New Methodologies for Interconnect Reliability Assessments of Integrated Circuits

by

STEFAN P. HAU-RIEGE

Diplom-Physiker
Universität Hamburg, Germany, 1996

Submitted to the Department of Materials Science and Engineering
In Partial Fulfillment of the Requirements of the Degree of

DOCTOR OF PHILOSOPHY IN ELECTRONIC MATERIALS

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2000

Signature of Author________________________________
Department of Materials Science and Engineering
April 28, 2000

Certified by________________________________
Carl V. Thompson
Stavros Salapatas Professor of Materials Science & Engineering
Thesis Supervisor

Accepted by________ __________________
Carl V. Thompson
Stavros Salapatas Professor of Materials Science & Engineering
Chairman, Department Committee on Graduate Students

New Methodologies for Interconnect Reliability Assessments of Integrated Circuits

by

STEFAN P. HAU-RIEGE

Submitted to the Department of Materials Science and Engineering on
April 28, 2000 in partial fulfillment of the requirements for the degree of
Doctor of Philosophy in Electronic Materials

ABSTRACT

The stringent performance and reliability demands that will accompany the development of next-generation circuits and new metallization technologies will require new and more accurate means of assessing interconnect reliability. Reliability assessments based on conventional methodologies are flawed in a number of very important ways, including the disregard of the effects of complex interconnect geometries on reliability. New models, simulations and experimental methodologies are required for the development of tools for circuit-level and process-sensitive reliability assessments.

Most modeling and experimental characterization of interconnect reliability has focused on simple straight lines terminating at pads or vias. However, laid-out integrated circuits usually have many interconnects with junctions and wide-to-narrow transitions. In carrying out circuit-level reliability assessments it is important to be able to assess the reliability of these more complex shapes, generally referred to as "trees". An interconnect tree consists of continuously connected high-conductivity metal within one layer of metallization. Trees terminate at diffusion barriers at vias and contacts, and, in the general case, can have more than one terminating branch when the tree includes junctions.

We have extended the understanding of "immortality" demonstrated and analyzed for straight stud-to-stud lines, to trees of arbitrary complexity. We verified the concept of immortality in interconnect trees through experiments on simple tree structures. This leads to a hierarchical approach for identifying immortal trees for specific circuit layouts and models for operation. We suggest a computationally efficient and flexible strategy for assessment of the reliability of entire integrated circuits. The proposed hierarchical reliability analysis can provide reliability assessments *during* the design and layout process (Reliability Computer Aided Design, RCAD). Design rules are suggested based on calculations of the electromigration-induced development of inhomogeneous steady-state mechanical stress states. Failure of interconnects by void nucleation in single-layer-metallization, as well as failure by void growth in the presence of refractory metal shunt layers are taken into account. The proposed methodology identifies a large fraction of interconnect trees in a typical design as immune to electromigration-induced failure.

To complete a circuit-level-reliability analysis, it is also necessary to estimate the lifetimes of the mortal trees. We have developed simulation tools that allow modeling of stress evolution and failure in arbitrarily complex trees. We have demonstrated the validity of these models and simulations through comparisons with experiments on simple trees, such as "L"- and "T"-shaped trees with different current configurations.

Because analyses made using simulations are computationally intensive, simulations should be used for analysis of the least reliable trees. The reliability of the majority of the mortal trees can be assessed using a conservative default model based on nodal reliability analyses for the assessment of electromigration-limited reliability of interconnect trees. The lifetimes of the nodes are calculated by estimating the times for void nucleation, void growth to failure, and formation of extrusions. The differences between straight stud-to-stud lines and interconnect trees are studied by investigating the effects of passive and active reservoirs on electromigration. Models and simulations were validated through comparisons with experiments on simple tree structures, such as lines broken into two limbs with different currents in each limb. Models, simulations and experimental results on the reliability of interconnect trees are shown to yield mutually consistent results.

Taken together, the results from this research have provided the basis for the development of the first RCAD tool capable of accurate circuit-level, processing-sensitive and layout-specific reliability analyses.

Thesis Supervisor: Carl V. Thompson
Stavros Salapatas Professor of Materials Science & Engineering

Table of Contents

List of Illustrations and Figures

List of Tables

Acknowledgments

There is a countless number of people who supported and assisted me in the completion of this work. First of all, I would like to thank Professor Carl Thompson for the inspiration, help, and guidance in this project. I also thank my thesis committee members, Professor Samuel Allen and Professor Subra Suresh. I would like to thank Professor Harold Frost and Professor Eduard Arzt for suggestions and advice, and Professor Ilan Blech for useful discussions at the beginning of my stay at MIT.

I am grateful to Dr. Michael Thomas and to the Advanced Technology Group of National Semiconductor for their support, processing of samples, and access to equipment for electromigration testing. I am also grateful to Dr. Joe Clement for his help and collaboration in developing the filtering algorithms. I thank Professor Don Troxel and Yonald Cheri for their collaboration in developing a prototype for ERNI.

I would like to thank the members of the Thin Film Dielectric and Metal Research Department and especially Dr. Judy Prybyla and Dr. Walter Brown for introducing me to electromigration during my stay at Bell Laboratories.

I thank the staff at the Microsystems Technology Laboratories and the Center for Materials Science and Engineering's Electron Microscopy Shared Experimental Facility.

Thanks also to all members of the Submicron Materials Group for their assistance and companionship, including Aaron, Brett; Felix, Mauro, Ram, Rob, Srikar, Steve, Vab, Walid, and Young-Joon. Thanks to Robert and Morgan for helping me overcome all the administrative hurdles at M.I.T.

I express thanks to my family for their support and encouragement. Graduate school if full of novel ideas, insight, and excitement. However, my greatest finding was the love for my wife Christine. I would like to thank her for her love, encouragement, and scientific insights throughout the course of this work, making it a meaningful experience with lots of fun and great memories.

This work was supported through the Semiconductor Research Society Graduate Fellowship Program.

Chapter 1 Introduction

The microelectronics industry has undeniable significance today. Both the transistor, which is now more than five decades old [Bar 48], and the one decade younger integrated circuit (IC) were radical developments which became part of revolutionary new technologies. Semiconductor-based junction transistors began to replace vacuum tubes in a growing number of electronic devices used for electronic computers and for a variety of military devices. The small-size requirements for transistors and the need for improved reliability at low cost led to the drive to miniaturize components and devices. Eventually, Texas Instruments produced the first, though crude, semiconductor integrated circuit in 1958. The first successful model of a "micro-programmable computer on a single chip", the 4004 microprocessor, was introduced by Intel in 1971. The subsequent history of IC's is punctuated by landmark decreases in physical size, dramatic increases in the number of components on a single chip, and the commercialization of crucial new devices embodying ever more functions on a single chip [Mor 99].

The vast majority of IC's are fabricated using single-crystal silicon wafers as the starting material. Active and passive devices are fabricated inside the silicon. Subsequently, the devices are electrically isolated by growth or deposition of a dielectric layer. The electrical terminals of the devices are connected by etching contact holes into the dielectric, and fabricating narrow metal lines terminating at the contact holes. These metal lines are referred to as *interconnects*. The process is repeated for several layers of metallization, and is referred to as the back-end process of the line (BEOL).

In today's Si VLSI technology, many meters of metal interconnects are required to build a single IC, such that millions of metal segments exist in each IC. Interconnects typically operate at a temperature of approximately 100°C, and carry a maximum current density of about 5×10^5 A/cm^2. In addition, thermal cycling during fabrication and operation can lead to large thermal-mismatch stresses inside the lines. Under these conditions, interconnects are susceptible to damage and, eventually, failure, mainly due to electromigration [Ble 67]. The reliability concern increases with the level of integration, where each new generation of Si technology requires the use of a larger number of narrower interconnects, stressed at higher current densities.

1.1 Electromigration

Electromigration is electronic-current-induced diffusion due to an electron wind force on metal-atoms. This force originates from scattering events with flowing electrons, whereby atoms migrate via a vacancy exchange mechanism. Electromigration is process-dependent, layout-specific, and materials-dependent.

The current metallization systems used in VLSI are based on pure Al, Al doped with a few weight percent of Cu, pure Cu, or alloyed Cu. In Al, the Cu-dopant has proven to improve the reliability of interconnects [Gan 75, Ble 77]. More recently, we are experiencing an important shift from the Al(Cu) alloy system to pure Cu and alloyed Cu because Cu has a lower electrical bulk resistivity and an anticipated higher resistance to electromigration-induced failure [Ede 97]. In all four materials systems, electromigration phenomena are observed and are a major reliability concern.

1.2 1D Model for Electromigration

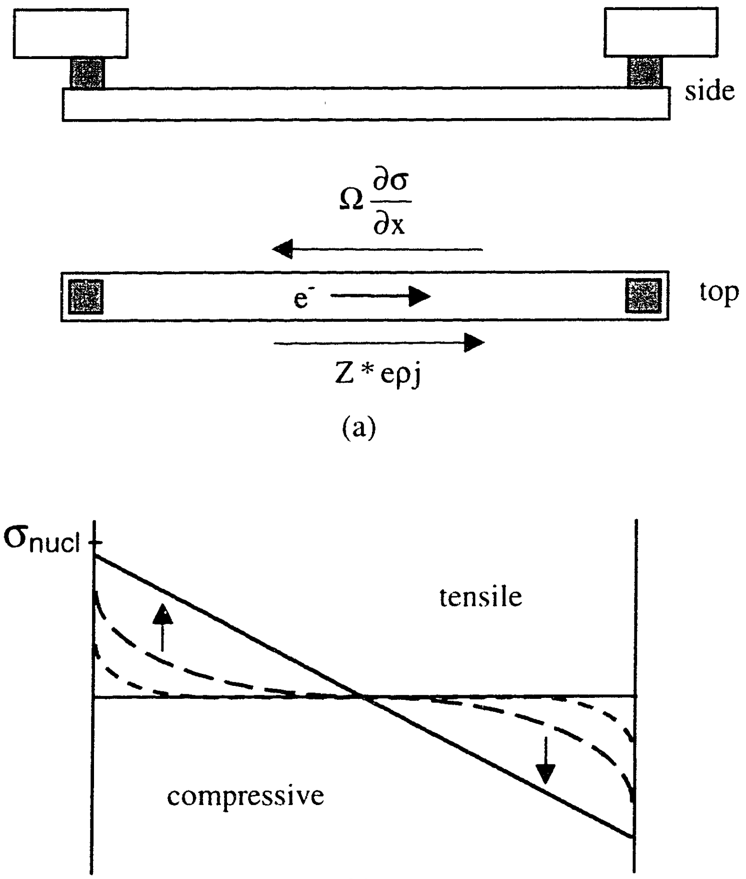
A number of theoretical formulations have been developed to describe electromigration [Sha 86, Kir 91, Kor 93, Cle 95, Par 97, Par 99]. We base our reliability analyses on the stress evolution model described by Korhonen *et al.* [Kor 93]. For the computer simulations of electromigration, we have used an extended Korhonen model by Park *et al.* [Par 97, Par 99], which accounts for the stress and alloy-concentration dependence of the diffusivity, as well as for chemical driving forces due to gradients in the concentration of alloys.

1.2.1 Korhonen Model

During electromigration, momentum is transferred from the electrons to the atoms of the conductor causing a biased atomic self-diffusion in the direction of the electron transport. The "electron wind" applies a force that can be expressed as

$$\vec{F}_e = -\frac{D}{kT} q^* \vec{E} = -\frac{D}{kT} Z^* e \rho \vec{j}, \tag{1.1}$$

where D is the atomic diffusivity, k is Boltzmann's constant, T is the temperature, $\vec{E} = \rho\vec{j}$ is the electric field, ρ is the electrical resistivity, and $\vec{j}$ is the current density. $q^* = Z^* e$ is the effective atomic charge, Z^* is the effective atomic charge number, and e is the fundamental charge. The magnitude of Z^* depends on two forces, which are the momentum transfer from the electrons to the atoms [Hun 75], and the electrostatic force



(a)

(b)

Figure 1.1: Illustrations of the origins of immortality of straight stud-to-stud lines without void nucleation, in which steady-state stress distributions develop without failure. (a) Side and top views of an interconnect, with the directions of the electron flow, electron wind force, and back-stress indicated. (b) Stress as a function of location for different times. The line will be immortal if the stress stays below the critical stress necessary for void nucleation, σ_{nucl}.

due to the applied electric field [Kin 96]. With a few exceptions, the electron wind dominates in most metals [She 89]. Empirically determined values of Z^* in Al and Al-alloy interconnects range from 1 to 100.

As atoms electromigrate, they accumulate down-wind, and are depleted up-wind. This redistribution of atoms leads to local changes in mechanical stress. The stresses become more compressive where atoms accumulate, and more tensile where atoms deplete. The resulting stress gradient leads to a gradient in chemical potential, which leads to a driving force (called the back-stress) opposing electromigration. The directions of electron flow, electron wind force, and back-stress are shown in figure 1.1 (a). The net atomic flux can be expressed as

$$\vec{J} = \frac{cD}{kT}\left(\nabla\mu + q^{*}\vec{E}\right), \tag{1.2}$$

where c is the concentration of the diffusing species. The chemical potential, μ, depends on the hydrostatic stress, σ, as $\mu = \mu_0 - \Omega\sigma$ [Her 50], where Ω is the atomic volume. The stress is tensile for $\sigma > 0$, and compressive for $\sigma < 0$. In the line direction, the atomic flux can be written as

$$J = \frac{cD}{kT}\left(-\Omega\frac{\partial\sigma}{\partial x} + q^{*}E\right), \tag{1.3}$$

where the distance x is measured along the length of the line. The continuity equation in the line direction gives the material balance as

$$-\frac{\partial J}{\partial x} = \frac{\partial c_v}{\partial t} - \frac{\partial c}{\partial t}. \tag{1.4}$$

In confined metal lines, the relative density change dc/c corresponds to an increment in stress, $\partial\sigma$, as

$$\frac{dc}{c} = -\frac{\partial\sigma}{B}, \tag{1.5}$$

where B is the applicable modulus [Kor 93]. Assuming that the vacancies are in equilibrium with the stress, the vacancy concentration can be written as [Bal 79, Kor 93]

$$c_v = c_{v,0} \exp\left(\frac{\Omega\sigma}{kT}\right). \tag{1.6}$$

$c_{v,0}$ is the vacancy concentration in the absence of stress. The continuity equation (1.4) becomes [Kor 93]

$$\left(\frac{B\Omega}{kT}\frac{c_v}{c}+1\right)\frac{c}{B}\frac{\partial\sigma}{\partial t}+\frac{\Omega}{kT}\frac{\partial}{\partial x}\left(c_v D_v\left(\frac{\partial\sigma}{\partial x}-\frac{q^*}{\Omega}E\right)\right)=0. \tag{1.7}$$

The electromigration-induced stress evolution in interconnects can be calculated using the Korhonen model. Figure 1.1 (b) shows the stress evolution in an interconnect which has an atomic diffusivity that is constant along the length of the line.

1.2.2 Blech Length

If void nucleation does not occur, the stress along the length of a line, bound by atomic diffusion barriers, will evolve until there is a uniform stress gradient, as shown in figure 1.1 (b), and the back force due to this gradient balances the electron wind force [Ble 76]. Under this condition, J = 0, and with equation (1.3),

$$Z^{*}e\rho\, j = \Omega \frac{\Delta\sigma_{\max}}{L}, \tag{1.8}$$

where L is the length of the line, and $\Delta\sigma_{max}$ is the difference in the stress at the anode and the cathode. The force-balanced stress gradient in the line is

$$\text{slope} = \frac{Z^{*}e\rho\, j}{\Omega}. \tag{1.9}$$

The maximum stress in the line is therefore a function of the product of the current density, j, and the line length, l. If the stress required for void nucleation is less than the maximum stress, the line will not fail and will be immortal. There is a critical line-length current-density product that defines the condition for immortality,

$$jL < \left(\frac{\Omega\Delta\sigma_{crit}}{Z^{*}e\rho\, j}\right) \equiv (jL)_{cnt}, \tag{1.10}$$

where $\Delta\sigma_{crit}$ is the minimum stress difference that leads to void nucleation. Short lines, and/or lines stressed at low currents, are more likely to be immortal. For a given current density, the critical line length defining immortality is called the *Blech length*, named after Ilan Blech who first reported experimental evidence for short-line effects [Ble 76].

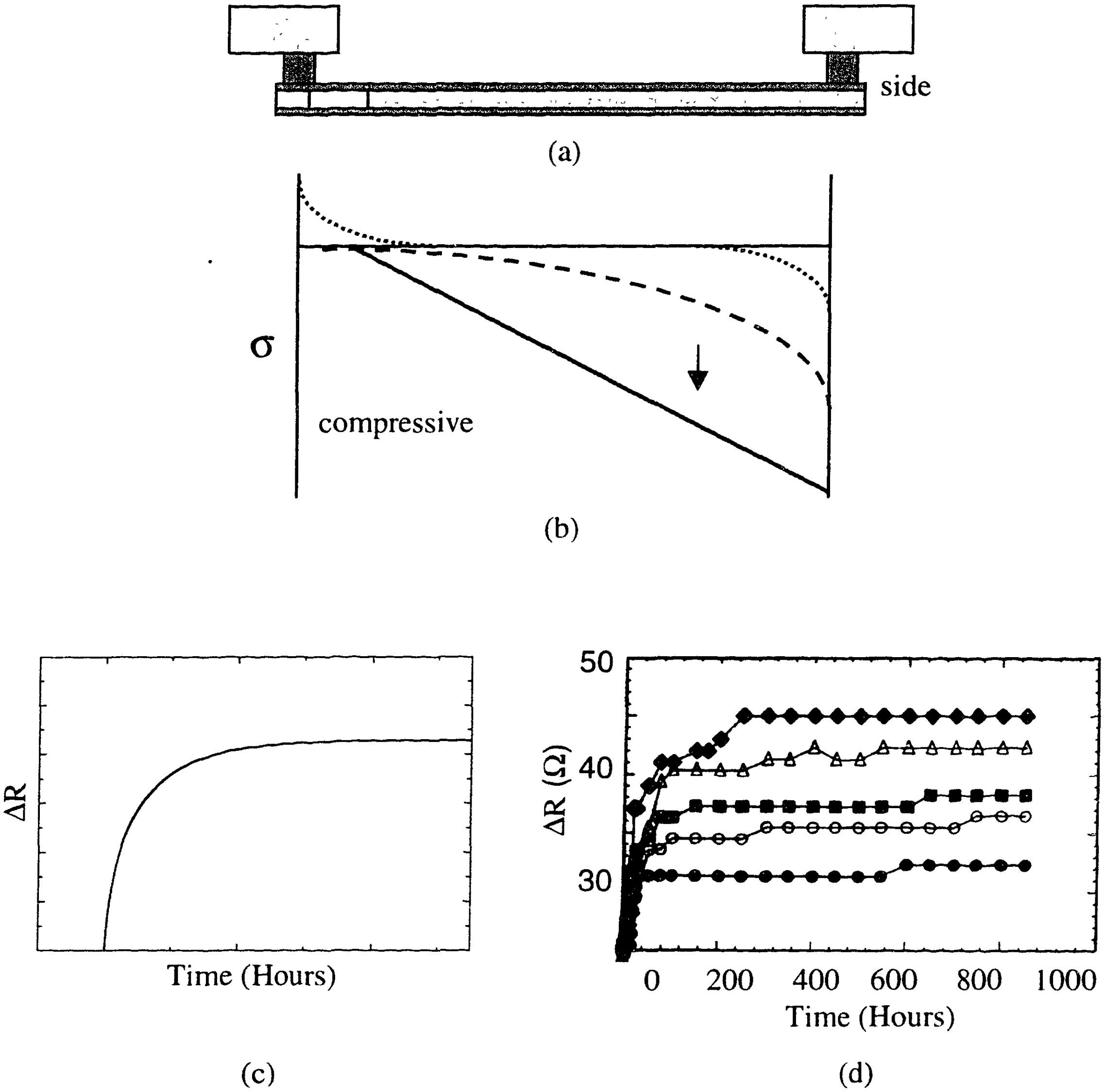


Figure 1.2: Illustrations of the origins of immortality in straight stud-to-stud lines with void nucleation, in which the line shunts current, and in which void growth saturates. (a) Side-view of the interconnect. (b) Stress as a function of location for different times after void nucleation. (c) Resistance increase due to void growth. (d) Experimentally observed resistance increases as reported in [Fil 96]. At longer times, void growth saturates, and the resistance ceases to increase.

If void nucleation *does* occur in lines with shunt layers, as shown in figure 1.2 (a), the stress at the void surface will fall to zero, and the stress in the nearby metal will quickly decrease as the void grows, as shown in figure 1.2 (b). The electron current flows through the shunt layers which do not electromigrate. This leads to a resistance increase as shown in figure 1.2 (c). The void grows until, eventually, a force balance will develop where equations (1.8) and (1.9) again apply, but the maximum stress is compressive and has twice the magnitude as in the case without void nucleation. If this maximum compressive stress does not cause yielding or fracture of the dielectric, the line will still be immortal even though a void has formed. The immortality condition is then that $\Delta\sigma_{max} < \Delta\sigma_{crit}$ for dielectric failure due to compressive stress in the metal [FIL 96, SUO 98], and a line will be immortal if

$$(jL) < \frac{\varrho/_{A}}{\varrho_{l}/_{A_{l}}} \frac{\Delta R_{max}}{R} \frac{2\Omega B}{e\varrho z^{*}}, \qquad (1.11)$$

where ρ and ρ_l are the resistivity of the high-conductivity metal and the shunt layers, respectively. R is the initial resistance of the line, and ΔR_{max} is the maximum acceptable resistance increase of the line. Line-length and current-density dependent saturation of the resistance increase has been demonstrated in experiments by Filippi *et al.* [Fil 95]. The results of their experiments are reproduced in figure 1.2 (d).

1.3 Factors Affecting Electromigration

The reliability of interconnects can be altered and enhanced in a number of ways. The effects of microstructure, alloys, passivation layers, multilayer metallizations including refractory-metal layers, and the effects of presence of contacts and vias are discussed below.

1.3.1 Microstructure

The product of the effective atomic diffusivity, D_{eff}, and the effective charge number, Z^*_{eff}, depends on the details of the microstructure of the interconnect. Diffusion can occur within the bulk material (subscript B), the line's top (subscript ST) and bottom (subscript SB) surface, the side walls (subscript SW), and the grain boundaries (subscript GB). Diffusion along dislocation cores has proven to be insignificant [Sri 98]. In general, each transport path has its own effective charge number, Z, and diffusivity, D. Assuming the transport paths are independent of each other, the product of Z^*_{eff} and D_{eff} can be written as

$$\begin{aligned} Z^*_{eff} D_{eff} &= Z^*_B n_B D_B + Z^*_{ST} D_{ST} \delta_{ST} / h + Z^*_{SB} D_{SB} \delta_{SB} / h \\ &+ 2Z^*_{SW} D_{SW} \delta_{SW} / w + Z^*_{GB} D_{GB} (\delta_{GB} / w)(1 - d / w), \end{aligned} \qquad (1.12)$$

where δ_{ST}, δ_{SB}, δ_{SW}, and δ_{GB} denote the width of the top surface, bottom surface, sidewall surfaces, and grain boundaries, respectively [Hu 99a]. d is the average grain size, w is the line width, and h is the line thickness. n_B, δ_{ST}/h, δ_{SB}/h, δ_{SW}/w, and $(\delta_{GB}/w)(1\text{-}d/w)$ are the fraction of atoms diffusing through the bulk, top surface, bottom surface, sidewall surfaces, and grain boundaries, respectively.

With decreasing line width, the microstructure changes from polygranular to bamboo for a given grain size [Cho 89]. For aluminum-based metallization schemes, polygranular lines have a continuous network of high-diffusivity grain boundaries along their length, which dominates the effective diffusivity [Kin 80]. However, in bamboo interconnects the grain boundaries are perpendicular to the electron flow, and transgranular transport becomes dominant, which is slower and leads to longer times to failure [Iye 84].

1.3.2 Alloys

It has been shown that the reliability of Al-based polygranular interconnects can increase by over an order of magnitude due to the presence of Cu as an alloying addition to Al [Gan 75, Ble 77]. In Al(Cu) alloys, copper atoms segregate to the grain boundaries and slow the diffusion of aluminum along the otherwise high-diffusivity grain boundaries. The reduced Al diffusivity slows the buildup of stress and void growth, and therefore increases the reliability of the interconnect. Several models have been proposed to account for the reduced diffusivity. In the data analyses, we have used the trapping model proposed by Rosenberg [Ros 72] based on the assumption of copper-vacancy trapping at the grain boundaries. While the addition of Cu affects the diffusion along grain boundaries, it does not affect the diffusion of Al along side walls, or top and bottom interfaces of interconnects [And 99].

1.3.3 Passivation and Interlevel Dielectrics

In integrated circuits, metal interconnects are surrounded by an interlevel dielectric, such as SiO_2, Si_3N_4, or other materials with a lower dielectric constant, to reduce signal delays. Dieletrics enhance interconnect reliability by suppressing the formation of metallic extrusions, which would otherwise possibly lead to short circuit. The dielectric's mechanical properties determine the relationship between a change in metal concentration due to electromigration and the corresponding change in mechanical stress. This is described by an effective modulus, B, which was introduced in equation (1.5). For softer dielectrics, metal-concentration changes lead to smaller stress changes than in the case of harder dielectrics. A dielectric with a high thermal conductivity also counteracts the temperature rise due to Joule heating of interconnects, which can further enhance electromigration resistance.

However, dielectrics can also be detrimental to reliability, because a large thermal stress can build up during deposition, possibly leading to catastrophic stress voiding. For example, SiO_2 is typically deposited at elevated temperatures, and since Al and Cu have a much larger coefficients of thermal expansion than Si and SiO_2, large thermal stresses can develop during cooling, and be present in the metal at service and test temperatures.

1.3.4 Multilayer Metallization

Modern interconnects also have refractory metal layers along their lengths, and these are not subject to significant electromigration even at testing conditions. Al-alloy interconnects usually have Ti, TiN and/or Al_3Ti under- and over-layers. Cu-based interconnects are surrounded on at least three sides by refractory metal liners that serve as diffusion barriers. When electromigration leads to voiding in these interconnects, current can shunt around the voids through the refractory metal layers so that even voids which span the full width and thickness of interconnects do not cause open-circuit failure. Instead, voiding leads to a resistance increase, and the resistance continues to increase as voids grow.

1.3.5 Contacts and Vias

In Al-alloy technologies, vias are usually filled with W, and in Cu-based technologies, there are refractory metal layers, which do not electromigrate, at the base of the vias. In both cases, the line ends form perfect blocking boundaries for the atomic flux. Vias and contacts therefore provide sites of flux divergence, so that when voids form, they generally nucleate and grow at vias and contacts. In recent years, testing of straight lines terminating at refractory-metal-filled vias has become more common. Narrow Al-alloy interconnects which have been annealed after patterning usually have bamboo grain structures without microstructural heterogeneities, so that their lifetimes are long. Refractory-metal-filled vias provide the sites at which an atomic flux divergence leads to the high tensile stress that causes voiding (at cathode vias) or the high compressive stress that leads to dielectric cracking and metal extrusion. This is true in actual circuits, and so it is appropriate that lines terminating in vias be tested, rather than lines terminating at pads [Sch 87].

1.4 Reliability Statistics

Electromigration-induced failure times are stochastically distributed due to the stochastic nature of the features leading to and determining electromigration, such as grain structure, texture, and alloy distribution. In the following, the basic concepts of reliability statistics are described, and the normal and lognormal distribution functions used in the data analysis are discussed in further detail. Finally, scaling of reliability data is discussed.

1.4.1 Reliability Concepts

The mathematical concepts for reliability measures appropriate to nonmaintained systems starting to operate at time t = 0 will now be summarized. The probability density function (pdf) is represented by f(t). $f(t)\cdot dt$ is the probability that a failure will occur between time t and time t+dt in an ensemble of N identical components. As a pdf, f has to be defined such that, for all times $t \geq 0$, $f \geq 0$ and $\int_0^\infty f(t)dt = 1$. f has the units $[\text{time}]^{-1}$.

The mean time to failure, τ_{av}, is defined by

$$\tau_{av} = \int_0^\infty t \cdot f(t)dt\,. \tag{1.13}$$

The unitless cumulative distribution function, F(t), is the fraction of components failed at or before time t, and relates to the pdf by

$$F(t) = \int_0^t f(t')dt'\,. \tag{1.14}$$

The survivor function S(t) gives the probability of surviving until time t without failure, so $S(t) = 1-F(t)$. The failure rate, $\lambda(t)$, is the instantaneous rate of failure for units of a population that have survived up to time t, so that $\lambda(t)\cdot dt$ is the conditional

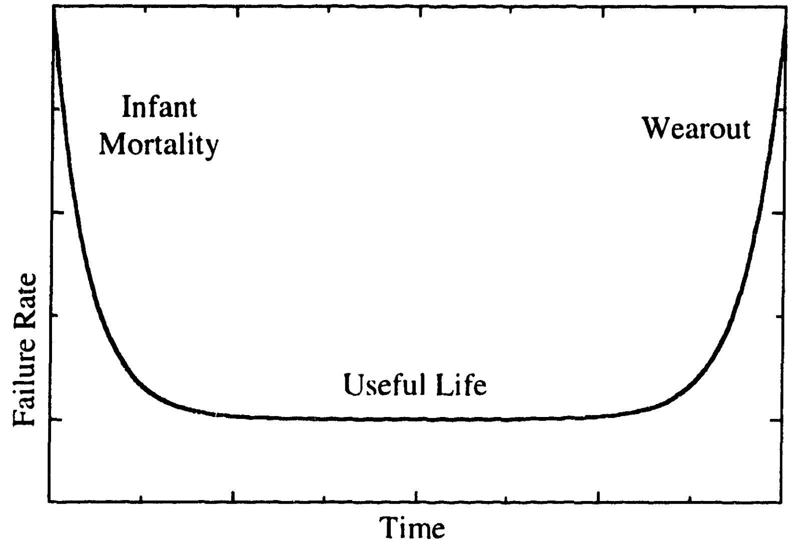


Figure 1.3: Typical "bathtub" distribution of failures.

probability that a component which survived until time t will fail within an additional time, dt. It is related to the cumulative distribution function by

$$\lambda(t)=\frac{f(t)}{1-F(t)}=-\frac{d}{dt}\ln(S(t)). \qquad (1.15)$$

The failure rate is typically given in units of FIT's (failure units), defined as one failure in 10^9 device-hours. For small times, $F(t) << 1$, and $f(t) \cong \lambda(t)$.

There are usually three components to the reliability lifetime of an integrated circuit: infant mortality, useful life, and wearout. Figure 1.3 shows a schematic of the failure rate as a function of time. The shape of this curve is generally referred to as a *bathtub curve*. The infant mortality period is the period of early failures, which are typically caused by manufacturing and assembly defects. To identify and eliminate devices with early failures, they are subjected to burn-in testing. During the useful life period, the failure rate is very low and constant, and after this, wearout mechanisms degrade the reliability of the device. A major failure mechanism contributing to wearout of IC's is electromigration. To focus our experiments on studying the electromigration effects on interconnects rather than defects due to fabrication, we screened the samples before testing to eliminate samples with unusually high or low resistances. This way, we did not observe unusually early failures.

1.4.2 Failure Distributions

The statistical distribution of electromigration test failures has been found in many studies to approximate a lognormal distribution [Att 71], even though there is no a priori reason that the failure times should be lognormally distributed [Att 71, LaC 86]. The ensemble size in typical tests is less than 60 DUT's (devices under tests), so that it is generally not possible to distinguish between a lognormal distribution and an extreme-value distribution [Sch 80]. However, LaCombe et al. [LaC 86] verified that the times-to-failure are lognormally distributed down to about 0.3% of cumulative failures. The ensemble sizes in our experiments are smaller than 30 DUT's, so the lifetimes can be described by either distribution fairly well. We will follow the practice of the general community and fit our data to a lognormal distribution function. The lognormal distribution can be derived from the normal distribution, both of which are described below.

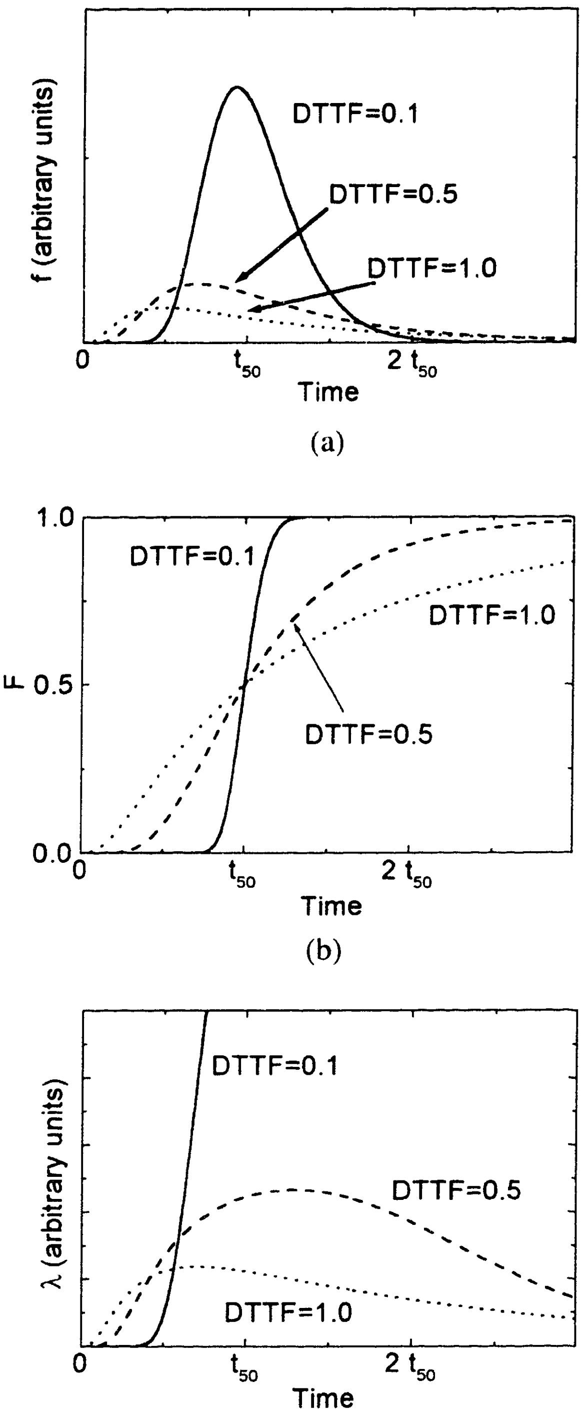


Figure 1.4: The lognormal distribution for several different deviations in the time to failure, DTTF. (a) Probability density function, f. (b) The cumulative distribution function, F. (c) The failure rate, λ.

1.4.2.1 The Normal Distribution

A random variable X is *normally distributed* with the mean μ ($-\infty < \mu < \infty$) and variance σ^2 ($\sigma > 0$) if the pdf of X is given by

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \qquad (-\infty<x<\infty). \qquad (1.16)$$

μ is the mean and the median, and σ is the standard deviation of the normal distribution. The cumulative distribution function of the normal distribution is given by

$$F(x) = \Phi\left(\frac{x-\mu}{\sigma}\right), \qquad (1.17)$$

where the cumulative distribution function of a standard normal random variable, Φ, is defined by

$$\Phi(z) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{z} \exp\left(-\frac{u^2}{2}\right) \cdot du = \frac{1}{2}\left(\operatorname{erf}\left(\frac{z}{\sqrt{2}}\right) + 1\right), \qquad (1.18)$$

where erf is the error function [Abr 74].

1.4.2.2 The Lognormal Distribution

When a random variable X is normally distributed with mean μ and variance σ^2, then $T = e^X$ is said to have a *lognormal distribution*, i.e. ln(T) is normally distributed with mean μ and variance σ^2. The pdf for lognormal distribution is

$$f(t) = \frac{1}{\sqrt{2\pi}\sigma t}\exp\left(-\left(\frac{1}{\sigma\sqrt{2}}\ln\left(\frac{t}{t_{50}}\right)\right)^2\right). \qquad (1.19)$$

$t_{50} = \exp(\mu)$ is median time to failure (MTTF), and σ is the standard deviation in the log of failure times (DTTF). Figure 1.4 (a) shows the pdf for different DTTF's. The lognormal cumulative distribution function can be written as

$$F(t) = \int_0^t f(t')dt' = \Phi\left(\frac{\ln(t) - \ln(t_{50})}{\sigma}\right) = \frac{1}{2}\left(\operatorname{erf}\left(\frac{\ln(t/t_{50})}{\sqrt{2}\sigma}\right) + 1\right), \qquad (1.20)$$

where $\Phi(t)$ is the cumulative distribution function of a standard normal random distribution. F is sketched in Figure 1.4 (b), and the failure rate, λ, is shown in Figure 1.4 (c). λ vanishes for t=0, then increases with increasing times, reaches a maximum, and decreases again approaching zero for $t \to \infty$.

1.4.3 Accelerated Life Testing

A typical IC can have more than a kilometer of total metallic interconnect length, so the interconnects must be *very* reliable. In order to observe electromigration wearout failures in a small ensemble of lines within a reasonable amount of time (typically under a thousand hours), special test conditions, such as the use of much higher current densities and temperatures than would be experienced in service conditions, are required to decrease the time-to-failure in a predictable and understandable way. These accelerated tests must be extrapolated back to service conditions, so that estimates of the reliability at normal operating conditions can be made. The basic equation which is still generally utilized to describe electromigration-induced failure, and to analyze results from accelerated tests, is a result of classic work by Black from the mid-1960's [Bla 67].

$$t_{50} = Aj^{-n} \exp\left(\frac{E_a}{kT}\right). \tag{1.21}$$

A is a temperature- and current-density-independent constant, j is the current density, k is Boltzmann's constant, T is the temperature, n is the current density exponent, and E_a is the activation energy. The validity of Black's equation over a wide range of current densities and temperatures, from a practical point of view, can not be proven, and scaling results should be taken as a rough estimate at best.

It has been shown that a current density exponent of n = 2 is consistent with void-nucleation-limited failures in models based on the Korhonen analysis [Kor 93, Kno 98]. The rate of unconstrained void growth is proportional to j, so that the current density exponent for void-growth-limited failure is expected to be 1. When Black's equation is used to extrapolate test results to service conditions, it is implicitly assumed that the failure mechanism does not change through the range of extrapolation. However, through use of simulations, this has been shown to not be the case [Par 99]. Failure maps obtained through computer simulation of electromigration as described in the next section indicate which different failure mechanisms are expected as a function of current density and line length [And 99]. Such maps can be readily used to estimate the reliability and current-density-scaling behavior of interconnects.

1.5 Numerical Simulation of Electromigration

A software tool for structure-sensitive simulation of electromigration-induced stresses and electromigration-induced failure of straight-line interconnects, MIT/EmSim, has been developed at MIT [Par 99, Ems]. The tool is a forward-Euler finite-element solver based on the 1D Korhonen equation (1.7), and in addition considers the stress-dependence of the atomic diffusivity [Par 97], and the effects of alloys on electromigration [Par 99]. The tool generates grain structures with known statistical variations as a function of median grain size and line width, simulates stress and void evolution in the interconnect, and predicts failure statistics as a function of various failure criteria, and as a function of the current density and temperature. MIT/EmSim has been used to simulate electromigration-induced failure in pure Al and Al-Cu interconnects, and in pure Cu and Cu alloy interconnects [And 99].

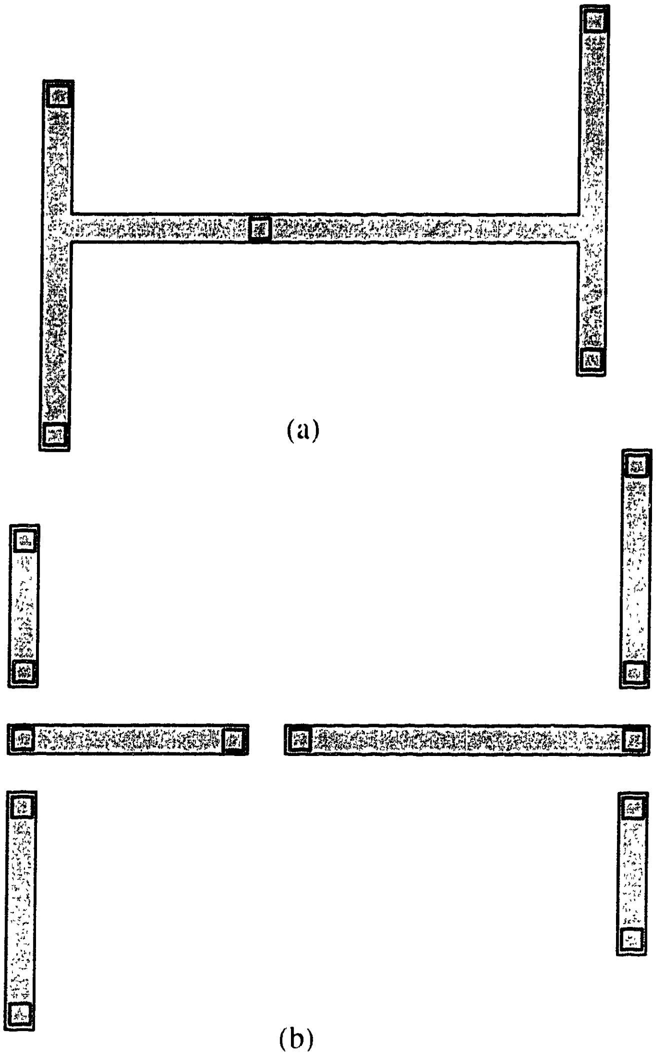


Figure 1.5: (a) Example of an interconnect tree, which is a continuously connected piece of metal within one layer of metallization, is bound by diffusion barriers. The stress evolution during electromigration testing or service is a complex function of the current configuration, line lengths, and connectedness. (b) An interconnect tree broken up into segments. In conventional reliability assessment approaches, the reliability of the segments are (incorrectly) independently estimated.

1.6 Interconnect Trees

Modeling and experimental characterization of interconnect reliability is usually focused on simple straight lines terminating at pads or vias, while laid-out integrated circuits often have interconnects with junctions. For carrying out circuit-level reliability assessments, it is necessary to assess the reliability of these more complex shapes, generally referred to as trees. In the context of this thesis, we identify the interconnect tree as the fundamental reliability unit. An *interconnect tree* consists of continuously connected high-conductivity metal within one layer of metallization [Rie 98a], which terminates at diffusion barriers such as vias and contacts, and, in the general case, can have more than one terminating branch when the tree includes junctions. An example of an interconnect tree is shown in figure 1.5 (a).

In conventional approaches, the reliability of a tree is assessed by breaking the tree up into segments, as shown in figure 1.5 (b), and estimating the reliability of each segment separately, using the results from straight stud-to-stud lines. This is generally inaccurate, and overly conservative at best, because the different parts of the tree are *not* independent of each other. Rather, material can diffusive freely within a tree, and the stress evolution in different parts of the tree is coupled. In order to estimate the reliability of interconnects correctly, the connectedness of the segments of a tree has to be accounted for [Hau 00b].

1.7 Circuit-Level Full-Chip Reliability Analyses

Models and techniques have been developed to estimate the reliability of interconnects in IC's. BERT (Berkeley Reliability Tool) is an example of a layout-level reliability estimator [Lie 90a], which is used to assess the performance and reliability of small test circuits. However, due to the computational complexity of the methods used, a full-chip reliability analysis was not achieved [Roc 98]. Instead, integrated circuits are currently designed using simple and conservative "design rules", typically based on

Black's equation (1.20) applied to line segments, to ensure that the resulting circuit will meet reliability goals. If an industry-typical maximum current density of $2\text{x}10^5$ A/cm^2 were assumed, and if this current density were passed through most interconnects, a chip dissipating kilowatts would be required [Roc 98], which is unrealistic and clearly overly conservative. This simplicity and conservatism leads to unnecessarily reduced performance for a given circuit and metallization technology.

A major research and development effort has been spent on obtaining more realistic reliability assessments. Two outcomes of this effort are ERNI and BERT. In the following section, we will describe the reliability analysis tool BERT, which performs electromigration-reliability estimates based on the analysis of individual line segments. The reliability analysis tool ERNI, which has been developed at MIT [Che 00], will then be described. ERNI is based on reliability analyses of full interconnect trees rather then individual line segments. In addition, ERNI is based on a hierarchical methodology for reliability analysis in order to keep the computation manageable, a concept which was developed in the context of the research to be described.

1.7.1 BERT

A circuit-level electromigration simulator (EM simulator) is part of the Berkeley Reliability Tool (BERT) package [Lie 90a]. The EM simulator can be operated in two modes [Lie 92]. In the first mode, the simulator generates guidelines for the width and length of each interconnect. In the second mode of operation, the simulator calculates the overall reliability of a given layout. The EM simulator is based on a reliability estimate

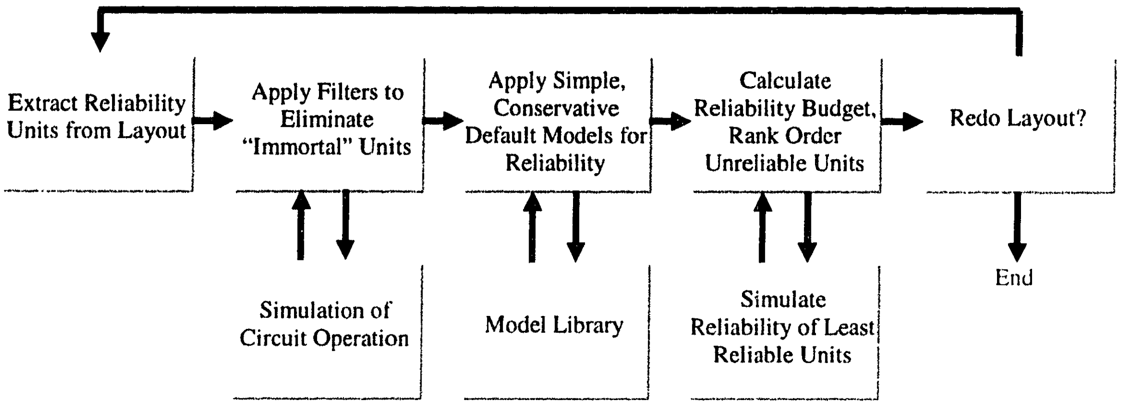

Extract Reliability Units from Layout
Apply Filters to Eliminate "Immortal" Units
Apply Simple, Conservative Default Models for Reliability
Calculate Reliability Budget, Rank Order Unreliable Units
Redo Layout?
Simulation of Circuit Operation
Model Library
Simulate Reliability of Least Reliable Units
End

Figure 1.6: A flow chart for a full hierarchical circuit-level reliability assessment, the basis for the prototype tool ERNI.

of individual line segments, as described in detail by Liew *et al.* [Lie 90b, Lie 92]. The EM simulator is linked to SPICE, a general-purpose circuit simulation program.

1.7.2 ERNI

To facilitate the development of IC designs with improved performance and high overall reliability by adjusting the design and fabrication process, we have developed a VLSI metal-interconnect-reliability prediction CAD tool, ERNI (Electromigration Reliability of Networked Interconnects) [Che 00], which allows process-sensitive and layout-specific reliability estimates for fully laid-out or partially laid-out integrated circuits. The performance is judged based on the results of a SPICE-level circuit simulator, whereas reliability is assessed using ERNI.

Figure 1.6 shows a flowchart describing the way in which ERNI operates. ERNI extracts the interconnects from the circuit layout, and identifies highly-reliable or immortal interconnects based on increasingly more complex reliability analyses. The less-reliable or mortal interconnects are ranked according to their reliabilities using a conservative default model for reliability estimation, and the reliability of the least reliable trees is assessed in detail using electromigration simulation tools. ERNI interacts with the computational tools GGSim [Wal 92, Fay 00] and MIT/EmSim [Ems, Par 97]. GGSim simulates the grain structure evolution in interconnects as a function of materials, processing, and geometry. MIT/EmSim predicts the electromigration-induced stress evolution and hence the reliability of interconnects based on the Korhonen model, as described in section 1.5.

ERNI is written in JAVA as a "plug in" client extension to MAJIC, which is a distributed VLSI layout editing tool written entirely in Java [Che 00]. MAJIC is based on MAGIC [Mag 85], a VLSI layout tool developed at UC Berkeley, but extends its functionality to support distributed design and collaboration. Users can view, edit, and share layouts via any Java-enabled World Wide Web browser.

The models and modeling methodologies that form the basis for the reliability assessment carried out by ERNI, were the primary results of the research described in this thesis.

1.8 Goals of the Thesis

The goal of this thesis is to provide modeling, simulation, and experimental background for ERNI. This includes gaining an understanding of the reliability of interconnect trees as opposed to simple, straight, stud-to-stud interconnects.

We have extended and experimentally verified the understanding of "immortality", which has been demonstrated and analyzed for straight stud-to-stud lines, to arbitrarily complex interconnect trees. Using the immortality concept for filtering algorithms that identify trees which are immune to electromigration failure leads to a hierarchical approach for reliability assessment. To complete a circuit-level-reliability analysis, it is also necessary to estimate the lifetimes of the mortal trees. We have extended our electromigration simulation tools to allow modeling of stress evolution and failure in arbitrarily complex trees. However, these simulations require intensive computation, and should only be used for the least reliable trees. For other mortal trees, a conservative default model is needed. We suggest a simple default model based on the analysis of individual nodes in trees. Default models, immortality filters, and simulations have been validated through comparison with electromigration experiments on basic tree structures, such as lines broken into two segments with different currents in each segment, as well as "L"- and "T"-shaped interconnects.

1.9 Organization of Thesis

In Chapter 2, we discuss the differences in reliability between straight, stud-to-stud interconnects and interconnect trees with active and passive reservoirs for metal atoms [Tho 99a, Hau 00b]. We have explored the effects of reservoirs through simulations of the stress evolution and voiding in trees, and through electromigration experiments on simple tree structures, such as "L"- and "T"-shaped interconnects. We have also performed electromigration experiments on straight lines with an additional via in the middle of the line, which allows to pass currents of different magnitudes and directions through the different limbs of the trees. A detailed description of the modifications of MIT/EmSim can be found in Appendix A.

In Chapter 3, we discuss immortality filters for identifying interconnect trees that are immune to electromigration-induced failure [Rie 98a, Cle 99]. We report on the effectiveness of the filtering algorithms by applying them to full microprocessor layouts. A hierarchical approach to circuit-level reliability assessment, based on the use of the filtering algorithms, is proposed. We have verified the concept of immortality in trees by performing electromigration experiments on saturation effects "L"-shaped interconnects [Hau 00d].

In Chapter 4, a default model for estimating the reliability of mortal interconnect trees based on nodal reliability analyses is presented [Hau 00b]. We found that the reliability of trees can be conservatively estimated by considering void-growth and void-nucleation limited failures at interconnect junctions, depending on geometry and current configuration. We have verified the default model through comparison with lifetime experiments on simple trees with junctions and different line widths stressed with different current-density configurations and at different temperatures [Hau 00c].

Finally, chapter 5 contains a summary of the results of the previous chapters, along with a discussion of their implications with regard to the design of IC's of improved performance and high overall reliability [Tho 99a, Tho 99b, Tho 00]. This chapter also contains suggestions for future research.

Research that was not directly related to the main goal of this thesis (which is the development and testing of models for tree-based circuit-level reliability analyses) has also been carried out, and is described in Appendices B through F. The results on modeling of the effects of changes in the mechanical properties of dielectric materials on electromigration is summarized in Appendix B [Hau 00e]. Electromigration depends on the microstructure and texture of interconnects [Kno 91], which, in electroplated Cu films, evolves at room temperature after deposition [Tom 85]. In Appendix C, results from in-situ TEM studies and a kinetic analysis of the "self-annealing" of Cu thin films are described [Hau 00a]. In Appendix D, we summarize calculations of the energy-minimizing texture resulting from annealing of Cu damascene structures [Hau 99]. Modeling the effects of particles on the grain structure evolution in thin films and the consequences for interconnect reliability are discussed in Appendices E and F [Rie 99, Rie 98b, Rie 98c]. Experimental and modeling work on grain growth in patterned Al thin films is described in Appendix F.

Chapter 2 Electromigration in Interconnect Trees

2.1 Background

Significant progress has been made in understanding the basic phenomenology observed in straight via-to-via lines, where quantitative predictions of interconnect lifetimes can be made as a function of current density, temperature, and interconnect microstructure. However, while straight via-to-via interconnects are common in interconnect circuits, they represent only a subset of interconnect configurations that appear in real integrated circuit layouts. More commonly, interconnects have junctions and narrow-to-wide transitions, and can have complexly connected straight segments. It is important to define a general interconnect unit which represents all possible layout configurations, and analyze the reliability of circuits through analyses of *these* reliability units. An interconnect "tree" provides such a unit [Rie 98a], where a tree is defined as continuously connected high conductivity metal, within one layer of metallization, bound by contacts or vias filled with diffusion barriers (see figure 1.5 (a)). Interconnect trees can have any number of junctions. The common approach to assessment of the reliability of a tree is to break the tree up into segments, as shown in figure 1.5 (b), and estimating the reliability of each segment separately using the results from straight stud-to-stud lines. This approach is generally inaccurate, and overly conservative at best, because the different parts of the tree can *not* assumed to be independent of each other. Rather, material can diffuse freely within a tree, thereby coupling the stress evolution in different parts of the tree.

The simplest interconnect tree is a via-to-via line such as the one shown in figure 1.1 and as discussed above, and will be referred to as "I's". The next simplest tree is a straight line with vias at both ends and a third via elsewhere in the line, and will be referred to as "dotted I's". The "L" structure is elbow-shaped, and has vias at the ends and at the corner, and as we will see, behaves similarily to the "dotted I's". A tree with a "T" shape has vias at the three line ends and at the junction, and will be called "T's". "T's" have three straight interconnect segments or "limbs", where a limb is defined is a straight segment that has a via, contact or junction at both ends, with no other junctions or vias, contacts or junctions. The limbs of a tree can have different lengths, $L_{i.}$ and carry different current densities, j_i. In the general case a tree has n limbs.

The challenge addressed in this chapter is to extend the understanding developed for interconnect "I's" to an understanding of the reliability of general interconnect trees, by considering the failure kinetics of trees through simulations based on the Korhonen model and through electromigration experiments.

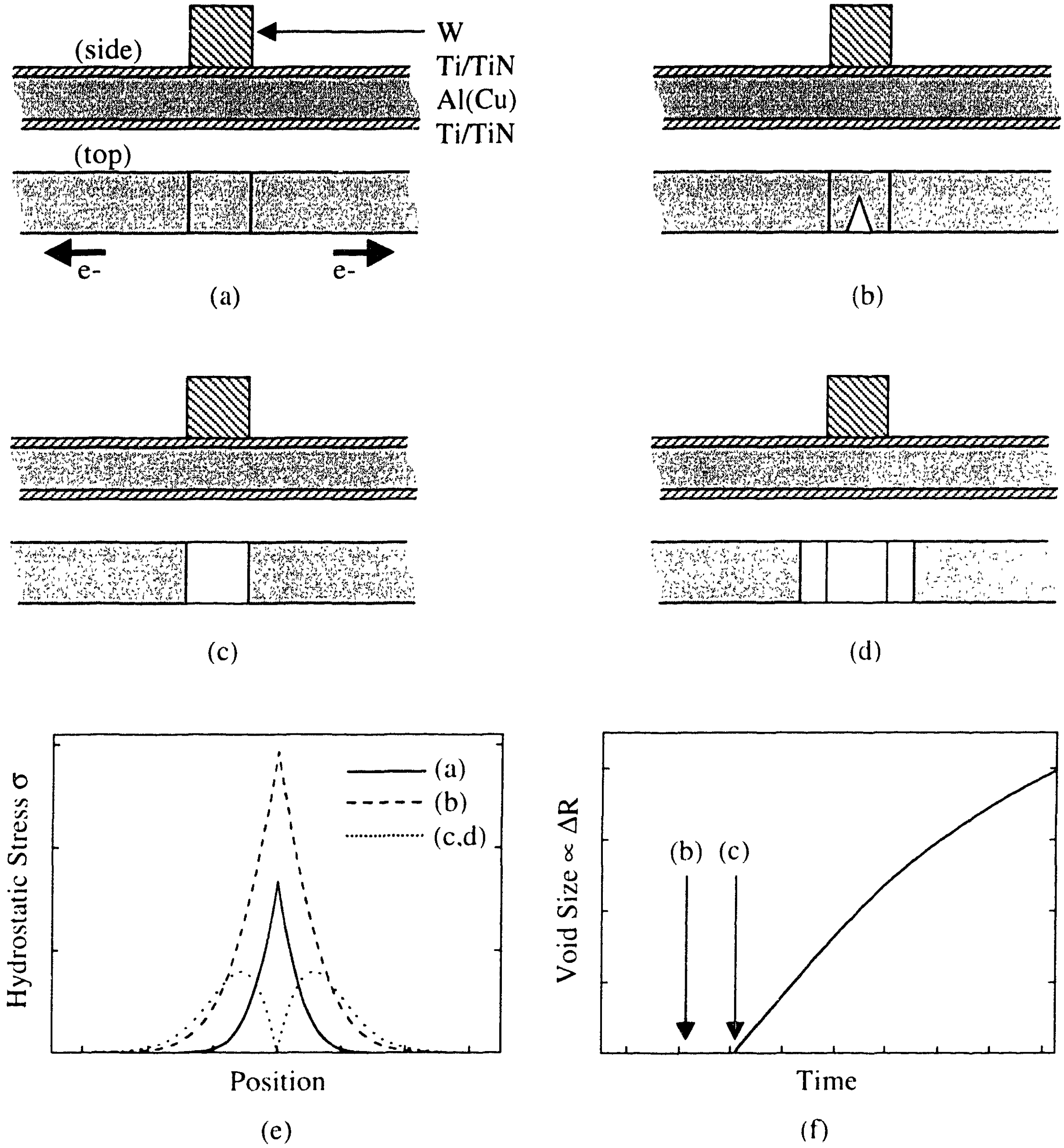


Figure 2.1: The Sequence of events leading to failure at a node with two active limbs. (a) A hydrostatic, tensile stress builds up in the line under the via. (b) If the tensile stress exceeds a threshold value, σ_{nucl}, a void nucleates and relieves the stress around the void. The void grows, but the electrical resistance of both lines has not increased significantly, yet. (c) Once the void passes the edge of the via, the via and the line are no longer connected by a high-conductivity path, as shown in (d), and the resistance of the lines increases. (e) Evolution of the stress profiles near the via. (f) Resistance increase, ΔR, as a function of time. The rate of resistance increase eventually decreases due to back stress effects.

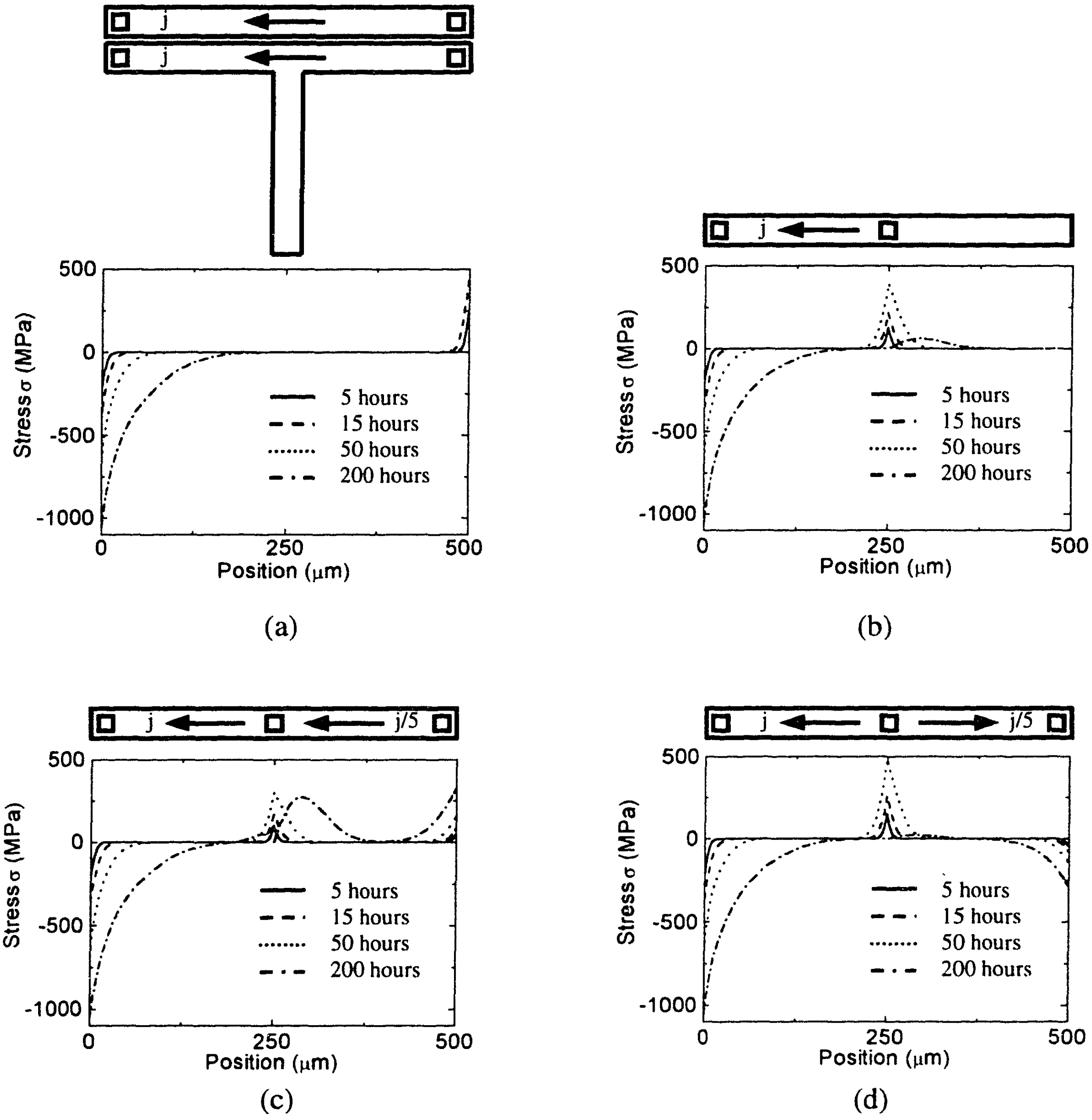
2.2 Computer Simulation of Electromigration in Trees

The Korhonen model [Kor 93] describing the electromigration-induced stress evolution in interconnects was originally proposed for straight-line interconnects. The model was extended to elbow-shaped interconnects as well as intersections of interconnects [Rie 98a, Cle 99]. Elbow-shaped interconnects and interconnect intersections can be treated in a similar manner to straight-line interconnects, because the hydrostatic stress along the length of a diffusion path (averaged over the line cross section) is considered. Experiments on simple tree structures of different shapes validate these assumptions, as described in section 2.4. The effects of bends and intersections can be captured by a spatially-varying effective elastic modulus, B. As shown in Appendix B, B can be determined by calculating the hydrostatic stress as a function of sets of homogeneous dilational free strains. Thermal stresses are similar to stresses induced by free strains in the x-, y-, and z-direction. It has been shown through finite element modeling that the volume-averaged hydrostatic component of thermal stresses in passivated "L"- and "T"-shaped interconnects varies by less than 15% along the interconnect [She 99]. Using this result in an analysis similar to the one shown in Appendix B, B also varies less than 15% along the interconnect, and therefore the effect of bends and intersections on electromigration is small. These results also show that bends and intersections affect the stress evolution only locally, and that these effects are neglegible considering the total interconnect length.

We simulated the effect of electromigration on interconnects using our electromigration simulator MIT/EmSim [Ems, Par 99]. MIT/EmSim is a 1D-electromigration simulation based on the Korhonen model [Kor 93]. When the diffusivities are input into MIT/EmSim, the atomic fluxes and stress evolution in the interconnect are calculated as a function of current density and temperature. We extended MIT/EmSim to allow the simulation of the effects of electromigration in interconnect trees. A typical failure scenario occurring at junctions in trees is shown in figure 2.1 (a) to (d). In this example, electrons are flowing from the top via into the interconnect to the left and to the right. A tensile stress peak develops at the junction, as shown in figure 2.1 (e). When the stress exceeds the critical stress necessary for void nucleation, a void forms, as shown in figure 2.1 (b), which relieves the stress around the void. The resistance of the lines, shown in figure 2.1 (f), does not immediately increase. Only after the void grows to a size large enough to isolate the via from the low-resistivity metal, as shown in figure 2.1 (c), does the resistance start to significantly increase. This happens because the current is forced to flow through the higher-resistivity shunt layer. Upon further stressing, the void grows and the resistance continues to increase as shown in figure 2.1 (f). Eventually, void growth slows down because of increasing back stress effects due to compressive stresses at the line ends. If a via is represented in the simulation by only a single junction cell, the exact void position under the via is not taken into account, which can lead to an overly optimistic reliability prediction. In this case, to be truly conservative, it is necessary to assume that voiding at the via leads to an immediate resistance increase in the connecting strands, which has been assumed in the following discussion. A detailed description of the modifications of MIT/EmSim can be found in Appendix A.

In the modeling and simulations for Al(Cu) described below, we used the following materials parameter, as described in reference [Par 99]: $\Omega = 1.66\text{x}10^{-29}$ m^{-3}, B

= 50 GPa, $D_{Al,bamboo,0}$ = $1.49x10^{-4}$ m^2/s, and $\Delta H_{Al,bamboo}$ = 0.9 eV, $D_{Al,gb,0}^{pure} = 1.9x10^{-5} m^2/s$, $\Delta H_{Al,gb}^{pure} = 0.6eV$, $D_{Al,gb,0}^{sat} = 1.9x10^{-5} m^2/s$, $\Delta H_{Al,gb}^{sat} = 0.8eV$, ΔH_B = 0.25 eV, the effective charge number for aluminum $Z_{Al}^{*} = 4$, and the effective charge number for copper $Z_{Cu}^{*} = 12$. The effective diffusion coefficient of the interconnect is the diffusivity normalized to correspond to the entire line cross section. In polygranular regions, grain boundaries provide the dominant diffusion paths, and the effective diffusion coefficient is approximated by the product of the grain boundary diffusivity and the ratio of grain boundary width and line width, assumed to be $1x10^{-5}$ [Kor 93]. Similarily, in bamboo regions, surface and interfaces provide the dominant diffusion paths, and the effective diffusion coefficient in bamboo regions is obtained by multiplying the diffusivity by the ratio of interface thickness and line thickness, assumed to be $4x10^{-5}$ [Sri 99].



(a)

(b)

(c)

(d)

Figure 2.2: Simulated hydrostatic stress profiles for different times before and after void nucleation. The arrow indicates the direction of the electron flow. (a) A limb in the center does not affect the stress evolution, whereas an inactive limb at the electron source side acts as a reservoir for metal atoms, as shown in (b). Similarly, a limb on the electron sink side would act as a sink for metal atoms. An electric current in the additional metal limb can decelerate (c) or accelerate (d) the stress buildup. The stress conditions are j = $2x10^6$ A/cm^2 and T = 250°C.

2.3 Simulation Results on Simple Tree Structures

In stud-to-stud lines, which are completely encapsulated by a rigid dielectric, atoms accumulate at the anode leading to a compressive stress, and atoms are depleted at the cathode end leading to a tensile stress, as shown in figure 2.2 (a). As the tensile stress rises during electromigration stressing, it eventually exceeds a critical stress for void nucleation, and a void forms. In this case, the void will grow at the cathode end of the line causing the current to flow through the shunt layer, thereby leading to an increase of the electrical resistance of the line. We will now discuss the effects of passive and active reservoirs on the stress evolution in interconnects, how reservoirs affect the steady-state stress distribution, and, finally, how reservoirs affect voiding in interconnects.

2.3.1 Passive Reservoirs

The additional metal limb in the middle of the line in the "T" structure does not significantly affect the stress evolution in the line because the limb is located at a stress-neutral point, shown in figure 2.2 (a). However, if the additional metal limb is at the cathode end of the line, the buildup of tensile stresses is slowed, as can be seen in figure 2.2 (b), because the passive reservoir acts as a source for metal atoms. This results in a longer lifetime of the interconnect. Similarly, a passive reservoir at the cathode end acts as a sink for metal atoms, slowing the buildup of compressive stress. Passive reservoirs provide an example in which the reliability prediction based on the assessment of the separate segments of a tree, as shown in figure 1.5 (b), is overly conservative.

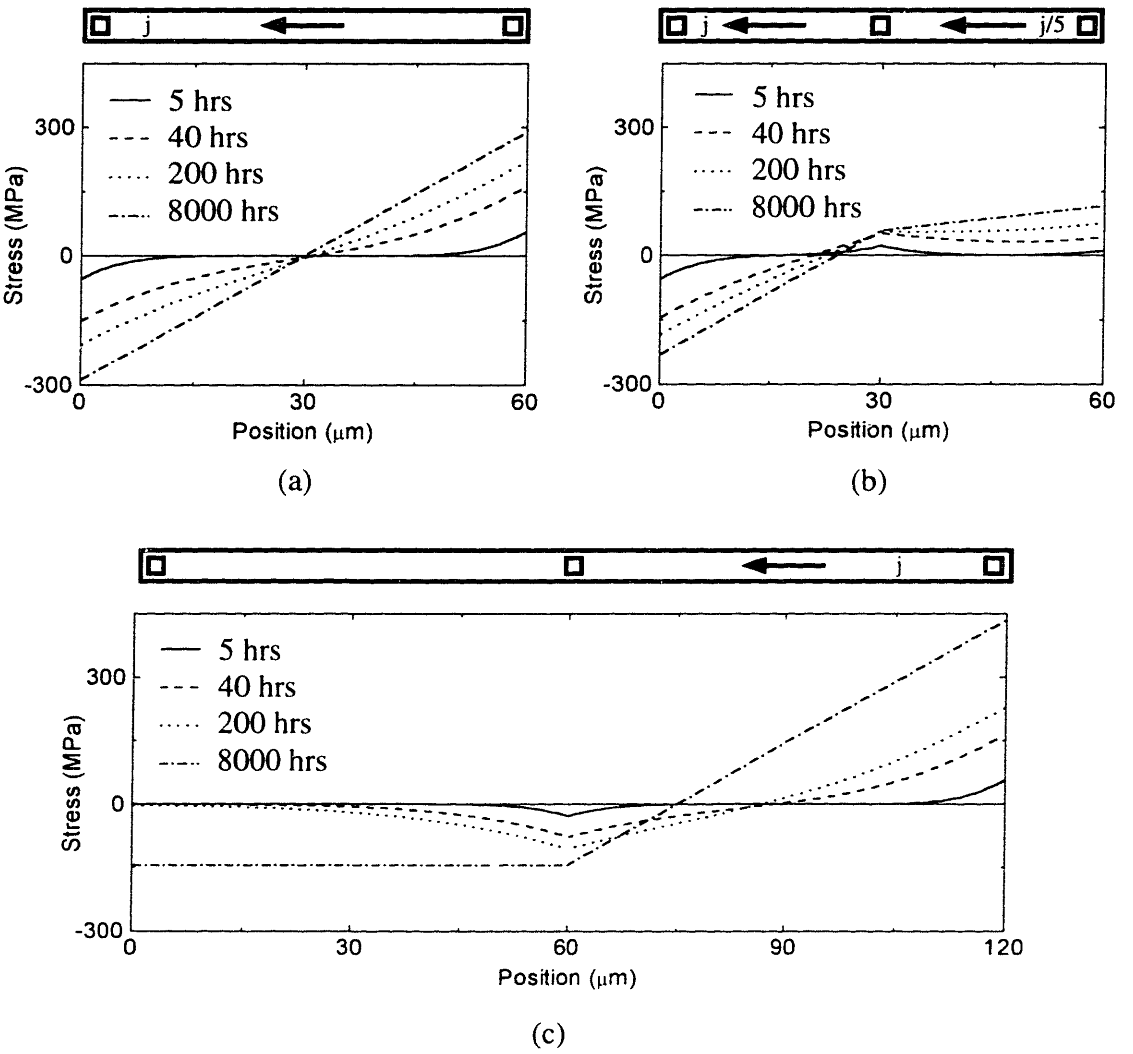


Figure 2.3: Simulated hydrostatic stress profiles for different times for the stress condition j = 5x10^5 A/cm^2 at T = 250°C. The arrows indicate the direction of the electron flow. (a) The stress stays below the critical stress necessary for void nucleation, and eventually a steady state is reached. (b) The steady-state stress distribution in the case of a "dotted-I" structure with different current densities in each limb. (c) A passive limb at the anode slows the buildup of compressive stresses, but the maximum tensile stress at steady state is larger than in the case without a reservoir.

2.3.2 Active Reservoirs

The effectiveness of the additional metal limb can be altered by passing a current through it. It is then called an "active" reservoir. For an active reservoir at the cathode end of the line, electron flow toward the via slows down the buildup of the tensile stress even further, see figure 2.2 (c), whereas an electron flow away from the via accelerates the buildup of the tensile stress, as shown in figure 2.2 (d).

2.3.3 Steady State

For low current densities and/or short interconnects, electromigration-induced stresses stay small enough that no void nucleation occurs. As shown in figure 2.3 (a), a linear steady-state stress distribution is eventually reached. Similarily, if different current densities flow through the different limbs of the tree, a steady state can still be reached, as shown in figure 2.3 (b). The slopes of the stress profiles at steady state are proportional to the current densities. As can be seen in figure 2.3 (c) and as discussed above, a passive limb at the anode acts as a sink for metal atoms, and slows the buildup of compressive stresses. However, the limb also allows more atoms to be transported toward the anode, so the tensile stress at steady state is *higher* than in case (a).

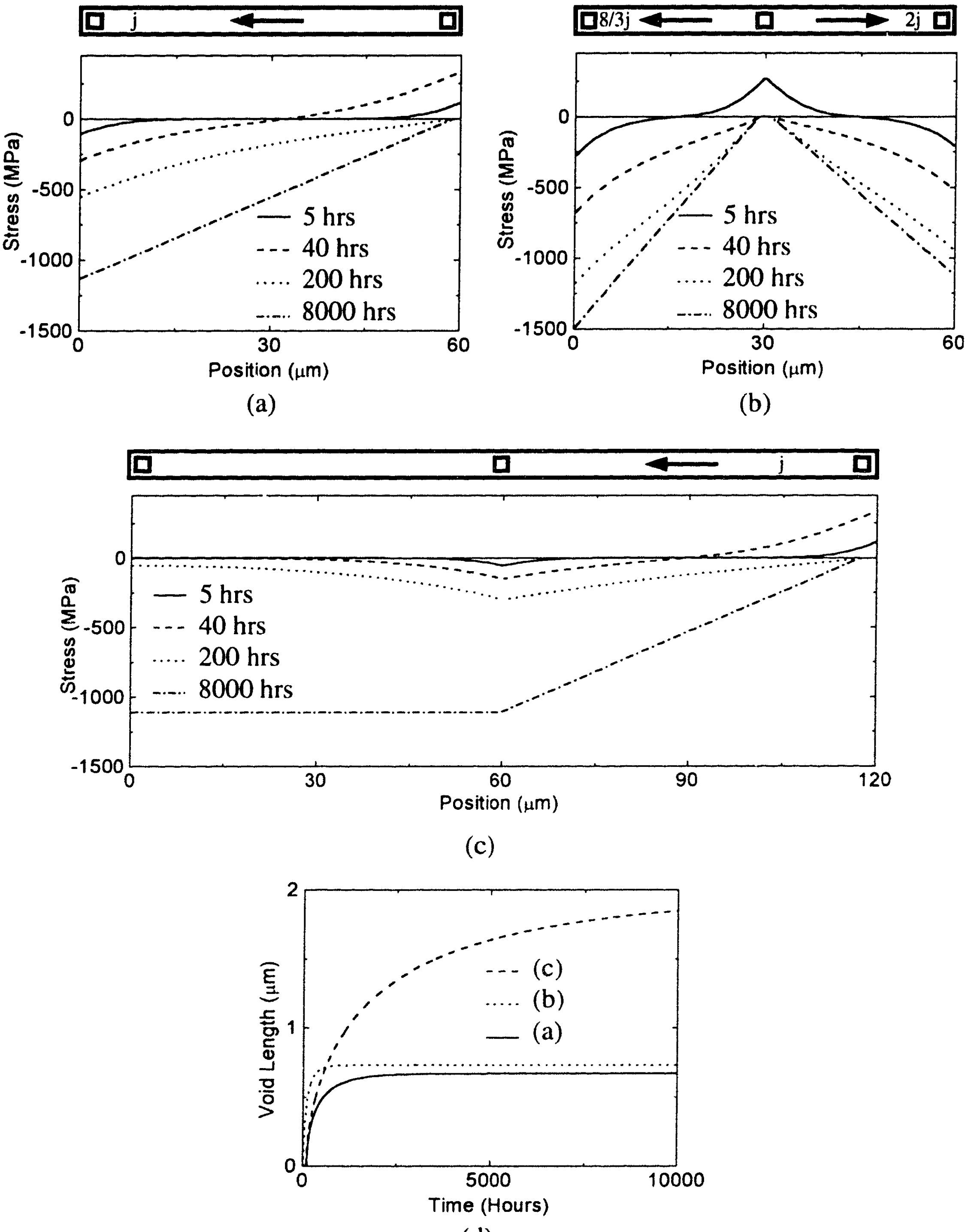


Figure 2.4: (a) to (c): simulated hydrostatic stress profiles for different times for the stress condition j = 1×10^6 A/cm^2 at T = 250°C are shown. The arrows indicate the direction of the electron flow. (a) The stress evolution in an "I" structure in which a void nucleates and grows to saturation. (b) The stress evolution in the case of a "dotted-I" structure. (c) The stress evolution in the case of a passive limb at the anode. (d) The void lengths as a function of time for the cases shown in (a), (b), and (c).

2.3.4 Voiding

If the tensile stress exceeds a threshold value, a void nucleates and relieves the stress around the void, as shown in figure 2.4 (a). The void grows until saturation effects due to back stress occur. Figure 2.4 (d) shows the void length as a function of time. As in the case without voiding, the stress profile is linear at steady state. Figure 2.4 (b) shows the stress evolution in the case when different parts of the tree carry different current densities, and voiding occurs. As shown in figure 2.4 (c), a passive limb at the anode allows more atoms to deplete from the cathode, so that the void at steady state is more than twice as large as the void in the case without a reservoir. The steady-state void size depends on the size of the reservoir.

2.4 Lifetime Experiments on Active and Passive Reservoirs

To test simulations of the reliability of trees, and to build the experience and database needed to develop analytic reliability models, we have tested the reliability of simple interconnect trees. We have designed test structures that allow the testing of "dotted I's", lines with bends ("L's" and "U's") as well as lines with junctions, leading to "T" shapes. In all cases, bends or junctions are connected to another layer of metallization (through a W-filled via), so that each segment within these simple trees can be made to carry different currents, including zero current (in order to test the effects of passive reservoirs). The interconnects were made of Al(0.5wt%Cu) terminating in tungsten vias. The line widths of the electromigration samples were 0.27 and 3.0 μm, respectively. The samples were fabricated in the Advanced Technology Group (ATG)

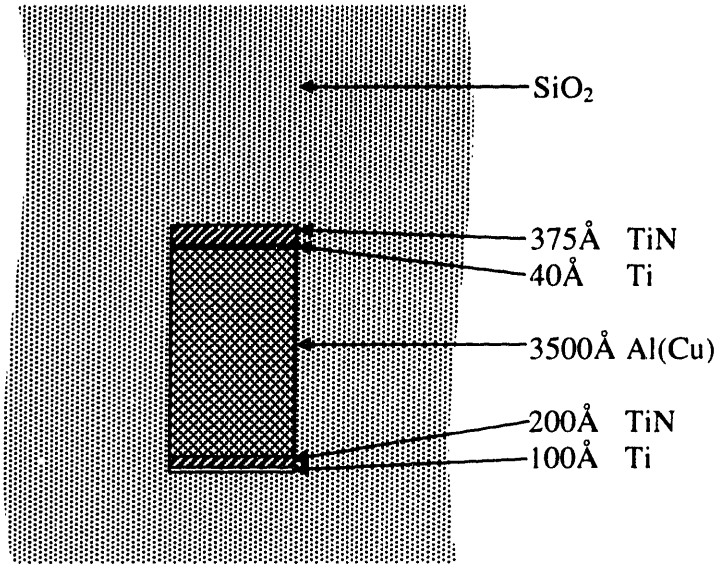


Figure 2.5: A cross section of the electromigration test structures.

clean-room facility of National Semiconductor. The wafer-level electromigration experiments were performed at MIT, and the package-level electromigration experiments were performed in collaboration with National Semiconductor (at National Semiconductor's ATG facilities).

2.4.1 Fabrication of Test Structures

The electromigration samples were Al(0.5wt%Cu) interconnects terminating in tungsten-filled vias. The samples were fabricated by thermally oxidizing p-doped [100]-oriented Si wafers, and subsequently depositing the first metallization stack (MT1) consisting of 100Å Ti / 200Å TiN / 3500Å Al(Cu) / 40Å Ti / 375Å TiN. After patterning the metal using reactive-ion etching, SiO_2 was deposited as an inter-level dielectric (ILD1), and planarized by chemical-mechanical polishing (CMP). A cross section of the interconnect is shown in figure 2.5. W-filled vias were formed by etching via holes into the ILD1, depositing W by chemical vapor deposition, and subsequent CMP of the W leaving W only in the via holes. The second metallization stack (MT2) with the same sequence of materials and thicknesses as MT1 was deposited, patterned, and passivated with SiO_2 (ILD2). ILD2 was planarized by CMP, and windows were opened over the bond pads. An additional 1 μm thick film of Al(Cu) was deposited and patterned over the bond pads in order to improve the quality of the wire bonding.

Finally, the wafers were diced. For package-level testing, the die were packaged in ceramic packages and cured at a maximum temperature of 350°C for one hour. The bonding pads were connected to the package pads using gold wires, and the packages were sealed under forming gas.

2.4.2 Electromigration Testing Procedure

Preliminary electromigration experiments at wafer-level were performed at MIT. The objectives of these experiments were (i) to find a testing procedure for interconnect trees, and (ii) to identify the emerging issues when dealing with interconnect trees instead of two-terminal, straight-line interconnects. During electromigration tests of trees, different limbs of a tree *simultaneously* carry different currents, which requires changes in the conventional test setup for two-terminal devices proposed by Thompson *et al.* [Tho 86].

After the preliminary test, we performed large-throughput package-level experiments to obtain statistically significant results. The ensemble size of each test was about 20 specimen. In the following sections, we will describe the experimental procedure for wafer-level and package-level electromigration experiments.

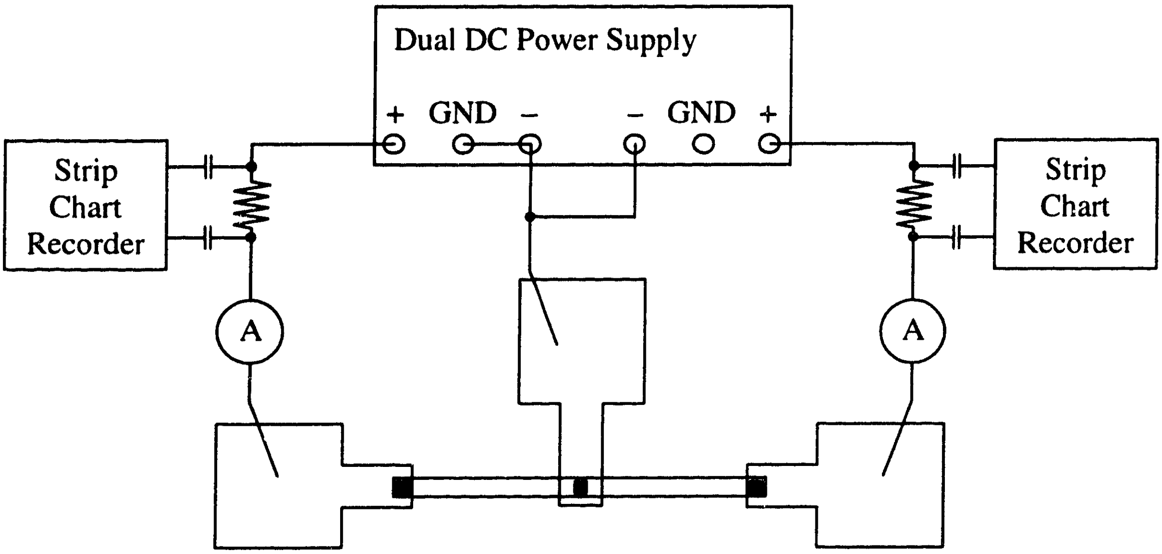


Figure 2.6: The electrical circuit of the electromigration test system. For clarity, the voltage sensors of the power supply, which are connected to the contact pads, were omitted in the drawing.

2.4.2.1 Wafer-Level Experiments

Constant-voltage electromigration lifetime tests were performed forcing an initial current density of $j = 2x10^6 A/cm^2$. An HP 6205C dual DC power supply with remote voltage sensing was used in order to maintain good load regulation, and the current was measured using a Keithley DMM 2000 digital multimeter. The circuitry is sketched in figure 2.6. The temperature during the test was maintained at 300 ± 4°C using a closed-loop temperature control system with a Eurotherm 983 Digital Indicating Controller, and a resistance heater under the wafer chuck which is 5 inches in diameter.

2.4.2.2 Package-Level Experiments

The package-level electromigration experiments were performed in a QualiTau MIRA electromigration test system [Qua], and two devices under test (DUT's) per package were stressed simultaneously. After heating the packages from room temperature to the test temperatures, which ranged from 250 to 350°C, the system was allowed to equilabrate for one hour. The lines were then stressed by forcing a constant current and monitoring the voltage drop over the interconnects. The voltage over each device was measured every minute, and recorded if the change exceeded the previously recorded measurement by 1%. The samples did not have additional, adjacent metal lines for detecting metallic extrusions, but we verified that no extrusions formed through post-mortem optical inspection.

2.4.3 Joule Heating Measurements

Stressing interconnects at high current densities can result in significant Joule heating. Different current densities in different limbs of an interconnect tree as well as connections through vias to upper and lower metal layers can result in highly non-uniform temperature profiles. These temperature gradients can be the origin of mass-flux divergencies, because the diffusivity is strongly temperature-dependent. To minimize these effects, we kept the average temperature increase due to Joule heating below 2°C at any time. The average Joule heating of the whole interconnect at test temperature was estimated by first measuring the temperature dependence of the resistance at low current densities (in order to have only a negligible amount of Joule heating), and then measuring the resistance as a function of current density at test temperature.

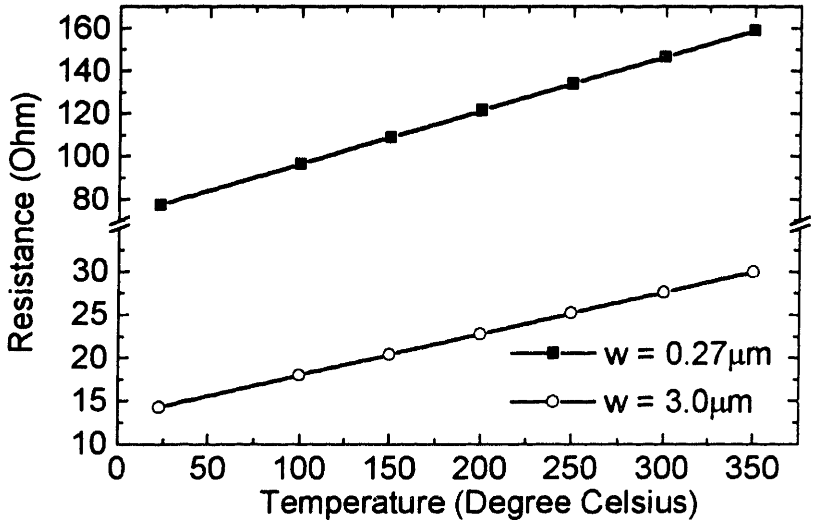


Figure 2.7: The electrical resistance of 0.27μm- and 3.0μm-wide lines as a function of temperature determined through a four-point measurement by applying a current density of 0.5×10^5A/cm^2 and measuring the voltage drop over the line.

The temperature coefficient of resistance, TCR, is the fractional change in the electrical resistance, ΔR, of the metal line due to a change in temperature, ΔT, at a given temperature T [Sch 94],

$$\mathrm{TCR(T)} = \frac{1}{\mathrm{R(T)}}\cdot\frac{\Delta \mathrm{R}}{\Delta \mathrm{T}}. \qquad (2.1)$$

The TCR varies with temperature, and a TCR at one temperature can be related to a TCR at another temperature by [EIA 95]

$$\mathrm{TCR(T')} = \frac{\mathrm{TCR(T)}}{1+\mathrm{TCR(T)}\cdot(\mathrm{T'}-\mathrm{T})}. \qquad (2.2)$$

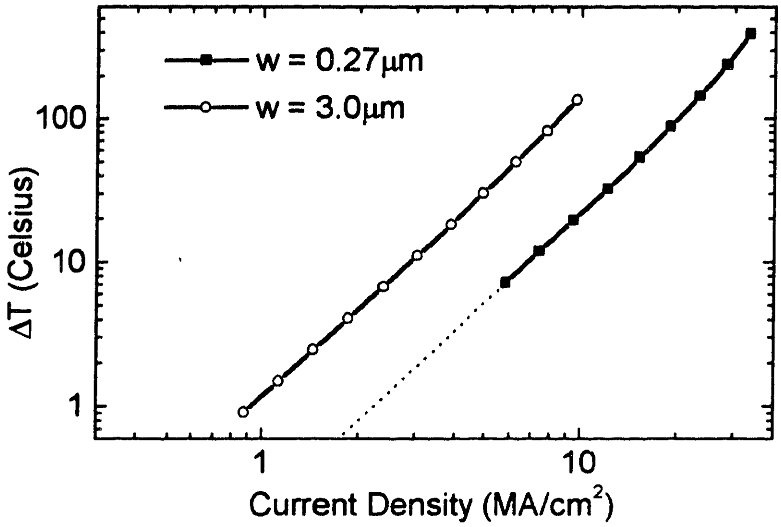


Figure 2.8: Average temperature rise due to Joule heating as a function of current density in 0.27μm- and 3.0μm-wide lines. The die were heated to 300°C.

Figure 2.7 shows the temperature-dependence of the electrical resistance for 3.0μm and 0.27μm-wide lines, which was measured using a four-point-probe configuration and forcing a current density of j = 0.5×10^5A/cm^2. The resistance change is proportional to the temperature change for temperatures ranging from 25°C to 350°C. Using equation (2.1), we obtain TCR(25°C) = 3.3×10^{-3}K^{-1} for the 0.27μm-wide lines, and TCR(25°C) = 3.5×10^{-3}K^{-1} for the 3.0μm-wide lines. We then measured the increase of the electrical resistance with increasing current density due to Joule heating at 300°C. Correlating the resistance increase due to Joule heating with the temperature-dependence of the resistance, shown in figure 2.7, the average temperature increase of the interconnect as a function of current density can be estimated, as shown in figure 2.8.

For the same current density, the temperature increase in the wide lines is significantly larger than in the narrow lines, because cooling of the wide lines through the substrate occurs in a quasi one-dimensional manner, whereas the narrow lines are cooled in a quasi two-dimensional manner. In order to keep the Joule heating of the lines below

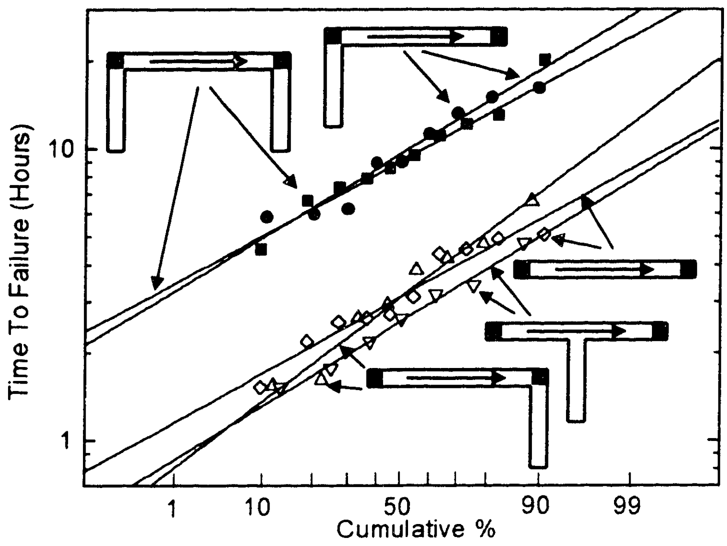


Figure 2.9: Lifetimes measured in experiments on simple interconnect trees with only one segment or limb carrying current. The lines were stressed with a current density of 2x10^6 A/cm^2 at 350°C. The arrows indicate the direction of the electron flow.

ΔT = 2°C, we kept the current densities in the 0.27μm-wide lines below 3.4 MA/cm^2, and the current densities in the 3.0μm-wide lines below 1.3 MA/cm^2.

2.4.4 Results on "I"-, "L"-, "U"-, and "T"-shaped Interconnects

First results of preliminary wafer-level electromigration experiments are shown in figures 2.9 and 2.10. These figures show results from highly accelerated tests on "L", "U", and "T" structures. Figure 2.9 shows results from structures tested with just one active (current carrying) limb. The active limb in these structures is 500μm long, and the inactive limbs are 250μm long. The current density in the active limbs was 2x10^{6}A/cm^2, with electron flow in the directions indicated by the arrows. The tests were conducted at 300°C. The time to failure was defined as the time at which the resistance of the structure (as measured in the active limb) had increased by 5%.

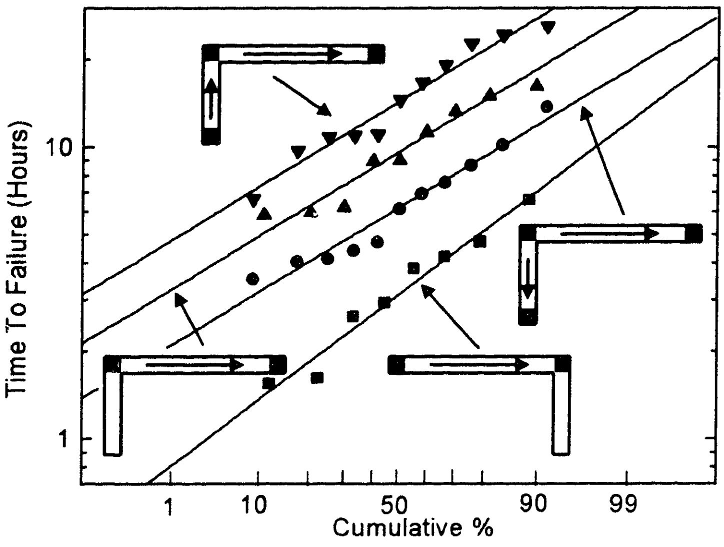


Figure 2.10: Comparison of lifetimes for populations of "L"-shaped interconect trees with only one or both limbs carrying current. The lines were stressed with a current density of $2x10^6$ A/cm^2 in the long limb, and the current density in the active reservoirs was $5x10^6$ A/cm^2. The temperature was 350°C.

Figure 2.10 compares results from "L" structures with one or both limbs active and with different current-direction configurations. The reported lifetimes are again the times at which the resistances in the longest limbs had increased by 5%. When the shorter limbs were active, they carried one fourth of the current density in the longer limbs (which was again $2x10^6$ A/cm^2). The tests were conducted at 300°C.

In all cases, the resistance of the monitored limb was essentially unchanged during most of the lifetime of the lines. There was an abrupt increase in the resistance by 5 to 10% at the end of the life of the test structures. These abrupt increases are thought to occur when voids grow to be large enough to extend beyond the edges of the cathode vias.

2.5 Discussion

As expected, and as simulation results suggest, electromigration-induced effects in interconnect trees differ in several ways from electromigration-induced effects observed in single straight stud-to-stud interconnects ("I" structures). The stress and void evolution and therefore the failure processes in trees are more complex than in "I" structures.

The results of experiments on interconnects with passive reservoirs, as shown in figure 2.9, illustrate the importance of inactive limbs which serve as reservoirs from which atoms replace electromigrating atoms. Likewise, the results of experiments on interconnects with active reservoirs, as shown in figure 2.10, illustrate the importance of the electronic current configuration. Using models that assume that only the electromigration-induced atomic flux contributes to the reliability of the nodes in these structures, would be conservative.

The ordering of the lifetime results in figures 2.9 and 2.10 is consistent with the results from simulations discussed in Sections 2.3.1 and 2.3.2 in that the ordering of the maximum stresses corresponds with the ordering of the observed lifetimes. This round of experiments suggests that the basic understanding of tree reliability as captured by the application of equation (1.19) in simulations, is valid. To explore tree reliability in more detail, we performed more electromigration experiments on simple tree structures using different temperatures, current densities, and linewidths. The results of these experiments are discussed in Chapter 4, where we also discuss the effects of active and passive reservoirs on lifetime in more detail.

Our simulations of electromigration suggest that a steady state, which has been observed in "I" structures, also occurs in interconnect trees. For low current densities and short interconnects, a steady state is reached before voiding occurs, and the tree is immune to electromigration-induced failure. Likewise, if voiding does occur, and the current can continue to flow through shunt layers which do not electromigrate, a steady state can be reached with voids in the interconnect. The voids lead to an increase in the electrical resistance of the tree. If this resistance increase is within acceptable limits, the tree can again be considered immortal. In the next chapter, we report on experiments demonstrating the occurance of saturation effects in trees, and we suggest means for using the immortality concept to facilitate reliability assessments of large layouts.

Chapter 3 Immortality Filters for Identifying Immune Interconnect Trees

3.1 Background

There has been an increased interest in developing new techniques for making realistic reliability assessments *during* the design and layout process (RCAD), so that analyses concerning reliability can be fed back into the design and layout process immediately. Reliability analyses applied in an RCAD framework can take into account the details of a layout and of circuit operation in order to achieve optimum performance while retaining high overall reliability. RCAD makes frequent reliability assessments for a large amount of interconnect necessary, so that a computationally efficient and flexible strategy for the assessment of interconnect reliability is needed.

To facilitate RCAD, we propose the concept of *Hierarchical Reliability Analysis*, which gives a computationally manageable approach for making circuit-level, layout-specific reliability analyses [Rie 98a]. This approach applies filtering algorithms to identify interconnect trees which are immune to electromigration-induced failure. In this chapter, the filtering algorithms are described [Rie 98a, Cle 99], the effectiveness of the filtering algorithms is demonstrated by application to typical layouts, and the concept of immortality in interconnect trees (as opposed to simple, straight-line interconnects) is verified experimentally through electromigration experiments on saturation effects in "L"-shaped interconnects [Hau 00d]. Finally, we discuss a hierarchical approach to circuit-level reliability analyses.

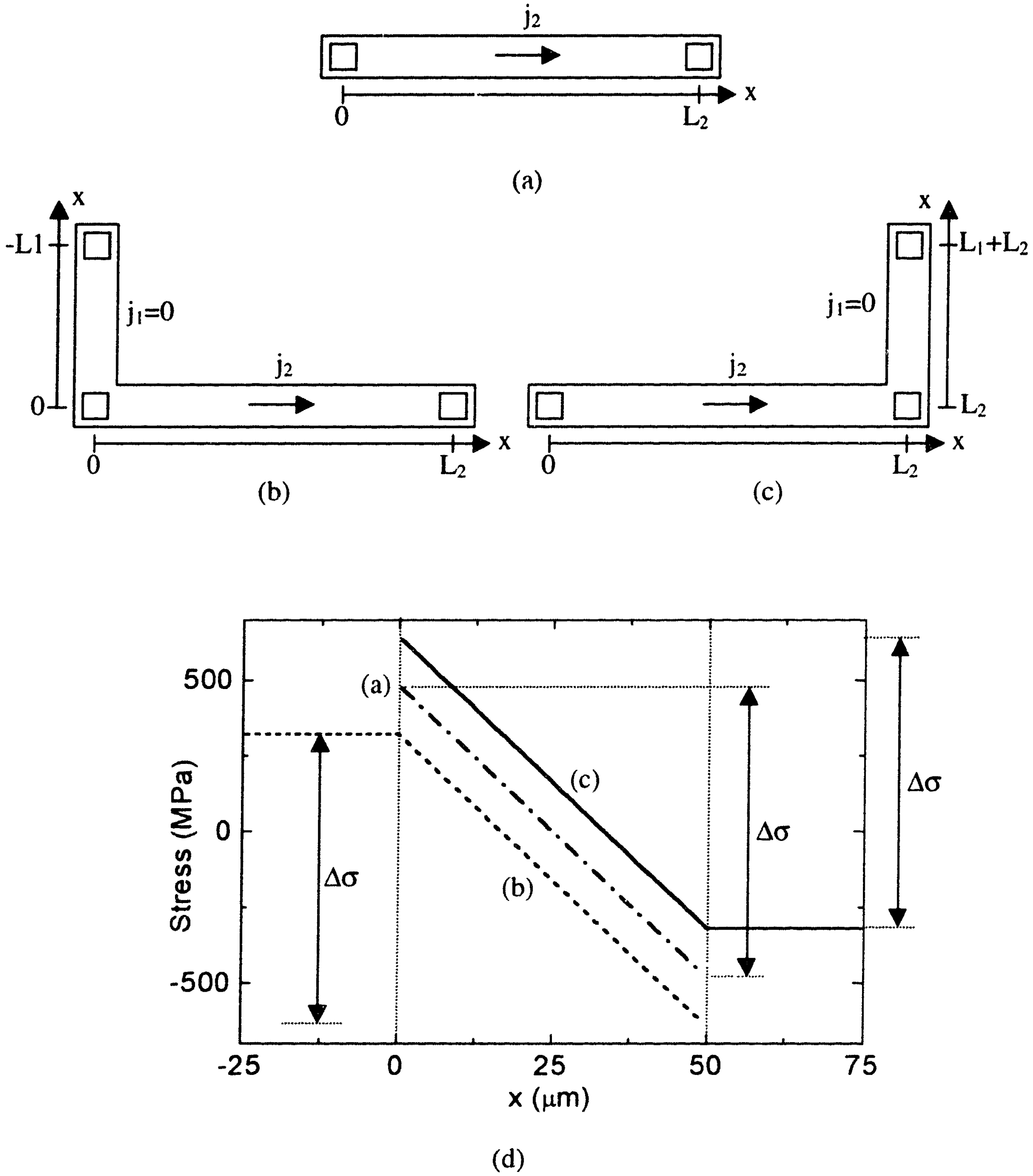


Figure 3.1: Steady-state stress distributions in "I"-shaped and "L"-shaped interconnect trees. The additional limbs are passive in the "L"-shaped structures. The arrows indicate the direction of the electron flow. The line length, L_2, is 50μm, the reservoir length, L_1, is 25μm, and the current density, j_2, is 1×10^6A/cm^2.

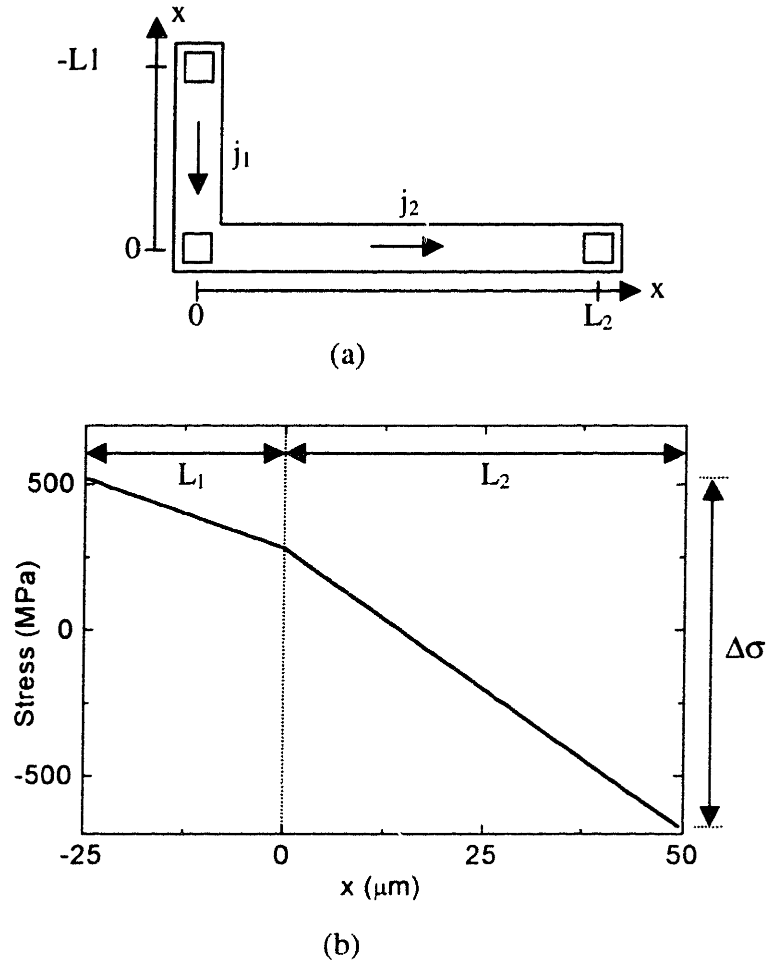


Figure 3.2: Steady-state stress distributions for the path through an "L"-shaped interconnect, in which both limbs are active. The arrows indicate the direction of the electron flow. The line length, L_2, is 50μm, the reservoir length, L_1, is 25μm, the current density in the line, j_1, is 0.5×10^6A/cm^2, and the current density in the reservoir, j_2, is 1×10^6A/cm^2.

3.2 Immortality Filters

In section 1.2.2, the immortality of simple, straight, stud-to-stud interconnects ("I" structures) was described. We will now extend this understanding of immortality to arbitrarily complex interconnect trees. We consider a tree which has n limbs. Each limb has a length L_i and width w_i, and carries a current density j_i. We will first discuss the steady state without void nucleation [Rie 98a], and then the steady state resulting from saturation of void growth [Cle 99].

3.2.1 Steady-State Stresses in Interconnect Trees

We will first consider a straight-line interconnect, as shown in figure 3.1 (a), and the effect of a passive reservoir at the electron-source side (figure 3.1 (b)) and at the electron-sink side (figure 3.1 (c)), respectively, on the steady-state stress distribution. We will then consider the case in which the reservoir is active, as shown in figure 3.2 (a), and located anywhere along the line, as shown in figure 3.3 (a), which is a "T" structure. Finally, we generalize the results to arbitrarily complex trees.

Consider the "L" structure shown in figure 3.1 (b) where limb 1 is not carrying a current. The inactive limb 1 will serve as a reservoir of material, and will slow the rate of increase of the stress at the via at the corner. Similarily, if limb 1 is located at the electron-sink side, the passive reservoir acts as a sink for material. The steady-state stress profiles in both cases, as well as for the case without reservoir, are shown in figure 3.1 (d). If limb 2 has the same current density, j_2, in all three cases, the slope of the stress profiles will be the same, and if in addition the length of limb 2, L_2, is the same, the maximum stress difference, $\Delta\sigma$, is also the same, given by

$$\Delta\sigma = \frac{q^*\rho}{\Omega}(j_2L_2), \tag{3.1}$$

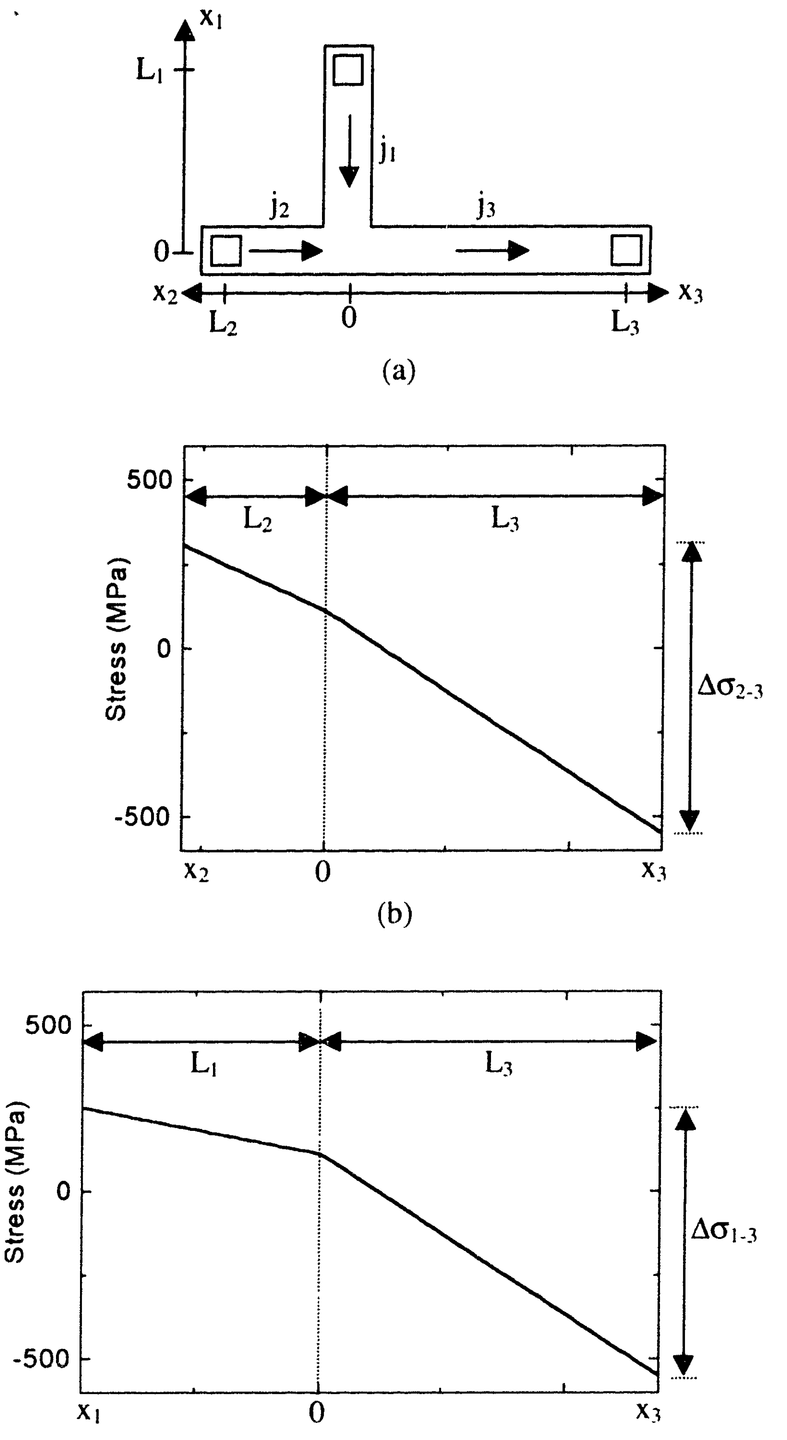


Figure 3.3: Steady-state stress distributions for two paths through a T-shaped interconnect tree, in which all limbs are active. The arrows indicate the direction of the electron flow. The lengths of the limbs are $L_1 = 25\mu m$, $L_2 = 15\mu m$, and $L_3 = 35\mu m$, and the current densities are $j_1 = 0.3x10^6 A/cm^2$, $j_2 = 0.7x10^6 A/cm^2$, and $j_3 = 1.0x10^6 A/cm^2$.

where ρ is the electrical resistivity, $q^* = Z^*e$ is the effective atomic charge, Z^* is the effective atomic charge number, and e is the fundamental charge, and Ω is the atomic volume. However, the absolute stress values are different in the three cases. A reservoir at the cathode can lead to higher compressive stresses, and a reservoir at the anode can lead to higher tensile stresses. The calculation of the absolute stress values is discussed later. For now, we will focus on the differences in stresses.

If limb 1 in figure 3.2 (a) is active, and electrons flow from limb 1 to limb 2 and out of the anode end of limb 2, the steady state stress will be as shown in figure 3.2 (b). In comparison with the case of a passive reservoir, as shown in figure 3.1 (b), the stress profile will be unchanged in limb 2, but the maximum tensile stress will be higher and will occur at the cathode end of limb 1. The steady state stress difference is given by

$$\Delta\sigma = \frac{Z^* e \rho}{\Omega}\left(j_1 L_1 + j_2 L_2\right) \tag{3.2}$$

Consider the T of figure 3.3 (a). The steady state can be analyzed by considering the paths between any two nodes. We will discuss the case in which the electron flow from limbs 1 and 2 into and through limb 3. The steady state profile for the path from the cathode of limb 1 through the anode of limb 3 is shown schematically in figure 3.3 (b), and the profile for the path from the cathode of limb 2 to the anode of limb 3 is shown in figure 3.3 (c). The steady state stress differences for these two paths are

$$\Delta\sigma_{1-3} = \frac{Z^{*}e\varrho}{\Omega}\left(j_1L_1 + j_3L_3\right) \tag{3.3}$$

and

$$\Delta\sigma_{2-3} = \frac{Z^{*}e\varrho}{\Omega}\left(j_2L_2 + j_3L_3\right). \tag{3.4}$$

The maximum stress difference is then given by the path that has the highest sum of the jL products, summing over the limbs in the path,

$$\Delta\sigma_{\max} = \frac{Z^{*}e\varrho}{\Omega}\left(jL\right)_{\mathrm{eff}} \tag{3.5}$$

with

$$\left(jL\right)_{\mathrm{eff}} \equiv \max_{\substack{\text{all junction}\\ \text{pairs } i,j}}\left(\sum_{k} j_k L_k\right) \tag{3.6}$$

The sum in equation (3.6) is taken over all limbs connecting junction i with junction j. This result is general for trees with n limbs.

The absolute magnitude of the steady state stress anywhere in the tree can be found, if the stress at a single reference location can be determined. If we choose the cathode of limb 1 as the reference location, the stress as a function of distance from the cathode in limb 1 is

$$\sigma(x_1) = \sigma_{ref} + a_1 x_1 \tag{3.6}$$

and the stress as a function of distance from the junction between limb 1 and 2 in limb 2 is

$$\sigma(x_2) = (\sigma_{ref} + a_1 L_1) + a_2 x_2 \ , \tag{3.7}$$

where

$$a_i \equiv \frac{Z^* e \rho}{\Omega} j_i \, . \tag{3.8}$$

In the general case,

$$\sigma_i = \sigma_{ref} + a x_i + b_i \, , \tag{3.9}$$

with

$$b_i \equiv \sum_{j}^{i-1} a_j L_j \tag{3.10}$$

The value of σ_{ref} can be determined by requiring mass conservation so that the total number of atoms in the initial state, $N_{initial}$, is the same as the total number of atoms in the steady state, N_{ss}, where

$$N = \text{total number of atoms} = \int_{tree} c(x)\ dV\,, \tag{3.11}$$

where the integral is taken over all the limbs of the tree. c is the atomic concentration (#/volume) which is related to the stress by equation (1.8), so that

$$N_{before} = c_0 \exp\left(-\frac{\sigma_0}{B}\right) \sum_i l_i A_i \tag{3.12}$$

and

$$N_{ss} = c_0\ B\ \exp\left(-\frac{\sigma_{ref}^{ss}}{B}\right) \sum_i \frac{A_i}{a_i} \exp\left(-\frac{b_i}{B}\right)\left(1-\exp\left(-\frac{a_i}{B} l_i\right)\right), \tag{3.13}$$

where A_i is the cross section of the line. Given σ_{ref}, $\sigma_{max}^{tensile}$ and $\sigma_{max}^{compressive}$ can be determined for the tree, and it can be determined if voids can nucleate and if the dielectric will fail.

The steady-state stress profiles are independent of the microstructure of the interconnects, because the steady state is not affected by the diffusivity nor by the stress-dependence of the diffusivity. Likewise, the steady-state stress profiles are independent of the presence of alloys, because the redistribution of the low number of alloy atoms does not lead to a significant stress change.

3.2.2 Steady-State Stresses with Voiding in Interconnect Trees

In the preceding discussion, we have focused on the steady state condition without void nucleation. If we instead consider the steady state that results from void growth saturation, the analysis remains the same. In this case, the void will be at the cathode with the highest negative voltage. The stress at the void surface is essentially zero, and it is convenient to define σ_{ref} as zero at this node. In this case, the atoms that cause the electromigration-induced stress gradients are supplied by the growth of the void. The number of atoms removed to form the void is given by $N_{initial}$-N_{ss}, with $N_{initial}$ and N_{ss} as defined in equations (3.12) and (3.13). This can be related to a void volume and to a resistance change, ΔR_{max}, so that $\sigma_{max}^{compressive}$ and ΔR_{max} can be calculated for comparison with the values that define mortality.

3.3 Effectiveness of Immortality Filters

3.3.1 Background

We will now demonstrate the effectiveness of the first filtering step by applying it to typical microprocessor layouts. The first filtering step is based on the calculation of the steady-state stresses without void nucleation, as discussed in section 3.2.1. Using equation (3.5), an upper limit for the electromigration-induced stress increase in a tree can be calculated. If the effective current-density line-length product, $(jL)_{eff}$, is below a critical product necessary for void nucleation, $(jL)_{crit}$, the tree will be immortal. Assuming an upper bound for the current density in the whole circuit, j_{max}, $(jL)_{eff}$ can be conservatively estimated by

$$(jL)_{eff} = \left(\sum_{i} j_i L_i \right)_{max} \leq j_{max} L_{max} , \tag{3.14}$$

with

$$L_{max} \equiv \max_{\substack{\text{all junction} \\ \text{pairs i, j}}} \left(\sum_{k} L_k \right). \tag{3.15}$$

L_{max} is the maximum of the smallest distances between two junctions, taken over all junction pairs. A tree is immortal if $L_{max} < (jL)_{crit} / j_{max} \equiv L_{crit}$. In the next section, we will describe a tool set developed to extract the distribution of L_{max} for each layer of metallization from a layout. We will then discuss L_{max} distributions extracted from microprocessor layouts, and demonstrate the effectiveness of the immortality filters.

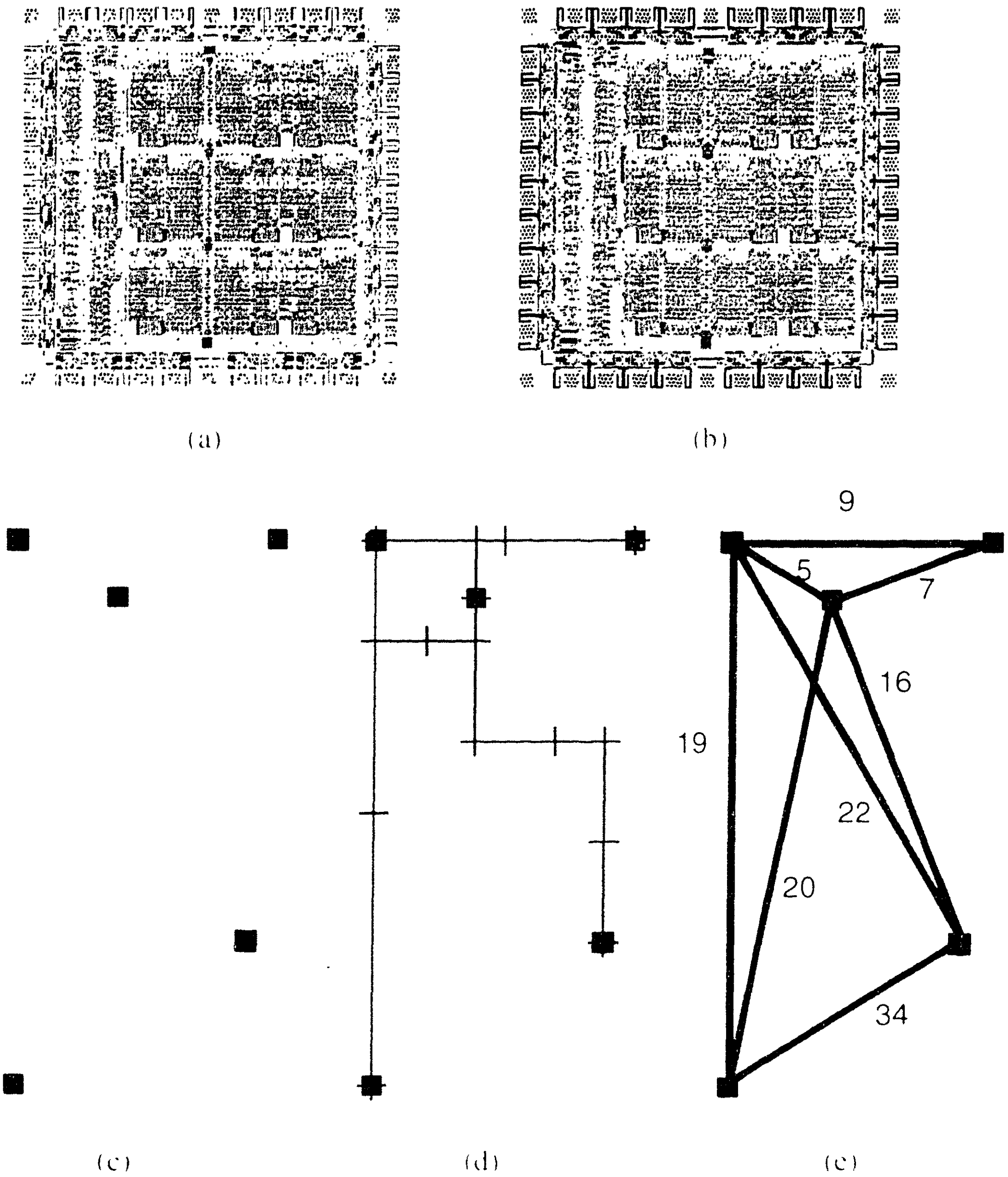


Figure 3.4: (a) An example of a microprocessor layout. The L_{max} distribution is extracted in four steps: (b) The layout is flattened by removing the cell hierarchy. (c) Interconnect trees are extracted from the layout. (d) Graphs are created to determine the connectedness of vias and contacts. (e) The shortest distances between all pairs of connected vias and contacts are calculated. Finally, a Floyd-Warshall algorithm is used to determine the largest distance in the graph, which is L_{max}. All lengths are given in μm.

3.3.2 Tool Set for Calculating L_{max}-Distribution

Figure 3.4 shows the steps necessary for extracting the L_{max} distribution. Starting with the physical layout, the cell hierarchy is removed by flattening the layout, as shown in figure 3.4 (a) and (b). Then, an extraction tool, *XT*, extracts all interconnect trees, i.e. all overlapping and abutting metallic rectangles within one layer of metallization. An example for such a structure is shown in figure 3.4 (c). The geometrical information in form of rectangle coordinates along with information about sizes and positions of vias and contacts are written into separate files. The extraction is done separately for each layer of metallization. In the second step, L_{max} is obtained for the trees in each layer of metallization through the use of an analysis tool, *ANT*, as shown in figure 3.4 (d) and (e).

In the extraction tool *XT*, a simple and straightforward algorithm for identifying trees, as described in by Ullmann *et al.* [Ull 84], is used, which essentially tests for overlap of rectangle pairs. The calculation time required for this algorithm is $O(n^2)$, which means that the growth rate of the calculation time for larger layouts is not greater than that of n^2, where n is the number of rectangles. This algorithm is appropriate for the small layouts discussed here, and could easily be improved by using well documented but more complex methods [McC 80].

In order to determine L_{max}, for each tree, the metallic rectangles are drawn onto an internal bitmap, as shown in figure 3.4 (c). An interconnect graph is created by (i) drawing connecting lines, which are crosses through the rectangles as well as lines radiating from vias and contacts; (ii) creating "temporary vias" whenever two lines cross (shown in figure 3.4 (d)); and (iii) finding the distances between all connected neighboring vias, contacts, and "temporary vias". The shortest distances between all

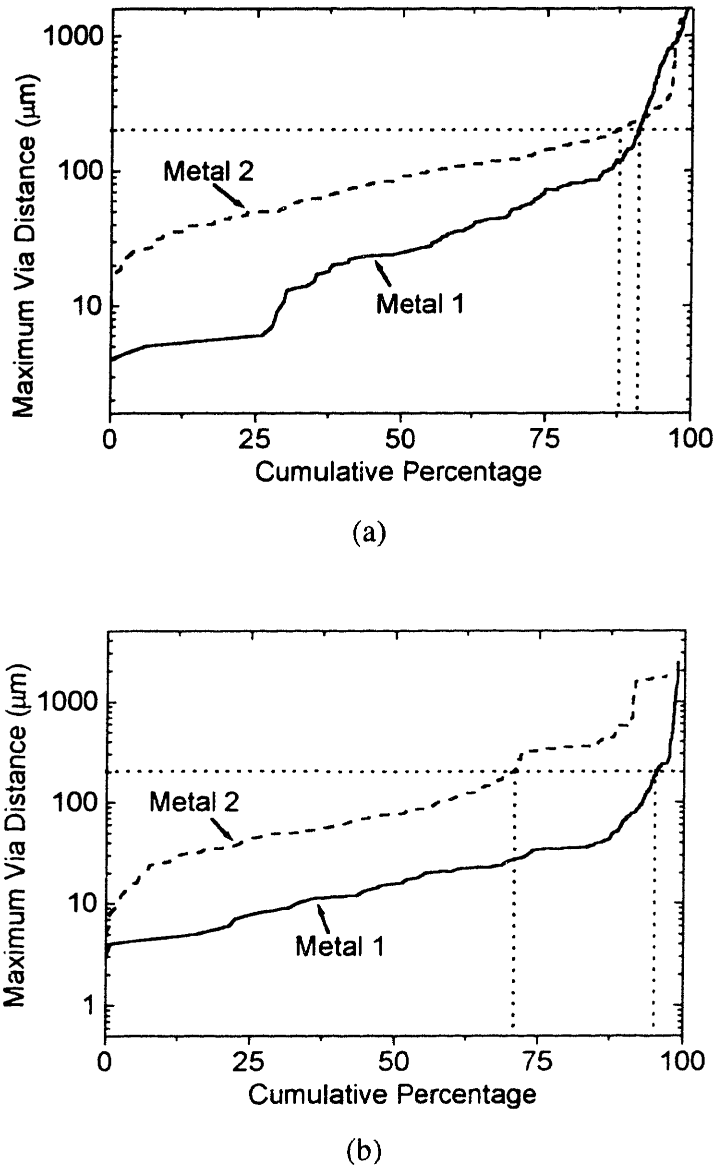


Figure 3.5: Distributions of the maximum via distances, L_{max}, for the first two metal layers in two microprocessor layouts. For L_{crit} = 200µm, 92% of metal-1 trees and 88% of metal-2 trees in case (a), and 95% of metal-1 trees and 71% of metal-2 trees in case (b) are identified as immortal to electromigration-induced failure.

connected vias and contacts are obtained by removing all "temporary vias", as shown in figure 3.4 (e). Finally, a Floyd-Warshall algorithm is used to find the longest distance in the graph, which is L_{max} [Cor 96].

3.3.3 Results

Two typical examples for the distribution of L_{max} are shown in figure 3.5 on cumulative plots. The layouts were taken from publicly available microprocessors. In the first layout, the first metal layer (metal-1) contained approximately 7300 trees, and the second metal layer (metal-2) contained approximately 1000 trees. In the second layout, metal-1 contained approximately 2100 trees, and metal-2 contained approximately 1000 trees. Taking the critical current-density line-length product for aluminum interconnect trees as $(jL)_{crit} = 2\times10^3$ A/cm [Kra 95], and assuming an industry-typical maximum current density of $j_{max} = 1\times10^5$ A/cm^2, the critical effective line length, L_{crit}, is 200μm. Using condition (3.14), we identified 92% of metal-1 trees and 88% of metal-2 trees in the first layout, and 95% of metal-1 trees and 71% of metal-2 trees in the second layout as immune to electromigration-induced failure.

The algorithm's efficiency can be improved significantly if a more detailed estimation of the maximum current density in the limbs of the trees is given. In this case, the distances, L_i, are weighted with the maximum current density for calculating $(jL)_{eff}$ using equation (3.6).

3.4 Experiments on Saturation Effects in Trees

To test the concept of immortality of interconnect trees, we performed electromigration experiments on passivated "L"-shaped Al(0.5wt % Cu) interconnects in the first level of metallization. The interconnects under test were electrically connected to the second layer of metallization through W-filled vias at the line ends and at the

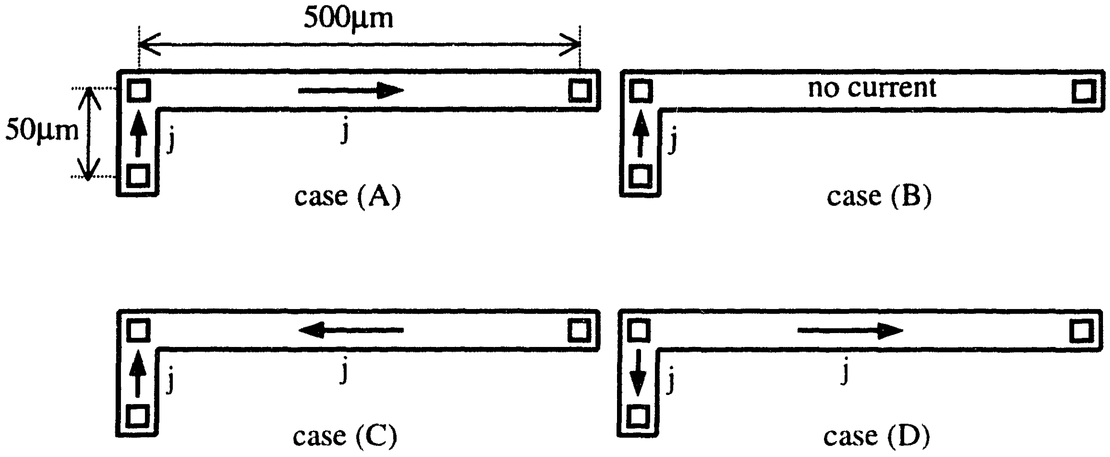
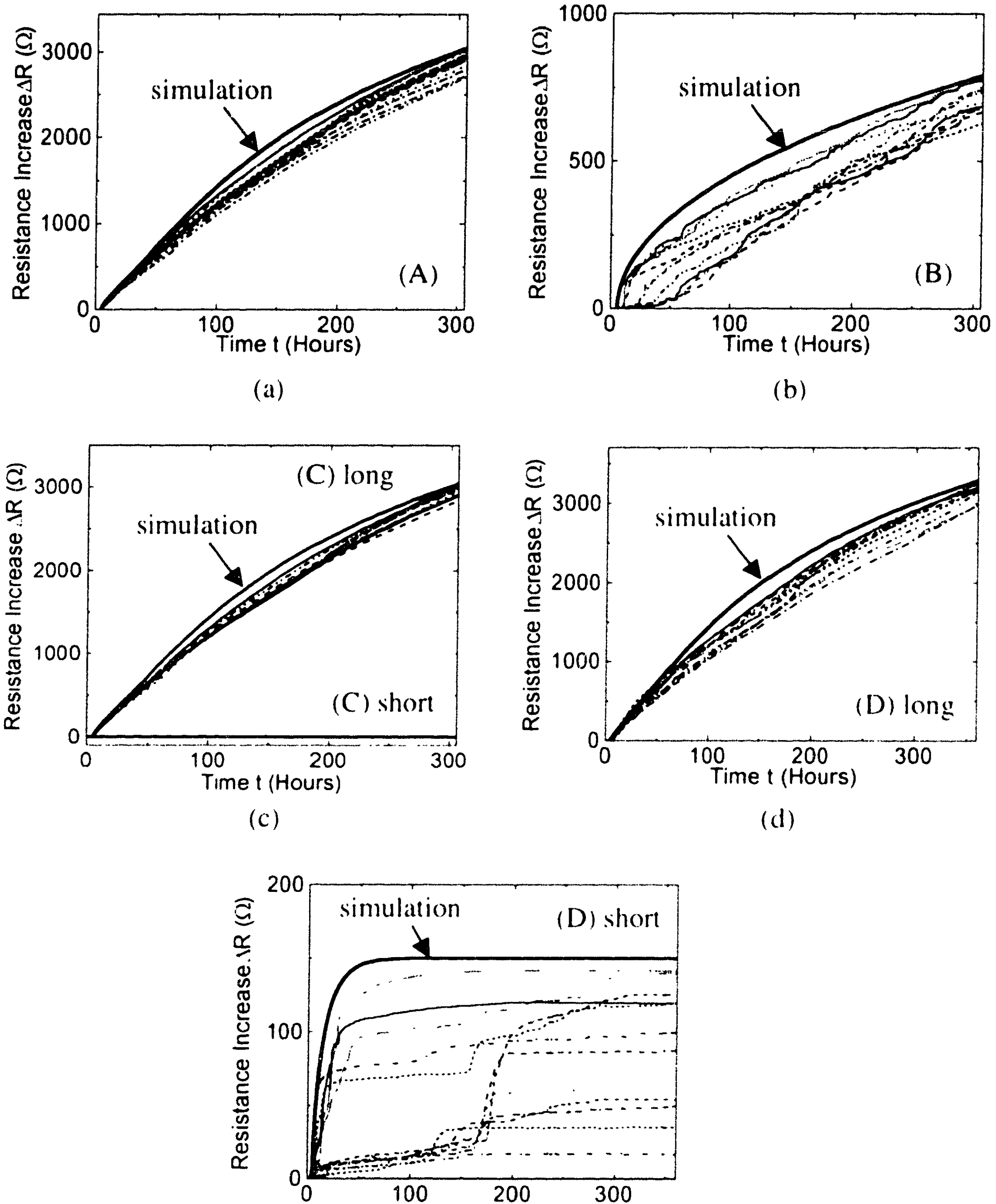


Figure 3.6: Sketches of the electromigration test structure and the current configurations used in experiments. The arrows indicate the direction of the electron flow.

corner of the "L", as shown in figure 3.6. This allowed independent application of currents of different magnitudes and directions in the two limbs of the "L". The short limb was 50μm long, the long limb was 500μm long. The line width was 0.27μm. The metallization stack consisted of 100Å Ti / 200Å TiN / 3500Å Al(0.5 wt % Cu) / 40Å Ti / 375Å TiN.

The samples were stressed in a QualiTau MIRA electromigration test system, at a temperature of 350°C, by forcing a constant current density of $j = 8x10^5 A/cm^2$. The voltage drops over the two limbs of the "L"-shaped interconnects were measured as a function of time. Figures 3.6 (a) to (d) show sketches of the different current configurations, which will be referred to as cases (A), (B), (C), and (D), respectively. Processing of the electromigration samples and electromigration testing were done in collaboration with National Semiconductor.



(a)

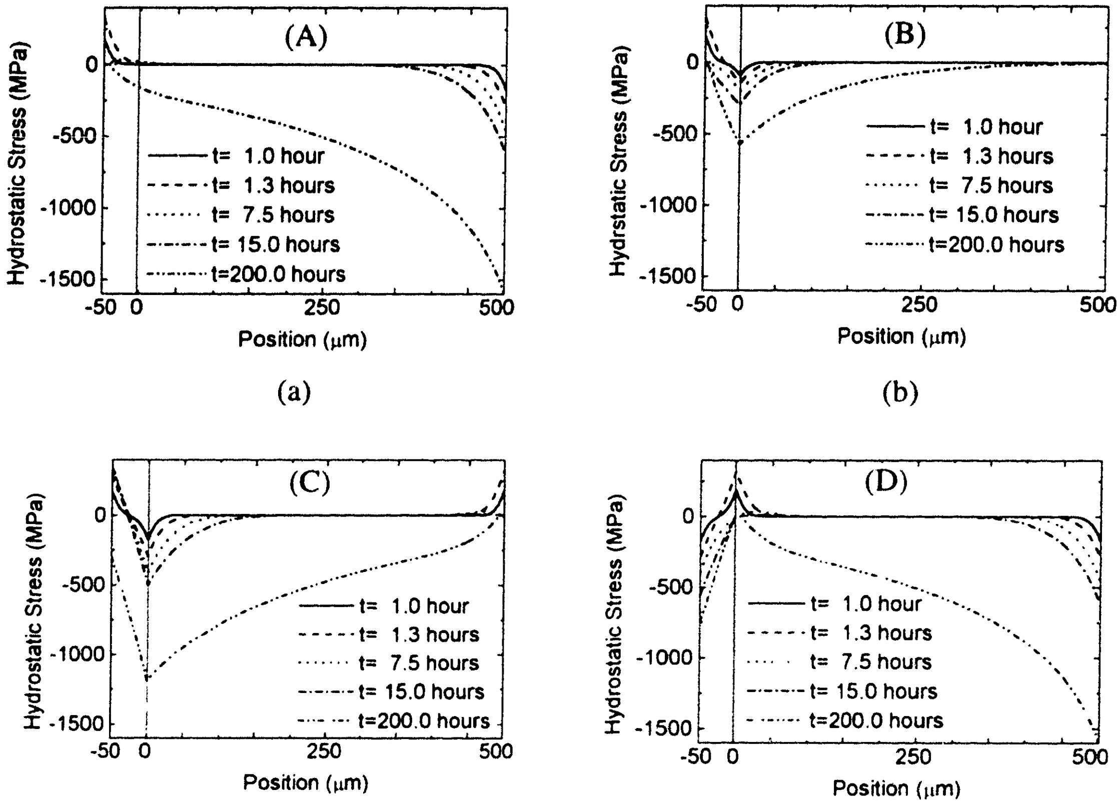
(b)

(c)

(d)

(e)

Figure 3.7: Increases in electrical resistance. ΔR. as a function of time observed in experiments for the cases (A) to (D) shown in figure 1. The time dependence of the resistance increase obtained through simulation for cases (A) to (D) is shown as a thick solid line.



(a) (b)

(c) (d)

Figure 3.8: Hydrostatic stress profiles for different times obtained by simulations for cases (A) to (D).

3.4.1 Experimental Results

Figure 3.7 (a) shows the resistance increase as a function of time of the total line (both limbs in series) for case (A). The resistance of the long limb does not change with time, so that the resistance increase is solely due to voiding in the short limb. Figure 3.7 (b) shows the resistance increase as a function of time for the short limb for case (B). The resistance increases for the short and long limb for case (C) are shown in figure 3.7 (c). Finally, the resistance increase for the long limb for case (D) is shown in figure 3.7 (d), and the resistance increase for the short limb for case (D) is shown in figure 3.7 (e). In all four cases it was verified by post-mortem inspection that the passivation was intact, and no metallic extrusions had formed. Focussed-ion-beam microscopy (FIB) on a representative subset of the samples revealed that the voids formed at the electron source via.

3.4.2 Simulation Results

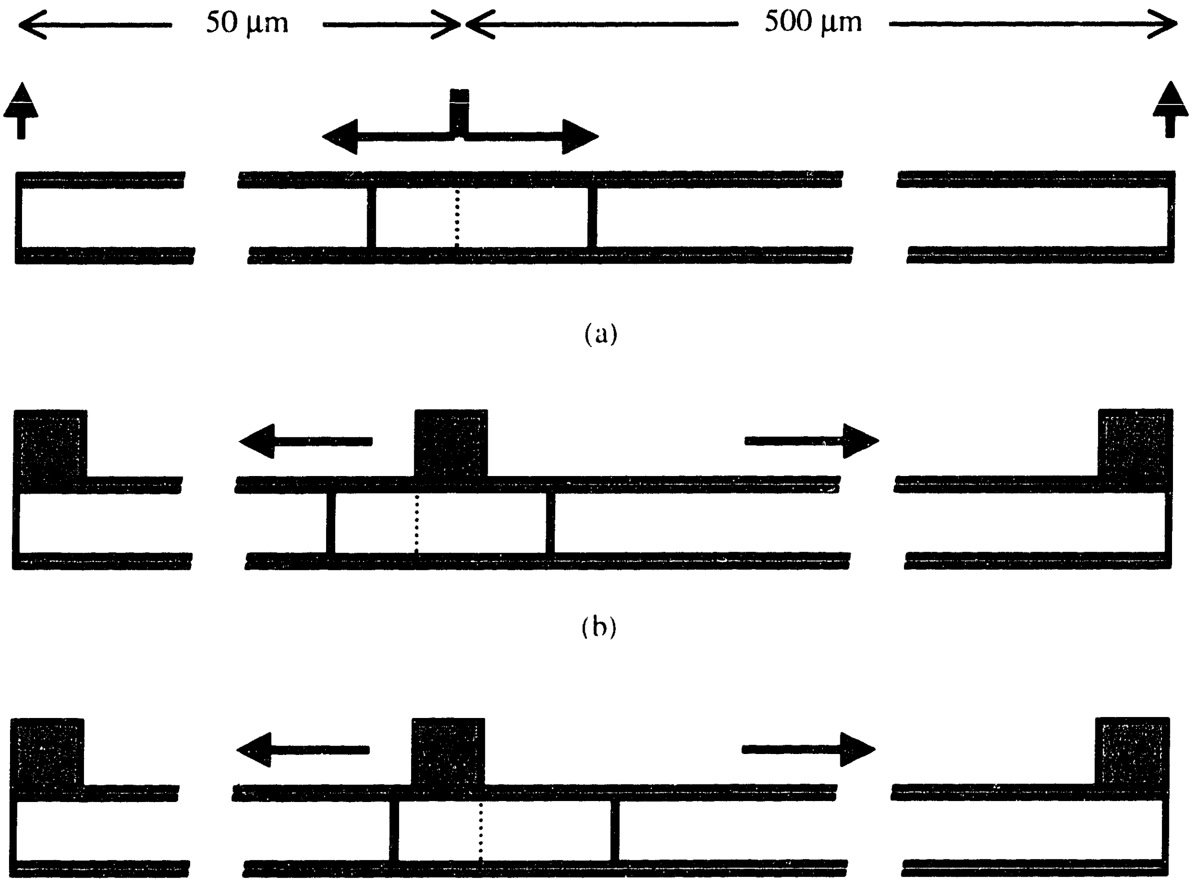
Figure 3.8 shows stress profiles for different times obtained using simulations for cases (A) to (D). Numerically positive stresses are tensile. It was assumed that a critical tensile stress of 350 MPa leads to voiding [Hau 00b]. Once the void passes the via, the resistance of the interconnect increases. The resistance increase as a function of time is shown in Figure 3.7.

3.4.3 Discussion

We will first discuss case (A), in which the electron current is constant along the line. The W-filled vias at the line ends are sites of atomic flux divergence, leading to the build up of hydrostatic stresses as shown in figure 3.8 (a). After a short incubation time, the tensile stress exceeds the critical stress necessary for void formation at the cathode via, and a void nucleates. Our simulations show that the void length initially increases linearly with time, but then increases sublinearly (as shown in figure 3.7 (a)) when the nonuniform stress fields at the ends pf the line begin to interact (figure 3.8 (a)). We observed this saturation of the resistance increase in experiments, as can be seen in figure 3.7 (a).

In case (B), no electrical current is flowing through the long limb of the tree. Compressive stresses build up around the center via, which is the anode in this case. The distance between the anode and cathode is only 50μm, so that the stress fields of the anode and cathode interact early in the evolution, as can be seen in figure 3.8 (b). However, the critical stress necessary for void nucleation is still reached at the cathode, so that a void nucleates and the resistance increases due to void growth, as shown in figure 3.7 (b). This is in agreement with the experimental results also shown in figure 3.7 (b). The rate of void growth is much slower than in case (A) because; (i) the electromigration wind force is nearly balanced by the back stress force, which leads to a nearly linear stress profile, and (ii) diffusion into the long limb is solely stress-driven.

In case (C), electrons flow toward the center via in both limbs, so that compressive stresses build up faster near the center via than in case (B). The stress profile in the short limb evolves toward a linear, steady-state profile before the critical tensile stress necessary for void nucleation is reached at the cathode, as can be seen in figure 3.8 (c). In agreement with experimental results shown in figure 3.7 (c), the resistance of the short limb does not change with time. As can be seen in figure 3.8 (c), atoms are continuously pushed from the long limb into the short limb, so that the stresses in the short limb eventually become compressive, but the stress profile stays nearly linear. At the cathode end of the long limb, the tensile stresses quickly become large enough to cause void formation, leading to an increase in the resistance which continues until void growth saturates, as shown in figures 3.7 (c).



(a)

(b)

(c)

Figure 3.9: The void positions (a) obtained from simulation, and (b) and (c), as observed in experiments. The full arrows indicate the electron flow, and the dotted line indicates the site of void nucleation. The void nucleates under the via, which is the site of the largest tensile stress and largest flux divergence. The void grows into the short and long limbs until saturation. The simulation conservatively predicts the void size because it is assumed that the via is infinitesimally small.

Finally, in case (D), electrons flow through the middle via toward both outer vias of the interconnect. A tensile hydrostatic stress builds up at the middle via until a void nucleates, as shown in figure 3.8 (d). The void grows into the long limb as well as into the short limb, until the back stress in the short limb becomes large enough to suppress further growth of the void, as shown in figure 3.7 (c). Void growth starts to saturate in the long limb much later, so that a larger resistance increase is observed in the long limb.

We believe that the experimentally observed variations of steady-state resistances of the short limb are due to variations in void shape and location. If the void nucleates in the short limb rather than in the long limb, the resistance is larger. The variation in the rate of approach to the steady state, as shown in figure 3.7 (e), is attributed to a variations in the void shape [Ata 98], which is not captured in the simplest version of our simulation.

To optimize performance of integrated circuits while retaining a high overall reliability, reliability assessments have to be conservative and as accurate as possible. With a single set of input parameters, our simulation can be used to make conservative predictions of the resistance increases associated with electromigration-induced void nucleation and growth. As shown in figure 3.7, the experimentally observed resistance increases are smaller than the resistance increases obtained through simulation. The simulation results are generally accurate, and in the cases in which they deviate from the experimental results, the simulation predicts the worst case resistance increase. This is the result of having built in worst case assumptions about the void locations in the simulation. For example, in case (D), the void is assumed to immediately span the line and to nucleate under the center via, and simultaneously grow into the long and short limbs. In the simulation, the vias are assumed to be infinitesimally small, as shown in figure 3.9 (a), so that the resistance of the short limb starts to increase immediately after void nucleation. In experiments, the voids can nucleate anywhere under the vias, as shown in figure 3.9 (b) and (c), and depending on the nucleation sites, different saturation resistances are observed for the short limbs. The largest saturation resistance occurs when a void nucleates at the edge of a via in the short limb, as shown in figure 3.7 (b), and this is the saturation resistance the simulation predicts. For Al-based metallization schemes, in the worst case, a shallow void forms under the via in such a way that the under-layer is no longer available to shunt current. However, our post-experiment FIB studies showed that voids fully span the failed interconnect, so that, given this information about void shapes, the simulation makes a conservative prediction of the resistance increase.

3.4.4 Summary and Conclusions

Previous studies of electromigration in simple interconnect tree structures with a single current source have shown that the Korhonen model can be used to predict the correct steady-state resistances in straight lines [Fil 96] and simple trees [Ric 98a, Cle 99, Gle 99]. Our experiments have shown that in addition, simulations based on the Korhonen model give valid and conservative predictions of the rate of electromigration-induced damage for more complex current configurations when two constant current sources are present. Experimentally, we observed a distribution in the steady state resistances, which we attribute to variations in the locations of voids along the interconnect length. We also observed variations in the rates of approach to steady state, which we attribute to a variation in void shape. Our simulation, based on the one-dimensional Korhonen model and using a single set of materials parameters, accurately predicts the time evolution of resistance changes caused by electromigration-induced voiding, and, given information about void shapes, can be used to make accurate worst-case projections of failure rates.

3.5 Discussion: A Hierarchical Approach to Circuit-Level Reliability Analyses

In summary, we presented a computationally straight-forward and therefore efficient method for identifying trees which are immune to failure. This method can be seen as an extension of the critical current-density line-length product found in drift-experiments and resulting in the well-known Blech-length effect [Ble 76]. We verified the concept of immortality in interconnect trees in electromigration-saturation experiments on simple tree structures. It was demonstrated that a large fraction of trees in a typical design are immortal.

The analyses outlined above, and discussed in more detail in references [Rie 98a] and [Cle 99], provide a simple means of testing for immortality of interconnect trees, based on knowledge of the effective jL product, $(jL)_{eff}$, as defined in equation (3.6), that defines the threshold for immortality for interconnect trees. Applying this concept, a computationally-manageable approach for making circuit-level layout-specific reliability analyses can be suggested. We propose the following *Hierarchical Reliability Analysis:*

1. Identify trees from layout information, cataloging limb lengths and connectedness.
2. Assume that all the limbs are at the maximum allowed current density, j_{max}, so that $(jL)_{eff}$ can be calculated for each tree.
3. Filter immune interconnect trees which cannot fail even under worst-case conditions.
4. Make detailed estimation of j_i for each segment.
5. Filter immune interconnect trees by detailed calculation of steady-state stresses.

6. Do analyses of the time-dependent reliability of the remaining trees using electromigration models.

If after step 5, the tree is still not identified as immune, a time-dependent analysis of the stress in the tree is necessary. The Korhonen model [Kor 93] cannot be solved analytically in general, so numerical methods have to be applied [Kno 95a, Kno 97]. Software packages like MIT/EmSim [Ems] allow detailed calculations of stress evolution, including alloying effects and simulations of electromigration in trees, as discussed in Chapter 2, but these calculations are computationally intensive and should only be done for the least reliable trees. The reliability of the majority of the mortal trees can be assessed using a conservative default model based on the analysis of individual nodes (junctions, contacts, or vias) in trees. This approach is described in the next chapter. Alternatively, it should be possible to make minor changes in the layout to create immune trees, e.g. using metallization level switching through W-studs to bound trees at immune dimensions.

Chapter 4 Default Model for Tree Electromigration Reliability

4.1 Background and Overview

In a typical layout of an integrated circuit, a large number of interconnect trees can be identified as immortal and are filtered from further analyses, as described in chapter 3. Only those trees that pass through the filters and are thus identified as potentially mortal will require reliability estimates. For circuit-level analyses, the reliability of the mortal trees must be estimated by a conservative default model, so that the trees are ranked according to their reliabilities, and only the least reliable trees can be analyzed in further detail by using more time-consuming electromigration simulation tools [Ems], which are described in chapter 2.

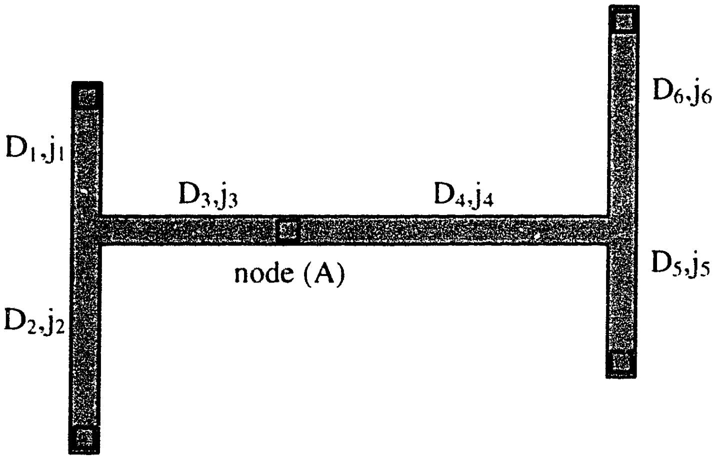


Figure 4.1: An example of an interconnect tree. In each limb i, the diffusivity is D_i and the current density is j_i.

In this chapter, we develop and apply a default model for estimating the reliability of arbitrarily complex interconnect trees based on the evolution of stress and void growth at vias, contacts, width transitions, and junctions (generally referred to as *nodes*) in trees. A part of a tree between two neighboring nodes is called a *limb*. For example, the tree in figure 4.1 is made up of 6 limbs. If we were to separate open a tree at a node, we would obtain several smaller trees, which are called *subtrees*. For example, node A in figure 4.1 connects two subtrees, where the first subtree consists of limbs number 1, 2, and 3, and the second subtree consists of limbs number 4, 5, and 6. Our default model allows the ranking of tree reliabilities in order to identify areas at risk to electromigration failure in laid-out circuits. We verify the validity of the default model through comparisons with simulations and through experiments on simple tree structures, such as lines broken into two limbs with different currents in each limb.

4.2 Description of Default Model

For a circuit-level reliability analysis, the reliability of mortal "I's" can be treated through direct analysis of test data. For other mortal trees, a conservative default model is needed. A simple default model can be based on the analysis of individual nodes in trees, because failure typically occurs at or near nodes. Failure can occur due to the resistance increase associated with voiding in the event of large tensile stresses, or due to cracks in the passivation and metallic extrusions in the event of large compressive stresses. Both failure modes are incorporated into our model. We will focus on voiding first, and we will consider extrusions later.

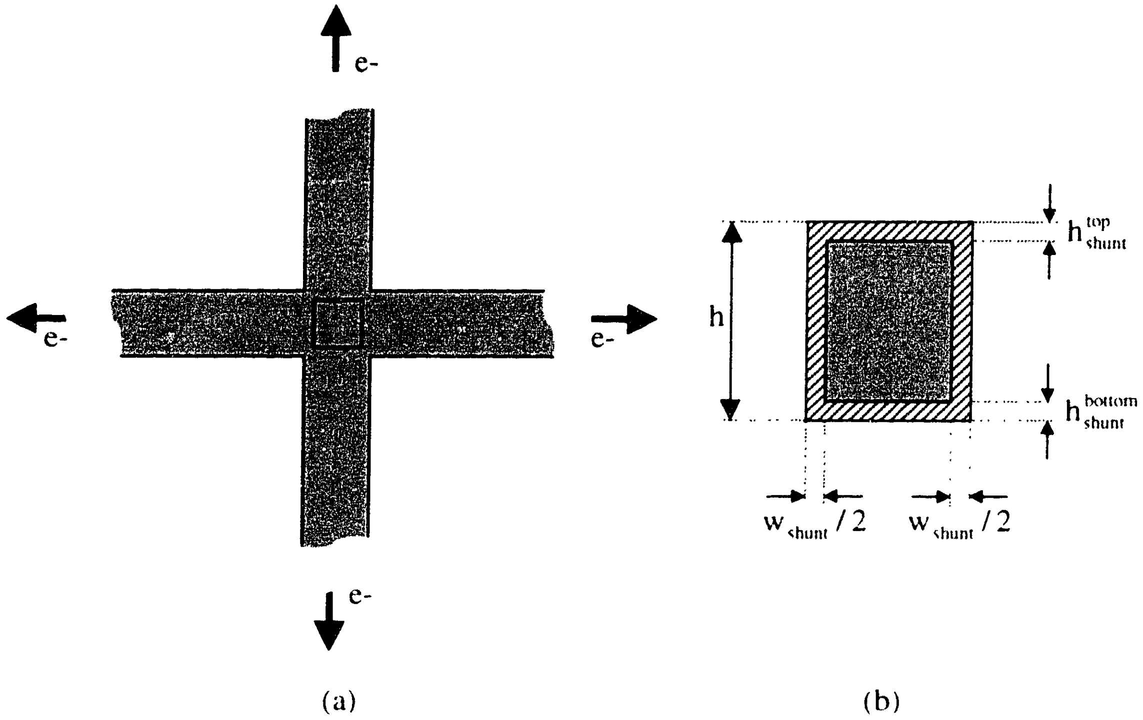


Figure 4.2: (a) A node draining into four semi-infinite limbs. (b) The cross section of an interconnect.

4.2.1 Overview of Junction-Based Reliability Analyses

The initial stress in the tree varies with line width [Hau 00z]. However, as a first order approximation, we assume that the stress distribution is initially uniform in the tree (i.e., σ_0 at time t = 0). Failure in trees due to voiding occurs in a sequence shown in figure 2.1 (a) to (d). When the tensile stress exceeds the critical stress necessary for void nucleation, σ_{nucl}, at time t_{nucl}, a void nucleates and starts growing. Eventually the size of the void leads to an unacceptably high resistance increase in one of the limbs at time t_{growth}, which is the time the node has failed. We will estimate t_{nucl} and t_{growth} separately, and we will take the longer of the two to be the time to failure due to voiding. Similarily, we will estimate the time for extrusions to form, $t_{extrusion}$. We will conservatively take the time to failure of the node, t_{fail}, to be the minimum of the time for failure due to voiding and the time to failure due to extrusions. t_{fail} is estimated for each node in the tree, and the smallest t_{fail} is taken to be the lifetime of the tree.

4.2.2 Hydrostatic Stress Evolution at the Intersection of Semi-Infinite Interconnects

Assuming a stress-independent diffusivity, we will calculate the hydrostatic stress evolution near an intersection of semi-infinite interconnects as sketched in figure 4.2 (a). The continuity equation for vacancies in the line direction gives the material balance as

$$\frac{\partial c_v}{\partial t}+\frac{\partial J_v}{\partial x}+\gamma=0, \tag{4.1}$$

where the rate of recombination/generation of vacancies, γ, is

$$\gamma=-\frac{\partial c}{\partial t}=\frac{c}{B}\frac{\partial \sigma}{\partial t}, \tag{4.2}$$

and where we have used dc/c = -dσ/B [Kor 93]. C is the concentration of lattice sites, and B is an effective elastic modulus. Using equations (1.3) and (4.2), the continuity equation becomes

$$\left(\frac{B\Omega}{kT}\frac{c_v}{c}+1\right)\frac{c}{B}\frac{\partial \sigma}{\partial t}+\frac{\Omega}{kT}\frac{\partial}{\partial x}\left(c_v D_v\left(\frac{\partial \sigma}{\partial x}-\frac{q^*}{\Omega}E\right)\right)=0. \tag{4.3}$$

Because $B\Omega c_v/(kTc) << 1$, taking D_v to be constant, and assuming that the vacancy concentration does not deviate significantly from the initial equilibrium concentration,

$$\frac{\partial\sigma}{\partial t} - D_{a0}\frac{B\Omega}{kT}\frac{\partial}{\partial x}\left(\frac{\partial\sigma}{\partial x} - \frac{q^*}{\Omega}E\right) = 0. \tag{4.4}$$

Now we consider an arbitrary number of semi-infinite interconnects intersecting at a node, as sketched in Figure 4.2 (a). We also assume that the diffusivity is independent of the stress. The electric field is constant within each limb, i, of the tree. With $K_i = D_i B\Omega/kT$, the Korhonen equation (1.7) becomes

$$\frac{\partial\sigma_i}{\partial t} = K_i\frac{\partial^2\sigma_i}{\partial x^2}, \tag{4.5}$$

where σ_i is the hydrostatic stress in limb i. The stress distribution σ_i in each limb satisfies equation (4.5) subject to the boundary conditions

At x=0 (intersection) : $\sigma_1=\sigma_2=\sigma_3=\ldots,$ (4.6)

and $\sum_i J^i_{\mathrm{atoms}} = 0,$ (4.7)

while at x=∞ : $\sigma_0=\sigma_1=\sigma_2=\sigma_3=\ldots,$ (4.8)

and at t=0 : $\sigma_i=\sigma_0$ everywhere. (4.9)

Here σ_0 is the initial hydrostatic stress in the tree, and

$$J^{i}_{\text{atoms}} = -\frac{D_{a0}c_i}{kT}\left(\nabla\mu_i + q^{*}E_i\right) \tag{4.10}$$

is the atomic flux in limb i at the intersection, and μ_i is the chemical potential function in limb i [Her 50]. The Laplace transform L[f] of a function f(x,t) is defined by [Abr 74]

$$L[f]\ (x,s) \equiv \int_0^{\infty} f(x,t)\exp(-st)\ dt \qquad (t \geq 0). \tag{4.11}$$

Taking the Laplace transform of equation (4.5) gives [Abr 74]

$$sL[\sigma_i] - K_i\frac{\partial^2 L[\sigma_i]}{\partial x^2} = \sigma_0, \tag{4.12}$$

where $L[\sigma_i]$ is the Laplace transform of σ_i. The boundary conditions (4.7) transform to

$$\sum_i D_i \left.\frac{\partial L[\sigma_i]}{\partial x}\right|_{x=0} = \frac{K}{s}, \tag{4.13}$$

with $K = \sum_i D_i j_i \rho q^{*}/\Omega$. The function

$$L[\sigma_i](x,s) = A\exp\left(-\sqrt{\frac{s}{K_i}}x\right) + \frac{\sigma_0}{s} \tag{4.14}$$

solves equation (4.12) and satisfies the boundary conditions (4.6) and (4.8). Substituting L[σ_i] into equation (4.13) gives

$$A = \frac{\alpha\beta}{s^{2/3}}, \tag{4.15}$$

with

$$\alpha = \frac{\rho q^*}{\Omega}\sqrt{\frac{B\Omega}{kT}} \tag{4.16}$$

and

$$\beta = \frac{\sum_i D_i j_i}{\sum_i \sqrt{D_i}}. \tag{4.17}$$

The inverse Laplace transformation results in [Abr 74]

$$\sigma_i(x,t) = \alpha\beta\left(\sqrt{\frac{4t}{\pi}}\exp\left(-\frac{x^2}{4K_i t}\right) - \frac{x}{\sqrt{K_i}}\operatorname{erfc}\left(\frac{x}{\sqrt{4K_i t}}\right)\right) + \sigma_0. \tag{4.18}$$

At the node at x=0,

$$\sigma(t) = \sqrt{\frac{4t}{\pi}}\frac{\rho q^*}{\Omega}\sqrt{\frac{B\Omega}{kT}}\frac{\sum_i D_i j_i}{\sum_i \sqrt{D_i}} + \sigma_0 \tag{4.19}$$

The distance in which the finite stress change σ_i-σ_0 extends from the node into the limb is proportional to $\sqrt{K_i t}$. At

$$l_c = 4\sqrt{K_i t} \qquad (4.20)$$

the change in stress falls below 1% from the stress at the node.

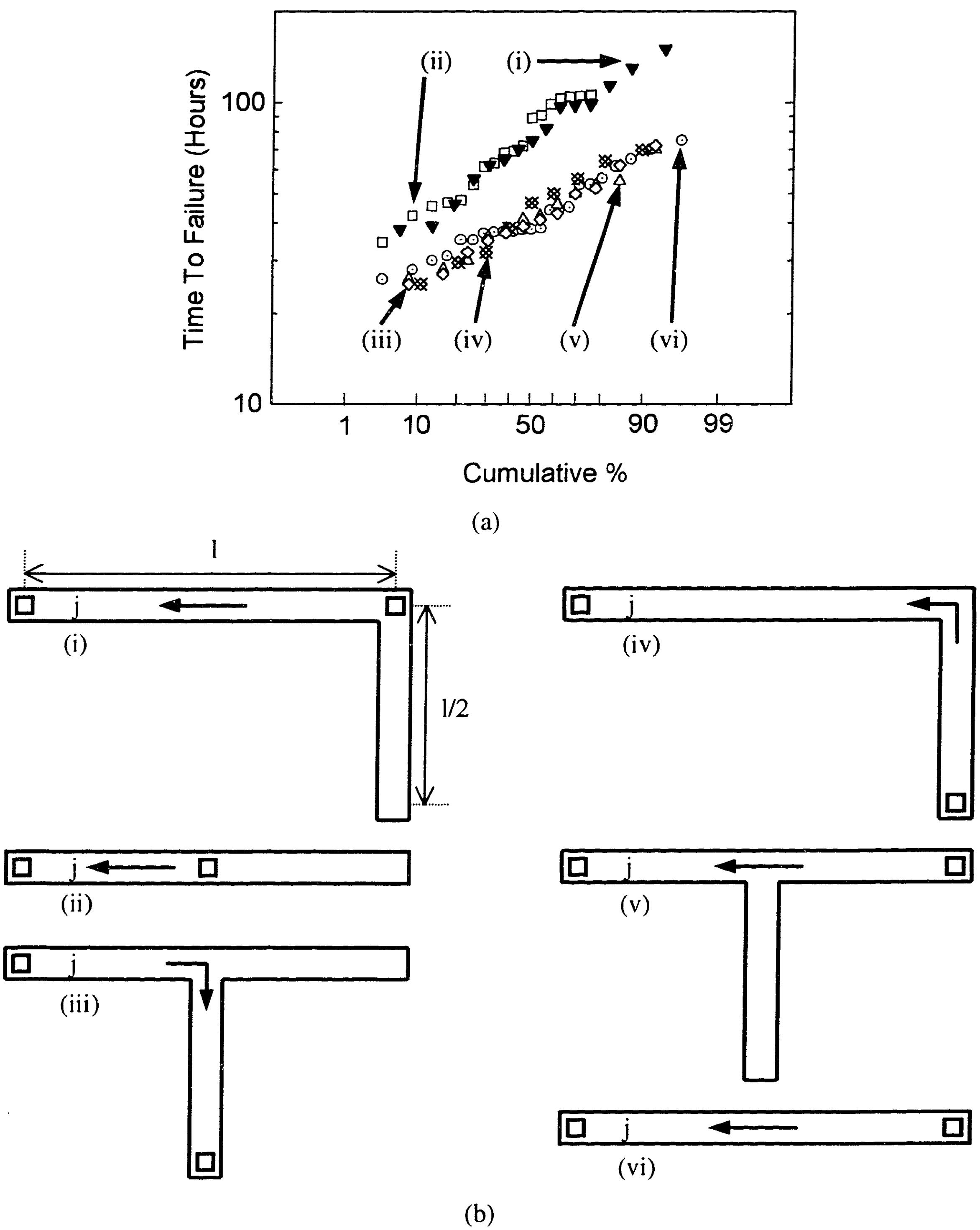
4.2.3 Failure Due to Void Nucleation

The effect of each limb in the tree can be described by its diffusivity D_i, current density j_i, and length l_i. We will estimate the time for void nucleation, t_{nucl}, for a node connecting n subtrees. As shown above, the stress evolution at a node draining into semi-infinite limbs, as shown in figure 4.2 (a), is given by equation (4.19). Assuming a constant and time-independent diffusivity along the limb, the stress increase, $\Delta\sigma$, at the node is proportional to

$$\frac{\sum_{i=1}^{n} D_i j_i}{\sum_{i=1}^{n} \sqrt{D_i}}. \qquad (4.21)$$

We will treat the case of stress-dependent and therefore time-dependent diffusivities, as well as the case of near-bamboo microstructures, in which the diffusivity varies spatially along the Al interconnects, in more detail later. For now, the diffusivity is assumed to be constant. An electron flow into a node slows the buildup of tensile stress, whereas an electron current out of a node can slow or accelerate the buildup of tensile stress, depending on the magnitude of the current. In order to use equation (4.19) and expression (4.21) to conservatively estimate the stress evolution at a node, it must be considered that the stress evolution is determined by the whole subtree rather than the limbs immediately connected to the node. We replace each subtree with a semi-infinite limb with a diffusivity and current density chosen from that limb of the subtree that maximizes expression (4.21). Taking the tree sketched in figure 4.1 (a) as an example, the worst-case stress evolution at node A is determined by the limbs i and j which maximize $(D_ij_i+D_jj_j)/(\sqrt{D_i}+\sqrt{D_j})$ with $1 \leq i \leq 3$ and $4 \leq j \leq 6$. The finite size of the subtree can lead to an overly optimistic reliability estimation only if the subtree slows the buildup of tensile stress. If the subtree is too small, the back stress due to tensile stresses at the end of the subtree inhibits atoms from flowing into the node. To estimate the onset of these back stress effects, we associate an effective length with the subtree, which is the maximum path length within the subtree, and compare the distance the stresses are extending into the subtree using equation (4.20) with the effective length. If back stress effects are present, the subtree is ignored, which is conservative. Once expression (4.19) is maximized, equation (4.19) can then be used to calculate t_{nucl} using $\sigma(t_{nucl}) = \sigma_{nucl}$.

In near-bamboo Al interconnects, the microstructure and therefore the diffusivity vary statistically along the interconnect. To obtain a truly conservative reliability estimate, the continuous range of possible diffusivities ranging from the diffusivity of fully-bamboo interconnects to the diffusivity of fully-polygranular interconnects has to be considered. Similarily, if the diffusivity is stress-dependent [Par 99], the diffusivity can range from the minimum diffusivity at the maximum compressive stress to the maximum diffusivity at the maximum tensile stress. Given the continuous range of diffusivities for each limb, expression (4.19) has to be maximized using numerical methods [Sto 80].



(a)

(b)

Figure 4.3: (a) Times to failure, defined as a 30% increase in electrical resistance, for 0.27μm-wide lines tested at T = 250°C and j = $2x10^6$ A/cm^2 for tree geometries and electron current configurations shown in (b). The line length, l, is 500μm.

4.2.4 Failure Due to Void Growth

Now we will estimate t_{growth}. Conservatively, we assume the void is present near the node from the beginning of the circuit operation, and we assume that the void spans the interconnect. The local net atomic flux at the node will determine the size of the void. The void length is given by

$$l_{void} = \frac{q^* \rho}{kT} t \sum_i D_i j_i \ . \tag{4.22}$$

Similar to the void nucleation case, D_i and j_i have to be chosen from a limb within each subtree connected to the node, so that the expression $\sum_i D_i j_i$ is maximized. In case all electron currents of a subtree are directed into the node, the most conservative assumption is that there is no current in the subtree. For the calculation of l_{void}, it does not matter in which limb next to the node the void nucleates. However, the resistance increase is related to the line width, w, of the limbs, and therefore has to be evaluated for the limb which maximizes the resistance increase, ΔR, of the limb, given by

$$\Delta R \approx \rho_{shunt} \frac{l}{wt_{shunt} + tw_{shunt}} - R^{sheet}_{Al/Cu} \frac{l}{w} \ . \tag{4.23}$$

ρ_{shunt} is the electrical resistivity of the shunt layer; h, w_{shunt}, and w are the line dimensions as sketched in figure 4.2 (b); and h_{shunt} = $h^{top}_{shunt} + h^{bottom}_{shunt}$. For aluminum-based metallization schemes, typically $w_{shunt} = 0$, and for copper-based metallization schemes,

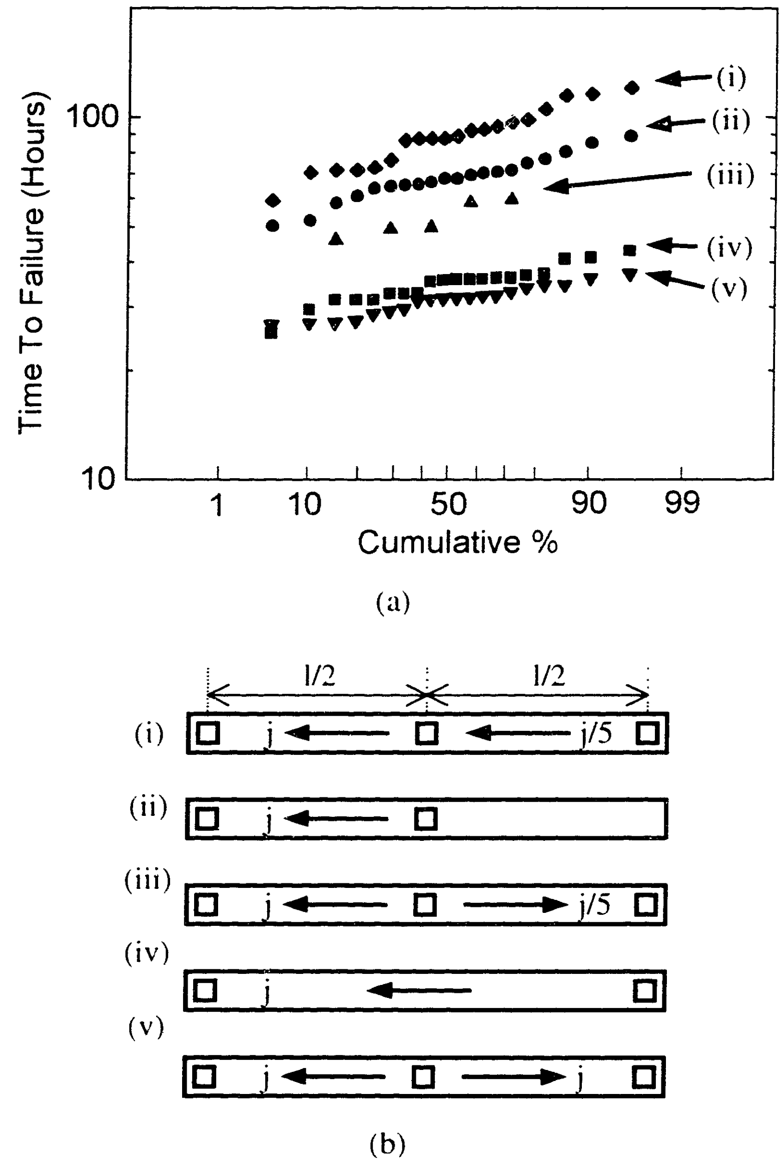
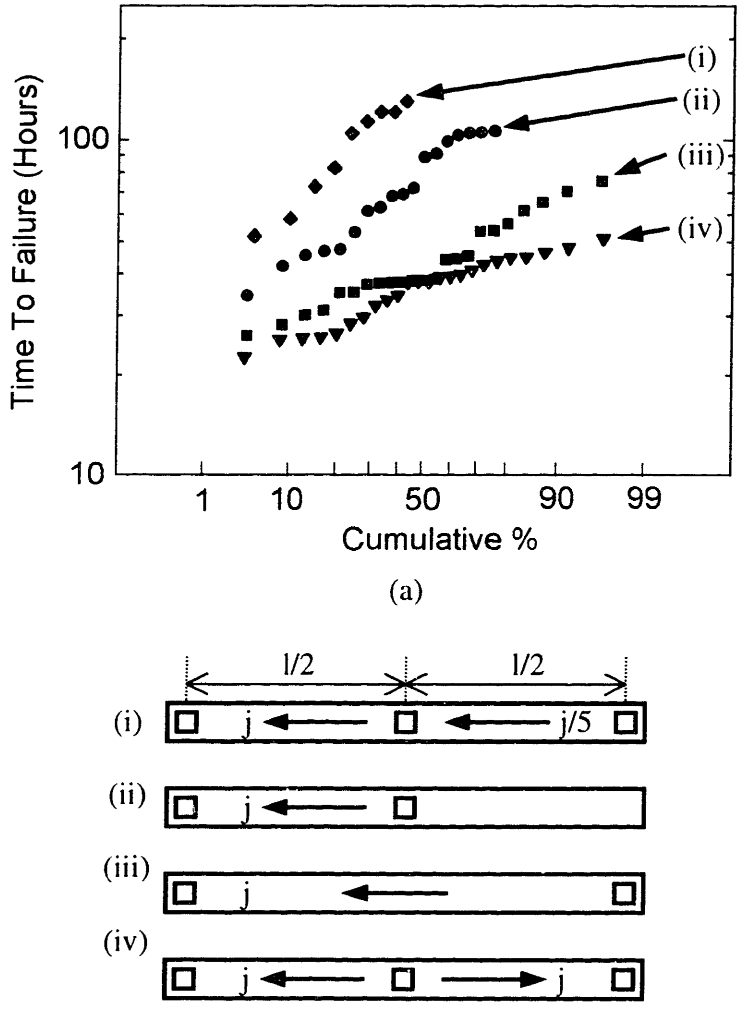


Figure 4.4: (a) Times to failure, defined as a 30% increase in electrical resistance, for 3.0μm-wide lines tested at T = 250°C and j = 1×10^6 A/cm^2 for electron current configurations shown in (b). The line length, l, is 500μm.

typically $h^{top}_{shunt} = 0$. t_{growth} can be estimated with $\Delta R(t_{growth}) = \Delta R_{max}$, where ΔR_{max} is the maximum allowed resistance increase.

4.2.5 Failure Due to Passivation Cracking and Extrusions

The second failure mode, i.e. failure due to too high compressive stresses, can be treated in a similar way as failure due to void nucleation. The only difference to the previous analysis is that the compressive rather than the tensile stresses at nodes are estimated.



(a)

(b)

Figure 4.5: (a) Times to failure, defined as a 30% increase in electrical resistance, for 0.27μm-wide lines tested at T = 250°C and j = 2x10^6 A/cm^2 for electron current configurations shown in (b). The line length, l, is 500μm.

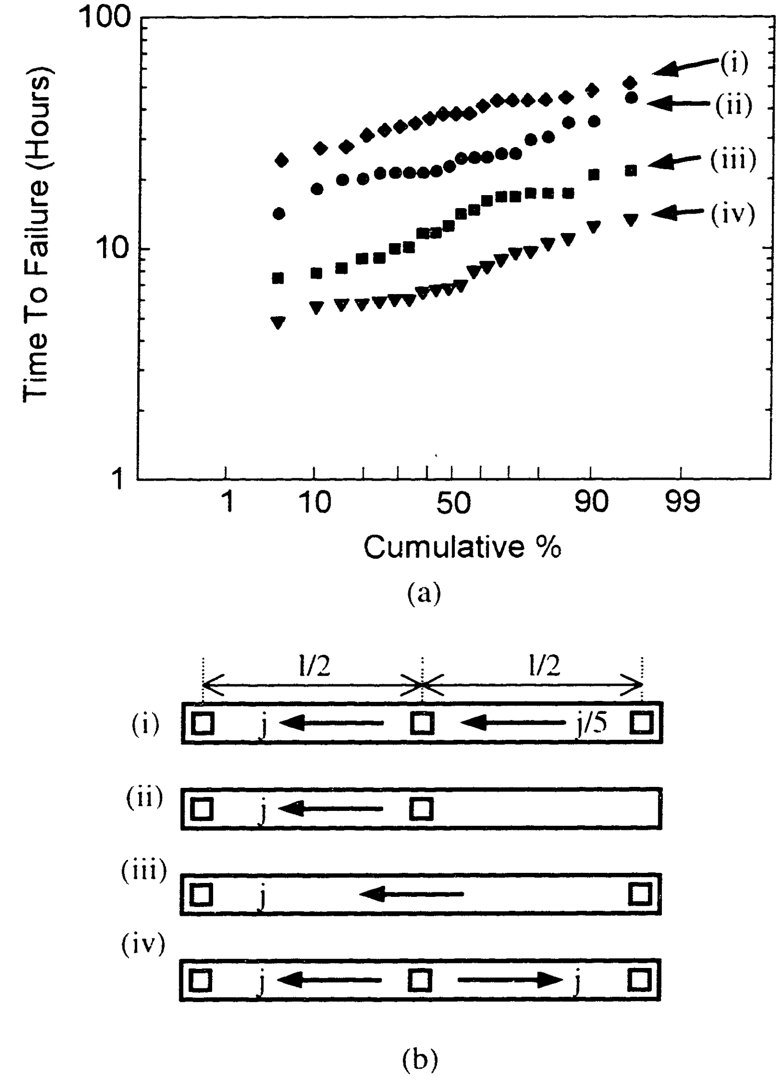


Figure 4.6: (a) Times to failure, defined as a 30% increase in electrical resistance, for 0.27μm-wide lines tested at T = 350°C and j = 4x10^5 A/cm^2 for electron current configurations shown in (b). The line length, l, is 500μm.

4.3 Results on Electromigration Experiments on Junctions

Figure 4.3 (a) shows times to failure on a cumulative lognormal probability plot for 0.27 μm-wide "I", "dotted I", "L", and "T" structures stressed with a current density of 2x10^6 A/cm^2 at 250°C. On a lognormal plot, times that are lognormally distributed fall on

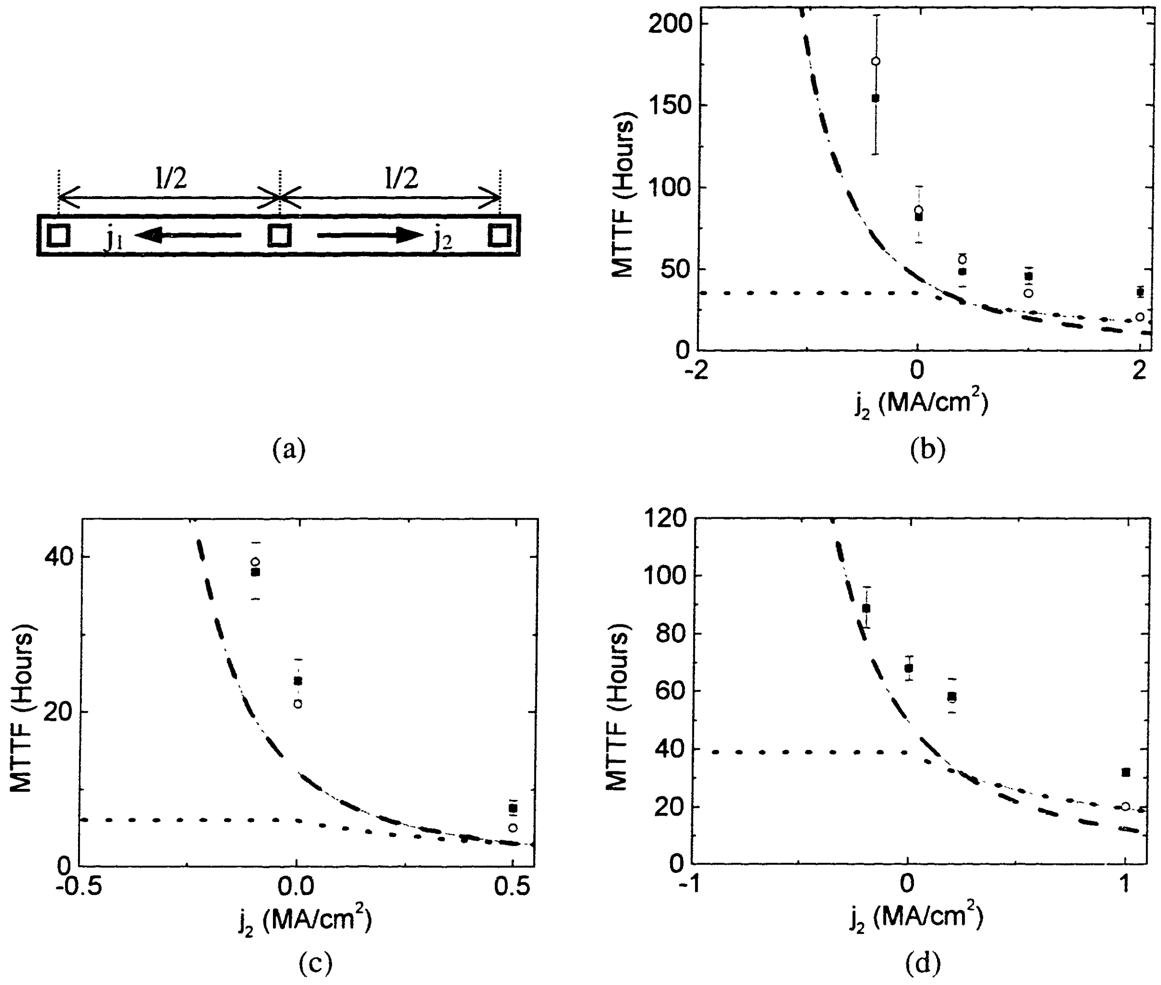


Figure 4.7: Comparison of experimental data with the default model for the structure shown in (a) with l = 500 μm, and (b) w = 0.27 μm, T = 250°C, j_1=2x10^6 A/cm^2, (c) w = 0.27 μm, T = 350°C, j_1 = 5x10^5 A/cm^2, and (d) w = 3.0μm, T = 250°C, j_1 = 1x10^6 A/cm^2. The dashed lines show the calculated times for void nucleation, t_{nucl}, the dotted lines show the times for void growth, t_{growth}, and the continuous lines show the estimated times to failure taken as the maximum of t_{nucl} and t_{growth}. Overlaid are experimentally obtained median times to failure represented by solid square symbols and error bars indicating a 95% confidence interval. Also overlaid are times to failure obtained through simulations, represented by open circles.

a straight line. The line length l is 500 μm. Failure is defined by a 30% increase in resistance, as it is in all the following experiments. In this experiment, we tested the effects of the presence and orientation of inactive metal limbs ("passive reservoirs"), as well as the effect of line bends on the reliability of interconnects.

We have varied the current configuration near vias to investigate the effects on nodal reliability. Figure 4.4 shows times to failure of 3.0 μm-wide and 500 μm-long "dotted I" structures stressed with a constant current density $j_1 = 1x10^6$ A/cm^2 in the left limb, and a current density with varying direction and magnitude, j_2, in the right limb at 250°C. We fit a lognormal distribution to the times to failure, and the median times to failure as a function of j_2 are shown in figure 4.7 (d). Figures 4.5 and 4.6 show times to failure of 0.27 μm-wide and 500 μm-long "dotted I" structures stressed with $j_1 = 2x10^6$A/cm^2 at 250°C and with $j_1 = 5x10^5$A/cm^2 at 350°C, respectively. The median times to failure as a function of j_2 are shown in figures 4.7 (b) and (c).

4.4 Discussion

We verified the default model through comparison to electromigration experiments on structures with the shape shown in figure 4.7 (a). For the different experiments, j_1 was kept constant, whereas j_2 was varied in magnitude and direction. The grain size of the as-deposited films is about 1μm, so after post-patterning annealing the 0.27μm-wide lines (annealing occurs during passivation, packaging, and pre-electromigration testing) are fully bamboo, so we do not consider the effects of the alloy additions or non-bamboo structures [Par 99]. The median times to failure were evaluated from the times to failure shown in figures 4.5 and 4.6, and are plotted as a function of j_2 in figures 4.7 (b) and (c), respectively. Also shown in figure 4.7 are the calculated t_{nucl} with σ_{nucl} = 500 MPa at 250°C, and σ_{nucl} = 350 MPa at 350°C. The nucleation stresses were obtained by matching experimental times to failure with simulation results. As in the experiment, t_{growth} was evaluated for a 30% resistance increase. The comparison of experiments with the models shows that the default models predict the lifetime of the center node conservatively, and that the right functional dependence on j_2 is predicted. Similar experiments were performed on lines with widths w = 3.0 μm, so that the lines were fully polygranular. The median times to failure were evaluated from the times to failure shown in figure 4.4, and are plotted in figures 4.7 (d). Also shown in these figures are the calculated t_{nucl} assuming σ_{nucl} = 500 MPa. Again, the comparison shows that the default model predicts the lifetime of the center node conservatively, and that the right functional dependence on j_2 is predicted.

The orientation of the limbs meeting at nodes as well as bends in limbs of the tree are not considered for the reliability assessment. This is justified by the results of lifetime experiments we performed on lines with bends as well as lines with passive reservoirs which meet at nodes at different angles, as it is shown in figure 4.3 (a).

4.5 Conclusion

An electromigration model for estimating the reliability of arbitrarily complex interconnect trees has been developed based on nodal reliability analyses. The time for void nucleation is estimated, and compared to the time for a void to grow to a size large enough to lead to a unacceptably large resistance increase. The longer time is taken as the lifetime of the node due to voiding. The time for the formation of extrusion is similarily estimated. The weakest node determines the lifetime of the tree. A comparison of the models with simulations and experiments shows very good agreement.

Chapter 5 Summary and Future Work

5.1 Summary of Results

In today's Si IC technology, several meters of metal interconnects are required to build a single high-performance circuit, so that in each IC many millions of metal segments exist. These metallic circuit elements are a great reliability concern owing mainly to electromigration [Ble 66]. This concern increases with the level of integration, with each new generation of Si technology requiring the use of a larger number of narrower interconnects, stressed at higher current densities.

Present design strategies rely on overdesign of interconnects to ensure that a circuit is immune to electromigration failure. The current process of applying unnecessarily conservative and inflexible design rules is no longer feasible when pushing the limits in IC performance for a given technology. Interest has increased in developing new techniques for making realistic reliability assessments *during* the design and layout process (RCAD), so that analyses concerning reliability can be fed back into the design and layout process immediately. Reliability analyses applied in an RCAD framework can take into account the details of a layout and of circuit operation in order to achieve optimum performance while retaining high overall reliability. This requires that reliability assessments be made frequently for a vast amount of interconnect, so that a computationally efficient and flexible strategy for assessment of interconnect reliability is necessary. This leads to the concept of hierarchical reliability analyses in an RCAD framework. Simple tesıs for interconnect immortality can be used in a hierarchical way to eliminate interconnect structures from further increasingly more computationally intensive reliability assessments.

Electromigration-induced phenomenology can be accounted for using the Korhonen model for electromigration [Kor 93] which tracks the effects of the electron-wind force, the back stress, and the action of vacancy sources and sinks. The Korhonen model was originally proposed for straight-line interconnects. However, laid-out IC's often have interconnects with junctions. For carrying out circuit-level reliability assessments it is important to assess the reliability of these more complex shapes, generally referred to as trees. We identified the interconnect tree as the fundamental reliability unit, and extended the Korhonen model to elbow-shaped interconnects as well as interconnect structures with intersections.

We have also extended the understanding of "immortality" demonstrated and analyzed for straight stud-to-stud lines, to trees of arbitrary complexity. The introduction of an effective jL product for trees provides a simple means for testing for immortality, which leads to the filtering of a large number of interconnect trees in a layout of a typical circuit from further analyses. We verified the concept of immortality in trees by performing experiments on saturation effects in trees.

To assess the reliability of mortal trees, we simulated the effect of electromigration in interconnects using our electromigration simulator MIT/EmSim [Ems], which is a 1D-electromigration simulation based on the Korhonen model [Kor 93]. We extended MIT/EmSim to allow the simulation of the effects of electromigration in interconnect trees. To validate the simulation, we performed electromigration lifetime experiments on simple interconnect tree structures. Models, simulations and experimental results on the reliability of interconnect trees were shown to be consistent.

However, these simulations are computationally intensive, and would best be reserved for the few least reliable trees. For other mortal trees, we have suggested a conservative default model based on the analysis of individual nodes in trees. We have validated our models and simulations through comparisons with experiments on simple trees, such as lines broken into two segments with different currents in each segment.

5.2 Implications of Results

Through completion of a full set of analytic models for hierarchical filtering of immortal trees, and for assessment of the reliability of mortal trees, this work enabled the development of a prototype tool for carrying out circuit-level reliability assessments. This tool is called ERNI (electromigration reliability of networked interconnect) and can be used with the layout tool Magic (or the Java-based version, Majic) to provide a reliability estimate for a specific circuit, and to provide a list of mortal trees, ranked according to their reliability using the default models. The trees identified as least reliable can be shipped to MIT/EmSim for more accurate, but computationally intensive, reliability estimates. The layout of these trees might also be modified and the circuit reliability reassessed. Also, the effects of modifications in the processing of interconnects or in materials selection or dimensions (e.g. liner thickness) can be assessed.

5.3 Future Directions

5.3.1 Reliability Assessment of Integrated Circuits

Our models and simulations are based on direct-current (DC) analyses. However, the vast majority of interconnects in CMOS IC's do not work under DC conditions, but carry bidirectional current signals and pulsed DC signals [Mai 89]. Maiz et al. [Mai 89] suggested that an *equivalent DC current*, which is a DC current that produces the same electromigration damage as the corresponding non-DC current, can be calculated by taking the time-average of the non-DC current. It has been shown experimentally [Tow 83, Mai 89] and through simulations [Cle 97] that "I"-shaped interconnects under repetitive pulsed DC stress behave similar to "I" structures stressed at DC with a magnitude that equals the average of the pulsed DC signal. We expect that our models are applicable for alternating currents through the use of equivalent direct currents, however, this concept applied to interconnect trees still needs to be demonstrated experimentally.

Although alternating currents tend to induce less electromigration-induced damage than direct currents [Mai 89], they still cause significant self-heating of interconnects [Sch 87], which accelerates electromigration. The amount of Joule heating can be estimated based on the RMS-value of the non-DC current [Mai 89]. The self-heating can also lead to temperature gradients, which lead to gradients in the atomic diffusivity, which, in turn, can be the source for atomic flux divergencies inducing failure. Our tool for circuit-level reliability assessment should be extended to calculate the heat generation due to Joule heating in interconnects, at vias, and at contacts, so that the design can be optimized to minimize temperature gradients. This becomes especially important if 3D integration schemes are pursued in the future [Kam 99]. In this case, interconnects are shorter, leading to less signal delay and improved reliability. However, power dissipation is difficult in these structures, and Joule heating is a major reliability concern.

5.3.2 Cu-Based Metallization and Low-K Dielectrics

We are in the midst of a major technology transition from aluminum alloys to copper as the metallization material for high performance integrated circuits [Ede 97]. This transition is driven primarily by the associated reduction of the electrical resistance of the metallization [Mur 95], which allows significantly higher clock speeds even as total interconnect lengths continue to increase [Ede 97]. Aluminum-based interconnects are fabricated using a subtractive process involving deposition of continuous aluminum films and subsequent patterning using reactive ion etching. Copper interconnects are fabricated by filling trenches in the dielectric which are lined with refractory-metal-based diffusion barriers. The Cu is patterned through the use of chemical mechanical polishing (CMP) to remove the Cu between trenches. This process allows the fabrication of very narrow and deep interconnects. SiO_2 is the current interlevel-dielectric material of choice. To further decrease the wiring delay, SiO_2 is being replaced with materials with lower dielectric constants, which are generally referred to as low-k dielectrics. Low-k dielectrics can have very different mechanical properties than SiO_2, which affects electromigration, as discussed in Appendix B. These effects and predictions such as those discussed in Appendix B should be investigated in experiments.

We have shown through experiments and modeling that the concepts of active and passive reservoirs, and steady state in interconnect trees leading to immortality, are valid for Al-Cu interconnects. However, while it is expected that similar phenomena should occur in Cu-based interconnects, this should be confirmed through experiments. In Cu-based metallization schemes, liners and diffusion barriers are shunt layers for the electron current if voiding occurs. Shunt layers are kept thin to reduce the electrical resistance of interconnects, but a reduction in thickness leads to large current densities in the shunt layers when voiding occurs, which, in turn, induces significant self-heating. In future work, experiments on the dynamics and steady state of electromigration damage in Cu-based interconnects should be carried out. In addition, a minimum shunt-layer thickness should be determined, at which electromigration saturation without liner degradation can be achieved.

The introduction of Cu as a new interconnect material also leads to new failure mechanisms. Cu diffuses rapidly in silicon and degrades semiconductor devices, because Cu has acceptor levels near the middle of the silicon bandgap, and thus acts as an effective recombination-generation center for charge carriers. If the liners or diffusion barriers are not able to sustain large electromigration-induced stresses, Cu atoms can escape the trench and degrade the device performance, or Cu can form dendrites leading to short circuit. In future work it must be shown if and how this phenomena affects the default model and the ability to achieve a steady state in interconnect trees.

Appendix A Simulation of Electromigration in Interconnect Trees Using MIT/EmSim

A.1 Modifications of MIT/EmSim

A.1.1 Introduction

The electromigration simulator MIT/EmSim was modified to simulate electromigration-induced evolution of hydrostatic stresses and voids in interconnect trees. Prior to these modifications, MIT/EmSim could only be used to simulate stress and void evolution in single, straight interconnects terminating in pads or vias ("I" structures). These building elements are called "strands". In the simulation, a strand is broken up into a sequence of cells, and each cell interacts, via the exchange of atoms, with its neighbor on the left ("west") and on the right ("east"). The modifications have been made in four steps: (i) Junctions are introduced into the simulation. Junctions are special cells that can not only interact with neighbors on the west and on the east side, but also with neighbors on the north and on the south sides. Junctions are not part of a strand. An example of the connectedess of junctions and strands is shown in figure A.1. (ii) Junctions are allowed to interact with the heads and tails of strands. (iii) Junctions are allowed to interact with each other. (vi) Voids can nucleate inside junctions. I would like to acknowledge the help of Vab Andleigh, with whom I collaborated especially on the junction-junction interaction (step (iii)). Instead of including a listing of the whole MIT/EmSim source code, only the key changes will be described. The complete source code can be found under version 1.20q in the MIT/EmSim source library.

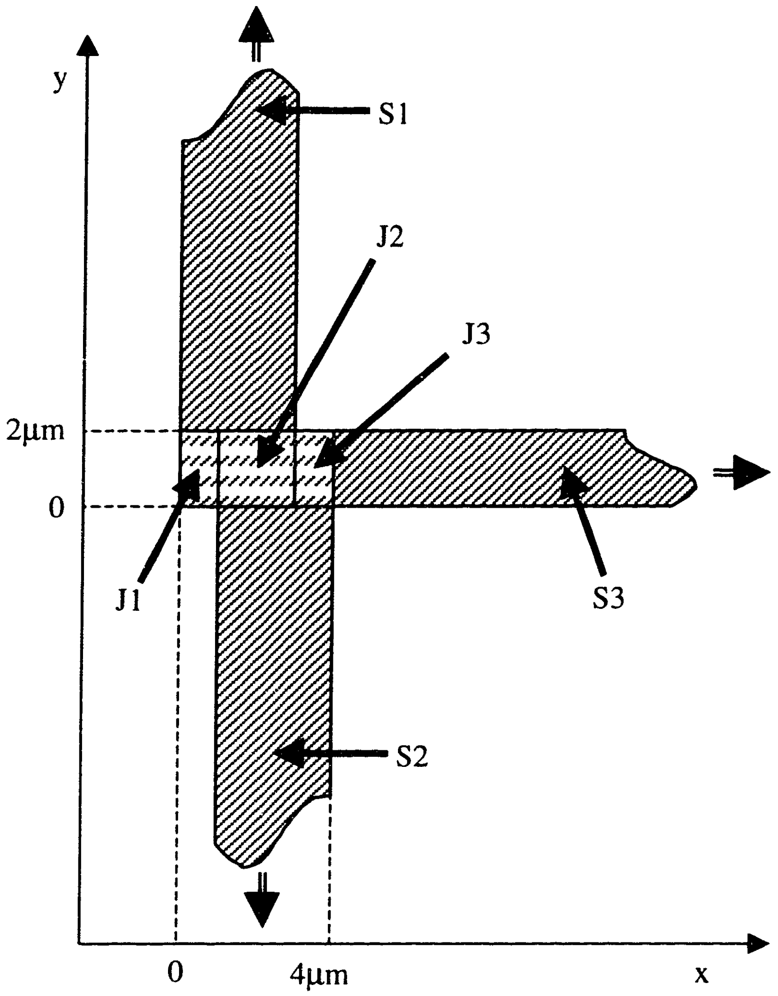


Figure A.1: An example of an interconnect tree. The double arrows indicate the direction of the electron flow. In all strands, the current density is j = 1×10^6A/cm^2. J1, J2, and J3 are junction cells, and S1, S2, and S3 are strands.

The data structure for **strand** variables is

```
struct Strandtype {
  int id;                  /* identification # for this strand    */
  Cellnode *head;          /* pointer to first cell of this strand */
  Cellnode *tail;          /* pointer to last cell of this strand  */
  int leftbc;              /* boundary condition at start of strand*/
  int rightbc;             /* boundary condition at end of strand  */
  Pointtype upleft;        /* x,y coordinates of upper left corner */
  Pointtype lowright;      /* x,y coordinates of lower right corner*/
  int orientation;         /* whether strand is vertical or horiz. */
  double start_x;          /* x position of start of the strand    */
  double length;           /* length in meters of strand           */
  double current;          /* electrical current throughout strand */
  char profile_fn[TEXTLEN];   /* stress_vs_x output file name          */
  double time_to_print[MAX_TTP]; /* array to hold time_to_print        */
  int ttp_counter;            /* array index value to time_to_print    */
  struct Strandtype *prev;    /* pointer to previous strand in line    */
  struct Strandtype *next;    /* pointer to the next strand in line    */
} ;                           /* strand node in between intersections */
```

The **strand** data structure provides pointers to the first and last cell of a strand, it specifies the boundary conditions for the head and tail cell, and it defines the coordinates of the strand. It also contains several variables sent to an output, to allow monitoring of the simulation.

A.1.2 Junctions in MIT/EmSim

The data structure for **junction** variables is

```
struct Junctiontype {
  int id;                     /* identification # of junction node       */
  struct Neighbortype *neigh; /* neighbors to north, east, south, west   */
  Alnode al;                  /* Al data for junction cell               */
  Cunode cu;                  /* Cu data for junction cell               */
  Stressnode stress;          /* stress data for junction cell           */
  double ave_x;               /* average position in the x direction     */
  double ave_y;               /* average position in the y direction     */
  double widthx;              /* width of junction in one direction      */
  double widthy;              /* width of junction in other direction    */
  double thick;               /* thickness of junction cell              */
  double jx;                  /* current density in x direction, e->East  */
  double jy;                  /* current density in y direction, e->North */
  struct Junctiontype *north; /* link to north junct neighbor if exists  */
  struct Junctiontype *east;  /* link to east junct neighbor if exists   */
  struct Junctiontype *south; /* link to south junct neighbor if exists  */
  struct Junctiontype *west;  /* link to west junct neighbor if exists   */
  struct Junctiontype *prev;  /* link list pointer to previous junction  */
  struct Junctiontype *next;  /* link list pointer to the next junction  */
  FILE *fp;                   /* file pointer to this junction outpt file */
  int    void_status;         /* voided? empty?                          */
} ;                           /* junction node                           */
```

In addition to the elements which are also part of the **cell** data structure, the **junction** data structure contains a current density **jx**, which is the current density out of the junction on the east, and a current density **jy**, which is the current density out of the junction on the north. The incoming currents on the west and on the south side are defined by the currents from the junctions and strands on the west and south side, respectively. The other components are described in the following sections.

A.1.3 Junction-Strand and Junction-Junction Interactions

The **neighbor** data structure is

```
struct Neighbortype {
  int id;                        /* identification # of neighbor node       */
  Cellnode *cell;                /* pointer to neighbor cell next to junction*/
  Strandnode *strand;            /* pointer to strand next to junction      */
  int location;                  /* says whether at head(+) or tail(-)      */
  Alcutype flux_in;              /* Al/Cu flux - cell has only "in" flux    */
  double start_x;                /* starting x-value for this neighbor      */
  double delta_x;                /* length of junction cell in strand direct.*/
  double ave_x;                  /* average x-value in strand direction     */
  Alcutype flux;                 /* flux of Al,Cu out of this junction      */
  double minarea_in;             /* minimum area of surface at junction     */
  int neightype;                 /* whether is strand or junction or none   */
  struct Neighbortype *prev;     /* link list pointer to previous direction */
  struct Neighbortype *next;     /* link list pointer to next direction     */
} ;                              /* structure for interaction between ends  */
```

It describes the interaction of a junction with a strand. In the **junction** data structure described above, ***neigh** points to a linked list of neighbors, and in the **neighbor** data structure, ***strand** links to the strand, and ***cell** links to the cell with which the junction interacts. The variable **minarea_in** is the area through which junction and strand are interacting.

Pointers to the north, east, south, and west sides called ***north**, ***east**, ***south**, and ***west**, respectively, in the **junction** data structure facilitate junction-junction interaction.

A.1.4 Calculation of the Stress Evolution in Junctions

A set of procedures located in the source file *emsim_trees.c* handles the stress evolution in junctions. The key functions are *calc_junction* and *update_junction*. In *calc_junction*, the change of the numbers of Al and Cu atoms in junctions is calculated during each time step:

```
/***-------------------------------------------------------------------------***/
void calc_junction(Junction *_junction, Difftype _diff,
                   Chem_2coeff_type _chem_coeff, double _cu_segr_gb,
                   double _delta_t, double _vac_diff, Alcutype _elec_coeff,
                   double _Q_vac_fm, double _kappa, double _omega, double _kT,
                   double _b, double *_maxstress, double *_minstress)
/***
Calculates fluxes at junction cells for Al, Al-Cu lines.  Constructs
a new strand linked list to send to function calc_bi_flux() since it
assumes the tree is Al-Cu (it handles both Al and Al-Cu cases).
The head of each strand is the neighbor cell while the tail of
the two cell strand is the juntion cell. After the
function call, it calculates the effects on # of Al,Cu atoms and
stress.
***/
{
  int create_strand;             /* boolean whether create strand*/
  Cellnode *cell,                /* pointer to cell reference    */
           *newcell,             /* pointer to new head cell     */
           *neighcell,           /* pointer to cell on neighbor  */
           *junctcell;           /* pointer to junction cell copy*/
  Strandnode *strandptr,         /* pointer to strand reference  */
             *newstrand;         /* temporary strand linked list */
  Neighbornode *neighbor,        /* pointer to neighbor reference*/
               *neigh_ref;       /* neighbor ref. for spec.strand*/
  Junction *junctionptr,         /* pointer to junction reference*/
           *neighjunc,           /* junc neighbor for junc-junc  */
           *junct;               /* pointer to junction reference*/
  Alcutype flow;                 /* # Al,Cu atoms into junction  */

  /*** loop through all the junctions ***/
  for(junctionptr=_junction;junctionptr!=NULL; junctionptr=junctionptr->next) {

    /*** create NEW strand linked list (strand copy) ***/
    newstrand = malloc(sizeof(Strandnode));
    newstrand->prev = NULL;
    strandptr = newstrand;
    strandptr->id = NONE;

    /*** loop through all the neighbors ***/
    for(neighbor = junctionptr->neigh;neighbor!=NULL;neighbor=neighbor->next) {

      /*** create strand if neighbor is strand or N/E junction neighbor ***/
      create_strand = NO;
      if( neighbor->neightype == STRANDBC )
        create_strand = YES;
```

```
if( neighbor->neightype == JUNCTIONBC )
  if( (neighbor->id == NORTH) || (neighbor->id == EAST) )
    create_strand = YES;

/*** only create this strand copy node if the neighbor strand exists ***/
if( create_strand == YES ) {

  /*** create NEW strand if necessary, otherwise use above strand ***/
  if( strandptr->id != NONE ) {

    /*** allocate memory for new strand ***/
    strandptr->next = malloc(sizeof(Strandnode));

    /*** fix linked list of newstrand copy ***/
    strandptr->next->prev = strandptr;

    /*** advance to newly created strand ***/
    strandptr = strandptr->next;
  }

  /*** write strand id# ***/
  strandptr->id = neighbor->id * -1;

  /*** allocate memory for strand head (NEIGHBOR) cell ***/
  newcell = malloc(sizeof(Cellnode));
  strandptr->head = newcell;
  newcell->prev = NULL;

  /*** if NEIGHBOR is STRAND, define newcell based on neighbor cell ***/
  if( neighbor->neightype == STRANDBC ) {

    /*** set neighbor cell to either head or tail of neighbor strand ***/
    if( neighbor->location == HEAD )
      neighcell = neighbor->strand->head;
    else {
      if( neighbor->location == TAIL )
        neighcell = neighbor->strand->tail;
      else {
        neighcell = NULL;
        break;
      }
    }

    /*** define properties for this newcell as neighbor strand cell ***/
    newcell->id = neighcell->id * -1;
    newcell->al = neighcell->al;
    newcell->cu = neighcell->cu;
    newcell->stress = neighcell->stress;
    newcell->minarea = neighcell->minarea;
    newcell->ave_x = neighcell->ave_x;
    newcell->delta_x = neighcell->delta_x;
    newcell->width = neighcell->width;
    newcell->thick = neighcell->thick;
    newcell->j = neighcell->j;  /* unused by bi_calc_flux */

  }  /* end of strand neighbor definition */

  /*** else do JUNCTION NEIGHBOR instead ***/
  else if( neighbor->neightype == JUNCTIONBC ) {

    /*** determine which direction neighbor is to be used ***/
    if( neighbor->id == NORTH )
      neighjunc = junctionptr->north;
```

```
          else if( neighbor->id == EAST )
             neighjunc = junctionptr->east;
          else
            printf("ERROR in calc_junction\n");

          /*** copy properties of neighboring junction cell ***/
          newcell->id = neighjunc->id * -1;
          newcell->al = neighjunc->al;
          newcell->cu = neighjunc->cu;
          newcell->stress = neighjunc->stress;
          newcell->minarea.in = neighbor->minarea_in;

          if( neighbor->id == NORTH ) {
            newcell->width = neighjunc->widthx;
            newcell->ave_x = neighjunc->ave_y;
            newcell->delta_x = neighjunc->widthy;
          } else {          /* EAST */
            newcell->width = neighjunc->widthy;
            newcell->ave_x = neighjunc->ave_x;
            newcell->delta_x = neighjunc->widthx;
          }
          newcell->thick = neighjunc->thick;
          newcell->j = neighjunc->jx;  /* unusued by bi_calc_flux..see below */
        }

        /*** allocate memory for JUNCTION NODE cell ***/
        junctcell = malloc(sizeof(Cellnode));
        strandptr->tail = junctcell;

        /*** adjust linked list in this strand copy ***/
        newcell->next = junctcell;
        junctcell->prev = newcell;
        junctcell->next = NULL;


        /*** copy properties of JUNCTION NODE itself ***/
        junctcell->id = junctionptr->id * -1;
        junctcell->al = junctionptr->al;
        junctcell->cu = junctionptr->cu;
        junctcell->stress = junctionptr->stress;
        junctcell->minarea.in = neighbor->minarea_in;
        if( neighbor->id == NORTH ) {
          junctcell->width = junctionptr->widthx;
          junctcell->ave_x   =   newcell->ave_x   +   0.5   *   (newcell->width   +
junctcell->width);
          junctcell->delta_x = junctionptr->widthy;
        }       else { /* EAST */
          junctcell->width = junctionptr->widthy;
          junctcell->ave_x   =   newcell->ave_x   +   0.5   *   (newcell->width   +
junctcell->width);
          junctcell->delta_x = junctionptr->widthx;
        }
        junctcell->thick = junctionptr->thick;

        /*** define current density for the newly copied strand ***/
        if( neighbor->neightype == STRANDBC ) {

          /*** define current density, accounting for HEAD/TAIL direction ***/
          if( neighbor->location == HEAD )
            junctcell->j = -1.0 * neighcell->j;
          else                   /* TAIL */
            junctcell->j = neighcell->j;
        }
        else if( neighbor->neightype == JUNCTIONBC ) {

          /*** define current density depending on NESW direction ***/
          if( neighbor->id == NORTH )
            junctcell->j = -1 * junctionptr->jy;
          if( neighbor->id == EAST )
            junctcell->j = -1 * junctionptr->jx;
        }

      }        /* end of whether to create strand */
    }      /* end of looping thru all neighbors */

    /*** fix last link of this strand ***/
    strandptr->next = NULL;

    /*** do the FLUX CALCULATION for these cell pairs for the strand ***/
    calc_bi_flux(newstrand,   _diff,   _chem_coeff,   _elec_coeff,   _Q_vac_fm,
_vac_diff, _cu_segr_gb, _kappa, _omega, _kT, _b);


    /*** loop thru strandptr neighbors to find which matches strand copy ***/
    for(strandptr = newstrand;  strandptr != NULL; strandptr=strandptr->next) {

      /*** advance to first neighbor with strand ***/
      neighbor = junctionptr->neigh;
      neigh_ref = NULL;
      cell = NULL;
      while( (neighbor != NULL) && (neigh_ref == NULL) ) {
        if( neighbor->id == (strandptr->id * -1) )
          neigh_ref = neighbor;
        neighbor = neighbor->next;
      }

      /*** verify found right strand neighbor combo, write cell reference ***/
      if( neigh_ref != NULL ) {
        if( neigh_ref->neightype == STRANDBC )
          cell = neigh_ref->cell;
        else if( neigh_ref->neightype == JUNCTIONBC ) {
          if( neigh_ref->id == NORTH )
            junct = junctionptr->north;
          if( neigh_ref->id == EAST )
            junct = junctionptr->east;
        } else
        printf("Error in copying newstrand fluxes over in calc_junction...\n");
      }

      /*
        PRINT_INT(neigh_ref->neightype);
        PRINT_NUM(strandptr->head->flux.al.out);
      */

      /*** if STRAND NEIGHBOR, adjust cell atoms accordingly ***/
      if( neigh_ref->neightype == STRANDBC ) {

        /*** calculate flow Al,Cu atoms into junction,write to actual cell ***/
        if( neigh_ref->location == HEAD ) {
          cell->flux.al.in = -1 * strandptr->head->flux.al.out;
          cell->flux.cu.in = -1 * strandptr->head->flux.cu.out;
          neigh_ref->flux.al = -1 * strandptr->head->flux.al.out;
          neigh_ref->flux.cu = -1 * strandptr->head->flux.cu.out;
          flow.al = strandptr->head->flux.al.out * cell->minarea.in * _delta_t;
          flow.cu = strandptr->head->flux.cu.out * cell->minarea.in * _delta_t;
        }
        else {                  /* TAIL */
          cell->flux.al.out = strandptr->head->flux.al.out;
          cell->flux.cu.out = strandptr->head->flux.cu.out;
          neigh_ref->flux.al = strandptr->head->flux.al.out;
          neigh_ref->flux.cu = strandptr->head->flux.cu.out;
          flow.al = strandptr->head->flux.al.out *cell->minarea.out * _delta_t;
          flow.cu = strandptr->head->flux.cu.out *cell->minarea.out * _delta_t;
        }

        /*** calculate change in # Al,Cu atoms in junction ***/

        /* only remove atoms when there are atoms to remove from! */
        if( ((junctionptr->al.num_atoms.t+junctionptr->cu.num_atoms.t)<
             -1   *    SUBEMPTY_PERC   *   (junctionptr->al.num_atoms.init   +
junctionptr->cu.num_atoms.init))
            && (flow.al+flow.cu<0.0) ) {
          if( neigh_ref->location == HEAD )
            cell->flux.al.in   =   cell->flux.cu.in   =   neigh_ref->flux.al   =
neigh_ref->flux.cu = 0.0;
          else
            cell->flux.al.out   =   cell->flux.cu.out   =   neigh_ref->flux.al   =
neigh_ref->flux.cu = 0.0;
          flow.al = flow.cu = 0.0;
        }
        else {
          junctionptr->al.num_atoms.tdt += flow.al;
          junctionptr->cu.num_atoms.tdt += flow.cu;
        }
      }  /* end of if strand neighbor */

      /*** if JUNCTION NEIGHBOR, adjust junction atoms accordingly ***/
      if( neigh_ref->neightype == JUNCTIONBC ) {

        /*** calculate flow Al,Cu atoms between junctions, write data ***/
        flow.al  = strandptr->head->flux.al.out * neigh_ref->minarea_in;
        flow.al *= _delta_t;
        flow.cu  = strandptr->head->flux.cu.out * neigh_ref->minarea_in;
        flow.cu *= _delta_t;

        /*** adjust # Al, Cu atoms according to direction ***/
        if( (neigh_ref->id == NORTH ) || (neigh_ref->id == EAST ) ) {

          /* only remove atoms if there is something in the junction cells */
          if( (((junct->al.num_atoms.t+junct->cu.num_atoms.t)<
                -1  *    SUBEMPTY_PERC   *   (junct->al.num_atoms.init   +   junct-
>cu.num_atoms.init))
              && (flow.al+flow.cu>0.0))
              ||
              (((junctionptr->al.num_atoms.t+junctionptr->cu.num_atoms.t)<
               -1   *     SUBEMPTY_PERC    *    (junctionptr->al.num_atoms.init   +
junctionptr->cu.num_atoms.init))
               && (flow.al+flow.cu<0.0)) ) {
            flow.al = flow.cu = 0.0;
            strandptr->head->flux.al.out = strandptr->head->flux.cu.out = 0.0;
          }

          junct->al.num_atoms.tdt -= flow.al;
          junct->cu.num_atoms.tdt -= flow.cu;
          junctionptr->al.num_atoms.tdt += flow.al;
          junctionptr->cu.num_atoms.tdt += flow.cu;
          neigh_ref->flux.al = strandptr->head->flux.al.out;
          neigh_ref->flux.cu = strandptr->head->flux.cu.out;
          if( neigh_ref->id == NORTH ) {
            junct->neigh->next->next->flux.al = -1 * neigh_ref->flux.al;
            junct->neigh->next->next->flux.cu = -1 * neigh_ref->flux.cu;
          }
          if( neigh_ref->id == EAST ) {
            junct->neigh->next->next->next->flux.al = -1 * neigh_ref->flux.al;
            junct->neigh->next->next->next->flux.cu = -1 * neigh_ref->flux.cu;
          }
        }

      }  /* end of if junction neighbor */

    }    /* end of strandptr for loop */

    /*** deallocate strandptr from memory ***/
    strandptr = newstrand;
    while( strandptr != NULL ) {
      newstrand = strandptr->next;
      delete_strand(strandptr);
      strandptr = newstrand;
    }

  }    /* end of looping through all the junctions */

  /*** update all the junction cells ***/
  for(junctionptr=_junction;junctionptr!=NULL; junctionptr=junctionptr->next)
    update_junction(junctionptr, _b, _omega);

}    /** end of function calc_junction() **/
```

In the function *update_junction*, the stress changes in junctions due to accumulation or depletion of atoms are calculated:

```
/***------------------------------------------------------------------------***/
void update_junction(Junction *junction, double _b, double _omega)
/***
Takes a junction pointer and recalculates basic junction cell parameters
including:  num_atoms.init for Al and Cu, atomic fractions, and stresses
in the cell.
***/
{
  double cell_vol,                          /* cell volume           */
         init_num_atoms,                    /* initial # of atoms    */
         tot_num_atoms,                     /* total # of atoms      */
         num_atom_ratio;                    /* ratio tot#/init# atoms */

  /*** recalculate # atoms present in stress free case ***/
  cell_vol = junction->widthx * junction->widthy * junction->thick;
  tot_num_atoms = junction->al.num_atoms.tdt + junction->cu.num_atoms.tdt;
  junction->al.at_frac = junction->al.num_atoms.tdt / tot_num_atoms;
  junction->cu.at_frac = junction->cu.num_atoms.tdt / tot_num_atoms;
  junction->al.num_atoms.init = (cell_vol / _omega) * junction->al.at_frac;
  junction->cu.num_atoms.init = (cell_vol / _omega) * junction->cu.at_frac;

  /*** recalculate stress ***/
  init_num_atoms = junction->al.num_atoms.init + junction->cu.num_atoms.init;
  num_atom_ratio = tot_num_atoms / init_num_atoms;
  if( num_atom_ratio != 1.0 )
    junction->stress.tdt = -1 * log(num_atom_ratio)* _b;
  else
    junction->stress.tdt = 0.0;

  if( junction->void_status != JNO_VOID )
    junction->stress.tdt = 0.0;

}  /** end of function update_junction() **/
```

A.1.5 Nucleation and Growth of Voids in Junctions

If the tensile stress in a junction exceeds the critical stress necessary for void nucleation, a void nucleates and relieves the stress inside the junction. The status of the junction changes from **NO_VOID** to **VOIDED**. When all atoms are depleted from a junction, its status changes from **VOIDED** to **EMPTY**. In this event, the neighboring junctions begin to void. If all junctions connected to one end of a strand are empty, a void nucleates and grows into the strand. However, if the number of atoms in a voided junction exceeds the number of atoms at zero stress in an unvoided junction, the junction status is changed back to **NO_VOID**. If the **CONSERVATIVE** switch, located in *switches.c*, is activated, the simulation changes the status of a junction from **VOIDED** to **EMPTY** immediately. As described in Chapter 2, this guarantees that the predictions of void sizes are conservative if it is assumed that voids span the interconnect.

The nucleation and growth of voids in junctions are handled by the procedure *nucl_which_void_junction*, located in the source file *emsim_voids.c*:

```
/***-----------------------------------------------------------------------***/
void nucl_which_void_junction(Voidnode *_voids, Junction *_junction,
                     double _ten_sigma_crit, double _void_zs_len,
                     double _omega, double _b, int _void_mode)
/***
Loops through all the junctions and determines which junctions have reached
the critical stress for void nucleation.  Set junction to "void nucleated"
then.  If all atoms are depleted from a junction, the status changes to JEMPTY,
and voids nucleate in the neighboring junctions and strands.
***/
```

```
{

  Junction *junctionptr,*neighjunc,*junctionptr1;
  Neighbornode *neighbor,                     /* neighbor of junction pointer */
    *neighbor1;
  int i;
  Voidnode *newvoid;                          /* new void pointer       */
  int nucleate_void_flag;
  char ch[5];
  double   junct_numatoms,
    tot_num_atoms;

  /*** loop through all the junction cells ***/
  for(junctionptr= _junction; junctionptr!=NULL; junctionptr=junctionptr->next)
{
    if( junctionptr->void_status==JVOIDED ) {
      if(
#ifndef CONSERVATIVE
       (junctionptr->al.num_atoms.tdt + junctionptr->cu.num_atoms.tdt) <
          -1  *  SUBEMPTY_PERC  *  (junctionptr->al.num_atoms.init  +  junctionptr-
>cu.num_atoms.init)
#endif
#ifdef CONSERVATIVE
         1
#endif
         ) {

        /* junction is completely voided.  nucleate voids in all the adjacent
strands */

        junctionptr->void_status=JEMPTY; /* nucleated voids */
        printf("** junction %d emptied.\n",junctionptr->id);

        /* start voiding in neighboring junctions */
        for(neighbor  =junctionptr->neigh;  neighbor!=NULL;  neighbor=neighbor-
>next) {
          if( neighbor->neightype==JUNCTIONBC ) {
            /*
            ** The junction which is now emptied is connected to a junction.
            ** Void the neighboring junction.
            */
            switch( neighbor->id ) {
              case NORTH : neighjunc = junctionptr->north;break;
              case EAST  : neig..junc = junctionptr->east;break;
              case SOUTH : neighjunc = junctionptr->south;break;
              case WEST  : neighjunc = junctionptr->west;break;
              default    : printf("can't find direction.\n");scanf("%c",ch);
            }
            if( neighjunc->void_status==JNO_VOID ) {
              neighjunc->void_status = JVOIDED;
              printf("    junction %d starts voiding (stress %e).\n",neighjunc-
>id,neighjunc->stress.tdt);
            }
          } /* end JUNCTIONBC */
          else if( neighbor->neightype==STRANDBC ) {
            /*
            ** The junction which is now emptied is connected to a strand.
            ** Check if all the junctions this strand is ending into are
            ** emptied.  If so, nucleate a void.
            */
            nucleate_void_flag = YES;
            for(junctionptr1=          _junction;          junctionptr1!=NULL;
junctionptr1=junctionptr1->next)
```

```
              for(neighbor1 =junctionptr1->neigh; neighbor1!=NULL;
neighbor1=neighbor1->next) {
                /* see if this neighbor connects to the same strand */
                if( neighbor1->neightype==STRANDBC )
                  if( neighbor1->strand->id == neighbor->strand->id )
                    if( neighbor1->location == neighbor->location ) {
                      if( junctionptr1->void_status != JEMPTY ) {
                        printf("    connecting junction %d is not empty
yet.\n",junctionptr1->id);
                        nucleate_void_flag = NO;
                      }
                      else {
                        printf("      connecting junction %d also
empty.\n",junctionptr1->id);
                      }
                    }
              }
            /*
            ** nucleate a void
            */
            if( nucleate_void_flag == YES ) {
              printf("    nucleating void in strand %d.\n",neighbor->strand-
>id);

              /* reduce the number of atoms in the first cell (in which
              ** void nucleates) so that the void size is >0 */
#ifndef CONSERVATIVE
              neighbor->cell->al.num_atoms.tdt += junctionptr->al.num_atoms.tdt
/ 4.0;
              neighbor->cell->cu.num_atoms.tdt += junctionptr->cu.num_atoms.tdt
/ 4.0;
              junctionptr->al.num_atoms.tdt *= 3.0/4.0;
              junctionptr->cu.num_atoms.tdt *= 3.0/4.0;
#endif
              /*** insert void into voids linked list ***/
              if( _voids->id != NONE )
                newvoid = malloc(sizeof(Voidnode));
              else
                newvoid = _voids;
              insert_void_linklist(_voids, newvoid, neighbor->strand, neighbor-
>cell, _void_zs_len);
              sprintf(newvoid->fn, "");
              /*** nucleate full span void ***/
              nucleate_void(newvoid, neighbor->strand, neighbor->cell,
_void_zs_len, _omega, _b);
            }

          } /* end STRANDBC */
        } /* end loop over neighbors */
      }
    }

    else if( junctionptr->void_status==JNO_VOID ) {
      /*** check if tensile stress exceeded ***/
      if (((junctionptr->stress.tdt + MID_TOL* _ten_sigma_crit) >=
_ten_sigma_crit)){
        junctionptr->void_status = JVOIDED;
        printf("*** junction %d starts voiding (stress
%.2f>%.2f).\n",junctionptr->id,
              junctionptr->stress.tdt/1e6,_ten_sigma_crit/1e6);
      }
    }
    else if( junctionptr->void_status==JEMPTY ) {
      /*** check if junction is filled up again so that voids are filled ***/
      junct_numatoms = junctionptr->widthx * junctionptr->widthy * junctionptr-
>thick / _omega;
      tot_num_atoms    =    junctionptr->al.num_atoms.tdt    +    junctionptr-
>cu.num_atoms.tdt;
      if( tot_num_atoms > junct_numatoms ) {
        printf("*** junction %d healed.\n",junctionptr->id);
        junction->void_status = JNO_VOID;
      }
    }

  }    /* end of looping through strands */

} /** End of function nucl_which_void_junction() **/
```

A.2 Using MIT/EmSim with Trees

We describe the usage of the modified version of MIT/EmSim by considering an interconnect tree as sketched in figure A.1 as an example. The tree consists of three junction cells, labeled J1, J2, and J3, and three strands, labeled S1, S2, and S3. Only the relevant parts of the tree near the junctions are included in the figure. The strands extend further and eventually terminate in pads. Strands S1 and S3 have their heads at the junction, while strand S2 has its tail at the junction.

MIT/EmSim requires the input files *emsim.inp*, *geometry*, and *diffdata*, which are described in detail in the MIT/EmSim manual [Ems Man]. In the following part, we will only discuss the input values relevant for trees. An *emsim.inp* input file describing the general simulation parameters is

```
// Temperature at which the electromigration test is conducted at
TEST_TEMPERATURE
250
// Refers to highest temperature annealing step after metal deposition
PASSIVATION_TEMPERATURE
250
// 0 = linear elastic model
// 1 = power law elastic model
// 2 = plastic compliance model (Suresh input table)
THERMAL_STRESS_MODEL
1
```

```
// 0 = No voids
// 1 = Partial voids
// 2 = Full span voids
VOID_MODEL
2
// Maximum simulation time to calculate to (in hrs)
RUN_TIME
3e12
// 1 = tensile failure
// 2 = resistance failure
// 3 = void length failures (only for full span voids)
FAILURE_MODE
3
// Critical stress for void nucleation (MPa)
CRIT_TENSILE_STRESS
500
// Assumes dielectric failure at this stress (MPa)
CRIT_COMPRESSIVE_STRESS
50000
// This parameter is only used when the failure mode is 3 (um)
VOID_FAILURE_LENGTH
25.0
// This parameter is only used for a failure mode of 2 (%)
RESISTANCE_FAILURE_PCT
100.0
// 0% is pure Al, 0-10% is Al-Cu, and 100% is pure Cu (%)
PERCENT_COPPER
0.0
```

Relevant for trees is that the value of VOID_MODEL is 2, so that voids span the full width of the interconnect. CRIT_TENSILE STRESS determines the stress necessary for void nucleation in the junction cells.

In the *geometry* file, the connectedness of the strands and junctions as well as the currents flowing through them are defined. For the tree shown in figure A.1, the *geometry* file is

```
// Just count the total number of strands in your tree structure
NUM_STRANDS
3
-1
// Number your junctions and determine the neighbors with format:
// junction# north east south west
// 0 indicates no neighbor
// positive integer denotes strand_id, negative denotes junction_id
TREE_DEFINITION
1  1 -2  0  0
2  1 -3  2 -1
3  0  3  2 -2
-1
```

```
// Specify junction cell information here
// junction# ave_x ave_y widthx widthy thick currentx currenty
//  current defined to be positive in North and East directions
JUNCTION_DEFINITION
1 0.5 1.0 1.0 2.0 1.0 0.0 0.0
2 2.0 1.0 2.0 2.0 1.0 0.0 0.0
3 3.5 1.0 1.0 2.0 1.0 0.0 0.0
-1
// Strand specifications here of format:
// strand# start_x end_x current BC_start BC_end
// Boundary Conditions
// 0  = none
// -1 = stud
// -2 = pad
// -3 = void
// #  = junction # (positive) or a list of junctions in the
//                                form (# # ...)
STRAND_DEFINITION
1    2 100  3E-2 (1 2) -1
2 -100   0 -3E-2  -1  (2 3)
3    4 100  2E-2   3   -1
-1

STRAND1
LOCATION
0 2 3 100
XSTEPS
1.0
-1
WIDTH
3.0
-1
THICK
1.0
-1
SMALL_VOIDS
-1
VOIDS
-1
PRECIPITATES
-1
TIME_TO_PRINT
strand1_
1
100
1000
1e12
-1
```

This section is repeated for STRAND2 and STRAND3.

[...] Definitions for STRAND2 and STRAND3 are left out here. They look very similar to the definition of STRAND1.

```
// Properties:
// Metal or alloy symbol
// resistivity (ohm-cm)
// bulk modulus (MPa)
// Poisson's ratio (dimensionless)
// atomic volume (m^3)
// z*  (dimensionless)
METAL_PROPERTIES
AL
5e-6
5e4
```

```
0.31
1.66e-29
4
ALCU
5e-6
5e4
0.31
1.66e-29
4 12
CU
1.9e-6
1.2e5
0.33
1.19e-29
4
// shunt properties:
// shunt thickness (um)
// shunt resistivity (ohm-cm)
SHUNT_PROPERTIES
0.040
2e-4
```

The fields that are relevant for trees include: NUM_STRANDS, which defines the number of strands in the tree, and TREE_DEFINITION, JUNCTION_DEFINITION, and STRAND_DEFINITION. TREE_DEFINITION describes, for each junction, the junctions and/or strands to which it connects. JUNCTION_DEFINITION describes the geometrical size and location of each junction, as well as the current in the x- and y-direction. Finally, STRAND_DEFINITION defines the line lengths and the currents in each strand, and the junction the strand connects to, or if the strand is connected to a pad or via. If the strand is connected to several junctions, the connecting junctions must be enumerated in parenthesis.

Finally, in the *diffdata* file the ratio of the bamboo to the polygranular diffusivity are defined:

```
OFFSETS
0.000000 0.000000
STRAND1
0.01
-1
STRAND2
0.01
```

```
-1
STRAND3
0.01
-1
JUNCTION1
0.01 0.01
JUNCTION2
0.01 0.01
JUNCTION3
0.01 0.01
```

In this example, the ratio is 0.01. For junctions, the ratio can be set independently for the x and the y direction.

Appendix B The Effects of the Mechanical Properties of the Confinement Material on Electromigration in Metallic Interconnects

B.1 Introduction

Electromigration continues to be one of the most important reliability issues for integrated circuit metallization systems [Hu 99b]. Extensive experimental and theoretical studies have been carried out leading to an in-depth understanding of electromigration and electromigration-induced phenomena. Computer simulations, which are often based on the Korhonen model for electromigration [Kor 93], provide important and efficient tools for understanding and predicting the kinetics of electromigration [Par 99, Ems]. The Korhonen model was originally developed for SiO_2-embedded Al-based metallization systems for which grain boundaries provide the fastest diffusion paths for electromigration [Tho 93]. However, the increase in ratio of wiring delay to the intrinsic transistor delay has provided the motivation for the IC industry to move from aluminum-based interconnects embedded in SiO_2 to copper-based metallization systems with inter-level dielectrics (ILD) having lower dielectric constants, k, than SiO_2 [Ede 97]. The Korhonen model for electromigration cannot be applied to Cu-based metallization systems without modifications. For example, electromigration experiments on Cu interconnects have shown that diffusion takes place primarily along the Cu/passivation interface, regardless of the grain boundary structure [Hu 99a]. In addition, low-k ILD's are often polymer-based, and are mechanically much softer than SiO_2 [Pri 97]. Typical examples of polymer-based ILD's are fluorinated poly(arylene ether) (FLARE) [Lau 96], SiLK [Wae 99], and polyimide [Lok 99], all of which have Young's moduli that are more than an order of magnitude smaller than that of SiO_2. In this paper, we report on studies of the effects of changing the mechanical properties of the ILD along with changes of line aspect ratio and the presence of liner or barrier materials on the electromigration behavior of Cu interconnects.

Electromigration is electronic-current-induced atomic diffusion due to scattering events of flowing electrons with host atoms. As atoms electromigrate, volumes in which atoms accumulate develop more compressive stresses, while volumes from which atoms are depleted develop more tensile stresses. These stress changes can result in gradients in stress, and therefore also in the chemical potential, that lead to back diffusion in opposition to electromigration. Electromigration-induced stress evolution in interconnects has been successfully described by the Korhonen model, in which the hydrostatic stress, σ, evolves according to [Kor 93]

$$\frac{\partial \sigma}{\partial t} = \frac{\partial}{\partial x}\left[\frac{DB\Omega}{kT}\left(\frac{\partial \sigma}{\partial x} + \frac{Eq^*}{\Omega}\right)\right], \qquad \text{(B.1)}$$

where t is the time, D is the atomic self diffusivity, k is Boltzmann's constant, T is temperature, E is the electric field, q* is the effective charge, and Ω is the atomic volume. B is an effective elastic modulus which describes the relationship between a change in the number of available lattice sites per unit volume, dC, and a change in the hydrostatic stress, $d\sigma$, through

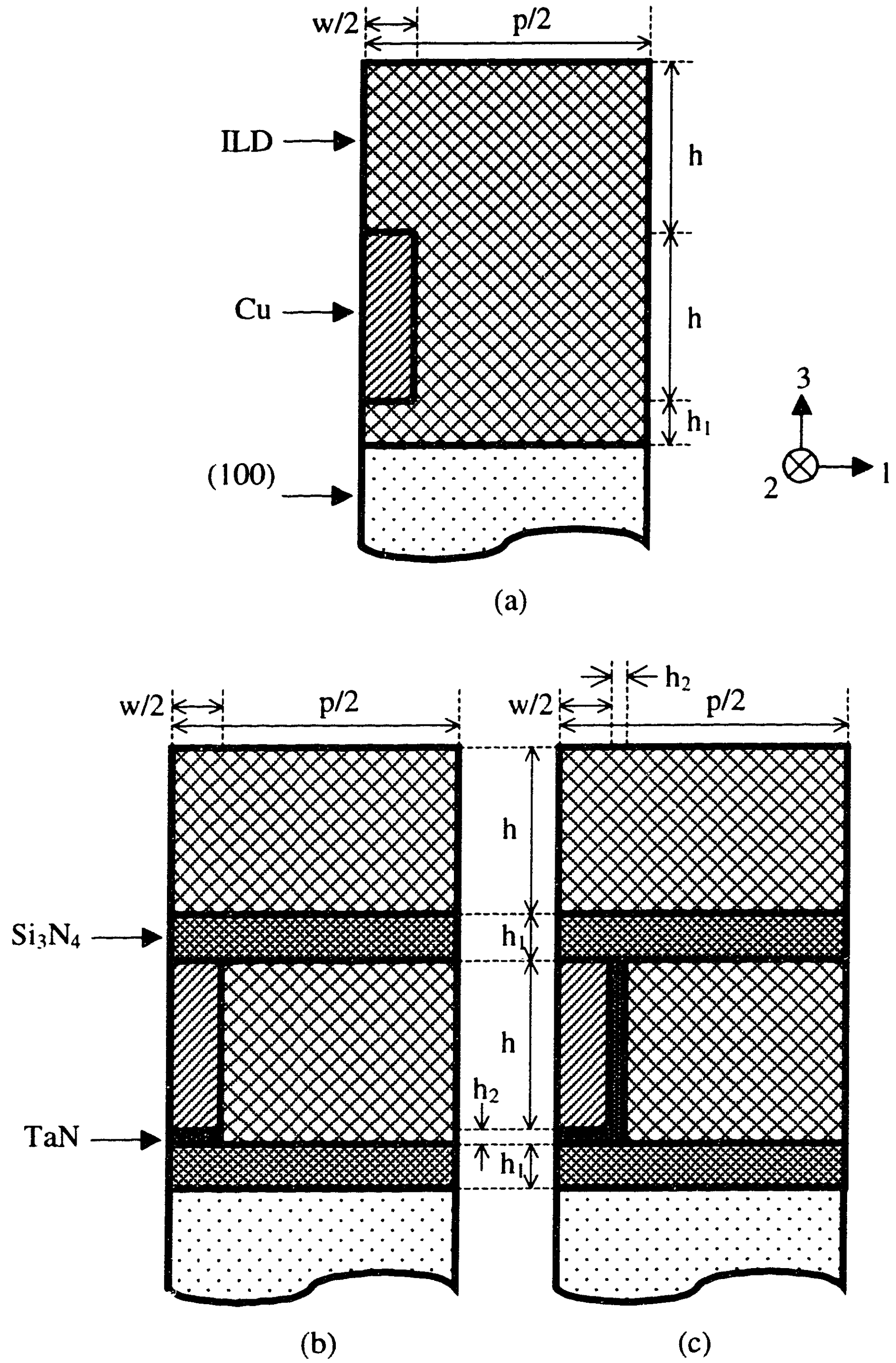


Figure B.1: Sketches of the geometry and coordinate system of the model. The finite element mesh is free to expand in the vertical direction. Mirror symmetry is applied horizontally, and translational symmetry is applied long the line in direction 2.

Material	Young's Modulus (GPa)	Poisson Ratio
Isotropic Cu	75.8	0.30
TaN	200.0	0.30
Si_3N_4	150.0	0.25
SiO_2	71.4	0.16

Table B.I: Mechanical properties used in the simulation.

$$\frac{dC}{C} = -\frac{d\sigma}{B}. \tag{B.2}$$

B has been calculated analytically for an elliptical aluminum interconnect embedded in an infinite SiO_2 or Si matrix [Kor 93], based on Eshelby's theory of inclusions [Esh 57]

A change of the mechanical properties of the ILD alters the degree to which electromigration-induced stresses build up in interconnects. In the Korhonen model for electromigration, this is accounted for by the effective modulus, B. In this paper, we analyze how B changes when the mechanical properties of the ILD are changed, by performing finite element analyses (FEM) of stress changes due to dilational strains in realistic interconnect structures. We also investigate the effects of the line aspect ratio, as well as the presence of liner or barrier materials on B. Finally, we discuss the effects of changes in B on electromigration and electromigration-induced failure.

B.2 Simulation Technique

Finite element calculations were performed using the commercial code Abaqus [Aba 98] with three-dimensional continuum-stress/displacement first-order-interpolation elements. The meshes were generated using our own mesh generation program with a refined mesh near edges and corners. The meshes were refined until a convergence of the stress distributions was achieved. The materials properties used in the simulations and shown in table B.I were chosen for a typical electromigration test temperature of 200°C. The silicon wafer was assumed to be oriented with a [100] surface normal and with elastic properties as listed in [Sim 71]. All materials were assumed to behave perfectly elastically. The geometry of the models representing three different metallization schemes are sketched in figures B.1 (a), (b), and (c). Due to symmetry, it is only necessary to model half the lines. The model sketched in figure B.1 (a) is closest to the geometry studied analytically by Korhonen et al. [Kor 93]. The thickness of the copper and the thickness of the top ILD, h, were taken to be 0.4 μm, the thickness of the Si_3N_4 underneath and on top of the Cu, h_1, was taken to be 0.1 μm, and the thickness of the liner material, h_2, was taken to be 35 nm. The substrate thickness, h_{wafer}, was taken to be 10 μm. The last assumption coupled with the prevention of wafer bending approximates an infinite substrate [Sau 92]. The errors in the calculations due to the reduced substrate thickness and the neglected wafer bending were calculated to be less than 5%. Three different line widths were modeled, w = 0.24 μm, 0.40 μm, and 1.0 μm. The pitch, p, of the structures was chosen such that $p/w \approx 9.3$, in order to avoid interaction of neighboring lines. We assumed perfect adhesion (traction) of Cu to the liner, ILD, and Si_3N_4. The coordinate axis 1 in figure B.1 is perpendicular to the length of the interconnect, the coordinate axis 2 lies along the length of the interconnect, and the coordinate axis 3 is perpendicular to the wafer surface.

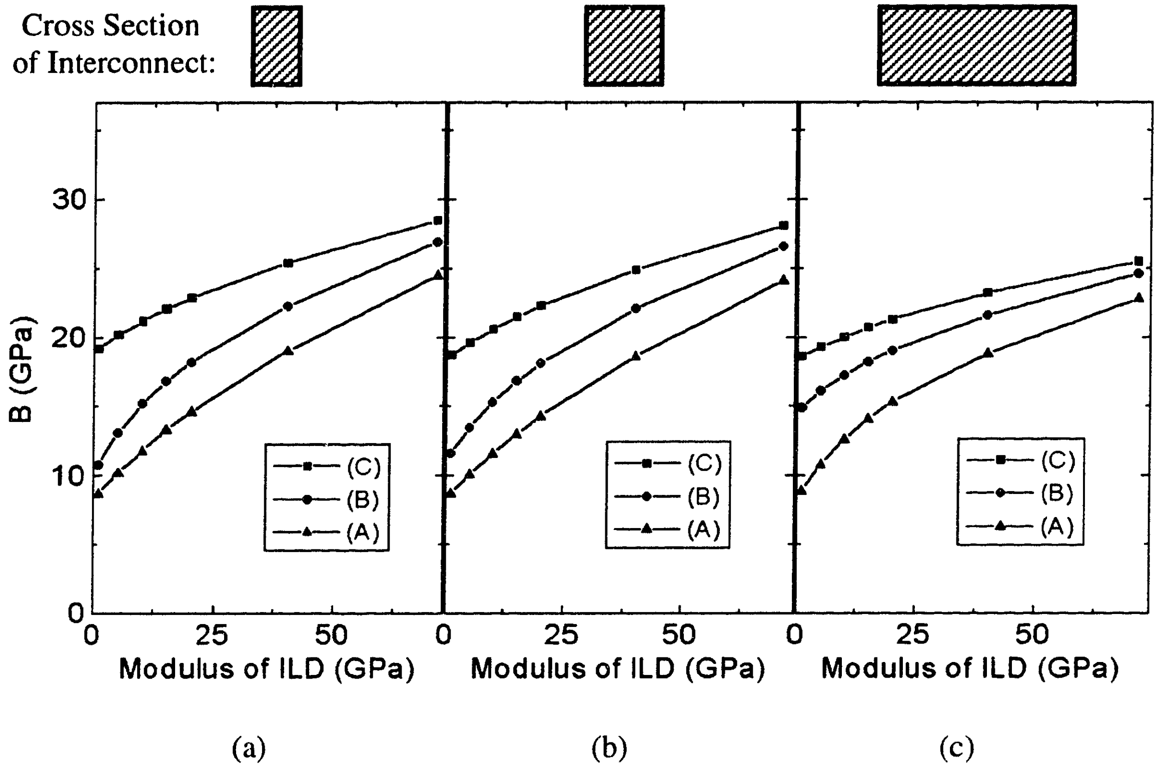
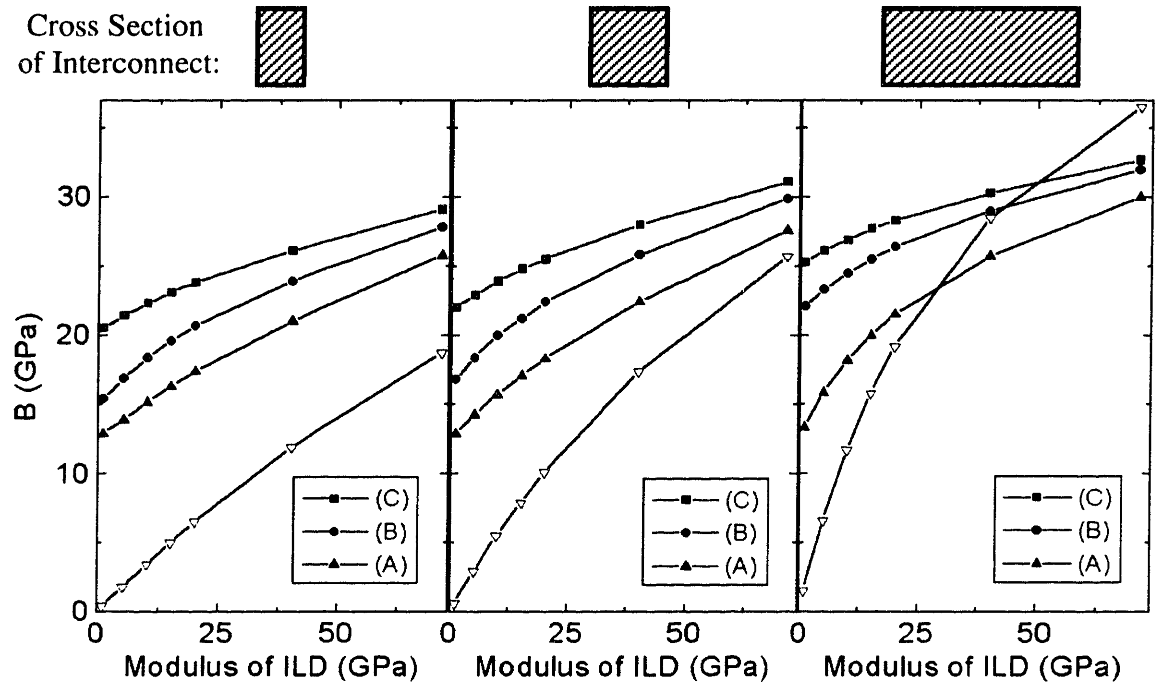


Figure B.2: The effective elastic modulus, B, as a function of the elastic modulus of the ILD for $\varepsilon_1^T = \varepsilon_2^T = \varepsilon_3^T$ for line widths: (a) w = 0.24 μm, (b) w = 0.40 μm, and (c) w = 1.00 μm. (A), (B), and (C) correspond to the models sketched in figures B.1 (a), (b), and (c).

Divergences in the electromigration flux lead to an accumulation or depletion of atoms. With reasoning similar to that described by Korhonen et al. [Kor 93], we have modeled the deposition of atoms in a continuum model by assuming a set of stress-free homogeneous dilational strains of the interconnect, ε_1^T, ε_2^T, and ε_3^T, as detailed below. The hydrostatic stress for a certain set of free strains was calculated using FEM and is related to the dilation by [Noy 87]

$$\Delta \equiv \varepsilon_1^T + \varepsilon_2^T + \varepsilon_3^T = \frac{dC}{C}. \qquad \text{(B.3)}$$

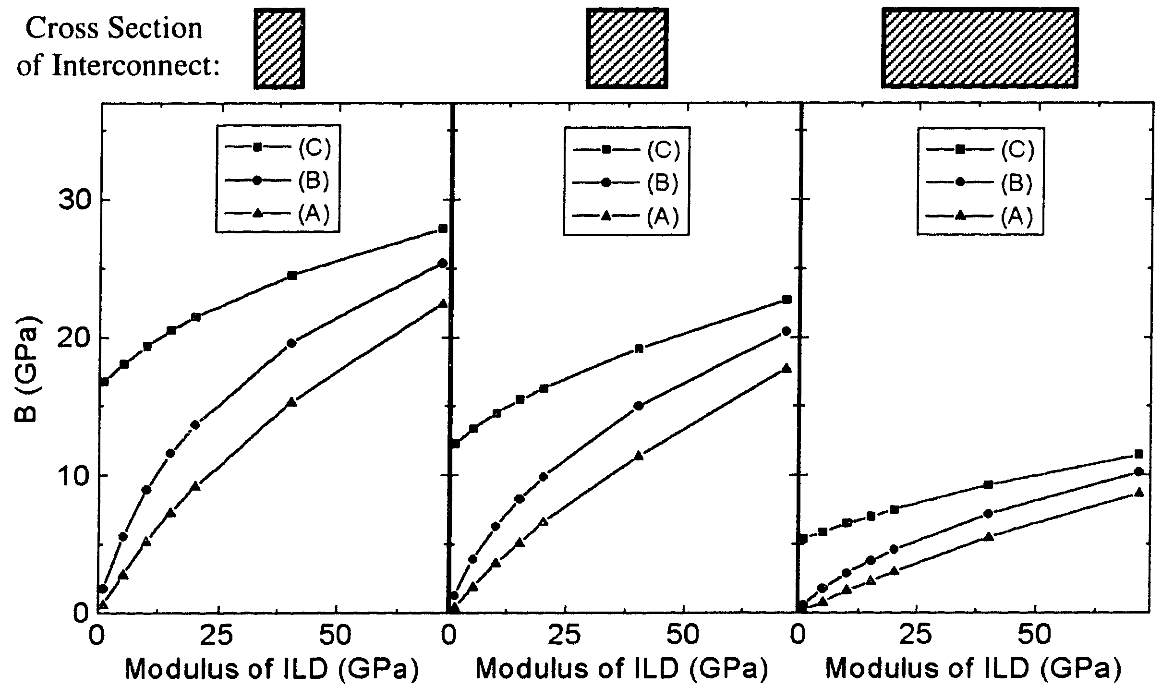


(a) (b) (c)

Figure B.3: The effective elastic modulus, B, as a function of the elastic modulus of the ILD for $\varepsilon_1^T = \varepsilon_2^T$, and $\varepsilon_3^T = 0$ for line widths: (a) w = 0.24 μm, (b) w = 0.40 μm, and (c) w = 1.00 μm. The results of the analytic model of Korhonen et al. [Kor 93] are shown by the curve marked with open triangles. (A), (B), and (C) correspond to the models sketched in figures B.1 (a), (b), and (c).

With this result the effective modulus was obtained by using equation (B.2).

Depending on the primary diffusion pathway, metallic atoms deposited in the interconnect can lead to different types of deformation. In our discussion, we focus on three different scenarios: (i) $\varepsilon_1^T = \varepsilon_2^T = \varepsilon_3^T$, which is the case for a polygranular interconnect with a three-dimensional grain structure, and in which the grain boundaries are the primary diffusion paths; (ii) $\varepsilon_1^T = \varepsilon_2^T$, and $\varepsilon_3^T = 0$, which is the case for a polygranular interconnect with columnar grain structure, and in which the grain boundaries are the primary diffusion pathways as well; and (iii) $\varepsilon_1^T = \varepsilon_2^T = 0$, which is the



(a) (b) (c)

Figure B.4: The effective elastic modulus, B, as a function of the elastic modulus of the ILD for $\varepsilon_1^T = \varepsilon_2^T = 0$ for line widths: (a) w = 0.24 μm, (b) w = 0.40 μm, and (c) w = 1.00 μm. (A), (B), and (C) correspond to the models sketched in figures B.1 (a), (b), and (c).

case for atoms diffusing along and deposited on the top surface of the interconnect (which, as will be discussed later, might apply to the case of Damascene Cu with Si_3N_4 as an interlayer diffusion barrier).

The dimensions and materials shown in figure B.1 (c) are representative for Cu-based metallization schemes that are currently used [Par 99]. As can be seen in table B.I, liner materials are stiffer than other back-end materials. This is true for TaN, which is used in the calculations here, as well as for Ta, Ti, and TiN, which are alternative liner materials. If liners are deposited by physical vapor deposition, the films tend to be thinner at the side walls of the trench than at the bottom. The metallization scheme shown in figure B.1 (b) represents the extreme case in which the liner is only present at

stress-free strains	E (GPa)	σ_{11}/σ_{22} (%)	B (GPa)
$\varepsilon_1^T = \varepsilon_2^T = \varepsilon_3^T$	1.0	1.8	8.7
	71.4	52.8	24.1
$\varepsilon_1^T = \varepsilon_2^T$ and $\varepsilon_3^T = 0$	1.0	1.7	12.9
	71.4	48.5	27.6
$\varepsilon_1^T = \varepsilon_2^T = 0$	1.0	26.7	0.4
	71.4	87.5	17.6

Table B.II: The ratio σ_{11}/σ_{22}, which describes the degree of deviation from the state of biaxial stress and the effective elastic modulus, B, for a line of square cross section for a metallization scheme depicted in figure 1 (a), for different elastic moduli, E, of the inter-level dielectric (ILD).

the bottom of the trench. This is similar to Al-based metallization schemes in which Ti, Al_3Ti and/or TiN over- and under-layers are used. Because decreased interconnect widths are critical for future increases in device density [Cha 99], the thickness of liners has to decrease with the cross-sectional area of the interconnects in order to keep the electrical resistance low. In the limit of no liner, a metallization scheme as shown in Figure B.1 (a) results. We have verified using simulations that for the sets of stress-free strains we have considered, the presence of the Si_3N_4 layers does not have a significant effect on the effective elastic modulus, B.

B.3 Simulation Results

The effective elastic modulus, B, as a function of the Young's modulus of the ILD, E, for the different metallization schemes sketched in figure B.1, and for three different line widths, is shown in figures B.2 (a) to (c) for the case of $\varepsilon_1^T = \varepsilon_2^T = \varepsilon_3^T$. The Young's modulus ranges from 71.4 GPa, which is the elastic modulus of amorphous SiO_2, down to 1 GPa. Figures B.3 (a) to (c) show B for the case of $\varepsilon_1^T = \varepsilon_2^T$, and $\varepsilon_3^T = 0$, and, finally, figures B.4 (a) to (c) show B for the case of $\varepsilon_1^T = \varepsilon_2^T = 0$. For comparison, the results of the analytic analysis described by Korhonen et al. [Kor 93] is overlaid on the curves in figure B.3, assuming an elliptical copper interconnect inside an infinite, isotropic ILD matrix. In table B.II, the values of σ_{11}/σ_{22} and B obtained using FEM analyses are shown for a line with a square cross section and a metallization scheme as depicted in figure B.1 (a).

B.4 Discussion of Simulation Results

We will first discuss the case of a metallization scheme without liner or barrier materials, and without Si_3N_4 layers, as sketched in figure B.1 (a). σ_{11}/σ_{22} describes the degree of deviation from a state of biaxial stress. If $\sigma_{11}/\sigma_{22} = 100\%$, the stress state is fully biaxial, which was assumed for the analytic model described by Korhonen et al. [Kor 93]. As shown in table B.II, our calculations show that, in general, the lines do not exhibit a biaxial stress state. For $\varepsilon_1^T = \varepsilon_2^T$ and $\varepsilon_3^T = 0$, $\sigma_{11}/\sigma_{22} \approx 53\%$ for an ILD with a Young's modulus similar to that of SiO_2, and $\sigma_{11}/\sigma_{22} \approx 2\%$ if the Young's modulus of the ILD is 1 GPa. The softer the dielectric, the larger is the deviation from the biaxial stress state. This is the case because the interconnect can expand more easily into the ILD in direction 1, whereas the Si substrate prevents the interconnect from expanding in direction 2 along the interconnect. It can therefore be expected, and we verified this using FEM calculations, that the analytic model described by Korhonen et al. [Kor 93] does not predict valid effective moduli, B, for soft ILD's. For the cases of $\varepsilon_3^T \neq 0$, the effective modulus decreases with decreasing aspect ratio h/w, because the material can expand more freely into direction 3 perpendicular to the wafer. Only if the material expands in direction 3 alone, which means in the case when $\varepsilon_1^T = \varepsilon_2^T = 0$ and $\varepsilon_3^T \neq 0$, does B become small for a small ILD Young's modulus, as can be seen in figure B.4. Because $\varepsilon_2^T = 0$, the Si substrate does not prevent the interconnect from expanding in direction 2. This analysis shows that for the metallization scheme sketched in figure B.1 (a) the effective elastic modulus strongly depends on the set of free dilational strains.

For the metallization scheme shown in figure B.1 (b), the effective modulus shows a similar dependence on the Young's modulus of the ILD as for the case shown in figure B.1 (a). B is slightly larger than in the case shown in figure B.1 (a), because the Si_3N_4 as well as the liner material on the bottom of the trench somewhat restrain the interconnect from expanding into the ILD.

Finally, the metallization scheme shown in figure B.1 (c) shows a different behavior for soft ILD's than the other two metallization schemes, because the interconnect is significantly constrained in all three directions. The liner material at the side wall of the trench prevents the interconnect from expanding into direction 3 or direction 1. As in other cases, the extension of the interconnect in direction 2 is prevented by the Si wafer, so that for a 35 nm-thick liner, B changes by less than a factor of 2 when the Young's modulus of the ILD decreases by nearly two orders of magnitudes. Only for very thin liners is B much smaller for soft ILD's than for SiO_2.

B.5 Discussion of the Effects of the Effective Elastic Modulus on Electromigration

The tensile stress that develops at the electron-source via eventually becomes large enough that the strain energy reduction associated with void formation is larger than the surface energy cost associated with voiding, and a void nucleates. In the Korhonen model, if it is assumed that the diffusivity is independent of stress, the stress evolution at the electron-source end of a semi-infinite line prior to voiding is given by [Kor 93]

$$\sigma(t) = \sigma_0 + \sqrt{\frac{4t}{\pi}\frac{\Omega D}{kT}\frac{\rho q^* j}{\Omega}} \cdot \sqrt{B}\,, \tag{B.4}$$

so that the stress increase is proportional to $\sqrt{B}$. ρ is the electrical resistivity of the high-conductivity metal, and j is the current density. The electromigration-induced atomic flux in the absence of back stress effects is independent of the effective elastic modulus and, once a void has nucleated, the void length is given by [Par 99]

$$l_{void} = \frac{q^* \rho j}{kT} Dt\,. \tag{B.5}$$

If it is easy for voids to nucleate, interconnect failure is controlled by void growth. In the case of little or no back-stress effects, void growth is independent of the effective elastic modulus, so that the mechanical properties of the ILD do not affect interconnect reliability in this case. A likely scenario for this behavior is a copper-damascene interconnect with a weakly adhering Si_3N_4 layer on top of the Cu, making it easy for Cu to delaminate from the Si_3N_4. On the other hand, if liner materials and over-layers adhere well to the metallic interconnect, voids are difficult to nucleate and interconnect failure is determined by the rate at which stresses build up, so that the time to failure is inversely proportional to the effective elastic modulus (assuming $\sigma_0 = 0$). In this case, once a void has nucleated the stress relief around the void will quickly lead to rapid void growth and to a resistance increase large enough for the line to fail [Par 99].

It has been shown that short interconnects and/or interconnects carrying a low current density can be immortal, either because the stresses in the interconnect do not become large enough for a void to nucleate [Ble 76], or because even when voids nucleate, they will eventually stop growing due to back-stress effects. In both cases, the electromigration driving force is balanced by a gradient in the chemical potential induced by a stress gradient which is constant along the length of the line. The stress gradient is given by

$$\frac{q^* \varrho j}{\Omega}, \tag{B.6}$$

which is independent of the effective elastic modulus. However, the total material transported along the line depends on B, and the resistance increase, ΔR_{sat}, at steady state is given by [Suo 98]

$$\Delta R_{sat} = R \frac{\rho_l A}{A_l} \cdot \frac{q^*}{2\Omega B} \cdot jL, \tag{B.7}$$

which is inversely proportional to B. R is the initial resistance of the interconnect, A is the cross-sectional area of the high-conductivity interconnect, A_l is the cross-sectional area of the liner, and L is the interconnect length. A smaller B leads to a larger steady-state resistance increase than a larger B.

B.6 Summary and Conclusion

The analytic model developed by Korhonen et al. [Kor 93] to describe the elastic response when lattice sites are added or removed from Al embedded in SiO_2 was applied to Cu embedded in a low-E ILD. The predicted response, as embodied in the effective modulus B, was found to be of the same order as the changes in the modulus of the dielectric E, so that the predicted changes in interconnect reliability would be very significant when low-E ILD's are used. However, using more accurate finite element analyses, we have found that for Cu-based metallization schemes with liners and with dimensions as shown in figure B.1 (c), a decrease in the elastic modulus of the ILD has little effect on the effective modulus, B, which suggests that the rates of electromigration-induced stress change and of void growth are nearly independent of the mechanical properties of the ILD. Because the copper is not in direct contact with the ILD, but is surrounded by a liner material, the critical tensile stress above which void nucleation occurs is also not affected by a change in the dielectric. This indicates that the reliability of interconnects should not significantly change if the ILD material is changed from SiO_2 to a softer material.

Our calculations also show that B only weakly depends on the line cross section. However, in future process generations, thinner and therefore weaker, liners will be used, so that the effective modulus will be lower. A lower effective modulus slows the buildup of stress, which, in turn, reduces the rate at which electromigration-induced damage initiates. On the other hand, the modulus does not affect the electromigration-induced atomic flux, and therefore the growth of voids, once nucleated, is unaffected. In addition, the failure mode of interconnects with thin liners may be different from the case of interconnects with thick liners, because if the liner is very thin, electromigration-induced stresses might lead to failure of the liner material itself, so that copper can diffuse into the ILD and Si to cause short-circuit failures or device failures. The extent to which the use of softer low-k ILD's will affect Cu-based interconnect reliability will depend most strongly on their effects on the mechanical reliability of diffusion-barrier liners.

In summary, our results suggest that as long as the liner materials for Cu interconnects remain intact, the mechanical properties of alternative (e.g., low-k) dielectrics will have less of an effect on interconnect reliability than the standard model by Korhonen et al. [Kor 93] would indicate. While the Korhonen model overestimates the effects of alternative dielectrics, the choice of dielectric material, the choice of liner material, and interconnect dimensions still have a significant effect on the mechanical reliability of interconnects, in ways that can be predicted using finite-element mechanical modeling in conjunction with electromigration models and simulations.

Appendix C In-Situ TEM Studies of the Kinetics of Abnormal Grain Growth in Electroplated Copper Films

C.1 Introduction

We are in the midst of a major technology transition from aluminum alloys to copper as the metallization material for high performance integrated circuits. This transition is driven primarily by the associated reduction of the electrical resistance of the metallization which allows significantly higher clock speeds as total interconnect lengths continue to increase [Ede 97]. Copper interconnects are fabricated by filling trenches in a dielectric and subsequently using chemical mechanical polishing (CMP) to remove Cu between trenches. Because electroplating shows very good trench filling characteristics and is cost-effective, the copper is deposited by electroplating. However, electroplated copper films transform through recrystallization to larger-grained structures at room temperature over a period of hours or days after deposition. This leads to changes in the electrical, mechanical, and microstructural characteristics of the films. Recrystallization has been observed for continuous films [Tom 85] and copper deposited into trench structures [Lin 98, Lin 99]. The mechanism for this transformation is not yet well understood.

We have performed in-situ transmission electron microscopy (TEM) studies of the transformation of electroplated copper films, starting minutes after the plating process by electroplating directly onto electron-transparent membranes. TEM allows observation of the microstructure of the metal through the full thickness of the film. Because TEM is non-destructive, we are able to continuously follow the time evolution of single grains. This can not be done with other techniques such as focused ion beam microscopy [Lin 98]. We will show that our observations of the time evolution of recrystallizing films allow quantitative kinetic characterizations that lead to insight into the mechanism of the recrystallization process.

C.2 Experimental Setup

The electron-transparent membranes were 1000Å-thick stoichiometric Si_3N_4 films made by depositing silicon nitride on the top and bottom of [100]-oriented silicon wafers at 750°C by low pressure chemical vapor deposition. The films are under tensile stress at room temperature which keeps the membranes taut. Square-shaped windows were etched in the silicon nitride on the backside of the wafer, and the exposed Si was etched through use of KOH which leaves Si_3N_4 membranes supported by a Si frame. We deposited a 350Å-thick copper seed layer on top of the silicon nitride membranes using electron beam deposition.

Copper films were electrodeposited on top of the silicon nitride/copper membranes using an Enthone-OMI SEL-REX CUBATH SC acid copper electroplating process [Ent]. The plating bath consists of sulfuric acid, hydrochloric acid, water, copper sulfate, and proprietary additives. It resides in a poly-propylene tank, and it is agitated by stirring. The plating was done at room temperature. The copper anode has a phosphorous content of 0.06% according to the bath manufacturer's recommendation, and the anode to cathode area ratio was approximately 2:1. Plating was carried out at a current density of 15 ASF using direct current. The cathode sample holder was also

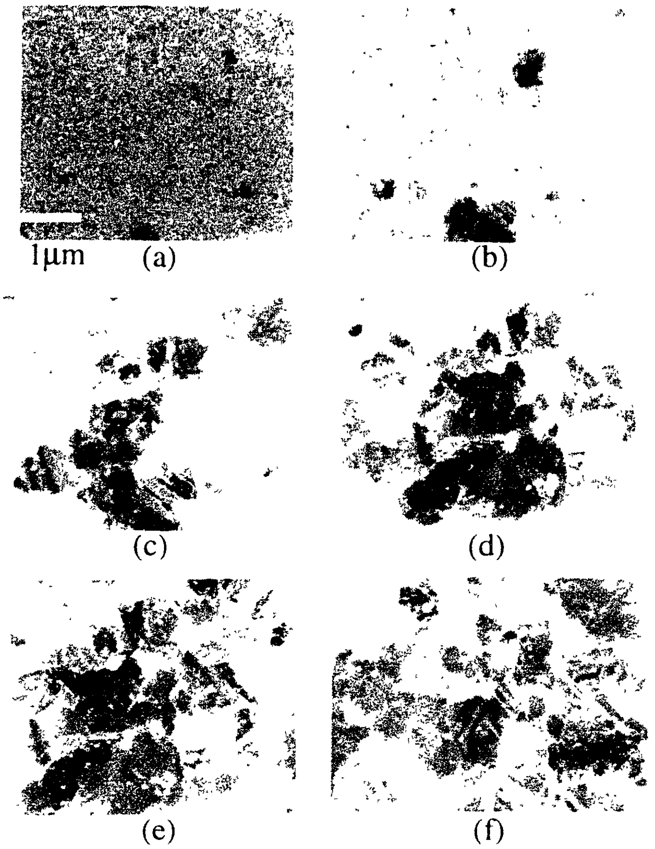


Figure C.1: Series of TEM micrographs for a 0.3μm-thick electroplated Cu film transforming at room temperature. (a) t =11 minutes, (b) t = 1 hour, (c) t = 2.5 hours, (d) t = 8 hours, (e) t = 23.5 hours, (f) t = 80 hours. After t = 80 hours, no further transformation took place.

made of copper, and it was much larger than the sample we deposited on to ensure a constant current density. The deposition rate was 0.4μm per minute.

0.3μm-thick Cu films were deposited on top of the seed layers and the recrystallization kinetics was studied through TEM observations starting immediately after deposition at room temperature. In other experiments, a hot stage was used and the samples were heated to elevated temperature (55°C) and observed until the grain structure stopped evolving. Subsequently, the films were heated to 200°C and finally to 300°C in order to observe the effects of annealing steps after recrystallization. The electron beam was turned off between observations, in order to minimize the effects of the electron beam on the transformation. However, areas not exposed to the beam were compared to the areas that had been studied, and we found no difference in the transformation state. Comparisons with films deposited onto bulk samples instead of membranes showed a similar grain size after the recrystallization process.

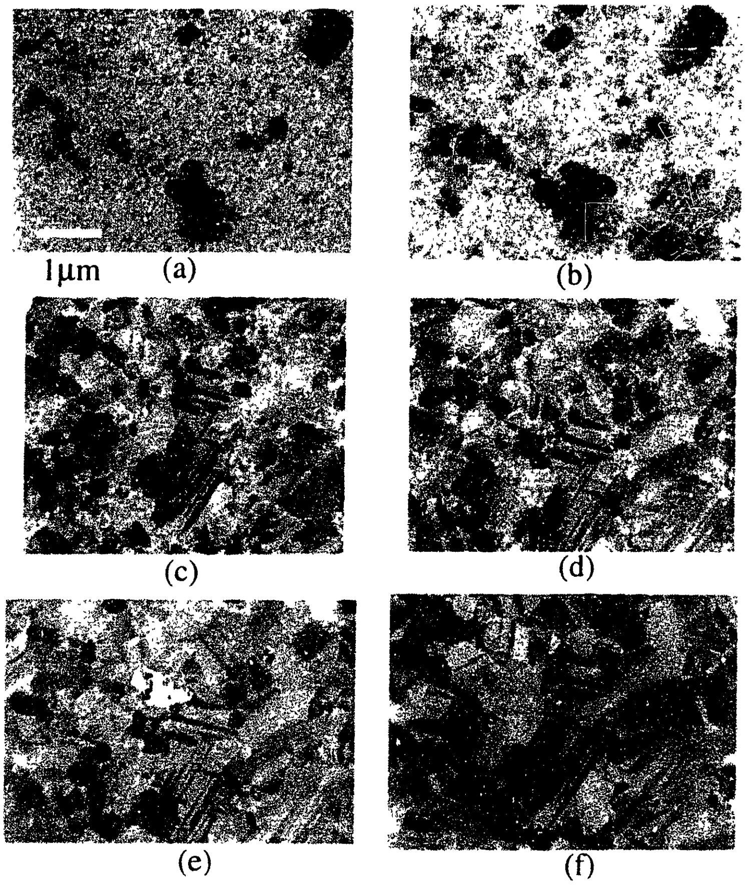


Figure C.2: Series of TEM micrographs for a 0.3μm-thick electroplated Cu film transforming at 55°C. The film was heated to 55°C 10 minutes after plating. (a) t = 5 minutes, (b) t = 12 minutes, (c) t = 49 minutes, (d) t = 1 hour after the sample was heated. Afterwards, no further transformation was observed. Subsequently, the film was annealed at 200°C for 10 minutes, shown in (e), and then annealed at 300°C for 10 minutes, shown in (f).

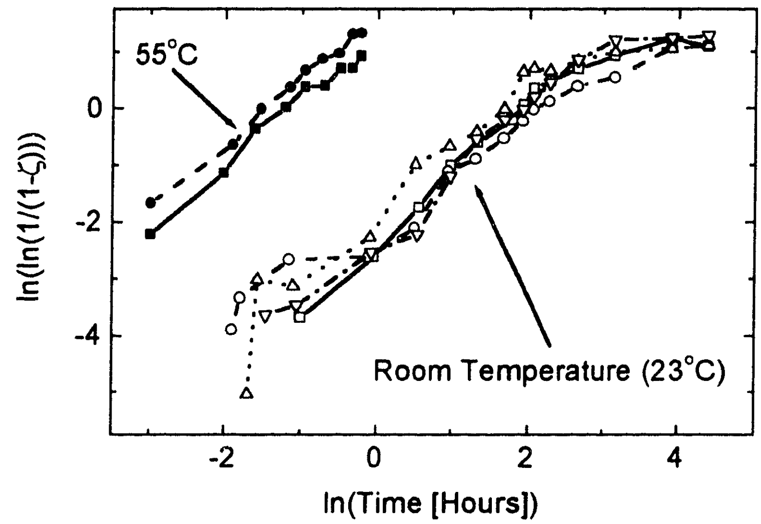


Figure C.3: The time dependence of the fraction of a 0.3μm-thick electroplated Cu film transformed, ζ, at room temperature and at 55°C.

C.3 Results

The plated films are reflective and fine-grained, as can be seen in figure C.1 (a), and the thickness is uniform within 5% over the area of observation. After deposition, the films undergo a transformation at room temperature by recrystallization over a period of hours to days, as shown in figure C.1 (a) to (f). Only a few grains grow and impinge to transform the entire film into a larger-grained structure. The structure of the untransformed part of the film is apparently unchanged. The grains that grow, quickly reach a diameter that is much larger than the film thickness, so that the majority of the time the system is undergoing a two-dimensional transformation.

At elevated temperatures, the transformation process is greatly accelerated. Figure C.2 (a) to (d) shows an electroplated film annealed at 55°C. The film nearly completely transforms within an hour. Upon further annealing at higher temperatures, the remaining untransformed area recrystallizes, as shown in figure C.2 (e) and (f), and

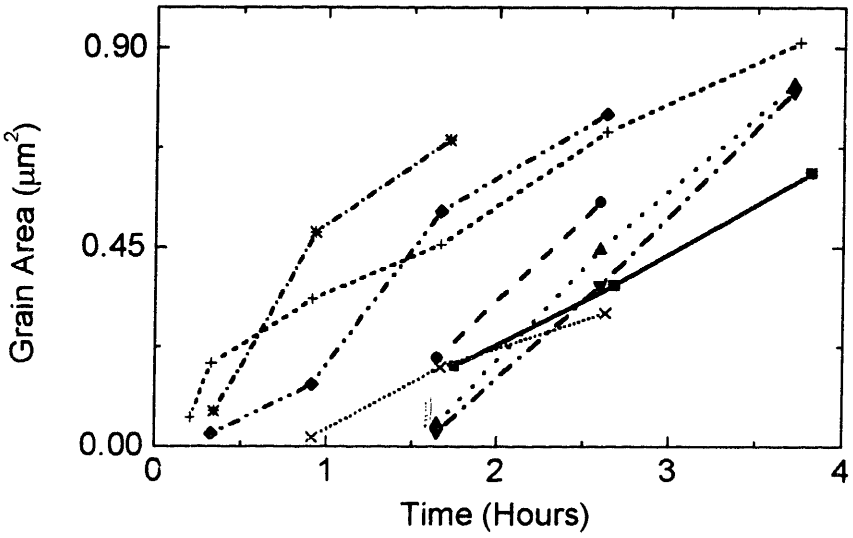


Figure C.4: The time evolution of the area of isolated grains, prior to impingement on other growing grains, at room temperature.

the images become clearer, probably due to the removal of line and other defects upon annealing.

The time-dependence of the volume fraction of the film transformed, ζ, is shown in figure C.3 for the case of films transforming at room temperature and for films transforming at 55°C. Each curve represents a separate experiment, and each data point is based on the evaluation of two TEM micrographs. At the beginning of the transformation, the grains which grow, grow independently until they impinge. The time dependence of the areas of these growing grains prior to impingement to other grains is shown in figure C.4 for growth at room temperature.

C.4 Discussion

The kinetics of phase transformations in the JMAK theory can generally be described by [Chr 65]

$$\zeta(t) = 1 - \exp\left(-K \cdot t^n\right), \qquad \text{(C.1)}$$

where $\zeta(t)$ is the volume fraction transformed at time t, K is the rate constant, and n is the Avrami exponent. The Avrami equation provides a reasonable approximation for the growth behavior especially at the early stages of transformation. From equation (C.1) it follows that

$$\ln\left(\ln\left(\frac{1}{1-\zeta(t)}\right)\right) = \ln K + n \cdot \ln t\,, \qquad \text{(C.2)}$$

so that the slope of $\ln(\ln(1/(1-\zeta)))$ as a function of ln(t) gives the exponent n, and the abscissa gives ln(K). In figure C.3 it can be seen that $\ln(\ln(1/(1-\zeta)))$ is linear for small and medium times, but increases slower for longer times. A least-squares fit of the data with the Avrami equation (equation (C.2)) results in n=1.0. An Avrami exponent of 1 for growth in 2D suggests a diffusion-limited growth from a fixed number of sites [Chr 65]. The time-dependence of the area of individual abnormally growing grains, prior to impingement, A, shown in figure C.4, indicates a linear relationship of A and time, t. Again, this is in accordance with a diffusion-limited growth. The activation energy of the rate constant, K, assuming an Arrhenius-type temperature dependence, is $E_a \sim 0.9$eV. This is in agreement with observations based on the measurement of the change in electrical resistivity in copper films [Jia 98] and focused-ion-beam studies [Cab 98].

Growth described by equation (C.1) with n=1 is consistent with the growth of a select sub-population of grains, which reject impurities as they grow, so that the impurities accumulate in (and must diffuse through) the untransformed matrix. Increasing the amount of additives in the electroplating bath is known to lead to a decreased grain size of the deposit [Har 99], presumably because the additives segregate to grain boundaries and inhibit boundary motion and therefore grain growth [Lüc 63]. The untransformed matrix consists of very small grains (d < 0.1μm), so it contains a dense grain boundary network, which provides both a set of fast diffusion pathways for impurities and a high driving force for grain growth. It has been observed that sulfur, carbon, and oxygen compounds desorb from the surface prior to and during the transformation process [Bro 99], and it was suggested that these compounds are the impurities pinning the grain boundaries. However, our results (n=1) suggest that these impurities are rejected by abnormally growing grains, and that their accumulation at the moving boundary impedes further growth. That the impurities are rejected, indicates that this rejection and the accompanying purification of the growing grains, contributes to the driving force for their growth.

C.5 Summary and Conclusion

In summary, our results suggest that impurities play several important roles in the room temperature "self annealing" of electroplated Cu films. First, three-dimensional normal growth of the grains in the untransformed matrix is impaired by solute drag due to the impurities. The resulting stabilized small grain size contributes to a high driving force for recrystallization or abnormal grain growth. Second, impurity rejection contributes to the energetics which drive the transformation to a large-grained structure. Third, accumulation of rejected impurities at the perimeters of the growing grains can slow the rate of transformation, so that the rate and mechanisms of impurity removal from the film can play an important role in controlling the kinetics of the recrystallization or abnormal grain growth process

.

Appendix D Modeling of Texture Evolution of Copper Interconnects Annealed in Trenches

D.1 Introduction

We are in the midst of a major technology transition from aluminum alloys to copper as the metallization material for high performance integrated circuits. This transition is driven primarily by the associated reduction of the electrical resistance of the metallization, which allows significantly higher clock speeds as total interconnect lengths continue to increase [Ede 97]. Aluminum-based interconnects are fabricated using a subtractive process involving deposition of continuous aluminum films and subsequent patterning using reactive ion etching. Copper interconnects are fabricated by filling trenches in the dielectric which are lined with refractory-metal-based diffusion barriers. The Cu is patterned through the use of chemical mechanical polishing (CMP) to remove the Cu between trenches. The Cu-filled trenches are covered with deposited diffusion barriers and inter-level dielectrics at elevated temperatures. During this final step, the interconnect is being thermally annealed and its grain structure evolves.

The grain structure and crystallographic texture of aluminum-based interconnects determine their resistance to electromigration-induced failure. A strong (111) texture results in a significant reliability improvement [Kno 93]. Similar effects appear to apply to copper metallization [Abe 98]. In this paper a model for annealing-induced texture evolution in copper damascene interconnects is presented, accounting for the effects of surface energy and mechanical anisotropies. The model provides predictions of energy minimizing textures which should be favored during grain structure evolution, as a function of the geometry and thermal history of the interconnects [Lin 98].

The texture in aluminum-based interconnects has been studied extensively, and it has been shown that the texture is a function of deposition conditions and the morphology of underlayers [Rod 92]. A more limited number of reports on as-deposited texture and texture evolution in copper in damascene trenches have been made [Lin 98, Gro 98, Lin 99], and most of these are based on X-ray diffraction Θ-2Θ scans [Hsu 98], which have been shown to be potentially misleading [Gro 98]. Pole figure measurements show evidence of a side-wall texture in the trenches in addition to the (111) texture observed in flat regions in the as-deposited as well as in the annealed states [Lin 99]. It has also been observed that the (111) texture decreases with decreasing ratios of line width to trench depth [Van 99].

Texture evolution in trench structures based on strain energy minimization [Vin 94], and, separately, based on surface- and interface-energy minimization [San 98] has been treated in earlier work. However, our work shows that in Cu the energy-minimizing texture in structures with dimensions of technological relevance can only be predicted if both driving forces are taken into account simultaneously. Carel et. al. [Car 96] showed that the energy-minimizing texture resulting from grain structure evolution in continuous silver thin films can be predicted by considering the simultaneous minimization of strain-energy density and surface- and interface-energies. In this work, we show how this concept can be extended to Cu damascene structures with a triaxial stress state.

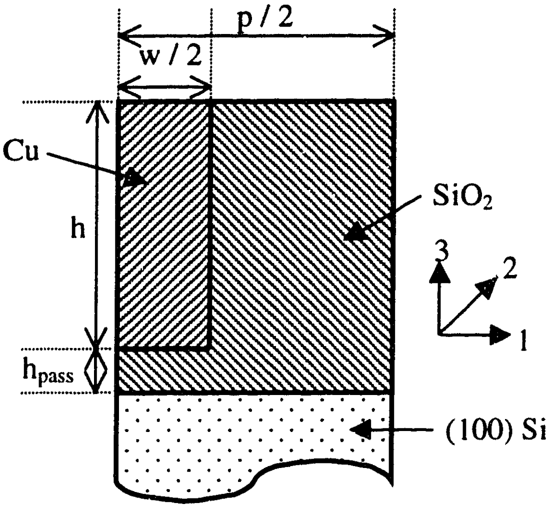


Figure D.1: Sketch of thegeometry and coordinate system for the model. The finite element mesh is free to expand in the vertical direction. Mirror symmetry is applied horizontally as well as long the line.

D.2 Model Description

In this analysis, it is assumed that the copper is stress-free after deposition (as is likely, for example, for electroplated Cu) and after chemical mechanical polishing (CMP). It is assumed that after CMP, the system is heated from room temperature (25°C) to 200°C. The geometry of the model is sketched in figure D.1. Due to symmetry, only half the line was modeled. The height or thickness of the copper is $h = 0.5$ μm, the thickness of the passivation underneath the Cu is $h_{pass} = 0.1$ μm. The substrate thickness is taken to be $h_{wafer} = 5$ μm. This and the disregard of the bending of the wafer approximates an infinite substrate [Sau 92]. The errors in the calculations due to the reduced substrate thickness and the neglected wafer curvature are less than 5%. The pitch of the structure was varied between 2.0 and 4.0. We have ignored the mechanical effects of the liner material, which should be negligible because its thickness is only a small fraction of the Cu thickness, but we do incorporate the effects of the liners on the interface energies. We assume perfect adhesion (traction) of Cu to the liner. The coordinate axis 1 in figure D.1 is perpendicular to the length of the trench, the coordinate axis 2 lies along the length of the trench, and the coordinate axis 3 is perpendicular to the wafer surface.

During annealing, the grain boundary mobility is increased so that the system can decrease its total energy through grain growth. The driving forces for grain growth are grain boundary energy, strain energy, and surface- and interface-energy. Texture evolution is governed by the *simultaneous* minimization of all these energy contributions. Due to anisotropic driving forces for grain growth, grains with energy-minimizing textures grow at the expense of grains with different textures. Surface-, interface-, and strain-energy minimization generally do not favor the same crystallographic texture, so that the energy that dominates in defining the texture that is favored during grain structure evolution depends on the relative magnitudes of the different driving energies. This, in turn, depends on geometry and thermal history of the interconnect.

Crystallographic Direction	$\gamma_{hkl}/\gamma_{100}$	γ_{hkl} (J/m^2)
(100)	1.000	2.610
(110)	1.048	2.734
(111)	0.971	2.534

Table D.I: Anisotropy of the surface energy of copper at 200°C.

Differences in the coefficients of thermal expansion (CTE) of copper, the passivating dielectric, and the wafer material lead to differences in thermal strains, $\varepsilon_{ij} = \delta_{ij}\alpha\Delta T$, for different materials selections and different ΔT's [Nye 95]. $[\varepsilon_{ij}]$ is the strain tensor, δ_{ij} is the Kronecker delta, α is the CTE, and ΔT is the temperature change from the zero stress condition. In the absence of plastic deformation, strains lead to stresses $[\sigma_{ij}]$ that are related to each other by $\sigma_{ij} = c_{ijkl}\varepsilon_{kl}$, where $[c_{ijkl}]$ is the stiffness tensor. The change in the strain energy density dW is given by $dW = \sigma_{ij}d\varepsilon_{ij} = c_{ijkl}\varepsilon_{kl}d\varepsilon_{ij}$ [Die 86]. Grains with different textures have different elastic constants in the plane of the wafer surface (in-plane) and out-of-plane, so that the strain energy will differ from grain to grain. Grains with orientations leading to low strain energy densities will grow at the expense of grains with other orientations. In FCC metals such as copper, the Young's modulus is maximum in the <111> direction and minimum in the <100> direction. Therefore, in continuous films a low strain energy density is expected for a <100> texture, whereas a high strain energy density is expected for a <111> texture. In our calculations for filled trenches, we used temperature-dependent elastic constants for copper, assumed that the passivation material was silicon dioxide, and assumed the wafer material was (100)-oriented silicon [Sim 71].

The surface- and interface-energy density of a line (energy/volume) is given by

$$W_{s,i} = \frac{2h\gamma_i^{sw} + w\gamma_i^{bottom} + w\gamma_{si}}{p(h + h_{pass} + h_{wafer})}, \qquad (D.1)$$

and increases with increasing aspect ratio h/w for constant pitch p/w. γ_i^{sw} is the interface energy at the sidewalls, γ_i^{bottom} is the interface energy at the bottom of the trench, and γ_s is the free surface energy. Experimental data for interface and surface energies for copper is rare and contradictory. The surface energy depends on temperature [McL 69] and deposition method [Li 97]. We extrapolated the values reported by McLean for the free surfaces of copper to 200°C, see Table D.I, and assumed that the interface energy was the same as the free surface energy. The close-packed (111) planes have the lowest surface energy. The lowest index plane perpendicular to (111) is (110). For a (111) texture, a (110) orientation perpendicular to the length of the line is assumed, whereas for a grain with a (111) direction perpendicular to the length of the line a (110) texture is assumed.

The course of texture evolution is determined by the competing driving forces due to strain energy and surface- and interface-energy anisotropy. We calculated the total energy density for single Cu crystals with different crystallographic orientations in trenches at elevated temperature assuming a zero CTE of the wafer, and using temperature-dependent differences in CTE's for the other materials [Tou 85]. Modeled in this way, the portions of the thermal strain common to both the film and the substrate which do not lead to film stresses [Vin 97] and therefore do not contribute to the strain energy are not included. A comparison of energy densities for different textures gives the energy-minimizing texture, i.e. the texture toward which the system will evolve, as a function of geometry and thermal history.

Finite element calculations were performed using the commercial code Abaqus [Aba 98] using three-dimensional continuum-stress/displacement first-order-interpolation elements. The meshes were generated using our own mesh generation program with a refined mesh near edges and corners. The meshes were refined until a convergence of the stress distributions was achieved.

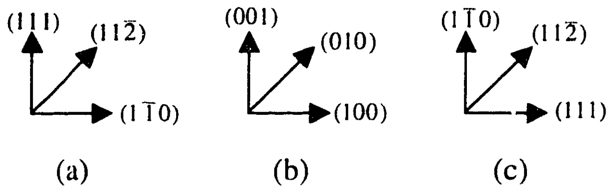


Figure D.2: Crystallographic orientations considered in the calculations. (a) (111) texture is assumed to minimize the top and bottom interface- and surface energies of the copper, (b) (100) texture minimizes the strain energy densities, and (c) (110) texture is favored when trenches are narrow so that interface-energies dominate.

D.3 Results

Calculations were done for the crystallographic orientations sketched in figure D.2, since, from the earlier discussion, we expect one of these crystallographic orientations to minimize the energy of the system. We ignore plastic deformation, but our finite-element-code can easily be modified to incorporate a geometry and crystallographic-orientation-dependent yield model.

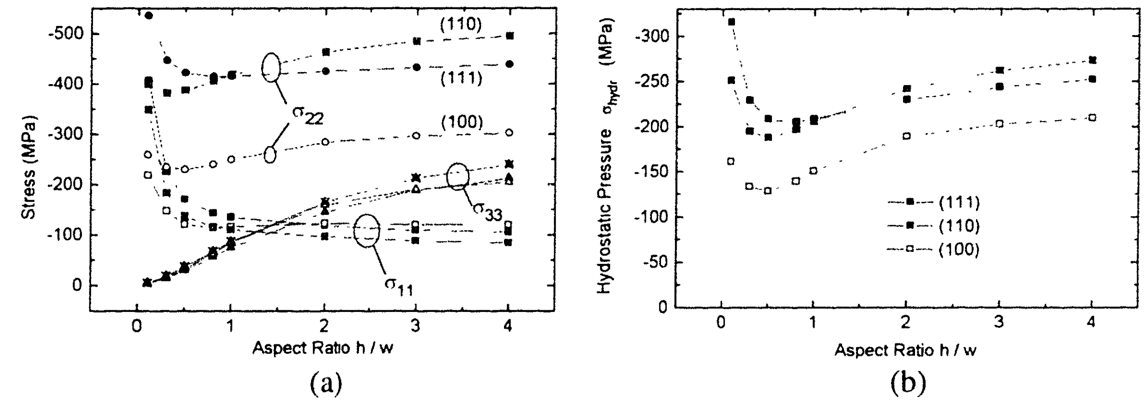


Figure D.3: (a) Principal stresses averaged over the cross section of the trench upon heating from 25°C (taken to be zero stress) to 200°C. (b) Corresponding hydrostatic stresses. The texture is indicated by the crystallographic orientations in brackets.

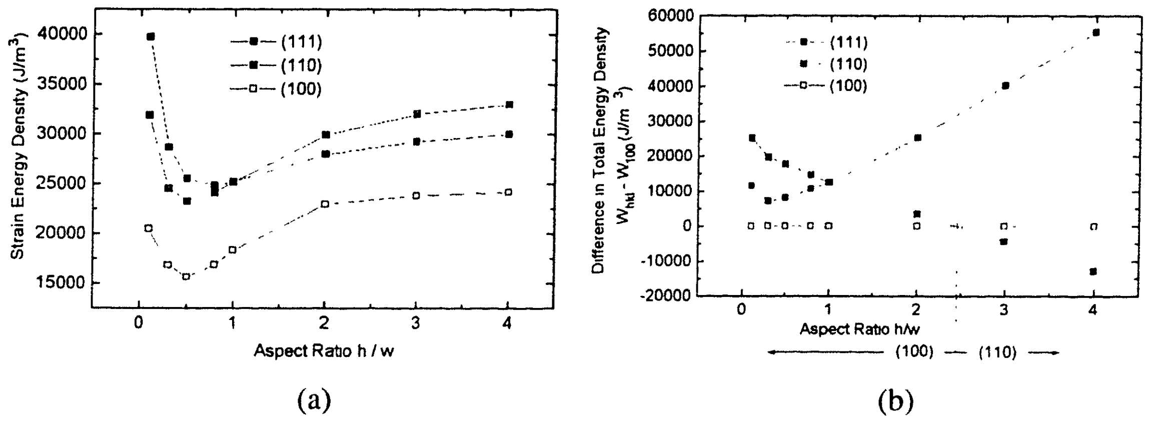


Figure D.4: (a) Strain energy density as a function of aspect ratio, h/w. (b) Difference in the total energy density for the (111), (110), and (100) textures, as a function of aspect ratio h/w. The texture is indicated by the crystallographic orientations in brackets.

Figure D.3 (a) shows the principal stresses σ_{11}, σ_{22}, and σ_{33} in the trench as a function of trench aspect ratio, h/w, averaged over the cross section of the trench for constant pitch, p/w = 3.0. Negative stresses are compressive. Figure D.3 (b) shows the corresponding hydrostatic stress $\sigma_{hydr} = (\sigma_{11} + \sigma_{22} + \sigma_{33})/3$. The strain energy density of the whole system is shown in figure D.4 (a), and the total energy density is sketched in figure D.4 (b).

D.4 Discussion

The CTE of copper exceeds the CTE of the passivation and the wafer, so compressive stresses develop upon heating, as show in figure D.3 (a) and (b). In the thin film limit, i.e., for small aspect ratios, h/w, the normal stress σ_{33} becomes very small for all textures, and σ_{11} approaches σ_{33} for the (100) texture. as expected. Due to the lack of a top passivation layer, the absolute value of the hydrostatic pressure exhibits a *minimum* at an aspect ratio of about 0.5, (see figure D.3 (b)). In the case of a fully encapsulated line, a maximum in the absolute value of the hydrostatic pressure is observed at an aspect ratio of about 1.0.

Because Young's modulus is minimized in the (100) direction for Cu, the strain energy density (energy/volume) is much smaller for lines with a (100) texture than for lines with a (110) or (111) texture (see figure D.4 (a)). Therefore, considering strain energy alone, a (100) texture is energetically favored, regardless of geometry.

It is assumed that (111) planes have the lowest surface and interface energy, so the surface and interface energy of the system is at minimum with a (111) texture for wide lines and with a (111) texture at the sidewalls for narrow lines. The point of transition depends on the specific anisotropy of the interface and surface energy, and may vary with the chemical nature and morphology of the liner material (the surface energy tends to be less anisotropic for amorphous, or weakly-textured fine-grained liner materials), but the surface-energy-minimizing texture for very large and very small aspect ratios is independent of the degree of anisotropy.

Figure D.4 (b) shows the total energy density of the system, and it can be seen that the energy-minimizing texture cannot be predicted by minimizing strain- or surface- and interface-energy alone. For wide lines, i.e. lines with small aspect ratios, a (100) texture is energy-minimizing, whereas for narrow lines, i.e. high-aspect-ratio lines, a (110) texture minimizes the total energy of the system. The crossover moves to larger aspect ratios for decreasing pitch p/w. For decreasing trench thickness h, surface- and interface-energy effects become more pronounced. Notice that as the line width decreases toward zero the contribution of the strain energy to the total energy vanishes (and the surface and interface energy approaches a constant value), whereas for wide lines the strain energy dominates.

The energy-minimizing texture after annealing also depends on the annealing temperature T_{anneal} relative to the zero-stress temperature. At high annealing temperatures, the strain energy difference for different textures is larger, and texture evolution is more likely to be strain-energy dominated so that a (100) texture is favored, as long as the strain is accommodated elastically. At low annealing temperatures, surface energy effects dominate.

D.5 Summary and Conclusion

Our calculations show that only the simultaneous minimization of surface- and interface-energy and strain energy gives the correct energy-minimizing texture. Under the assumptions of our model, lines annealed to high temperatures at which high strains result, will evolve toward a (100) texture. Lines in which the texture evolves under relatively low strain conditions will develop surface energy minimizing textures. For wide lines this leads to (111) texture, and for narrow lines this leads to (110) texture. The energy-minimizing texture is therefore seen to be a function of the geometry and thermal history of a Damascene-Cu interconnect. Our calculations are consistent with experimental observations of the development of (111) side wall texture in narrow trenches [Van 99]. In addition, our calculations suggest than an as yet unobserved evolution toward a (100) texture for trenches of appropriate geometry and thermal history should occur.

Appendix E Incorporating the Effects of Particles into the Grain Growth Simulator

E.1 Modifications of the Grain Growth Simulator

To incorporate particles into the grain growth simulator, it was necessary to modify the simulation code at several locations, and it was necessary to add about 11,000 lines of C source code. Instead of including the source code in this chapter, the major changes will be discussed, and the reader is referred to the library version of the grain growth simulator for more details. The files added to the grain growth simulator are *ppt_add.c, ppt_modification.c, ppt_numerics.c, ppt_support.c*, and *ppt.h*.

E.1.1 Data Structures

The data structure for **particle** variables is

```
/* particle structure */
#define PPT_POINT_LINKS         8      /* max point links in particle */
#define PPT_TRIPLE_LINKS        10     /* max triple links in particle */
typedef struct
{
  int   point[PPT_POINT_LINKS];    /* segment points on ppt border   */
  int   triple[PPT_TRIPLE_LINKS]; /* triple points inside ppt */
  float x,y;                       /* center position ppt */
  float r;                         /* radius ppt */
  int   free;                      /* -1:not used, 0:used */
  int   grain;                     /* grain # ppt belongs to; only relevant for
                                      in-grain ppt;
                                      this field is NOT processed upon loading/
                                      writing data (but it is recalculated) */
} particle;
```

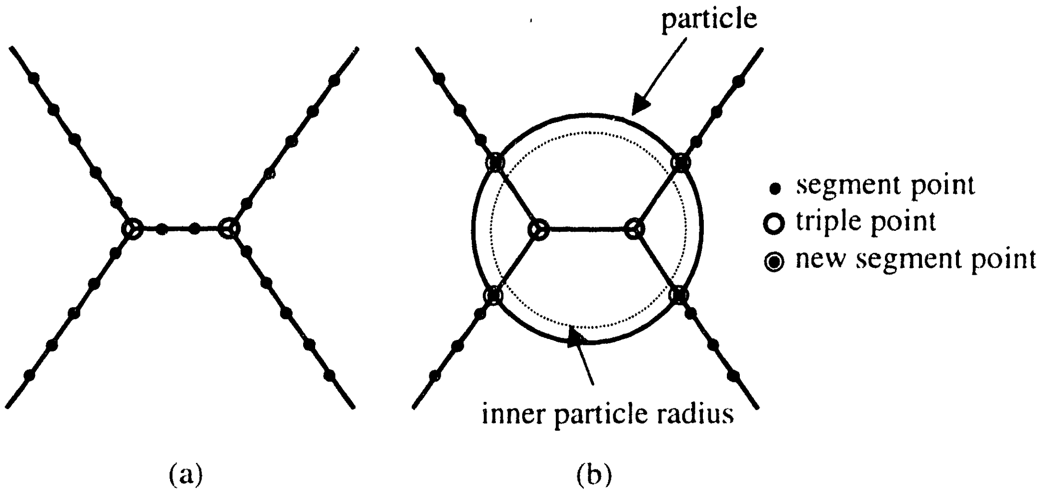


Figure E.1: Changes of the data structure (a) before and (b) after a particle is added.

Figure E.1 shows how the placement of a particle affects the data structure describing the grain structure. When a particle is added to a film, segment points are added at the intersection of grain boundaries and the particle circumference, and the segment points are listed in the **point** array of the **particle** structure. Segment points that lie inside of particles are removed. Triple points can lie inside of a particle and are listed in the **triple** array of the **particle** data structure. If necessary, triple points are relocated in a way that they lie within the inner particle radius, which is 90% of the full particle radius. Grains that are fully covered by a particle are removed. The **triple** and **segment** point data structure are both extended by the field

```
int ppt;                  /* precipitate # segment point lies on (or 0) */,
```

which determines the particle number the point is associated with. A flag was added to the **flag** data structure to signal if particles are present in the film,

```
int particle;             /* particles turned on (!=0) or off (=0) */
```

Throughout the simulation, it becomes necessary to determine if a point (x,y) lies near or within a particle, which, in turn, lies at a grain boundary. To expedite this process, the film is divided into quadrants. The quadrant number is calculated through

```
#define QUADRANT_SIZE            5.0    /* x and y size of a quadrant */
quadrant = ((int)(x/QUADRANT_SIZE)) + quadrantsx * ((int)(y/QUADRANT_SIZE ));
```

where **quadrantsx** equals the number of quadrants in the x direction. The array **num_ppts_in_quadrant[]** determines how many particles are present in a quadrant, and the array **ppts_in_quadrant[]** contains a list of the particles. A similar method is used for particles inside of grains using the arrays **ig_num_ppts_in_quadrant[]** and **ig_ppts_in_quadrant[]**.

E.1.2 Adding Particles to the Film

The development of the program code for adding particles to the grain structure was initiated by Brett Knowlton [Kno 97b], who wrote the C source file *ppt_add.c*. In this work, *ppt_add.c* was revised, and the modifications of the remaining grain growth code were finished. The function *addppt* reads the parameters from a command file, initializes the data structures, and adds particles by calling the function *add_ppt_loop*.

In *add_ppt_loop*, the program walks around each grain of the film, and for each segment and triple point encountered, it tests (considering the particle distribution function) if the point will be covered by a particle. After the particle radius is calculated, the particle is added to the film in case it does not overlap with other particles.

Similarily, particles are added inside of the grain. Finally, the function *delete_segment_inside_ppts* deletes segment points covered by particles.

E.1.3 Grain Growth in the Presence of Particles

The main simulation loop in the file *run.c* at case statement 7 calls the function *init_ppt_structure*, which initializes the quadrant data structure, and calls the function *ppt_kinetics* at every time step after *unwrinkle*, which governs the growth and shrinkage of particles during the simulation. The function *exit_ppt_structure* frees the allocated memory. The function *unwrinkle* is responsible for moving the segment points during a time step. In *unwrinkle*, the attributes of the segment points, such as **x**, **y**, and **stop**, are stored in local arrays. The arrays **pptl[]** (array of particle flags), **type[]** (array of type flags), and **ppt_segment_pos[]** (array of the point positions in **ppt[].point**) were added to store variables associated with particles. The following function initializes these arrays:

```
/*====================================================*/
/*
Called by unwrinkle().
Keeps track of point attributes.
*/
void   ppt_unwrinkle_point(int   seek,   int   point,   int   *stop_a,   int
stop[MAX_SEG_PNTS])
{
  int k;

  if( point<0 ) {
    pptl[seek] = triple[-point].ppt;
    type[seek] = 0;
    ppt_segment_pos[seek] = -1;
    if( triple[-point].ppt ) {
      stop[seek] = 1;
      if( seek == 0 ) *stop_a = 1;
    }
  }
  else {
    pptl[seek] = segment[point].ppt;
    type[seek] = segment[point].type;
    ppt_segment_pos[seek] = -1;
    if(segment[point].ppt) {
      stop[seek] = 1;
      for( k=0; k<PPT_POINT_LINKS; k++) {
       if ( ppt[pptl[seek]].point[k] == point)
         ppt_segment_pos[seek] = k;
      }
    }
  }
}
```

The function *spacing* called from the function *unwrinkle* has been replaced by the function *ppt_spacing*, which accounts for particles. The function *move_segment* has been modified to move a segment point only if it does not lie on top of a particle. It also checks if a segment point has encountered a particle by calling the function *ppt_cross_gb*:

```
/********************************************
** Check, if gb crossed a ppt.
** 3/24/97 -SPR-
**
** Xing types:
** -----------
** Type 1 - ppt lies >inside< the area the gb moves over
**            This is the most challanging case, because ppt does not X gb
** Type 2 - ppt lies on the new gb segment
** Type 3 - one new segment point lies on ppt
** Type 4 - two new segment points lie on ppt
********************************************/
void ppt_cross_gb(float *x, float *y, int *stop, float *x1mx0, float *y1my0,
float *len, int segmtpts, float *x_motion, float *y_motion)
{
  int     i, k, im1, j, quadr_x, temp_x, quadr_y, point_on_ppt, quadrant,
quadrantbase;
  float xmin=xsize, xmax=-xsize, ymin=ysize, ymax=-ysize, xmid, ymid;
  float new_x[MAX_SEG_PNTS],new_y[MAX_SEG_PNTS];
  int   ppt_candidates[MAX_PPT_CANDS], cand_list_pos;
  int   temp_ppt, pptnum, entry, segpos, redo;
  float s1, s2;
  float dx, dy, u, v, x1, y1, x2, y2;
  float max_motion=0.0;

  /*
  ** Initialize global variables for Xing
  */
  X_ppt_cnt = 0;                  /* number of crossed ppts */

  /*
  ** Get dimensions of rectangle the gb fits in
  */

  for( i=0; i<segmtpts+2; i++ ) {
    new_x[i] = x[i] + x_motion[i];
    new_y[i] = y[i] + y_motion[i];
    if( x[i]>xmax )     xmax = x[i];
    if( x[i]<xmin )     xmin = x[i];
    if( new_x[i]>xmax ) xmax = new_x[i];
    if( new_x[i]<xmin ) xmin = new_x[i];
    if( y[i]>ymax )     ymax = y[i];
    if( y[i]<ymin )     ymin = y[i];
    if( new_y[i]>ymax ) ymax = new_y[i];
    if( new_y[i]<ymin ) ymin = new_y[i];
    /* max_motion contains max of motion distance  */
    max_motion                 =                 (max_motion                 <
 (x_motion[i]*x_motion[i]+y_motion[i]*y_motion[i])) ?
      (x_motion[i]*x_motion[i]+y_motion[i]*y_motion[i]) : max_motion;
  }
  max_motion = sqrt(max_motion);

  /* (xmin, ymin) should lie inside 0..xsize,0..ysize */

  if( xmin<0 )      { xmin+=xsize; xmax+=xsize; }
  if( xmin>xsize ) { xmin-=xsize; xmax-=xsize; }
  if( ymin<0 )      { ymin+=ysize; ymax+=ysize; }
  if( ymin>ysize ) { ymin-=ysize; ymax-=ysize; }

  xmid = (xmin+xmax) / 2;
  ymid = (ymin+ymax) / 2;

  /* get line coordinates of xmid, ymid -> x1, y1 */
  if( strip ) {
    u  = xmid; v = ymid; swatch( &u,&v,0 );
    if( u<0 ) {u += xsize * n_regions;  v += ysize;}
    x1 =  cos_t*u + sin_t*v;
    y1 = -sin_t*u + cos_t*v;
  }

  /*
  ** Create a list of ppts that are candidates for Xing
  */

  /* get quadrant-position of left upper corner of rectangle */
  quadr_x = (int) ( xmin / IG_QUADRANT_SIZE );
  quadr_y = (int) ( ymin / IG_QUADRANT_SIZE );
  cand_list_pos = 0;
  /* loop over y */
  do {
    temp_x   = quadr_x;
    /* loop over x */
    do {
      /* modulus operator % makes sure we stay within quadrant field */
      quadrant       = (quadr_y % ig_quadrantsy)  *  ig_quadrantsx  +  (temp_x  %
ig_quadrantsx);
      quadrantbase = ig_max_ppts_per_quadrant * quadrant;
      /* add quadrantlist to list of ppt candidates */
      for( i=0;i<ig_num_ppts_in_quadrant[quadrant];i++ ) {
       temp_ppt = ig_ppts_in_quadrant[quadrantbase + i];    /* ppt # */
       /* look if this ppt is in list already */
       for( j=0;j<cand_list_pos;j++ )
         if( ppt_candidates[j]==temp_ppt ) j=cand_list_pos+MAGIC;
       if( j!=cand_list_pos+MAGICp1 ) {
         /* ppt not in list yet; check if ppt in right line segment */
         if( strip ) {
           /* wrap wrt xmid, ymid */
           x2 = ppt[temp_ppt].x;
           y2 = ppt[temp_ppt].y;
```

```
          rel_subwrap( &x2, &y2, xmid, ymid );
          /*SPRNEW
          dx = x2 - xmid;
          dy = y2 - ymid;
          if ( dx >  xsized2) x2 -= xsize;
          if ( dx < -xsized2) x2 += xsize;
          if ( dy >  ysized2) y2 -= ysize;
          if ( dy < -ysized2) y2 += ysize;*/
          /* Convert to line coordinates */
          u  = x2; v = y2; swatch( &u,&v,0 );
          if( u<0 ) {u += xsize * n_regions;  v += ysize;}
          x2 =  cos_t*u + sin_t*v;
          y2 = -sin_t*u + cos_t*v;
          /* wrap x w.r.t. x1 */
          dx = x2 - x1;
          if( dx > xsize * n_regions / 2.0 ) x2 -= xsize * n_regions;
          if( dx < -xsize * n_regions / 2.0 ) x2 += xsize * n_regions;
        }

        /* inside line distance is smaller than half the line length (one part)
*/
        if( (fabs(x2-x1) < (xsize / 2.0)) || (strip==0) ) {
          ppt_candidates[cand_list_pos] = temp_ppt;
          cand_list_pos++;
          if( cand_list_pos==MAX_PPT_CANDS ) {
            printf("ERROR - ppt candidate array overlow.\n");
            WAIT_CHAR;
          }
        }

        /* if diameter ppt < max_motion -> warning */
        if( (2*ppt[temp_ppt].r) < max_motion ) {
            printf("WARNING - possible gb X ppt interaction of type 1 w/out
noticing.\n");
          if( online ) scanf("%c",ch);
        }
       }
      } /* end loop over members of quadrant list */
      temp_x++;
    } while ( (temp_x*IG_QUADRANT_SIZE) < xmax );
    /* next line */
    quadr_y++;
  } while ( (quadr_y*IG_QUADRANT_SIZE) < ymax );

  /*
  ** Loop from first segment upto last triple (including)
  ** and check for ppt Xing
  */
  if( cand_list_pos ) {
    /* printf("cand_list_pos=%d\n",cand_list_pos); */

    for (i=1;i<(segmtpts+2);i++) {

      iml = i-1; /* so we do not have to re-calculate over and over again */

      /*
      ** if either last or this point moved, check for Xing
      */
      if ( mobile_flag || (stop[i] == 0)||(stop[i-1] == 0) ) {

       /* Type 1 Xing is not recognized yet!
       ** Get rectangle P(i-1) and P(i) (both old and new) lie in
       */
       /*
       ** Check for Xing type 2-4
       */
       for( j=0;j<cand_list_pos;j++ ) {

         /* calc intersection position with this ppt */
         pptnum = ppt_candidates[j];
         point_on_ppt=intersect_ppt_segment(ppt[pptnum].x, ppt[pptnum].y,
              ppt[pptnum].r   /*  *(2-PPT_TOLERANCE)  */,  x[im1],  y[im1],  x[i],
y[i], &s1, &s2);
         if( point_on_ppt ) {

           printf("   ppt gb interaction of type 3 or 4 with ppt %d\n",pptnum);
           printf("   ppt.x=%f, ppt.y=%f\n",ppt[pptnum].x,ppt[pptnum].y);
           printf("   point_on_ppt=%d, i=%d\n",point_on_ppt,i);

           for(k=0;k<segmtpts+2;k++) {
             dx=ppt[pptnum].x-x[k];
             dy=ppt[pptnum].y-y[k];
                                        printf("                 #=%d      d^2=%f
r^2=%f\n",k,dx*dx+dy*dy,ppt[pptnum].r*ppt[pptnum].r);
           }
           printf("   s1=%f, s2=%f\n",s1,s2);

           /*
           ** New points stranded on ppt - type 3 and 4
           */

            /* make sure, stop flag is cleared (otherwise it is possible we do
not
           ** detect some Xing */
           stop[i] = 0;
           /* flag do not do whole loop over again */
           redo    = 0;

           /* if segment(i-1) on ppt, an entry in table has to exist */
           if( point_on_ppt%2 ) {
             entry = -1;
             /* look for entry */
             for( k=0; (k<X_ppt_cnt) && (entry==-1); k++ )
              if( X_ppt_num[k]==pptnum ) entry = k;
             if( entry==-1 ) {
              printf("WARNING in gb_cross_ppt() - cannot find first entry.\n");
              printf("        redoing the whole grain boundary.\n");
              for( k=0;k<segmtpts+2;k++ )
                printf("point %d: x,y=%f,%f\n",k,x[k],y[k]);
              if( online ) scanf("%c",ch);
              for ( i=1; i<(segmtpts+2); i++ )
                stop[i] = 0;
              /* do gb over again */
              i = 0;
              j = cand_list_pos+1;
              redo = -1;
             }
           }

          /* if segment(i-1) not on ppt, create an entry in table */
          else {
            /* first check if ppt really not in list yet */
            entry = -1;
            for( k=0; (k<X_ppt_cnt) && (entry==-1); k++ )
             if( X_ppt_num[k]==pptnum ) entry = k;
```

```
        if( entry!=-1 ) {
         printf("ERROR in gb_cross_ppt() - ppt in list already.\n");
         WAIT_CHAR;
        }
        /* put ppt right at end of list */
        entry = X_ppt_cnt;
        X_ppt_cnt++;
        if( X_ppt_cnt==MAX_X_PPT_CNT ) {
         printf("ERROR in ppt_cross_gb() : X-table overflow.\n");
         WAIT_CHAR;
        }
        X_ppt_segment[entry][0]=im1; /* insert segment # on the left */
        X_ppt_flag[entry][0]=1;
        X_ppt_segment[entry][1]=-1;  /* end of list */
        X_ppt_num[entry] = pptnum;
      }

      if( redo==0 ) {
        /* get position in X_ppt_segment_table */
        segpos = 0;
        while( X_ppt_segment[entry][segpos] != -1 )
         segpos++;
        if( segpos == X_MAX_SEGMENTS ) {
         printf("ERROR in ppt_cross_gb() : X-table overflow.\n");
         WAIT_CHAR;
        }

        /* add segment[i] to list */
        if( point_on_ppt>=2 ) {
         /* segment(i) lies on ppt -> add it to X_ppt_segment-table */
         X_ppt_segment[entry][segpos]=i; /* insert segment # */
         X_ppt_flag[entry][segpos] = 0;  /* mark: it's on ppt */
         X_ppt_segment[entry][segpos+1]=-1; /* mark end of table */
        }
        else {
         /* segment(i) not on ppt -> mark end of X_ppt_segment-table */
         X_ppt_segment[entry][segpos]=i; /* insert segment # */
         X_ppt_flag[entry][segpos] = 2;  /* mark: it's to the left */
         X_ppt_segment[entry][segpos+1]=-1; /* mark end of table */
        }

     } /* end redo */
    } /* end point_on_ppt */

    else if( ((s1<1)&&(s1>0)) || ((s2<1)&&(s2>0)) ) {
     printf("   ppt gb interaction of type 2 with ppt %d\n",pptnum);
     printf("   ppt.x=%f, ppt.y=%f\n",ppt[pptnum].x,ppt[pptnum].y);
     printf("   s1=%f, s2=%f  i=%d\n",s1,s2,i);

     /*
     ** New grain boundary X ppt (type 2)
     */
      /* make sure, stop flag is cleared (otherwise it is possible we do
not
     ** detect some Xing */
     stop[i] = 0;
     /* check, if ppt already in list -> error */
     for( k=0;k<X_ppt_cnt; k++ )
       if( X_ppt_num[k]==pptnum ) {
        printf("ERROR in ppt_cross_gb() : gb X ppt more than once.\n");
        if( online ) scanf("%c",ch);
        X_ppt_cnt-=1; /* rough way to repair it */
       }
```

```
           /* check table overflow */
           if( X_ppt_cnt==MAX_X_PPT_CNT ) {
             printf("ERROR in ppt_cross_gb() : X-table overflow.\n");
             if( online ) scanf("%c",ch);
           }
           /* insert segments into end of X-table */
           X_ppt_num[X_ppt_cnt]        = pptnum;
           X_ppt_segment[X_ppt_cnt][0] = im1;
           X_ppt_segment[X_ppt_cnt][1] = i;
           X_ppt_segment[X_ppt_cnt][2] = -1; /* end of table */
           X_ppt_flag[X_ppt_cnt][0]    = 1;
           X_ppt_flag[X_ppt_cnt][1]    = 2;
           X_ppt_cnt+=1;
         } /* end of s1,s2 in 0..1 */

         else if ( (ppt[pptnum].x<xmax) && (ppt[pptnum].x>xmin)
                  && (ppt[pptnum].y<ymax) && (ppt[pptnum].y>ymin)) {
           /*
           ** There is a ppt very close to our point pair.
           ** This is a possible candidate for Type 1 boundary Xing.
           */
           printf("***POSSIBLY a Type 1 Interaction\n");
           scanf("%c",ch);
         }
       } /* end list over candidate-ppts */
      } /* end points-moved */
    } /* end loop over gb points */
  } /* end cand_list_pos!=0 */
} /* end ppt_cross_gb() */
```

If a segment point does encounter a particle, the function *gb_encountered_ppt* updates the **particle** data structure:

```
/*====================================================*/
/*
Handle cases where gb encountered a ppt.
X_ppt_cnt gives the number of ppts the gb crossed during move_segment().
*/
void gb_encountered_ppt(float *x, float *y, int *stop, float *x1mx0, float
*y1my0, float *len, int *segmtpts)
{
  int   i, j, k;
  int   num_segs_in_ppt;
  int   left_pos, quadrant;
  float pptx,ppty,pptr,s1x,s1y,s2x,s2y,dx,dy,proport,xnew,ynew;
  float dist1;

  /*
  ** Note: right, now, only X_ppt_cnt=1 is allowed. Otherwise,
  ** X_ppt_segment[][] has to be changed during adding/deleting
  ** segment points-> not implemented yet.
  */
  if( X_ppt_cnt>1 ) {
    printf("ERROR in gb_encountered ppt - cannot handle more than one ppt
yet.\n");
    if( online ) scanf("%c",ch);
  }

  /* loop over crossed ppts */
  for( i=0;i<X_ppt_cnt;i++ ) {

    printf("Grain boundary encountered a particle!\n");

    /*
    ** Check if data structure valid
    **   1. flags in order 1 0 ... 0 2
    **   2. no ppt flags set on segments
    */
    if( (X_ppt_flag[i][0] != 1) || (X_ppt_flag[i][1] == -1) ) {
      printf("ERROR in gb_encountered_ppt() - boundary segment(s) missing.\n");
      if( online ) scanf("%c",ch);
    }
    num_segs_in_ppt = 0;
    while( X_ppt_segment[i][num_segs_in_ppt+2]!= -1 ) {
      num_segs_in_ppt++;
      if( X_ppt_flag[i][num_segs_in_ppt] != 0 ) {
       printf("ERROR in gb_encountered_ppt() - segment not on ppt.\n");
       if( online ) scanf("%c",ch);
      }
    }
    if(         (X_ppt_flag[i][num_segs_in_ppt+1]         !=         2)          ||
(X_ppt_segment[i][num_segs_in_ppt+2] != -1) ) {
      printf("ERROR in gb_encountered_ppt() - boundary segment(s) missing.\n");
      if( online ) scanf("%c",ch);
    }
    /* check for ppt flag */
    for( j=1;j<num_segs_in_ppt+1;j++ )
      if( ppt1[X_ppt_segment[i][j]] ) {
       printf("ERROR in gb_encountered_ppt() - segment already on a ppt.\n");
       if( online ) scanf("%c",ch);
      }

    /*
    ** Remove all segments with flag 0 (ie segments inside ppt)
    */
    left_pos = X_ppt_segment[i][0];
    for( j=0;j<num_segs_in_ppt;j++ ) {
      /* delete point */
      ppt_delete( x, y, stop, x1mx0, y1my0, len, (left_pos+1), *segmtpts);
      *segmtpts = *segmtpts - 1;
    }

    /*
    ** Insert three more segments: two for ppt boundary,
    ** one for ppt center
    */

    pptx = ppt[X_ppt_num[i]].x;
    ppty = ppt[X_ppt_num[i]].y;
    pptr = ppt[X_ppt_num[i]].r;
    s1x  = x[left_pos];     s1y  = y[left_pos];
    s2x  = x[left_pos+1];   s2y  = y[left_pos+1];

    /* wrap x, y according to ppt */
    for( j=0;j<*segmtpts+2;j++ )
      rel_subwrap( &x[j], &y[j], pptx, ppty );

    /* insert left boundary point*/
    dx = s1x - pptx;    dy = s1y - ppty;
    dist1 = sqrt( dx*dx + dy*dy );
    proport = pptr / dist1 * PPT_TOLERANCE;
    xnew = (1-proport)*pptx + proport*s1x;
    ynew = (1-proport)*ppty + proport*s1y;
    ppt_insert( x, y, stop, x1mx0, y1my0, len, left_pos+1, xnew, ynew,
*segmtpts );
    *segmtpts = *segmtpts+1;
    ppt1[left_pos+1] = X_ppt_num[i];

    /* insert center */
    ppt_insert( x, y, stop, x1mx0, y1my0, len, left_pos+2, pptx, ppty,
*segmtpts );
    *segmtpts = *segmtpts+1;
    ppt1[left_pos+2] = X_ppt_num[i];

    /* insert right boundary point */
    dx = s2x - pptx;   dy = s2y - ppty;
    dist1 = sqrt( dx*dx + dy*dy );
    proport = pptr / dist1 * PPT_TOLERANCE;
    xnew = (1-proport)*pptx + proport*s2x;
    ynew = (1-proport)*ppty + proport*s2y;
    ppt_insert( x, y, stop, x1mx0, y1my0, len, left_pos+3, xnew, ynew,
*segmtpts );
    *segmtpts = *segmtpts+1;
    ppt1[left_pos+3] = X_ppt_num[i];

    /* take care of ppt.point */
    ppt_segment_pos[left_pos+1] = 0;
    ppt_segment_pos[left_pos+3] = 1;

    /*
    ** Remove ppt from in-grain-ppt list
    */
    for( quadrant=0;quadrant<ig_quadrantsx*ig_quadrantsy;quadrant++ ) {
      for( j=0;j<ig_num_ppts_in_quadrant[quadrant];j++ ) {
       if( ig_ppts_in_quadrant[ quadrant * ig_max_ppts_per_quadrant + j ] ==
X_ppt_num[i] ) {
         /* found ppt in ig_quadrant list. needs to be deleted */
         for( k=(j+1); k<ig_num_ppts_in_quadrant[quadrant]; k++) {
           ig_ppts_in_quadrant[ quadrant * ig_max_ppts_per_quadrant + k - 1 ] =
             ig_ppts_in_quadrant[ quadrant * ig_max_ppts_per_quadrant + k ];
         }
         ig_ppts_in_quadrant[ quadrant * ig_max_ppts_per_quadrant
                         + ig_num_ppts_in_quadrant[quadrant] - 1 ] = 0;
          ig_num_ppts_in_quadrant[quadrant] = ig_num_ppts_in_quadrant[quadrant]-
1;
       }
      }
    }/* end loop over quadrants */
    ppt[X_ppt_num[i]].grain = 0;

    /*
    ** Add ppt to on-boundary-ppt list
    */
    /* get quadrant # this point lies in */
    quadrant = ( (int) ( pptx / QUADRANT_SIZE )) + quadrantsx * ( (int) ( ppty
/ QUADRANT_SIZE ) );

    /* add ppt to table (if not already in there) */
    ppt[X_ppt_num[i]].point[0] = -1;       /* temporary marker: gb-ppt */
    check_ppts_in_quadrant(X_ppt_num[i], pptx, ppty );
    check_ppts_in_quadrant(X_ppt_num[i], pptx + pptr * (2-PPT_TOLERANCE), ppty
+ pptr * (2-PPT_TOLERANCE));
```

```
    check_ppts_in_quadrant(X_ppt_num[i], pptx - pptr * (2-PPT_TOLERANCE), ppty
+ pptr * (2-PPT_TOLERANCE));
    check_ppts_in_quadrant(X_ppt_num[i], pptx + pptr * (2-PPT_TOLERANCE), ppty
- pptr * (2-PPT_TOLERANCE));
    check_ppts_in_quadrant(X_ppt_num[i], pptx - pptr * (2-PPT_TOLERANCE), ppty
- pptr * (2-PPT_TOLERANCE));
    ppt[X_ppt_num[i]].point[0] = 0;        /* clear temp marker again */

    for( j=0;j<num_ppts_in_quadrant[quadrant];j++ ) {
      if( ppts_in_quadrant[ quadrant * max_ppts_per_quadrant + j ] ==
X_ppt_num[i] ) {
       j = num_ppts_in_quadrant[quadrant]+1;
      }
    }
    if( j != num_ppts_in_quadrant[quadrant]+2 ) {
      /* beware of table overflow */
      if( num_ppts_in_quadrant[quadrant]==max_ppts_per_quadrant ) {
       printf("ERROR - quadrants array overflow, program aborted.\n");
       if( online ) scanf("%c",ch);
      }
      /* add it to table */
      ppts_in_quadrant[ quadrant * max_ppts_per_quadrant +
num_ppts_in_quadrant[quadrant] ] = X_ppt_num[i];
      num_ppts_in_quadrant[ quadrant ]++;
    }
  } /* end loop over i (0..X_ppt_cnt-1) */
}
```

The function *bamboo_check* finds boundaries traversing a strip, and removes the segment points on such boundaries. Instead of *bamboo_check*, the program uses *ppt_erase_bamboo* which makes sure that segment points lying on the circumference of particles are not removed during this process. The function *triple_update*, which determines the equilibrium triple point position, has been modified to only move a triple point if it does not lie on a particle. However, if the triple point is moved, it is checked if the triple point encountered a particle. The function *reorient* performs a reorientation, or neighbor-switching event. Reorientations are not allowed if one of the triple junctions is covered by a particle.

E.1.4 Other Modifications

The current grain structure configuration can be printed into the file *gbplot.dat* by calling the function *WriteGraphic*. We extended this function so that the file *gbplot.dat* contains the number of particles in the film:

```
if( flag.particle ) {
  n_ppt = 0;
  for (i=1 ; i < (ppt_max+1) ; i++)
   if (ppt[i].free>=0) n_ppt++;
  fprintf (fgraphics," %d %d %d %d %d\n",0,0,S.n,n_ppt,n_grn);
 }
 else {
  fprintf (fgraphics," %d %d %d %d %d\n",0,0,S.n,0,n_grn);
 }
```

In addition, the function *ppt_write_segment* located in the file *ppt_modifications.c* is called at the end of the function *WriteGraphic* to append the particle positions and radii to the end of the file *gbplot.dat*:

```
/*==================================================*/
/*
Called from Write_Seg (graphics.c):
Write the ppt into the "gbplot.dat" file.
*/
void ppt_write_seg()
{
 int   i;
 float x, y, u, v;

 /* add the data about the particles =SPR= */
 for (i=1 ; i<(ppt_max+1) ; i++) {
  if( ppt[i].free>=0 ) {
   if(strip) {
   u = ppt[i].x; v = ppt[i].y;
   u += ppt[i].free*xsize;
   if( (ppt[i].free==n_regions-1) && v<b_inc ) v+=ysize;
   if (stripcoords(&x,&y,u,v)==0) return;
   while (x > uwind) {
    x -= uwind;
    y += 1.25*s_vsize;
   }
   while (x < 0.0) {
    x += uwind;
    y -= 1.25*s_vsize;
   }
   if ( (y > vwind - .20*s_vsize) && (ppt[i].free!=n_regions-1) ) {
         y -= vwind;
       }
       if (y < 0.0 - .20*s_vsize) {
         y += vwind;
       }
       fprintf(fgraphics,"%f %f %f\n",x,y,ppt[i].r);
      }
      else
       fprintf(fgraphics,"%f %f %f\n",ppt[i].x,ppt[i].y,ppt[i].r);
    }
  }
}
```

E.2 Using the Grain Growth Simulator with Particles

E.2.1 Adding Particles to a Continuous Film

A continuous film is created by continuously nucleating grains, and letting the grains grow until impingement [Fro 88, Fro 90]. The command

```
nucl_impinge 15 15 1
```

creates the directory *15_15_1* in the directory *io* along with files defining the data structure of the film. The film is initialized and annealed until simulation time $\tau = 0.7$. At this point, the film has reached a steady-state grain-size distribution. The command

```
anneal init
```

reads the command file *init.com*, creates the director *15_15_1_init*, and starts the evolution of the grain structure of the film. The results can be displayed using the command

```
View -d,
```

excecuted inside of the directory *15_15_1_init*. The command file *init.com* is

```
0 "init.com"
0 Initialization sequence
0
0 100 -> directory containing nucl_impinge files
0 101 -> directory containing initialized structures
0
100  15_15_1
200  15_15_1_init
1  0.001        mobility
2  0.080        tooclose
3  0.000        critical curvature
5  15_15_1_init
6  15_15_1_init
9  0            graphics dump
7  0.7          simulate a tiny little bit
9  0            graphics dump
11              dynamic dump
12              exit simulation
```

The command file *addppt.com*, excecuted by

```
anneal addppt,
```

adds particles to the film and writes the resulting data structure into *15_15_1_ppt*:

```
0  "addppt.com"
0  Add particles to sheet film
0
5 15_15_1_addppt
6 15_15_1_init
40 1 0.2 0.02 0 30 0           addppt (constant radii)
   0.07 0.08 0.06              radius ppt on segment, triple, grain
41                             ppt statistics
9 0                            normal gfx
11                             dynamic data dump
12                             end simulation
```

The command 40 requires the following parameters: (i) a random seed, (ii) the probability that a triple junction is covered by a particle (0...1), (iii) the probability that a segment point is covered by a particle (0...1), (iv) the maximum number of particles in the film (0 means that there is no limit), (v) the number of tries to put particles anywhere in the film inside of grains, and (vi) an identifier determining the particle-size distribution function. If the identifier for the particle-size distribution function is zero, the data in the next line determines the radii of particles at segment points and at triple points, and for intragranular particles. If the identifier for the particle-size distribution function is one, a box-shaped distribution function is assumed. In this case, the data in the next line determines the average diameter of particles at segment points, triple points, and intragranular particles, respectively, and the respective spread of the three distributions.

E.2.2 Annealing a Continuous Film with Particles

After particles have been added to a film, it is possible to let the grain structure evolve using the standard annealing commands for the grain growth simulator. For example,

```
anneal annealppt
```

executes the command file *annealppt.com*, which is:

```
0 "annealppt.com"
0 Annealing sequence for sheet with precipitates
0
0 5 -> output directory
0 6 -> input directory
0
1  0.001                        mobility
2  0.080                        tooclose
3  0.000                        critical curvature
5  15_15_1_annealppt            output directory
6  15_15_1_addppt               input directory
9  0                            graphics dump
7  1.0
9  0
7  5.0
```

```
9  0
7  30.0
9  0
12                                exit simulation
```

This command file creates the directory *15_15_1_annealppt,* starts the grain growth process, and writes the results of the simulation and the data structure describing the final microstructure into files inside of this directory.

E.2.3 Annealing a Stripe Structure with Particles

A stripe structure is created by executing the script file

```
0  "etchppt.com"
0  Etch a continuous film with particles.
0
5  15_15_1_pptetch        output
6  15_15_1_addppt         input
9  0                      graphics dump sheet
60 4                      etch into 4 strips
9  1                      graphics dump strips
11                        dynamic dump
12                        exit
```

using the command

```
anneal etchppt.
```

The data structure for the etched film is located in files inside of *15_15_1_pptetch.* The command

```
anneal annealpptetch
```

starts the grain structure evolution in the etched film. The command file *annealpptetch.com* is:

```
0 "annealpptetch.com"
0 Annealing sequence for lines with precipitates
0
0 5 -> output directory
0 6 -> input directory
0
1  0.001                        mobility
2  0.080                        tooclose
3  0.000                        critical curvature
5  15_15_1_annealpptetch        output directory
6  15_15_1_pptetch              input directory
9  1                            graphics dump
7  1.0                          anneal
9  1                            graphics dump
7  5.0                          anneal
9  1                            graphics dump
7  30.0                         anneal
9  1                            graphics dump
11                              save data
12                              exit simulation
```

E.2.4 Particle Kinetics

During the simulation of grain growth, it is possible to let particles grow or shrink at a constant rate until they dissolve. It is also possible to let particles evolve according to the LSW theory [Chr 65]. As an example for the former, the command file *shrink.com* dissolves all particles while letting the grains structure evolve:

```
0 "shrink.com"
0 Annealing sequence for sheet with precipitates
0 Particles shrink while annealing
0
0 5 -> output directory
0 6 -> input directory
0
1  0.001                        mobility
2  0.080                        tooclose
3  0.000                        critical curvature
5  15_15_1_shrink               output directory
6  15_15_1_addppt               input directory
43 0 -0.04                      particle kinetics: shrink
9  0                            graphics dump
7  1.0                          anneal
9  0                            graphics dump
7  5.0                          anneal
9  0                            graphics dump
7  30.0                         anneal
9  0                            graphics dump
11                              save data
12                              exit simulation
```

The shrink rate, k, assumed to be -0.04 in this example, relates the change in particle radius, Δr, to an increment in simulation time, $\Delta\tau$, by

$$\Delta r = k \cdot \Delta\tau . \tag{E.1}$$

The following command file allows particles to evolve according to the LSW theory:

```
0 "lsw.com"
0 Annealing sequence for sheet with precipitates
0 Particles coarsen according to LSW theory
0
0 5 -> output directory
0 6 -> input directory
0
1  0.001                          mobility
2  0.080                          tooclose
3  0.000                          critical curvature
5  15_15_1_lsw                    output directory
6  15_15_1_addppt                 input directory
43 1 0.004                        LSW coarsening of particles
9  0                              graphics dump
7  1.0                            anneal
9  0                              graphics dump
7  5.0                            anneal
9  0                              graphics dump
7  30.0                           anneal
9  0                              graphics dump
11                                save data
12                                exit simulation
```

In this case, the shrink rate, k, assumed to be 0.004, relates the change in particle radius, Δr, to an increment in simulation time, $\Delta\tau$, by

$$\Delta r = \frac{k}{r}\left(\frac{1}{< r >} - \frac{1}{r}\right)\Delta\tau , \tag{E.2}$$

where <r> is the average radius of the particles at time τ.

Appendix F Grain Structure Evolution in the Presence of Particles

F.1 Simulation of the Influence of Particles on Grain Structure Evolution in 2D Systems and Thin Films

F.1.1 Introduction

Polycrystalline thin films are used in a wide variety of applications in which their properties, performance and reliability are strongly dependent on the detailed characteristics of their microstructure. For example, the distribution of grain sizes and orientations determines the yield strength, the fracture characteristics and the creep mechanisms of thin films [Nix 89]. Also details of the grain structures of films patterned for use as wires which interconnect devices in integrated circuits are known to profoundly affect their reliability through influences on electromigration [Tho 93a, Kno 97]. The final grain sizes in both continuous and patterned thin films can be controlled by adding impurities which lead to second phase particles [Gan 73, Lon 91, Kno 95, Wei 98]. However, in order to use the effects of particles to tailor the final microstructures of thin film materials, for optimization of performance and reliability, a detailed understanding of the effects of particles on grain structure evolution is required.

Of special technological interest is the grain structure in metallic materials systems used for interconnects in highly complex integrated circuits, such as aluminum doped with a few weight percent copper. In-situ TEM studies of grain growth in annealed Al(1 wt%Cu) thin films showed a significant effect of Al_2Cu particles on grain

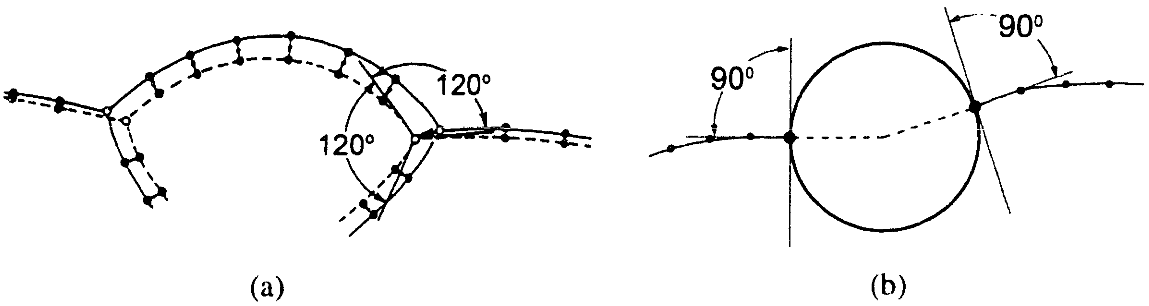


Figure F.1.1: Schematic illustrations of the simulation techniques (a) for modeling grain boundary migration and triple point motion, and (b) the interaction of particles with grain boundaries.

growth and the resulting characteristics of the grain structures [Kno 95a]. Intergranular particles effectively pin grain boundaries, and moving grain boundaries are captured and pinned by intragranular particles. Particles not only affect the kinetics of grain growth and lead ultimately to the stagnation of grain growth, they also affect the statistical characteristics of the stagnant grain structure. The latter is known to have a direct effect on the failures statistics of interconnects [Kno 97], and is likely, by analogy, to affect the statistics of reliability of films in other applications.

In this paper, we report on modifications of a 2D front-tracking grain growth simulation to treat the effects of precipitate pinning on grain growth in polycrystalline thin films. The goal is to investigate precipitate-induced stagnation, and to obtain statistical grain size and topological characteristics of the stagnant structures.

F.1.2 Simulation Techniques

To simulate grain growth, we have developed a curvature-driven front tracking model [Fro 88, Fro 90, Fro 92, Wal 92, Car 96]. The velocity, v, of each grain boundary segment is proportional to the local curvature, κ, through a constant of proportionality, μ, which is defined as the grain boundary mobility. At grain boundary triple junctions, a local force balance is enforced, so that grain boundaries meet at 120°, as illustrated in figure F.1.1 (a). Periodic boundary conditions are imposed on the area of simulation. In the simulations reported on here, we start with Johnson-Mehl structures that result from continuous nucleation with a constant nucleation and growth rate [Fro 87]. When grain boundary motion is allowed in these structures, after a short time on the simulation time-scale, a steady-state structure develops in which the normalized grain-size and number-of-sides distributions are time invariant, and for which $(A-A_0)/A_0$ is linear in time, where A is the average grain area at simulation time τ, and A_0 is the average grain area of the structure at the beginning of the simulation. These are taken to be the defining characteristics of 2D "normal grain growth".

We have shown earlier that this grain growth model can be extended to account for phenomena which impede grain boundary motion, such as solute drag [Fro 94] or grain boundary grooving [Fro 90]. Frost et.al. showed that the effects of solute drag can be accounted for by modifying the relationship between grain boundary mobility and the boundary curvature. At high driving forces, corresponding to high curvatures, the boundaries are given a mobility corresponding to drag-free motion. At low driving forces, corresponding to curvatures less than some critical value, the boundaries are given a lower mobility which models the effect of solute drag. When the boundary curvatures begin to fall below the critical curvature during grain growth, the grain size distribution evolves to a lognormal distribution, as is observed in experiments. Frost et al. also showed that the effect of grain boundary grooving, as described by Mullins [Mul 58], can be accounted for by assuming that a grain boundary becomes locally stagnant due to groove formation when its local curvature falls below a critical level corresponding to a critical in-plane curvature κ_{crit} [Fro 90]. The critical curvature is inversely proportional to the film thickness. If this stagnation criteria is enforced on a steady-state structure with an average curvature significantly larger than κ_{crit}, and the simulation is continued, microstructure evolution eventually stagnates with a lognormal grain size distribution with an average in-plane grain size related to κ_{crit} so as to be proportional to the film thickness, as is observed in experiments.

We now report modification of the 2D normal grain growth simulation to account for the effects of particle pinning. Particles with arbitrary shape and size distributions can be introduced inside grains, on grain boundaries, or on grain boundary triple junctions. However, in this paper we have considered monosized circular particles that do not change their size or location with time. Grain boundaries are represented in the simulation as a series of grain boundary points. We place particles, and then create new grain boundary points where grain boundaries intersect the precipitate perimeters (see

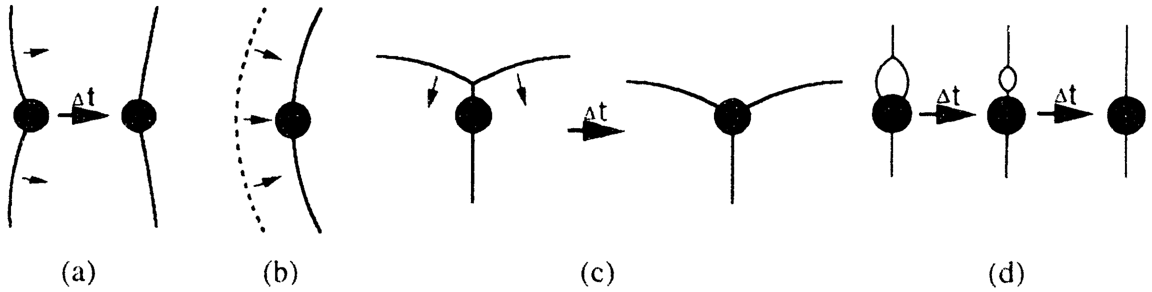


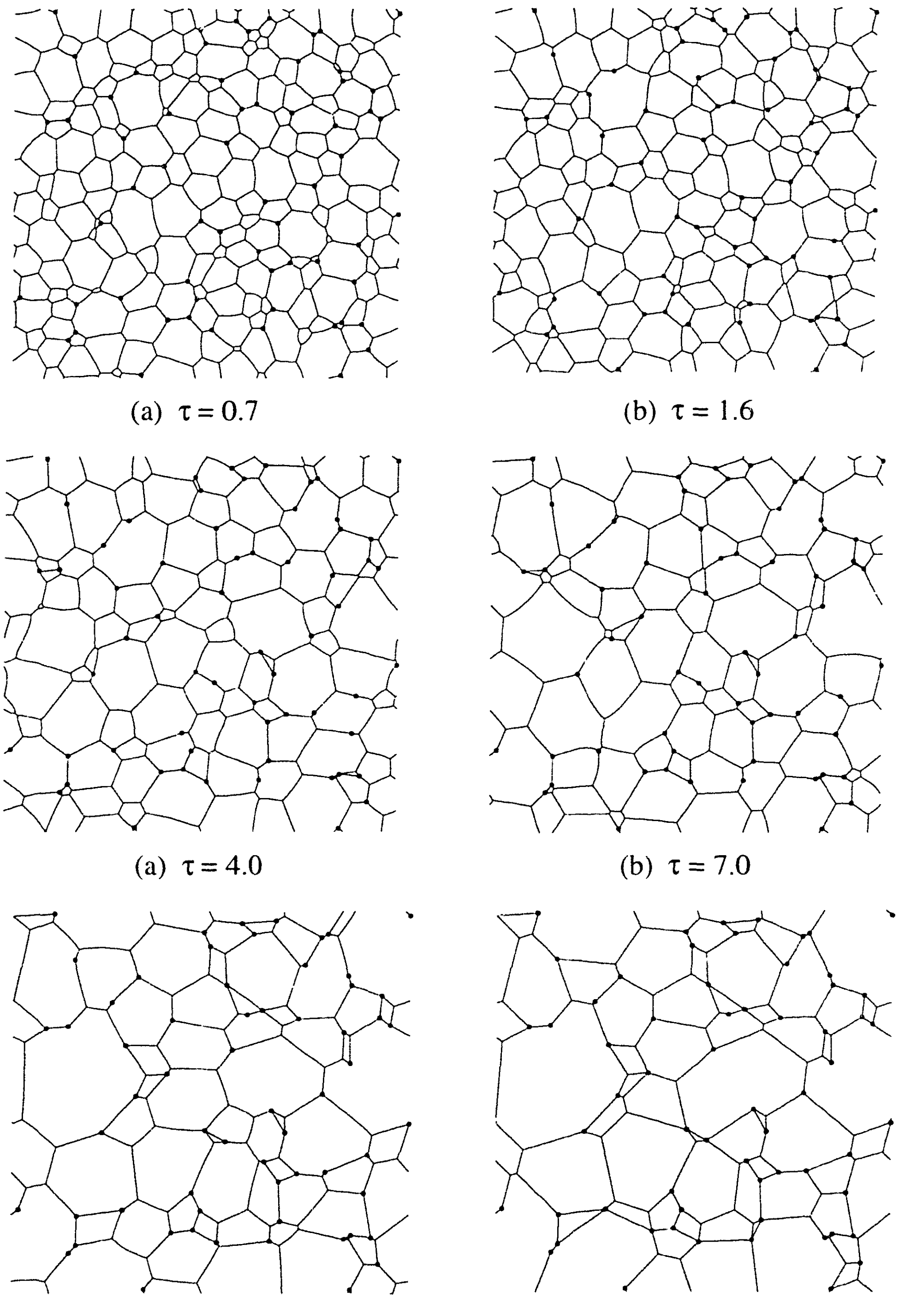
Figure F.1.2: Examples of the necessary modifications of the grain growth simulator to account for particle pinning. (a) Particles pin grain boundaries, (b) capture grain boundaries if overrun, and (c) capture grain boundary triple junctions if they run into a precipitate. If a grain lens lies at a precipitate and collapses, it gets deleted (d).

figure F.1.1 (b)). The positions of these points are updated in every time step, in a way which forces the grain boundary to contact the particle perimeters at right angles. This represents the physical situation in which the interface energies between grains and particles are isotropic.

Common examples of the local effects of particles on grain growth which are dealt with are sketched in figure F.1.2. When grain boundaries are pinned by particles (figure F.1.2 (a)), for each time step $\Delta\tau$ of the simulation, the grain boundary segment points nearest to the precipitate are moved according to $v = \mu\kappa$. Subsequently, the segment points at the intersection of grain boundaries and particles are repositioned in order to enforce a 90^o contact angle. We assume isotropic surface and interface energies, which implies that grain boundaries can not meet on the precipitate-grain interface so that a boundary can not break free from a precipitate once it has been trapped by one (though two boundaries can meet to form a triple point as in figure F.1.2 (d)). The time steps $\Delta\tau$ of the simulation are chosen to be small enough that the distances the grain boundaries move are much smaller than the radii of the particles. In the case illustrated in figure F.1.2 (b), boundaries are captured by particles. In this case, when it is found that any part of a particle will lie between two boundary points after a time step, two new boundary points are created on the particle perimeter and their positions are chosen so as to give 90^0 contact angles. Again, the spacing between the segment points on the relevant boundaries is readjusted to ensure that the segment points are equally spaced. If a grain boundary triple junction moves into a precipitate, as in figure F.1.2 (c), it is captured as well. The sequence of events in this case is essentially the same as when a grain boundary crosses a precipitate. Small grains at particles often collapse as shown in figure

F.1.2 (d). If the two segment points on the precipitate border would meet during a time step, a new triple junction and boundary segment are generated and a lens-shaped grain is formed. The triple junction is placed directly adjacent to the precipitate. This lens-shaped grain then collapses (rapidly) in the normal way.

In all the simulations to be described here, we started with steady-state grain structures consisting of 17,000 grains and with an average grain diameter of d = 1.74 (where $d \equiv \sqrt{4A/\pi}$, and A is the average grain area), and populated these grain structures with particles of radius r = 0.07. In the simulation, length-scales relate to time by the grain boundary velocity $v = \mu\kappa$, where μ is the grain boundary mobility, and κ the local curvature of the grain boundary. Time is measured in dimensionless units as $\tau = t\mu/A_0$, where A_0 is the initial average grain area at the time of particles placement. The precipitate area fraction f was varied from $2.6\text{x}10^{-4}$ to $1\text{x}10^{-2}$ by changing the number of particles per unit area, N, from $1.7\text{x}10^{-2}$ to $6.5\text{x}10^{-1}$, but keeping the radius of the particles, r, constant at 0.07. We also varied the particle radius from 0.03 to 0.40 for constant $N = 8.6\text{x}10^{-2}$. Of practical importance for integrated circuit metallization is the case in which the particle radius is much smaller than the average grain diameter. In this regime, the effect of particles on grain structure evolution is expected, to first order, to be independent of the particle size [Has 90]. We will report our results as a function of the number of particles per unit area, N, because the average grain diameter at stagnation is inversely proportional to the square of the average interparticle separation distance in these simulations. We have mainly considered the effects of particles originally placed at triple junctions. However, for the purpose of comparison, we have also considered randomly placed particles. As will be described below, the observed differences in grain structure evolution in these two cases are minor.



(a) $\tau = 0.7$

(b) $\tau = 1.6$

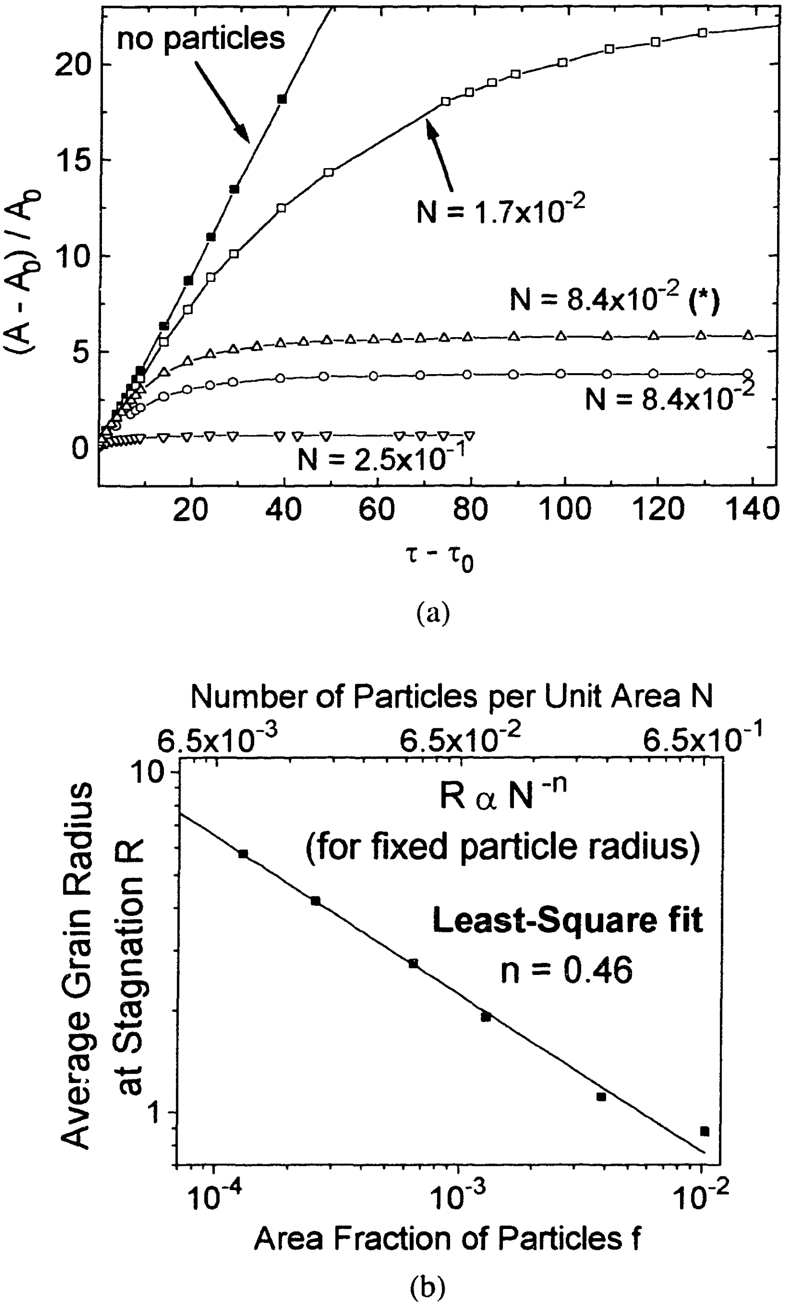
(a) $\tau = 4.0$

(b) $\tau = 7.0$

(a) $\tau = 14.0$

(b) $\tau = 50.0$

Figure F.1.3: Evolution of a grain structure in the presence of particles at various normalized times $\tau = \mu t/A_0$. μ is the grain boundary mobility and A_0 the initial average grain area.



(a)

(b)

Figure F.1.4: (a) Evolution of the normalized average grain area with and without precipitate-pinning, but without the effects of grain boundary grooves. (b) The dependence of the average grain size in stagnant structures as a function of the number of particles per unit area, N. For these simulations, the particles were initially randomly placed only on grain boundary triple junctions, except for the case marked by *, for which particles were initially placed at random throughout the simulated area. The stagnated structures consisted of approximately 1,700 grains (small f) and 63,000 grains (large f).

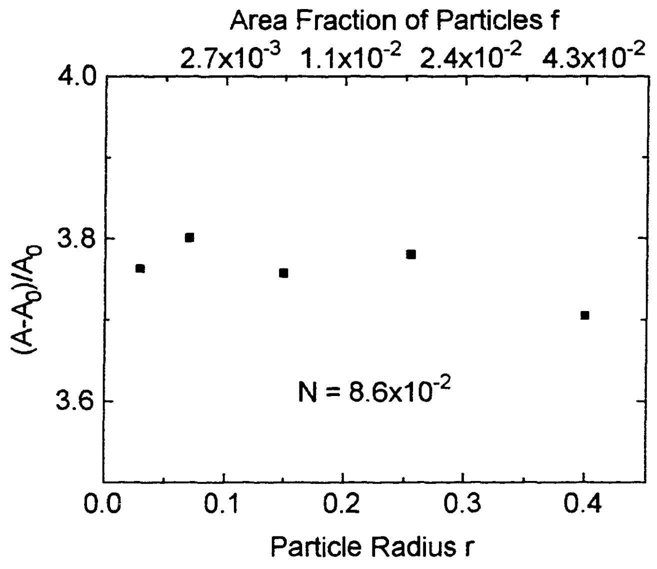
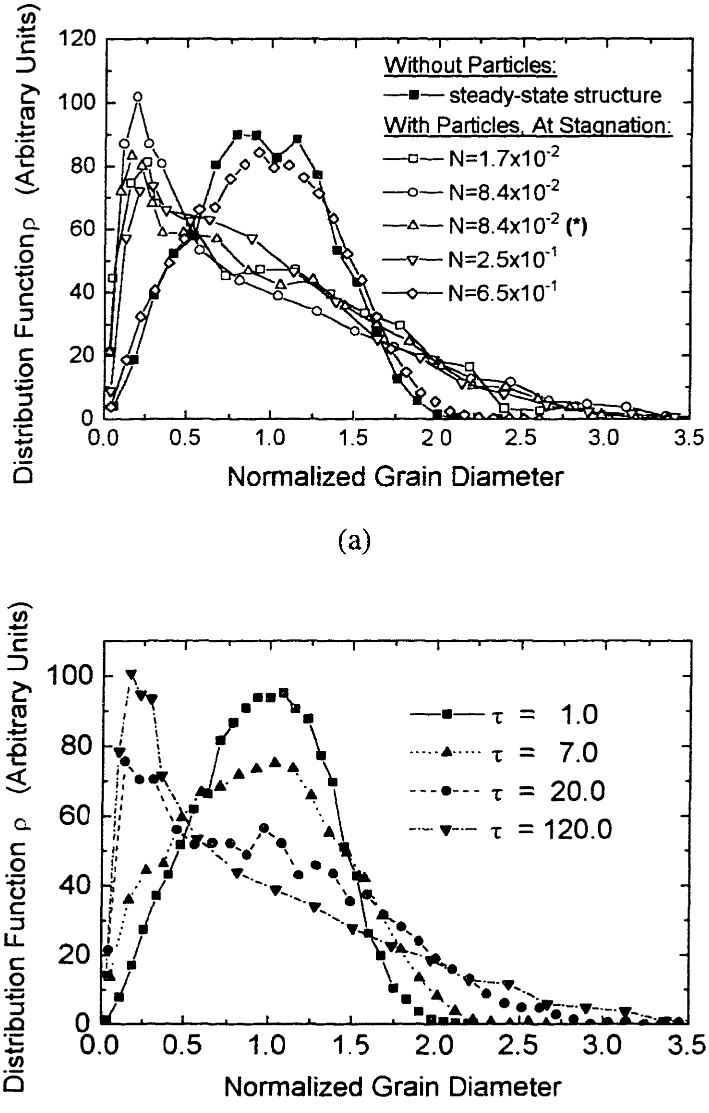


Figure F.1.5: The average grain area A in stagnant structures which evolved from structures with initial average grain area A_0 as a function of the particle radius r for constant N = 8.6x10^{-2}. For these simulations, the particles were initially randomly placed only on grain boundary triple junctions.

F.1.3 Simulation Results

We will first describe simulation results for grain growth in the presence of particles which have been randomly placed on triple junctions, without accounting for the effects of grain boundary grooving. An example of grain structure evolution under these conditions is shown in figure F.1.3. The number of particles per unit area is N = 4.5x10^{-1}, and the average grain area increases by a factor of 31 before stagnation occurs. We find that even a small area fraction of particles retards grain growth, lowers the ultimate average grain size, and leads to significant changes in the grain size distribution.

Figure F.1.4 (a) shows the evolution of the average grain area for different numbers of particles per unit area. As discussed earlier, if no particles are present in the film, the average normalized grain area $(A-A_0)/A_0$ is proportional to τ. However, with particles present in the structure, the normalized average grain area reaches a fixed saturation value. In our simulation, this transition is gradual and we do not observe the incubation, steady state, and abrupt stagnation stages reported for precipitate-induced stagnation in the Potts model [Kad 97]. The lower the number of particles per unit area, N, the later the stage at which the effects of the particles on grain growth becomes important. In agreement with experiments on bulk systems [Olg 86], for a fixed particle size, the mean grain size at stagnation increases with decreasing N. This is demonstrated in figure F.1.4 (b), in which the final average grain radius $R \equiv \sqrt{A/\pi}$ is plotted as a function of the number of particles per unit area, N, for constant particle radius. A least squares fit of the log-log data shows that R is proportional to $N^{-0.46}$, which is similar to the dependence found using the Potts model [Sro 84], $R \alpha N^{-0.5}$, and which, as will be discussed below, consistent with expectations for precipitate pinning in 2D systems [Doh 87]. Varying the particle radius r for constant number of particles per unit area, $N = 8.6 \times 10^{-2}$, does not change the average grain radius at stagnation, as shown in figure F.1.5.



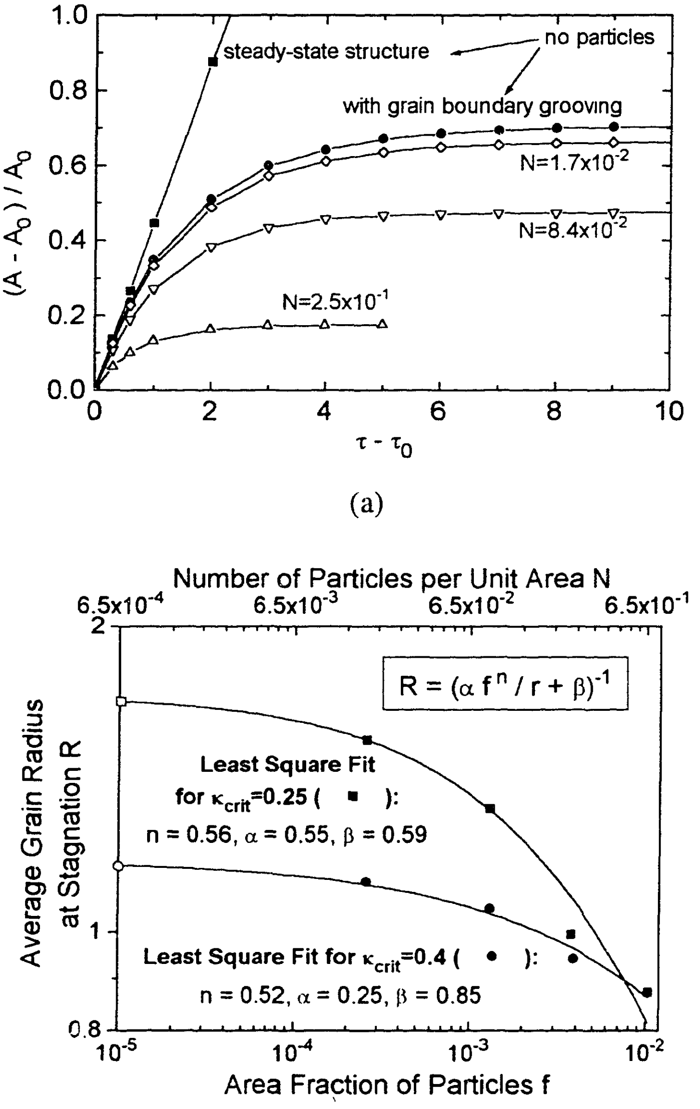
(a)

(b)

Figure F.1.6: The grain size distribution, taken as the distribution of grains among normalized (with respect to the average) grain diameter, with and without the effects of particles but *not* including the effects of grain boundary grooves: (a) at steady state or stagnation, and (b) evolution for N = 8.4×10^{-2}. For these simulations, the particles were initially randomly placed only on grain boundary triple junctions.

A more quantitative understanding of the effects of particles on grain structure evolution can be obtained by examining the grain size distributions at stagnation, as shown in figure F.1.6 (a) for different particle area fractions. As mentioned above, in the absence of particles, the steady-state normalized grain size distribution, shown by the filled squares in figure F.1.6 (a), is time-invariant. For a high number of particles per unit area (e.g., $6.5x10^{-1}$), little evolution occurs before stagnation, so that the grain size distribution does not change significantly. However, at lower particle area fractions, the grain size distribution evolves to a distinct shape before stagnation. Figure F.1.6 (b) shows the time-evolution of the grain size distribution for $N = 8.4x10^{-2}$. It can be seen that the distribution first flattens out, and then a peak develops at smaller grain sizes. A majority of the grains in the pinned structure are very small compared to the average grain diameter $d \equiv \sqrt{4A/\pi}$, so that the grain sizes corresponding to the peaks of the distributions and also the median grain size d_{50} are significantly below d. Also, these distributions "spread out" so that there is an increased fraction of larger grains compared to the steady-state distribution. This distinct particle-induced grain size distribution at stagnation observed using this front-tracking model is not observed in grain growth simulations based on the Potts model [Sro 84]. The grain size distribution reported for pinned structures in reference [Sro 84] is much more similar to the distribution obtained without particles. Also, the pinned grain size distribution predicted using the Potts model peaks near the average grain size.

All of the observations described above are independent of the initial particle distribution. We obtained similarly shaped grain size distributions when particles were initially distributed randomly throughout the simulation area and not just on triple junctions, as illustrated for $N = 8.4x10^{-2}$ seen in figure F.1.6 (a). In this example. the average grain size grows slightly faster and stagnates at a somewhat larger average grain size than in the case when the particles are initially all located on grain boundary triple junctions.



(a)

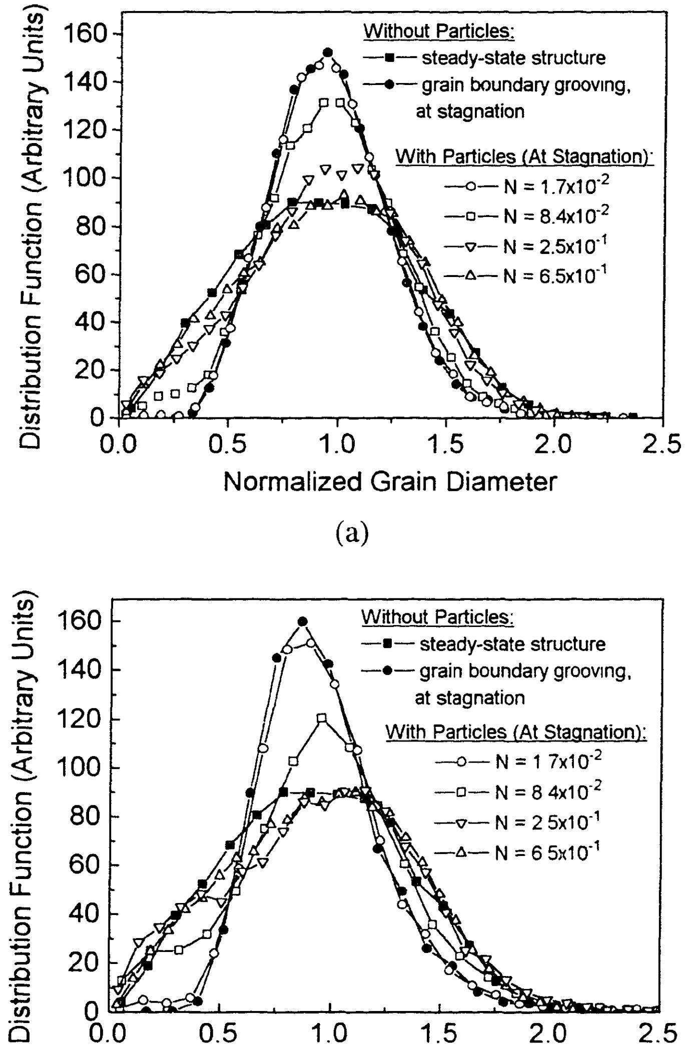
(b)

Figure F.1.7: (a) Evolution of the normalized average grain area with and without precipitate-pinning, *including* the effects of grain boundary grooves. (b) The dependence of the average grain size in stagnant structures as a function of the number of particles per unit area, N. For these simulations, the particles were initially randomly placed only on grain boundary triple junctions.

We next consider the consequences of including the effects of other mechanisms which impede grain boundary motion. Here we will focus on grain boundary grooving in conjunction with the effects of particle pinning. We observe similar trends if we account for the effects of solute drag. It is assumed that $v = \mu\kappa$ only when the local curvature κ is greater than or equal to κ_{crit}, and zero when $\kappa < \kappa_{crit}$. As mentioned above, we expect different results for different film thicknesses, because the effect of grain boundary grooving becomes stronger in thinner films. Specifically, $\kappa_{crit} = \gamma_s / (\gamma_{gb} h)$, where γ_s is the energy of the film surface and γ_{gb} is the grain boundary energy .

It can be seen from the evolution of the average normalized grain area shown in figure F.1.7 (a), that when both the effects of grooves and particles are accounted for, the transition to saturation is still gradual. Figure F.1.7 (b) shows the dependence of R on the number of particles per unit area. Clearly the dependence of R on N is different from what is observed when only particle pinning is present (figure F.1.4 (b)).

When both groove-induced and precipitate-induced stagnation occurs, the grain size distribution is affected by a competition between these two effects. For small number of particles per unit area ($N = 1.7 \times 10^{-2}$ in figure F.1.8 (a)), the grain size distribution at stagnation is the same as is found without particles. In this case, grain boundary grooves lead to stagnation before the average grain size has become large enough that precipitate pinning becomes important. With increasing area fraction of particles, though, precipitate-pinning becomes more and more important in defining the nature of the stagnant grain structure. However, an evolution toward the grain size distributions observed without grain boundary grooving *does not* take place. For the physically reasonable value of $\kappa_{crit} = 0.4$ used in this simulation, particle pinning does not dominate over groove-induced pinning. Particle pinning can at most add to the effects of grooves. It is interesting to note that when both types of pinning are important, the grain size distribution at stagnation becomes similar to the steady-state distribution. These effects are also seen when a smaller value of κ_{crit} is used (0.25), as shown in figure F.1.8



(a)

(b)

Figure F.1.8: The grain size distribution, taken as the distribution of grains among normalized (with respect to the average) grain diameter, with and without the effects of particles and *including* the effects of grain boundary grooves for: (a) κ_{crit}=0.4, and (b) κ_{crit}=0.25. For these simulations, the particles were initially randomly placed only on grain boundary triple.

(b). An increase in κ_{crit} above 0.4 would further suppress the effect of the particles on the grain structure evolution.

Figure F.1.9 shows the distribution of grains among topological number-of-neighbors categories (or equivalently, number-of-sides categories). Again, this

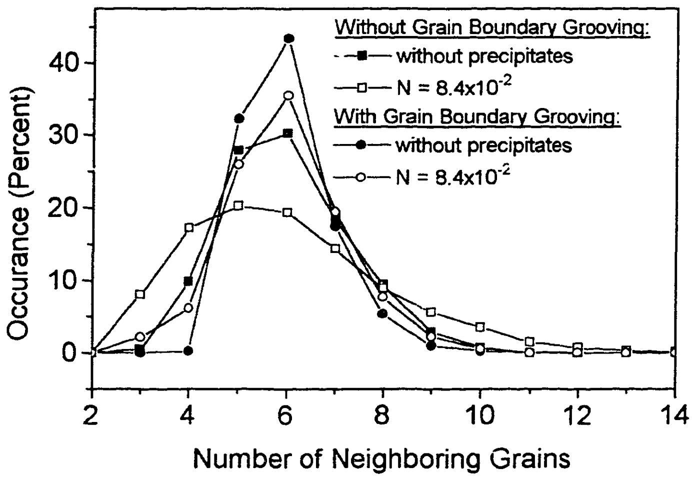


Figure F.1.9: The distribution of the number or neighboring grains at stagnation or steady state. For the case with grain boundary grooving, κ_{crit}=0.4.

distribution is time invariant for normal grain growth without stagnation-inducing phenomenon. The distributions for stagnant structures are different when only precipitate effects are considered, from what is observed when both particles and grooves affect the stagnation process. Figure F.1.10 shows the normalized average grain area for grains with a given number of sides n for stagnant structures resulting from particle pinning without groove-induced pinning.

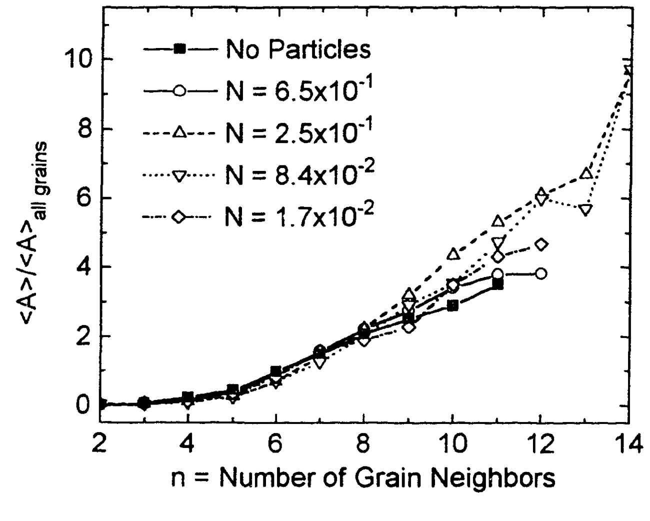


Figure F.1.10: The normalized average grain area for grains with n neighbors or sides without the effect of grain boundary grooving.

F.1.4 Discussion

Insight into the origin of the differences in the grain size distributions resulting from stagnation and the steady-state grain size distribution can be obtained by considering the topological information given in figures F.1.9 and F.1.10. In the steady state, it has been shown [Fan 97] that, on average, the area of an n-sided grain obeys the Mullins-von Neumann relationship in that the rate of change of the area is proportional to (n-6) [Neu 52, Mul 56]. It is therefore not surprising that 3-sided grains rapidly disappear and are very rare in the steady-state structure, and also that 4-sided grains are rare as well. In the case of particle pinning however, if particles function as triple points (as in figure F.1.2 (c)), the requirement that boundaries meet at 120 degrees is removed so that the boundaries of 3 and 4 sided grains can have zero curvature and stabilize. The stagnant structures therefor have a much larger fraction of three and four sided grains than the steady-state structures. As seen in figure F.1.10, the average grain area in both the steady-state and stagnant structures increases with the number of sides, and is especially small for 3 and 4 sided grains, so that the stagnant structures have a greater fraction of smaller grains. In the steady-state structure, it follows from Euler's theorem [Sta 93] that the average number of sides per grain is 6. While this expectation is no longer exact for stagnant structures, since 4-fold vertices (at particles) are possible, it is still very nearly true since 4-fold and higher order vertices are very rare. A relative increase in the number of 3 and 4 sided grains must therefore correspond with an increase in the number of grains with $n > 6$, and therefore in a relative increase in the number of large grains.

This qualitatively explains the features of the grain size distributions when only particle pinning is accounted for.

When only groove-induced pinning is considered, the number of 4 sided grains is even lower than in the steady-state structures, and there are also fewer large grains. This is a consequence of the fact that grains with numbers of sides most different from 6 will have boundaries which are moving relatively fast (as expected from the Mullins-von Neumann relationship) and are less likely to reach the stagnation condition. This is counter to the effects of particle pinning, so that when both groove-induced and particle pinning are considered, their effects on the number-of-sides and grain-size distributions tend to compete, leading to the more limited differences from the steady-state distributions that are observed when both stagnation phenomena are included than when either stagnation process is considered in isolation.

We next consider the effects of particles, and the combined effects of particles and grain boundary grooves on the average grain size in stagnant structures. Zener estimated the interaction force between a grain boundary and spherical particles, and he also estimated the average grain radius R after precipitate-induced stagnation of grain growth [Zen 49]. Several refinements to Zener's treatment have been put forward [Olg 86, Wör 89]. A simplified expression for the Zener relation which summarizes most of the revisions is given by

$$\frac{R}{r} = \frac{\lambda}{f^n}, \qquad \text{(F.1.1)}$$

where λ and n are constants, R is the average grain radius at stagnation, r the particle radius, and f is the volume fraction in 3D, or area fraction in 2D, of the dispersed particles [Nis 97]. This equation can be derived by considering a force balance in the stagnant grain structure [Hun 92]: The driving pressure P_d for grain boundary movement due to capillarity is balanced by the restraining pressure P_p of the particles. P_p physically originates from the decrease in the free energy when a grain boundary intersects a precipitate and thereby eliminates the grain boundary area replaced by the particle, and reduces the energy of the system by the corresponding grain boundary energy. P_p can be written as

$$P_p = \alpha_1 \frac{f^n}{r}, \tag{F.1.2}$$

where α_1 is a constant. P_d can be expressed as $P_d = 2\gamma\kappa$, where γ is the grain boundary energy, and κ is the effective grain boundary curvature.

Of special practical interest here is the influence of particles on grain growth in thin films. Srolovitz et.al used a Monte Carlo simulation for modeling grain growth, and empirically obtained a relationship between the pinned average grain size and area fraction of particles in a 2D system which matches equation (F.1.1) with n = 0.5 [Sro 84]. Doherty later presented an analysis which rationalizes this result [Doh 87]. In combination with equation (F.1.1) we obtain

$$R \propto \frac{1}{\sqrt{N}} \tag{F.1.3}$$

Our simulations in which the effects of grain boundary grooves are not accounted for, give n = 0.46, as illustrated in figure F.1.4 (b). Furthermore, in our simulation R turns out to be independent of the particle radius, in accordance with equation (F.1.3).

According to equation (F.1.1), the average grain radius at stagnation diverges for infinitely small particle fractions. The extensions reported in the literature so far do not take the effects of additional mechanisms that impede grain boundary motion, such as solute drag or grain boundary grooving, into account. In order to modify equation (F.1.1) for application to thin films with a small area fraction of particles, we have to take the pressure due to solute drag or grain boundary grooving, P_{stag}, into account. The force balance when both solute drag or grain boundary grooving and particle-pinning are acting on grain boundaries becomes

$$P_d = P_p + P_{stag}\,. \qquad \text{(F.1.3)}$$

A first order approximation for the average grain radius at stagnation is therefore given by

$$R = \frac{1}{\alpha f^{n}/r + \beta}\,. \qquad \text{(F.1.4)}$$

Here we have used the fact that κ is proportional to 1/R. α and β are appropriate constants, with β proportional to κ_{crit} in the case of grain boundary grooving. The results obtained from simulation of the combined effects of grooves and particles are shown to be consistent with this relationship in figure F.1.7 (b).

F.1.5 Summary and Conclusion

We have modified a 2D front-tracking simulation of grain growth in thin films to account for the effects of second phase particles on grain structure evolution. We have also investigated the effects of particle pinning in conjunction with the effects of other mechanisms that impede grain boundary motion, such as grain-boundary-grooving-induced pinning, which are known to be important in thin films. The average grain sizes in the stagnant structures, R, are found to be a strong function of the number of particles per unit area, N, for all cases. In the case of particle-induced stagnation without grooves, we obtain a dependence of R on N which is similar to that found using simulations based on the Potts model, and which has been explained by Doherty et al [Doh 87]. When the effects of other grain-growth-impeding mechanisms are added, this relationship changes in a way which can be accounted for in a simple first order extension of the explanation of the effects of particles alone.

We have also found, that even without additional pinning mechanisms, the effects of particle pinning observed in our simulation are not the same as observed using the Potts model, in that the evolution toward stagnation is continuous, and in that we find a significantly different stagnant grain size distribution. The grain size distribution observed when only grooving leads to stagnation is distinctly different from that observed when only particles lead to stagnation, and these both are very different from the grain size distributions that result from stagnation caused by both grain boundary grooving and particle pinning. Interestingly, the latter is similar to the steady-state grain size distribution for 2D normal grain growth. Qualitative explanations for these observations, based on considerations of the effects of these pinning forces on grain topologies, are discussed.

Our results show that the effects of particle-pinning on the stagnant-grain-structure characteristics of thin films, in which the effects of grain boundary grooving are known to important, will be very different from those predicted based on simple 2D pinning models and simulations which do not account for the effects of grooves. Details of the statistical characteristics of grain structures can profoundly affect the performance and reliabilities of thin films used in applications. Accurate simulations of the effects of particles on grain growth in thin films can serve as important tools for investigating microstructure optimization for engineering applications.

F.2 The Effect of Particle-Pinning on Grain Size Distributions in 2D Simulations of Grain Growth

F.2.1 Introduction

Computer simulations are well suited to modeling of the complex interrelated phenomena involved in grain growth, and 2D simulations of grain growth have been shown to give generally good agreement with experiments on soap froths evolving between glass sheets [Sta 93]. 2D simulations have also been modified for simulation of grain growth in polycrystalline thin films [Fro 90, Fro 92, Car 96], whose properties are known to be strongly affected by their grain structures [Nix 89, Tho 95, Tho 93a].

In thin film systems, dispersed inert second-phase particles retard grain growth by pinning grain boundaries, and can either lead to undesired properties, or can be used to produce desired structures. An example of the former is the effect of Al_2Cu precipitates on suppressing post-patterning grain structure evolution in IC interconnects [Tho 93a]. Examples of the use of precipitates to engineer desirable microstructures include the use of Al-Cu-Cr precipitates in Al films [Gan 72, Gan 73, Lon 91] and the use of Yittria precipitates in Cu films to cause precipitate-induced abnormal grain growth [Wei 98].

With the goal of eventually developing simulations which can be used for quantitatively predicting the statistical characteristics of polycrystalline thin films with second phase particles, we have begun by modifying a 2 dimensional front-tracking simulation of grain growth [Fro 88]. In the current paper, we report the extension of our 2D grain growth simulation technique to treat the effects of grain boundary pinning by inert second phase particles, and report comparisons with experimental results on froths and results obtained by others using simulations based on the Potts model. Further extension of our simulation approach to treat particle-pinning in thin films will be reported elsewhere [Rie 99].

The effects of particles on the average grain size in 3D stagnant grain structures has been treated in a number of theoretical studies, starting with the suggestion by Zener [Zen 49] that

$$\frac{d}{r} \propto \frac{1}{f}, \tag{F.2.1}$$

where d is the average grain diameter at stagnation, r the particle radius, and f the volume fraction of particles. If the intersection of grain boundaries with pinning particles is non-random, this equation does not apply [Doh 87]. For grain growth in two dimensions, Doherty et. al. [Doh 87] derived the relation

$$\frac{d}{r} \propto \frac{1}{\sqrt{f}}. \tag{F.2.2}$$

It should be noted that in 2 dimensions, f α r^2N, where N is the number of particles per unit area, so that

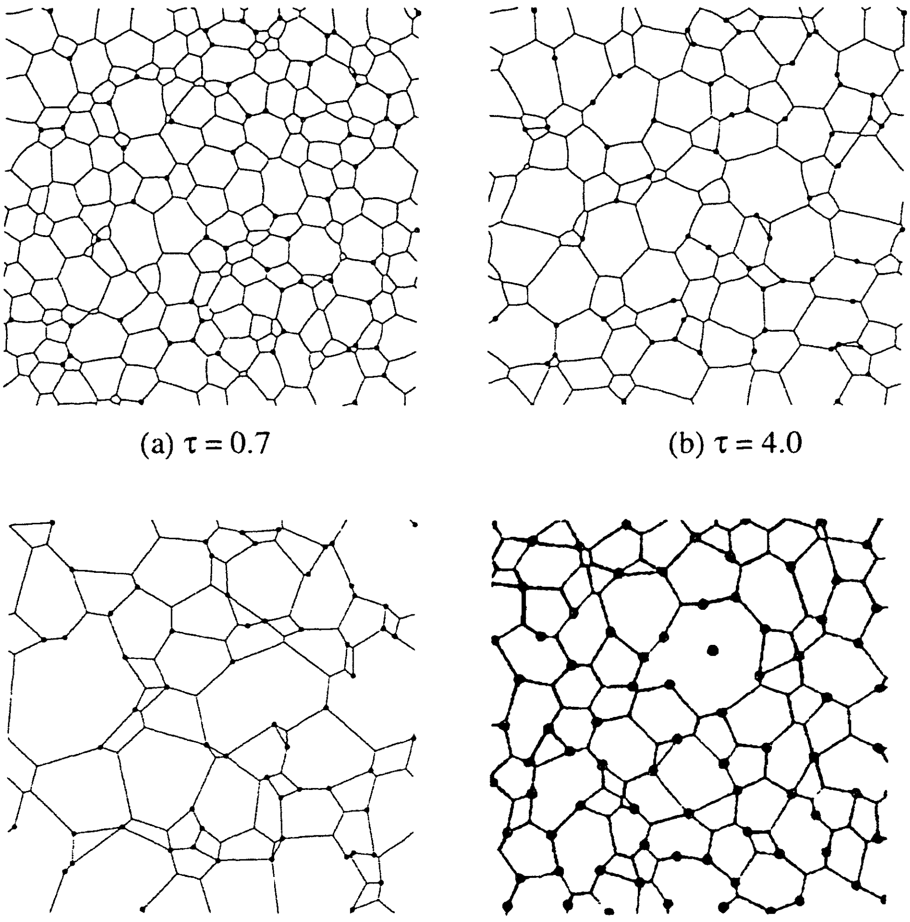
$$d \propto \frac{1}{\sqrt{N}}, \tag{F.2.3}$$

where d is now independent of r. The dependence of on f indicated by equation 2 was also found when particle pinning was treated using simulations of 2D grain growth based on the Potts model [And 84, Sro 84].

F.2.2 Simulation Techniqu es

Our grain growth simulation technique is described in reference [Fro 88] and its algorithm is illustrated in figure F.1.1 (a). Boundary points are moved with a velocity v given by $\mu\kappa$, where μ is the grain boundary mobility and κ is the local boundary curvature. For the simulations described here, it has been assumed that all the boundaries are characterized by the same mobility and energy. The latter leads to a force balance when boundaries meet at triple junctions with 120 degree angles. In agreement with experiments on soap froths [Sta 93], evolution in this basic simulation leads to a steady state in which the average grain area scales with t, and in which the shape of the normalized-grain-size distribution is time-invariant [Fro 88, Fay 99].

When particles are included in the simulation, intergranular particles effectively pin grain boundaries, and moving grain boundaries are captured by intragranular grain boundaries. As illustrated in figure F.1.1 (b), when boundaries make contact with particle perimeters, we create perimeter points which are positioned in each time step so as to enforce a 90 degree contact angle, corresponding to the physical case in which the energies of all particle-grain interfaces are the same. The simulation has also been modified to allow for trapping of moving grain boundaries by intergranular particles, for grain boundary triple junction trapping, and for particle-perimeter switching events in



(a) $\tau = 0.7$

(b) $\tau = 4.0$

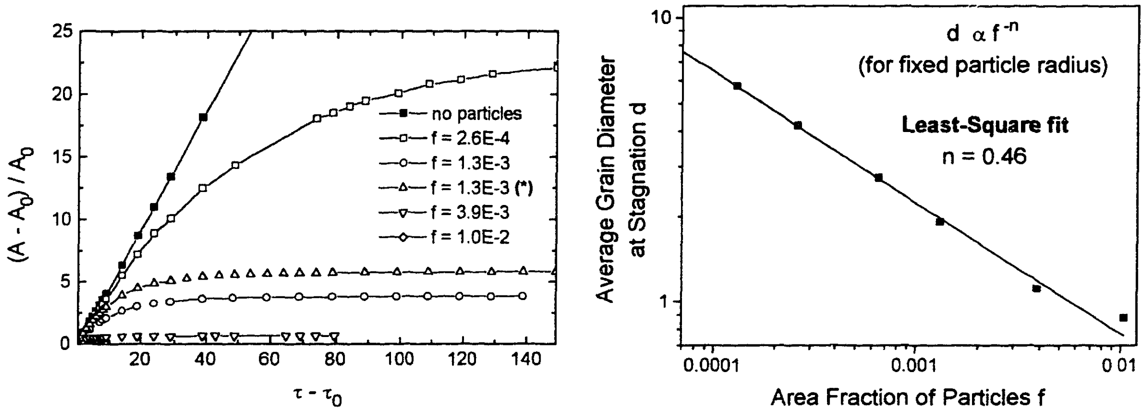
(c) $\tau = 50.0$ at stagnation

(d) Experiments on soap froths

Figure F.2.1: (a)-(c) Grain structure evolution in the presence of randomly distributed particles on grain boundary triple junctions. At $\tau = 50.0$, the grain boundaries are straight and no further evolution takes place. For comparison in part (d), a soap froth structure with pinning centers [Her 97].

which two boundary points on particle perimeters meet to generate a new grain boundary triple junction [Rie 99].

Periodic boundary conditions were enforced, and for simulations for which grain structures were statistically characterized, the starting structure had 17,000 grains with an initial average grain diameter of 1.74 (where $\bar{d} \equiv \sqrt{4\bar{A}/\pi}$). Starting structures were generated by first creating a Johnson -Mehl structure [Fro 87] and then simulating grain growth to normalized time $\tau = 1.0$ ($\tau = \mu t \bar{A}_0$ and $\bar{A}_0$ is the initial average grain area).



(a)

(b)

Figure F.2.2: (a) Evolution of the normalized average grain area. The particles are populated on grain boundary triple junctions initially except for the graph marked by (*), where particles are distributed randomly at the beginning. (b) Average grain size at grain growth stagnation. The number of grains present at stagnation ranges from 1,000 to 10,000.

which is within the scaling state regime [Fro 88]. Particles were then created on randomly selected grain boundary triple junctions of this steady-state structure at $\tau = 1.0$. For comparison, simulations based on a completely random particle distributions were run, and were found to lead to similar results. f was varied by varying the number of particles per unit area for constant particle radius, $r = 0.07$, as well as varying the r from 0.03 to 0.40 for a constant number of particles per unit area $N = 0.086$.

F.2.3 Simulation Results

Figures F.2.1 (a) through (c) show an example of the evolution of a 2D grain structure with particle pinning. Once stagnation occurs (figure F.2.1 (c)), all grain boundaries are straight lines. The time evolution of the normalized average grain area for a variety of area fractions of particles is shown in figure F.2.2 (a). For small area fractions of particles, we see a gradual transition from the regime in which grain growth

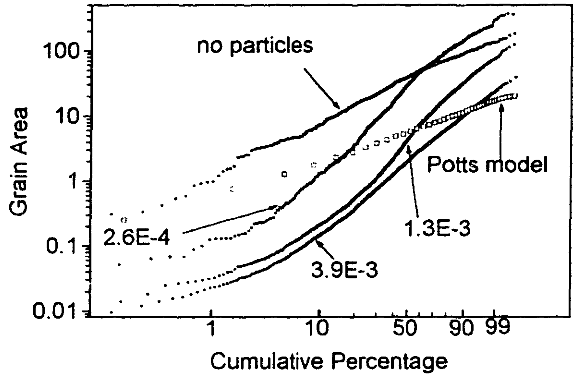
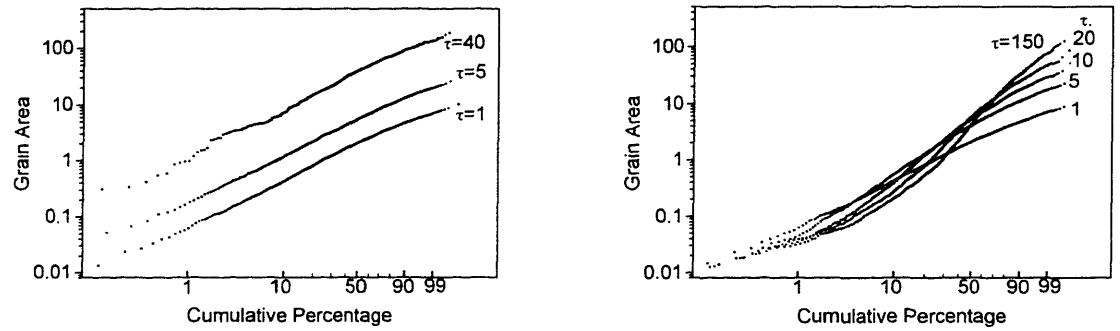


Figure F.2.3: Grain size distributions at grain growth stagnation on a cumulative plot for different particle area fractions f. On this plot, a Weibull distribution shows up as a straight line. Also shown is the steady state structure in case of no particles. The open symbols show the grain size distribution of the stagnated structure as of reference [Sro 84].

is not affected by particles, into the regime in which particle pinning is important and finally results in the complete stagnation of grain growth. If the average grain diameter at the beginning is comparable to the average distance between particles, grain growth is strongly affected by the particles from the very beginning of the simulation. As shown in figure F.2.2 (b), the average grain diameter at stagnation is proportional to f $^{-0.46}$ for constant r. For constant number of particles per unit area the average grain diameter is constant, independent of the particle size r (given that r is much smaller then the average grain size). The grain boundary arrangements near particles are similar to Steiner trees [Ber 89, Ste 91].

Figure F.2.3 shows the grain size distributions at stagnation for various area fractions of particles on a cumulative plot, as well as the grain size distribution for a structure without particles at $\tau = 40$, after a steady-state distribution has developed. We find that the cumulative distribution curves for simulations which include the effects of particles are more steeply sloped (indicating a larger variation in grain sizes) than those for simulations without particles. Figure F.2.4 shows the time evolution of the grain size



(a) (b)

Figure F.2.4: Evolution of grain size distributions (a) without particles and (b) with an area fraction of particles $f = 1.3 \times 10^{-3}$ (for a fixed radius) on a cumulative plot. On these plots, a Weibull distribution function would appear as a straight line [Fay 99].

distributions for simulations without (figure F.2.4 (a)) and with particles (figure F.2.4 (b)). Figure F.2.4 (a) illustrates self-similar evolution, while figure F.2.4 (b) does not. In the case in which particle pinning is included, the rate of change of the average grain size decreases to zero while the deviation in the grain size *increases* with time, until stagnation is complete. Figure F.2.5 shows the fraction of grains with numbers of neighbors (or sides) m. When particle pinning is included, the number of grains with m = 3 or 4, as well as the number with m ≥ 9, increases with time, whereas the fraction of grains with m = 5, 6, or 7 decreases with time. The fraction of grains with m = 8 stays roughly constant.

F.2.4 Discussion

We first discuss the origin of the shape of the grain size distribution after particle-pinning-induced stagnation. Without particles, an m-sided grain obeys the Mullins-von Neumann relationship [Mul 56, Neu 52], i.e. the change of the grain area with time is proportional to (m-6), so that grains with small m disappear quickly. However, particles

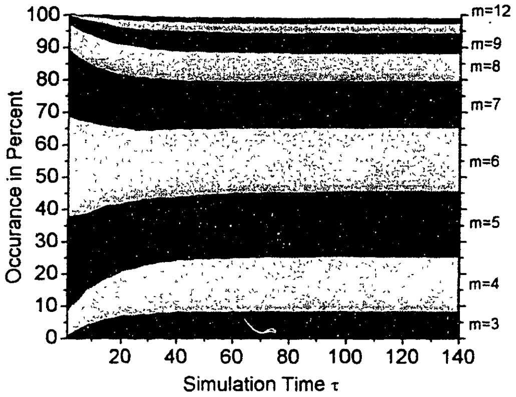


Figure F.2.5: Evolution of the fraction of number of nearest neighbors of grains. m is the number of nearest neighbors. For m ≥ 13, the fraction of grains approaches zero.

stabilize grains with $m < 6$ that would have disappeared without particles. The average number of nearest neighbors in the stagnant structure is still close to 6, because 4- and higher-fold vertices are still rare. Therefore, the number of grains with $m > 6$ increases, as well. The average grain area increases with increasing m, and 3- and 4-sided grains are especially small. Therefore, the particle-induced stagnant structure has a grain size distribution with a large fraction of small grains and a large deviation from the mean.

Simulations of grain growth in the presence of particles based on the Potts model indicate an evolution of the grain size distribution [Sro 84] that differs from our results. We observe significant evolution in the distribution, while, once normalized by the average grain area, as seen in figure F.2.4, the Potts-model results are much more similar to the steady-state distribution. Our distribution after particle pinning peaks around the average grain size at first, but the peak shifts below the average grain size with time. Also, the deviation from the mean increases with time. In the Potts model, the distribution peaks close to the average grain size during the evolution *as well as* at stagnation. Furthermore, the effect of particles on the grain size distribution as predicted by the Potts model is much less pronounced, see figure 12 in reference [Sro 84]. All these differences arise from a larger fraction of grains with small-angle grain corners and with only a few sides in our grain structures, as illustrated, for example, by the three-sided grain at the bottom of figure F.2.1 (c). Grain structures resulting from simulations based on the Potts model do not show this type of grain if the particles are represented by only a few Potts elements, because as an artifact of the simulation due to the finite Potts element size, the grain boundary would pinch off from the particle. Therefore, in the Potts model, small grains are eliminated leading to a smaller deviation from the mean grain size, and in turn leading to a grain size distribution more similar to the steady state distribution.

The dependence of the average grain diameter at stagnation, d, with area fraction f, for our simulations with a fixed particle radius r is $d \propto f^{-0.46}$, in agreement with the results from simulations based on the Potts model ($d/r \propto f^{-0.5}$) [Sro 84] and experiments on soap froths ($d/r \propto f^{-0.5}$) [Her 97, Kri 92].

Also in agreement with soap froth experiments, the grain boundaries at stagnation have zero curvatures, as can be seen by comparison of the simulated structure in figure F.2.1 (c) and the image of a stagnant froth structure in figure F.2.1 (d). In both grain structures, small grain boundary angles occur at particles, and especially the 3- and 4-sided grains have similar shapes. In both cases, a large fraction of smaller grains is pinned by particles. Concerning the time dependence of the average grain area, our simulation predicts a gradual transition from the stage where particles do not affect grain growth, through the regime in which effects become dominant and eventually lead to stagnation, which is qualitatively different from what is seen in Potts model simulations [Sro 84], but similar to what is seen experimentally [Her 97, Kri 92].

F.2.5 Conclusion

Our front-tracking simulation of grain growth in the presence of pinning particle reproduces many of the observations made in soap froths. Our simulation agrees with the 2D Zener relation given in equation (F.2.2) and (F.2.3). Our simulations result in a significantly different grain size distribution for the stagnant structure than those reported for similar simulations based on the Potts model. The difference between the stagnant and steady state structures that we observe can be understood as resulting from the retention of grains with small vertex angles at particles. We speculate that the differences between our simulations (and froth experiments) from those based on the Potts model arise from finite gridding effects which result in weaker particle-pinning of grain boundaries in that boundary pinch-off occurs.

Soap froths provide good experimental systems for comparison with 2D simulations of grain growth and particle pinning. However, there is great interest in using simulations to understand grain structure evolution in polycrystalline thin films. In thin film experimental systems, unless particles extend through the thickness of the film, boundaries will be able to more readily escape particles than in the simulations reported here. In thin films it is also known that grain boundary grooves lead to pinning forces that can have dominant effects on final grain structures. Therefore, comparison of simulations of particle-pinning effects on grain structure evolution in thin films also requires consideration of the pinning effects of grain boundary grooves [Rie 99].

F.3 Modeling of grain structure evolution and its impact on the reliability of Al(Cu) thin film interconnects

F.3.1 Introduction

The rate of electromigration-induced failure in Cu-doped Al interconnects is governed by their grain structure, which is governed by post-patterning grain structure evolution, which, in turn, is governed by the presence and characteristics of precipitates. The characteristics of precipitates are controlled by alloy composition and thermal history [Kno 95]. When interconnects are annealed, they evolve toward the bamboo structure, in which individual grains span the full width of the interconnect. This results in a dramatic improvements in reliability. However, recent TEM experiments [Kno 95] have shown that *precipitates* gate grain structure evolution during post-pattern annealing. Therefore, to fully understand the effects of post-pattern annealing on the reliability of alloy interconnects, it is necessary to account for the effects of precipitates.

Our goal is to predict the effect of interconnect processing on the statistics of electromigration-induced failure of Al(Cu) interconnects. This ability will allow optimization of manufacturing processes to maximize interconnect reliability. Our approach is to modify our grain growth simulator to treat the effects of precipitates on post-patterning annealing, then extract the spatial dependence of the diffusivity from the grain growth results, and finally use the electromigration simulator MIT/EmSim [Ems] to predict lifetime information. This gives us a suite of tools to investigate the relationship between processing and interconnect reliability.

F.3.2 Simulation Tools

F.3.2.1 Grain Growth Simulation

Our grain growth simulator is based on a curvature-driven front tracking model [Fro 90]. The velocity of each grain boundary segment is proportional to the local curvature. At grain boundary triple junctions, a local force balance is enforced, so that grain boundaries meet at 120°. The simulation is based on a 2D model where 3D effects such as those due to grain boundary grooving are accounted for a posteriori [Fro 90]. Specifically, if the grain boundary curvature drops below a critical curvature, the grain boundary becomes stagnant locally [Mul 58]. A continuous film can be patterned into stripes and the effects of post-pattern annealing is subsequently simulated [Wal 91, Wal 92]. A force balance at the interconnect side wall forces the grain boundaries to meet the line edges at right angles, or when grain boundary grooving is allowed on the side walls, intersection angles slightly different from 90° [Wal 92].

Our in-situ TEM studies of annealing of Al(1wt% Cu) alloys suggested the need for significant modifications of the grain growth simulator to account for the effects of precipitates on post-pattern annealing [Kno 95]. Intergranular precipitates effectively pin grain boundaries or grain boundaries are captured and pinned by intragranular precipitates. The spatial precipitate distribution and the size distribution of precipitates are a function of the annealing history. Precipitates are located preferentially at grain boundary triple junctions and grain boundaries, whereas intragranular precipitates are more rare. In our simulations, we assumed that precipitates nucleated at grain boundary triple junctions. We expect similar trends if precipitates were located at arbitrary grain boundary positions.

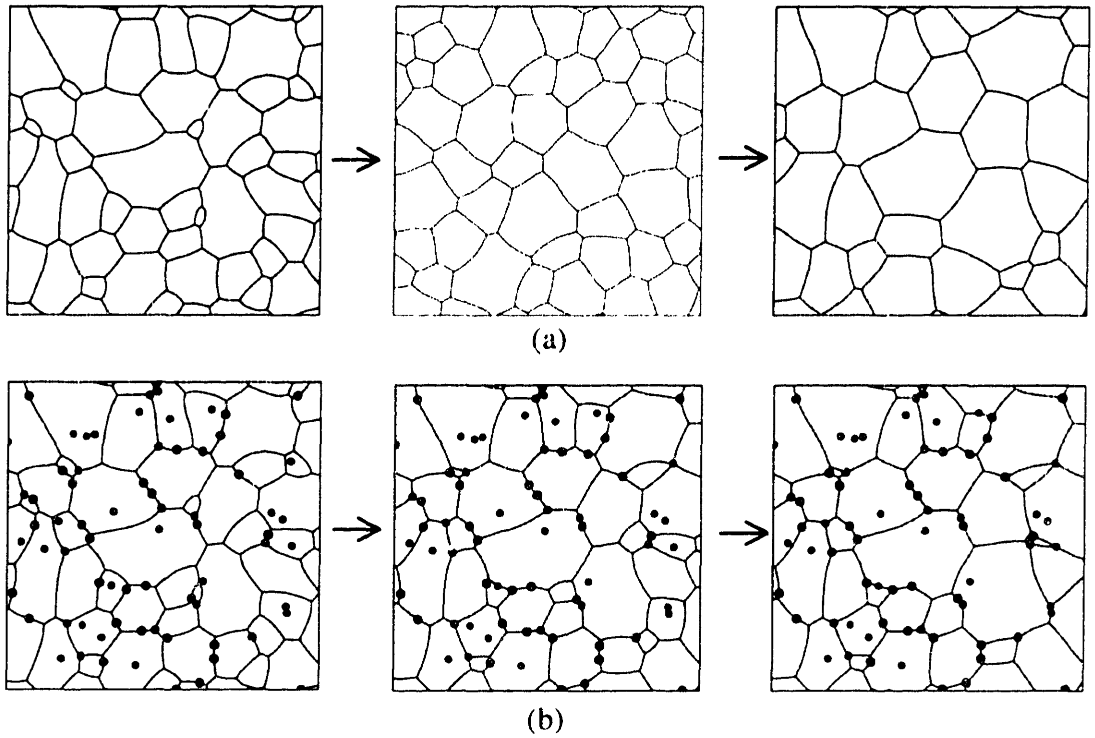


Figure F.3.1: Effect of particles on grain growth in continuous films. In (a), the structure reaches grain growth stagnation due to surface grooving only. In (b), precipitates inhibit grain growth and lead to stagnation at much smaller grains.

We modified the grain growth simulator to account for the effect of particle pinning of grain boundaries. The most important modifications are illustrated in figure F.1.2. For isotropic surface energies, pinch-off of grain boundaries from precipitates does not occur and is therefor not considered. The particles have a profound impact on the grain structure evolution in thin films, as is illustrated in figure F.3.1. Whereas a continuous film comes to growth-stagnation due only to *grain boundary grooving* (figure F.3.1 (a)) [Wal 91], the film containing particles stagnates with a much smaller average grain diameter d_{50} due to the effect of particle pinning (figure F.3.1 (b)).

In this paper, we will focus on the microstructural evolution in *interconnects*. Patterning the continuous film shown in figure F.3.1 and simulating a post-patterning annealing results in different microstructures depending on whether or not precipitates

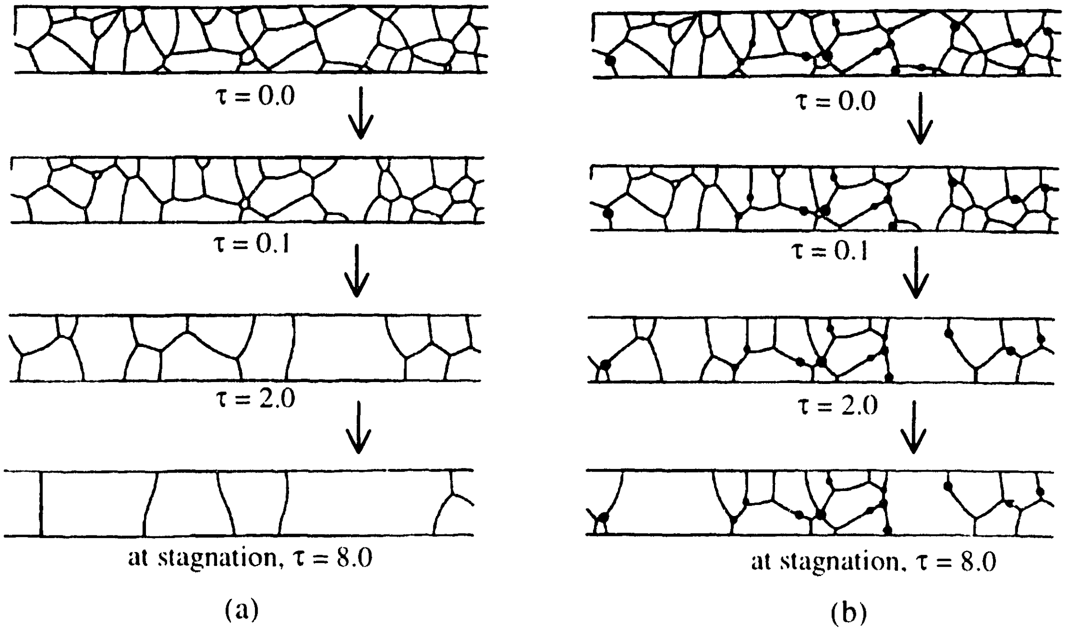


Figure F.3. 2: Effect of particles on grain growth in interconnects. Particles inhibit grain growth, and the line fails to reach the fully bamboo structure. Polygranular regions are pinned on each side by precipitates.

are present. Figure F.3.2 shows the evolution of an interconnect with and without particles. Particles have a significant impact on the resulting microstructure: At stagnation in figure F.3.1 (b), polygranular cluster regions are pinned on both sides by precipitates. These precipitates prevent the line from reaching the far more reliable full-bamboo structure.

F.3.2.2 Electromigration Simulation

We simulated the effect of electromigration on interconnects using our electromigration simulator MIT/EmSim [Ems, Kno 97, Kno 95a]. MIT/EmSim is a 1D electromigration simulation based on the Korhonen model [Kor 93]. Due to the electron wind force, atomic redistribution in the interconnect takes place. At locations of flux divergence, a pile-up or deficiency of atoms occurs, and results in a compressive or

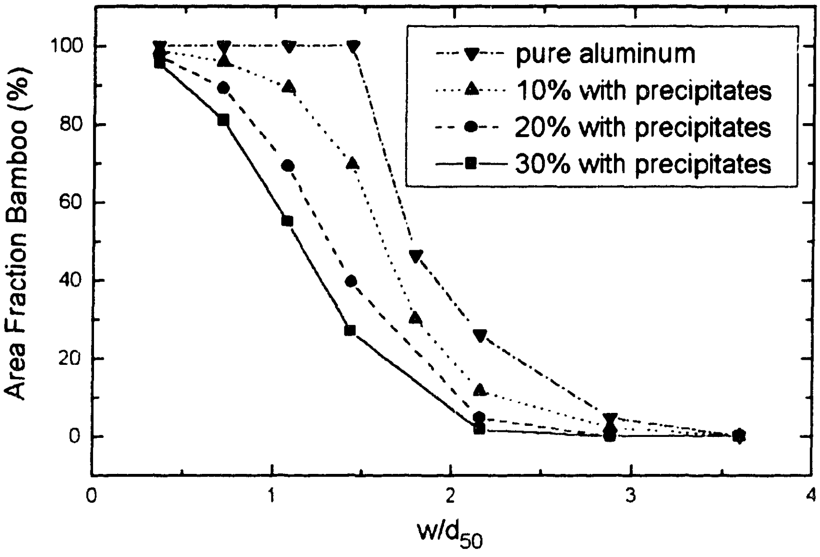


Figure F.3. 3: For different fractions of grain boundary triple junctions covered with precipitates, the area-fraction-bamboo versus the ratio or line width w to the initial median grain diameter d_{50} after post-patterning annealing. With precipitates, the interconnect fails to become fully bamboo even if the line width is smaller than the initial median grain size.

tensile hydrostatic stress, respectively. If the increasing stresses exceed certain limits dictated, for example, by the surrounding material or the metal itself, the interconnect fails due to cracks in the passivation or fatal voiding in the metal [Kno 97]. We predict the microstructure variation in interconnects using the grain growth simulator, and use these structures to set diffusivity variations along interconnects. Polygranular clusters have the higher diffusivity associated with grain boundary diffusion, whereas regions with bamboo microstructure have lower transgranular diffusivities [Gan 75]. When the spatial dependence of the diffusivity is fed into MIT/EmSim the atomic fluxes and stress evolution in the interconnect is calculated as a function of current density and temperature [Kno 97a].

In this paper, we simulated the effects of precipitates and thermal history on the reliability of 200 μm long lines. Presumably, for longer lines the results would change somewhat but they will show the same trends. Critical tensile and compressive stresses of 500 MPa were assumed to lead to failure, and the simulations discussed here were carried out for a current density of 2×10^6 A/cm^2 and a test temperature of 200°C.

F.3.3 Results

F.3.3.1 Grain Growth Simulatio n

Precipitates retard grain growth in interconnects and suppress evolution toward the more reliable bamboo structure. Figure F.3.3 shows the area-fraction-bamboo in interconnects versus the ratio of the line width, w, to the median initial grain diameter, d_{50}, after a post-pattern anneal. As w/d_{50} is decreased the fraction bamboo after annealing increases, with or without precipitates present. For a given line width, precipitate-induced suppression of the evolution towards the bamboo structure increases with an increasing number of precipitates, or an increasing precipitate volume fraction for a given average precipitate size. Interestingly, interconnects with precipitates do not reach the fully bamboo structure even if the line width is smaller than the initial median grain size.

In our grain growth simulations, precipitates capture and effectively pin grain boundaries. Our simulation is consistent with experimental results which show that precipitates inhibit grain growth and the evolution towards the bamboo structure [Kno 95]. Even a small volume fraction of precipitates prevents sufficiently wide interconnects from becoming fully bamboo.

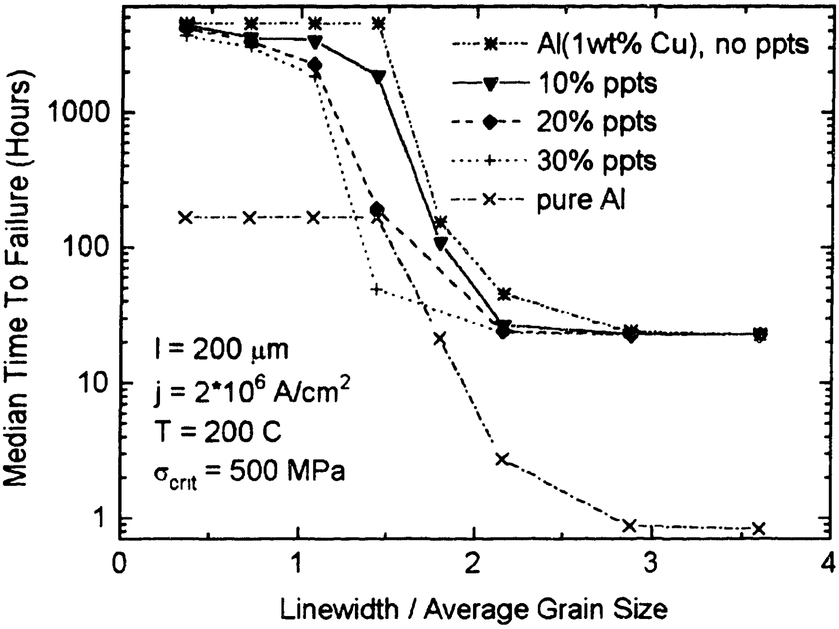


Figure F.3. 4: Median times to failure (MTTF) as a function of the ratio of the linewidth to the initial median grain size for pure Al interconnects, interconnects with copper in solid solution, and interconnects with precipitates. For certain line widths, for a high number of precipitates, or a high precipitate volume fraction for precipitates of a given average size, the reliability is worse for alloyed lines than for similar pure aluminum lines.

F.3.3.2 Electromigration Simulation

Adding a few weight percent copper to aluminum reduces the electromigration rate [Ame 70], leading to a significant increase in the reliability of Al interconnects. However, it is also now well established that post-patterning annealing leads to significant lifetime improvements, especially in narrow lines which develop predominantly bamboo structures [Kno 97, Tho 93a, Kan 97]. Our simulations predict that a high number of precipitates will suppress this post-patterning microstructural evolution to bamboo structures.

To investigate these competing effects of Cu alloy additions, we performed simulations of microstructural evolution during post-pattern annealing of populations of aluminum lines. We subsequently performed electromigration simulations using the results. Each population contained ten lines with the same geometrical dimensions but statistically varying microstructures. Calculations were done for different populations of lines with different line widths. We fit the predicted lifetimes to lognormal distribution functions and determined the median time to failure (MTTF). Figure F.3.4 shows the MTTF versus the line-width-to-initial-median-grain-diameter ratio for pure and alloyed aluminum lines. In all cases, narrower lines are nearly fully bamboo and therefore have a higher MTTF. Wider lines are polygranular and fail much faster due to the higher polygranular diffusivity.

When copper is added, but assumed to be in solution, the expected improvement in reliability is observed, and the MTTF curve essentially shifts upward. However, if the copper addition leads to precipitate formation, it inhibits the microstructural evolution towards the bamboo structure, so that the lifetime for narrower lines is reduced. The results shown in figure F.3.4 show that for high precipitate numbers, the reliability for some alloyed lines can actually be worse than that of similar pure aluminum lines.

F.3.4 Conclusions

Linked precipitate and grain structure evolution and electromigration simulation allow the prediction of the effects of interconnect processing on the statistics of electromigration-induced failure. Our simulations predict that while copper-alloying improves the reliability of interconnects if the copper is in solid solution, this improvement can be reduced or even eliminated if copper addition leads to Al_2Cu precipitate formation, so that the precipitates inhibit the microstructural evolution of interconnects during post-pattern annealing. Whether or not this is the case depends on the copper composition and on the thermal history.

To fully simulate the effects of Cu and thermal history on interconnect reliability, it will be necessary to develop simulations of precipitation, precipitate dissolution, and precipitate coarsening to couple with our grain growth simulation. Once this is done, these tools will provide a means of assessing the impact of process modifications on circuit reliability, and for developing processes and materials which lead to improved reliability.

Appendix G Grain Growth in Patterned Aluminum Thin Film

G.1 Background

Modern submicron interconnects have dimensions comparable to, or less than, those of their microstructure, e.g., their grain sizes, so that their reliability is very sensitive to process-induced microstructural variations. Reliability assessments based on conventional methodologies alone are flawed because they do no account for the effects of complex interconnect geometries on grain structures and reliability. To accurately assess the reliability of circuits, the complex shapes, geometries, and interconnectedness of actual interconnect structures must be accounted for.

We focus on the microstructural evolution in continuous and patterned thin films due to grain growth, by which the initial microstructure can evolve after deposition. A tool for simulating grain structure evolution in interconnect trees with complex geometries has been developed by Fayad et al. [Fay 00]. In this work, we experimentally characterize the grain structure evolution in complex geometries by carrying out direct comparisons of in-situ TEM annealing experiments and simulations. To do this, we developed the capacity for in-situ TEM studies of grain structure evolution in films patterned into straight lines and other shapes. The results of the TEM experiments suggest modifications of the grain growth simulator developed by Fayad et al. [Fay 00] concerning the interaction of pattern edges and grain boundaries. The results obtained with the modified simulation are in agreement with experiment.

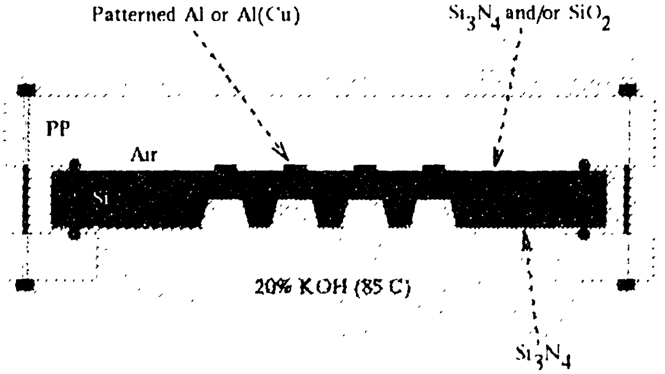


Figure G.1: Schematic of the wafer chuck protecting the processed front-side of the Si wafer during the micromachining step in a KOH bath. The anisotropic Si etchant KOH:H_20 in the concentration 1:5 at 85^{0}C removes the Si from underneath and exposes the TEM transparent Si_3N_4/SiO_2 bilayer membranes.

In the next sections, we describe the results of the TEM experiments, explain the simulation technique, and summarize the results of simulations of the grain structure evolution in patterns. Finally, experimental results and simulations are compared.

G.2 In-Situ TEM Experiments of Post-Patterning Annealing

We developed a technique that facilitates the processing of a large number of TEM samples of films patterned in a variety of shapes. The technique involves the use of an apparatus that allows immersion of a fully processed wafer into a KOH bath for etching of electron-transparent membranes at locations defined by a patterned Si_3N_4 film on the backside of the wafer, as sketched in figure G.1. Using this technique, we fabricated TEM samples with 7500 Å-thick patterned Al films on top of a Si_3N_4/SiO_2 bi-layer.

We carried out hot stage TEM experiments in which we observed grain growth in aluminum films patterned into "T's" and "squares" at different temperatures. Figure G.2 shows an example of the microstructural evolution of a square-shaped pattern. We find

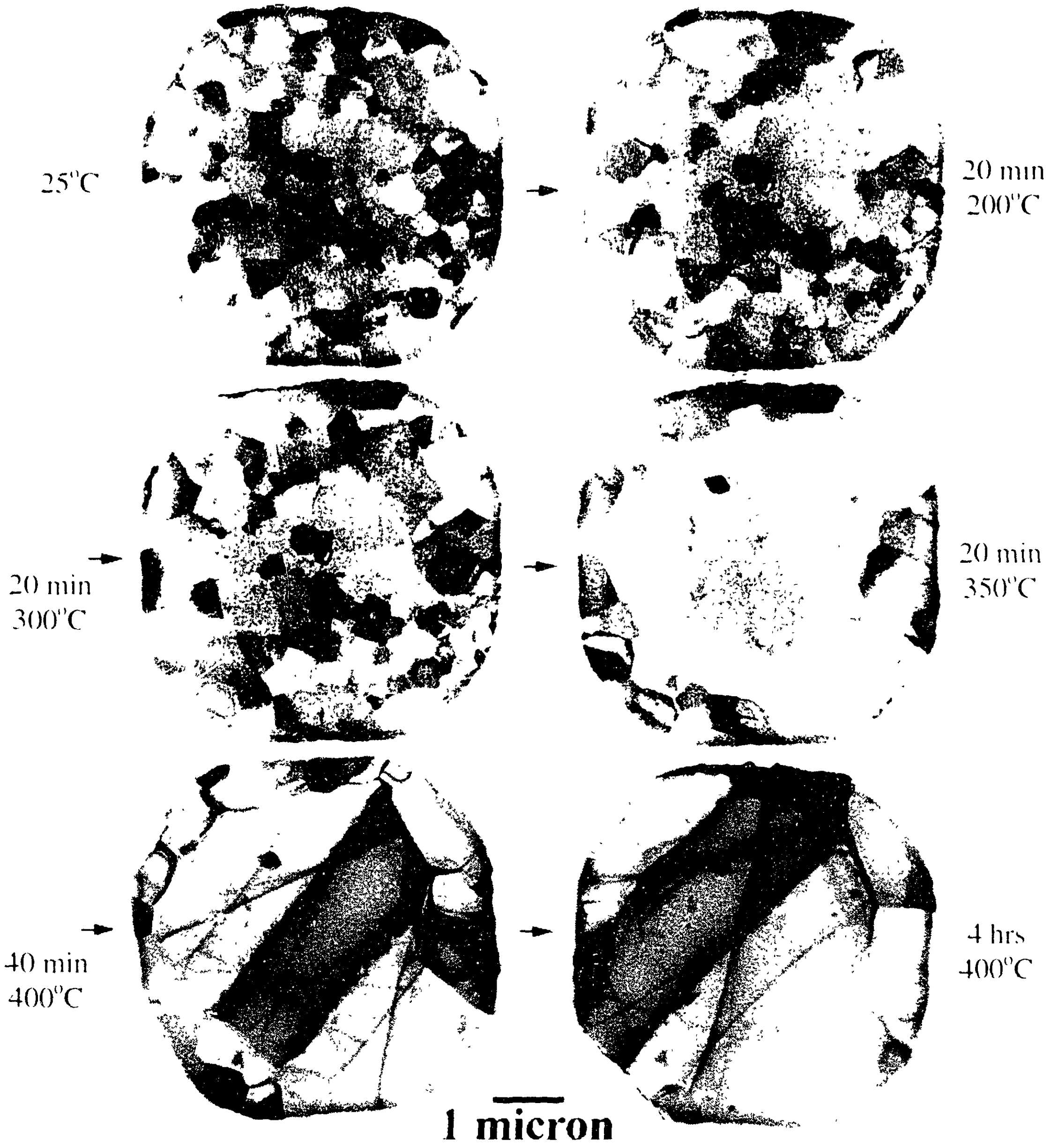


Figure G.2: Time series of TEM micrographs showing the microstructural evolution of a square-shaped Al thin film.

smaller grains at the edge of the structure, whereas the inner grains have a much larger size.

G.3 Simulation of Grain Growth in Patterned Films

To simulate grain growth, we have developed a curvature-driven front tracking model [Fro 88, Fro 90, Fro 92, Wal 92, Car 96]. The velocity, v, of each grain boundary segment is proportional to the local curvature, κ, through a constant of proportionality, μ, which is defined as the grain boundary mobility. At grain boundary triple junctions, a local force balance is enforced such that grain boundaries meet at 120^o. We have shown earlier that this 2D model can be modified to account for some of the 3D effects known to be important in thin films, such as those due to grain boundary grooving [Fro 90]. Frost et al. showed that the effect of grain boundary grooving, as described by Mullins [Mul 56], can be accounted for by assuming that a grain boundary becomes locally stagnant due to groove formation when its local curvature falls below a critical level corresponding to a critical in-plane curvature, κ_{crit} [Fro 90], which is inversely proportional to the film thickness. If this stagnation criteria is enforced on a steady-state structure with an average curvature significantly larger than κ_{crit}, and the simulation is continued, microstructure evolution eventually stagnates with a lognormal grain size distribution, with an average in-plane grain size related to κ_{crit} so as to be proportional to the film thickness, as is observed in experiments [Tho 90].

A continuous film can be patterned into arbitrarily complex shapes, and the effects of post-pattern annealing is subsequently simulated [Fay 00]. A force balance at the interconnect side wall forces the grain boundaries to meet the line edges at right angles.

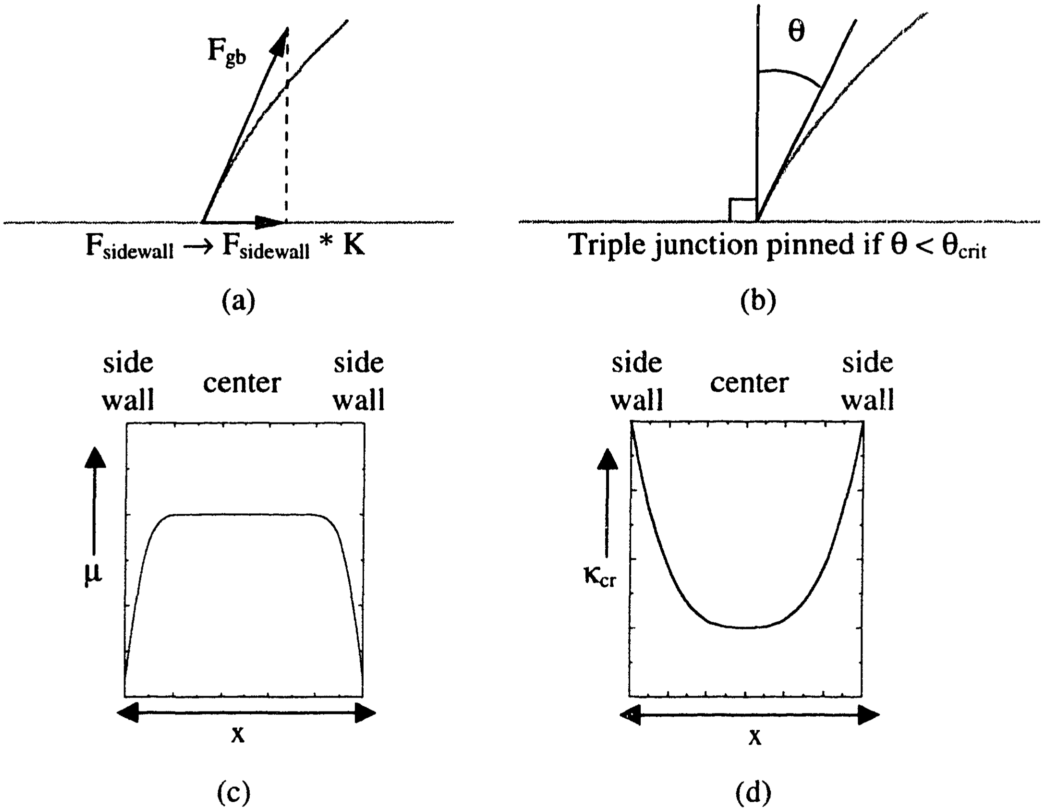

F_{gb}
$F_{sidewall} \rightarrow F_{sidewall} * K$
(a)
θ
Triple junction pinned if $\theta < \theta_{crit}$
(b)
side wall
center
side wall
μ
x
(c)
side wall
center
side wall
κ_{cr}
x
(d)

Figure G.3: The modifications of the grain growth simulator. (a) Due to edge drag, the force on grain boundary triple junctions at side walls is reduced by a factor, K. (b) Due to grain boundary grooving, grain boundaries with an inclination angle, θ, smaller than θ_{crit} are pinned. (c) The grain boundary mobility, μ, varies as a function of location, x. (d) Due to a thickness variation, the critical curvature, κ_{cr}, changes with location, x.

We modified the grain growth simulator to account for the effects of side wall drag and sidewall pinning on grain growth in arbitrarily patterned structures. Prior to the modifications, grain boundary grooving at side walls was only possible for interconnect stripes [Wal 92]. For sidewall drag, the driving force for grain-boundary-triple-point motion at the pattern edge is reduced by a factor, K, as illustrated in figure G.3 (a). This accounts for impeded grain boundary motion at the sidewall due to surface inhomogenities. For sidewall pinning, the grain boundary triple junctions are pinned at the side walls if the grain boundary intersects the pattern edge below a critical angle, as shown in figure G.3 (b). This represents the physical situation of grain boundary grooving applied to the sidewall in addition to the surface of thin films as suggested by Mullins [Mul 56, Mul 58]. In the other two approaches, the effects of the pattern edges are extending into the metal. In the first approach, the grain boundary mobility decreases near the edge, as shown in figure G.3 (c), which can be caused by impurities or etch residuals diffusing from the pattern edge into the metal. In the second approach, it is assumed that the pattern evolves to a "dome" shape during annealing, and that grain boundary grooving occurs earlier at the pattern edges than in the thicker center of the pattern. As shown in figure G.3 (d), the critical curvature for grain growth stagnation changes with position in this case.

G.3.1 Modifications of the Grain Growth Simulator

In order to allow the enforcement of various boundary conditions at pattern edges, a body of code was developed, which is located in the file *edge_bc.c*. In addition, smaller changes at various locations in the grain growth simulator code were necessary. In this section, the new data structures and the program code accounting for edge drag effects, for edge grooving, and for a mobility and a critical curvature varying with location are discussed.

Two boolean flags were added to the data structure **flag_struct** located in *struct.h*, which determine if edge drag or edge grooving is activated:

```
{
[...]
  int edge_drag;     /* edge drag turned on          */
  int edge_bc;       /* edge grooving turned on      */
} flag_struct;
```

If edge drag is activated, the variable **edge_drag** defined in *external.h* determines the factor by which the driving force acting on edge triple points is reduced. If edge grooving is activated, the variables **stagnant_angles** and **moving_angles** determine the critical angles:

```
float edge_drag;
float stagnant_angle, moving_angle;
```

Edge drag effects are activated through use of command 50 in the command input file, interpreted by the following code fragment which is located in the file *run.c*:

```
/* Set edge drag, for example
** 50 0      edge drag off
** 50 0.01   edge drag factor of 1/100
*/
case 50:
  if ( fscanf(f.exec,"%f",&edge_drag) != 1) status = 0;
  else if (edge_drag==-1) flag.edge_drag = 0;
  else {
    flag.edge_drag = 1;
    printf("Set edge drag factor: %.5f\n",edge_drag);
  }
  fgets(text,80,f.exec);
  fprintf(f.commands,"50 %0.5f %s",edge_drag,text);
  break;
```

The forces on grain boundary triple points at pattern edges are calculated in the function *triple_update* located in the file *numerics.c*. The following program code, which has been added to *triple_update,* reduces the force on triple points at edges by the factor **edge_drag**:

```
if ( flag.edge_drag && triple[triplept].type==1 ) {
  xtrip = triple[triplept].x
    + edge_drag * (xtrip - triple[triplept].x);
  ytrip = triple[triplept].y
    + edge_drag * (ytrip - triple[triplept].y);
}
```

Edge grooving effects are activated through use of command 51 in the command input file, interpreted by the following code fragment:

```
/* Set edge boundary conditions for patterned structures,
** for example,
** 51 0        90 degree BC
** 51 1 10     inclination of 10 degrees necessary, otherwise
**             triple point stagnant
** 51 2 10 20  if stopped, inclination of 20 degrees necessary,
**             if moving, inclination of 10 degrees necessary.
**/
case 51:
  if ( fscanf(f.exec,"%d",&flag.edge_bc) != 1) status = 0;
  switch(flag.edge_bc) {
    case 0 :
      fgets(text,80,f.exec);
      fprintf(f.commands,"51 0 %s",text);
      break;
    case 1 :
      if ( fscanf(f.exec,"%f",&stagnant_angle) != 1) status = 0;
      stagnant_angle *= 2.0*PI/360.0;
      moving_angle = stagnant_angle;
      fgets(text,80,f.exec);
      fprintf(f.commands,"51 1 %f %s",stagnant_angle,text);
      break;
    case 2 :
      if ( fscanf(f.exec,"%f %f",&stagnant_angle,&moving_angle) != 2)
      status = 0;
      stagnant_angle *= 2.0*PI/360.0;
      moving_angle   *= 2.0*PI/360.0;
      fgets(text,80,f.exec);
      fprintf(f.commands,"51 2 %f %f %s",stagnant_angle,
      moving_angle,text);
      break;
    default:
      printf("Unknown option.\n");
      status = 0;
      break;
    }
    break;
```

This program code located in the file *edge_bc.c* corrects the position of the triple junctions at the line edge according to the specified boundary conditions:

```
extern float xsize, ysize, xlo, xhi, ylo, yhi;

int edge_bc(float *glo_x, float *glo_y, float *glo_xtrip, float *glo_ytrip,
           int *linear_flag, int triplept, int *flag_corner)
```

```
{
  float  incl, cos_incl, sin_incl;
  /* rotated variables */
  float  theta;
  float  x[6], y[6], xtrip, ytrip, xtrip_old, ytrip_old;
  float  xvect,yvect;
  int    i, j;

  float  Vu, Vu_pin, xmove;

  /* corner not hit so far */
  *flag_corner = -1;

  /* angle edge with x axis */

  /* wrap coordinates */
  for( i=0;i<=2;i+=2 ) {
    if( glo_x[i]<xlo ) glo_x[i] += xsize;
    if( glo_x[i]>xhi ) glo_x[i] -= xsize;
    if( glo_y[i]<ylo ) glo_y[i] += ysize;
    if( glo_y[i]>yhi ) glo_y[i] -= ysize;
  }

  incl = atan2(glo_y[2]-glo_y[0], glo_x[2]-glo_x[0]);

  cos_incl = cos(incl);
  sin_incl = sin(incl);

  /* transform variables: glo_xtrip,glo_ytrip as center, edge as x-axis */
  xtrip=ytrip=0.0;
  for( i=0;i<6; ) {
    /* wrap coordinates */
    if( glo_x[i]<xlo ) glo_x[i] += xsize;
    if( glo_x[i]>xhi ) glo_x[i] -= xsize;
    if( glo_y[i]<ylo ) glo_y[i] += ysize;
    if( glo_y[i]>yhi ) glo_y[i] -= ysize;

    x[i]= cos_incl*(glo_x[i]-glo_xtrip[0])+sin_incl*(glo_y[i]-glo_ytrip[0]);
    y[i]=-sin_incl*(glo_x[i]-glo_xtrip[0])+cos_incl*(glo_y[i]-glo_ytrip[0]);
    switch (i) {
    case 0 :
    case 2 :
      i+=2;break;
    case 4 :
if (linear_flag[2]!=1) i++;   else i+=2;
      break;
    case 5 : i++;
    }
  }
  xtrip_old= cos_incl*(triple[triplept].xold-glo_xtrip[0])
        +sin_incl*(triple[triplept].yold-glo_ytrip[0]);
  ytrip_old=-sin_incl*(triple[triplept].xold-glo_xtrip[0])
        +cos_incl*(triple[triplept].yold-glo_ytrip[0]);

  /* double check order */
  if(!(x[0]<xtrip && xtrip<x[2])) {
        printf("ERROR(edge_bc): wrong order %f %f %f.\n",x[0],xtrip,x[2]);
        printf("                %f, %f, %f\n",y[0],ytrip,y[2]);
        gfxon=1;gfx();gfxon=0;
        WAIT_CHAR;
  }

  /* calculate angle grain boundary and edge */
```

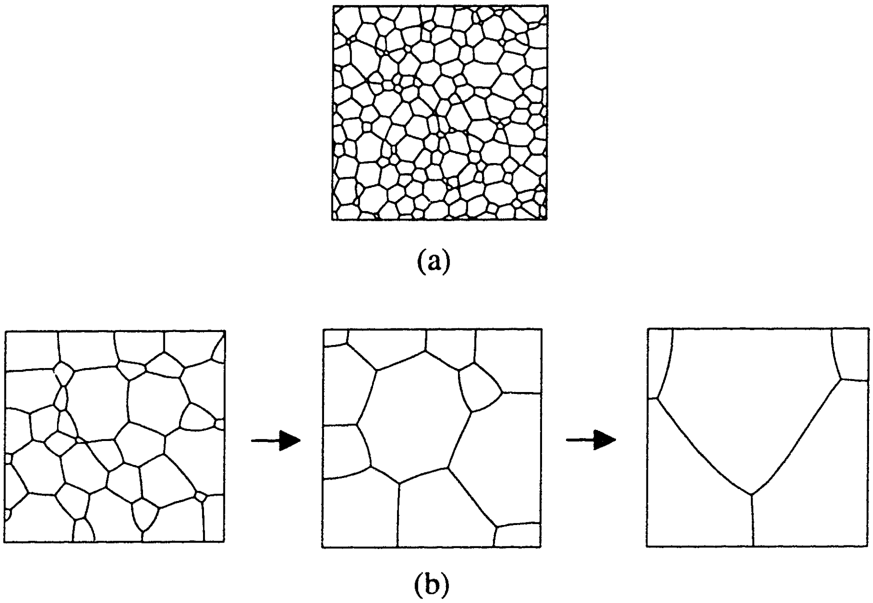
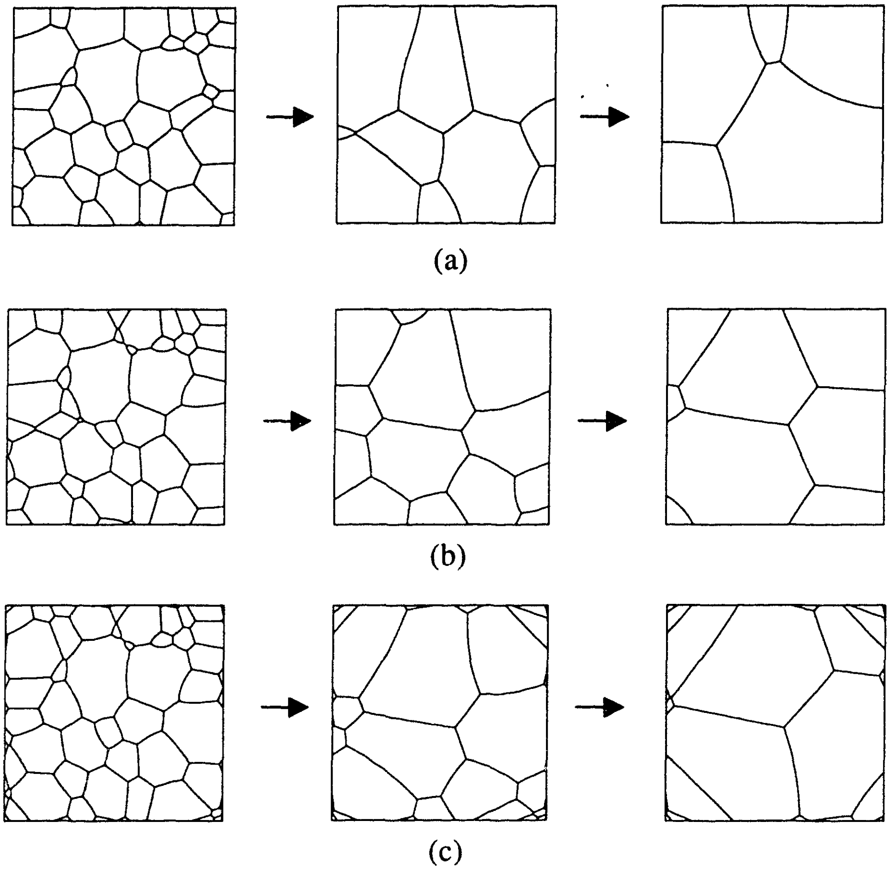


Figure G.4: (a) Starting microstructure obtained from a continuous film with a steady-state grain size distribution. (b) Grain structure evolution in which a force balance is applied at the pattern edge, so that the grain boundaries intersect the pattern edges at a 90° angle [Fay 00].

```
  if(single_tangent(x,y,xtrip,ytrip,&xvect,&yvect,linear_flag)==0 )
           {printf("ERROR in edge_bc.\n");WAIT_CHAR}
  theta = atan2(xvect, yvect)-PI/2.0;
  while (theta < -PI) theta += 2.0*PI;
  while (theta >  PI) theta -= 2.0*PI;

  theta = fabs(PI/2.0-fabs(theta));

  if(  fabs(theta)>(stagnant_angle)
       || (fabs(theta)>(moving_angle) && triple[triplept].stop==0) ) {

    Vu = xvect;
    Vu_pin = sin(stagnant_angle);  if (Vu < 0.0) Vu_pin *= -1.0;

    /* first guess: linear extrapolation from last motion */
    xtrip = -xtrip_old;

    /* Iterate until convergence criterion is reached or maximum iterations. */
    j=TRIPLE_ITERS;
    do
        { if (single_tangent(x,y,xtrip,ytrip,&xvect,&yvect,linear_flag)==0)
{ printf("ERROR in edge_bc().\n");WAIT_CHAR; return(-1); }
          Vu = xvect;
          xmove = triple_k*(Vu - Vu_pin);
          xtrip += xmove;  j--;
        }
    while ( (fabs(xmove)>TRIPLE_TOLERANCE) && (j>0) );

    /* hit corner? */
    if( xtrip>=x[2] ) { xtrip=x[2]; *flag_corner=0;}
    if( xtrip<=x[0] ) { xtrip=x[0]; *flag_corner=1;}

    /* transform result back */
```



(a)

(b)

(c)

Figure G.5: (a) Grain structure evolution in the case of an edge drag factor of 1%. (b) Grain structure evolution with a critical inclination angle for grain boundary grooving of 30%. (c) Grain structure evolution in the case of fixed edge triple junctions.

```
      glo_xtrip[0] = cos_incl*xtrip-sin_incl*ytrip + glo_xtrip[0];
      glo_ytrip[0] = sin_incl*xtrip+cos_incl*ytrip + glo_ytrip[0];

      return(j);
    }

    /* triple point stagnant */
    else {
      return(TRIPLE_ITERS);
    }
  }
```

Spatially varying drag effects and grain boundary grooving were introduced into the simulation by directly manipulating the grain boundary mobility in the function *move_segments* located in the file *numerics.c*.

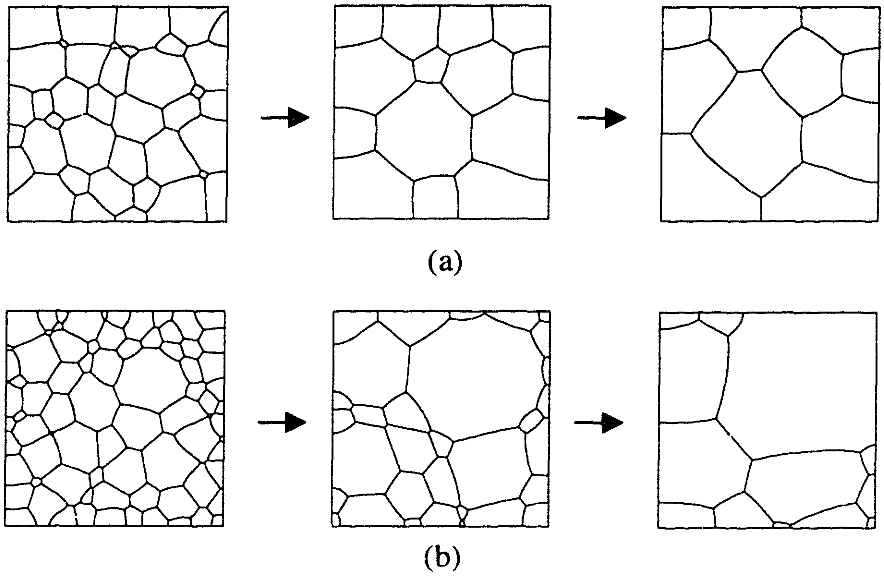


Figure G.6: (a) Grain structure evolution in a square-shaped film with inhomogenities in the thickness up to 50%, which leads to a gradient in κ_{cr}. (b) Grain structure evolution in the presence of a mobility gradient. The mobility decreases continuously toward the pattern edge.

G.3.2 Results

When grain boundary motion is allowed in continuous films, after a short time on the simulation time-scale, a steady-state structure develops in which the normalized grain-size and number-of-sides distributions are time invariant [Fay 99]. We patterned a square-shaped structure as shown in figure G.4 (a) from these films. Figure G.4 (b) shows the grain structure evolution when a force balance is enforced at the pattern edge, so that the grain boundaries intersect the pattern edges at a 90° angle [Fay 00].

Figure G.5 (a) shows the grain structure evolution in the case of an edge drag factor of 1%. Figure G.5 (b) shows the grain structure evolution assuming a critical curvature for grain boundary grooving at the pattern edge of 30%, and figure G.5 (c) shows the grain structure evolution in the case of fixed edge triple junctions, which translates to an edge drag factor of zero. If it is assumed that the thickness varies across the pattern by 50%, leading to a gradient in κ_{cr}, a grain structure evolution as shown in figure G.6 (a) is obtained. Finally, if the mobility decreases continuously toward the pattern edge, the grain structure evolves as shown in figure G.6 (b).

G.4 Discussion

Our experimental results shown in figure G.2 suggest that the simulation developed by Fayad et al. [Fay 00] does not accurately account for the effects of pattern edges on grain growth. We modified the simulation to account for the effects of pinning or drag forces on grain boundaries at the pattern edge, but they do not lead to the observed effects, as shown in figure G.5.

To reproduce the large difference in grain sizes between grains at the pattern edge and at the center of the pattern, the effects of the edges must extend into the metal. Assuming a thickness variation in the metal leading to a variation in the conditions for grain boundary grooving, the differences in grain sizes can be reproduced, but cross-sectional SEM studies have shown that the structures have a homogeneous thickness. Only a grain boundary mobility decreasing toward the edge leads to the observed effects. A reduction of the grain boundary mobility could be due to impurities or etch residuals diffusing from the pattern edge into the metal.

Our experiments also suggest that the grain growth simulator must be modified to be able to deal with rounded corners. It is expected that the corner shape affects the grain structure evolution especially in the case of concave corners, which are found, for example, in "T"-shaped structures.

Bibliography

[Aba 98] Abaqus, Version 5.8, general purpose finite element program, Hibbit, Karlson, and Sorensen, Inc., Pawtucket, RI, 1998.

[Abe 98] K. Abe, Y. Harada, and H. Onada, *Proceedings of the 36th International Reliability Physics Symposium*, 342 (1998).

[Abr 74] M. Abramovitz and I.R. Stegun, Handbook of Mathematical Functions, Dover Publ., New York, NY (1974).

[Ame 70] I. Ames, F.M. d'Heurle, and R.E. Horstmann, *IBM J. Res. Develop.* **14**, 461 (1970).

[And 84] M.P. Anderson, D.J. Srolovitz, G.S. Grest, and P.S. Sahni, *Acta metall.* **32**, 783 (1984).

[And 99] V. K. Andleigh, V. T. Srikar, Y.-J. Park, and C. V. Thompson, *J. Appl. Phys.* **86**, 6737 (1999).

[Ata 98] E.M. Atakov, T.S. Sriram, D. Dunnel, and S. Pizzanello, *Proceedings of the 36th Annual International Reliability Physics Symposium*, 348 (1998).

[Att 71] M.J. Attardo, R. Rutledge, and R.C. Jack, *J. Appl. Phys.* **42**, 4343 (1971).

[ATT 90] AT&T Reliability Manual, edited by D.J. Klinger, Y. Nakada, and M.A. Menendez, Van Nostrand Reinhold, New York, 1990.

[Bal 79] R.W. Balluffi and A.V. Granato, in *Dislocations in Solids*, edited by F.R.N. Nabarro (North-Holland, Amsterdam, 1979), Vol. 4.

[Bar 48] J. Bardeen and W. Brattain, *Phys. Rev.* **74**, 230 (1948).

[Ber 89] M.W. Bern and R.L. Graham, *Sci. Am.* **260**, 66 (1989).

[Bla 67] J.R. Black, in *Proceedings of the 6th Annual International Reliability Physics Symposium* (IEEE, New York, 1967), p.148.

[Ble 67] I.A. Blech and H. Sello, *Physics of Failures in Electronics* **5**, ed. T.S. Shilliday and J. Vacarro, Rome Air Development Center, 496 (1967).

[Ble 76] I. A. Blech and C. Herring, *Appl. Phys. Lett.* **29**, 131 (1976).

[Ble 77] I. A. Blech, *J. Appl. Phys.* **48**, 473 (1977).

[Bro 99] S.H. Brongersma, E.Richard, I.Vervoort, H.Bender, W.Vandervorst, S. Lagrange, G. Beyer, and K. Maex, *J. Appl. Phys.* **86**, 3642 (1999).

[Cab 98] C. Cabral, Jr. and J.M.E. Harper, *Proc. Adv. Metall. Conf.*, pp. 81-87, Materials Research Society, Pittsburgh, PA, 1998.

[Car 96] R. Carel, C. V. Thompson, and H. J. Frost, *Acta mater.* **44**, 2479 (1996).

[Cha 99] A. Chatterjee, *Europ. Semicond.* **21**, 39 (1999).

[Che 00] Y. Chery, D. Troxel, and C.V. Thompson, ongoing research. A demonstration of the tool is accessible on the World Wide Web at *http://nirvana.mit.edu/darpademo/all3tools*.

[Cho 89] J. Cho and C.V. Thompson, *Appl. Phys. Lett.* **54**, 2577 (1989).

[Chr 65] J.W. Christian, *The Theory of Transformations in Metals and Alloys*, International Series of Monographs on Metal Physics and Physical Metallurgy **7**, Pergamon Press, Oxford, First Edition 1965.

[Cle 95] J. J. Clement and C. V. Thompson, *J. Appl. Phys.* **78**, 900 (1995).

[Cle 97] J.J. Clement, *J. Appl. Phys.* **82,** 5991 (1997).

[Cle 99] J.J. Clement, S.P. Riege, R. Cvijetic, C.V. Thompson, *IEEE Trans. CAD* **18**, 576 (1999).

[Cor 96] T. H. Corman, C. E. Leiserson, and R. L. Rivest, *Introduction to Algorithms*, pp. 558. McGraw-Hill Book Comp., New York, NY (1996).

[Die 86] G. E. Dieter, *Mechanical Metallurgy*, McGraw-Hill (1986).

[Doh 87] R.D. Doherty, D.J. Srolovitz, A.D. Rollet, and M.P. Anderson, *Script mater.* **21**, 675 (1987).

[Ede 97] D. Edelstein, J. Heidenreich, R. Goldblatt, W. Cote, C. Uzoh, N. Lustig, P. Roper, T. McDevitt, W. Motsiff, A. Simon, J. Dukovic, R. Wachnik, H. Rathore, R. Schulz, L. Su, S. Luce, and J. Slattery, *IEEE Intl. Electron Devices Meeting Digest*, 773 (1997).

[EIA 95] EIA/JEDEC Standard JESD33A, Electronic Industry Association, Arlington, VA (1995).

[Ems] A demonstration of MIT/EmSim is accessible on the World Wide Web at *http://nirvana.mit.edu/emsim.*

[Ems Man] A user manual for MIT/EmSim is accessible on the World Wide Web at http://nirvana.mit.edu/~vab/emsimhelp

[Ent] Enthone-OMI, Inc., P.O. Box 1900, West Haven, CT.

[Esh 57] J. D. Eshelby, *Proc. Roy. Soc.* **A 241**, 376 (1957).

[Fan 97] D. Fan, C. Geng, L-Q. Chen, *Acta mater.* **45**, 1115 (1997).

[Fay 99] W. Fayad, C.V. Thompson, and H.J. Frost, *Scr. Met.* **40**, 1199 (1999).

[Fay 00] W. Fayad, *Ph.D. Thesis*, Massachusetts Institute of Technology (2000).

[Fil 95] R.G. Filippi et al , *J. Appl. Phys.* **78**, 3756 (1995).

[Fil 96] R.G. Filippi, R.A. Wachnik, H. Aochi, J.R. Lloyd, and M.A. Korhonen. *Appl. Phys. Lett.* **69**, 2350. (1996).

[Fro 87] H. J. Frost and C.V. Thompson, and D.T. Walton, *Acta Mat.* **35**, 529 (1987).

[Fro 88] H.J. Frost, C.V. Thompson, C.L. Howe, and J. Whang, *Scr. metall.* **22**, 65 (1988).

[Fro 90] H.J. Frost, C.V. Thompson, and D.T. Walton, *Acta Mat.* **38**, 1455 (1990).

[Fro 92] H.J. Frost, C.V. Thompson, and D.T. Walton, *Acta Mat.* **40**, 779 (1992).

[Fro 94] H.J. Frost, Y. Hayashi, C.V. Thomspon, and D. T. Walton, , *Mat. Res. Soc. Symp. Proc.* **317**, 431 (1994).

[Gan 72] A. Gangulee and F.M. d'Heurle, *Thin Solid Films* **12**, 399 (1972).

[Gan 73] A. Gangulee and F.M. d'Heurle, *Thin Solid Films* **16**, 227 (1973).

[Gan 75] A. Gangulee and F.M. d'Heurle, *Thin Solid Films* **25**, 317 (1975).

[Glei 99] R.J. Gleixner, W.K. Meyer, and W.D. Nix, *AIP Conference Proceedings* **491** of the 5th International Workshop on Stress-Induced Phenomena in Metallization, Stuttgart, Germany, pp.51 (1999).

[Gro 98] M. E. Gross, C. Lingk, T. Siegrist, E. Coleman, W. L. Brown, K. Ueno, Y. Tsuchiya, N. Itoh, T. Ritzdorf, J. Turner, K. Gibbons, E. Klawuhn, M. Biberger, W. Y. C. Lai, J. F. Miner, G. Hu, and F. Zhang, *Proc. Mater. Res. Soc.* **518**, pp. 293 (1998).

[Har 99] J.M.E. Harper, C. Cabral, Jr., P.C. Andricacos, L. Gignac, I.C. Noyan, K.P. Rodbell, and C.K. Hu, *J. Appl. Phys.* **86**, 2516 (1999).

[Has 90] G.N. Hassold, E.A. Holm, and D.J. Srolovitz, *Scripta metall,.* **24**, 101 (1990).

[Hau 99] S. P. Hau-Riege and C. V. Thompson, *Scr. Mat.* **41**, 403 (1999).

[Hau 00a] S. P. Hau-Riege and C. V. Thompson, *Appl. Phys. Lett.* **76**, 309 (2000).

[Hau 00b] S. P. Hau-Riege and C. V. Thompson, *"Modeling and Experimental Characterization of The Reliability of Interconnect Trees"*, unpublished research.

[Hau 00c] S.P. Hau-Riege and C.V. Thompson, to be pubished in the Proceedings of the Materials Research Society meeting, San Francisco, CA, Spring 2000.

[Hau 00d] S. P. Hau-Riege and C. V. Thompson, *"Electromigration-Saturation in a Simple Interconnect Tree"*, unpublished research.

[Hau 00e] S. P. Hau-Riege and C. V. Thompson, *"The Effects of the Mechanical Properties of the Confinement Material on Electromigration in Metallic Interconnects*", unpublished research.

[Hau 00z] C.S. Hau-Riege and C.V. Thomson, unpublished research.

[Her 50] C. Herring, *J. Appl. Phys.* **21**, 437 (1950).

[Her 97] O. Herrera, A. Olguin, M. Ortiz, and C.H. Woerner, *Phys. Rev.* **E55**, 4475 (1997).

[Hsu 98] W.-Y. Hsu, Q.-Z. Hong, L.-Y. Chen, Y.-C. Lu, A. Wilson, V. Cordasco, M. Jain, R. Tao, J. D. Luttmer, R. H. Havemann, and R. Mosely, *Advanced Metallization and Interconnect Systems for ULSI Applications* (Mater. Res. Soc., Warrendale, PA (1998)), pp. 413.

[Hu 99a] C. K. Hu, R. Rosenberg, and K. Y. Lee, *Appl. Phys. Lett.* **74**, 2945 (1999).

[Hu 99b] C.-K. Hu, R. Rosenberg, H.S. Rathore, D.B. Nguyen, and B. Agarwala, *Proceedings of the IEEE 1999 International Interconnect Technology Conference*, pp.267-9, IEEE, Piscataway, NJ (1999).

[Hun 75] H. B. Huntington, *Diffusion in Solids*. Academic Press, New York, 1975.

[Hun 92] O. Hunderi and N. Ryum, *Acta metall.* **40**, 543 (1992).

[Hun 97] A.W. Hunt, S.P. Riege, and J.A. Prybyla, Appl. Phys. Lett. **70**, 2541 (1997).

[Iye 84] S.S. Iyer and C.-Y. Tong, *IEEE Trans. Electron. Dev.* **ED-31**, 1468 (1984).

[Kad 97] B.K. Kad and P.M. Hazzledine, *Mat. Sci. and Eng.* **A238**, 70 (1997).

[Kam 99] M. Kamon, N. Marques, Y. Massoud, L. Silveira, and J. White, *Proceedings 1999 Design Automation Conference*, 910 (1999).

[Kan 97] S.H. Kang and J.M. Morris, Jr., *J. Appl. Phys.* **82**, 1592 (1997).

[Kin 96] W. D. Kingery, Y. M. Chiang, and D. Birnie, *Physical Ceramics*. John Wiley and Sons, Inc., New York, 1996.

[Jia 98] Q.-T. Jiang, R. Mikkola, B. Carpenter, and M.E. Thomas, *Proc. Adv. Metall. Conf.*, pp. 177-181, Materials Research Society, Pittsburgh, PA, 1998.

[Kin 80] E. Kinsbron, *Appl. Phys. Lett.* **36**, 968 (1980).

[Kir 91] R. Kircheim and U. Kaeber, *J. Appl. Phys.* **70**, 172 (1991).

[Kit 98] J. Kitchin and T.S. Sriram, AIP Conference Proceedings **418** of the 4th International Workshop on Stress Induced Phenomena in Metallization, Tokyo, Japan, pp.495 (1998).

[Kno 91] D.B. Knorr, D.B. Tracy, and K.P. Rodbell, *Appl. Phys. Lett.* **59**, 3241 (1991).

[Kno 93] D. B. Knorr, *Proc. Mat. Res. Soc. Symp.* **309**, 75 (1993).

[Kno 95] B.D. Knowlton, R.I. Frank, and C.V. Thompson, *MRS Symposium Proceedings* **391,** 361, (1995).

[Kno 95a] B.D. Knowlton, J.J. Clement, R.I. Frank, and C.V. Thompson, *MRS Symposium Proceedings* **391,** 189 (1995).

[Kno 97] B.D. Knowlton, J.J. Clement, and C.V. Thompson, *J. Appl. Phys.* **81**, 6073 (1997).

[Kno 97b] B.D. Knowlton, *Ph.D. Thesis*, Massachusetts Institute of Technology (1997).

[Kno 98] B. D. Knowlton and C. V. Thompson, *J. Mater. Res.* **13**, 1164 (1998).

[Kor 93] M. A. Korhonen, P. Boergesen, K.N. Tu, and Che-Yu Li, *J. Appl. Phys.* **73**, 3790 (1993).

[Kra 95] J. R. Kraayeveld, A. H. Verbruggen, A. W.-J. Willemsen, and S. Radelaar, *Appl. Phys. Lett.* **67**, 1226 (1995).

[Kri 92] O. Krichevsky and J. Stavans, *Phys. Rev.* **B46**, 10579 (1992).

[LaC 86] D.J. LaCombe and E.L. Parks, *Proc. 24th Int. Reliab. Phys. Symp.* 1 (1986).

[Lau 96] K.S.Y. Lau, J.S. Drage, N.P. Hacker, N.M. Rutherford, R.R. Katsanes, B.A. Korolev, T.A. Krajewski, S.P. Lefferts, H. Sayad, P.R. Sebahar, A.R. Smith, W.B. Wan, and E.C. White, *Proceedings Thirteenth International VLSI Multilevel Interconnection Conference (VMIC),* pp. 92-7. Tampa, FL (1996).

[Li 97] D. Y. Li and J. A. Szpunar, *J. Mater. Sci.* **32**, 5513 (1997).

[Lie 90a] B.-K. Liew, P. Fang, N. W. Cheung, and C. Hu, UCB/Electronics Research Lab. Rep. M90/3, Electronic Research Lab. of University of California, Berkeley, Jan. 1990.

[Lie 90b] B.-K. Liew, P. Fang, N. W. Cheung, and C. Hu, *Proc. 28th Int. Reliab. Phys. Symp.* 111 (1990).

[Lie 92] B.-K. Liew, P. Fang, N. W. Cheung, and C. Hu, *IEEE Trans. ED* **39**, 2472 (1992).

[Lin 98] C. Lingk and M.E. Gross, *J. Appl. Phys.* **84**, 5547 (1998).

[Lin 99] C. Lingk, M.E. Gross, and W.L. Brown, *Appl. Phys. Lett.* **74**, 682 (1999).

[Lok 99] A.L.S. Loke, J.T. Wetzel, P.H. Townsend, T. Tanabe, R.N. Vrtis, M.P. Zussman, D. Kumar, C. Ryu, and S.S. Wong, *IEEE Trans. ED* **46**, 2178 (1999).

[Lon 91] H.P. Longworth and C.V. Thompson, *J. Appl. Phys.* **69**, 3929 (1991).

[Lüc 63] K. Lücke and H.P. Stuwe, in *Recovery and Recrystallization of Metals*, edited by L. Himmel (Interscience, New York, 1963), p. 171.

[Mag 85] J.K. Outserhout, G.T. Hamachi, R.N. Mayo, W.S. Scott, and G.S. Taylor, *IEEE Design & Test of Computers* **2**, 19 (1985).

[Mai 89] J. Maiz, *Proc. 27th Int. Reliab. Phys. Symp.* 220 (1989).

[McC 80] E. M. McCreight, CSL-80-9, Xerox PARC, Palo Alto, CA (1980).

[McL 69] M. McLean and B. Gale, *Phil. Mag.* **20**, 1033 (1969); M. McLean, *Acta Met.* **19**, 387 (1971).

[Mor 99] D. Morton, *Proc. IEEE* **6**, 1049 (1999).

[Mul 56] W.W. Mullins, *J. Appl. Phys.* **27**, 900 (1956).

[Mul 58] W.W. Mullins, *Acta Met.* **4**, 414 (1958).

[Mur 95] S.P. Murarka and S.W. Hymes, *Crit. Rev. Solid State Mat. Sci.* **20**, 87 (1995).

[Neu 52] J. von Neumann, *Metal Interfaces* (Cleveland, OH: American Society for Metals), p. 108 (1952).

[Nis 97] T. Nishizawa, I. Ohnuma, and K. Ishida, *Mater. Trans. JIM,* **38**, 950 (1997).

[Nix 89] W.D. Nix, *Met. Trans.* **20A**, 2217-2245 (1989).

[Noy 87] I.C. Noyan and J.B. Cohen, *Residual Stress*, Springer Verlag, New York, NY, 1987.

[Nye 95] J. F. Nye, *Physical Properties of Crystals*, Oxford University Press, New York (1995).

[Olg 86] D.L. Olgaard and B. Evans, *J. Am. Ceram. Soc.* **69**, C272 (1986).

[Par 97] Y.-J. Park and C.V. Thompson, *J. Appl. Phys.* **82**, 4277 (1997).

[Par 99] Y.-J. Park, V.K. Andleigh, and C.V. Thompson, *J. Appl. Phys.* 85, 3546 (1999).

[Par 99b] A. Paranjpe, R. Bubber, L. Veto, Chang Guihua, S. Gopinath, J. Dalton, and M. Moslehi, *Proceedings of the IEEE 1999 International Interconnect Technology Conference*, pp.119-21, IEEE, Piscataway, NJ (1999).

[Pri 97] D.T. Price, R.J. Gutmann, and S.P. Murarka, *Thin Solid Films* **308-309**, 523 (1997).

[Pry 96] J.A. Prybyla, S.P. Riege, and A.W. Hunt, *Materials Research Society Proceedings, Materials Reliability in Microelectronics VI*, Materials Research Society, Pittsburgh (1996).

[Pry 97] J.A. Prybyla, S.P. Riege, and A.W. Hunt, 1997 TMS Annual Meeting: Advances in Electrically Conductive Materials.

[Pry 98] J.A. Prybyla, S.P. Riege, S.P. Grabowski, and A.W. Hunt, *Appl. Phys. Lett.* **73**, 1083 (1998).

[Qua] Modular Integrated Reliability Analyzer (MIRA), QualiTau Inc., Sunnyvale, CA 94086.

[Rao 93] G.K. Rao, *Multilevel Interconnect Technology*, Intel/McGraw-Hill, New York, 1993.

[Rie 95a] S.P. Riege, A.W. Hunt, and J.A. Prybyla, Materials Research Society Spring Meeting 1995, *Materials Research Society Proceedings*, Materials Reliability in Microelectronics V, pp.249, Materials Research Society, Pittsburgh (1995).

[Rie 95b] S.P. Riege, A.W. Hunt, and J.A. Prybyla, 1995 International Conference on Solid State Devices and Materials, Osaka, August 21-24, published in *1995 Extended Abstracts of the 1995 International Conference on Solid State Materials and Devices*, The Japan Society of Appied Physics, Tokyo (1995).

[Rie 96] S. P. Riege, A. W. Hunt, and J. A. Prybyla, *Appl. Phys. Lett.* **69**, 2367 (1996).

[Rie 98a] S. P. Riege, C. V. Thompson, and J. J. Clement, *IEEE Trans. ED* **45**, 2254 (1998).

[Rie 98b] S.P. Riege, C.V. Thompson, and H.J. Frost, *Proceedings of the Third International Conference on Grain Growth (ICGG-III)*, Pittsburgh, PA, pp. 295 (1998).

[Rie 98c] S.P. Riege, V. Andleigh, C.V. Thompson, and H.J. Frost, Materials Research Society Proceedings, Semiconductor Process and Device Performance Modeling , pp.219-224, Materials Research Society, Warrendale, PA (1998).

[Rie 99] S.P. Riege, C.V. Thompson, and H.J. Frost, *Acta Mat.* **47**, 1879 (1999).

[Roc 98] S. Rochel, G. Steele, J. R. Lloyd, S. Z. Hussain, and D. Overhauser, *Proc. 36th Int. Reliab. Phys. Symp.* 356 (1998).

[Rod 92] K. P. Rodbell, D. B. Knorr, and D. P. Tracy, *Proc. Mat. Res. Soc. Symp.* **265**, 107 (1992).

[Ros 77] R. Rosenberg, *J. Vac. Sci. Technol.* **9**, 263 (1972).

[San 98] J. Sanchez, Jr., and P. R. Besser, *Proc. 1998 IEEE Internat. Interconnect Conf.*, San Francisco, CA, pp. 247 (1998).

[Sau 92] A. I. Sauter and W. D. Nix, *IEEE Trans. Comp.* 15, 594 (1992).

[Sch 80] J.M. Schoen, *J. Appl. Phys.* **51**, 513 (1980).

[Sch 87] H. A. Schafft, *IEEE Trans. ED* **34**, 664 (1987).

[Sha 86] M. Shatzkes and J.R. Lloyd, *J. Appl. Phys.* **59**, 3890 (1986).

[Sch 94] H. A. Schafft, J. S. Suehle, and J. Albers, Integrated Reliability Workshop, Final Report (1994), pp.12 –19.

[Sch 87] H.A. Schafft, *IEEE Trans Electron. Dev.* **ED-34**, 664 (1987).

[She 89] P. Shewmon, *Diffusion in Solids.* Minerals, Metals and Materials Soc., Warrendale, PA, 1989.

[She 99] Y.-L. Shen, *Proc. 37th Int. Reliab. Phys. Symp.* 283 (1999).

[Sim 71] G. Simmons and H. Wang, *Single Crystal Elastic Constants and Calculated Aggregate Properties: A Handbook,* M.I.T. Press, Cambridge (1971).

[Sri 98] V. T. Srikar and C. V. Thompson, *Appl. Phys. Lett.* **72**, 2677 (1998).

[Sri 99] V.T. Srikar and C.V. Thompson, *Appl. Phys. Lett.* **74**, 37 (1999).

[Sro 84] D.J. Srolovitz, M.P. Anderson, G.S. Grest, and P.S. Sahni, *Acta metall.* **32**, 1429 (1984).

[Sta 93] J. Stavans, *Rep. Prog. Phys.* **56**, 733 (1993).

[Ste 91] I. Stewart, *New Scientist* **16**, 26 (1991).

[Sto 80] J. Stoer and R. Bulirsch, *Introduction to Numerical Analysis*, Springer-Verlag, New York, NY (1980).

[Suo 98] Z. Suo. *Acta mater.* **46**, 3725 (1998).

[Tam 99] M. Tammaro, *J. Appl. Phys.* **86**, 3612 (1999).

[Tho 86] C. V. Thompson and J. Cho, *IEEE Electron. Dev. Lett.* **12**, 667 (1986).

[Tho 90] C.V. Thompson, *Ann. Rev. Mat. Sci.* **20**, 245 (1990).

[Tho 93] C.V. Thompson and H. Kahn, *J.Electron. Mater.* **22**, 581 (1993).

[Tho 93a] C.V. Thompson and J.R. Lloyd, *MRS Bulletin* **18**, 19 (1993).

[Tho 95] C.V. Thompson and R. Carel, *Mat. Sci. Eng.* **B32**, 211 (1995).

[Tho 99a] C.V. Thompson, S.P. Riege, and V. Andleigh, *AIP Conference Proceedings* **491** of the 5th International Workshop on Stress-Induced Phenomena in Metallization, Stuttgart, Germany, pp.62 (1999).

[Tho 99b] C.V. Thompson and S.P. Riege, Materials Research Society Spring Meeting 1999, San Francisco, CA.

[Tho 00] C.V. Thompson, S.P. Hau-Riege, V.K. Andleigh, C.S. Hau-Riege, Y. Cherry, and D.E. Troxel, Materials Research Society Meeting, San Francisco, CA, Spring 2000.

[Tom 85] I.V. Tomov, D.S. Stoychev, and I.B Vitanova, *J. Appl. Electrochem.* **15**, 887 (1985).

[Tou 85] Y. S. Touloukian, R. K. Kirby, R. E. Taylor, and P. D. Desai, in *Thermophysical Properties of Matter* 12, IFI/Plenum Press, New York, 1985.

[Tow 83] J.M. Towner and E.P. van de Ven, *Proc. 21th Int. Reliab. Phys. Symp.* 36 (1983).

[Tow 90] J.M. Towner, *28th Annual Proceedings Reliability Physics* 100 (1990).

[Ull 84] J. D. Ullmann, *Computational Aspects of VLSI*, pp. 382. Computer Science Press, Inc., Rockville, MD, (1984).

[Van 99] L. Vanasupa, Y.-C. Joo, P. R. Besser, and S. Pramanick, *J. Appl. Phys.* **85**, 2383 (1999).

[Vin 94] R. P. Vinci, E. M. Zielinski, and J. C. Bravman, *Proc. Mat. Res. Soc. Symp.* **338**, 289 (1994).

[Vin 97] R.P. Vinci and J. C. Bravman, *Proc. Mat. Res. Soc. Symp.* **436**, 411 (1997).

[Wae 99] J. Waeterloos, M. Simmonds, A. Achen, and M. Meier, *Europ. Semicond.* **21**, 26 (1999).

[Wal 91] D.T. Walton, H.J. Frost, and C.V. Thompson, *MRS Symposium Proceedings* **225**, 219 (1991).

[Wal 92] D.T. Walton, H.J. Frost, and C.V. Thompson, *Appl. Phys. Lett.* **61**, 402 (1992).

[Wan 98] P.-C. Wang, G.S. Cargill III, I.C. Noyan, and C.-K. Hu, *Appl. Phys. Lett.* **72**, 1296 (1998).

[Wei 98] D.Weiss, P. Müllner, and C.V. Thompson, and E. Arzt, *Proceedings of the Third International Conference on Grain Growth (ICGG-III)*, Pittsburgh, PA, pp. 537 (1998).

[Wör 89] C.H. Wörner, *Scr. metall.* **23**, 1909 (1989).

[Zen 49] Zener, C., quoted by C.S. Smith, *Trans. TMS-AIME*, **175**, 15 (1949).

THESIS PROCESSING SLIP

FIXED FIELD: ill ____________ name ____________

index ____________ biblio ____________

► COPIES: [✓ Archives] Aero Dewey Eng Hum

Lindgren Music Rotch [✓ Science]

TITLE VARIES: ►□ ____________

NAME VARIES: ►☑ Peter

IMPRINT: (COPYRIGHT) ____________

► COLLATION: ____________

► ADD: DEGREE: ________ ► DEPT.: ____________

SUPERVISORS: ____________

NOTES:

cat'r date

page

► J170

► DEPT: Mat. Sci. & E.

► YEAR: 2000 ► DEGREE: Ph.D.

► NAME: HAU-RIEGE, Stefan P.