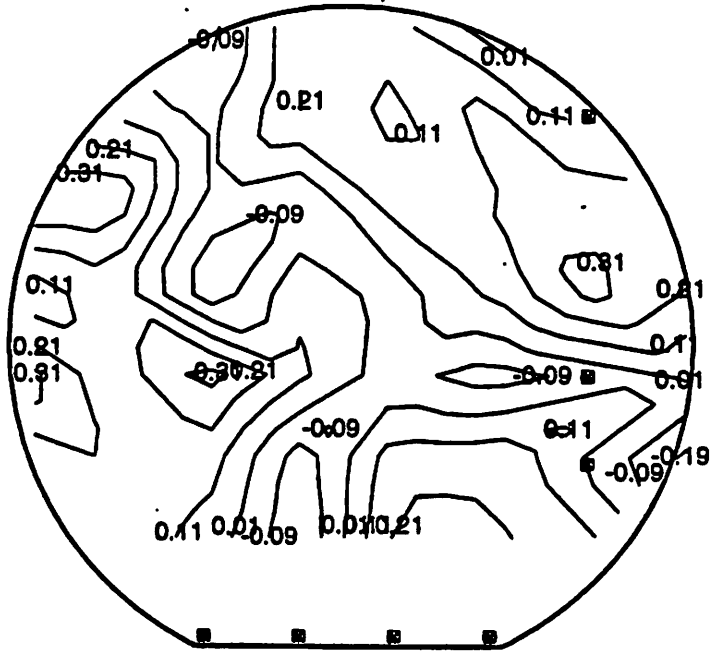


Figure 34 Wafer contour map of Δlinewidth values on wafer 2.

Chapter 7

Conclusion

A comprehensive set of test structures has been designed to provide process and device characterization, and to detect catastrophic failures and reliability problems. These structures have been arranged in such a manner that a comprehensive coverage of both stepper field and wafer area is provided. Furthermore, the organization is such that a subset of the entire test structure set can be used in the scribe lanes of all wafers fabricated in the Berkeley Microfabrication Laboratory. As illustrated in Chapter 6, the scribe lane provides a sufficient coverage of the die such that process characterization and debugging can be performed, thus providing a consistent method of process control and device characterization from lot to lot.

Furthermore, the BCAM test structure set has been designed in conjunction with the development of an automated probing system, which has provided for an efficient means of collecting the large amount of data usually required for characterization. This includes data results written in a form which is both human readable, and readable by a number of statistical analysis packages. The examples of statistical analyses in Chapter 6 show that the test structures and autoprobing system can be used to provide practical results in an efficient manner.

Further study remains concerning the specific statistical analyses necessary to provide both process and device characterization, and process control. Nonetheless, the necessary test structures and characterization routines are available for that use.

References

[1] M.G. Buehler, "Microelectronic Test Chips for VLSI Electronics," *VLSI Electronics: Microstructure Science*, vol. 6, 1983.

[2] N.H.E. Weste, K. Eshraghian, *Principles of CMOS VLSI Design*, Reading Mass.: Addison-Wesley, 1993, pp. 238.

[3] W. Lukaszek, W. Yarbrough, T. Walker, J. Meindl, "CMOS Test Chip Design for Process Problem Debugging and Yield Prediction Experiments," *Solid State Technology*, p.87-93, March 1986.

[4] C.A. Pina and G.A. Jennings, "Wafer Acceptance" in Product Assurance Technology for Procuring Reliable, Radiation-Hard, *Custom LSI/VLSI Electronics Semi-Annual Technical Report*, JPL VLSI Technology Group (February 4, 1987).

[5] U. Lieneweg and D.J. Hannaman, "Flange correction to four-terminal contact resistance measurements," *Proc. IEEE Intern. Conf. Microelectronic Test Structures*, vol. 3, p. 27, 1990.

[6] U. Lieneweg and Hoshyar R. Sayah, "Extracting Contact Misalignment From 4- and 6-Terminal Contact Resistors," *Proc. IEEE Intern. Conf. Microelectronic Test Structures*, vol. 5, p. 190, 1992.

[7] L.J. van der Pauw, "A method of measuring specific resistivity and Hall effect of discs of arbitrary shape," *Phil. Res. Rep.*, vol. 13, no. 1, 1958.

[8] M.G. Buehler and Charles W. Hershey, "The Split-Cross-Bridge Resistor for Measuring the Sheet Resistance, Linewidth, and Line Spacing of Conducting Layers," *IEEE Transactions on Electron Devices*, vol. ED-33, no. 10, 1986.

[9] D.Yen, L.W. Linholm, M.G. Buehler, "A Cross-Bridge Test Structure for Evaluating the Linewidth Uniformity of an Integrated Circuit Lithography System," *J. Electrochemical Society*, vol. 129, No. 10, 1982.

[10] P. Troccolo, L. Mantalas, R. Allen, L. Linholm, "Extending Electrical Measurements to the 0.5mm Regime", *SPIE-Integrated Circuit Metrology, Inspection, and Process Control V*, vol. 1464, 1991.

[11] M. Fallon and A.J. Walton, "Measurement of Minimum Line Widths Using Fallon Ladders," *Proc. IEEE Int. Conference on Microelectronic Test Structures*, vol. 6, March 1993.

[12] M.A. Mitchell, J. Huang and L. Forner, "Issues with Contact Defect Test Structures," *Proc. IEEE 1992 Int. Conf. on Microelectronic Test Structures*, vol. 5, p.53, 1992.

[13] M.A. Mitchell, "Defect Test Structures for Characterization of VLSI Technologies," *Solid State Technology*, p. 207, May 1985.

[14] W. Maly, "Realistic Fault Modeling for VLSI Testing," *24th ACM/IEEE Design Automation Conference*, paper 10.1, p.173, 1987.

References (cont.)

[15] Y. Uraoka, K. Eriguchi, T. Tamaki and K. Tsuji, "Evaluation Technique of Gate Oxide Damage," *Proc. IEEE Int. Conference on Microelectronic Test Structures*, vol. 6, March 1993.

[16] V. Gutnik, "Microfabrication Laboratory Manual," Chapter 8.18, Electronics Research Laboratory, College of Engineering, University of California, Berkeley, 1994.

[17] D.K. Schroder, *Semiconductor Material and Device Characterization*, New York: John Wiley & Sons, Inc., 1990, p. 123.

Appendix I

The remaining pages of this appendix contain Chapter 8.18 of the Microfabrication Laboratory Manual [16], which was written by Vadim Gutnik of the Berkeley Microfabrication Laboratory. The text contains a detailed explanation of the automated testing system, including specifics on using the hardware, adding subroutines, and understanding the Sunbase code.

Chapter 8.18

Electroglas Autoprobe in DCL
(autoprobe)

1. Introduction

This is a users' manual for the Sunbase program and the Electroglas autoprober that it runs. It is split into three parts:

- **Setup files** These files specify which dice and which devices are to be measured and what tests to execute. This also covers the output data format.
- **Hardware** Once the setup files are prepared, this describes how to turn on the instruments, align the wafer, etc.
- **Software** This describes how sunbase itself is written, how to add measurement routines and new devices, and what improvements should be added in the near future.

2. Operating Procedures

2.1. Setup files

Sunbase depends on two configuration files to run: The files are "/usr/tools/lib/eglas/die.map", which holds information that depends only on the die, and "prober.text" which specifies the tests to be run. Sunbase expects to find a "prober.text" in the directory from which it is run. The directory "~eglas" (which is identical to "/usr/users/eglas") is the usual place to run sunbase because it has plenty of disk space and a link to "die.map" but it can be run from any directory. The preliminary output is written to "output.text" and that is processed to give "final.out". Once the files are created, run sunbase simply as "sunbase." When it is finished, the script "postproc" (run it as "~eglas/code/postproc output.text") will generate a cleaner output into "final.out".

Sunbase is only executable by members of the eglas group, currently gutnik, boskin, dr, cyu, hjwann, spanos, & voros.

2.1.1. /usr/tools/lib/eglas/die.map

This file describes the locations of the devices on the die. In principle, it should contain all information about the die that can be known before testing. The general format is:

DeviceName X,Y Terminal1, Terminal2, ..., OtherInfo

The first letter of the device name determines the type of the device. X and Y are measured in microns, and increase to the right and up, respectively.

Formats for the devices already created follow:

mosfet x,y drain gate source bulk
4ptprb x,y iin gnd v1 v2
conr x,y iin gnd
fallon x,y iin gnd v1 v2 lw
scbr x,y i1 i2 i3 v1 v2 v3 v4 v5 v6 v7 layer
serp x,y

serp_comb x,y
cchain x,y

A typical mosfet might look like this:
m2 0,0 7 8 4 5

The "m" shows it is a mosfet. The only way to refence this device is by name. The two integers separated by a comma give the location of this device in microns relative to the starting location. Thus, the "0,0" indicates that sunbase will assume the probes start out touching this device every time the die is probed, specifically with probe #7 on the drain, #8 on the gate, #4 on the source, and #5 on the bulk. (Actually, unless the device is called in prober.text, sunbase will ignore it completely, so there can be several different devices at any location as long as prober.text doesn't call more than one per run. Probes are numbered clockwise from the upper left probe as they appear in the microscope:

probe #'s:
1 2 3 4 5
10 9 8 7 6

34 36 38 40 42
18 16 14 12 10

Another example: mosfet31 320,640 2 3 9 10

Another mosfet, 320 microns to the right and 640 microns up from m2.

"die.map" can contain any number of entries, and sunbase ignores blank lines and lines starting with "*" which can be used as comments. The line "@home 0,0" must appear in die.map. Sunbase goes to this device after the wafer is finished to put the probes on a device similar to the ones it started on.

Sunbase does no error checking of die.map. If necessary fields are not specified, it will crash.

2.1.2. prober.text

prober.text must be in the directory from which sunbase is run.

This file specifies the tests that the user wants to run. It starts with a 9x9 array of characters, either 0 (zero), 1 (one), or x.

```
000000100
000000000
000101000
000000101
000000000
00000x000
000000000
000000000
000000000
```

This is a crude model of the wafer to be probed: x marks the location of the initial die, 1 marks a die that will be be probed, 0 marks a die that will not be probed. Sunbase will step from the die marked x through all dice marked 1 in a non-obvious, but complete pattern. It is suggested that the dies marked with an x be near the center of the wafer, to alleviate misalignment problems.

This is followed by names of modules to be run on specific devices, and the names of the devices (from die.map) to be probed. The list of devices passed to any routine is terminated with a period. Lines beginning with "*" are ignored (comments) and "+" should be used to pass parameters to the routines. The module name must match one of the names defined below (case is important) and the device name must match one of the

devices from die.map exactly.

So, a few lines may look like this:

```
IDVG        m2          mosfet31        .

IDVDS       + VDSstop = 2       + VDSstep = .05       mosfet31       .
```

Sunbase will run the IDVG routine on m2 and mosfet31, then run IDVDS (with the given changes) on mosfet 31, print out the data, step on to the next die, and repeat the process.

2.1.3. output.text

The output is written into "output.text" in the order it was measured. The script "postproc" will rearrange the data into a format that should be readable by other programs. Run it as: "postproc output.text" to generate a file called "final.out" The conversion may take a minute or more on lead for a file of several transistors.

2.2. Hardware

(1) Everything (autoprober, 4141A source, switching matrix controller) is off.

(2) Turn on equipment: the 4084A switching matrix controller (SMC), and the 4141A DC source/monitor.

(3) Turn on prober (there's a power switch on the front panel), and the vacuum (pull the black knob next to the power switch up.) The eglas powers up with the message "XY motor blank." This means that the chuck is free to move.

(4) Push the chuck into corner closest to user (the chuck should be within a few centimeters of the corner already).

(5) Push red button on left side of joystick box (recessed in plastic) - this initializes prober and engages the motor. Do not move the chuck by hand after this point.

(6) Load the wafer onto the chuck with the flat toward the top. Check that the orientation of the dice is roughly parallel to the axes of the prober workspace. The wafer can only be rotated by ~15 degrees after it is loaded, and it's much easier to do it right from the start.

(7) Turn on vacuum to the chuck (push the [VAC] button) - the prober status lines should change to say "wafer ON."

(8) Profile the wafer height- go to the program menu (press the [PROG] button), then chose profile as the option (it's option 4; press the numerical key). The chuck will move to a pre-specified location and will move around a bit as the wafer is profiled. Occasionally, the chuck will not move from the right bumper of the work area. If this happens, press "load" to move the chuck home, press red button to release the motor, move the chuck the far (left) side manually, and back, and restart at step 5.

(9) Go back to the main menu, turn on lamp (lamp key).

(10) Align the wafer.

(a) Press [ALIGN-SCAN]. The chuck should move to the right of the area visible in the microscope and stop.

(b) [Z] moves wafer up to the probes. (If the wafer has not been profiled, the eglas will beep and will not raise the chuck.) Moving the wafer up to the probes will bring it into focus.

(c) After the first contact, [Z] toggles wafer up and down in small amounts.

(d) Use the joystick to position the wafer over some easily recognizable set of pads. These need not be the starting pads. The joystick has three modes; turn the handle to switch between them. Holding the red button moves the chuck faster. The modes are: scan - smooth movement (to scan around a die) index - moves by 1 die at a time (to step around the wafer), and jog - moves very very slowly (to align the probes to probe pads.) The mode is indicated on the eglas screen.

(e) Press [PAUSE]; the wafer will move to the left. Align the probes over pads corresponding to the ones in step (d) on this new die. If this motion did not involve movement along the Y-axis, the theta is now aligned, and you should skip to step (g).

(f) If the wafer is not aligned, press [PAUSE] again. The chuck should rotate to compensate for the calculated theta misalignment and start moving back to the right. Press [PAUSE] to stop it, and go back to step (a).

(g) After the theta alignment has been done (this is should only take a few passes) move the pads to the starting position on the correct die. (You may want to check that the alignment holds over several dice. Align the probes over the starting dice, rotate the joystick to "index" mode, and step around the wafer. The alignment should be acceptable over the entire wafer.)

(11) Run sunbase

(12) After sunbase exits, press load to unload the wafer and load the next wafer.

(13) Be sure to bring chuck back into the lower right corner before shutting off prober.

2.3. Software

2.3.1. Adding a routine (i.e. IC_VCE)

(1) Write it. Probably best to find a routine similar to what you're writing and copy it- (i.e. that's why IdVdg looks like IdVds)

(2) Add it to modules.c and modules.h in the obvious way.

(3) Add it to makefile.

(4) Recompile sunbase.

2.3.2. Adding a structure

(1) Add the name to the DevType enumeration in hash.h

(2) Add the structure declaration to hash.h

(3) Add a line to identify the device in FindDev (hash.c)

(4) Write a routine to "fill" the structure, in hash.c, and a declaration for the fill routine into hash.h

2.3.3. Usage Comments/Troubleshooting

(1) Be sure to run the most current version of sunbase; during development, the latest version is pointed to by ~eglas/sunbase.

(2) Sunbase reads the prober.text in the directory from which it is run- if you're not sure, just before you type "sunbase," type "more prober.text"; you should see the prober.text you expect to run.

(3) Check spelling. Sunbase will ignore routines names that don't match the spelling (and capitalization) in modules.c, and will abort if it looks for a nonexistant device.

(4) Don't run more than one version of sunbase at a time; the second incarnation will not be able to use the gpib and will stall. This tends to happen during debugging.

(5) Start with the probes down, (more correctly, the chuck up). There's no error checking for this.

(6) Check the position on the wafer- sunbase has no way of checking if it has moved off the wafer; this is important when probing several dice.

(7) When running long jobs, redirect screen output to /dev/null; you won't be around to see it and it slows things down.

2.3.4. Description of the Code

Sunbase consists of a core that handles communication to the instruments, reading configuration files and finding devices, and a set of "modules" that do the measurement. For the sake of consistency, I will call the actual user routines (i.e. FPP_meas()), "mroutines" and anything passed to them, be it device names or options,

"mparameters."

The core files are: main.c initial.c hash.c instruments.c modules.c modtools.c. Each routine is described in turn, in order of appearance in the source file.

(1) main.c ::: The primary probing loop.

(a) main: initialize, probe, clean up. No actual work done here, just function calls.

(b) probe: This is the loop that does all the probing. It calls preprocess to extract the wafer description (i.e. which dice to probe) and sets puts bookmark at the place where probe information (what routines and what devices to probe) starts. Then the repeated loop starts: As long as there's another die to probe, go back to the bookmark, parse, call the mroutine, continue to the end of the file. At the end of prober.text, move the wafer to the next location and loop. At the end of the wafer, move the probes to the location of the device they started on.

(c) parse: This collects the name of the mroutine (I.E. 4ptprb) as written in prober.text and the mparameters into a list, and figures out what the function name of mroutine is (in this case, FPP_meas). Basically, read a line, if it isn't empty and isn't a comment, add it to the list until the line start with a '.' which signifies the end of the list. Then call another routine to figure out which mroutine the first string in the list corresponds to.

(d) preprocess: Count the number of dice to probe and set the bookmark. This should be a little smarter, but isn't.

(e) Wafermove: Finds the next wafer to probe in the wafer, updates the current location (Xcurrent, Ycurrent) and calls "move."

(f) Pdie: prints the current die coordinates separated by tabs into a string.

(g) cleanup: Should close files, check the gpib, etc, but it just prints a message to the eglas.

(2) initial.c ::: Preliminary stuff- initialization, etc.

(a) opening_message: hello to terminal and eglas.

(b) startup: Open the probefile (typically prober.text) the outputfile (typically output.text) and device file (typically die.map) for reading and writing as necessary. Checks the gpib (sort of), converts the device file into an array.

(3) hash.c ::: Routines that relate to devices (i.e. mosfets).

(a) FindDev: Converts a string like "m123 0,0 1 2 3 4" into a device structure for, in this case, a mosfet, by calling the appropriate routine.

(b) *Fill: Actual routines to accomplish the above. There should be one for every type of device.

(c) hash: This should put the devices into some sort of hash table for quick searches. In fact it just puts everything into a sequential array.

(d) locate: Given a device name, looks up the entry (originally from device.map, put into an array by hash) corresponding to the name.

(4) instruments.c ::: Routines that relate to test equipment (i.e. eglas)

(a) devwrt: Send characters over gpib to an instrument. Adds the requisite newline.

(b) init_devs: Opens the device files for the various instruments, sends some initialization codes. This should detect errors, but doesn't.

(c) check_gpib: Placeholder for any real testing of communication.

(d) screenwrite: Writes a string to the screen of the eglas terminal.

(e) move: Lowers the chuck, moves it (in die steps), catches acknowledgement from eglas (though it doesn't check errors) and raises the chuck back. Also resets the "microdie," or intradie position.

(f) align wafer: Prompts the user through alignment. Written out during testing.

(g) prober_self_test: This will someday check all the instruments and the gpib before operation. Does absolutely nothing now.

(h) need_move: keeps track of current location within a die to avoid moving by 0 (which takes time and chuck lowering/raising).

(i) MoveTo: Lowers the chuck, moves it to a new location *within the die* and raises the chuck back. Updates current location.

(j) connect: "port" refers to the signal input/output channel on the 4141A. "pin" refers to one of the ten probes that touch pads on the wafer. If both port and pin are nonzero, this connects the given port to that pin through the SMC. (NOTE: the SMC will not allow two ports to connect to the same pin. For example, if port 1 is connected to pin 10, and you send a command to connect port 2 to pin 10, the SMC will break the connection between port 1 and pin 10 before making the new connection. It is best to make this explicit in the code, though.)

(k) ReadBuf: Returns, as a string, the contents of the DCS buffer.

(l) DCShold: Directs the DCS to source a current or voltage at a given setting and with a specified compliance. "source" is the port number, "mode" is either 'V' or 'I' for voltage or current, "setting" and "compliance" are both floats NOT DOUBLES.

(m) DCSMeasure: Single point measurement of a given channel and mode. Note that the DCS will not measure the voltage of a voltage source or the current from a current source directly, so if you really need to confirm the source output, you have to connect another channel to it. See connect for caveat. DCSmeasure returns the contents of the DCS output buffer.

(n) DCSsweep: sets up sweep parameters for a channel; no measurement is actually done. Some of the options (i.e. linlog, SECONDARY vs PRIMARY sweep) have not been fully tested.

(o) DCStrack: ...is the parallel to DCSMeasure- given a list of sources, it directs the DCS to keep track of the measurements for those channels, then triggers a sweep measurement. DCStrack returns a string; see the DCS manual for the specific format.

(p) dmake: This routine parses the output string into a dtype, keeping all the information (i.e. flags, which channel it was, etc.) This should be used to parse the output of DCSMeasure.

(q) DatFormat: Parses an array, as returned by DCS track. Check DCS documentation for the format of the output, or look at idvds.c for an example.

(r) numpoints: Queries the DCS for the number of measurement points in the last sweep; this is mostly a sanity check, because DatFormat can simply count the points as they're read.

(5) modules.c ::: Command-> routine parser.

(a) ident: Matches a string from prober.text to the appropriate mroutine.

(b) Ignore: Throws away arguments passed to a command that doesn't match an mroutine, and handles comments, etc.

(6) modtools.c ::: Declarations and utilities for the modules.

(a) V_diff: Finds voltage difference between two given pins and cleans up.

(b) Discon: Disconnect a list of pads.

2.3.5. List of Current Routines The modules: idvds.c, FPP.c, Fallon.c, ... are located in ~gutnik/dcl/code.

The existing modules are:

Module	Writer	What it does
~gutnik/dcl/code/scbr.2.c	dr	
~gutnik/dcl/code/FPP.c	gutnik	Four-point-probe (any resist.)
~gutnik/dcl/code/vdp.c	gutnik	Van-der-Pauw resistance
~gutnik/dcl/code/idvds.c	gutnik	Data for (nmos) Id-Vds curves
~gutnik/dcl/code/idvdg.c	gutnik	Data for (nmos) Id-Vdg curves
~gutnik/dcl/code/Fallon.c	dr	
~gutnik/dcl/code/conr.c	dr	
~gutnik/dcl/code/serp.c	dr	
~gutnik/dcl/code/comb.c	dr	
~gutnik/dcl/code/serpcomb.c	dr	

V Gutnik
5/20/94
D Rodriguez
7/13/94
KKH
7/14/94

APPENDIX A
Programming Notes

General Notes

1. REMEMBER - all compiling must be done on lead - sun3 with GPIB-SCSI library.

2. The device created in /dev are spa (address 7), eglas (address x), dcs (address 23), smc (add 22), cv (address 17). Addresses in decimal. To add device, root must run ibconf and then reboot the machine.

3. The pad numbers are in sunbase.h as comments. if there is a mistake in the numbers in die.map, no error will be flagged.

4. You don't need to recompile when changing die.map.

5. Numbers below are in mils - electroglas powers up to move in microns!!!!

To Identify Coordinates of Second Transistor

1. best method - find correct VEM/KIC/MAGIC layout and measure.

2. otherwise -
start ibic (/usr/tools/gpib/bin)
ibfind eglas - talking to eglas. use name from /dev file
ibwrt "?H0 - use capitals, and to terminate.
should echo cmpl (without err)
ibrsp
ibrd 20 - reads 20 bytes
eglas: ibrd 20
[2100] (end cmpl)
count: 15
48 58 38 38 34 38 39 59 H X 8 8 4 8 9 Y
36 33 32 39 35 0d 0a 6 3 2 9 5 . .

x is x position in absolute machine position, in mils, y is y position

move to "next" transistor
eglas: ibrsp
[0100] (cmpl)
Poll: 0x00

eglas: ibwrt "?H0
[0100] (cmpl)
count: 3

eglas: ibrsp
[0100] (cmpl)
Poll: 0x40

eglas: ibrd 20
[2100] (end cmpl)
count: 15
48 58 38 38 34 38 39 59 H X 8 8 4 8 9 Y
36 33 31 37 30 0d 0a 6 3 1 7 0 . .

we have moved 125 mils up!
quit exits ibic. must exit or you'll tie up hpib.

In case of problems:

ibrsp - clears serial poll in case of mistake.
hit online to clear eglas if it looks confused.
try again.

Changed air sensor x & y from 23768, 47022 respectively to 23000, 46000 to avoid profiling errors. In SET PARAMETER, 12-3 menu.

V Gutnik
5/20/94
D Rodriguez
7/13/94
KKH
7/14/94

Appendix II

This appendix contains a listing of each measurement subroutine currently programmed into Sunbase. The code is available on the Argon cluster, and is contained in the "~gutnik/dcl/code" directory. The following table lists the subroutines currently available, their file names, author, and output. The actual code listing follows in the remainder of this appendix.

Currently Available Measurement Subroutines for Autoprober.

Measurement	File name	Author	Output
I_d-V_{ds} curve	idvds.c	V. Gutnik	Id-Vds characteristic
I_d-V_g curve	idvg.c	V. Gutnik	I_d-V_g characteristic
Four Point Probe	FPP.c	V. Gutnik	Resistance
Van Der Pauw	vdp.c	V. Gutnik	Sheet resistance (R_S)
Split-Cross-Bridge	scbr.2.c	D. Rodriguez	R_S, ΔW, Spacing, Pitch
Fallon Ladder	Fallon.c	D. Rodriguez	Ladder resistance, min. linewidth resolved
Contact Resistance	conr.c	D. Rodriguez	Contact resistances (left, right, avg.)
Comb Defect	comb.c	D. Rodriguez	5 Binary result showing shorts (defects)
Serpentine Resistance	serp.c	D. Rodriguez	Resistances for 5 serpentines in pad set
Serpentine/Comb Defect	serpcomb.c	D. Rodriguez	2 Binary result showing opens/shorts (defects)

idvds.h

```
#define MODULE "IDVDS"
#define FORMAT STDFORM "VDS\tvgs\tID"

#include "modtools.h"

#define VGSstart_def    0
#define VGSstop_def     6
#define VGSstep_def     1
#define VDSstart_def    0
#define VDSstop_def     6
#define VDSstep_def     .1

module_function ID_VDS;
~
```

idvds.c

```
#include "idvds.h"

void *ID_VDS (char **paramlist, FILE *output) {
  FetType *dut;
  int numpoints,j;
  float i;
  float vgsstart = VGSstart_def;
  float vgsstop = VGSstop_def;
  float vgsstep = VGSstep_def;
  float vdsstart = VDSstart_def;
  float vdsstop = VDSstop_def;
  float vdsstep = VDSstep_def;
  char *result;
  DatArrType datarray;
  D(printf ("< IDVDS\n");)
  PARSEBEGIN;
  while (*++paramlist) {
    if (**paramlist == '+') {
      sscanf (*paramlist,"+ VGSstart = %g",&vgsstart);
      sscanf (*paramlist,"+ VGSstop = %g",&vgsstop);
      sscanf (*paramlist,"+ VGSstep = %g",&vgsstep);
      sscanf (*paramlist,"+ VDSstart = %g",&vdsstart);
      sscanf (*paramlist,"+ VDSstop = %g",&vdsstop);
      sscanf (*paramlist,"+ VDSstep = %g",&vdsstep);
      continue;
    }
    dut = FindDev (*paramlist);
    MoveTo (dut);
    DCSturnoff(0);
    connect (4,dut->source);
    connect (3,dut->bulk);
    connect (2,dut->gate);
    connect (1,dut->drain);
    DCShold (4,'V',0,.1);
    DCShold (3,'V',0,.1);
    for (i=vgsstart; i<= vgsstop; i+=vgsstep) {
      DCShold (2,'V',i,.01);
      DCSsweep (1,VOLTAGE,LINEAR,vdsstart,vdsstop,vdsstep,.01);
```

```
      result = DCStrack ("1");
      numpoints=DatFormat (&datarray,result,1);
      free (result);
      for (j=0;j<numpoints;j++) {
        fprintf (output,"%s\t%s\t%d\t%d\t", Pdie(),dut->Name,dut->X,dut->Y);
        fprintf (output,"%g\t%g\t",datarray[j][1]->value,i);
        fprintf (output,"%g\n" ,datarray[j][0]->value);
      }
    }

  DCSturnoff (0);
  }
  PARSEEND;
  return NULL;
}
```

idvg.h

```
#define MODULE "IDVGS"
#define FORMAT STDFORM "VGS\tID\tvbs"
#include "modtools.h"

#define VGstart   0
#define VGstop    7
#define VGstep    .1
#define VBstart   0
#define VBstep    -1
#define VBstop    -4
#define VDS       50e-3

module_function IDVG;
```

idvg.c

```
#include "idvg.h"

void *ID_VG (char **paramlist, FILE *output) {
  FetType *dut;
  int numpoints;
  float i;
  int j;
  char *result;
  DatArrType datarray;
  D(printf ("< IDVG\n");)
  PARSEBEGIN;
  while (*++paramlist) {
    dut = FindDev (*paramlist);
    MoveTo (dut);
    DCSturnoff(0);
    connect (4,dut->source);
    connect (3,dut->bulk);
    connect (2,dut->gate);
    connect (1,dut->drain);
    DCShold (1,'V',VDS,.1);
    DCShold (4,'V',0,.1);
    for (i=VBstart; i>= VBstop; i+=VBstep) {
      DCShold (3,'V',i,.01);
      DCSsweep (2,VOLTAGE,LINEAR,VGstart,VGstop,VGstep,.01);
      result = DCStrack ("1");
      numpoints=DatFormat (&datarray,result,1);
      free (result);
      for (j=0;j<numpoints;j++) {
        fprintf (output,"%s\t%s\t%d\t%d\t", Pdie(),dut->Name,dut->X,dut->Y);
        fprintf (output,"%g\t",datarray[j][1]->value);
        fprintf (output,"%g\t%g\n" ,datarray[j][0]->value,i);
      }
    }
    DCSturnoff (0);
  }
  PARSEEND;
  return NULL;
}
```

FPP.h

```
#define MODULE "four_point_probe"
#define FORMAT STDFORM "v3\tvdiff\tiout(mA)\tRESISTANCE"

#include "modtools.h"

module_function FPP_meas;
#define RSCURRENT 0.006
#define GVLT 0
#define DLAY 0

/*  CRID'S FUNKY NUMBERS */
/* #define RSCURRENT 0.010 */
/* #define GVLT -1.50 */
/* #define DLAY 200 */
```

FPP.c

```
#include "FPP.h"

void *FPP_meas (char **paramlist, FILE *output) {
  double vdiff,v3,iout;
  int i;
  static float Resistance;
  FPPType *dut;
  D(printf ("< FPP\n");)
  PARSEBEGIN;
  while (*++paramlist) {
    dut = FindDev (*paramlist);
    MoveTo (dut);
    connect (1,dut->GND);                  /* connect sources 1&2 to */
    connect (2,dut->iin);                  /* the right pads. */
    DCShold (1,'V', GVLT, .05);                 /* set up a ground */
    DCShold (2,'I', RSCURRENT ,8);               /* source the current */
    for (i=0;i<=DLAY;i++)
     printf("delay %d\n",i);
    vdiff = V_diff (dut->v2, dut->v1)->value;
    v3 = dmake (DCSMeasure (2,'V'))->value;
    iout = -1.0*dmake(DCSMeasure (1,'I'))->value;
    Resistance = vdiff/iout;
    fprintf (output,"%s\t%s\t%d\t%d\t%g\t%g\t%7.4f\t%g\n", \
             Pdie(),dut->Name,dut->X,dut->Y,v3,vdiff,1000.0*iout, Resistance);

    DCSturnoff (0);                        /* Disconnect all the pins, */
                                           /* set all sources to Zero */
                                           /* Output */
  }
  PARSEEND;
  return &Resistance;
}
```

vdp.h

```
#define MODULE "Van Der Paw"
#include "modtools.h"

#define PI 3.14
module_function Van_der_Pauw;
module_function FPP_meas;
```

vdp.c

```
#include "vdp.h"

void *Van_der_Pauw (char **paramlist, FILE *output) {
  float *R1;
  R1 = FPP_meas(paramlist,output);
  fprintf (output,"The VdP sheet resistance is %g\n",*R1*PI/log(2.0));
  return NULL;
}
```

scbr.2.h

```
#define MODULE "SCBR"
/* #define FORMAT STDFORM "Layer\tRs\tWbDrawn\tDeltaWb\tWsDrawn\tDelWsbot\tdelWstop\tS\tP"
*/
#define FORMAT STDFORM "irs\tibridge\tv1Rs\tv2Rs\tv2lw\tv3lw\tv4lw\tv5lw\tv6lw\tv7lw"
#include "modtools.h"

#define TestCurrent 0.0005
#define RSCURRENT 0.008
#define PI 3.14159
#define DLAY 800

/* double strtoval(char *spastring); */
double pad_voltage (int source, int pad);
module_function SCBR_meas;
```

scbr.2.c

```
#include "scbr.2.h"

/* This will need a bit more processing to strip the N ( or T,...) and the
comma's from the string.  The HP manual has a section on what the output
looks like. It may be easiest to just use that format. ?? */

double pad_voltage (int source, int pad) {
  double result;
  connect (source,0);                              /* Be sure not to short
                                                     different pads. */
  connect (source,pad);
  result = dmake(DCSMeasure (source,'V'))->value;   /* get the result */
  return (result);
}

void *SCBR_meas (char **paramlist, FILE *output) {
  SCBRType *dut;
  char *result;
  double v1Rs,v2Rs,v2lw,v3lw,v4lw,v5lw,v6lw,v7lw,Rs,S;
  double Wb,deltaWb,Wstop,deltaWstop,Wsbot,deltaWsbot;
  double P, ignd,ignd2;
  double WBDRAWN, LStop,LSbot,WSDRAWN,LB;
  int i;


  D(printf ("< SCBR_meas\n");)
  PARSEBEGIN;
  result = (char *) calloc (200, sizeof(char));
  while (*++paramlist) {
    /* probe first device */
    dut =  FindDev (*paramlist);

    MoveTo (dut);
    WBDRAWN=  dut->WBDRAWN;
    LStop=  dut->LStop;
    LSbot=  dut->LSbot;
```

```
    WSDRAWN=  dut->WSDRAWN;
    LB=  dut->LB;

    /* Measure voltages for Rs calculations */


    connect (1,dut->i1);                    /* connect sources 1&2 to */
                                            /* the right pads. */
    connect (2,dut->i2);

    DCShold (2,'V',0,.1);                   /* set up a ground */
    DCShold (1,'I',RSCURRENT,10);      /* source the current */


    v1Rs = pad_voltage (5,dut->v1);         /* Use the routine above to */
                                            /* get the appropriate voltage */
    v2Rs = pad_voltage (5,dut->v2);

    ignd = -1*dmake(DCSMeasure (2,'I'))->value;

    /* Measure voltages for linewidth calculations */

    DCShold (1,'I',TestCurrent,10);       /* source the current */

    connect (2,dut->i3);                    /* CONNECT GROUND to another */
                                            /* pad of the device */

    connect (0,dut->i2);                    /* DISCONNECT GROUND from the */
                                            /* first pad. */
/*v2lw=dmake(DCSMeasure (5,'V'))->value;*/
    v2lw = pad_voltage (5,dut->v2);
    v3lw = pad_voltage (5,dut->v3);
    v4lw = pad_voltage (5,dut->v4);
    v5lw = pad_voltage (5,dut->v5);
    v6lw = pad_voltage (5,dut->v6);
    v7lw = pad_voltage (5,dut->v7);
    ignd2 = -1*dmake(DCSMeasure (2,'I'))->value;

    DCSturnoff (0);                         /* Disconnect all the pins, */
                                            /* set all sources to "Zero */
                                            /* Output" */

    /**** calc Rs ****/
    Rs = (v1Rs-v2Rs)*(PI/log(2.0))/(ignd);

    /**** Width of Bridge (Wb) ****/
    Wb = Rs*LB*ignd2/(v2lw-v3lw);
    deltaWb = WBDRAWN-Wb;

    /* Width of bottom split-bridge */

    Wsbot = .5*Rs*LSbot*ignd2/(v4lw-v5lw);
    deltaWsbot = WSDRAWN-Wsbot;


    /* Width of top split-bridge */
    Wstop = .5*Rs*LStop*ignd2/(v6lw-v7lw);
    deltaWstop = WSDRAWN-Wstop;
```

```
    /**** Line Spacing (S) ****/
    S = Wb-Wstop-Wsbot;

    /**** Pitch (P) ****/
    P = .5*(Wsbot+Wstop) + S;

    fprintf (output,"%s\t%s\t%d\t%d\t%s\t",Pdie(),dut->Name,dut->X, dut->Y,dut->Layer);
/*    fprintf (output,"%g\t%g\t%g\t%g\t%g\t%g\t%g\t%g\n",
                    Rs,WBDRAWN,deltaWb,WSDRAWN,deltaWsbot,deltaWstop,S,P); */

/*    D(fprintf (output, "i1: %g, i2: %g v1Rs: %g v2Rs: %g v2:%g v3:%g v4:%g v5:%g v6:%g
v7:%g\n",ignd,
     ignd2,v1Rs,v2Rs,v2lw,v3lw,v4lw, v5lw, v6lw, v7lw);) */
    fprintf (output, "%g\t%g\t%g\t%g\t%g\t%g\t%g\t%g\t%g\t%g\n",ignd,
     ignd2,v1Rs,v2Rs,v2lw,v3lw,v4lw, v5lw, v6lw, v7lw);


    for (i=0;i<=DLAY;i++)
       printf("delay %d\n",i);

  }
    PARSEEND;
    return NULL;
}
```

Fallon.h

```
#define MODULE "fallon"
#define FORMAT STDFORM "lw0(d)\tR0\tlw1(d)\tR1\tlw2(d)\tlw2(m)\tR2\tlw3(d)\tlw3(m)\tR3"

#include "modtools.h"

module_function Fallon_meas;
float FallonR (FILE *output, FallonType *NextDev);
~
```

Fallon.c

```
#include "Fallon.h"

float FallonR (FILE *output, FallonType *NextDev) {
  double vdiff, v3, iout;
  double NextR;
  MoveTo (NextDev);
  connect (1,NextDev->GND);
  connect (2,NextDev->iin);
  DCShold (1,'V',0,.05);
  DCShold (2,'I',.001,5);
  vdiff = V_diff (NextDev->v1,NextDev->v2)->value;
  v3 = dmake(DCSMeasure (2,'V'))->value;
  iout = -1.0*dmake(DCSMeasure (1,'I'))->value;
  NextR = vdiff/iout;
/*  fprintf (output,"vdiff=%g v3=%g iout=%2g",vdiff, v3,1000*iout); */
  DCSturnoff (0);
  return NextR;
}

void *Fallon_meas (char **paramlist, FILE *output) {
  double R[5];
  double lwdrawn[5];
  double lw2,lw3,slope,b;
  int i;
  FallonType *dut;
  fprintf ("< Fallon_meas\n");)
  PARSEBEGIN;
  while (*++paramlist) {
    for (i=0; i < 4; i++) {
      /* PROBE Ith DEVICE */
      dut = FindDev (*paramlist++);
      R[i]= (double) FallonR (output,dut);
      lwdrawn[i] = dut->lw;
/*      fprintf (output,"r%d= %g lw%d= %g\n",i, R[i], i,lwdrawn[i]); */
    }

    /* CALCULATE MIN. LINEWIDTH RESOLVED */

    /* Calculate equation for Resistance vs. Linewidth */
    slope = (R[1]-R[0])/(lwdrawn[1]-lwdrawn[0]);
    b = R[0] - slope*(lwdrawn[0]);

    /* Use line to estimate min. linewidth resolved. */
    lw2 = (R[2]-b)/slope;
```

```
    lw3 = (R[3]-b)/slope;

    fprintf (output,"%s\t%s\t%d\t%d\t",Pdie(),dut->Name,dut->X, dut->Y);
    fprintf (output,"%g\t%g\t%g\t%g\t%g\t%1.4f\t%g\t%g\t%1.4f\t%g\n",- \
            lwdrawn[0],R[0],lwdrawn[1],R[1],lwdrawn[2],lw2,R[2],lwdrawn[3],lw3,R[3]);

  }
  PARSEEND;
  return NULL;
}
```

conr.h

```
#define MODULE "CONR"
#define FORMAT STDFORM "v3\tiout(mA)\tRLavg\tRRavg\tRavg"

#include "modtools.h"

module_function Conr_meas;
```

conr.c

```
#include "conr.h"

void *Conr_meas (char **paramlist, FILE *output)
{
  double v3,iout,Rleft1,Rleft2, Rright1,Rright2;
  double RLavg,RRavg,Ravg;
  ConrType *dut;
  D(printf ("< conr\n");)
  PARSEBEGIN;

  while (*++paramlist) {
    /* probe first device */
    dut = FindDev (*paramlist);
    MoveTo (dut);
    /* Measure voltages for contact resistance  calculations */
    connect (1,dut->GND);                  /* connect sources 1&2 to */
                                           /* the right pads. */
    connect (2,dut->iin);
    DCShold (1,'V',0,.05);                  /* set up a ground */
    DCShold (2,'I',.003,8);                 /* source the current */
    v3 = dmake (DCSMeasure (2,'V'))->value;
    iout = -1.0*dmake(DCSMeasure (1,'I'))->value;
    Rleft1 = (V_diff (10,2)->value)/iout;
    Rright1 = (V_diff (9,3)->value)/iout;
    Rleft2 = (V_diff (8,4)->value)/iout;
    Rright2 = (V_diff (7,5)->value)/iout;

    RLavg = (Rleft1+Rleft2)/2;
    RRavg = (Rright1 + Rright2)/2;
    Ravg = (RLavg + RRavg)/2;
    fprintf (output,"%s\t%s\t%d\t%d\t",Pdie(),dut->Name,dut->X, dut->Y);
    fprintf (output,"%3.4f\t%3.4f\t%g\t%g\t%g\n",v3,1000.*iout, RLavg, RRavg,Ravg);

    DCSturnoff (0);                        /* Disconnect all the pins, */
                                           /* set all sources to Zero */
                                           /* Output */
  }
  PARSEEND;
  return NULL;
}
```

comb.h

```
#define MODULE
#include "instruments.h"
#include "hash.h"

module_function Comb_meas;
```

comb.c

```
#include "comb.h"
#define IHOLD .0001

void *Comb_meas (char **paramlist, FILE *output)
{
  double v1,iout;
  GenericDev *dut;
  int x,aligned=1,shrt[6];

  D(printf ("< comb\n");)
  while (*++paramlist) {
    /* probe first device */
    dut = FindDev (*paramlist);
    MoveTo (dut);

    /* check for proper alignment using pads 1 & 5 */
    connect (1,5);
    connect (2,1);
    DCShold (1,'V',0,.05);
    DCShold (2,'I',IHOLD,8);
    iout = dmake(DCSMeasure (1,'I'))->value;
    connect (2,0);
    connect (1,0);

    /* IF NOT aligned properly then skip measurements */

    if ( -1*(iout) < (.2*IHOLD)) {
        shrt[0]=shrt[1]=shrt[2]=shrt[3]=shrt[4]=shrt[5]=-10;
        aligned=0;
       }
    else {
      aligned =1;
      for (x=1;x<6; x++) {
        /* Measure voltages for contact resistance  calculations */
        connect (1,x);
        connect (2,10-x+1);
        DCShold (1,'V',0,.05);                  /* set up a ground */
        DCShold (2,'I',IHOLD,8);                 /* source the current */
        v1 = dmake(DCSMeasure (2,'V'))->value;
        iout = dmake(DCSMeasure (1,'I'))->value;
        if ( (-1.0*iout) < (.1*IHOLD))
          shrt[x]=0;
        else
          shrt[x] = 1;
       DCSturnoff fprintf (output,"%s (ALIGNED=%d):DEFECT[1..5]= ",dut->Name,aligned);
    for (x=1;x<6;x++)
      fprintf (output,"%d",shrt[x]);
```

```
    fprintf (output,"\n");
  }
  return NULL;
}
(0);                          /* Disconnect all the pins, */
      }
    }
```

serp.h

```
#define MODULE "Serp"
#define FORMAT STDFORM "iout(mA)\talign.\tX\tR1\tR2\tR3\tR4\tR5"
#include "modtools.h"

module_function Serp_meas;
```

serp.c

```
#include "serp.h"
#define IHOLD .0001

void *Serp_meas (char **paramlist, FILE *output)
{
  double v1,iout,R[6];
  GenericDev *dut;
  int x,aligned=1;
  FILE *Soutput;

  Soutput=fopen("Soutput.text","a");
  D(printf ("< serp\n");)
  PARSEBEGIN;
  while (*++paramlist) {
    /* probe first device */
    dut = FindDev (*paramlist);
    MoveTo (dut);
    /* check for proper alignment */
    connect (1,6);
    connect (2,10);
    DCShold (1,'V',0,.05);
    DCShold (2,'I',IHOLD,8);
    iout = dmake(DCSMeasure (1,'I'))->value;
    connect (2,0);
    connect (1,0);
/*    D(fprintf (output,"-1*iout = %g\n",-1.0*iout);) */
    if ( -1*(iout) < (.8*IHOLD)) {
        R[0]=R[1]=R[2]=R[3]=R[4]=R[5]=9e10;
        aligned=0;
       }
    else {
      for (x=1;x<6; x++) {
        aligned=1;
        /* Measure voltages for contact resistance  calculations */
        connect (1,9);
        connect (2,x);
        DCShold (1,'V',0,.05);                  /* set up a ground */
        DCShold (2,'I',IHOLD,8);                 /* source the current */
        v1 = dmake(DCSMeasure (2,'V'))->value;
        iout = dmake(DCSMeasure (1,'I'))->value;
        if ( (-1.0*iout) < (.8*IHOLD))
          R[x]=1e30;
        else
          R[x] = (-1*v1)/iout;
        DCSturnoff (0);                         /* Disconnect all the pins, */

/*      D(fprintf (output,"%s: v1 %g iout %2g R[%d]= %g\n", \
```

```
            dut->Name,v1,1000*iout,x, R[x]);) */
      }
    }
    fprintf (output,"%s\t%s\t%d\t%d\t",Pdie(),dut->Name,dut->X, dut->Y);

    fprintf (output,"%1.3f\t%d\t%d\t%g\t%g\t%g\t%g\t%g\n", \
               1000.0*iout,aligned,x,R[1],R[2],R[3],R[4],R[5]);

    /* OUTPUT Human-readable FILE output.text */
/*    for (x=1; x < 6; x++)
        fprintf (output,"%2g\t%d\t%d\t%g\n",1000.0*iout,aligned,x, R[x]);
    fprintf (output,"\n"); */

    /* OUTPUT S-readable FILE Soutput.text */
/*    if (aligned) { */
      /* OUTPUT Values for Splus */
/*      fprintf (Soutput,"Resistance.%s = c(%g",dut->Name,R[1]);
      for (x=2; x < 6; x++)
        fprintf (Soutput,",%g", R[x]);
      fprintf (Soutput,")\n");
    } */
  }
  fclose(Soutput);
  PARSEEND;
  return NULL;
}
```

serpcomb.h

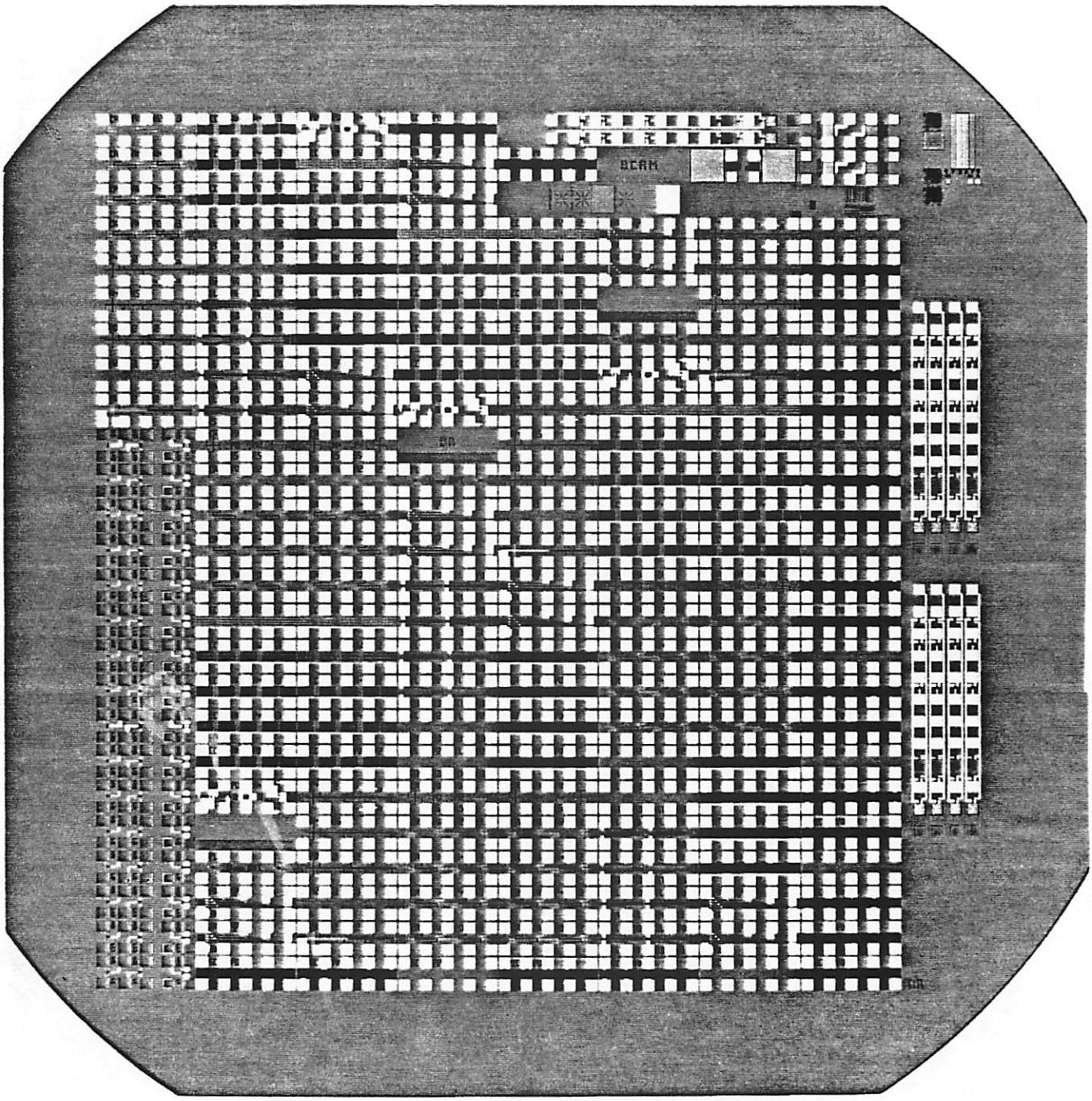
```
#define MODULE
#include "instruments.h"
#include "hash.h"

module_function Serpcomb_meas;
```

serpcomb.c

```
#include "serpcomb.h"
#define IHOLD .00005

void *Serpcomb_meas (char **paramlist, FILE *output)
{
  double v1, iout,iout1,iout2;
  GenericDev *dut;
  int x,aligned=1,open[3],shrt[3];
  FILE *Soutput;

  Soutput=fopen("Soutput.text","a");
  D(printf ("< serpcomb\n");)
  while (*++paramlist) {
    /* probe first device */
    dut =  FindDev (*paramlist);
    MoveTo (dut);
    /* check for proper alignment using pads 10 & 9 */
    connect (1,9);
    connect (2,10);
    DCShold (1,'V',0,.05);
    DCShold (2,'I',IHOLD,8);
    iout1 = dmake(DCSMeasure (1,'I'))->value;
    connect (2,0);
    connect (1,0);
    /* check for proper alignment using pads 4 & 5 */
    connect (1,5);
    connect (2,4);
    DCShold (1,'V',0,.05);
    DCShold (2,'I',IHOLD,8);
    iout2 = dmake(DCSMeasure (1,'I'))->value;
    connect (2,0);
    connect (1,0);
    D(fprintf (output,"-1*iout1 = %g\n",-1.0*iout1*1000);)
    D(fprintf (output,"-1*iout2 = %g\n",-1.0*iout2*1000);)
    /* IF NOT aligned-properly then skip measurements */
    if (( -1*(iout1) < (.2*IHOLD)) || ( -1*(iout2) < (.2*IHOLD))) {
        open[1]=open[2]=shrt[1]=shrt[2]=-10;
        aligned=0;
       }
    else {
      for (x=1;x<3; x++) {
        /* CONTINUITY CHECK FOR SERPENTINE ONLY */

        connect (2,2*x-1);
```

```
    connect (1,10-2*x);
    DCShold (1,'V',0,.05);                    /* set up a ground */
    DCShold (2,'I',IHOLD,8);                   /* source the current */
    v1 = dmake(DCSMeasure (2,'V'))->value;
    iout = dmake(DCSMeasure (1,'I'))->value;
    if ( (-1.0*iout) < (.2*IHOLD))
      open[x] = 1;
    else
      open[x] = 0;
    DCSturnoff (0);                           /* Disconnect all the pins, */
    D(fprintf (output,"%s: v1 %g iout %2g open[%d]= %d\n", \
        dut->Name,v1,1000*iout,x, open[x]);)
  }

  /* CONTINUITY CHECK BETWEEN SERPENTINE AND COMBS */
  for (x=1;x<3; x++) {
    /* CONTINUITY CHECK BETWEEN BOTTOM COMB AND SERPENTINE */
    connect (2,2*x-1);
    connect (1,10-2*x);
    DCShold (1,'V',0,.05);                    /* set 1st ground     */
    DCShold (2,'V',0,.05);                    /* set 2nd ground     */
                                              /* (2 gnds necessary  */
                                              /* in case serp open) */
    connect (3,10-2*x+1);
    DCShold (3,'I',IHOLD,8);                   /* source the current */
    v1 = dmake(DCSMeasure (3,'V'))->value;
    iout1 = dmake(DCSMeasure (1,'I'))->value;
    iout2 = dmake(DCSMeasure (2,'I'))->value;
    if (( -1*(iout1) > (.2*IHOLD)) || ( -1*(iout2) > (.2*IHOLD)))
      shrt[x] = 1;
    else
      shrt[x] = 0;
  D(fprintf (output,"BOT:\n%s: v1 %g iout1 %2g iout2 %2g short[%d]= %d\n", \
        dut->Name,v1,1000.0*iout1,1000.0*iout2,x, shrt[x]);)
    connect(3,0);                          /* Disconnect current source */
    /* CONTINUITY CHECK BETWEEN TOP COMB AND SERPENTINE */
    connect (3,2*x);
    DCShold (3,'I',IHOLD,8);                   /* source the current */
    v1 = dmake(DCSMeasure (3,'V'))->value;
    iout1 = dmake(DCSMeasure (1,'I'))->value;
    iout2 = dmake(DCSMeasure (2,'I'))->value;
    if (( -1*(iout1) > (.2*IHOLD)) || ( -1*(iout2) > (.2*IHOLD)))
      shrt[x] += 1;
   D(fprintf (output,"TOP:\n%s: v1 %g iout1 %2g iout2 %2g short[%d]= %d\n", \
        dut->Name,v1,1000.0*iout1,1000.0*iout2,x, shrt[x]);)
   DCSturnoff (0);                         /* Disconnect all the pins, */
   }
}
/* OUTPUT Human-readable FILE output.text */
for (x=1; x < 3; x++)
  fprintf (output,"%s (aligned=%d): open[%d]=%d short[%d]=%d\n", \
          dut->Name,aligned,x,open[x],x,shrt[x]);
fprintf (output,"\n");
/* OUTPUT S-readable FILE Soutput.text */
if (aligned) {
   /* OUTPUT Values for Splus */
   fprintf (Soutput,"open.%s = c(%d,%d)\n",dut->Name,open[1],open[2]);
}
if (aligned) {
```

```
        /* OUTPUT Values for Splus */
        fprintf (Soutput,"short.%s = c(%d,%d)\n",dut->Name,shrt[1],shrt[2]);
      }
    }
    fclose(Soutput);
    return NULL;
}
```


BEAM