MULTI SIGNAL-DOMAIN MODELING OF SOLID-STATE TRANSDUCERS

MULTI SIGNAL-DOMAIN MODELING OF SOLID-STATE TRANSDUCERS

A Theoretical Framework Based on Irreversible Thermodynamics Mixed Finite Elements and Multigrid

Een onderzoek naar het fysisch en numeriek modelleren van (silicium) halfgeleider sensoren in het thermische en/of elektrische energiedomein.

PROEFSCHRIFT

ter verkrijging van de graad van doctor
aan de Technische Universiteit Delft,
op gezag van de Rector Magnificus,
prof. ir. K.F. Wakker,
in het openbaar te verdedigen
ten overstaan van een commissie
aangewezen door het College van Dekanen
op maandag 29 november 1993 te 14.00 uur

door

Dave C. van Duyn

elektrotechnisch ingenieur
geboren te Christchurch, New Zealand

Delft University Press / 1993

Dit proefschrift is goedgekeurd door de promotor:
Prof. dr. ir. S. Middelhoek.

Leden van de promotiecommissie:

De Wnd. Rector Magnificus	Technische Universiteit Delft
Prof. dr. ir. S. Middelhoek	Technische Universiteit Delft
Prof. dr. M. Kleefstra	Technische Universiteit Delft
Prof. dr. ir. P.M. Dewilde	Technische Universiteit Delft
Prof. dr. H. Wallinga	Universiteit Twente
Dr. G. Wachutka	ETH Zurich
Dr. ir. G. Mur	Technische Universiteit Delft
Dr. ir. A.W. van Herwaarden	Xensor Integration B.V.

Publisher: Delft University Press
Stevinweg 1
2628 CN Delft, The Netherlands
Tel: +31 15 783254
Fax: +31 15 781661

Typeset: This dissertation was typeset in Roman, Helvetica and Courier by the author using FRAMEMAKER ™

CIP-GEGEVENS KONINKLIJKE BIBLIOTHEEK, DEN HAAG

van Duyn, Dave Cornelis

Multi Signal-Domain Modeling of Solid-State Transducers
Dave Cornelis van Duyn. - Delft: Delft University Press.
Ph.D. Dissertation Delft University of Technology. -Ill, - With summary in Dutch.
ISBN 90-6275-932-7/CIP
NUGI 841

Subject headings: Solid-State Transducers, Semiconductors, Irreversible Thermodynamics, Mixed Finite Elements, Multigrid.

The world is very large compared to one's own background and past experience may have little relevance to the world of tomorrow. If experience is truly to add value to a lesson, one must extract the universals and not the specifics.

IEEE President Jerome J. Suran, IEEE Spectrum 1991.

To Edith, LJ and my parents

Table of Contents

Chapter 1 A General Introduction

Chapter 2 Generic Aspects of Physically Based Models

Chapter 3 A Thermodynamic Model for the Thermal and Electrical Energy Domains

Chapter 4 A Mixed Finite Element Discretization Method for the Thermoelectric Problem

Chapter 5 An Adaptive Multigrid Solution Method for the Thermoelectric Problem

GIVE ME A LIGHT,
that I may tread safely into the unknown

Chapter 1

A General Introduction

The purpose of this chapter is to introduce the reader to the research reported in this dissertation, which is a (partial) presentation of a systematic investigation on the concept of "Multi-Signal Domain Modeling and Simulation of Solid-State Transducers". Obviously, such a broad objective involves many different subjects, borrowing from physics, mathematics and computer science. Moreover, the time needed to fully achieve such an objective may easily exceed the time available for a normal Ph.D. research project. In this chapter we therefore try to establish a unified view of the subject and define the context in which we wish to carry out the work. The emphasis is on the extraction of the generic features that allow the definition of a general framework for the modeling and simulation of solid-state transducers. Such a framework can then be used as guidance in the definition of a tool kit on top of which the framework can (partially) be implemented. Note that the definition of a generic tool kit supports the addition of yet unimplemented features of the framework. This chapter is structured as follows. Section 1.1 discusses the main reasons for carrying out this research. Section 1.2 presents a discussion on the basic aspects of physically based modeling and simulation. Section 1.3 discusses the topic of modeling and simulation in the context of solid-state transducer research. Section 1.4 discusses the more specific aims of this dissertation, which is the modeling and simulation of the thermoelectric effects in silicon solid-state transducers. It also briefly discusses some possible future research projects that lie within the scope of this field. Section 1.5 gives an outline of the work presented in subsequent chapters. Section 1.6 discusses some issues related to the notation used in this work. Finally, Section 1.7 discusses some features related to the software (tool kit) developed during the research.

1.1 Solid-State Transducer Research

In the past two decades we have seen a rapidly growing interest, from academic as well as industrial laboratories, in the field of solid-state transducer research and development. This tendency has led to many new transduction principles being described in the literature. For instance, timely information on solid-state transducer research can be found in dedicated journals like "Sensors and Actuators" and "Sensors and Materials". Also the proceedings of the "Transducers" and "Eurosensors" conferences provide valuable information. However, this wealth of information has also led to the problem of the researcher and engineer loosing the overall view on the subject, making it increasingly difficult to find one's way in the transducer "forest". It was therefore only natural that, besides the more practical and experimental research effort, a general science of transducers started to evolve with its main objectives to compile, systematize and unify knowledge on transducing principles. Noticeable examples are the "sensor effect cube" proposed in [Middelhoek 1981][Middelhoek 1986], the equilibrium thermodynamic treatment of sensors in [Ylilammi 1989] and the thermodynamic systems approach in [Duyn 1990]. An excellent further elaboration on the thermodynamics of transducers was put forward in [Kirschner 1991][Kirschner 1992].

In this dissertation it is argued that the merits of the previously mentioned contributions to the general science of transducers, besides providing a unified view of the solid-state transducer field, also provides a systematic approach to the modeling and simulation of solid-state transducers [Duyn 1990]. The impetus to solid-state transducer modeling and simulation is of course, just as in the "plain" IC modeling and simulation field [Selberherr 1984], driven by the desire to be able to perform rapid prototyping of designs, to reduce the costly trial-and-error fabrication cycles, and to obtain insight into the physical properties of the device interior not readily accessible by experiment. At present, despite the maturity of the modeling and simulation of semiconductor IC devices, the field of solid-state transducer modeling and simulation is still in its infancy and has yet to achieve the same level of maturity [Duyn 1992]. Major difficulties are, of course, the diversity of the transducer field and the complicated interrelationships between the various signal domains [Middelhoek 1986][Duyn 1990].

1.2 Physically Based Modeling

In this section some considerations on basic modeling paradigms are discussed that may help to put the subject into perspective.

"Modeling" and "Model" are words with a remarkably broad range of meanings and are likely to be interpreted very differently by people from different disciplines. The reason for this is that the process of modeling is extremely domain specific, that is, it is intricately tied to the particular discipline one is in. To name a few: climate modeling, cosmic modeling, economic modeling, performance modeling, cognitive modeling, etc. This fact makes it extremely difficult to devise generic (high-level) modeling strategies, and we usually must be satisfied with a domain-specific approach to modeling. The modeling process usually starts out with some cognitive notion of the model. At this stage it includes ideas and properties that are implicit, that is, they depend on context and real-world experience in order to make sense. Gradually, a series of refinements is applied to this notion; these refinements pin down the notion, transforming tacit properties into explicit ones and narrowing the scope to focus on elements that are relevant to the purpose of the model. This process can be formalized to some extend by identifying the following stages: *abstraction*[1], *representation* and *implementation.*

The abstraction stage is concerned with the identification of the set of features and characteristics of the entity to be modeled. The process results in an abstract notion (abstraction) of the entity, which unavoidably involves some form of idealization. We then proceed to a representation by formalizing the abstraction. Unlike the abstraction the representation should be concrete and precise, that is, it should not be a source of ambiguity. Hence, it should provide sufficient information to create an actual implementation of the model in some medium[2] (blueprint). Note that there is no one-to-one correspondence between the three stages, for instance, it is possible to have many different implementations of a single representation. Also many different representations of a single abstraction may be possible. Which one to choose is often a design decision guided by accuracy and reliability constraints. The ultimate aim is to support the modeling paradigm of a given domain by means of a computer program, where the user supplies the abstraction and subsequently the computer chooses the representation and implementation and proceeds directly to the simulation of the model[3].

We may narrow down the field a little by interpreting the process of modeling as the creation of abstract mathematical models of observed or hypothetical physical objects. We might call this *physically-based modeling*. The resulting model may be used to predict the actual (real-time) behavior of a physical object via (numerical) simulation. Note that, depending on the basic physical modeling paradigm used, it is very likely that for a single physical object there exists a whole hierarchy of models, each with its characteristic complexity, reliability and accuracy. Which one to choose is often a design decision dictated by complexity, accuracy and reliability constraints. Also this process can to some extent be formalized (cf. Figure 1-1).

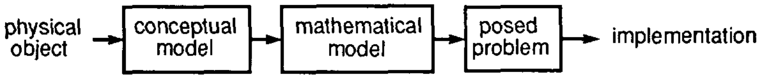

Figure 1-1 The three stages in physically based modeling.

[1] Abstraction: The act or process of separating the inherent qualities or properties of something from the actual physical object or concept to which they belong [TAHD 1978].

[2] The medium we focus on is a computer program but, in general, this does not have to be the case.

[3] Obviously, such an approach has its merits and demerits. However, we shall not dwell on this.

In this case, the choice of the physical modeling paradigm to a large extent embodies the abstraction stage. Hence, the attention shifts to the representation stage. Essentially, the modeling process may now be seen as a series of refinements to the representation, starting with a *conceptual model* of the physical object to be modeled. The conceptual model is a (loose) description of the properties, features, characteristics, etc. of the physical object. The conceptual model may then be used to build an abstract *mathematical model,* which essentially is a collection of mathematical equations and definitions of terms that go into the equations, that describe the behavior of the model. The equations are "context free", that is, they are purely mathematical expressions that are complete (closed) without requiring properties or definitions from the conceptual model, neither do they depend on the (computer) implementation details[4]. Note that the physical interpretation of the mathematical model is embedded in the conceptual model.

The mathematical equations stating the model are not "problems" in themselves but rather *predicates,* i.e. statements of relationships between entities in the model. From a computer science point of view, a mathematical model is *declarative* rather then *procedural.* In other words, the mathematical model does not contain any posed problems or instructions for solving problems, but merely states equations. However, in general, we want a physically based model to do something, i.e. simulate the behavior of the physical object being modeled. Mathematically, this translates into specifying and solving a *posed problem.* For this purpose we may formulate (procedural) mathematical problems that more or less configure the mathematical model for that problem. In general, a posed problem includes both a conceptual notion of the task to be performed and a precise formulation of the corresponding mathematical problem. The result of these three steps, going from a real or hypothetical physical object to the implementation, is a physically based model.

The separation of the conceptual model and the posed problems from the mathematical model is in fact a matter of convenience; it allows a single mathematical model to be reused for many conceptual models and posed problems. The mathematical model can be implemented in such a way that it is configurable and the configuration phase can be taken care of by means of an interactive user interface or a simple file base interpretative interface. This is the philosophy on which many existing programs which support high- level *computer based modeling* are based. One and other is shown in Figure 1-2.

Note that the above implies a separation between *simulation,* which emphasizes computation of the behavior of a given physically based model, and

[4] The notion of an explicit mathematical model of natural phenomenon, separate from both its numerical solution and its physical interpretation, is well known in applied mathematics.

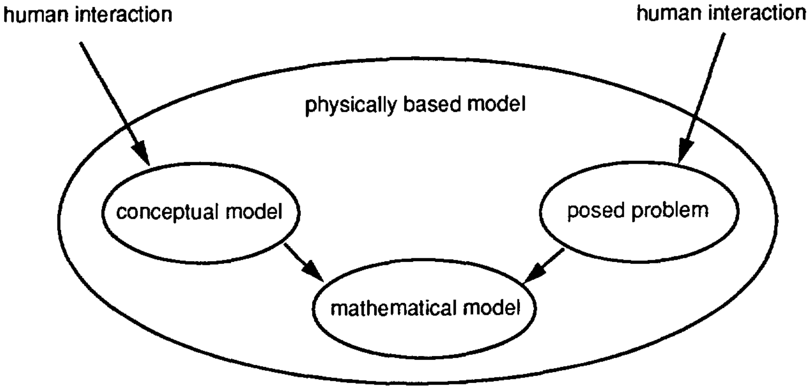

Figure 1-2 A physically based model basically consists of a conceptual model, a mathematical model and a posed problem. The mathematical model may support a host of conceptual models and posed problems.

modeling, which focuses on the creation and implementation of the (blackboard) model. Obviously, in order to perform simulations, the given model has to be implemented in a computer program that can be executed by a computer, i.e. it has to be transformed into a computational model. Opposed to modeling, which is the field of physics and mathematics, the implementation of the model is the field of numerical mathematics and software engineering. Numerical mathematics provides techniques to design procedural descriptions which state how to solve the posed problem step by step. Since programming languages are essentially procedural this is a great step towards an actual implementation. More formally, one could say that numerical mathematics bridges the gap between the inherent continuous formulation of the model and the discrete nature of the software and hardware that is used to implement the model. Often a host of applicable numerical techniques is available, each with its specific merits with respect to the problem to be solved. Which technique to choose is a design decision to be taken by the modeler. Finally, software engineering provides the tools to actually implement the model. Various software implementation paradigms are in existence, however, the recently introduced object-oriented programming paradigm (OOP) has received much attention [Stroustrup 1991]. As is well known, OOP can significantly enhance software development speed, ease of software maintenance, reliability and reusability, however, we return to this topic later.

In the following section we discuss the specific case of the modeling of solid state transducers.

1.3 Modeling of Solid-State Transducers

The process of modeling solid-state transducers may be viewed as a domain-specific application of physically based modeling. However, in addition we can distinguish two basic approaches. On one side we have the approach where we strive for "simple" analytical models. Here the abstract mathematical model consists of a set of equations which is explicit with respect to the desired results. Such models can easily be implemented in some general-purpose simulation program[5]. Such an approach can be very useful in gaining an understanding of the basic operation principle of the transducing device and is in fact an indispensable tool for checking the initial feasibility of an idea. However, because of the numerous heuristic approximations needed to keep the model analytical, this approach often is only reliable under very special operating conditions and device configurations[6].

On the other side we have the approach where as many as possible of the physical principles are kept in the model, that is, as many as are needed to achieve the desired level of realism. As a result, the abstract mathematical model usually consists of a set of equations which is implicit with respect to the desired results. Here we are usually dealing with a set of partial differential equations (PDE) with appropriate boundary and initial conditions defined on some computational domain. The obvious choice to implement such models is to use some variant of the finite element method (FEM)[7]. In fact the FEM is a collective name for a host of numerical tools to solve PDEs. For an introduction to this subject the reader is referred to [Becker 1981][Carey 1983][Oden 1983][Carey 1984][Hughes 1987].

One may say that the current state of the art of the FEM has evolved to a fairly advanced level in various fields, such as: fluid dynamics, structural engineering, electromagnetics and semiconductor devices. However, in the field of research on transducers these disciplines all come together and a flexible unified approach is needed to deal with it. In such an approach one should be able to deal, at the user's choice, with a number of signal domains (cf. Section 1.3.1) on an equal footing and, moreover, it should also be possible to couple these domains, in order to evaluate the interactions between them. This is an important feature since in transducers it is essential to couple one or more signal domains to achieve transduction of information between them. For instance, when dealing with a thermal acceleration sensor [Hiratsuka 1991] [Hiratsuka 1992], the electrical, thermal and mechanical domains are involved. Another example is the evaluation of offset caused by thermal and mechanical stress in the "spinning-current Hall plate" [Munter 1992]. Yet an-

[5] For example MATHEMATICA, MATLAB or MATHCAD.

[6] For instance 1-D models applied to truly 2-D or 3-D devices.

[7] Other techniques for solving partial differential equations are, for instance, the finite difference method and the boundary element method.

other example is the thermal flow sensor [Oudheusden 1992] which basically operates in the mechanical (fluid flow), thermal and electrical domain.

Although a generic high level interface to such modeling problems is yet out of range, we may represent such a generic modeling and simulation system by the diagram presented in Figure 1-3. The cycle usually starts with a symbolic model formulation in terms of some high-level language. Next, the model is interpreted and as a result the data structures for storing the physical interpretation, the geometric features, the desired results and the required mathematical equations are specified. The cycle then proceeds to a configuration stage, where the program is configured according to the specifications and is subsequently solved. The configuration can either be carried out at run-time or as a separate pre-compilation stage. Especially, the second method is interesting, because it allows the run-time program to be tailored to the problem to be solved. The synthetic data produced by a simulation may be compared with real-world measurements and the differences can be interpreted to adapt some details of the model.

In the next two sections we wish to present some paradigms that can aid in the modeling of solid-state transducers. In Section 1.3.1 we discuss in greater detail the previously mentioned idea of the "signal domains" [Middelhoek 1986][Duyn 1990]. In Section 1.3.2 we exemplify the basic strategy with respect to the physical modeling of solid-state transducers and in Section 1.3.3 the basic strategy with respect to the numerical modeling is explained.

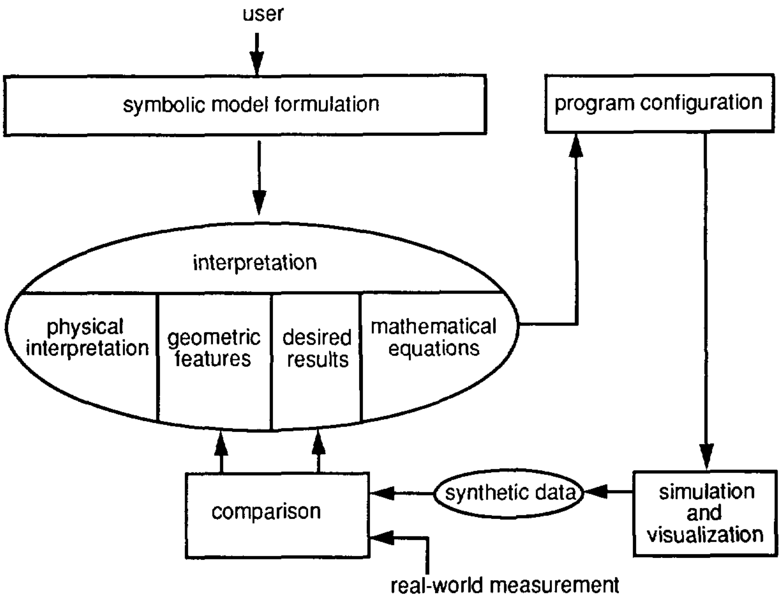

Figure 1-3 Prototype modeling and simulation system.

1.3.1 Energy-Domains

A convenient way of viewing the transduction process taking place in a transducer is obtained by considering several energy domains. Based on physical experience we may roughly distinguish between several macroscopic forms of energy, that is, thermal, chemical, mechanical and electromagnetic energy[8]. For the sake of convenience, one also separates the electromagnetic energy into radiant[9], electrical and magnetic energy. Hence, we have six basic energy domains between which transduction of information can take place. Although the term energy domain seems more appropriate, the name "signal domain", is more commonly used. In the remainder of this work we shall use both terms intermixed.

If we assume that each energy domain can be equipped with a complete set of state variables describing its state, then these state variables are said to represent its information content. From a theoretical point of view, the process of information transduction can now be viewed as a coupling between the states of two (or more) signal or energy domains. Therefore, if a change of state of one energy domain affects the state of the other energy domain, we say that the domains are coupled and that this coupling enables the transduction of information. More specifically, the coupling establishes a functional relationship between the numerical values and the physical representations of the state variables, allowing the information modulated on the numerical values to be transduced into different physical representations. Using the six energy domains, the information transduction process taking place in a transducer can be depicted by the general diagram given in Figure 1-4.

The question remains of how to conveniently define the state variables describing each energy domain and of how to describe the dynamic behavior of the state variables. In the following section we discuss a powerful physical modeling paradigm to answer the above question.

1.3.2 Physical Modeling

A convenient framework for the (phenomenological) physical modeling of solid-state transducers in the space-time domain is the theory of non-equilibrium thermodynamics [Groot 1969][Gyarmati 1970][Duyn 1990]. In principle, non-equilibrium thermodynamics is a serious attempt to link thermodynamics[10] (thermal and chemical energy domain), mechanics (mechanical energy domain) and electrodynamics (magnetic, electrical and radiant domain) into a unified field theory.

[8] This distinction may also be found in basic text books on equilibrium thermodynamics.

[9] Here we refer to a strictly classical interpretation of radiant energy, not including the α and β types of radioactive radiation.

[10] The name thermostatics is more appropriate, however, history has followed its own mysterious ways.

We now give a rather heuristic description of the basic principles of non-equilibrium thermodynamics and see how it fits into the signal or energy domain approach. For more detail the reader is referred to [Groot 1969]. The starting point is the assumption of local equilibrium[11] which allows the use of all thermostatic relations such as the Gibbs relation and the fundamental equations of state. The equations of state describe the relations between the *intensive* and *extensive* state variables and are referred to as the equilibrium thermodynamic equations of state[12]. However, the equilibrium thermodynamic equations strictly refer to the description of thermodynamic equilibrium, or at the most to quasi static or reversible processes. To extend the theory to the case of truly dynamic processes, also called irreversible or dissipative processes, the balance equation for the entropy plays a central role. This equation expresses the fact that the entropy in a volume element of a material changes in time for two reasons. First, it changes because entropy flows into the volume element and, second, because of the entropy production caused by irreversible phenomena inside the volume element. Only in the case of reversible processes will the entropy production be zero, in all other processes it is a non-negative quantity. In fact, the entropy production is a local formulation of the well-known second law of thermostatics.

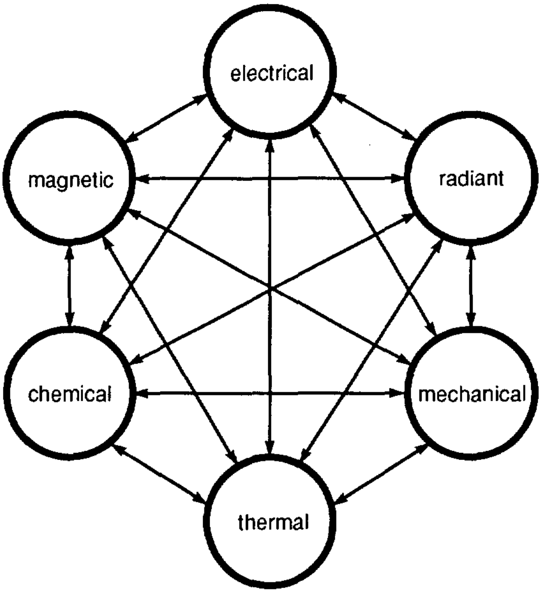

Figure 1-4 Diagram indicating the possible energy conversions in transducers

[11] Whether this assumption is justified is difficult to prove a priori, however, in many practical cases experimental justification has been shown.

[12] For instance, the ideal gas law for the electron gas in a solid.

The aim is to connect the entropy source with the various irreversible processes that occur in a true physical system. One way to proceed in this direction is to set up the macroscopic conservation laws expressing the conservation of mass, momentum and energy in their local form[13]. These conservation laws contain a number of quantities which are related to the transport of mass, momentum and energy. The entropy production can then be calculated using the Gibbs equation from thermostatics which connects, for instance, in an isotropic multi-component fluid, the rate of change of entropy in each volume element to the rate of change of energy and the rates of change in composition. The entropy production has a very simple appearance: it is a sum of terms each of which are the product of a *generalized flux* characterizing an irreversible process and a quantity called a *generalized force*. Moreover, each term can be assigned to one of the six energy domains. The generalized forces are a measure for the non-uniformity of the system or the deviations of the internal state variables from their thermal equilibrium variables. In fact, the generalized forces appear as spatial gradients or local differences of the intensive state variables and the generalized currents appear as the time rates of the extensive state variables [Li 1962].

Just as the thermostatic potential in the case of thermostatics, the entropy production equation then guides us in setting up the non-equilibrium constitutive relations. The simplest choice is to assume linear Markovian[14] relations between the generalized currents and forces. With this choice the entropy production is guaranteed to be a non-negative quantity, as it should be. More complex choices are, of course, possible, but this largely depends on the complexity with which we need to describe the material. One now has at one's disposal a consistent set of partial differential equations governing the dynamic behavior of the state variables of a material system, which may be solved using appropriate initial and boundary conditions.

It has to be emphasized that non-equilibrium thermodynamics does not make any statements about the exact nature of the equilibrium and non-equilibrium constitutive relations. As far as non-equilibrium thermodynamics is concerned, these relations must be obtained experimentally or by using a more fundamental microscopic theory[15]. However, some rather important statements of a macroscopic nature can be made concerning the matrix of phenomenological coefficients which relate the thermodynamic fluxes and forces. First, the Onsager-Casimir reciprocity theorem [Miller 1974] gives rise to a number of relations amongst these coefficients, thus reducing the possi-

[13] In fact, this is what is commonly referred to as the hydrodynamic model which is extensively used in fluid dynamics and recently also in the simulation of special semiconductor devices. However, in the case of "simple" solids some additional simplifications can be made.

[14] In general, a Markov process is a process without memory.

[15] Note that the use of Monte-Carlo simulations have made it possible to estimate the constitutive parameters [Moglestue 1992].

ble number of independent quantities and relating distinct physical effects to each other. Second, the spatial symmetries of the material system may further simplify the scheme of phenomenological coefficients (Curie's theorem).

Referring again to Figure 1-4, we can now view each sphere as representing an energy or signal domain with its intrinsic and extrinsic state variables and its generalized forces and fluxes. Moreover, the dynamic behavior of the state variables in each energy domain is governed by a set of phenomenological equations together with the equilibrium and non-equilibrium constitutive relations. Let us, for example, consider the case of a piezoresistive transducer, realized by means of a thin-film polycrystalline layer, which can be used to convert a strain from the mechanical domain into the electrical domain. The transducer is operated as follows, a constant electrical current is forced through the device and the resulting electrochemical potential difference is measured. Since the mechanical strain (deformation) affects the proportionality factor between the current and the electrochemical potential difference, changes in the mechanical strain can be detected as changes in the electrochemical potential difference. Now let us characterize the state variables in the electrical energy domain. The energy in the electrical domain is proportional to the product of the charge carrier concentration and the electrochemical potential which respectively act as the extensive and intensive state variable. The entropy production is proportional to the product of the electric current and the gradient of the electrochemical potential, which respectively act as the generalized flux and force. The equilibrium constitutive equation is a relation relating the charge carrier concentration to the electrochemical potential. The non-equilibrium constitutive equation is a linear Markovian relation relating the electrical flux to the gradient of the electrochemical potential[16]. The phenomenological equation is given by a balance equation relating the time rate of the charge carrier concentration to the divergence of the electrical flux. One now has at one's disposal a consistent set of partial differential equations governing the dynamic behavior of the state variables which can be solved using appropriate initial and boundary conditions[17].

1.3.3 Numerical Modeling

It is obvious that the space-time continuous mathematical models that emerge from the physically based modeling discussed in the previous sections are not directly suitable for a numerical treatment using a computer. To enable a numerical treatment the mathematical model first needs to be discretized in space as well as time. Subsequently, the resulting discrete mathe-

[16] In fact this is the well known Ohm's law.

[17] Of course the entire set of equations has to obey the Maxwell equations. In this case the Poisson equation, which acts as a compatibility relation, is sufficient.

matical model needs to be complemented with an algorithm for solving the model. Finally, the model must be implemented into a computer program. It is only after these steps that we may perform actual simulations.

With respect to the discretization in space, several methods can be found in the literature, noticeably the finite difference method (FDM), the finite volume method (FVM) [Wesseling 1992], the finite volume element method (FVEM) [McCormick 1989] and the finite element method (FEM) [Hughes 1987]. In the remainder of this work we prefer to use the FEM, because of its advanced status with respect to both the theory and experiment. In some cases the FVEM gives equivalent results and may also be considered as a serious candidate for discretizing. The main difference between the FEM and the FVEM is that the FVEM starts from a physical conservation law in its integral form, whereas the FEM starts from the physical conservation law in its differential form from which a weak (variational) form is obtained by means of the (Petrov)-Galerkin method [Hughes 1987]. In a sense, the finite element discretization can be regarded as more general because a particular finite volume element discretization can always be obtained as a special case of a finite element discretization. In this work we exclusively use the FEM. In particular, we explore the use of the mixed-hybrid finite element discretization method [Arnold 1985] which, besides allowing a more or less independent numerical approximation of the intensive state variables and their corresponding fluxes, also has some additional advantages, to be discussed later.

Besides discretization in space we also need to discretize in time. Again several methods can be found in the literature. Roughly we can distinguish between time-stepping methods and waveform relaxation methods[18]. Of the time-stepping methods we mention the well-known Runga-Kutta method, the forward Euler method, the Crank-Nicolson method and the backward Euler method. The waveform relaxation method is commonly used for the iterative solution of systems of ordinary differential equations and has been used successfully in solving the systems of equations that arise in the simulation of large and complex VLSI devices. Waveform relaxation has the advantage over time-stepping methods that it can be vectorized and parallelized in a rather "trivial" way, moreover, the waveform method does not suffer from error accumulation from previous time steps as in the time-stepping methods. With respect to the time stepping methods, explicit time stepping is fast but is not unconditionally stable, whereas the implicit time differencing and Crank-Nicholson schemes are unconditionally stable but are less fast. Runga-Kutta methods can be very robust, but stability is somewhat difficult to prove and in the context of multigrid (see next page) not recommended [Brandt 1984].

[18] Also called dynamic iteration or Picard-Lindelöf iteration.

If we reconsider the concept of "Multi-Signal Domain Modeling and Simulation of Solid-State Transducers", in particular the simulation part, we should realize that this, in its complete or even partial generality, is a task of considerable complexity demanding quite a lot of computational power. The increasing availability of very fast vector and parallel computers at increasingly lower prizes make these large-scale numerical computations more and more feasible[19]. However, the introduction of brute force computational power alone is rather limited because the reduction in computation time because of faster hardware is "only" a constant factor. Consequently, with an algorithm of $O(n^3)$ complexity[20], the increase in the number of nodes is only proportional to the cubic root of this constant factor. For example, to solve a 2D problem on a uniformly refined mesh in the same time it takes to solve it on the unrefined mesh requires a computer that is no less then 64 times faster. Further, the use of a Newton-Raphson algorithm in the case of a nonlinear problem then also requires 16 times the storage capacity. Effectively this means that if the hardware and the maximum allowed computation time are fixed, the maximum problem size is also fixed. Increasing the problem size by refining the grid, for instance to achieve additional accuracy, then requires a computer that is 64 times faster with 16 times more storage. Therefore, one should also pay considerable attention to the use of more efficient numerical solution methods.

At this time a very promising technique, especially for non-linear elliptic problems, is the multigrid (MG) technique. For an introduction to this subject the reader is referred to [Briggs 1987] and [Wesseling 1992]. The application of MG to a variety of problems is extensively described in [Brandt 1977][Brandt 1979] .[Brandt 1984]. Here we just present a brief outline of the basic idea. The concept of multilevel[21] fast solvers is based upon a certain understanding of the convergence behavior of iterative solution processes such as a simple Gauss-Seidel iteration. In many cases it can be shown that error components with a wavelength of the order of the mesh size, that is, the high frequency components, are efficiently reduced by the iteration process. However, error components with wavelengths much larger than the mesh size are hardly reduced and, consequently, after a few iterations convergence slows down and the asymptotic convergence rate becomes very small. This means that after a few iterations, the error in the approximation is smooth compared to the mesh size. Obviously, the remaining error can be accurately represented and solved on a coarser mesh. Therefore, instead of continuing the relaxation process when, after a few iterations, convergence slows down, one switches over to a coarser mesh. Once an accurate approximation to the

[19] An interesting trend is the use of add-on boards for PC s and work stations offering multi- and vector processing facilities.

[20] For instance, if we use straight forward Gaussian elimination.

[21] The preferred terminology of A. Brandt, one of the pioneers in this field.

error is obtained on this mesh it is used to correct the solution on the fine mesh.

Solving the error on a coarser mesh generally means solving a problem similar to the original one and therefore the same iterative procedure can be used. However, compared to the computational cost of iteration on the finer mesh, the amount of work needed for the solution of the error on the coarse mesh is much smaller. The first reason for this reduction is that the number of elements on the coarser mesh is smaller and consequently one iteration sweep on this mesh requires fewer operations. The second reason is that, because of the larger ratio between wavelength and mesh size, the iterative process on the coarser mesh converges faster and, hence, a given error reduction requires fewer iterations.

The same line of reasoning applies to the error on the coarse mesh. If the number of elements on this mesh is still relatively large, the convergence will slow down again after a number of iterations. The remaining error on this mesh will be smooth and can be accurately approximated and solved on an even coarser mesh. This process can be repeated until a mesh is reached on which the problem can be solved in only a few iterations. The results are then used to correct the solution of the problem on the next finer mesh and so on until the finest mesh is reached again. Hence, to solve the problem on a certain mesh, a sequence of coarser and coarser meshes is used. On each mesh only a few iterations are carried out and only on the coarsest mesh is the problem accurately solved. Usually the number of elements on the coarsest mesh is very small so this problem can be solved with virtually no effort.

The total number of operations needed to solve a problem up to the level of the discretization error[22] with the above-mentioned procedure is generally equivalent to the amount of work of 10 to 25 iterations on the finest mesh, independent of the number of elements used in that grid. Hence, a respectable reduction in the computational cost can be achieved.

1.4 The Objective of the Dissertation

As will be obvious to the reader, the normal duration of a Ph.D. research project is not sufficient to implement the entire scheme as presented above. We, therefore, must confine our global goal to something more comprehensible, however, by conforming as much as possible to the basic principles as set in the previous sections, in order to allow for future extensions. This work and the accompanying software are therefore confined to the modeling and in lesser extend to the simulation of silicon transducers in the thermal and electrical energy domain [Duyn 1992]. It presents a survey of basic models and a mathematical toolkit to tackle these models numerically. The mod-

[22] truncation error ~ discretization error

els are based on the general principles of non-equilibrium thermodynamics providing a modular and extensible framework. The mathematical toolkit is based on the finite element method using mixed-hybrid finite elements and multigrid acceleration techniques. The type of computers we have in mind are work stations with a computing power comparable to a SUN SPARC 2. We do not assume special hardware such as add-on boards offering vector and/or parallel processing facilities.

The results of the work presented should enable full simulation of silicon thermopile sensors, such as the thermal acceleration sensor [Hiratsuka 1991][Hiratsuka 1992]. Also the evaluation of thermal offset causes in the spinning-current Hall plate becomes feasible [Munter 1992]. Obvious future projects are the inclusion of other energy domains, of which the mechanical domain in the form of structural and fluid-flow models is the most interesting from both practical and theoretical point of view.

1.5 The Organization of the Dissertation

Taking the above into consideration, this work is roughly split into two main parts. The first part (Chapters 2 and 3) deals with the construction of a physical model of the thermoelectric effects in the continuous space-time domain. The second part (Chapters 4, 5 and 6) deals with the implementation of the thermoelectric model. This work is, therefore, organized as follows. In Chapter 2, we discuss some generic aspects of physically based system models that can be used to model the (real-time) behavior of solid-state transducer configurations. In particular we discuss the abstract representation of a generic transducer configuration. We also discuss the operation of transducers when viewed from the thermodynamic perspective. Moreover, we discuss the idea of abstract mathematical models as an organizing tool for the representation of model equations. In Chapter 3, a concrete example of the considerations presented in Chapter 2 is discussed, which is the modeling of the thermal and electrical energy domain for the case of a generic semiconductor. The result of this chapter is a rigorous thermoelectric model for semiconductors that can be used for simulation purposes. Note that the general ideas presented in this chapter are not strictly reserved to the thermal and electrical energy domain. In Chapter 4, we discuss the transformation of the continuous space-time model into its discrete equivalent. To achieve this we use the finite element method in combination with mixed-hybrid finite elements which have very attractive properties with respect to the thermoelectric model. The result of this chapter is a fully space-time discrete equivalent of the space-time continuous thermoelectric model. Next, in Chapter 5 we discuss a rather unconventional method for solving these discrete equations numerically. The method is based on the multilevel concept and is capable of solving the discrete model in linear time. Chapter 6 deals with some of the practical implementation issues. Our implementation is based on the object oriented programming method. Finally, in Chapter 7 we give an illustrative examples. Also there are several appendices to introduce the reader to some related subjects he might not be familiar with.

1.6 Some Remarks on the Notation

Before we move on to Chapter 2, it may be instructive to set out some rules on the notation used in this thesis:

- All abbreviations used in the text are explained the first time they are used. An explanatory list of abbreviations is given on page 247.
- For the meaning of all the used symbols, physical as well as mathematical we refer to the list of symbols starting on page 241.
- For the physical models the notation is mainly in cartesian tensor notation, with the notion that tensor subscripts are always in Greek and implicit use of the summation convention is assumed. Occasionally, position and velocity are denoted as $\boldsymbol{r}$ and $\boldsymbol{v}$, to avoid superscripts and subscripts where they are not necessary.
- Partial differentiation is denoted by the symbol ∂; ∂_α denotes differentiation with respect to the position coordinate x_α and ∂_ι denotes differentiation with respect to the time coordinate t. The thermodynamic partial differentials are usually written in full.
- Tensors sampled in space and/or time are denoted as: $\sigma_{\alpha....\beta}(\boldsymbol{r}_m, t_n) = \sigma_{\alpha....\beta}(m, n)$
- For the discrete models we use a combination of tensor and matrix notation. Matrices and vectors resulting from discretizations are always written in upper case, e.g. the matrix A. The individual elements of A are referred to by means of $[A]_{ij}$ or a_{ij}.
- Approximate quantities in the space-time domain $(\boldsymbol{x},t)$ are indicated by the subscripts or superscripts h and z. For example, $u(x, t) \cong u_{h, z}$.

1.7 Software

Most of the material presented in this work is implemented in a C++ class library which was developed by using the public domain GNU G++ compiler in a UNIX SVR4 environment. Note that the library is not a "press a button and go" simulation package, the ultimate aim is rather to provide a well-organized programming interface for fast and efficient prototyping of generic solid-state transducer problems. It is emphasized that at the current revision (1.0) is still an experimental package, because it uses several unconventional but powerful features, such as mixed finite elements, multigrid and a systematic use of the object-oriented programming features of the C++ language. These three topics are research goals in themselves, leaving room for many improvements and fine tuning. The current revision implements the electric and thermal signal domain, and provides simple file based graphical output that can be used by the MATHEMATICA™ program. Likely a future release will incorporate a graphical interface based on the industry standards X (Windows), the Motif Toolkit and 2-D and 3-D graphical output using the PEX library. However, at the present time our focus is more on the physical and numerical aspects.

Simple elements define the main story
Through autumn !

Chapter 2

Generic Aspects of Physically Based Models

2.1 Introduction

The aim of this chapter is to discuss some generic aspects of physically based system models that can be used to model the (real-time) behavior of solid-state transducer configurations. The considerations presented in this chapter follow the basic ideas presented in Chapter 1. We emphasize that in order to manage the complexity of the models, we first should try to create a logical modeling framework, which helps to replaces a vague or amorphous modeling problem with one that is well organized and based on universal and not specific principles. In the terminology of Chapter 1, this stage can be identified as the definition of a *conceptual model* for the specific class of observed or hypothetical physical objects under consideration, in this case a generic solid-state transducer configuration. Such an approach also has the advantage that future extensions to a model, e.g. in the form of additional signal or energy domains[1], can be made with relative ease. The availability of a generic modeling paradigm also provides a powerful tool in dealing with the problem of the organization of the computer programs that implement the model.

As discussed in Chapter 1, our physical modeling paradigm is based on the thermodynamics of irreversible processes (TIP) [Duyn 1990][Gyarmati 1970][Groot 1969]. In other words, we view the solid-state transducer configuration as a physical object that can be described by means of the basic modeling principles from TIP. Unfortunately, TIP is a rather abstract theory and therefore somewhat difficult to grasp at first, however, mastering the principles of TIP, can be rewarding as we show in this chapter. Within the realm of TIP, it is easily possible to develop models way beyond what is commonly accepted as computationally feasible, however, with the tremendous decrease in the price performance-ratio of computing power over the last years,

[1] For example the mechanical domain in the form of the Navier-Stokes equations to model convective cooling effects at device surfaces, or equations describing the deformation of matter due to thermally generated mechanical forces.

it will only be a matter of time before even the more computationally demanding models can be solved in reasonable computing times[2].

The structure of this chapter is as follows. In Section 2.2 we discuss an abstract geometrical representation of a generic transducer configuration. Essentially, the idea is to divide the entire configuration into a number of regularly shaped regions separated by interfaces and boundaries. Each region, interface and boundary is associated with its characteristic set of model equations, based on the notion of the energy domain(s) a region operates in. In Section 2.3 we discuss the generic properties of the class of thermodynamic system models. In Section 2.4 we discuss the generalized mathematical form of the models to be defined on the regions of a configuration. This section is particularly helpful in abstracting the common properties of various apparently different types of models. Certainly, the model equations in the electrical, thermal and mechanical domain conform to this generalized mathematical model. In Section 2.5 and Section 2.6, we respectively discuss the treatment of interface and boundary conditions. Finally, in Section 2.7 some concluding remarks are made.

2.2 Description of the Configuration

In this section, an abstraction of a generic *solid-state transducer configuration* is developed. There are a number of reasons why such an abstraction is necessary. First, abstraction is needed to reduce the complexity of the actual geometrical configuration. For instance, by approximating it with a polygonal surface or polyhedronal volume. We might even consider applying a coordinate transformation, or reducing a two- or three-dimensional configuration to one of lower dimensionality. Second, geometrical abstraction is needed with respect to the binding of appropriate models to the configuration; often different parts of the configuration need different characteristic sets of model equations and parameters to describe the problem adequately.

We shall take the physical domain, occupied by the transducer configuration, as a time-invariant[3], (possibly) multiple-connected region Σ, with a piecewise smooth boundary $\partial\Sigma$. It is assumed that Σ is a bounded portion of a 1D, 2D or 3D universe U, that is, $\Sigma \subset \mathrm{R}^d$ $(d = 1,2,3)$. The interaction of Σ with the remaining part of the "universe" is taken into account by specifying known *boundary conditions* on $\partial\Sigma$. Essentially, this means that any physical modeling problem first has to be reduced to some bounded physical domain for which the boundary conditions are known. The physical domain Σ is then di-

[2] An interesting trend is the use of add-on boards for PCs and work stations, offering multi- and vector processing capabilities.

[3] Time invariance of a physical domain implies that its geometry is not explicitly parametrized by the time coordinate.

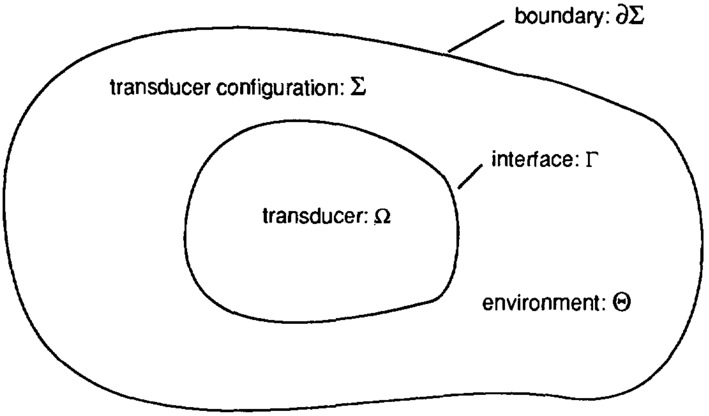
boundary: $\partial\Sigma$
transducer configuration: Σ
interface: Γ
transducer: Ω
environment: Θ

Figure 2-1 An abstract geometrical representation of a generic transducer configuration.

vided into two (possibly) multiple-connected regions Ω and Θ, such that $\Sigma = \Omega \cup \Theta$, where Ω represents the space occupied by the actual solid-state transducing device and Θ represents the space occupied by the environment of the transducing device[4]. The transducer is assumed to interact with its environment (and vice versa) through the interface between regions Ω and Θ. The interface is denoted as Γ and is assumed to be piecewise smooth, moreover, the nature of the interaction is indicated by the specification of interface conditions on Γ. In the simpler case where we already know the boundary conditions on Γ, it is perfectly legal to shrink the environment Θ to zero. In this case the interface Γ simply becomes the boundary of Ω. The above definitions are exemplified in Figure 2-1.

The next step of abstraction is the partitioning of Ω and Θ into the union of a number of disjoint *regions* R^k, separated from each other by *internal interfaces* S_{II}^{kl}, and from the outside world by *boundary interfaces* S_{BI}^{k}, such that[5]

$$\Sigma = \bigcup_{k=1}^{N} R^k \tag{2-1a}$$

$$S_{II}^{kl} = \bar{R}^k \cap \bar{R}^l \quad k, l = 1, \ldots, N \tag{2-1b}$$

$$S_{BI}^{k} = \bar{R}^k \cap (U \backslash \Sigma) \quad k = 1, \ldots, N \tag{2-1c}$$

Obviously, we must have

[4] This separation is not necessary from a computational point of view, however, it does make sense from a conceptual point of view.

[5] Over bars are used to denote the closure of a region.

$$\partial\Sigma = \bigcup_{k=1}^{N} S_{BI}^{k} \qquad \partial R^{k} = \bigcup_{l=1}^{N} S_{II}^{kl} \tag{2-1d}$$

With respect to the complexity of each region, we assume that the geometry is restricted to the use of straight lines in 1D, the use of polygons in 2D, and the use of polyhedrons in 3D. Note that in the context of the finite element method each region in its turn is partitioned (discretized) into a number of elementary geometrical primitives called the elements. Commonly used element types[6] are edges in 1D space, triangles and quadrilaterals in 2D space, and wedges, tetrahedrons and hexahedrons in 3D space. The above definitions leave us with a set of regions, internal interfaces and boundary interfaces.

A commonly used strategy to identify the regions is to divide the configuration into regions of identical chemical phase, that is, the gas, fluid or solid phase. This way, we take into account the solid-fluid, the solid-gas, and the fluid-gas interfaces. In each region so obtained we then look for further regions by examining the jump discontinuities in chemical composition. In the case of the gas and fluid phase this does not make a lot of sense[7], however, for the solid phase it certainly does, for example, the discontinuity at a silicon-oxide interface or at an abrupt P-N junction. This way, the configuration is constructed by patching together several regions, each region representing a different type of material. The purpose is to confine the jump discontinuities in the material parameters to the interfaces between the regions. Usually, these jump discontinuities serve as indicators, where kinks and discontinuities in the global solution can be expected. This can be taken care of by specifying the continuity properties of the state variables, in the form of *interface conditions*, at the internal interfaces. Since within a region the material properties vary continuously with position, it is likely that the entire region can be described by a characteristic set of model equations. In fact this can be seen as another purpose of the partitioning into regions.

The first step, towards the binding of a characteristic set of model equations to each of the regions, is to specify in which signal or energy domain each region operates. In other words, the notion of the type of material, in conjunction with the signal domain(s) the material operates in, may provide some information towards making the choice of a characteristic set of model equations. As discussed in Section 2.3, each signal domain implies a set of state variables, from which the relevant ones can be selected. Hence, each region is characterized by a set of state variables, in terms of which the characteristic set of model equations needs to specified. Moreover, for each internal

[6] An obvious extension is the possibility to use curved geometrical primitives to accommodate curved boundaries and interfaces.

[7] Mixtures of different fluids or gasses have the property to mix very quickly, which precludes the definition of a rigid interface.

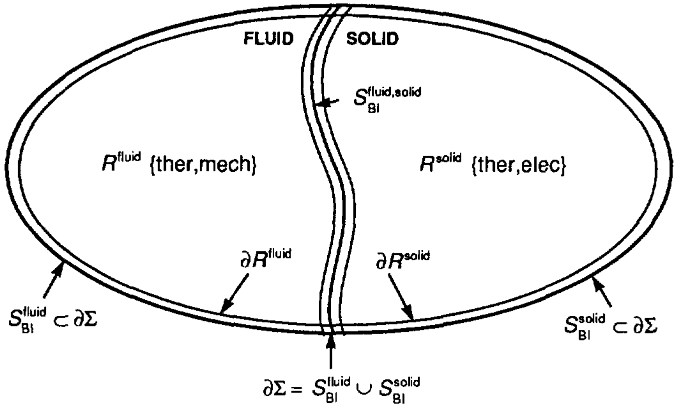
FLUID
SOLID
$S_{BI}^{\text{fluid,solid}}$
R^{fluid} {ther,mech}
R^{solid} {ther,elec}
$\partial R^{\text{fluid}}$
$\partial R^{\text{solid}}$
$S_{BI}^{\text{fluid}} \subset \partial\Sigma$
$S_{BI}^{\text{solid}} \subset \partial\Sigma$
$\partial\Sigma = S_{BI}^{\text{fluid}} \cup S_{BI}^{\text{solid}}$

Figure 2-2 Abstract transducer configuration consisting of two regions: a solid region operating in the thermal and mechanical signal domain, and a fluid region operating in the thermal and electrical signal domain.

interface we must investigate whether a particular state variable is defined at both sides of the interface or only at one side. In the former case interface conditions and, in the latter case, boundary conditions need to be specified at that internal interface. For example, the configuration shown in Figure 2-2 consists of two regions, one region is a fluid R^{fluid} and the other region is a solid R^{solid}. The regions are separated from each other by an internal interface $S_{II}^{\text{fluid, solid}}$. The boundary interfaces are S_{BI}^{solid} and S_{BI}^{fluid}. The fluid region operates in the thermal {ther} and mechanical {mech} signal domain, and the solid operates in the thermal and electrical {elec} signal domain. Note that the region R^{fluid} is closed with respect to the mechanical domain, hence, the boundary $\partial R^{\text{fluid}}$ must be a true boundary with respect to the state variables implied by the mechanical signal domain. This is also the case for the boundary $\partial R^{\text{solid}}$ with respect to the state variables implied by the electrical signal domain. As a result, at interface $S_{II}^{\text{fluid, solid}}$ boundary conditions for the mechanical as well as the electrical state variables need to be specified. However, the thermal state variables implied by the thermal signal domain are present at both sides of the internal interface, hence, in this case interface conditions for the thermal state variables need to be specified.

As a concrete example of the use of the above definitions, we loosely describe the thermal flow sensor, described in [Oudheusden 1992]. The actual transducing device occupies the physical domain Ω and operates in the thermal and electrical domain, moreover, it is a composite of solid materials like silicon, silicon dioxide and aluminum. Obviously, each composite can be represented as separate region. The environment of the transducer is in the gaseous phase and occupies the physical domain Θ, moreover it operates in the mechanical (flow) and thermal domain. The interaction between the transducer and its environment takes place at the interface Γ between Ω and Θ, is of thermal origin (convection and conduction), and must be specified by means of interface conditions. With respect to the electrical signal domain the physical domain Ω is closed, hence, electrical boundary conditions need to be specified at the boundary $\partial\Omega = \Gamma$. With respect to the mechanical signal domain the physical domain Θ is closed, hence, mechanical boundary conditions must be specified at the boundary $\partial\Sigma$. With respect to the thermal signal domain the physical domain Σ is closed, hence, thermal boundary conditions must be specified at the boundary $\partial\Sigma$ and thermal interface conditions need to be specified at the interface Γ.

What we learn from the above example is that essentially we are dealing with a layered problem, where all layers need to be adequately described in terms of their model equations, boundary conditions and interface conditions. Essentially, this is the idea of multi-signal domain modeling. Note that above observations also give sufficient heuristic guidelines with respect to the implementation of the data structures that implement a geometrical model of the transducer configuration, however, this is (in partial form) the subject of Chapter 6 and will not be discussed here. This concludes our description of the transducer configuration.

2.3 Thermodynamic System Models[8]

In the previous section, we discussed the abstraction of the transducer configuration and arrived at a description in terms of a set of regions each linked to a characteristic set of model equations. In this section the emphasis is on the physical modeling paradigm to be employed in the construction of the model equations. In particular, we focus on the thermodynamic modeling paradigm. As such it presents the basis of a general strategy towards the physical modeling of complex processes occurring in solid-state transducers.

In Section 2.3.1, we deal with the equilibrium case and discuss an abstract framework for the description of thermodynamic systems in equilibrium. In Section 2.3.2, we deal with the non-equilibrium case, which deals with an extension of the framework, in order to be able to describe thermodynamic systems not in equilibrium. Finally, in Section 2.3.3 some conclusions are stated.

2.3.1 The Equilibrium Case

This subsection deals with the description of the state of thermodynamic systems in equilibrium. First, the abstract concept of intensive and extensive state variables and their general properties are introduced. Second, two in-

[8] The material presented in this section also appeared, more thoroughly discussed, in [Duyn 1990].

tensive state variables of strict thermodynamic origin are introduced; the temperature and chemical potential. Third, we introduce Gibbs fundamental relation and its properties. Finally, the abstract definitions of the intensive and extensive state variables are linked to well-known physical variables.

2.3.1.1 Intensive and Extensive State Variables

Let us first consider a closed[9] system with an adiabatic boundary[10], which is in thermodynamic equilibrium. We assume that, due to the external work done, the system is subjected to a reversible process. During the process the work done on the system is reflected by a change of the energy stored in the material of the system. Since, by definition, in a reversible process no work is converted into heat, the stored energy should be independent of the path followed by the process to get from state A to B. In this case the stored energy may, in analogy to conservative forces in mechanics[11], be written as a potential function

$$U = U(E^1, \ldots, E^N) \tag{2-2}$$

where U is the stored or internal energy and the E^i the *extensive state variables* or work coordinates. The change in energy, when an infinitesimal trajectory of the process is traversed, can be written as a proper total differential

$$\mathrm{d}U = \sum_{n=1}^{N} I^n \mathrm{d}E^n \qquad \text{with} \qquad I^n = \frac{\partial U}{\partial E^n} \tag{2-3}$$

where the I^n are called the *intensive state variables* and are defined as the partial differentials of the internal energy function. The above equation is commonly referred to as the first law of thermodynamics.

Before we can start identifying the physical nature of the intensive and extensive state variables, we need to introduce two special intensive state quantities which are of strictly thermodynamic origin; the temperature and the chemical potential.

2.3.1.2 Absolute Temperature

To introduce the notion of temperature we must remove the restriction of the adiabatic boundary. In this case we also must deal with the possibility that energy, in the form of heat, enters or leaves the system through the boundary[12]. Let us write the change in the internal energy as

[9] A closed system has a boundary that does not permit mass to cross it.

[10] An adiabatic boundary by definition does not allow heat to pass.

[11] $dW = \boldsymbol{F}\cdot d\boldsymbol{r} = \phi(r + dr) - \phi(r) = (\partial\phi/\partial r)\cdot dr$.

[12] We still assume the process itself does not convert work into heat.

$$\mathrm{d}U = \mathrm{d}Q + \mathrm{d}W \tag{2-4}$$

This means that the change in internal energy consists of a change in the thermal energy dQ plus a change dW because of the work done on the system. Unfortunately, experience has shown that dU, as expressed in equation (2-4), does not behave as a proper total differential[13], instead, it is commonly referred to as a Pfaffian form [Münster 1970]. Pfaffian forms can be integrated along a path in $\{E^i\}$ space but the value of the integral, in general, depends on the path traversed. The question then is how to solve this problem. In the axiomatic theory of thermodynamics [Münster 1970] it is shown that, based on an experience of fact, an integrating factor should exist such that

$$\mathrm{d}S = \frac{\mathrm{d}Q}{T} \tag{2-5}$$

where T is the absolute thermodynamic temperature and S the entropy of the system. In contrast to dQ, the behavior of dS is that of a proper total differential, hence, the change in internal energy is again independent of the path traversed by the process in $\{S,E^i\}$ space, and may again be written as a potential function. We may now write the internal energy and its total differential as

$$U = U(S, E^1, \ldots, E^N) \tag{2-6a}$$

$$\mathrm{d}U = T\mathrm{d}S + \sum_{n=1}^{N} I^n \mathrm{d}E^n \quad \text{with} \quad T = \frac{\partial U}{\partial S} \tag{2-6b}$$

2.3.1.3 Chemical Potential

Up to now we have only dealt with closed systems without chemical reactions. The extension of thermodynamic system theory to also include chemical reactions was first brought forward by the famous J.W. Gibbs. According to Gibbs we may write the internal energy of an open system with M chemical components[14]

$$U = U(S, N^1, \ldots, N^M, E^1, \ldots, E^N) \tag{2-7a}$$

$$\mathrm{d}U = T\mathrm{d}S + \sum_{m=1}^{M} \phi^m \mathrm{d}N^m + \sum_{n=1}^{N} I^n \mathrm{d}E^n \tag{2-7b}$$

[13] Although dQ and dW add together to give dU, there are no quantities Q nor W which have separate meanings.

[14] The line of reasoning does not necessarily hold only for true chemical components, but also for the distribution of electrons over available energy states.

where ϕ^k is the chemical potential of component k and N^k the number of particles in component k. Note that ϕ^k may be viewed as the average energy per particle in component k.

2.3.1.4 Gibbs Fundamental Relation

Equation **(2-8a)** is commonly referred to as Gibbs' fundamental relation in the energy representation. According to the postulates of thermodynamics Gibbs' fundamental relation is a homogeneous, linear, single-valued, continuous and differentiable function of the extensive state variables, in particular it is a monotonic function of the entropy S. We may, therefore, invert equation **(2-8a)** to obtain a form where the entropy is on the left-hand side of the equation, this is referred to as Gibbs' fundamental relation in the entropy representation

$$S = S(U, N^1, \ldots, N^M, E^1, \ldots, E^N) \qquad \textbf{(2-8a)}$$

$$\mathrm{d}S = \left(\frac{1}{T}\right)\mathrm{d}U - \sum_{m=1}^{M}\left(\frac{\phi^m}{T}\right)\mathrm{d}N^m - \sum_{n=1}^{N}\left(\frac{I^n}{T}\right)\mathrm{d}E^n \qquad \textbf{(2-8b)}$$

When a complete set of extensive state variables for the system under consideration is known, the Gibbs relation, in the energy as well as in the entropy representation, completely describes the system. Effectively, this means that any possible statement to be made about the system can be deduced from the fundamental equation.

Perhaps the most important (postulated) property of the Gibbs relation in the entropy representation (equation **(2-8a)**) is that it has a well-defined extremum (maximum), towards which the internal state of an isolated system tends to be driven. At the extremum the system is said to be in thermodynamic equilibrium. The evolution towards the extremum is expressed by the second law of thermodynamics

$$\frac{\mathrm{d}S}{\mathrm{d}t} \geq 0 \qquad \textbf{(2-9)}$$

which expresses the fact that the entropy in the system is an increasing quantity during the evolution of the system towards the equilibrium state. Effectively, this means that we can calculate the equilibrium state of a system by calculating the extremum of the fundamental relation [Kirschner 1991][Duyn 1990][Münster 1970].

2.3.1.5 Equilibrium Equations of State

Obviously, the intensive state variables can be expressed in terms of the extensive state variables by means of the equilibrium equations of state, that is

$$I^k = I^k(U, N^1, \ldots, N^M, E^1, \ldots, E^N) \tag{2-10a}$$

where the symbol I^k may stand for either the inverse of the temperature, the ratio of the chemical potentials and the temperature, or the ratio of some other intensive state variable and the temperature. Since the differential form of the fundamental relation **(2-8b)** is a proper differential, the equations of state must satisfy the following reciprocal relations

$$\frac{\partial I^i}{\partial E^j} = \frac{\partial I^j}{\partial E^i} \tag{2-11}$$

These are the reciprocal relations for the intensive and extensive state variables. In differential form the equations of state can conveniently be represented in matrix form

$$\begin{bmatrix} \partial I^1 \\ \vdots \\ \partial I^K \end{bmatrix} = \begin{bmatrix} a^{11} & \ldots & a^{1K} \\ \vdots & & \vdots \\ a^{K1} & \ldots & a^{KK} \end{bmatrix} \cdot \begin{bmatrix} \partial E^1 \\ \vdots \\ \partial E^K \end{bmatrix} \tag{2-12}$$

with the entries of the effect matrix defined as

$$a^{ij} = \left(\frac{\partial I^j}{\partial E^i}\right)_0 \tag{2-13}$$

Clearly, according to relation **(2-11)** the "effect" matrix relating the intensive and extensive state variables must be symmetric. Moreover, the "effect" matrix also has to be negative definite, as can be observed from the following argument. If the fundamental relation in the entropy representation has an extremum (maximum) we may expand it in a Taylor series around the equilibrium state, that is

$$\mathrm{d}S = \frac{1}{2}\sum_i \sum_j a^{ij} \mathrm{d}E^i \mathrm{d}E^j \le 0 \tag{2-14a}$$

where

$$a^{ij} = \left(\frac{\partial^2 S}{\partial E^i \partial E^j}\right)_0 = \left(\frac{\partial I^j}{\partial E^i}\right)_0 \tag{2-14b}$$

Clearly, because of the extremum property, the change in entropy must be negative for a virtual deviation from the equilibrium state. However, this proves the negative definiteness of the coefficient matrix given in equation **(2-14b)**, moreover, since this coefficient matrix is identical to the effect matrix given in equation **(2-13)** it also proves the negative definiteness of the "effect" matrix. Specific examples of the effect matrix are given in Chapter 3.

2.3.1.6 Additional State Variables

Having described the general form of the Gibbs equation, let us now turn to the identification of some additional intensive and extensive state variables. Loosely speaking, extensive state variables can be identified as physical quantities which obey some form of a conservation law, whereas the intensive state variables are usually associated with some form of contact equilibrium[15]. From our general knowledge of mechanics and electrodynamics we may readily identify a number of pairs of intensive and extensive state variables that conform to the above theory [Münster 1970]. In the following, we shall associate each pair with an appropriate signal domain. Let us write equation **(2-8b)** as a sum of energies (reversibly) stored in the signal domains by means of a quasi-static process, that is

$$\mathrm{d}u = \mathrm{d}u^{TH} + \mathrm{d}u^{CH} + \mathrm{d}u^{ME} + \mathrm{d}u^{EL} + \mathrm{d}u^{MA} + \mathrm{d}u^{RA} \qquad \textbf{(2-15a)}$$

where u is the total internal energy density[16], u^{TH} the thermal energy density, u^{CH} the chemical energy density, u^{ME} the mechanical energy density, u^{EL} the electrical energy density, u^{MA} the magnetic energy density and u^{RA} the electromagnetic energy density[17]. As stated before, the thermal and chemical energy domains are characterized by

$$\mathrm{d}u^{TH} = T\mathrm{d}s \qquad \textbf{(2-15b)}$$

$$\mathrm{d}u^{CH} = \sum_k \phi^k \mathrm{d}n^k \qquad \textbf{(2-15c)}$$

The mechanical energy domain is characterized by the mechanical deformation energy stored in matter. It is given by the tensor product of the mechanical stress tensor $\sigma_{\alpha\beta}$ and the mechanical strain tensor $\epsilon_{\alpha\beta}$, that is

$$\mathrm{d}u^{ME} = \sigma_{\alpha\beta}\mathrm{d}\epsilon_{\alpha\beta} \qquad \textbf{(2-15d)}$$

The electrical energy domain is characterized by the electrostatic[18] energy stored in matter, and follows from the Maxwell equations. It is given by a

[15] The term contact equilibrium must be interpreted rather loosely, because it may imply a spatial form of contact equilibrium between two systems, it may also imply a local form of contact equilibrium between two coexisting systems, or it may even imply the deviation of an intensive state variable from its rest value.

[16] Note that in this case we have used quantities referred to the unit volume, which is allowed because the Gibbs relation was assumed to be a homogeneous function of first degree.

[17] We only consider radiant energy implied by the Maxwell equations.

[18] As long as the relaxation effects can be neglected, we speak of electrostatic energy.

sum of the products of the electrostatic potential φ and the electric charge density ξ^k of each (chemical) species k, that is

$$du^{EL} = \sum_k \varphi d\xi^k \qquad \textbf{(2-15e)}$$

From a conceptual point of view, the magnetic energy domain is somewhat difficult to introduce because we would expect it to be the dual of the electrical energy domain, however, this is not the case[19]. Formally, the magnetic energy domain is characterized by the magnetostatic[20] energy stored in matter, and follows from the Maxwell equations. It is given by a sum of products of the magnetic vector potential a_α and the microscopic magnetic-polarization current[21] density ξ_α^k for each species k, that is

$$du^{MA} = \sum_k a_\alpha d\xi_\alpha^k \qquad \textbf{(2-15f)}$$

The Maxwell part of the radiant domain is characterized by the electromagnetic energy[22] stored in matter, and follows from the Maxwell equations. It is described by the product of the electric field strength e_α and the electric polarization density p_α, plus the product of the magnetic field strength h_α and m_α the magnetic polarization density, that is

$$du^{EM} = e_\alpha dp_\alpha + h_\alpha dm_\alpha \qquad \textbf{(2-15g)}$$

Note 2-1: When the chemical potential refers to a species consisting of charged particles, say the electrons, then we do not consider the chemical domain as a separate entity for that species. Instead, the so called electrochemical potential is defined, and as a result the electrical and chemical domain are merged. This matter is more explicitly discussed in Chapter 3.

Note 2-2: The magnetic domain is in fact already implied by the radiant domain, because the notation used in equation **(2-15f)** is just a dual formulation of the notation used for the magnetic part in equation **(2-15g)**. It depends on the particular case we are studying whether we want to explicitly consider a magnetic domain, or just leave it embedded in the radiant domain. We might also choose to extract only that part of the magnetostatic energy

[19] A magnetic domain dual to the electrical domain can only be defined in the case that magnetic mono poles exist.

[20] As long as relaxation effects can be neglected, we speak of magnetostatic energy.

[21] This is not a macroscopic current but it is of microscopic origin, e.g. electrons evolving round the nucleus of an atom. Also macroscopically observable currents can be magnetically polarized (Hall effect), however, in this case we enter the realm of irreversible thermodynamics, to be discussed in the next section.

[22] We refer strictly to the microscopic electric and magnetic polarization energy stored in each of the species.

that can be ascribed to the free carriers. In the following section we see that it is possible to define a separate magnetic domain when the system is not in equilibrium.

Note 2-3: The effect matrix, in the differential form of the equilibrium equations of state (cf. equation (2-12)), with the entries according to the above results, reflects the various possible equilibrium physical effects which are in existence. In particular, the diagonal entries represent the direct effects taking place within a signal domain, and the off-diagonal entries represent the cross effects taking place between signal domains.

2.3.2 The Non-Equilibrium Case

2.3.2.1 Local Equilibrium

The above line of reasoning is, in principle, only valid for systems in thermodynamic equilibrium or systems subjected to a quasi-static process[23]. What is needed is a framework that also can handle non-equilibrium situations. True non-equilibrium processes are characterized by the fact that during the process entropy is produced, or equivalently, heat is generated. More specifically, when a system is subjected to a process in $\{E^i\}$ space, some of the work done on the system is irreversibly transformed into heat (dissipation). In analogy with mechanics, the "forces" driving the process are non-conservative and as a result the change in internal energy now depends on the path traversed in $\{E^i\}$ space.

In order to describe non-equilibrium states we need to find the entropy production. At least we should identify the general properties of this function, just as we identified the general properties of the fundamental Gibbs relation in the previous section. This is where the ideas of Non-Equilibrium Thermodynamics or Irreversible Thermodynamics enter the picture. The key idea in obtaining the entropy production is to realize that equations (2-8a) and (2-8b) are valid for any subsystem which is internally in thermodynamic equilibrium. Dividing the system into a large number of small subsystems gives us a possibility to describe non-equilibrium systems through the assumption of *local equilibrium*[24] [Groot 1969]. In the spatial domain, the local equilibrium assumption can approximately be assured by requiring the subsystems to be sufficiently small[25]. The internal relaxation time of each subsystem may then be neglected with respect to the time needed to reach equilibrium with respect to its nearest neighbors. In the case of coexisting subsystems, such as

[23] In a quasi-static process, it is assumed that the rate of the process is very slow. Since dissipation is proportional to the rate of the process, we may assume that for such processes negligible dissipation occurs, hence, the system may be assumed to be in thermodynamic equilibrium during the entire process.

[24] This is often called the hypothesis of cellular equilibrium. It has the important consequence that all thermodynamic relations and statements are considered valid for each subsystem.

the electron and the hole system, the situation is more complex but often similar assumptions can be made [Landsberg 1991]. In general, however, it is not possible to give a decisive answer as to whether subsystems can be assumed to be in internal thermodynamic equilibrium, the justification usually has to be verified afterwards by experimentation. Nevertheless, the above assumption is vital to the description of many aspects of the physical operation of solid-state transducers.

2.3.2.2 Thermodynamic Flux and Force State Variables

The assumption of local equilibrium makes it possible to use the equilibrium thermodynamic statements within each subsystem. In particular, it can be assumed that in each subsystem a fundamental Gibbs relation holds. This assures that each subsystem is thermodynamically stable by itself and that all intensive and extensive state variables have a well-defined meaning, except for the fact that they now have become field variables[26]. In particular, the extensive state variables are now treated as specific (volume density) quantities[27]. Clearly, the assumption of local equilibrium shifts the concept of non-equilibrium states to the local and spatial differences in the intensive state variables, and the local deviations of the intensive state variables from their equilibrium values. By local differences we mean the differences in intensive state variables between two coexisting subsystems, for instance, the difference in the chemical potential of two chemical species. By spatial differences we mean the difference in an intensive state variable between two neighboring subsystems. We now proceed by introducing the generalized thermodynamic fluxes and forces describing the non-equilibrium state.

Since the dissipation or entropy production should be proportional to the rate of change of the state of the system, it is obvious that we first should look at the rate of change of the various (local) extensive state variables. We first notice that in many cases the time rate of change of the extensive state variables can be related to *generalized scalar, vectorial and tensorial thermodynamic fluxes* by means of balance equations[28]. In the simple case of a scalar extensive state variable we may write

$$\partial_t e^n = -\partial_\alpha e^n_\alpha + \sum_m \dot{e}^{n \leftarrow m} \tag{2-16}$$

[25] On the other hand we cannot choose a subsystem too small because its macroscopic properties are determined by averaging of the microscopic properties. The averages tend to become less well defined for smaller and smaller subsystems.

[26] To indicate this, the field variables are written in lower case.

[27] This is allowed because the Gibbs relation is a homogeneous function of first degree.

[28] Certainly, in the thermal, electrical and mechanical domain this is possible, however, the Maxwell equations in the radiant domain are somewhat stubborn and need special treatment [Groot 1969] (Chapter 13).

where e^n_α is the spatial vectorial flux and $\dot{e}^{n\leftarrow m}$ the local scalar flux (of the m'th reaction) associated with e^n. In the case of a vectorial extensive state quantity we may write

$$\partial_t e^p_\alpha = -\partial_\beta e^p_{\alpha\beta} + \sum_m \dot{e}^{p\leftarrow m}_\alpha \qquad \text{(2-17)}$$

where $e^p_{\alpha\beta}$ is the tensorial spatial flux and $\dot{e}^{p\leftarrow m}_\alpha$ the local vectorial flux (of the m'th reaction) associated with e^p_α. Similarly, we can also write down a balance equation for general tensorial extensive state quantities.

The next step is to define the *generalized scalar, vectorial and tensorial thermodynamic forces*. The only "measure" of non-equilibrium we have at this stage are the local and spatial differences of the intensive state variables. It is therefore logical to take these as the thermodynamic forces. In the case of the balance equation (2-16), we can write the forces associated with the spatial vector flux and local scalar flux as

$$f^n_\alpha = \partial_\alpha i^n \qquad \text{(2-18a)}$$

$$f^{n\leftarrow m} = i^m - i^n \qquad \text{(2-18b)}$$

In the case of the balance equation (2-16), we can write the forces associated with the spatial tensorial flux and local vectorial flux as

$$f^m_{\alpha\beta} = \partial_\alpha i^m_\beta \qquad \text{(2-18c)}$$

$$f^{p\leftarrow m}_\alpha = i^m_\alpha - i^p_\alpha \qquad \text{(2-18d)}$$

Although not needed for our present objectives, it is sometimes convenient to define a thermodynamic force as the difference between the equilibrium value and the actual value of an intensive state variable. In this case, the associated flux is simply the time rate of the extensive state variable. Having introduced the notion of generalized fluxes and forces we turn to the problem of identifying the relations between these.

2.3.2.3 Entropy Production

Just as in the equilibrium case, we may postulate the existence of a characteristic function σ, called the entropy production function, which is a function of the generalized thermodynamic forces only, that is

$$\sigma = \sigma(f^1, \ldots, f^K) \qquad \text{(2-19)}$$

where the f^k are the generalized thermodynamic forces, as discussed in the previous section. Clearly, the above function must vanish when the forces vanish. Just as in the case of the Gibbs relation, we postulate that the dissipation function has the important property of having a well-defined extremum (minimum), towards which the non-equilibrium state tends to be driven in the case the constraints on the system are such that it cannot reach the equilibrium state. At the extremum the system is said to be in a stable stationary state. Note that this statement is true for a large class of systems, amongst the ones we will study later on in this chapter, however, there are also real life systems which do not comply to this property [Li 1962]. Later on in this chapter we show that the above property yields models which in mathematical language are called elliptic problems[29]. The evolution of the state of the system towards the extremum is expressed by the relation

$$\frac{\mathrm{d}\sigma}{\mathrm{d}t} \leq 0 \tag{2-20}$$

The above relation expresses the fact that the entropy production or dissipation in the system is a decreasing quantity during the evolution of the system towards the stable stationary state. Effectively, this means that we can calculate the stable stationary state of a system by calculating the extremum of the dissipation function.

The change in the dissipation function, when an infinitesimal trajectory of the (irreversible) process is traversed, can be written as

$$\mathrm{d}\sigma = \sum_{k=1}^{K} j^k \mathrm{d}f^k \qquad \text{with} \qquad j^k = \frac{\partial\sigma}{\partial f^k} \tag{2-21}$$

The j^k are the generalized thermodynamic forces and are defined as the partial differentials of the dissipation function. At first sight, it might seem a bit strange that the partial differentials of σ are equal to the thermodynamic fluxes. However, [Li 1962] has shown, that in the "simple" case of scalar extensive state variables with scalar associated fluxes and forces the following relation indeed holds

$$\mathrm{d}\sigma = \sum_k j^k \mathrm{d}f^k \leq 0 \tag{2-22}$$

Note that by definition σ is a potential function, hence, $\mathrm{d}\sigma$ is a proper total differential. This means that the following relations must hold

$$\frac{\partial j^k}{\partial f^l} = \frac{\partial j^l}{\partial f^k} \tag{2-23}$$

In fact these are Onsager's reciprocal relations between the generalized thermodynamic fluxes and forces. We might also reverse the argument, that is,

[29] Elliptic mathematical models have special features we can make use of.

the experimental evidence of the validity of the Onsager relations proves the existence of the entropy production. For a large class of systems, very convincing evidence for this assumption can be found in [Miller 1974].

2.3.2.4 Non-Equilibrium Equations of State

In analogy with the definition of the equilibrium equations of state in equation **(2-10a)** we may now, based on equation **(2-22)**, define the non-equilibrium equations of state as

$$j^k = j^k(f^1, \ldots, f^m) \qquad \textbf{(2-24)}$$

In differential form these can be represented as

$$\begin{bmatrix} \mathrm{d}j^1 \\ \vdots \\ \mathrm{d}j^K \end{bmatrix} = \begin{bmatrix} a^{11} & \ldots & a^{1K} \\ \vdots & & \vdots \\ a^{K1} & \ldots & a^{KK} \end{bmatrix} \cdot \begin{bmatrix} \mathrm{d}f^1 \\ \vdots \\ \mathrm{d}f^K \end{bmatrix} \qquad \textbf{(2-25)}$$

with the entries of the non-equilibrium effect matrix defined by

$$a^{ij} = \left(\frac{\partial j^j}{\partial f^i}\right)_0 \qquad \textbf{(2-26)}$$

The non-equilibrium effect matrix, reflects the various possible non-equilibrium physical effects which can occur. In particular, the diagonal entries represent the direct effects taking place within a signal domain, and the off-diagonal entries represent the cross effects between the signal domains. Clearly, by virtue of the previous assumptions the non-equilibrium effect matrix, relating the generalized fluxes and forces, is symmetric and positive definite (the derivation is similar to the one followed in Section 2.3.1.5).

2.3.3 Conclusions

Perhaps not so obvious at first sight, however, the balance equations together with the equilibrium and non-equilibrium equations of state do constitute a closed computational model. This leaves us the task of identifying the balance equations and equations of state for a particular configuration we want to study. In Chapter 3, we will present a rigorous treatment of the thermal and electrical domain, which yields a computational model that can be used to simulate the behavior of semiconductors in the thermal and/or electrical domain. However, I do wish to emphasize that the application of the theory is not restricted to this particular case. Space and time limitations make it impossible to go beyond the present objective.

In order to somewhat lighten the future task of incorporating other signal domains we conclude this section with Table 2-1, which heuristically indicates the main state quantities in each signal domain.

Table 2-1 The main state quantities in each signal domain.

	chem.	elec.	magn.	therm.	mech.	rad.
intensive	ϕ^k	φ	a_α	T^k	$\sigma_{\alpha\beta}$	e_α, b_α
extensive	n^k	ξ	ξ_α	s^k	$\in_{\alpha\beta}$	p_α, m_α
force	$\partial_\alpha \phi^k$	$\partial_\alpha \varphi$	$\partial_\beta \in_{\alpha\beta\gamma} a_\gamma$	$\partial_\alpha T^k$	$\sigma^{eq}_{\alpha\beta} - \sigma_{\alpha\beta}$	$e^{eq}_\alpha - e_\alpha$
	$\phi^k - \phi^l$			$T^k - T^l$		$h^{eq}_\alpha - h_\alpha$
flux	n^k_α	ξ_α	h_α	s^k_α	$\partial_t \in_{\alpha\beta}$	$\partial_t p_\alpha$
	$\dot{n}^k$			$\dot{s}^k$		$\partial_t m_\alpha$

In the chemical domain we have (from top to bottom) the chemical potentials, the particle densities, the gradients and local differences of the chemical potentials, and the spatial and local particle fluxes. In the electrical domain we have the electrostatic potential, the charge density, the gradient of the electrostatic potential, and the current density. In the magnetic domain we have the magnetic vector potential, the electric current density, the rotation of the magnetic vector potential, and the magnetic field strength. Note that this interpretation of the magnetic domain refers to the case of macroscopic electrical currents (Eddy currents). In the thermal domain we have the temperature, the entropy, the gradient of the temperature, the entropy flow, and the entropy production. In the mechanical domain we have the stress tensor, the strain (deformation) tensor, the difference between the equilibrium stress and the actual stress, and the time rate of the strain. In the radiant domain we have the electric and magnetic field strength, the electric and magnetic polarization, the difference between the equilibrium {electric,magnetic} field strength and the actual {electric,magnetic} field strength, and the time rates of the electric and magnetic polarization.

2.4 Generalized Mathematical System Models

In the previous section we discussed a physical modeling paradigm that provided us with some general insights into the construction of models, to be linked to each of the regions in the configuration. In this section the focus is on the formal mathematical representation of the model equations. In particular, we emphasize the similarities between apparently different models.

As discussed in the previous sections, each region of the configuration has to be linked to a set of model equations, in the form of interior, interface and boundary models. In our case we deal with models in the form of partial differential equations (PDEs).

A fact often overlooked is that in most physical disciplines, such as electricity, magnetism, fluid flow and heat flow, the various governing physical models show striking mathematical similarities. Hence, in order to reduce the effort needed to translate the various models into computational algorithms we should try to exploit these similarities. Following the ideas of [Tonti 1972][Penman 1986] we, therefore, develop a strategy based on the notion that a large class of physical models, if not all, can be expressed in a canonical form. A formulation of the canonical model, expressing its mathematical properties, can be given as follows [Penman 1986]

$$\left.\begin{aligned} &T(x,t)\cdot u(x,t) = v(x,t) - v_s(x,t)\\ &M(x,t)\cdot v(x,t) = w(x,t)\\ &T^a(x,t)\cdot w(x,t) = p(u)\end{aligned}\right\}\ (x,t)\in\Omega\times[0,T]$$

$$\begin{aligned} B(x,t)\cdot u(x,t) &= g(x,t) \qquad && (x,t)\in\partial\Omega_D\times[0,T]\\ B^+(x,t)\cdot w(x,t) &= h(x,t) && (x,t)\in\partial\Omega_N\times[0,T]\end{aligned} \tag{2-27}$$

The above equation is called the primal canonical form of the space-time problem in the space-time variables u, v, p and w. Although [Penman 1986] does not explicitly say so, the (full) canonical form of a model implicitly relates the intensive, extensive, generalized flux and force state variables (cf. Section 2.3). The operators T, T^a and M describe the interior properties of the domain and the operators B and B^+ describe the boundary conditions at the Neumann $\partial\Omega_N$ and Dirichlett $\partial\Omega_D$ parts of the boundary (cf. Section 2.2). It is important to observe that the operators T^a and B^+ are the adjoint operators of resp. T and B (for an appropriate definition of adjoint operators consult [Penman 1986]). This feature can be important in the discretization of the model equations (cf. Chapter 4). Commonly encountered adjoined operator pairs are given by

$$\begin{aligned} T &= -\nabla \qquad & T^a &= \nabla\cdot\\ T &= \nabla\times & T^a &= \nabla\times \end{aligned} \tag{2-28a}$$

$$\begin{aligned} B &= I \qquad & B^+ &= -n\cdot\\ B &= -n\times & B^+ &= -n\times \end{aligned} \tag{2-28b}$$

It can also be shown that equation (2-27) can be rewritten in a completely equivalent form called the dual canonical form. The dual canonical form can, for instance, be used to obtain a different type of discretization (cf. Chapter 4). For more details on this subject the reader should consult [Penman 1986].

Following the terminology of [Penman 1986], the relations appearing in equation (2-27) are usually identified (from top to bottom) as the:

- Compatibility relation
- Constitutive relation
- Equilibrium relation
- Dirichlett boundary condition
- Neumann boundary condition

In the terminology of thermodynamics of irreversible processes the equilibrium relations[30] can be associated with the phenomenological equations (cf. Section 2.3), which usually take the form of a balance or conservation law (either vectorial or tensorial). The compatibility and constitutive relations can be associated with the non-equilibrium thermodynamic equations of state. If the physical domain Ω on which the model is defined contains an interface between two different materials, we should be aware of the continuity properties of the relevant physical state variables at these interfaces. For such cases we must add interface conditions to the problem definition given in equation **(2-27)**. In Chapter 4, we show that for a particular problem we can (locally) choose the discretization such that the interface conditions are automatically satisfied for that problem.

Example 2-1: Consider the set of model equations for solving the electrostatic state variables on a domain Ω with boundary $\partial\Omega = \partial\Omega_D \cup \partial\Omega_N$, and interface Γ^{AB} connecting the regions R^A and R^B. Assuming that both regions can be linked to the same characteristic set of model equations, we may write the model as

$$\begin{aligned}
&\left.\begin{aligned} \partial_\alpha d_\alpha &= \xi \\ \varepsilon_{\alpha\beta} e_\beta &= d_\alpha \\ -\partial_\alpha \varphi &= e_\alpha \end{aligned}\right\} x \in \Omega \\
&\varphi = g \qquad\qquad x \in \partial\Omega_D \\
&n_\alpha \cdot d_\alpha = h \qquad x \in \partial\Omega_N
\end{aligned} \tag{2-29a}$$

where $\boldsymbol{n}$ is the normal vector at position x on the boundary. The interface conditions to be added follow from the Maxwell equations and can be stated as

$$\begin{aligned}
n_\alpha \cdot \, [d_\alpha|_A - d_\alpha|_B] &= 0 \quad x \in \Gamma^{AB} \\
\epsilon_{\alpha\beta\gamma} n_\beta [e_\gamma|_A - e_\gamma|_B] &= 0 \quad x \in \Gamma^{AB}
\end{aligned} \tag{2-29b}$$

where $\boldsymbol{n}$ is the normal vector at position x on the interface. In equation **(2-29a)**, from top to bottom, we have the equilibrium relation relating the divergence of the dielectric flux density $\boldsymbol{d}$ to the space charge density ξ, the con-

[30] Note that the term equilibrium relation is somewhat confusing because it implies that the time dependence is not taken into account, which is not the case.

stitutive equation relating the electric field strength $\boldsymbol{e}$ to the dielectric flux density $\boldsymbol{d}$, the compatibility relation, relating the gradient of the electrostatic potential φ to the electric field strength $\boldsymbol{e}$ and two relations representing the Dirichlett and Neumann boundary conditions. Equation **(2-29b)** expresses the fact that the normal component of the dielectric flux density $\boldsymbol{d}$ and the tangential component of the electric field strength $\boldsymbol{e}$ must be continuous when crossing the interface. In thermodynamic terms, φ is an intensive state variable, ξ is an extensive state variable, $\boldsymbol{e}$ is the thermodynamic force and $\boldsymbol{d}$ is the thermodynamic flux.

Let us now pay a little more attention to the generic properties of the operators involved. For any particular problem we must identify the phenomenological operators: T, B and M of the model. This largely depends on the physical modeling paradigm used. For instance, the thermodynamic method leads to typical operators, which we encounter in Chapter 3. With respect to the constitutive operator M, the following classification can be used [Hoop 1986],

- linear vs. non-linear
- homogeneous vs. inhomogeneous
- isotropic vs. anisotropic
- locally vs. non-locally reacting
- Markovian[31] vs. non-Markovian
- time-invariant vs. time-variant

Referring to the example given on the previous page, the relation between the electric field strength and the dielectric flux density can be characterized as a linear, homogenous, anisotropic, locally reacting, Markovian and time invariant constitutive relation. It is clear that this is the most simple constitutive relation conceivable, hence, we should pay considerable attention to the range of validity of such a relation. In the following chapter, where we discuss a rigorous thermodynamic model for the thermal and/or electrical energy domain, we shall adopt parametrized, quasi-linear, inhomogeneous, anisotropic, locally reacting, Markovian and time invariant constitutive relations. With respect to the parameters we refer to what is available in the literature, noticeably in physical data books such as Landold and Börnstein.

2.5 Generic Interface Conditions

When a specific region of the transducer configuration (cf. Section 2.2) is closed[32] with respect to a set of state variables, we need to specify boundary conditions at the boundary of this region. However, if this region is a composite of several different materials, it will prove to be convenient to shed

[31] A Markov type of constitutive relation is one that exhibits no memory effects. The terminology is borrowed from the theory of random processes.

some light on the general continuity properties of the state variables, at the interfaces between these different materials. We start with the Maxwell equations.

It is well known that by means of the Stokes and Gauss theorems we can derive the following interface conditions for the electromagnetic field state variables

$$\epsilon_{\alpha\beta\gamma} n_\beta (h_\gamma|_1 - h_\gamma|_2) = \xi_\alpha^{surface} \qquad \textbf{(2-30a)}$$

$$\epsilon_{\alpha\beta\gamma} n_\beta (e_\gamma|_1 - e_\gamma|_2) = 0 \qquad \textbf{(2-30b)}$$

$$n_\alpha (b_\alpha|_1 - b_\alpha|_2) = 0 \qquad \textbf{(2-30c)}$$

$$n_\alpha (d_\alpha|_1 - d_\alpha|_2) = \xi^{surface} \qquad \textbf{(2-30d)}$$

where n_α is the normal vector of the interface and is directed from material 2 to material 1. Equation **(2-30a)** states that the difference in the tangential component of the magnetic field strength on both sides of the interface must be equal to the surface current density[33]. Equation **(2-30b)** states that the tangential component of the electric field strength must be continuous upon crossing the interface. Equation **(2-30c)** states that the normal component of the magnetic flux density must be continuous upon crossing the interface. Finally, equation **(2-30d)** states that the difference in the normal component of the electric flux density on both sides of the interface must be equal to the surface charge density. Note that for our present purpose only equation **(2-30d)** is relevant.

Similar equations are needed for the energy and particle balances. Since the mathematical forms of the energy and particle balances are identical, we only consider the general (scalar) form[34] (cf. Section 2.3.2.2), that is

$$\partial_t e^k = -\partial_\beta e_\beta^k + \dot{e}^k \qquad \textbf{(2-31)}$$

[32] In this context, "closed" means that regions immediately neighboring the one under consideration do not possess the state variables of the region under consideration as free state variables. For example, in an ideal oxide-semiconductor interface we need to solve the Poisson equation in both the oxide and semiconductor, however, the current continuity equations only need to be solved in the semiconductor. Obviously, with respect to the Poisson equation, we need to specify interface conditions at the interface, and with respect to the current continuity equation we need to specify boundary conditions at the interface.

[33] The concept of surface charge and current is, of course, a mathematical idealization of the problem.

[34] In the case we want to study the mechanical domain we also need to consider the vectorial balance equations in order to reveal the continuity properties of the pressure tensor.

We shall assume that both the extensive state variable in the left-hand side and the scalar flux in the right-hand side can have a singularity in the form of a Dirac delta function, that is

$$e = e^{bulk} + e^{surface}\delta(x - x^{surface}) \tag{2-32a}$$

$$\dot{e} = \dot{e}^{bulk} + \dot{e}^{surface}\delta(x - x^{surface}) \tag{2-32b}$$

Now by using the Gauss theorem on equation (2-31) together with equations (2-32a) and (2-32b) we obtain the following result[35],

$$n_\alpha (e_\alpha|_1 - e_\alpha|_2) = -\partial_t e^{surface} + \dot{e}^{surface} \tag{2-33}$$

When the surface contributions are zero, the above relation states that the normal component of the flux through the interface must be continuous. In fact, the above equation represents an ordinary differential equation with respect to time and in order to solve it, it needs to be supplemented with yet another constitutive model.

With respect to the interface conditions we can make life a little easier by assuming that the surface contributions can all be assumed zero. In this case the normal components of the particle fluxes, the energy fluxes and the electric flux density are continuous.

2.6 Generic Boundary Conditions

As mentioned in the previous section, when a specific region of the transducer configuration is closed with respect to a set of state variables, we need to specify known boundary conditions at the boundary of this region. These boundary conditions take into account the interaction of the transducer configuration with the remaining part of the universe. Mathematically speaking, boundary conditions can be of the Dirichlett, Neumann, Robin and periodic types [Farlow 1982].

In the case of Dirichlett boundary conditions, we specify a model for the behavior of an intensive state variable at the boundary. For instance, at an ideal voltage-controlled electrical contact, we can specify the values of the electrochemical potentials[36]. In the case of Neumann boundary conditions, we specify a model for the behavior of the flux at the boundary. For example, in an ideal current-controlled electrical contact, we can specify the values of the electrical currents. A more general type of boundary condition is the Robin type, in this case we specify the flux in terms of the intensive state

[35] Note that the symbol e refers to an extensive state variable, not to the electric field strength.

[36] A neat feature of the electrochemical potentials is that they are equal to the specified driving voltage at the contact.

variables (or vice versa) at the boundary. For example, to take into account the non idealities of an electric contact[37], we can specify a current-voltage relationship. Periodical boundary conditions are convenient when it is known that the solution at one surface is identical to the solution at an opposite surface.

2.7 Concluding Remarks

In this chapter we have presented various aspects that are helpful to come to a unified abstract description of a solid-state transducer configuration. In particular, we have discussed an abstract geometrical representation of a generic transducer configuration, in terms of regions, interfaces and boundaries. In order to describe their characteristic (physical) features appropriate model equations are linked to each of the regions, interfaces and boundaries. With respect to obtaining explicit model equations we have argued that the thermodynamics of irreversible processes provides a powerful physical modeling paradigm that can be applied to a variety of physical situations. In order to manage the complexity somewhat we proposed the energy domains with the tacit assumption that a certain region of a transducer configuration operates in one or more energy domains. We also discussed a generic abstract representation of the model equations and argued that, for the sake of simplicity, it is convenient to represent apparently different physical models in the same abstract (canonical) mathematical form. Also some attention was paid to the general treatment of interface and boundary conditions.

In the following chapter the objective is to use the formalism of the thermodynamics of irreversible processes to derive the model equations for a certain region of the transducer configuration operating in the thermal and electrical energy domain.

[37] For instance, Ohmic and Shottky contacts.

Tact is the ability to tell a man he has an open mind when he has a hole in his head.

Chapter 3

A Thermodynamic Model for the Thermal and Electrical Energy Domains[1]

3.1 Introduction

The current trend in solid-state transducer research is to convert measurands to the electrical energy domain, so that the response of a transducer can be evaluated in terms of the electrical voltages and currents measured at the contacted areas of the device. This has the advantage that an abundant collection of electronic signal processing techniques is at one's disposal, enabling the design and fabrication of "smart sensors"[2] [Middelhoek 1989b]. From this point of view, it is interesting to try to couple the electrical energy domain to the various other energy domains discussed in the previous chapter. As a specific example, we investigate the application of the framework presented in the previous chapter to the construction of a computational model capable of describing a region of the transducer configuration operating in the thermal and electrical energy domain. This means that the effects of all other signal domains are by definition negligible. Of course, the question as to whether this is true has to be decided a posteriori by experimentation, or a priori by using a more complex model which is subsequently simplified by means of a number of well-chosen rules.

A number of basic phenomenological models can be found in the literature for which a numerical treatment of the electrical signal domain is more or less feasible, noticeably (in decreasing degree of complexity): (a) the Boltzmann equation approach, (b) the hydrodynamic approach and (c) the drift-diffusion approach. The Boltzmann equation approach itself is derived from the Liouville equation by using a truncated BBGKY hierarchy [Holt 1965]. The hydrodynamic approach is obtained by taking moments of the Boltzmann equation, usually the first three moments, although higher-order mo-

[1] The material in this section was also presented in [Duyn 1992].

[2] This poses the idea of combining multi-signal domain transducer simulation with lumped circuit models connected to the periphery of the transducer. Perhaps the basic models used in the public domain electrical circuit simulation package SPICE could be merged with the present models.

ments[3] are also conceivable [Bringer 1990]. The drift-diffusion model was originally brought forward by van Roosbroeck [Roosbroeck 1950] and is a gross simplification of the hydrodynamic model, however, with remarkable success in the field of semiconductor modeling and simulation.

As was pointed out in the previous chapter, we shall pursue a different line of reasoning by putting the modeling process into the realm of thermodynamic system theory. This gives us the advantage of a modular approach where several building blocks can be distinguished, moreover, it leads to an extensible framework. However, obvious extensions, such as the mechanical and magnetic domain, will be the subject of future research. The apparent power of the thermodynamic approach can be inferred from the fact that both the hydrodynamic and the drift diffusion model can be viewed as special cases of the thermodynamic model. Within the realm of thermodynamics of irreversible processes, several models with varying degree of complexity, describing the thermal, electric and thermoelectric features of a material, are conceivable. In order to keep things tractable we' take the drift-diffusion model as the point of departure and extend it to the thermal domain by using the principles of thermodynamics of irreversible processes. Where appropriate, we try to pay special attention to the set of constraints under which the model is developed. This gives some guidelines in relaxing some of these constraints in order to aid future extensions.

The structure of this chapter is as follows. Section 3.2 gives a description of a semiconductor from the thermodynamic perspective. Section 3.3 deals with the phenomenological model equations. Section 3.4 deals with the bulk constitutive equations. In Section 3.5 a "simplified" version of the thermoelectric model is summarized. Finally, in Section 3.6 we close the chapter with some concluding remarks.

3.2 Semiconductors from a Thermodynamic Perspective

In this section, a description of a semiconductor, when viewed from the thermodynamic perspective, is developed. It is to be used in the next sections as a basis for the discussion of the bulk phenomenological and constitutive equations. Although we specifically pay attention to semiconductor materials, descriptions for conductors and insulators follow as simplifications of the semiconductor model. To make life somewhat easier we assume that the entire configuration is of fixed chemical composition. This means, for instance, that we do not take into account the redistribution of doping impurities because of chemical diffusion.

According to the principles of thermodynamics, we can view the total system as a composite of a number of virtually independent thermodynamic

[3] Higher order moments are usually needed to deal with ballistic effects in submicron devices.

subsystems. According to the (simple) band theory of semiconductors we may distinguish the following subsystems[4]: (1) the lattice, (2) the electrons in the conduction band, (3) the electrons in the valence band, (4) the electrons in the donor-like localized states, and (5) the electrons in the acceptor-like localized states. The localized states[5] are capable of capturing (releasing) electrons from (to) the conduction and valence bands. Note that all subsystems are governed by fermion statistics except for the lattice which is governed by boson statistics.

As argued in Section 2.3.1, the next step is to assume that each of these subsystems is internally in local equilibrium [Holt 1965]. This means that, if we divide the material into small volume elements, in each volume element the subsystems are in internal equilibrium. Note that this does not necessarily mean that they have to be in mutual equilibrium. Depending on the governing statistics[6] of a subsystem this assumption allows the definition of a local temperature and a chemical potential. Since the lattice is governed by boson statistics a local lattice temperature is defined. The remaining subsystems are governed by fermi-statistics, which means that for each of these subsystems a local temperature and a local chemical potential can be defined [Blakemore 1962][Landsberg 1991].

Note 3-1: For the valence band it is often more convenient to refer to the number of vacant electron states. Such a vacant state is called a hole and the collection of holes in the valence band is referred to as the hole system. Likewise, the collection of electrons in the conduction band is referred to as the electron system. However, one must take great care in dealing with quantities defined for the hole system, because these quantities usually implicitly refer to the electrons in the valence band.

In order to identify each of the subsystems we define the set

$$\Sigma = \{l, e, h, d_m, a_n\} \qquad \textbf{(3-1)}$$

where l stands for the lattice system, e for the electron system, h for the hole system, d_m for the m'th donor-like localized state and a_n for the n'th acceptor-like localized state. In the case an element from this set is used as a superscript the corresponding quantity refers to that subsystem. For example, h^l is the thermal energy density of the lattice, n^e is the electron density in the electron system, and $n^{d_m^+}$ is the density of positively charged donors. The

[4] We can make it more complex by assuming that each subsystem can be divided into other subsystems, for instance, the hole system can be separated into heavy and light holes, the conduction band can be split into six equivalent band minima, and for the lattice we may distinguish between several phonon modes.

[5] We do not yet distinguish between donor, acceptor and trap-like states, instead, the traps are considered to be special donor-like or acceptor-like localized states.

[6] For systems governed by boson statistics we may only define the temperature. For systems governed by fermi statistics a temperature and chemical potential may be defined.

subsystems relevant to the thermal energy domain are indicated by means of the subset $S = \{l, e, h\}$. Similarly, the subsystems relevant to the electrical domain are indicated by the subset $N = \{e, h, d_m, a_n\}$. As implicated by the subsets S and N we do not explicitly deal with the thermal properties of the localized states. Certainly, the localized states will have some influence on the thermal properties (heat capacity and heat conductivity), however, for the sake of simplicity, we consider the localized states as an integral part of the lattice as far as the thermal properties are concerned[7]. We may now assume that the following Gibbs relations are valid

$$ds^l = \left(\frac{1}{T^l}\right) du^l \tag{3-2a}$$

$$ds^e = \left(\frac{1}{T^e}\right) du^e - \left(\frac{\bar{\nu}^e}{T^e}\right) dn^e \qquad ds^h = \left(\frac{1}{T^h}\right) du^h - \left(\frac{\bar{\nu}^h}{T^h}\right) dn^h \tag{3-2b}$$

$$ds^{a_n^-} = -\left(\frac{\bar{\nu}^{a_n^-}}{T^l}\right) dn^{a_n^-} \qquad ds^{d_m^+} = -\left(\frac{\bar{\nu}^{d_m^+}}{T^l}\right) dn^{d_m^+} \tag{3-2c}$$

where T^l, T^e and T^h are the temperatures of resp. the lattice, the electron system and the hole system, $\bar{\nu}^e$, $\bar{\nu}^h$, $\bar{\nu}^{a_n^-}$, $\bar{\nu}^{d_m^+}$ are the chemical potentials of respectively the electrons in the electron system, the holes in the hole system, the negatively charged (occupied) acceptors and the positively charged (free) donors. Moreover, s^k, u^k and n^k are the volume densities of resp. the entropy, internal energy, particles in subsystem k. In the case of a donor-like localized state, $n^{d_m^+}$ refers to the density of positively charged (free) donors, and in the case of an acceptor-like localized state $n^{a_n^-}$ refers to the density of negatively charged (occupied) acceptors. In the above equations the definition of the electrochemical potentials are such that they have equal sign. This for example means that the chemical potential $\bar{\nu}^h$ refers to the holes in the valence band. The reason for this is that it greatly simplifies notation because we do not have to deal with sign changes. However, in most cases we want to refer to the electronic chemical potentials. In that case we may use the following transformation,

$$\bar{\nu}^k \to -\frac{q^k}{|q^k|}\bar{\nu}^k \tag{3-3}$$

Note that the above relation introduces a sign reversal for the electrochemical potential in case we are dealing with positively charged particles. For example, the chemical potential of the electrons in the valence band is the

[7] Anyhow, including these effects yields models for which it is very hard to find the appropriate model parameters.

negative of the chemical potential of the holes in the valence band. In cases where we refer to the electronic chemical potential it is explicitly stated.

Note 3-2: In the above equations we have implicitly assumed that the following relation holds,

$$T^{l} = T^{d_m} = T^{a_n} \tag{3-4}$$

The reason for this is that the (thermal) coupling between the lattice and the localized states is so strong that it does not make any sense to define a separate temperature for a localized state. Moreover, the thermal contribution of the localized states to the entropy is assumed to be implicitly contained in the lattice contribution to the entropy. Note that the definition of separate temperatures for the lattice, electron and hole systems is questionable for low carrier densities, however, at high carrier densities it can be a valuable extension to the model.

Of course, the above Gibbs relation is only valid if the internal energy of each component were of thermal and chemical origin alone. However, for the present case this is certainly not so, hence, we should also include the other forms of energy stored in the lattice, electron, hole and localized state systems. According to the results stated in Section 2.3.1, this additional energy can basically be of mechanical origin (mechanical domain) and of electromagnetic origin (radiant, electric and magnetic domain). With respect to the mechanical domain, we make the following two assumptions. First, we assume that the host lattice is mechanically rigid, which, for instance, means that effects such as thermal expansion of the host lattice are ignored. This assumption tremendously simplifies the entire scheme. Second, we assume that the electron and hole systems are in mechanical equilibrium. This means that the time rates in the momentum and kinetic energy balances vanish[8]. As a result the charge carriers instantaneously react to applied electromagnetic forces and thus almost immediately reach the state where the supplied electromagnetic momentum and energy balances the loss caused by collisions with the localized states (and imperfections) of the lattice. For most materials this assumption is reasonable, however, exceptions such as GaAs do exist [Bringer 1990]. To include these effects we also need to relax the assumption of mechanical equilibrium in the electron and hole system, however, this is beyond the present aim.

By virtue of the previous assumptions, we may ignore the mechanical energy terms in the Gibbs relations. The only additional energy we have to take into account is the energy of electromagnetic origin. As a further restriction, let us now also assume that the transducer is not subjected to external electromagnetic fields. This means that we only have to take into account its "self

[8] This is what essentially makes the difference between a hydrodynamic model and a drift-diffusion model.

induced electromagnetic field". Moreover, if we assume that this field is quasi-static, the electromagnetic energy can be stored in the material as electric and magnetic polarization energy and as the electrostatic energy of a charge distribution in an electric field (cf. Section 2.3.1.6). In the following, we ignore the electric and magnetic polarization[9] energy of both the lattice and the carrier systems. This leaves only the electrostatic energy to be discussed. Using the Maxwell equations and the assumption of an electrostatic field, this additional energy can be written as

$$u^{el,\,k} = n^k q^k \varphi \tag{3-5a}$$

The above relation relates the electrostatic energy in component k to the product of the carrier density, the charge of a particle and the electrostatic potential. In differential form we may write

$$\mathrm{d}u^{el,\,k} = q^k \varphi \mathrm{d}n^k \tag{3-5b}$$

Defining the electrochemical potentials[10] as

$$\nu^k = \bar{\nu}^k + q^k \varphi \tag{3-6}$$

the Gibbs equations can be rewritten as

$$ds^l = \left(\frac{1}{T^l}\right) du^l \tag{3-7a}$$

$$ds^e = \left(\frac{1}{T^e}\right) du^e - \left(\frac{\nu^e}{T^e}\right) dn^e \qquad ds^h = \left(\frac{1}{T^h}\right) du^h - \left(\frac{\nu^h}{T^h}\right) dn^h \tag{3-7b}$$

$$ds^{a_n^-} = -\left(\frac{\nu^{a_n^-}}{T^l}\right) dn^{a_n^-} \qquad ds^{d_m^+} = -\left(\frac{\nu^{d_m^+}}{T^l}\right) dn^{d_m^+} \tag{3-7c}$$

The quantity ν^k is commonly referred to as the electrochemical potential. It can be interpreted as the average chemical plus electrostatic energy of a particle in subsystem k. Note that in the above formulation the electrochemical potential is defined in a similar fashion as in equations **(3-2a)-(3-2c)**. In order to refer to the electronic electrochemical potentials we can use equation **(3-3)** and **(3-6)** to arrive at the following transformation rule,

$$\nu^k \to -\frac{q^k}{|q^k|} \nu^k = -\frac{q^k}{|q^k|} (\bar{\nu}^k - |q^k| \varphi) \tag{3-8}$$

[9] The microscopic as well as the macroscopic magnetic polarization energy is ignored.

[10] In all cases we strictly refer to electronic energies, even in the case of the hole system and the donor-like localized states.

Note that in cases where we refer to the electronic chemical potential it is explicitly stated.

In the following two sections we take the above Gibbs relations to set up the frame of the phenomenological part of the model, according to the rules set out in Section 2.3.

3.3 The Bulk Phenomenological Equations

If we take a look at the right-hand sides of the Gibbs relations stated in equations **(3-7a)**, **(3-7b)** and **(3-7c)**, we immediately observe which phenomenological equations we in fact need. First, in order to describe the thermal domain, we need balance equations expressing the changes in the energy densities u^k. Second, in order to deal with the electrical domain, we need balance equations expressing the changes in the particle densities n^k. Let us first consider the energy balances, and then discuss the particle balances. Third, as indicated in the previous section the phenomenological equations should be compatible with the Maxwell equations. This will yield an additional phenomenological equation, which according to the theory in Section 2.4 actually should be viewed as a non-local constitutive relation.

3.3.1 Thermal Energy Balances

The Gibbs relation, stated in equations **(3-7a)**-**(3-7c)**, implies that there are three major contributions to the internal energy u^k of each component, that is, the thermal, chemical and electrical (electrostatic) energy

$$u^k = u^{th,\,k} + u^{ch,\,k} + u^{el,\,k} \qquad \textbf{(3-9)}$$

The only knowledge we have of the total internal energy u, is that it must be a conserved quantity. The balance equation for the total internal energy may then be written as[11]

$$\partial_t u = -\partial_\beta u_\beta \qquad \textbf{(3-10a)}$$

The term on the left-hand side represents the time rate of the internal energy and the first term on the right-hand side represents the divergence of the total internal energy flow. However, the balance equations for the internal energy of each component may then be written as

$$\partial_t u^k = -\partial_\beta u^k_\beta + \dot{u}^k \qquad \textbf{(3-10b)}$$

[11] With respect to the used notation, the flux quantity associated with an extensive state quantity can immediately be recognized because of the rank one tensor subscript. For example if e is some extensive state quantity, then e_α is the associated vectorial flux and $\dot{e}$ can be recognized as the associated scalar flux, also called the source term of extensive quantity e.

The additional source term on the right-hand side is because of the possible exchange of thermal, chemical and electrical energy between the components. Obviously, when equation (3-10b) is summed over the individual components, we necessarily obtain equation (3-10a). What we also know are the balance equations expressing the rate of change of the chemical and electrical energy. These equations can be written as[12]

$$\partial_t u^{el,k} = -\partial_\beta (n^k_\beta q^k \varphi) + n^k_\beta \partial_\beta q^k \varphi + \dot{n}^k q^k \varphi \tag{3-11a}$$

$$\partial_t u^{ch,k} = -\partial_\beta (n^k_\beta \bar{\nu}^k) + n^k_\beta \partial_\beta \bar{\nu}^k + \dot{n}^k \bar{\nu}^k \tag{3-11b}$$

Note that here, the part that is operated on by the divergence operator is purely convective. Subtracting equations (3-11a) and (3-11b) from equation (3-10b) and using equation (3-6) we may derive the balance equations for the thermal part of the internal energy (heat) of each component

$$\partial_t u^{th,k} = \partial_t h^k = -\partial_\beta (u^k_\beta - n^k_\beta \nu^k) + \dot{u}^k - n^k_\beta \partial_\beta \nu^k - \dot{n}^k \nu^k \tag{3-11c}$$

Clearly, the part operated on by the divergence operator represents the total heat flow in component k. Obviously, in the case where in equations (3-11a)-(3-11c) the superscript k refers to the donor or acceptor-like localized states, the corresponding flux must be set to zero, because we assume that no conduction caused by electron or hole hopping can take place in the localized states. The balance equation for the total thermal energy can be obtained by summing over the individual components in equation (3-11c). This results in

$$\partial_t u^{th} = -\partial_\beta \Big(u_\beta - \sum_{k=e,h} n^k_\beta \nu^k\Big) - \sum_{k=e,h} n^k_\beta \partial_\beta \nu^k - \sum_{k \in N} \dot{n}^k \nu^k \tag{3-11d}$$

Clearly, the part operated on by the divergence operator must represent the total heat flow in the system. The second term on the right-hand side is the Joule heat. Essentially, this is the energy supplied to the carriers by the electric field. This energy is subsequently lost as heat to the lattice, because of the collisions of the carriers with the localized states and the lattice imperfections[13]. The third term on the right-hand side represents the heat production caused by carrier exchange between the electron, hole, and localized states subsystems.

[12] In deriving these relations we have already implicitly used the particle balances.

[13] Note that because we ignore the momentum balance this loss of energy is instantaneous.

3.3.2 Particle Balances

For each component[14] a particle balance equation can be identified. First, we have the electron and hole particle balances governing the number of electrons and holes in the conduction and valence band

$$\partial_t n^e = -\partial_\alpha n_\alpha^e + \left[\dot{n}^{h \to e} + \sum_m \dot{n}^{d_m \to e} + \sum_n \dot{n}^{a_n \to e} \right] \qquad \textbf{(3-12a)}$$

$$\partial_t n^h = -\partial_\alpha n_\alpha^h + \left[\dot{n}^{h \to e} + \sum_m \dot{n}^{h \to d_m} + \sum_n \dot{n}^{h \to a_n} \right] \qquad \textbf{(3-12b)}$$

Note that in the above notation, the arrow superscript to indicate the local fluxes always refers to the exchange of an electron. For example, $h \to e$ means the net flux of electrons from the hole system (valence band) to the electron system. In equation **(3-12a)** the first term on the right-hand side represents the divergence of the electron particle flux, the second term represents the rate at which electrons move to the hole system (creation and annihilation of electron hole pairs), the third and fourth terms respectively represent the rates at which electrons are trapped in the donor and acceptor-like localized states. A similar argument holds for equation **(3-12b)**.

Next, we have the particle balance equations for the localized states in the band gap. Distinguishing between several non-interacting donor and acceptor-like localized states (as already implied by equations **(3-12a)** and **(3-12b)**) we may write

$$\partial_t n^{d_m^+} = \dot{n}^{d_m \to h} - \dot{n}^{e \to d_m} \qquad \textbf{(3-12c)}$$

$$\partial_t n^{a_k^-} = \dot{n}^{e \to a_n} - \dot{n}^{a_n \to h} \qquad \textbf{(3-12d)}$$

Note that in equations **(3-12c)** and **(3-12d)** we have omitted the transport terms. This corresponds to the assumption that no conduction caused by electron or hole hopping between localized states takes place [Barrie 1987]. This assumption is valid in the case where the quantum wave functions of the trapped electrons in two nearest neighbor localized states do not significantly overlap. For impurity densities[15] that are not too high, or operating temperatures that are not too low this is a good approximation.

3.3.3 Maxwell Equations

As said earlier, the above-stated phenomenological equations must be compatible with the Maxwell equations. To put it more clearly, the Maxwell

[14] Except the lattice of course, because nothing is assumed to be movable in the lattice.

[15] For silicon the critical impurity density lies somewhere around $3 \cdot 10^{18}$ cm^{-3}. This is known as the Mott transition.

equations impose an additional relation between the electrostatic potential and the electric charge. In a strict thermodynamic sense, such a relation should be classified as a non-local equilibrium constitutive relation (cf. Section 2.3.2.4), however, we shall not do so, instead we shall interpret it as a special phenomenological equation. In order to find this relation we proceed as follows.

In the case of zero external electromagnetic sources the Maxwell equations are given by

$$\partial_\alpha d_\alpha = \xi \tag{3-13a}$$

$$\partial_t d_\alpha = \partial_\beta \in_{\alpha\beta\gamma} h_\gamma - \xi_\alpha \tag{3-13b}$$

$$\partial_\alpha b_\alpha = 0 \tag{3-13c}$$

$$\partial_t b_\alpha = -\partial_\beta \in_{\alpha\beta\gamma} e_\gamma \tag{3-13d}$$

where, d_α is the electric flux density, e_α the electric field strength, b_α the magnetic flux density, h_α the magnetic field strength, ξ the electric charge density and ξ_α the electric current density. Next, we introduce the magnetic vector potential a_α and the scalar electric potential φ through the relations

$$b_\alpha = \partial_\beta \in_{\alpha\beta\gamma} a_\gamma \tag{3-14a}$$

$$e_\alpha = -\partial_t a_\alpha - \partial_\alpha \varphi \tag{3-14b}$$

It can easily be verified that the above two definitions are compatible with the Maxwell equations (cf. Appendix A). Now, using the electromagnetic constitutive equations[16]

$$b_\alpha = \mu_{\alpha\beta} h_\beta = \nu_{\alpha\beta}^{-1} h_\beta \tag{3-15a}$$

$$d_\alpha = \varepsilon_{\alpha\beta} e_\beta \tag{3-15b}$$

the Maxwell equations can be rewritten into an equivalent set of equations relating the magnetic and electric potentials

$$\begin{bmatrix} \partial_\sigma \in_{\alpha\sigma\omega} & -\partial_t \\ 0 & -\partial_\alpha \end{bmatrix} \cdot \begin{bmatrix} \nu_{\omega\tau} & 0 \\ 0 & -\varepsilon_{\alpha\beta} \end{bmatrix} \cdot \begin{bmatrix} \partial_\nu \in_{\tau\nu\beta} & 0 \\ \partial_t & \partial_\beta \end{bmatrix} \cdot \begin{bmatrix} a_\beta \\ \varphi \end{bmatrix} = \begin{bmatrix} \xi_\alpha \\ -\xi \end{bmatrix} \tag{3-16}$$

In fact, the above expression represents the Maxwell equations in terms of two wave equations. Note that the above equation conforms to the canonical

[16] These constitutive relations should be characterized as: linear, inhomogeneous, anisotropic, locally-reacting, Markovian and time-invariant.

form of a model equation as discussed in Section 2.4. Now, if we assume that the wave phenomena, described by equation **(3-16)**, travel with infinite speed[17] the time derivatives can be set to zero. This yields the following set of equations

$$\partial_\beta \in_{\alpha\beta\gamma} h_\gamma = \xi_\alpha \tag{3-17a}$$

$$h_\alpha = \nu_{\alpha\beta} \partial_\sigma \in_{\beta\sigma\gamma} a_\gamma \tag{3-17b}$$

$$\partial_\alpha d_\alpha = \xi \tag{3-18a}$$

$$d_\alpha = -\varepsilon_{\alpha\beta} \partial_\beta \varphi \tag{3-18b}$$

Equations **(3-17a)** and **(3-17b)** describe the magnetostatic behavior of the material (e.g. Eddy currents). Equations **(3-18a)** and **(3-18b)** describe the electrostatic behavior of the material. The second set is referred to as the Poisson equation. Moreover, note that both sets of equations conform to the canonical form of the model equation as discussed in Section 2.4. If we interpret the above equations in a thermodynamic sense then **(3-17a)** and **(3-18a)** are the phenomenological equations and **(3-17b)** and **(3-18b)** are the non-equilibrium constitutive relations (cf. Section 2.3.2.4).

We shall now make another simplifying assumption. If the drift velocity of the charge carriers is not too high, the force on the carriers caused by the "self induced" magnetic field is always orders of magnitudes smaller then the "self induced" electric field. Therefore equations **(3-17a)** and **(3-17b)** can usually be ignored when the current densities are not too high and no external magnetic field is present.

In order to relate equations **(3-18a)** and **(3-18b)** to the energy and particle balances, we proceed as follows. By taking the divergence of equation **(3-13b)** and then using equation **(3-13a)** we obtain the charge balance equation

$$\partial_t \xi = -\partial_\alpha \xi_\alpha \tag{3-19}$$

Obviously, this equation expresses the conservation of electric charge. Now if we multiply each of the equations **(3-12a)** to **(3-12c)** by their corresponding "particle" charge q^k and take into consideration that the sum of all carrier production terms or local fluxes must be zero because of the conservation of the total number of electrons, we end up with the following equivalent charge balance equation

$$\partial_t (q^e n^e + q^h n^h + q^h \sum_m n^{d_m^+} + q^e \sum_n n^{a_n^-}) = -\partial_\alpha (q^h n_\alpha^h + q^e n_\alpha^e) \tag{3-20}$$

[17] In semiconductors this is usually valid up to a few gigahertz.

By comparing equations **(3-19)** and **(3-20)**, we observe that the space charge and the corresponding electric current are given by the relations

$$\xi = q^e n^e + q^h n^h + q^h \sum_m n^{d_m^+} + q^e \sum_n n^{a_n^-} \qquad \textbf{(3-21a)}$$

$$\xi_\alpha = q^h n_\alpha^h + q^e n_\alpha^e \qquad \textbf{(3-21b)}$$

Note that in our case, the electric current is taken positive in the direction where the holes flow and negative $(q^e < 0)$ in the direction where the electrons flow. The space charge density is the sum of the individual charge densities in the conduction and valence band and in the discrete energy levels of the localized states in the band gap.

3.4 The Bulk Constitutive Relations

The equations presented in the previous sections do not yet form a closed computational model. To achieve this we need additional information in the form of constitutive relations. According to Section 2.3, these can be of two types: (a) equilibrium relations defined in equilibrium thermodynamics, which express the relation between the intensive and extensive state variables, and (b) non-equilibrium relations defined in irreversible thermodynamics, which express the relation between the thermodynamic forces and fluxes. Usually, a whole range of such constitutive relations, with varying degrees of complexity, is conceivable (cf. Section 2.4), however, in this section we restrict ourselves to (quasi) non-linear, inhomogeneous, anisotropic, locally-reacting, Markovian, and time-invariant constitutive relations[18]. Moreover, in the semiconductor field, the refinement of the constitutive relations is still an active field of research, therefore, we restrict ourselves to the general framework and exemplify it with some of the more established constitutive models. In Section 3.4.1 we discuss the equilibrium equations of state and in Section 3.4.2 the non-equilibrium equations of state.

3.4.1 Equilibrium Equations of State

The purpose of this subsection is to express the intensive state variables in terms of the extensive state variables. In the spirit of thermodynamics that "everything depends on everything" the Gibbs equations **(3-7a)**-**(3-7c)** imply the following equations of state [Münster 1970]

$$T^k = T^k(s^l, s^e, s^h; n^e, n^h, n^{d_m}, n^{a_n}) \quad k = \{l, e, h\} \qquad \textbf{(3-22a)}$$

[18] The formal treatment of more general constitutive relations is a very interesting research subject, however, the theory as presented in this chapter already presents severe numerical implementation difficulties that first need to be overcome.

$$\mathrm{v}^k = \mathrm{v}^k(s^l, s^e, s^h; n^e, n^h, n^{d_m}, n^{a_n}) \quad k = \{e, h, d_m^+, a_n^-\} \tag{3-22b}$$

These equations are valid for each volume element in local thermodynamic equilibrium and express the local temperatures and electrochemical potentials as functions of the entropy of the lattice, electron and hole systems, and "particle" densities in the electron, hole and localized state systems.

However, in order to use a statistical theory to find explicit relations for the above equations, it is more convenient to rewrite them as expressions relating the extensive to the intensive state variables. The formal procedure is to apply a Legendre transform [Münster 1970] to the fundamental equation to obtain the grand potential[19], that is

$$\mathrm{d}\Omega^k = -s^k \mathrm{d}T^k + n^k \mathrm{d}\mathrm{v}^k \tag{3-23}$$

where it is implicitly assumed that the volume is constant. We may then write the equations of state as

$$s^l = s^l(T^l) \tag{3-24a}$$

$$s^k = s^k(T^k, \mathrm{v}^k) \quad k = \{e, h\} \tag{3-24b}$$

$$n^k = n^k(T^k, \mathrm{v}^k) \quad k = \{e, h, d_m^+, a_n^-\} \tag{3-24c}$$

Note that we have already tremendously simplified the equations by assuming that certain dependencies are not present. Explicit expressions for the above equations of state can be obtained for simple systems by means of equilibrium statistical thermodynamic principles [Toda 1983][Kubo 1985][Blakemore 1962][Landsberg 1991]. As far as the electrical domain is concerned, the usual approach is to treat the electrons and holes as ideal Fermi-Dirac gases moving in a vessel determined by the physical boundaries of the host lattice. The effect of the periodic lattice is taken into account by means of the effective mass approximation. For non-degenerate semiconductors, the electrons and holes can be treated as a Maxwell-Boltzmann gas. We shall not (yet) simplify the statistics to Maxwell-Boltzmann but rather use the Fermi-Dirac statistics. This, in principle, validates the use of the relations in the low temperature region[20]. As far as the thermal domain is concerned, the usual approach is to assume that the lattice can be represented as a phonon gas[21] (boson statistics) moving in a vessel determined by the physical boundary of the host lattice.

[19] The grand potential is used to link thermodynamics with statistical thermodynamics.

[20] Apart from some parameters which should be experimentally or theoretically determined at low temperatures.

[21] For this, a technique called second quantization is used [Ashcroft 1976]

Let us now give the explicit forms of the commonly used equilibrium equations of state. First, we give the ones of the electrical domain and, second, those of the thermal domain.

3.4.1.1 Electrical Domain

In the following, we ignore heavy doping effects. The "electrical" equations of state for the electron and hole systems can be written in terms of the Fermi-Dirac integrals (cf. Appendix D). Below, these are written in a form relating the carrier densities to the temperature and electrochemical potential[22]

$$n^e = N^{cb} F_{1/2}\left(\frac{\nu^e - E^{cb}}{kT^e}\right) \qquad \textbf{(3-25a)}$$

$$n^h = N^{vb} F_{1/2}\left(\frac{E^{vb} - \nu^h}{kT^h}\right) \qquad \textbf{(3-25b)}$$

The conduction band density of states N^{cb} and the valence band density of states N^{vb} can be expressed as

$$N^{cb} = 2\left(\frac{2\pi m^{cb} kT^l}{h^2}\right)^{\frac{3}{2}} \qquad \textbf{(3-26a)}$$

$$N^{vb} = 2\left(\frac{2\pi m^{vb} kT^l}{h^2}\right)^{\frac{3}{2}} \qquad \textbf{(3-26b)}$$

where k is the Boltzmann constant, h is the Planck constant, and m^{cb} and m^{vb} respectively are the conduction band and valence band effective density of states mass, which can be expressed as [Barber 1967]

$$m^{cb} = m^0\,[1.045 + 4.5\times10^{-4}\cdot T] \quad (50 \le T \le 350\text{ K}) \qquad \textbf{(3-27a)}$$

$$m^{vb} = m^0\,[0.523 + 1.4\times10^{-3}T - 1.48\times10^{-6}\cdot T^2] \quad (50 \le T \le 350\text{ K}) \qquad \textbf{(3-27b)}$$

The band edge energy of the conduction band E^{cb}, and the band edge energy of the valence band E^{vb} can be expressed as [Marshak 1989]

$$E^{cb} = E_0^{cb} - q\varphi = E_0 - \chi + \frac{1}{2}E^{gp} - q\varphi \qquad \textbf{(3-28a)}$$

$$E^{vb} = E_0^{vb} - q\varphi = E_0 - \chi - \frac{1}{2}E^{gp} - q\varphi \qquad \textbf{(3-28b)}$$

[22] Note that in the second equation the sign of the electrochemical potential is reversed because we refer to the electronic electrochemical potential.

where E_0 is an arbitrary reference level, χ is the electron affinity, and E^{gp} is the bandgap defined as $E^{gp} = E^{cb} - E^{vb}$. The electron affinity is usually needed to model hetero-structures (e.g. Ge-Si). In the literature, there is still much controversy with respect to an appropriate model for χ, valid for several types of semiconductor materials. As long as we do not need to model hetero-structures, we may set χ to zero.

The bandgap for silicon can be expressed as a temperature-dependent function [Bladau 1974][Selberherr 1989]

$$E^{gp} = \begin{cases} q\,[1.1700 + 1.059\times10^{-5} \cdot T - 6.05\times10^{-7} \cdot T^2] & (T \le 170\ \mathrm{K}) \\ q\,[1.1785 - 9.025\times10^{-5} \cdot T - 3.05\times10^{-7} \cdot T^2] & (T > 170\ \mathrm{K}) \end{cases} \tag{3-29}$$

The equations of state for the donor- and acceptor-like localized states can be written in terms of the Fermi-Dirac distribution function. Below, these are written in a form relating the number of electrons and holes in the donor- and acceptor-like localized states to the temperature and electrochemical potentials [Landsberg 1991][23]

$$n^{a_n^-} = n^{a_n}\left(\frac{1}{1 + g^{a_n}\exp\left(\frac{E^{a_n} - \nu^{a_n^-}}{kT^l}\right)}\right) \tag{3-30a}$$

$$n^{d_m^+} = n^{d_m}\left(\frac{1}{1 + g^{d_m}\exp\left(\frac{\nu^{d_m^+} - E^{d_m}}{kT^l}\right)}\right) \tag{3-30b}$$

where $n^{d_m^+}$ and $n^{a_n^-}$ respectively are the densities of the free donor-like localized states and the occupied acceptor-like localized states, n^{d_m} and n^{a_n} are the densities of the donor- and acceptor-like localized states, E^{d_m} and E^{a_n} are the (discrete) energy levels of the m'th donor-like localized state and the n'th acceptor-like localized state. The factors g^{d_m} and g^{a_n} are the spin degeneracy factors of the donor and acceptor localized states. In the case of silicon with phosphor as donors and boron as acceptors, the spin degeneracy factors take the values 2 and 4.

Note 3-3: Equations **(3-25a)** and **(3-25b)** are only valid for parabolic band edges, and can be evaluated analytically only in the non-degenerate or the highly degenerate case. In the intermediate region, it is best to use a look up table to calculate the Fermi-Dirac integrals (cf. Appendix D). For non-

[23] Note that in the second equation the sign of the electrochemical potential is reversed, because we refer to the electronic electrochemical potential.

parabolic band edges, one can treat the electrons and holes as a Kane gas, for which analytical expressions can be found in the non-degenerate case [Landsberg 1991]. The degenerate case has to be tabulated, just as in the case of the Fermi-Dirac integrals. An alternative approach could be to approximate the band non-parabolicity in terms of powers of the energy. The carrier densities can then be obtained as an expansion in Fermi-Dirac integrals of various orders [Blakemore 1962].

Note 3-4: Equations **(3-30b)** and **(3-30a)** are only valid for a set of non-interacting monovalent localized states [Landsberg 1991]. When several interacting localized states are present, the situation becomes extremely complex. In this case it is not sufficient to just account for other impurity levels by compensation as usually is done with semiconductors with donor and acceptor states. This viewpoint is only justifiable when the Fermi level is either far above or far below all other kinds of impurity levels. Especially, when the temperature behavior must be modeled accurately over a wide range, a more appropriate model of the localized states is important.

3.4.1.2 Thermal Domain

As already assumed in equation **(3-24a)**, the effect of the localized states on the lattice entropy can usually be neglected, because in most cases the number of lattice atoms is much larger than the number of localized states. Further, the contribution to the entropy of the electrons and holes trapped in the localized states is usually negligible.

In differential form, the equation of state for the lattice entropy can formally be written as

$$\mathrm{d}h^l = T^l \mathrm{d}s^l = c_P^l \mathrm{d}T^l \qquad \textbf{(3-31a)}$$

with the heat capacity of the lattice at constant pressure defined as

$$c_P^l = T^l \left(\frac{\partial s^l}{\partial T^l}\right)_P \qquad \textbf{(3-31b)}$$

At this stage, we should perhaps address an important issue concerning the fact that here the heat capacity referred to corresponds to the heat capacity at constant pressure. However, when the experimental configuration corresponds to the situation that the volume instead of the pressure is constant, we should use the heat capacity at constant volume. The two are related to each other by[24]

[24] This can easily be proven by using the Maxwell relations for transforming thermodynamic partial differentials.

$$c_V^l = c_P^l - \frac{Tv\alpha_P^2}{\kappa_T} \tag{3-32}$$

where v is the specific volume, α_P the thermal expansion coefficient of the lattice at constant pressure, and κ_T the compressibility at constant temperature. These are defined by

$$\alpha_P = \frac{1}{v}\left(\frac{\partial v}{\partial T}\right)_P \qquad \kappa_T = \frac{-1}{v}\left(\frac{\partial v}{\partial P}\right)_T \tag{3-33}$$

Fortunately, in the case of solids, the correction factor as expressed in equation **(3-32)** is usually very small, therefore, the difference between c_V and c_P is negligible.

Data for the lattice heat capacity at constant pressure as a function of the temperature can, for silicon, be found in Landolt and Börnstein on page 398-399. For temperatures well below the Debye temperature Θ_D, a simple power law will do (Debye interpolation scheme)

$$c_P(T^l) = c \cdot \left[\frac{T^l}{\Theta_D}\right]^3 \quad (T \ll \Theta_D) \quad \left[\frac{\mathrm{J}}{\mathrm{K} \cdot \mathrm{cm}^3}\right] \tag{3-34a}$$

The Debye temperature depends weakly on the temperature and the density of the localized states. For silicon, the following values can be taken

$$\begin{aligned} \Theta_D &\cong 645\ K \\ c &\cong 16.27 \end{aligned} \tag{3-34b}$$

When the internal deviation from the operating temperature is not too large, we may simply evaluate the lattice heat capacity at the operating temperature. For instance, at 300 K we have $c_P \cong 1.637$.

In order to derive some useful formulas for the "thermal" equations of state of the electron and hole systems we proceed as follows. In differential form we may write[25]

$$\mathrm{d}h^k = T^k \mathrm{d}s^k = T^k\left[\left(\frac{\partial s}{\partial T}\right)_{\mathrm{v}^k} \mathrm{d}T + \left(\frac{\partial s}{\partial \mathrm{v}^k}\right)_T \mathrm{d}\mathrm{v}^k\right] \quad k = e, h \tag{3-35a}$$

Using the Maxwell relations for transforming thermodynamic partial differentials [Callen 1960] we may transform the above relation to a form

$$\mathrm{d}h^k = T^k \mathrm{d}s^k = c_n^k \mathrm{d}T - T^k\left(\frac{\partial \mathrm{v}^k}{\partial T^k}\right)_{n^k} \mathrm{d}n^k \quad k = e, h \tag{3-35b}$$

[25] We implicitly assume the relations to hold at either constant pressure or constant volume.

where c_n^k is the heat capacity at constant charge carrier density of either the electron or the hole system. The carrier heat capacity at constant carrier density can be approximated by a linear temperature dependence (free electron heat capacity) [Ashcroft 1976]

$$c_n^k = T^k\left(\frac{\partial s^k}{\partial T^k}\right)_{n^k} = c^k T^k \qquad k = e, h \tag{3-35c}$$

where c^k approximately equals 2.5×10^{-6} in the case of silicon. Note that the carrier heat capacities can safely be neglected with respect to the lattice heat capacity. The second term on the right-hand side of equation (3-35b) is more important, using the Maxwell relations and equation (3-25a) and (3-25b) it can (in the non-degenerate case) be written as

$$T^k\left(\frac{\partial \nu^k}{\partial T^k}\right)_{n^k} = -T^k\frac{\left(\frac{\partial n^k}{\partial T^k}\right)_{\nu^k}}{\left(\frac{\partial n^k}{\partial \nu^k}\right)_{T^k}} = \begin{cases} \nu^e - E^{cb} + T^e\left(\frac{\partial E^{cb}}{\partial T^e}\right)_{\nu^k} + (3/2)\,kT \\ \nu^h - E^{vb} + T^h\left(\frac{\partial E^{vb}}{\partial T^h}\right)_{\nu^k} + (3/2)\,kT \end{cases} \tag{3-35d}$$

For parabolic band edges, the "thermal" equations of state for the electron and hole systems can be represented in terms of the Fermi-Dirac integrals, that is

$$h^e = (kT^e)N^{cb}F_{3/2}\left(\frac{\nu^e - E^{cb}}{kT^e}\right) + E^{cb}N^{cb}F_{1/2}\left(\frac{\nu^e - E^{cb}}{kT^e}\right) \tag{3-36a}$$

$$h^h = (kT^h)N^{vb}F_{3/2}\left(\frac{E^{vb} + \nu^h}{kT^h}\right) - E^{vb}N^{vb}F_{1/2}\left(\frac{E^{vb} + \nu^h}{kT^h}\right) \tag{3-36b}$$

where, h^e and h^h are the thermal energy densities. Note that the above relations can be used to calculate equation (3-35a).

Note 3-5: Equation (3-35a) relates the change of the thermal energy, stored in the electron and hole systems, to the change in temperature and electrochemical potential. Similarly, equation (3-35b) relates the change in thermal energy to the change in temperature and carrier densities.

Note 3-6: In most attempts to model the thermal signal domain, the effect of the second term in equation (3-35a) is not taken into account. In time-dependent processes, this contribution should not be neglected, because it enters the equations when transforming the term $\partial_t u^{th} = \partial_t h$, in the total heat balance equation (3-11c), to an expression explicit in the thermodynamic state variables, e.g.

$$\partial_t(h^l + h^e + h^h) \rightarrow (c^l + c_n^e + c_n^h)\,\partial_t T + a_1\partial_t n^e + a_2\partial_t n^h \tag{3-37}$$

According to equation (3-35d) the proportionality factors a_1, a_2 are of the order of the bandgap, hence, for rapid transient processes, that is, large $\partial_t n^e$ and $\partial_t n^h$, a significant amount of heat can be produced.

Note 3-7: Equation (3-35d) implicitly contains the derivative of the electrostatic potential with respect to the temperature. In principle, this quantity behaves as a non-local constitutive parameter. It can be evaluated by taking the partial differential with respect to T on both sides of the Poisson equation and then solving for $\partial\varphi/\partial T$. This term becomes important in the case of fast transient processes. However, since the inclusion of this term in the model severely complicates the numerical scheme we leave it for future research.

3.4.2 Non-Equilibrium Equations of State

In the previous section, we identified the equilibrium equations of state. Similarly, we must also identify the non-equilibrium equations of state, relating the scalar and vectorial fluxes to their appropriate thermodynamic forces.

As discussed earlier, a proper way to set up the non-equilibrium equations of state is to use thermodynamics of irreversible processes [Duyn 1990][Duyn 1992][Callen 1948][Li 1962][Groot 1969][Gyarmati 1970]. According to the central theorem of irreversible thermodynamics, the proper fluxes and forces, for which the Onsager-Casimir reciprocal relations (cf. Section 2.3.2.4) hold, can be found by calculating the entropy production [Groot 1969]. Moreover, the requirement of positive definiteness of the entropy production term, discussed in Section 2.3.2.3, then gives us a clear guideline in choosing the constitutive relations between the thermodynamic fluxes and forces. Further, the entropy source term multiplied by the temperature is an explicit form of the dissipation function discussed in Section 2.3.2.3. In other words the entropy production can be used to calculate the total heat production in a thermodynamic system in an unambiguous manner. This opposed to the rather heuristic models usually encountered in device physics [Selberherr 1984].

The entropy balance and consequently the entropy production can be calculated by substituting the particle and energy balances, discussed in Section 3.3, in the Gibbs relations given in equations (3-7a)-(3-7c). After some lengthy but straightforward algebraic manipulations we then find for the total entropy balance

$$\partial_t s = -\partial_\alpha s_\alpha + \dot{s}^{vector} + \dot{s}^{scalar} \qquad \text{(3-38a)}$$

with

$$s_\alpha = \sum_{k=e,h,l} \left(\frac{h^k_\beta}{T^k}\right) \qquad \text{(3-38b)}$$

$$\dot{s}^{vector} = -\sum_{k=e,h,l} h_\alpha^k\left(\frac{\partial_\alpha T^k}{(T^k)^2}\right) - \sum_{k=e,h} n_\alpha^k\left(\frac{\partial_\alpha v^k}{T^k}\right) \qquad \textbf{(3-38c)}$$

$$\dot{s}^{scalar} = \sum_{k=e,h,l} \dot{u}^k\left(\frac{1}{T^k}\right) - \sum_{k\in N} \dot{n}^k\left(\frac{v^k}{T^k}\right) \qquad \textbf{(3-38d)}$$

Clearly, equation **(3-38a)** relates the time rate of the entropy density to the divergence of the entropy flux and the entropy production or source term. As can be observed, the entropy production is split into a vectorial part and a scalar part. Equation **(3-38b)** must be interpreted as the total entropy flow. It consists of the sum of the heat flow in the electron, hole and lattice subsystems divided by the temperature of each subsystem. These heat flows are respectively defined as

$$h_\beta^e = u_\beta^e - v^e n_\beta^e \qquad h_\beta^h = u_\beta^h - v^h n_\beta^h \qquad h_\beta^l = u_\beta^l \qquad \textbf{(3-39)}$$

The first summation in the vectorial entropy production given in equation **(3-38c)** is the sum of the products of the vectorial heat fluxes and their associated vectorial thermal forces in the electron, hole and lattice subsystems. The second summation in equation **(3-38c)** is the sum of the products of the vectorial carrier fluxes (electrons and holes) and their associated vectorial electrical forces.

To interpret the scalar entropy source term in equation **(3-38d)**, we first write the carrier and energy source term in each component as a sum of scalar fluxes

$$\dot{n}^e = \sum_{m\neq e} \dot{n}^{e\leftarrow m} \qquad ; (m\in N) \qquad \textbf{(3-40a)}$$

$$\dot{n}^h = \sum_{m\neq h} \dot{n}^{h\rightarrow m} \qquad ; (m\in N) \qquad \textbf{(3-40b)}$$

$$\dot{n}^{d_m^+} = \dot{n}^{d_m\rightarrow h} + \dot{n}^{d_m\rightarrow e} \qquad \dot{n}^{a_m^-} = \dot{n}^{a_m\leftarrow h} + \dot{n}^{a_m\leftarrow e} \qquad \textbf{(3-40c)}$$

$$\dot{u}^k = \sum_{m\neq k} \dot{u}^{k\leftarrow m} \qquad ; (k, m\in S) \qquad \textbf{(3-40d)}$$

Now, using the above relations and the fact that a scalar flux from component *k* to component *m* must be the opposite of the flux from component *m* to component *k*, we may rewrite the scalar entropy production in equation **(3-38d)** as

$$\dot{s}^{scalar} = \sum_{k \in N} \sum_{\substack{m \in N \\ m > k}} \left[\dot{n}^{k \to m} \left(\frac{\mathrm{v}^k}{T^k} - \frac{\mathrm{v}^m}{T^m} \right) \right] + \sum_{k = l, e, h} \sum_{\substack{m = l, e, h \\ m > k}} \left[\dot{u}^{k \to m} \left(\frac{1}{T^m} - \frac{1}{T^k} \right) \right] \tag{3-41}$$

where the electrochemical potentials now refer to the electrons. Clearly, the scalar entropy production as expressed in the above equation is now in the form of a sum of products of respectively scalar fluxes with their associated scalar forces. Note that with respect to the first double summation in equation **(3-38d)**, we implicitly assume relation **(3-4)** to hold. The various vectorial and scalar fluxes are respectively summarized in Table 3-1 and Table 3-2.

In the remainder of this section, we identify the scalar and vectorial non-equilibrium equations of state[26]. First, we discuss the vectorial phenomena and second the scalar phenomena.

Table 3-1 Summary of the main thermal and electrical scalar fluxes and forces (the Fermi levels refer to the electrons).

thermal domain		electrical domain	
fluxes:	forces:	fluxes:	forces:
$\dot{u}^{e \to h}$	$(\frac{1}{T^h} - \frac{1}{T^e})$	$\dot{n}^{e \to h}$	$\left(\frac{\mathrm{v}^e}{T^e} - \frac{\mathrm{v}^h}{T^l}\right)$
$\dot{u}^{e \to l}$	$(\frac{1}{T^l} - \frac{1}{T^e})$	$\dot{n}^{e \to d_m}$	$\left(\frac{\mathrm{v}^e}{T^e} - \frac{\mathrm{v}^{d_m}}{T^l}\right)$
$\dot{u}^{h \to l}$	$(\frac{1}{T^l} - \frac{1}{T^h})$	$\dot{n}^{e \to a_n}$	$\left(\frac{\mathrm{v}^e}{T^e} - \frac{\mathrm{v}^{a_n}}{T^l}\right)$
		$\dot{n}^{h \to d_m}$	$\left(\frac{\mathrm{v}^h}{T^h} - \frac{\mathrm{v}^{d_m}}{T^l}\right)$
		$\dot{n}^{h \to a_n}$	$\left(\frac{\mathrm{v}^h}{T^h} - \frac{\mathrm{v}^{a_n}}{T^l}\right)$

[26] Note that we implicitly assumed Curie's principle to hold, which states that the vectorial and scalar effects do not mix in isotropic materials.

Table 3-2 Summary of the main thermal and electrical vectorial fluxes and forces (the Fermi levels refer to the electrons).

thermal domain		electrical domain	
fluxes:	forces:	fluxes:	forces:
h^e_α	$-\dfrac{\partial_\alpha T^e}{(T^e)^2}$	n^e_α	$-\dfrac{\partial_\alpha v^e}{T^e}$
h^h_α	$-\dfrac{\partial_\alpha T^h}{(T^h)^2}$	n^h_α	$\dfrac{\partial_\alpha v^h}{T^h}$
h^l_α	$-\dfrac{\partial_\alpha T^l}{(T^l)^2}$		

3.4.2.1 Vectorial Phenomena

The purpose of this subsection is to set up the general form of the non-equilibrium equations of state for the carrier flows in the electron and hole subsystems, and the heat flows in the electron, hole and lattice subsystems (cf. Table 3-2).

Remembering the remarks made on the general form of the constitutive equations, (cf. Section 2.4) let us assume that the vectorial non-equilibrium equations of state relating the carrier and heat fluxes to the vectorial thermodynamic forces may be written as Markov expressions in terms of the forces, that is

$$j^k_\alpha = j^k_\alpha(\ldots, f^m_\beta, \ldots) \qquad \textbf{(3-42)}$$

where it is assumed that the f^m_β are the vectorial forces and the j^k_α are the vectorial fluxes listed in Table 3-2. The question as to whether or not the above relations can be assumed to be truly Markovian is hard to justify in advance, however, some specific remarks can be made. Consider, for instance, the physical effect of velocity overshoot of the charge carriers upon the sudden application of a force. In this case the flux cannot be Markovian with respect to the applied force. This is not a deficiency in the basic thermodynamic framework we have used, on the contrary, this shortcoming can be traced back to the omitting of the momentum balance equation for the charge carrier systems (assumption of mechanical equilibrium). In almost all practical cases this is a valid assumption because momentum relaxation typically takes place on a time scale of pico to femto seconds. There is a general rule in irreversible thermodynamic system theory which states that whenever a constitutive relation is encountered that is non-Markovian we must remedy the problem by looking for hidden state variables, hence, to remedy the problem of velocity overshoot, the momentum balance together with its associated state variables should be taken into account. Effectively, this procedure leads to a hydrodynamic model and the constitutive relations then emanating can again assumed to be Markovian.

Since each vectorial flux is known to vanish as the forces vanish, we can relate the fluxes to the forces by means of the following quasi-linear expansion

$$j_\alpha^k = \sum_m \left(a_{\alpha\beta}^{k,m} + \sum_n a_{\alpha\beta\gamma}^{k,m,n} f_\gamma^n + \dots\right) f_\beta^m = \sum_m \hat{a}_{\alpha\beta}^{k,m} f_\beta^m \qquad \textbf{(3-43)}$$

Note that the expansion coefficients in general are non-linear functions of the intensive state variables and the generalized forces. Also note that the expansion coefficients must be chosen such that the entropy production $\dot{s}^{vector}$, discussed in Section 3.4.2, is positive ($\hat{a}_{\alpha\beta}^{k,m} \geq 0$). We now rewrite equation **(3-43)** in the form of a matrix equation relating the vectorial fluxes to the forces (cf. Table 3-2) by means of the transport matrix, that is

$$\begin{bmatrix} n_\alpha^e \\ n_\alpha^h \\ h_\alpha^e \\ h_\alpha^h \\ h_\alpha^l \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} \hat{a}_{\alpha\beta}^{ee} & \hat{a}_{\alpha\beta}^{eh} \\ \hat{a}_{\alpha\beta}^{he} & \hat{a}_{\alpha\beta}^{hh} \end{bmatrix} & \begin{bmatrix} \hat{a}_{\alpha\beta}^{eh^e} & \hat{a}_{\alpha\beta}^{eh^h} & \hat{a}_{\alpha\beta}^{eh^l} \\ \hat{a}_{\alpha\beta}^{hh^e} & \hat{a}_{\alpha\beta}^{hh^h} & \hat{a}_{\alpha\beta}^{hh^l} \end{bmatrix} \\ \begin{bmatrix} \hat{a}_{\alpha\beta}^{h^e e} & \hat{a}_{\alpha\beta}^{h^e h} \\ \hat{a}_{\alpha\beta}^{h^h e} & \hat{a}_{\alpha\beta}^{h^h h} \\ \hat{a}_{\alpha\beta}^{h^l e} & \hat{a}_{\alpha\beta}^{h^l h} \end{bmatrix} & \begin{bmatrix} \hat{a}_{\alpha\beta}^{h^e h^e} & \hat{a}_{\alpha\beta}^{h^e h^h} & \hat{a}_{\alpha\beta}^{h^e h^l} \\ \hat{a}_{\alpha\beta}^{h^h h^e} & \hat{a}_{\alpha\beta}^{h^h h^h} & \hat{a}_{\alpha\beta}^{h^h h^l} \\ \hat{a}_{\alpha\beta}^{h^l h^e} & \hat{a}_{\alpha\beta}^{h^l h^h} & \hat{a}_{\alpha\beta}^{h^l h^l} \end{bmatrix} \end{bmatrix} \cdot \begin{bmatrix} -\dfrac{\partial_\alpha v^e}{T^e} \\ +\dfrac{\partial_\alpha v^h}{T^h} \\ -\dfrac{\partial_\alpha T^e}{(T^e)^2} \\ -\dfrac{\partial_\alpha T^h}{(T^h)^2} \\ -\dfrac{\partial_\alpha T^l}{(T^l)^2} \end{bmatrix} \qquad \textbf{(3-44)}$$

The upper left block in the transport matrix describes the electrical domain, the upper right block describes the cross effects between the electrical and the thermal domain, the lower right block describes the thermal domain and the lower left block describes the cross effects between the thermal and the electrical domain. If we assume the Onsager reciprocal relations and the postulate of minimum entropy production (cf. Section 2.3.2.3) to hold, the transport matrix is symmetric[27] and positive definite. The property of positive

[27] More precisely, it is antisymmetric when an *external* magnetic field is present.

definiteness guarantees that the problem has a stable steady-state solution. In mathematical terminology these types of problems are called elliptic (cf. Chapter 4).

We now describe each block of the transport matrix in more detail. First, the electrical domain, followed by the thermal domain. However, we simplify the discussion to the case that the lattice, electron and hole temperatures are equal[28], thus equation **(3-44)** can be simplified to

$$\begin{bmatrix} n_\alpha^e \\ n_\alpha^h \\ h_\alpha \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} \hat{a}_{\alpha\beta}^{ee} & \hat{a}_{\alpha\beta}^{eh} \\ \hat{a}_{\alpha\beta}^{he} & \hat{a}_{\alpha\beta}^{hh} \end{bmatrix} & \begin{bmatrix} \hat{a}_{\alpha\beta}^{eu} \\ \hat{a}_{\alpha\beta}^{hu} \end{bmatrix} \\ \begin{bmatrix} \hat{a}_{\alpha\beta}^{ue} & \hat{a}_{\alpha\beta}^{uh} \end{bmatrix} & \begin{bmatrix} \hat{a}_{\alpha\beta}^{uu} \end{bmatrix} \end{bmatrix} \cdot \begin{bmatrix} -\dfrac{\partial_\alpha \nu^e}{T} \\ +\dfrac{\partial_\alpha \nu^h}{T} \\ -\dfrac{\partial_\alpha T}{T^2} \end{bmatrix} \qquad \textbf{(3-45)}$$

3.4.2.1.1 Electrical Domain

In practical cases the off diagonal entries or cross terms of the upper left block in the transport matrix **(3-44)** can be neglected. In principle, these cross terms are caused by the momentum exchange between the electron and hole systems caused by electron-hole scattering; it is as if the electrons (holes) are dragged along with the holes (electrons). However, these effects are usually very small.

This leaves us to give a description of the diagonal terms of the upper left block and the entries in the upper right block of equation **(3-44)**. Note that for a cubic material, such as silicon, the entries in the transport matrix do not have to be tensors. Only when some symmetry-breaking external force is present, such as a magnetic field or mechanical stress, does the resulting anisotropy reflect itself in the tensorial behavior of the transport matrix. Since we have already assumed that no magnetic field or mechanical stress is present, we assume that the entries of the transport matrix are scalars. Thus we can link the above abstract formulas to some well-known experimental parameters, as shown below

$$\hat{a}^{ee} = q^{-1} n^e \mu^e T \qquad \textbf{(3-46a)}$$

$$\hat{a}^{hh} = q^{-1} n^h \mu^h T \qquad \textbf{(3-46b)}$$

$$\hat{a}^{eu} = n^e \mu^e T^2 P^e \qquad \textbf{(3-46c)}$$

[28] This assumption is valid when the carrier densities are not too high (< 10^{17} cm^{-3})

$$\hat{a}^{hu} = n^h \mu^h T^2 P^h \tag{3-46d}$$

where the experimental parameters μ^e (> 0) and μ^h (> 0) are defined as the electron and hole mobilities, and P^e (> 0)[29] and P^h (> 0) as the electron and hole thermoelectric powers. According to the general principles outlined in Section 2.3, we have written these parameters as functions of the intensive state parameters and the thermodynamic forces. These dependencies are determined by carrier-ionized impurity, carrier-neutral impurity, carrier-phonon and carrier-carrier scattering processes. For the readers convenience we list some of the more well-known models for the mobilities and thermoelectric powers.

Mobility

When going through the literature, many papers can be found which try to express the mobilities in terms of the intensive state parameters and the forces. A very popular model which is commonly used for simulation purposes is the empirical model described in [Caughey 1967]. It includes the effects of phonon and impurity scattering and can be summarized in the following relations

$$\mu^k_{phon,\,imp} = \mu^k_{min} + \left[\frac{\mu^k_{phon} - \mu^k_{min}}{1 + [n^{scat}/N^k_{ref}]^{a^k}}\right] \tag{3-47a}$$

$$\mu^k_{min} = \mu^k_{0,\,min}\left[\frac{T}{T_{ref}}\right]^{-\eta^k_2} \tag{3-47b}$$

$$\mu^k_{phon} = \mu^k_{0,\,phon}\left[\frac{T}{T_{ref}}\right]^{-\eta^k_1} \tag{3-47c}$$

$$N^k_{ref} = N^k_{0,\,ref}\left[\frac{T}{T_{ref}}\right]^{\eta^k_3} \tag{3-47d}$$

$$a^k = a^k_0\left[\frac{T}{T_{ref}}\right]^{-\eta^k_4} \tag{3-47e}$$

$$n^{scat} = \sum_m n^{d_m} + \sum_n n^{a_n} \tag{3-47f}$$

where the index k can take the values e and h, $\mu^k_{0,min}$ is the minimum mobility as expected at the highest doping densities, $\mu^k_{0,phon}$ is the maximum mobility as expected at the lowest doping densities, N^k_{ref} is a reference concentration, T_{ref} is the reference temperature of 300K, n^{scat} is the total

[29] Note that the definition of the sign of P^e is the opposite of what is usually encountered.

number of ionized scattering centers, which at room temperature is equal to the total impurity concentration.

The empirical parameters must be obtained by fitting the model to experimental data. A table of commonly used values for silicon is given below.

Table 3-3 Coefficients of the Caughey-Thomas model for phonon-impurity scattering.

k	$\mu_{0,min}$	$\mu_{0,phon}$	$N_{0,ref}$	a_0	η_1	η_2	η_3	η_4
electron	88	1340	$1.26\ 10^{17}$	0.88	2.33	0.57	2.4	0.146
hole	54	461	$2.35\ 10^{17}$	0.88	2.23	0.57	2.4	0.146

Two important deficiencies of the Caughey-Thomas model can be identified. First, the effect of carrier-carrier scattering and, second, the effect of velocity saturation of the electrons and holes.

Carrier-carrier scattering is the cause of a reduction in mobility at high injection levels, for instance, injection across a PN junction. A common approach in modeling the effect of carrier-carrier scattering on mobility is to use a modified Caughey-Thomas model where the total number of scattering centers in equation (3-47f) is replaced by an effective number also involving the carrier concentrations, that is

$$n_{eff}^{scat} = \alpha n^{scat} + (1-\alpha)\ (n^e + n^h) \tag{3-48}$$

According to [Engl 1981] the parameter α equals 0.34 for silicon.

The velocity saturation effect can be qualitatively explained as follows. In our line of reasoning we have implicitly assumed that the continuous feed of momentum and energy to the electrons and the holes by the application of an external electrostatic field is quickly released to the lattice by means of collisions, that is, upon the application of an electric field the carriers quickly reach a steady state velocity proportional to the applied field. However, for increasing electrostatic fields the carriers, instead of quickly being accelerated to their new steady state velocity, are now heated up. In this situation the carriers are no longer in thermal equilibrium with the lattice. The effect can be summarized in the following empirical formulas

$$\mu^k = \frac{\alpha^k \mu^k_{phon,\,imp}}{\delta^k + \left[1 + \left(\frac{\alpha^k \mu^k_{phon,\,imp} E_{eff}}{v^k_{sat}}\right)^{\beta^k}\right]^{1/\beta^k}} \tag{3-49a}$$

$$E_{eff} = \frac{n^k_\alpha \partial_\alpha \varphi}{\left|n^k_\alpha\right|} \tag{3-49b}$$

$$v^k_{sat} = v^k_{0,\,sat}\left[\frac{T}{T_{ref}}\right]^{-\gamma} \tag{3-49c}$$

where E_{eff} is the effective applied electric field, and $v^k_{0,\,sat}$ the saturation velocity at the reference temperature. Again the empirical parameters must be obtained by fitting the model to experimental data. A table of commonly used values for silicon is given below.

Table 3-4 Coefficients of the velocity saturation model.

k	α	β	δ	$v_{0,\,sat}$[cm/s]	γ
electrons	2	2	1	$1.07\ 10^7$	0.87
holes	1	1	0	$8.34\ 10^6$	0.52

Thermoelectric Power

Let us now continue with the thermoelectric powers P^e and P^h. A very clear qualitative description can be found in [Middelhoek 1989b]. For a non-degenerate semiconductor with dominant scattering mechanism, described by, $\tau \sim E^{-s}$, we may use the following simple relations

$$P^e = +\frac{k}{q}\left[\left(\frac{E^{cb} - \nu^e}{kT}\right) + \frac{5}{2} - s\right] \tag{3-50a}$$

$$P^h = +\frac{k}{q}\left[\left(\frac{\nu^h - E^{vb}}{kT}\right) + \frac{5}{2} - s\right] \tag{3-50b}$$

For acoustic phonon scattering, the coefficient s is equal to 1/2 and for ionized impurity scattering it is equal to -3/2 [Nag 1972][Nag 1980]. The correspondence of the above relations to experiment is reasonable, provided the sample is non-degenerate and the operating temperature is not too low. At low temperatures, the discrepancy between theory and experiment is attributed to the phonon drag effect. The phonon drag effect causes the thermoelectric power to increase rapidly at low temperatures [Geballe 1955]. The phonon drag effect is believed to be caused by the increased mechanical coupling between the electron gas and the (lattice) phonon gas, at low temperatures. In the scattering process between the charge carriers and the (thermal) phonons, the exchange of momentum is not completely randomized, therefore, because of the much larger effective mass of the phonons, some of the momentum of the phonons is transferred to the electrons and thus the electrons are dragged along with the phonon flux.

Usually, the phonon drag effect is taken into account by adding an extra term between the brackets of equations (3-50a) and (3-50b), however, we will not do so because a useful model of the phonon drag contribution is not yet available. An extension to equations (3-50a) and (3-50b), which makes these also valid in the case of a degenerate semiconductor, can be found in [Nakwaski 1983]. However, this model seems computationally too involved to be of practical use.

3.4.2.1.2 Thermal Domain

With respect to the lower left block of the transport matrix describing the cross effects between the thermal domain and the electrical domain, we can be brief. Because of the supposed symmetry of the transport matrix, the following relations must hold

$$\hat{a}^{ue} = \hat{a}^{eu} = n^e \mu^e P^e T^2 \tag{3-51a}$$

$$\hat{a}^{uh} = \hat{a}^{hu} = n^h \mu^h P^h T^2 \tag{3-51b}$$

This leaves only the lower right entry of the transport matrix to be identified. This entry represents the relation between the gradient of the temperature and the total heat flux in the lattice, electron and hole system, when the gradients in the electrochemical potentials are zero. To make this term somewhat more transparent we proceed as follows. First, we calculate the heat flux in the case when both the electron and hole particle fluxes are zero, then we have

$$h_\alpha \bigg|_{\substack{n^e_\alpha = 0 \\ n^h_\alpha = 0}} = \left[\left(\frac{\hat{a}^{ue} \hat{a}^{eu}}{\hat{a}^{ee}} \right) + \left(\frac{\hat{a}^{uh} \hat{a}^{hu}}{\hat{a}^{hh}} \right) - \hat{a}^{uu} \right] \left(\frac{\partial_\alpha T}{T^2} \right) = -\kappa_t \cdot \partial_\alpha T \tag{3-52}$$

where κ_t is defined as the total heat conductivity in the absence of the particle fluxes. Note that κ_t is a measurable quantity[30] and is the sum of the lattice thermal conductivity and the (diffusion) thermal conductivities of the electron and hole system. Using the above relation and the original definition of the heat flow as implied by equation (3-44) we can write $\hat{a}^{uu}$ as

$$\hat{a}^{uu} = T^2 [\kappa_t + q n^e \mu^e (P^e)^2 T + q n^h \mu^h (P^h)^2 T] = T^2 \kappa_t^{eff} \tag{3-53}$$

A simple model for calculating κ_t can be found in [Prakash 1978][Glassbrenner 1964] and is given below

$$\kappa_t = \kappa_{0,t} \left[1 - \alpha \left(\frac{T - T_0}{T} \right)^{\beta} \right] \tag{3-54}$$

[30] This is the quantity that is usually measured in thermal conductivity experiments.

For silicon: $\alpha = 1.093$, $\beta = 0.8705$ and $\kappa_{0,\,l} = 1.56$ W/cm K. Note that at room temperature and with rather large estimates for the mobilities, the thermoelectric powers, and carrier densities we can neglect the second and third term within the brackets of equation (3-53), hence $\kappa_l = \kappa_l^{eff}$. Only in the case of extreme high carrier densities (> 10^{21}) some effects will be noticeable. At such high carrier densities we should also extend the theory to include heavy doping effects [Wachutka 1990].

This concludes the discussion on the non-equilibrium vectorial phenomena, next we discuss the scalar phenomena.

3.4.2.2 Scalar Phenomena

The purpose of this subsection is to set up the general form of the non-equilibrium equations of state for the scalar fluxes as given in Table 3-1. As in the case of the vectorial fluxes, we assume that the lattice, electron and hole temperatures are equal. In this case the heat fluxes between the lattice, electron and hole systems vanish. We also assume that the scalar phenomena do no not couple to the vectorial phenomena discussed in the previous section. For large electric fields this is not a realistic assumption, because of impact ionization or avalanche multiplication. Within the realm of the present framework, these effects can be interpreted as a coupling between the scalar fluxes and the vectorial forces. This leaves us to give a description of the scalar carrier fluxes between the electron, hole and localized state systems.

Usually, these scalar fluxes are interpreted as pseudo chemical reactions taking place between the free charge carriers and the localized states in the band gap. In the literature, these are also commonly referred to as generation/recombination processes. Recapitulating, we have (1) a net flux of electrons from the conduction to the valence band, (2) a net flux of electrons from the conduction band to the donor-like localized states, (3) a net flux of electrons from the conduction band to the acceptor-like localized states, (4) a net flux of holes from the valence band to the donor-like localized states and (5) a net flux of holes from the valence band to the acceptor-like localized states. The above local fluxes are schematically represented in Figure 3-1.

As stated earlier, the above point of view ignores the possible interaction (electron fluxes) between the localized states in the band gap. For complex localized states configurations this is not a valid assumption, especially in the case of multiple interacting multivalent flaws [Landsberg 1991]. In [Otaredian 1992] it was shown that even for high quality processed silicon semiconductors the assumption of non-interacting localized states does not always hold.

According to [Landsberg 1991], each of the above-mentioned transition rates consists of contributions of transitions involving two particles and transitions involving three particles. The three-particle processes are usually

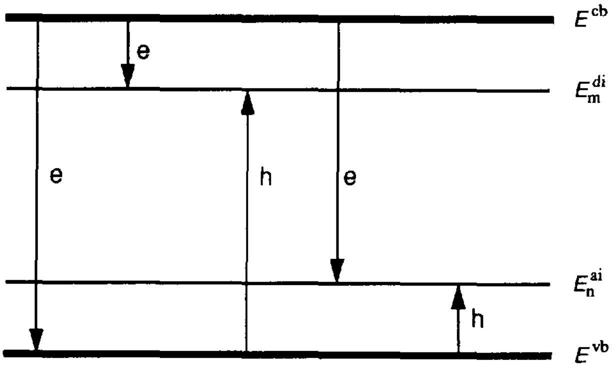

Figure 3-1 Pseudo chemical reactions occurring in semiconductors represented in a simple band diagram.

called Auger transitions. For instance, in the case of a band-to-band Auger transition, an electron drops from the conduction band to the valence band thereby transferring its excess energy to another electron in the conduction band. This electron is thus brought into a higher energy level of the conduction band, where it subsequently loses its excess energy because of scattering. The other Auger processes all operate according to the same principle.

The band-to-band transitions can be written in terms of chemical reaction equations

$$e + h \Leftrightarrow e^{vb} \tag{3-55a}$$

$$e + e + h \Leftrightarrow e^{vb} + e \tag{3-55b}$$

$$e + h + h \Leftrightarrow e^{vb} + h \tag{3-55c}$$

Similarly, the transitions of electrons from the conduction band to donor-like localized states can be written as

$$e + d_m^+ \Leftrightarrow d_m^\times \tag{3-56a}$$

$$e + e + d_m^+ \Leftrightarrow d_m^\times + e \tag{3-56b}$$

$$e + h + d_m^+ \Leftrightarrow d_m^\times + h \tag{3-56c}$$

For the transitions of electrons from the conduction band to acceptor-like localized states we have

$$e + a_n^+ \Leftrightarrow a_n^- \tag{3-57a}$$

$$e + e + a_n^{\times} \Leftrightarrow a_n^{-} + e \qquad \text{(3-57b)}$$

$$e + h + a_n^{\times} \Leftrightarrow a_n^{-} + h \qquad \text{(3-57c)}$$

For the transition of holes from the valence band to donor-like localized states we have

$$h + d_m^{\times} \Leftrightarrow d_m^{+} \qquad \text{(3-58a)}$$

$$h + e + d_m^{\times} \Leftrightarrow d_m^{+} + e \qquad \text{(3-58b)}$$

$$h + h + d_m^{\times} \Leftrightarrow d_m^{+} + h \qquad \text{(3-58c)}$$

Finally, for the transition of holes from the valence band to acceptor-like localized states we have

$$h + a_n^{-} \Leftrightarrow a_n^{\times} \qquad \text{(3-59a)}$$

$$h + e + a_n^{-} \Leftrightarrow a_n^{\times} + e \qquad \text{(3-59b)}$$

$$h + h + a_n^{-} \Leftrightarrow a_n^{\times} + h \qquad \text{(3-59c)}$$

To also assume linear constitutive relations between the scalar fluxes and forces, just as in the case of the vectorial phenomena, is not a good approximation, except in the case of minute excursions from the thermal equilibrium state. Further, because of the assumption of non-interacting localized states the matrix relating the local fluxes to the forces is diagonal, hence

$$\begin{bmatrix} \dot{n}^{e \to h} \\ \dot{n}^{e \to d_m} \\ \dot{n}^{e \to a_n} \\ \dot{n}^{h \to d_m} \\ \dot{n}^{h \to a_n} \end{bmatrix} = \begin{bmatrix} \hat{a}^{eh} & 0 & 0 & 0 & 0 \\ 0 & \hat{a}^{ed_m} & 0 & 0 & 0 \\ 0 & 0 & \hat{a}^{ea_n} & 0 & 0 \\ 0 & 0 & 0 & \hat{a}^{hd_m} & 0 \\ 0 & 0 & 0 & 0 & \hat{a}^{ha_n} \end{bmatrix} \cdot \frac{1}{T} \begin{bmatrix} \nu^{e} - \nu^{h} \\ \nu^{e} - \nu^{d_m} \\ \nu^{e} - \nu^{a_n} \\ \nu^{h} - \nu^{d_m} \\ \nu^{h} - \nu^{a_n} \end{bmatrix} \qquad \text{(3-60)}$$

where it is understood that the matrix entries are non-linear operators operating on the right-hand side vector, and the electrochemical potentials refer to the electronic energy scale. Note that the matrix entries must be chosen such that the entropy production $\dot{s}^{scalar}$ is positive, hence $\hat{a}^{kl} \geq 0$. Because the scalar fluxes are supposed to vanish at thermal equilibrium, we expect the scalar fluxes to be vanishing functions of the differences between the electrochemical potentials. Skipping the tedious details [Landsberg 1991] we end up with the following general model

$$\dot{n}^{k\to s} = c^{k\to s}\cdot n^k\cdot n^s\cdot\left[1-\exp\left(\frac{v^s - v^k}{kT}\right)\right] \qquad \textbf{(3-61a)}$$

where the reaction coefficients c are defined as,

$$c^{k\to s} = c_0^{k\to s} + n^e c_1^{k\to s} + n^h c_2^{k\to s} \qquad \textbf{(3-61b)}$$

and it is understood that

$$k, s \in \{e, h, d_m, a_n\} \qquad \textbf{(3-61c)}$$

The above equations describe band-to-band transitions, Auger-assisted band-to-band transitions, band-to-localized-states transitions, and Auger-assisted band-to-localized-states transitions. Note that for very small deviations from thermal equilibrium, equation **(3-61a)** indeed reduces to linear constitutive equations just as in the case of the vectorial fluxes. Often, the expansion coefficients in the reaction coefficients **(3-61b)** can be treated as constants, however, for degenerate semiconductors, they depend at least on the carrier concentrations. Here we treat the expansion coefficients as empirical parameters which have to be determined for each material and each type of localized state.

In principle, if the reaction coefficients c are known, we have a closed computational model with respect to the scalar fluxes. However, we can go one step further and assume steady-state conditions with respect to the scalar fluxes. Steady-state implies that the time rates occurring in the balance equations **(3-12c)** and **(3-12d)** are supposed to vanish, hence

$$\dot{n}^{e\to d_m} = -\dot{n}^{h\to d_m} \qquad \textbf{(3-62a)}$$

$$\dot{n}^{e\to a_n} = -\dot{n}^{h\to a_n} \qquad \textbf{(3-62b)}$$

According to [Landsberg 1991] the characteristics of band-to-localized-states transitions then reduce to a generalized Shockley-Read-Hall recombination process, given by

$$\dot{n}^{e\to d_m} = \frac{n^e n^h\left[1-\exp\left(\frac{v^h - v^e}{kT}\right)\right]}{\tau^e_{d_m}\,[n^h + n^h_1] + \tau^h_{d_m}\,[n^e + n^e_1]} \qquad \textbf{(3-63a)}$$

where

$$\tau^e_{d_m} = [n^{d_m} c^{e\to d_m}]^{-1} \qquad \textbf{(3-63b)}$$

$$\tau^h_{d_m} = [n^{d_m} c^{d_m\to h}]^{-1} \qquad \textbf{(3-63c)}$$

$$n_1^e = n^e\left(\frac{n^{d_m} - n^{d_m^+}}{n^{d_m^+}}\right)\exp\left(\frac{\nu^{d_m} - \nu^e}{kT}\right) \tag{3-63d}$$

$$n_1^h = n^h\left(\frac{n^{d_m} - n^{d_m^+}}{n^{d_m^+}}\right)^{-1}\exp\left(\frac{\nu^h - \nu^{d_m}}{kT}\right) \tag{3-63e}$$

A similar expression holds for an acceptor-like localized state. In contrast to the standard SRH model, the above model also accounts (if necessary) for direct band-to-band transitions and Auger-assisted transitions. Note that the assumption of the steady-state condition is only valid if the recombination process is much faster than the other processes taking place. The recombination process usually takes place on a time scale of micro-seconds, hence, for fast transient simulations, this assumption likely no longer holds.

3.5 Scaled Thermoelectric Model Equations

In this section, we summarize a *scaled* version of the *simplified* thermoelectric model. In order to keep things tractable we use the following simplifications: (a) a single shallow donor level of density n^d, which is in thermodynamic equilibrium with the conduction band, (b) a single shallow acceptor level of density n^a, which is in thermodynamic equilibrium with the valence band, and (c) a single steady-state deep-level trap of density n^t, for which the generalized SRH model can be assumed to hold. Taking into account more complex doping and deep-level traps configurations is not expected to add significant difficulties other than implementation overhead.

Basically, the scaling of the model equations is used to serve three purposes: (a) to avoid numerical overflow, (b) to reduce the number of floating point operations, and (c) to keep the order of magnitudes of the parameter of the PDE in the same range. Obviously, this involves a compromise, since it is difficult to satisfy all requirements simultaneously. In respect of requirement (a) we can state that with the current generation of computers (work stations) with floating point co-processors with standard double precision arithmetic, the danger of numerical overflow is significantly reduced. Hence, we can focus on requirements (b) and (c). Various scaling methods are known [Markowich 1990], however, for the full thermoelectric model the usual scaling methods tend to complicate the model equations because the temperature now is a degree of freedom and not a constant. We use the following almost trivial scaling (probably does not work at low temperatures)

$$\bar{n}_\alpha^e = +qn_\alpha^e \qquad \bar{n}_\alpha^h = +qn_\alpha^h \tag{3-64a}$$

$$\bar{T} = \frac{kT}{q} \qquad \bar{E}_f^e = \frac{E_f^e}{q} \qquad \bar{E}_f^h = \frac{E_f^h}{q} \qquad \bar{\varphi} = \varphi \tag{3-64b}$$

$$\bar{n} = qn \qquad \bar{p} = qp \qquad \bar{n}^d = qn^d \qquad \bar{n}^a = qn^a \qquad \bar{n}^t = qn^t \tag{3-64c}$$

$$\bar{P}^e = \frac{qP^e}{k} \qquad \bar{P}^h = \frac{qP^h}{k} \qquad \bar{\kappa}_t^{eff} = \frac{q\kappa_t^{eff}}{k} \tag{3-64d}$$

$$\bar{d}_\alpha = \varepsilon_0^{-1} d_\alpha \qquad \bar{\xi} = \varepsilon_0^{-1}\xi \tag{3-64e}$$

where it is assumed that the quantities with the overbars are the scaled quantities. Note that $\varepsilon = \varepsilon_0\varepsilon_r$, where ε_0 is the permittivity of the vacuum and ε_r the relative permittivity of the material. Now all equations proposed earlier in this chapter need to be rephrased in terms of the scaled quantities. Clearly, this is a rather trivial task and therefore we only state the main results.

In Section 3.5.1, we summarize the scaled bulk equations. In Section 3.5.2 the thermodynamic equilibrium solution is discussed. Finally, in Section 3.5.3, the boundary conditions are discussed.

3.5.1 Scaled Bulk Equations

The scaled Poisson equation, and the scaled electron, hole and heat balances are taken as

$$0 = -\partial_\alpha \bar{d}_\alpha + \bar{\xi} \tag{3-65a}$$

$$\partial_t \bar{n}^e = -\partial_\alpha \bar{n}^e_\alpha + \dot{\bar{n}}^e \tag{3-65b}$$

$$\partial_t \bar{n}^h = -\partial_\alpha \bar{n}^h_\alpha + \dot{\bar{n}}^h \tag{3-65c}$$

$$\partial_t h = -\partial_\alpha h_\alpha + \dot{h} \tag{3-65d}$$

The corresponding non-equilibrium constitutive equations are taken as

$$d_\alpha = -\varepsilon_r \partial_\alpha \varphi \tag{3-66a}$$

$$\bar{n}^e_\alpha = -\bar{n}^e \mu^e [\partial_\alpha \bar{E}^e_f + \bar{P}^e \partial_\alpha \bar{T}] \tag{3-66b}$$

$$\bar{n}^h_\alpha = +\bar{n}^h \mu^h [\partial_\alpha \bar{E}^h_f - \bar{P}^h \partial_\alpha \bar{T}] \tag{3-66c}$$

$$h_\alpha = [-\bar{n}^e \mu^e \bar{T}\bar{P}^e]\,\partial_\alpha \bar{E}^e_f + [\bar{n}^h \mu^h \bar{T}\bar{P}^h]\,\partial_\alpha \bar{E}^h_f - \bar{\kappa}^{eff}_t \partial_\alpha \bar{T} \tag{3-66d}$$

The source terms are taken as

$$\xi = [\bar{n}^h - \bar{n}^e + \bar{n}^{d^+} - \bar{n}^{a^-}] \tag{3-67a}$$

$$\dot{\bar{n}}^e = \dot{\bar{n}}^h = -\left(\frac{\bar{n}^e\bar{n}^h\left[1-\exp\left(\frac{\bar{E}_f^h-\bar{E}_f^e}{\bar{\bar{T}}}\right)\right]}{\tau^e\,[\bar{n}^h+\bar{n}_t^h]+\tau^h\,[\bar{n}^e+\bar{n}_t^e]}\right) \tag{3-67b}$$

$$\dot{h} = [-\bar{n}_\alpha^e\partial_\alpha\bar{E}_f^e+\bar{n}_\alpha^h\partial_\alpha\bar{E}_f^h-\dot{\bar{n}}^e\bar{E}_f^e+\dot{\bar{n}}^h\bar{E}_f^h] \tag{3-67c}$$

where

$$\tau^e = \frac{1}{c^{e\to t}n^t} \qquad \tau^h = \frac{1}{c^{t\to h}n^t} \tag{3-68}$$

$$\bar{n}_t^e = \bar{n}^e\mathrm{e}^{\frac{\bar{E}^t-\bar{E}_f^e}{\bar{\bar{T}}}} \qquad \bar{n}_t^h = \bar{n}^h\mathrm{e}^{\frac{\bar{E}_f^h-\bar{E}^t}{\bar{\bar{T}}}} \tag{3-69}$$

Finally, the equilibrium equations of state are taken as

$$\bar{n}^e = \bar{N}^{cb}(\bar{T})F_{1/2}\left(\frac{-\,(1/2)\,E^{gp}+\,(\bar{E}_f^e+\varphi)}{\bar{\bar{T}}}\right) \tag{3-70a}$$

$$\bar{n}^h = \bar{N}^{vb}(\bar{T})F_{1/2}\left(\frac{-\,(1/2)\,E^{gp}-\,(\bar{E}_f^h+\varphi)}{\bar{\bar{T}}}\right) \tag{3-70b}$$

$$\bar{n}^{d^+} = \bar{n}^d\left(\frac{1}{1+2\exp\left(\frac{\bar{E}_f^e-\bar{E}^d}{\bar{\bar{T}}}\right)}\right) \tag{3-70c}$$

$$\bar{n}^{a^-} = \bar{n}^a\left(\frac{1}{1+4\exp\left(\frac{\bar{E}^a-\bar{E}_f^h}{\bar{\bar{T}}}\right)}\right) \tag{3-70d}$$

In the above model equations we assume that the mobilities μ^k, the thermoelectric powers P^k and the reaction coefficients $c^{k\to l}$ are constants. Taking into account the more complex models, as discussed in this chapter, is not expected to add significant difficulties other than implementation overhead.

3.5.2 Thermodynamic Equilibrium

The thermodynamic equilibrium solution is defined by setting the Dirichlett and Neumann boundary conditions (cf. Chapter 2) to zero. Essentially, this action forces the device into the state of thermodynamic equilibrium. Since, no currents flow through the device, the Fermi-levels as well as the temperature must be constant. In particular, the temperature is set to the ambient temperature and the Fermi-levels are set to zero. This means that we only need to solve for the electrostatic potential at thermodynamic equilibrium. For this we need to solve the non-linear Poisson equation. However, for uniformly doped semiconductors with completely ionized doping levels, the trivial solution to the Poisson equation is found by setting the space charge density to zero and solving for the built-in electrostatic potential

$$\varphi_{bi} = (\frac{kT}{q})\ln\left[\frac{1}{2}\left(\left(\frac{n^d - n^a}{N^{cb}}\right)e^{\frac{E^{gp}}{2kT}} + \sqrt{\left(\frac{n^d - n^a}{N^{cb}}\right)^2 e^{\frac{E^{gp}}{kT}} + 4\left(\frac{N^{vb}}{N^{cb}}\right)}\right)\right] \quad \textbf{(3-71)}$$

Note that for uniform temperature, completely ionized uniform doping, and position independent bandgap, the built-in potential φ_{bi} is indeed the trivial solution of the Poisson equation. In the case the doping levels are not completely ionized the above equation does not hold and φ_{bi} needs to be calculated (iteratively) from the full form of the zero space charge condition (cf. equation **(3-21a)**). If φ_{bi} also becomes position dependent we cannot escape the solution of the non-linear Poisson equation, however, we can use φ_{bi} as a good initial guess.

Note 3-8: Although equation **(3-71)** arises naturally from the zero space charge condition it is not numerically well conditioned, e.g. equation **(3-71)** is not usable at low (77 K) temperatures due to cancelation errors. However, by means of some algebraic tricks it is possible to rewrite equation **(3-71)** to

$$\varphi_{bi} = E_i + \varphi^*_{bi} \quad \textbf{(3-72)}$$

with

$$E_i = \frac{1}{2}(\frac{kT}{q})\ln\left[\frac{N^{vb}}{N^{cb}}\right] \qquad \varphi^*_{bi} = (\frac{kT}{q})\,\mathrm{asinh}\left[\frac{n^d - n^a}{2n_i}\right] \quad \textbf{(3-73)}$$

and

$$n_i = \sqrt{N^{cb}N^{vb}}e^{-\frac{E^{gp}}{2kT}} \quad \textbf{(3-74)}$$

Clearly E_i, n_i can respectively be recognized as the intrinsic Fermi-level and the intrinsic carrier density. Moreover, φ^*_{bi} is the definition of the built-in potential one usually encounters in the standard treatment of semiconductor theory. The above set of equations does not suffer from cancelation errors and performs very well at room temperature (300 K) as well as liquid nitrogen temperature (77 K).

3.5.3 Boundary Conditions

For simplicity we assume that the silicon semiconductor is embedded in oxide and metal surface layers. Hence, we only have to deal with silicon-metal interfaces and silicon-oxide interfaces. We deal with these interfaces by specifying boundary conditions (cf. Chapter 2).

At the metal-silicon interfaces we assume ideal properties, that is, thermal equilibrium and zero space charge. At the voltage controlled metal-silicon interfaces the following Dirichlett boundary conditions are used

$$\varphi = \varphi_{bi} \qquad \bar{E}_f^e = -V_{app} \qquad \bar{E}_f^h = -V_{appl} \tag{3-75}$$

where the built-in potential φ_{bi} is obtained by setting the space charge to zero and solving for the electrostatic potential, and V_{app} is the applied bias voltage. Note that in this case the build-in potential is different from the one discussed in Section 3.5.2, because the Fermi-levels are now equal to the applied bias voltage.

At the current controlled metal-silicon interfaces the following Neumann boundary conditions are used

$$d_\alpha \cdot n_\alpha = 0 \qquad \xi_\alpha^e \cdot n_\alpha = J_{app}^e \qquad \xi_\alpha^h \cdot n_\alpha = J_{app}^h \tag{3-76}$$

At the silicon-oxide interfaces we also assume ideal properties, that is, no interface charge and no interface recombination effects (cf. Section 2.5). Moreover, we assume that the normal components of the dielectric flux density and the electron and hole current densities are zero, hence, the following Neumann boundary conditions hold

$$d_\alpha \cdot n_\alpha = 0 \qquad \xi_\alpha^e \cdot n_\alpha = 0 \qquad \xi_\alpha^h \cdot n_\alpha = 0 \tag{3-77}$$

This leaves us to specify the thermal boundary conditions. Also in this case we choose for the simplest versions. At the parts of the boundary that act as a heat-sink we specify the temperature (Dirichlett boundary). At those parts of the boundary where heat is extracted at a certain rate we specify the normal component of the heat flux (Neumann boundary).

3.6 Concluding Remarks

In this chapter we have presented a basic framework for the modeling of solid-state transducer configurations. It was argued and shown that the modeling framework can conveniently be based on the thermodynamics of irreversible processes (TIP). As such TIP provides a modular and extensible physical modeling paradigm of great power. In particular, the modeling of the thermal and electrical energy domain was thoroughly discussed, under the simplifying assumptions of

- fixed chemical composition
- isostress
- mechanical rigidity
- mechanical equilibrium
- zero magnetic field
- electrostatic conditions

Using the principles of TIP a closed mathematical model, in terms of balance equations, equilibrium equations of state, and non-equilibrium equations of state was derived. Although not explicitly stated the derived model conforms to the generic mathematical model as discussed in Chapter 2. Where appropriate, we have also compiled several useful (parametrized) constitutive models from the available literature, however, it should be mentioned that most of these models have not been tested for their compatibility with the available fabrication process. Essentially, this means that for each semiconductor fabrication process the parameters of each constitutive model should be calibrated to that particular process. This is an aspect often neglected in device modeling.

Future research in this field should obviously be directed towards the relaxation of the various simplifying conditions. This way we hopefully evolve towards a complete modular computer implementation of the model, which can be applied to various practical cases. As argued earlier the implementation of such a model should be modular, in a sense that the apparent complexity of the model should be configurable. Perhaps the rather new object oriented programming techniques could be applied for this purpose. Also the use of high level compilers that compile the actual simulation program from a high level problem definition language are expected to be useful.

Despite these interesting aspects we tend to digress from the present subject, because we feel that more substantial knowledge is necessary with respect to the implementation of the models. Therefore, in the following chapter the objective is to define a convenient framework for the discretization of the thermoelectric model equations. The proposed framework can later on be extended to cope with more complex physical models.

Only the ignorant exactly know what they are doing !

Chapter 4

A Mixed Finite Element Discretization Method for the Thermoelectric Problem

4.1 Introduction

In order to be able to calculate a numerical solution to a given model, the model first needs to be mapped on to an approximately equivalent discrete model. Basically, such a mapping is defined by a discretization of the spatial as well as the time coordinate. The purpose of this chapter is, therefore, to present a number of useful tools for the discretization of the thermoelectric model. Before we discuss the details, we give some general background information on spatial and time discretization.

Spatial Discretization

A bird's-eye view of the various methods that can be used for the purpose of discretization in space are, with increasing degree of complexity, the finite difference method (FDM), the finite volume method (FVM) [Wesseling 1992], the finite volume element method (FVEM) [McCormick 1989], the finite element method (FEM) [Hughes 1987] and the boundary finite element method (BFEM). In the remainder of this chapter, we use the FEM, because of its advanced status with respect to both theory and numerical experiment. In some cases the FVEM gives equivalent results and may also be considered as a serious candidate. The main difference is that the FVEM starts from a physical conservation law in its integral form [McCormick 1989], whereas the FEM starts from the physical conservation law in its differential form, from which a weak (variational) form is obtained by means of the (Petrov)-Galerkin method [Hughes 1987]. In a sense, the FEM can be regarded as more general because a particular finite volume discretization can usually be obtained as a special case of a finite element discretization.

The key idea behind the FEM is first to divide the computational domain into a number of subdomains, called the finite elements. These finite elements usually are edges in the case of 1D models, triangles and quadrilaterals for 2D models, and tetrahedrons, hexahedrons and wedges for 3D models. To complete the task of discretization, the restriction of the (yet unknown) exact global solution on each element is approximated by expanding it in terms of a set of local element basis functions[1]. By choosing appropriate basis functions the expansion coefficients may represent the (discrete) solution in, for instance, the vertices of the element. On each element one then can set up a local matrix equation in terms of the expansion coefficients. The global system of equations can be assembled from the individual element contributions. The global matrix equation is sparse and can be solved, for the global (approximate) solution, by a direct solution method of $O(n^2)$, where n is the number of interior elements. Essentially, this procedure transforms the space continuous problem into the algebraic (discrete) problem of determining the expansion coefficients.

The BFEM is rather different from the FEM, because it needs the problem to be reformulated into a boundary integral form, moreover, the spatial discretization only needs to be defined for the boundary and not for the interior. Apparently this reduces the problem size to a lower dimensional manifold, however, one must keep in mind that the resulting global matrix equation is not sparse, hence a direct solution method is of $O(n^3)$, where n is the number of boundary elements.

Within the context of the FEM a vast number of different discretization methods, differing in the way the original equations are approximated, can be found in the literature. However, this dissertation is exclusively based on the mixed finite element method and its hybridization (mixed and hybrid-mixed finite elements). This method is particularly interesting because it provides the possibility to independently approximate the thermodynamic state variables and the thermodynamic fluxes so that both can be approximated with the same degree of freedom.

Time Discretization

With respect to the discretization in time, several methods can be found in the literature. Roughly, we may distinguish between two fundamental classes, which are the time-stepping method (TSM) [Hughes 1987], and the waveform relaxation method[2] (WRM) [White 1985]. Of the TSM we mention the well-known generalized trapezoidal family of methods, with special cases: forward Euler (explicit scheme), Crank-Nicolson (midpoint rule) and backward Euler (implicit scheme). The explicit scheme is very efficient but not unconditionally stable, whereas the implicit scheme and Crank-Nicolson scheme are unconditionally stable[3] but less efficient. Moreover, the Crank-Nicolson scheme has an accuracy of $O(h^2)$, whereas the other two schemes have an accuracy of $O(h)$, where h is the size of the time step. In contrast to

[1] Usually, low-order polynomial functions, however, other elementary functions are also conceivable.

[2] Also called dynamic iteration or Picard-Lindelöf iteration.

[3] At least for linear problems they are.

the TSM, which calculates the solution at the current time step using the solution at the previous time step, the WRM starts with a global approximation of the solution in a specified time slot and subsequently improves this approximation by a global iterative method. The advantage of this method with respect to the TSM is that the WRM has the potential of a much better performance on multiprocessor systems[4], moreover, it does not suffer from error accumulation from previous time steps, as in the TSM. However, a major drawback is the need to store[5] the solution at all time steps in the time slot, whereas in the TSM, only the solution at the previous time step needs to be stored. In this chapter, for reasons of simplicity, we only briefly discuss the use of the TSM, in particular the Crank-Nicolson method.

Of course, the details of the above topics are well known within the context of the FEM[6], however, the implementation of specific problems, such as the one discussed in the previous chapter, still remains a challenge. In Section 4.2, we discuss our motivation for using the mixed finite element discretization method. We also present a comprehensible outline of this method. Next, in Section 4.3 the hybrid variant of the mixed method is discussed. In Section 4.4, we present a method for the time discretization of the mixed method. Next, Section 4.5 deals with the non-linear aspects of the model equations and how these can be incorporated into the mixed method. Finally, in Section 4.6 some concluding remark are given.

4.2 Mixed Finite Element Discretization

In this section we pay attention to the discretization of the spatial part of a model problem which reflects the essential features of the thermoelectric model equations as discussed in Chapter 2. Loosely, we may distinguish a number of essential steps:

- transform the original equations into a variational form
- discretize the variational equations
- partition the computational domain into a set of disjoint elements
- define a set of element basis functions
- expand the exact solution in terms of the basis functions
- set up the element contributions to the global system of equations
- set up the global system of equations

In the literature, an overwhelming amount of spatial discretization methods can be found that more or less follow the above procedure. However, one particularly neat method is the use of mixed finite elements. The use of this type of element dates back to 1967, where the term was first used on a course

[4] For instance, the Convex C3 series. However, it will only be a matter of a short time before PCs and work stations offer standard vector and parallel processing capabilities.

[5] In this context storing means to keep the data in core.

[6] For a prototype of a finite element program the reader may consult [Hughes 1987].

on "Variational and Matrix Methods in Structural Mechanics" at M.I.T. [Pian 1977]. Unfortunately, the available literature on the mixed finite element discretizations is very theoretical, which hampers its application to practical problems, such as the problem discussed in Chapter 3. In this section, we try to boil down this rather abstract theory to a more comprehensible, though not mathematically rigorous, form. For the more fundamental aspects, such as error estimates and stability criteria, of mixed finite element discretizations for elliptic and parabolic type of problems we refer to [Raviart 1977][Johnson 1981][Arnold 1985][Cristina 1987][Chou 1992].

Before we proceed to the theory let us give a list of the main heuristic reasons[7] for choosing the mixed method in relation to the thermoelectric problem:

- In the mixed method the second order elliptic partial PDE is formulated in terms of two first order PDEs. Mixed finite element discretizations, therefore, naturally fit the way we have represented the model equations in Chapter 2, that is, in terms of balance equations together with non-equilibrium constitutive equations.
- Instead of first eliminating the thermodynamic flux state variables from the original model equations and then approximating the intensive state variables in terms of conforming basis functions, as is done in a conforming finite element discretization, the mixed finite element discretization approximates both the intensive (or extensive) state variables and the thermodynamic fluxes separately, and retains the original form in which the problem is stated. Opposed to a conforming discretization, this allows the intensive state variables and the generalized fluxes to be approximated with the "same" degree, reflecting the equal importance of each.
- Mixed finite elements are current preserving, which means that the discretized problem, just like the continuous problem, obeys a (discrete) current conservation law. Obviously, this must be because in the mixed finite element discretization the original continuous balance equation is rewritten in a discrete form. In a conforming finite element discretization the current preservation property is lost, which in many cases may give rise to loss of accuracy and numerical instabilities.
- Particular mixed finite element discretizations provide discretizations for which the normal component of the flux on an interface between two elements is continuous. If the physics of the problem indicates that the normal component of the thermodynamic flux to be approximated is continuous when crossing an interface, using mixed finite elements is a great advantage, because then no special interface condition needs

[7] Some of these arguments may not be directly clear, however, as we proceed they will start to make more sense.

to be specified[8]. In the case of Si-Ge interfaces in semiconductor hetero structures this is very convenient.

- In contrast to the more conventional control volume approach (classical box method) [Polak 1987], the flux variables are well defined on the interior of an element instead of only at the edges. This makes it more easy to calculate right-hand sides which depend on the fluxes. This is particularly convenient in the case of the thermoelectric model, because it allows a consistent treatment of the source terms.
- Mixed finite element discretizations appear to be more accurate then the box method often used for the discretization of the semiconductor equations [Polak 1988].
- Mixed finite elements can be designed such that they exhibit upwind characteristics similar to the Scharfetter-Gummel discretization, often used in one-dimensional semiconductor device simulation [Brezzi 1989a][Brezzi 1989b].

Of course, the use of mixed-hybrid finite elements also has its difficulties. The major difficulty is to predict whether the model equations in mixed form satisfy the LBB (Ladyzhenskaya-Babuska-Brezzi) compatibility condition [Brezzi 1990], which ensures the theoretical stability of the model equations. The LBB condition depends on the operators involved in the model equations and in principle has to be checked for each different type of operator involved. In the discrete domain, the LBB condition requires the approximation of the intensive state variable and the flux variable to be compatible. This means that the discrete spaces in which both approximations are sought cannot be chosen truly independent, however, they can usually be chosen such that the degree of approximation for both state quantities is the same. Fortunately, for most elliptic PDEs the LBB condition can be proven to hold and during the course of this chapter we rely on these results.

4.2.1 A Model Problem

In this section, we discuss a model problem representative of the problem defined in Chapter 3. Note that for the sake of simplicity, this section only discusses the case of a single set of state variables, that is, a single potential and flux. Apart from some algebraic details this simplification is not a limiting factor.

We assume the problem is defined on a bounded computational domain Ω in R^d (d=1,2,3) with a smooth boundary $\partial\Omega$ The boundary consists of a part $\partial\Omega_D$, on which Dirichlett boundary conditions are prescribed, and a part $\partial\Omega_N$, on which Neumann boundary conditions are prescribed. Moreover, we as-

[8] Note, however, that in the case where the thermodynamic flux is not continuous, as for instance the normal component of the dielectric flux density upon crossing an interface containing surface charge, special measures must be taken.

sume that $\partial\Omega_D \cap \partial\Omega_N = 0$ and $\partial\Omega_D \cup \partial\Omega_N = \partial\Omega$. Further, the vector $\boldsymbol{n}$ is defined as the outward normal vector on the boundary $\partial\Omega$.

The model problem to be solved is represented by the following nonlinear parabolic PDE in mixed form[9]

$$\begin{bmatrix} a_{\alpha\beta}(x;\phi) & \partial_\alpha \\ \partial_\beta & d(x)\partial_t + c(x) \end{bmatrix} \cdot \begin{bmatrix} u_\beta(x,t) \\ \phi(x,t) \end{bmatrix} = \begin{bmatrix} 0 \\ f(x,t;\phi,\boldsymbol{u}) \end{bmatrix} \quad (x,t) \in \Omega \times [0,T] \qquad \textbf{(4-1a)}$$

with the initial condition and the Dirichlett and Neumann boundary conditions given by

$$\phi(x,0) = \phi_0(x) \quad x \in \Omega \qquad \textbf{(4-1b)}$$

$$\phi(x,t) = g_D(t) \quad (x,t) \in \partial\Omega_D \times [0,T] \qquad \textbf{(4-1c)}$$

$$u_\alpha(x,t)n_\alpha(x) = g_N(t) \quad (x,t) \in \partial\Omega_N \times [0,T] \qquad \textbf{(4-1d)}$$

In the above model, it is assumed that the symmetric part of the tensor function $a_{\alpha\beta}$ is uniformly positive definite, the functions c and d are smooth functions bounded below by a positive constant. Although not discussed in Chapter 3, the anti-symmetric part of the tensor function $a_{\alpha\beta}$ could arise from the application of a magnetic field or a mechanical stress field [Duyn 1991][Munter 1992][Callen 1960]. In the following, we try to retain a formulation where the tensor function $a_{\alpha\beta}$ is permitted to be asymmetric. However, the possible physical reasons for this are not discussed.

With respect to the physical interpretation of the above model, the symbol ϕ stands for some intensive thermodynamic state variable, for instance the electrostatic potential, the electron or hole Fermi level or the absolute temperature. The symbol u_β stands for the corresponding vectorial thermodynamic flux[10], for instance, the dielectric flux density, the electron flux, the hole flux or the heat flux. Using the results from Chapter 3, it should not be too difficult to figure out that, in fact, the above-stated problem is a simplified form of either the Poisson equation, the carrier continuity equations or the heat flow equations.

Note 4-1: The initial condition $\phi_0(x)$ as stated in equation **(4-1b)** is usually first calculated by solving the corresponding elliptic problem ($\partial_t = 0$) with appropriate Dirichlett and Neumann boundary conditions.

[9] In contrast to the usual formulation of elliptic second order partial differential equations, the mixed formulation states the problem in terms of two first order partial differential equations, which opens up many interesting algorithmic properties.

[10] Model equations in terms of tensorial thermodynamic state variables can also be casted in this form, however, because of the tensorial nature of the state variables, the numerical treatment requires some additional features not discussed in this work.

Note 4-2: In the following we shall discuss the so-called semi-discrete problem, where the discretization of the ∂_t operator is omitted. In other words we may either omit ∂_t from equation **(4-1a)** or just treat it as a symbolic constant. A method for dealing with the time-dependence is briefly discussed in Section 4.4.

Note 4-3: The coefficients appearing in the above model problem are at this stage treated as quasi non-linear. In principle, this means that the coefficients are calculated using the result of the previous iteration (successive substitution). A method for treating the non-linearity is discussed in Section 4.5.

4.2.2 Dual Mixed Variational Formulation

In order to exploit the discrete properties of the computational domain and to construct the finite element approximations, we recast the original formulation in equations **(4-1a)** to **(4-1d)** into a variational or weak form[11]. A popular method to achieve this is Galerkin's method for which the trial functions are chosen in the same spaces as the test or weight functions [Becker 1981][Hughes 1987]. For the above case, this method leads to a variational problem of the form[12] [Huijben 1987][Kaasschieter 1990]:

Find $(\boldsymbol{u}, \phi) \in V_*(\Omega) \times W(\Omega)$ such that

$$\begin{aligned} a(\boldsymbol{u}, \boldsymbol{v}) + b(\phi, \boldsymbol{v}) &= (g_D, \boldsymbol{v})_{\partial\Omega_D} \quad \forall \boldsymbol{v} \in V_N(\Omega) \\ b(\boldsymbol{u}, \psi) + c(\phi, \psi) &= (f, \psi)_\Omega \quad \forall \psi \in W(\Omega) \end{aligned} \tag{4-2}$$

For the dual mixed method, the function spaces V and W are defined as

$$V(\Omega) = H(\mathrm{div}\,;\Omega) \qquad W(\Omega) = L^2(\Omega) \tag{4-3}$$

Note 4-4: The notation V_* means that this function space satisfies the Neumann boundary conditions (if present) and the notation V_N means that this function space satisfies the homogeneous Neumann boundary conditions (if present). Both are subspaces of V, that is $V_* \subset V$ and $V_N \subset V$.

The bilinear mappings a: $V \times V \rightarrow R$, b: $V \times W \rightarrow R$ and c: $W \times W \rightarrow R$ appearing in equation **(4-7)** are defined as

[11] The weak or variational form also has the advantage of being capable of dealing with generalized functions, such as the Dirac delta function, in a natural way.

[12] For a notion of the various function spaces involved, the reader is referred to Appendix B.

$$a(\boldsymbol{u}, \boldsymbol{w}) = \int_{\Omega} (a_{\alpha\beta} u_{\beta}) w_{\alpha} dx \qquad b(\boldsymbol{u}, \xi) = -\int_{\Omega} (\partial_{\alpha} u_{\alpha}) \xi dx$$
$$c(\omega, \xi) = -\int_{\Omega} c\omega\xi dx \tag{4-4}$$

and the right-hand sides as

$$(g_D, \boldsymbol{v})_{\partial\Omega_D} = -\int_{\partial\Omega_D} g_D v_{\alpha} n_{\alpha} ds \qquad (f, \psi)_{\Omega} = -\int_{\Omega} f\psi dx \tag{4-5}$$

4.2.3 Discrete Mixed Variational Formulation

In this section, we discuss a semi-discrete formulation of the mixed variational problem stated in the previous section. In order to do so it is necessary to define finite-dimensional subspaces of the spaces $V(\Omega)$ and $W(\Omega)$. Formally, we may write these finite-dimensional subspaces as

$$\begin{aligned} V^h &\subset V \quad , (\dim(V^h) \ll \infty) \\ W^h &\subset W \quad , (\dim(V^h) \ll \infty) \end{aligned} \tag{4-6}$$

where h is some discretization parameter. Denoting the discretized quantities by a superscript h, the discrete variational problem can then be written as: find $(\boldsymbol{u}^h, \phi^h) \in V^h_* \times W^h$ such that

$$\begin{aligned} a(\boldsymbol{u}^h, \boldsymbol{v}^h) + b(\phi^h, \boldsymbol{v}^h) &= (g^h_D, \boldsymbol{v}^h)_{\partial\Omega^h_D} \quad \forall \boldsymbol{v}^h \in V^h_N \\ b(\boldsymbol{u}^h, \psi^h) + c(\phi^h, \psi^h) &= (f^h, \psi^h)_{\Omega^h} \quad \forall \psi^h \in W^h \end{aligned} \tag{4-7}$$

To bring the above discrete variational formulation to an equivalent set of matrix equations which can be tackled numerically, we must choose a finite basis for V^h and W^h, that is

$$\begin{aligned} V^h &= \mathrm{span}(\boldsymbol{w}_i) \quad ; i = 1, \ldots, I \\ W^h &= \mathrm{span}(\xi_j) \quad ; j = 1, \ldots, J \end{aligned} \tag{4-8}$$

Now we expand $\boldsymbol{u}^h$, $\boldsymbol{v}^h$, ϕ^h and ψ^h into their corresponding global basis functions, that is

$$\boldsymbol{u}^h = \sum_{i=1}^{I} u_i \boldsymbol{w}_i \qquad \boldsymbol{v}^h = \sum_{k=1}^{I} v_k \boldsymbol{w}_k$$
$$\phi^h = \sum_{j=1}^{J} \phi_j \xi_j \qquad \psi^h = \sum_{l=1}^{J} \psi_l \xi_l \tag{4-9}$$

Substituting equation **(4-9)** into equation **(4-7)** then, after rearranging terms and noting that the equations must be valid for any choice of v^h and ϕ^h, we obtain an equivalent matrix equation

$$\begin{bmatrix} A & B \\ B^T & C \end{bmatrix} \cdot \begin{bmatrix} U \\ \Phi \end{bmatrix} = \begin{bmatrix} G \\ F \end{bmatrix} \qquad \textbf{(4-10)}$$

where, the matrix and vector entries are given by

$$\begin{gathered} [U]_i = u_i \qquad [\Phi]_j = \phi_j \\ [A]_{ki} = a(w_k, w_i) \qquad [B]_{kj} = b(w_k, \xi_j) \qquad [C]_{ij} = c(\xi_i, \xi_j) \\ [G]_i = (g_D^h, w_i)_{\partial\Omega_D^h} \qquad [F]_j = (f^h, \xi_j)_{\Omega^h} \end{gathered} \qquad \textbf{(4-11)}$$

Note 4-5: In equation **(4-10)** the matrix A is positive definite and the matrix C is negative definite.

In the above formulation the Neumann boundaries have not been taken into account. This is because we have expanded both u^h and v^h in terms of the basis functions of V instead of V_* and V_N. In practice, this means that in equation **(4-10)**, the rows corresponding to the known degrees of freedom in the solution vector U should be removed from the system of equations, and the columns corresponding to the known degrees of freedom in the solution vector U should be brought to the right-hand side. We do not dwell on this because in the local relaxation solver, to be discussed in Chapter 5, this process is easily carried out locally.

Finally, since the matrix A is invertible, we can eliminate the solution vector U from equation **(4-10)**, that is

$$(B^T A^{-1} B - C)\,\Phi = B^T A^{-1} G - F \qquad \textbf{(4-12)}$$

After calculation of the solution vector Φ, the solution vector U can be reconstructed by using the relation

$$U = A^{-1}(G - B\Phi) \qquad \textbf{(4-13)}$$

4.2.4 Element Basis Functions

In this section, we address the choice of the basis functions w_i and ξ_j of the approximation function spaces V^h and W^h. For convenience, the problem is only stated in R^2. In principle, the equivalents in R^1 and R^3 can be obtained in the same way.

Let us assume that the set S^h is a quasi-regular partition of Ω such that $S \in S^h$ is either a triangle or a quadrilateral. It is assumed that: (1) all vertex angles satisfy some lower bound[13], that is $\theta > \theta_0 > 0$, and (2) the ratio of the length of the edges of the quadrilaterals is bounded below by a positive constant, and (3) h is a measure for the maximum size of the elements. The union of these elements is an approximation Ω^h to the computational domain Ω. The corresponding approximation to the boundary $\partial\Omega$ is denoted as $\partial\Omega^h$. The set E^h is the collection of all the edges $E \in E^h$ of the elements. For the sake of convenience we distinguish between the set of internal edges $E_I^h = \{E \in E^h |$ $E \notin \partial\Omega^h\}$ and the set of boundary edges $E_B^h = \{E \in E^h | E \in \partial\Omega^h\}$. Moreover, the edges lying on the Dirichlett part of the boundary are denoted by the set $E_D^h = \{E \in E^h | E \in \partial\Omega_D^h\}$. Similarly, the edges lying on the Neumann part of the boundary are denoted by the set $E_N^h = \{E \in E^h | E \in \partial\Omega_N^h\}$.

The general idea is to approximate the restrictions $\boldsymbol{u}_S(\boldsymbol{x}) = \boldsymbol{u}(\boldsymbol{x}) \bullet \Pi(\boldsymbol{x}; \boldsymbol{x} \in S)$ and $\phi_S(\boldsymbol{x}) = \phi(\boldsymbol{x}) \bullet \Pi(\boldsymbol{x}; \boldsymbol{x} \in S)$ of the global solution on each element in terms of an expansion in a set of local basis functions, usually low-order polynomials. The global basis for the approximation function spaces V^h and W^h can then be constructed from the local basis. In the literature, an abundant collection of possibilities can be found [Raviart 1977][Nedelec 1980][Brezzi 1985] [Nedelec 1986], which can all be fitted into the general framework discussed in Section 4.2.2 and Section 4.2.3.

4.2.4.1 Raviart-Thomas Basis Functions

The local Raviart-Thomas basis functions of order k for the vector and scalar degrees of freedom can be constructed according to the following rule

$$\begin{aligned} P^k(S) &= \text{Polynomials of degree k on S} \\ \tilde{P}^k(S) &= P^k(S) \times P^k(S) \\ RT^k(S) &= \{ u_S^h \,|\, u_S^h = p(x) + xq(x),\ \ p \in \tilde{P}^k(S),\ \ q \in P^k(S) \} \end{aligned} \quad \textbf{(4-14)}$$

which is valid for any integer $k \geq 0$ and for any domain $\Omega \subseteq \mathrm{R}^d$ d=1,2,3. The global basis can now be constructed as follows. First, we define the global Raviart-Thomas space obtained by patching together the local Raviart-Thomas spaces, that is

$$RT_{-1}^k(S^h) = \{ u^h \,|\, u^h \in L^2(\Omega), u_S^h \in RT^k(S) \quad \forall S \in S^h \} \quad \textbf{(4-15)}$$

Then in order to enforce the continuity of the normal component of the fluxes, the following subspace is defined

$$RT_0^k(S^h) = \{ u^h \,|\, u^h \in RT_{-1}^k(S^h), \text{the normal component is continuous across the interelement boundaries} \} \quad \textbf{(4-16)}$$

Finally, the multiplier space is defined as

[13] This condition is intended to restrict the use of degenerate triangles and quadrilaterals.

$$M_{-1}^{k}(S^{h}) = \{\phi^{h} \mid \phi^{h} \in L^{2}(\Omega), \phi_{S}^{h} \in P^{k}(S) \quad \forall S \in S^{h}\} \tag{4-17}$$

Using the above results the k'th order Raviart-Thomas mixed formulation is equivalent to equation **(4-7)**, however, with the abstract spaces V^{h} and W^{h} replaced by

$$\begin{aligned} V^{h}(\Omega) &= RT_{0}^{k}(S^{h}) \\ V_{*}^{h}(\Omega) &= RT_{0,*}^{k}(S^{h}) \\ V_{N}^{h}(\Omega) &= RT_{0,N}^{k}(S^{h}) \\ W^{h}(\Omega) &= M_{-1}^{k}(S^{h}) \end{aligned} \tag{4-18}$$

with

$$RT_{0,N}^{k}(S^{h}) = \{\boldsymbol{u}^{h} \mid \boldsymbol{u}^{h} \in RT_{0}^{k}(S^{h}),\ \boldsymbol{n} \cdot \boldsymbol{u}^{h} = 0 \ \ \forall E \in E_{N}^{h}\} \tag{4-19}$$

$$RT_{0,*}^{k}(S^{h}) = \{\boldsymbol{u}^{h} \mid \boldsymbol{u}^{h} \in RT_{0}^{k}(S^{h}),\ \boldsymbol{n} \cdot \boldsymbol{u}^{h} = g_{N}^{h} \ \ \forall E \in E_{N}^{h}\} \tag{4-20}$$

As an example we give the lowest order ($k = 0$) Raviart-Thomas basis functions on the reference triangle and quadrilateral[14].

reference triangle

From equation **(4-14)** we find that the approximation to the vector quantity $\boldsymbol{u}_S$ can be written as

$$\boldsymbol{u}_{S}^{h} = \begin{bmatrix} u_{1}^{h} \\ u_{2}^{h} \end{bmatrix} = \begin{bmatrix} a_{0} \\ a_{1} \end{bmatrix} + a_{2}\begin{bmatrix} x \\ y \end{bmatrix} \tag{4-21}$$

To ensure the continuity of the normal component of the flux across the inter element boundaries we proceed as follows. The approximation $\boldsymbol{u}_{S}^{h}$ is defined such that the restriction of its projection on to the normal vector of each of the edges is constant (cf. Figure 4-1), that is

$$\begin{aligned} (\boldsymbol{u}^{h} \cdot \boldsymbol{n}_{1})\big|_{E_{1}} &= -a_{1} \\ (\boldsymbol{u}^{h} \cdot \boldsymbol{n}_{2})\big|_{E_{2}} &= -a_{0} \\ (\boldsymbol{u}^{h} \cdot \boldsymbol{n}_{3})\big|_{E_{3}} &= \frac{1}{\sqrt{2}}(a_{0} + a_{1} + a_{2}) \end{aligned} \tag{4-22}$$

The net flux U_j crossing each edge E_j is formally given by

[14] As pointed out in Appendix C, it suffices to deal with the reference elements.

$$U_j = \int_{E_j} u^h \cdot n_j dS \tag{4-23}$$

Obviously, we then must have

$$\begin{aligned} U_1 &= -a_1 \\ U_2 &= -a_0 \\ U_3 &= a_0 + a_1 + a_2 \end{aligned} \tag{4-24}$$

Now by taking the net fluxes U_1, U_2 and U_3 as the basic degrees of freedom[15] and by assigning these to the midpoints of the edges (cf. Figure 4-1), the continuity of the normal component of the flux is guaranteed. That this must be the case is simply because of the fact that a degree of freedom defined on an edge is shared by its two neighboring elements.

Some care must be taken in defining the orientation of the net flux through an edge. In the above line of reasoning, the fluxes are by definition taken positive when directed outward from the reference element (cf. Figure 4-1), however, in the global mesh this can no longer hold because the flux through an edge is shared by the two elements that share that edge. Hence, some sign convention should be defined. We shall use the following convention. Each edge is defined by the nodes it connects and the orientation of the edge is from the first node to the second node. An element may use this edge in order to define its boundary. However, when an element uses an edge, a sign should be specified that indicates the orientation of the edge with respect to the orientation of the boundary of the element which is defined by traversing the edges of the element in an anti-clockwise fashion. This way elements that share an edge always do this with opposite signs. Hence, the sign can be

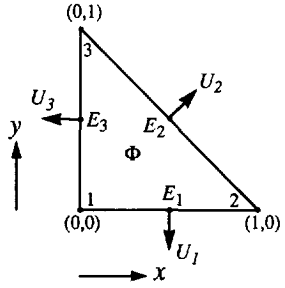

Figure 4-1 Definition of the local vertex numbering, local edge numbering and local degrees of freedom on a triangular reference element.

[15] Note that in contrast to some other approaches we take the net flux through an edge as the degree of freedom, not the flux density $U_j/|E_j|$.

used to uniquely define the orientation of the flux through an edge used by the element.

Taking the U_j as the basic degrees of freedom on the element, we may rewrite equation **(4-21)** as an expansion in terms of a set of local basis functions

$$\begin{bmatrix} u_1^h \\ u_2^h \end{bmatrix} = U_1 \begin{bmatrix} x \\ y-1 \end{bmatrix} + U_2 \begin{bmatrix} x \\ y \end{bmatrix} + U_3 \begin{bmatrix} x-1 \\ y \end{bmatrix} \tag{4-25}$$

Hence, the local basis functions (on the reference element) are identified as

$$\begin{aligned} \boldsymbol{W}_1 &= \begin{bmatrix} x\ y-1 \end{bmatrix}^{\mathrm{T}} \\ \boldsymbol{W}_2 &= \begin{bmatrix} x\ y \end{bmatrix}^{\mathrm{T}} \\ \boldsymbol{W}_3 &= \begin{bmatrix} x-1\ y \end{bmatrix}^{\mathrm{T}} \end{aligned} \tag{4-26}$$

The global approximation $\boldsymbol{u}^h(x)$ can now be written as

$$\boldsymbol{u}^h(x) = \sum_k U_k \boldsymbol{w}_k(x) \tag{4-27}$$

where k runs over all edges in the grid. The basis functions $\boldsymbol{w}_k(x)$ are constructed according to

$$\boldsymbol{w}_k(x) = \begin{cases} \mathrm{Sgn}(k,1)\boldsymbol{W}_{k,1} + \mathrm{Sgn}(k,2)\boldsymbol{W}_{k,2} & (E_k \in E_I^h) \\ \mathrm{Sgn}(k,1)\boldsymbol{W}_{k,1} \quad (E_k \in E_B^h) \end{cases} \tag{4-28}$$

where $\mathrm{Sgn}(k,j)$ is defined as the sign with which element S_j uses edge E_k to define its boundary, that is, if the sign is positive (negative) then the orientation of the edge is equal (opposite) to the orientation of the boundary of the element. Moreover, $\boldsymbol{W}_{k,1}$ and $\boldsymbol{W}_{k,2}$ are the basis functions associated with edge E_k of the two elements that have edge E_k in common.

According to equation **(4-17)**, the scalar quantity ϕ_S should be chosen constant on the interior of the element. This means that the global approximation ϕ^h is piecewise constant and can be written as

$$\phi^h(x) = \sum_k \Phi_k \xi_k(x) \tag{4-29}$$

where k runs over all elements in the grid and Φ_k is taken as the degree of freedom on element S_k, and $\xi_k(x)$ is the local basis function such that

$$\xi_i(x) = \begin{cases} 1 & x \in S_i \\ 0 & x \notin S_i \end{cases} \tag{4-30}$$

reference quadrilateral (parallelogram, rectangle, square)

In the case of a quadrilateral, the procedure is essentially the same. According to equation (4-14), the approximation to the vector quantity u_S can be written as

$$u_S^h = \begin{bmatrix} u_1^h \\ u_2^h \end{bmatrix} = \begin{bmatrix} a_0 \\ a_1 \end{bmatrix} + \begin{bmatrix} a_2 x \\ a_3 y \end{bmatrix} \tag{4-31}$$

Note that we now have four degrees of freedom instead of three as in the previous case. As in the previous case, the restriction of the projection of u_S^h onto the normal vector of each of the edges is constant (cf. Figure 4-2).

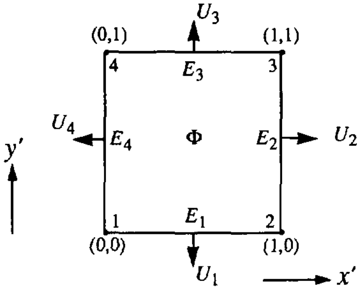

Figure 4-2 Definition of the local vertex numbering, local edge numbering and local degrees of freedom on the quadrilateral reference element.

The net fluxes across the edges can be calculated as

$$\begin{aligned} U_1 &= -a_1 \\ U_2 &= a_0 + a_2 \\ U_3 &= a_1 + a_3 \\ U_4 &= -a_0 \end{aligned} \tag{4-32}$$

Taking the U_j as the basic degrees of freedom on the element, we may rewrite equation (4-21) as an expansion in terms of the local basis functions

$$\begin{bmatrix} u_1^h \\ u_2^h \end{bmatrix} = U_1 \begin{bmatrix} 0 \\ y-1 \end{bmatrix} + U_2 \begin{bmatrix} x \\ 0 \end{bmatrix} + U_3 \begin{bmatrix} 0 \\ y \end{bmatrix} + U_4 \begin{bmatrix} x-1 \\ 0 \end{bmatrix} \tag{4-33}$$

Hence, the local basis functions (on the reference element) are identified as

$$\begin{aligned} W_1 &= \begin{bmatrix} 0 & y-1 \end{bmatrix}^{\mathrm{T}} \\ W_2 &= \begin{bmatrix} x & 0 \end{bmatrix}^{\mathrm{T}} \\ W_3 &= \begin{bmatrix} 0 & y \end{bmatrix}^{\mathrm{T}} \\ W_4 &= \begin{bmatrix} x-1 & 0 \end{bmatrix}^{\mathrm{T}} \end{aligned} \qquad \text{(4-34)}$$

The global approximations of the vector quantity $u^h(x)$ and the scalar quantity $\phi^h(x)$ are, just as in the triangular case, given by equations **(4-27)-(4-28)** and **(4-29)-(4-30)**.

4.2.5 Assembling the Global System of Equations

Now that the (global) basis functions are defined, we are in a position to calculate the global system of equations[16], as stated in equation **(4-10)**. There are several ways to achieve this: analogously to the finite difference method, in which the equations are assembled pointwise, or as in the finite element method where the assembly is performed elementwise. For the mixed discretization elementwise assembly seems the most appropriate. This means that for each element we calculate its contribution to the global system of equations by means of equation **(4-11)** and subsequently add this contribution to the global system of equations. For this scheme to work, we need to know how to add an element contribution to the global system of equations. One way to achieve this is to use a unique global numbering of the degrees of freedom, hence, each element and edge needs to be given a unique identification number[17]. By using the global numbering of the degrees of freedom, we know where to add the element contribution to the global system of equations. As an example, let us calculate the contributions of the reference triangle and quadrilateral[18].

4.2.5.1 Triangles

In the following order, we discuss the contributions to the submatrices A, B, C and subvectors G and F. In doing so we shall for the sake of simplicity only consider the simplest possible quadrature rules. Note, however, that for practical cases more accurate quadrature rules might be necessary.

[16] Note that, if we have a total of N_U edges and N_Φ elements, we also must have N_U vectorial degrees of freedom and N_Φ scalar degrees of freedom. This means that in equation **(4-10)** the length of the vector U is N_U, the length of the vector Φ is N_Φ, the size of the matrix A is $N_U \times N_U$, the size of the matrix B is $N_U \times N_\Phi$, and the size of the matrix C is $N_\Phi \times N_\Phi$. This makes the total size of the system of equations $(N_U+N_\Phi)\times(N_U+N_\Phi)$.

[17] If we also have degrees of freedom associated with the vertices, then the vertices also need unique identification numbers.

[18] As pointed out in Appendix C, it suffices to deal with the reference elements.

The contribution A_k of an element S_k to the submatrix A can be written as follows (cf. equations (4-4), (4-5) and (4-11))

$$A_k = \int_{S_k} a_{\alpha\beta}(x) w^i_\beta(x) w^j_\alpha(x) dx \qquad (i,j) \in \{1, 2, 3\} \tag{4-35}$$

According to the contravariant transformation rules discussed in Appendix C, the above integral can be transformed to the reference element as follows

$$A_k = \int_{S'} a'_{\sigma\tau}(x') w'^i_\tau(x') w'^j_\sigma(x') dx' \qquad (i,j) \in \{1, 2, 3\} \tag{4-36}$$

with the basis functions $w'^i_\tau(x')$ as in equation (4-26) and

$$a'_{\sigma\tau}(x') = J^{-1}(x')\ b_{\alpha\sigma}(x')\ \hat{a}_{\alpha\beta}(x')\ b_{\beta\tau}(x') \tag{4-37}$$

with

$$\hat{a}_{\alpha\beta}(x') = a_{\alpha\beta}(\boldsymbol{F}(x')) \tag{4-38}$$

where $\boldsymbol{F}(x')$ is the transformation carrying the reference element to the actual element in the mesh, $b_{\alpha\beta}(x')$ the functional matrix of the transformation and $J^{-1}(x')$ the functional determinant of the transformation (cf. Appendix C).

Now let us assume that $\hat{a}_{\alpha\beta}(x')$ can be written as

$$\hat{a}_{\alpha\beta}(x') = \hat{n}(x')\,(\delta_{\alpha\beta} - \hat{\mu}(x')\ \epsilon_{\alpha\beta}) \tag{4-39}$$

with

$$\delta_{\alpha\beta} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \qquad \epsilon_{\alpha\beta} = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \tag{4-40}$$

The above form is, for instance, useful when the application of a magnetic field or a mechanical stress field induces an antisymmetric part (cf. Appendix A). Now the element matrix A_k can be split into two parts, one symmetric and the other antisymmetric

$$A^s_k = \int_{S'} \hat{n}\ J^{-1}\ [b_{\alpha\sigma} b_{\alpha\tau}]\ w'^i_\tau w'^j_\sigma dx' \tag{4-41}$$

$$A^a_k = \int_{S'} \hat{n}\hat{\mu}\ J^{-1} \left[b_{\alpha\sigma}\epsilon_{\alpha\beta} b_{\beta\tau}\right] w'^i_\tau w'^j_\sigma dx' \tag{4-42}$$

The part between the square brackets in equation (4-41) can easily be recognized as the metric of the transformation (cf. Appendix C, equation (C-4)). In equation **(4-42)**, the part between the square brackets is recognized as the determinant of the transformation (Jacobian) multiplied by the Levi-Civita tensor of rank 2 (cf. Appendix A). Hence

$$g_{\sigma\tau} = b_{\alpha\sigma}b_{\alpha\tau} = \begin{bmatrix} l_1^2 & L^2 \\ L^2 & l_3^2 \end{bmatrix} \qquad \textbf{(4-43)}$$

$$b_{\alpha\sigma}\epsilon_{\alpha\beta}b_{\beta\tau} = J\epsilon_{\sigma\tau} \qquad \textbf{(4-44)}$$

Assuming that $\hat{n}(x') = n_k$ and $\hat{\mu}(x') = \mu_k$ are piecewise constant[19] and using equations **(4-26)** and **(4-41)**-**(4-44)**, the element matrix can be calculated as

$$A_k^s = \frac{n_k}{12J}\begin{bmatrix} (l_1^2 + 3l_3^2 - 3L^2) & (l_1^2 - l_3^2 - L^2) & (-l_1^2 - l_3^2 + 3L^2) \\ (l_1^2 - l_3^2 - L^2) & (l_1^2 + l_3^2 + L^2) & (-l_1^2 + l_3^2 - L^2) \\ (-l_1^2 - l_3^2 + 3L^2) & (-l_1^2 + l_3^2 - L^2) & (3l_1^2 + l_3^2 - 3L^2) \end{bmatrix} \qquad \textbf{(4-45)}$$

$$A_k^a = (n_k\mu_k)\begin{bmatrix} 0 & 1/6 & -1/6 \\ -1/6 & 0 & 1/6 \\ 1/6 & -1/6 & 0 \end{bmatrix} \qquad \textbf{(4-46)}$$

where l_i is the length of edge i of element S_k and $L = \sqrt{l_1 l_3 \cos\theta}$.

Each entry of equations **(4-45)** and **(4-46)** should be added to the matrix A according to the global numbering of the edges of the element, and according to the sign with which the element uses the edge. To give an example, suppose the edges of triangle S_k have global edge numbers (+86, -67, -45), where the signs indicate the orientation of the edge with respect to the orientation of the boundary of triangle S_k. A plus sign means that the net flux through the edge is directed outward and a minus sign means that it is directed inward. Moreover, suppose the calculated element matrix is given by

$$A_k = \begin{array}{cc} \begin{array}{ccc} 1 & 2 & 3 \end{array} & \\ \begin{bmatrix} \frac{1}{3} & \frac{1}{6} & 0 \\ \frac{1}{6} & \frac{1}{3} & \frac{1}{6} \\ 0 & \frac{1}{6} & \frac{1}{3} \end{bmatrix} & \begin{array}{c} 1 \\ 2 \\ 3 \end{array} \end{array} \qquad \textbf{(4-47)}$$

[19] For the semiconductor equations we need to use more sophisticated quadrature rules, however, this is irrelevant at this point of the discussion.

where the additional row and column indicate the local edge numbers. Now the above element matrix should be added to the global matrix as follows

$$A_k = \begin{array}{c|c} \begin{matrix} 86 & 67 & 45 \end{matrix} & \\ \begin{bmatrix} +\frac{1}{3} & -\frac{1}{6} & 0 \\ -\frac{1}{6} & +\frac{1}{3} & +\frac{1}{6} \\ 0 & +\frac{1}{6} & +\frac{1}{3} \end{bmatrix} & \begin{matrix} 86 \\ 67 \\ 45 \end{matrix} \end{array} \qquad (4\text{-}48)$$

where the additional row and column are used to indicate the global edge numbers and should be used to add the entry to the corresponding entry in the global matrix A. Note that each entry now has a sign corresponding to the product of the signs of the corresponding global edge numbers.

The contribution B_k of an element S_k to the submatrix B (or B^T) can be written as follows (cf. equations **(4-4)**, **(4-5)** and **(4-11)**)

$$B_k = -\int_{S_k} \left[\partial_\alpha w_\alpha^i(x)\right] \xi^j(x) dx = -\int_{S'} \left[\partial_\alpha w'^i_\alpha(x')\right] \xi'^j(x') dx' \qquad (4\text{-}49)$$

Note that we have already carried out the transformation to the reference element. Since only basis function ξ^k differs from zero on element S_k, the above integral is easily calculated to be

$$B_k = -\begin{array}{c|c} 1 & \\ \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} & \begin{matrix} 1 \\ 2 \\ 3 \end{matrix} \end{array} \qquad (4\text{-}50)$$

where the additional column indicates the local edge numbers and the additional row indicates the local element number. As in the previous case, the above contribution should be correctly added to the global matrix B. Following the same procedure, assuming the global edge numbers are (+86, -67, -45) and the global element number is (33) we arrive at

$$B_k = -\begin{array}{c|c} 33 & \\ \begin{bmatrix} +1 \\ -1 \\ -1 \end{bmatrix} & \begin{matrix} 86 \\ 67 \\ 45 \end{matrix} \end{array} \qquad (4\text{-}51)$$

The contribution C_k of an element S_k to the submatrix C can be written as follows

$$C_k = -\int_{S_k} c(x)\xi^i(x)\xi^j(x)dx = -\int_{S'} \hat{c}(x')\xi'^i(x')\xi'^j(x')J(x')dx' \qquad \textbf{(4-52)}$$

Also here we have already carried out the transformation to the reference element. Assuming $\hat{c}(x') = c_k$ is piecewise constant we arrive at

$$C_k = (-\frac{J}{2}) \begin{array}{cc} 33 & \\ \left[c_k\right] & 33 \end{array} \qquad \textbf{(4-53)}$$

where J is the functional determinant which equals two times the area of element S_k and we have used the global element number (33) as in the case of the contribution to B.

Now the left-hand side of the global matrix equation is completely determined. The next step is to determine the right-hand side. For this we need to determine the element contributions to the vectors G and F in equation **(4-10)**.

The contribution G_k of an element S_k to the subvector G can be written as (cf. equations **(4-4)**, **(4-5)** and **(4-11)**)

$$G_k = -\int_{(\partial S_k \cup \partial\Omega_D)} g_D w^i_\alpha n_\alpha ds = -\int_{(\partial S' \cup \partial\Omega'_D)} \hat{g}_D w'^i_\alpha n'_\alpha ds' \qquad \textbf{(4-54)}$$

Note that the surface integral is only to be taken over those edges of the element that lie on the Dirichlett part of the boundary. We now assume $\hat{g}_D(s) = g^j_D$ is piecewise constant on the Dirichlett boundary edge E_j. Moreover, we split the integral into the contributions of each of the edges of the element. This yields the following expression

$$G_k = -\sum_j \int_{(E'_j \cup \partial\Omega'_D)} g^j_D w'^i_\alpha n'^j_\alpha ds' = -\begin{bmatrix} g^1_D \\ g^2_D \\ g^3_D \end{bmatrix} \qquad \textbf{(4-55)}$$

where it is understood that g^j_D is taken to be zero if the corresponding E_i edge does not lie on the Dirichlett part of the boundary. Also in this case each entry of equation **(4-55)** must be added to the global vector G according to the global numbering of the edges of the element, and according to the sign of each edge. Suppose we have a triangle with the second edge (local numbering) on the Dirichlett boundary, moreover, assume the global edge numbers are (+86, -67, -45) (as in the previous cases). We then have

$$G_k = -\left[\begin{array}{c} +0 \\ -g_D^2 \\ -0 \end{array}\right]\begin{array}{c} 86 \\ 67 \\ 45 \end{array} \tag{4-56}$$

The additional column indicates where to put the entries in the global vector G, also note the change in sign of the entries of G_k according to the signs of the boundary edges.

The contribution F_k of an element S_k to the subvector F can be written as

$$F_k = -\int_{S_k} f(x)\xi^i(x)dx = -\int_{S'} \hat{f}(x')\xi'^i(x')J(x')dx' \tag{4-57}$$

Again the basis function $\xi'^k(x')$ is equal to one on element S_k. If we assume that $f(x)$ is piecewise linear on the element, we can, because the coordinate transform is linear, also assume that $\hat{f}(x')$ is piecewise linear on the reference element. Thus we can expand it in terms of the conforming basis functions of the reference triangle (cf. Appendix C, equation (C-2))

$$\hat{f}(x') = \sum_{i=1}^{3} f_i \psi'_i(x', y') \tag{4-58}$$

where the f_i are the values of $f(x)$ at the vertices x_i of the triangle in the mesh. Now because the Jacobian $J(x')$ does not depend on x, in the case of the transformation for triangles (cf. Appendix C, equation (C-6)), the integral can easily be evaluated to

$$F_k = -\left[\frac{J}{2}\left(\frac{f_1 + f_2 + f_3}{3}\right)\right] \tag{4-59}$$

How to add the contribution F_k to the global vector F should be obvious from the preceding discussion. However, note that in this case we do not have to deal with sign changes.

4.2.5.2 Quadrilaterals

In the case of arbitrary quadrilaterals, the coordinate transformation is no longer linear, that is, the functional matrix and determinant depend on the local coordinates. This significantly complicates the evaluation of the element contributions to the global system of equations. For the sake of simplicity we, therefore, restrict the elements in the mesh to parallelograms, rectangles and squares. For these types of elements the coordinate transform reduces to a linear transform very similar to the one used in the triangular case. The derivation of the element contributions is now completely analogous to the triangular case.

The contribution A_k of an element S_k to the submatrix A can be written as follows,

$$A_k^s = \frac{n_k}{24J}\begin{bmatrix} 8l_4^2 & -6L^2 & -4l_4^2 & 6L^2 \\ -6L^2 & 8l_1^2 & 6L^2 & -4l_1^2 \\ -4l_4^2 & 6L^2 & 8l_4^2 & -6L^2 \\ 6L^2 & -4l_1^2 & -6L^2 & 8l_1^2 \end{bmatrix} \tag{4-60}$$

$$A_k^a = (n_k\mu_k)\begin{bmatrix} 0 & -\frac{1}{4} & 0 & +\frac{1}{4} \\ +\frac{1}{4} & 0 & +\frac{1}{4} & 0 \\ 0 & -\frac{1}{4} & 0 & -\frac{1}{4} \\ -\frac{1}{4} & 0 & +\frac{1}{4} & 0 \end{bmatrix} \tag{4-61}$$

where l_i is the length of edge i of element S_k and $L = \sqrt{l_1 l_4 \cos\theta}$.

The contribution B_k of an element S_k to the submatrix B (or B^T) can be written as

$$B_k = -\begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \tag{4-62}$$

Assuming $\hat{c}(x') = c_k$ is piecewise constant, the contribution C_k of an element S_k to the submatrix C can be written as

$$C_k = (-\frac{J}{2})\begin{bmatrix} c_k \end{bmatrix} \tag{4-63}$$

Assuming the contribution G_k of an element S_k to the subvector G can be written as

$$G_k = -\begin{bmatrix} g_D^1 \\ g_D^2 \\ g_D^3 \\ g_D^4 \end{bmatrix} \tag{4-64}$$

with g_D^i evaluated at the midpoint of the edge. It is also understood that g_D^i is taken to be zero when the corresponding edge E_i is not part of the Dirichlett boundary.

Assuming that $f(x)$ is piecewise bilinear on the element the contribution F_k of an element S_k to the subvector F can be written as

$$F_k = -\left[J\left(\frac{f_1+f_2+f_3+f_4}{4}\right)\right] \qquad \textbf{(4-65)}$$

where the f_i are the values of $f(x)$ at the vertices x_i of the element in the mesh.

4.3 The Hybrid Formulation of the Mixed Method

In this section, we discuss the hybrid form of the mixed method. For the sake of simplicity, we again omit the time dependence. The hybrid method was originally developed to simplify the solution procedure for the large system of equations obtained from the mixed discretization. The system of equations as expressed by equation **(4-10)** is large and is not positive definite. For a computational domain containing N elements, using the Thomas-Raviart discretization as discussed in Section 4.2.4.1, the size of the system can be estimated as $(5/2)N$ for the case of triangles, and $3N$ for the case of quadrilaterals. This is much larger than the actual number of degrees of freedom that we eventually need to know. Further, the fact that the system of equations is usually indefinite reduces the number of available solution methods. We might remedy the problem by reducing the system to the form given in equation **(4-12)**. In this case, we obtain a positive definite system of size N. Unfortunately, the resulting system of equations is not sparse, which makes it (almost) impossible to solve the equations efficiently by means of a direct method. Hence, we must resort to iterative solution methods, however, this is a topic to be discussed in the following chapter.

The key idea behind the hybrid method is to relax the continuity properties at the interelement boundaries, as expressed by equation **(4-16)**. In order to compensate for the lack of continuity, Lagrange multipliers are introduced which enforce the continuity of the normal component of the flux at the interelement boundaries. Referring to the definitions stated in Section 4.2.4, the method can be outlined as follows.

4.3.1 Lagrange Multipliers

First, we introduce the multiplier spaces of functions defined on the set of edges E^h of the partitioning

$$M_{-1}^k(E^h) = \{\lambda^h | \lambda_E^h \in P^k(E) \quad \forall E \in E^h\} \qquad \textbf{(4-66a)}$$

$$M_{-1,D}^k(E^h) = \{\lambda^h | \lambda_E^h = 0 \quad \forall E \in E_D^h\} \qquad \textbf{(4-66b)}$$

$$M^k_{-1,*}(E^h) = \{\lambda^h | \lambda^h_E = g^h_D \quad \forall E \in E^h_D\} \tag{4-66c}$$

where $P^k(E)$ is the space of polynomials of order k on edge E. The relation between the Raviart-Thomas spaces $RT^k_{-1}(S^h)$ and $RT^k_{0,N}(S^h)$, defined by equations **(4-15)** and **(4-19)**, can now be stated as:

If $\boldsymbol{u}^h \in RT^k_{-1}(S^h)$ then $\boldsymbol{u}^h \in RT^k_{0,*}(S^h)$ if, and only if

$$\sum_{S \in S^h} \int_{\partial S} (\boldsymbol{n} \cdot \boldsymbol{u}^h)\, \mu^h ds = \int_{\partial\Omega_N} g_N \mu^h ds \quad \forall \mu^h \in M^k_{-1,D}(E^h) \tag{4-67}$$

The proof is elementary and follows from

$$\sum_{S \in S^h} \int_{\partial S} (\boldsymbol{n} \cdot \boldsymbol{u}^h)\, \mu^h ds = \sum_{E \in E^h_I} \int_E (\boldsymbol{n} \cdot \boldsymbol{u}^h|_{left} + \boldsymbol{n} \cdot \boldsymbol{u}^h|_{right})\, \mu^h ds$$
$$+ \sum_{E \in E^h_D} \int_E (\boldsymbol{n} \cdot \boldsymbol{u}^h)\, \mu^h ds + \sum_{E \in E^h_N} \int_E (\boldsymbol{n} \cdot \boldsymbol{u}^h)\, \mu^h ds \tag{4-68}$$

Obviously, the first summation on the right-hand side is zero if the continuity property of $RT^k_{0,*}(S^h)$ is satisfied, and the second summation is zero because $\mu^h \in M^k_{-1,D}(E^h)$, finally, the third summation obviously equals the right-hand side of equation **(4-67)**.

Using the above result, we may reformulate the variational problem expressed in equation **(4-7)** as:

Find $(\boldsymbol{u}^h, \phi^h, \lambda^h) \in RT^k_{-1}(S^h) \times M^k_{-1}(S^h) \times M^k_{-1,D}(E^h)$ such that

$$a(\boldsymbol{u}^h, \boldsymbol{v}^h) + b(\boldsymbol{v}^h, \phi^h) + d(\boldsymbol{v}^h, \lambda^h) = (g^h_D, \boldsymbol{v}^h)_{\partial\Omega^h_D} \quad \forall \boldsymbol{v}^h \in RT^k_{-1}(S^h)$$
$$b(\boldsymbol{u}^h, \psi^h) + c(\phi^h, \psi^h) = (f^h, \psi^h)_{\Omega^h} \quad \forall \psi^h \in M^k_{-1}(S^h)$$
$$d(\boldsymbol{u}^h, \mu^h) = (g^h_N, \mu^h)_{\partial\Omega^h_N} \quad \forall \mu^h \in M^k_{-1,D}(E^h) \tag{4-69}$$

with

$$a(\boldsymbol{u}^h, \boldsymbol{v}^h) = \sum_{S_k \in S^h} \int_{S_k} (a_{\alpha\beta} u_\beta)\, v_\alpha dx \qquad b(\boldsymbol{w}^h, \xi^h) = \sum_{S_k \in S^h} \int_{S_k} (\partial_\alpha w^h_\alpha)\, \xi^h ds$$
$$d(\boldsymbol{w}^h, \xi^h) = \sum_{S_k \in S^h} \int_{\partial S_k} (n_\alpha \cdot w^h_\alpha)\, \xi^h ds \qquad c(\omega^h, \xi^h) = -\int_\Omega c\omega\xi dx \tag{4-70}$$

and

$$(f,\psi)_{\Omega} = -\int_{\Omega} f\psi dx$$

$$(g_D,v)_{\partial\Omega_D} = -\int_{\partial\Omega_D} g_D v_\alpha n_\alpha ds \qquad (g_N^h,\mu^h)_{\partial\Omega_N^h} = \int_{\partial S} g_N \mu^h ds \qquad \text{(4-71)}$$

Note 4-6: When the original mixed formulation yields a symmetric system of equations the above formulation can easily be recognized as Lagrange's method of undetermined multipliers. For non-symmetric problems, the result is essentially the same, however, the line of reasoning is somewhat more involved.

Note 4-7: The Lagrange multipliers are not just mathematical parameters, they can be given a very useful physical interpretation. This can be inferred from the following argument. By using Green's theorem on each element separately, the first relation of the variational formulation in equation (4-7) can also be rewritten as

$$a(u^h, v^h) + b(u^h, \psi^h) + \sum_{S \in S^h} \int_{\partial S} (n \cdot v^h)\,\phi^h ds = 0 \qquad \text{(4-72)}$$

Now, by comparing equation (4-72) to the first relation in equation (4-69) it seems that the Lagrange multiplier λ on an edge E is an approximation of the restriction of ϕ on that edge. For a more rigorous proof we refer to [Arnold 1985].

4.3.2 System of Equations

The equivalent system of equations can be obtained by choosing a basis for the approximation spaces. For the Raviart-Thomas space $RT^0_{-1}(S^h)$ we may again use the basis functions as discussed in Section 4.2.4.1. However, we must keep in mind that the dimension of the space is substantially enlarged, because all local vectorial basis functions have now global status. Obviously, this is because we relaxed the interelement continuity property. For the multiplier space $M^k_{-1}(S^h)$ the situation is also identical to the mixed case described in Section 4.2.4.1. A basis for the lowest order multiplier space $M^0_{-1,D}(E^h)$ is spanned by the basis functions μ^h_i given by

$$\mu^h_i(x) = \delta_{ij} \quad x \in E_j, \quad \forall (E^i, E^j) \in E^h \backslash E^h_D \qquad \text{(4-73)}$$

Hence, we may write

$$u^h(x) = \sum_{i=1}^{I} u_i v_i(x) \qquad \phi^h(x) = \sum_{j=1}^{J} \phi_j \psi_j(x) \qquad \lambda^h(x) = \sum_{k=1}^{K} \lambda_k \mu_k(x) \qquad \text{(4-74)}$$

where K is the total number of edges, J the total number of elements, and I given by the relation

$$I = \sum_{j=1}^{J} \text{Number of Edges of Element } S_j \tag{4-75}$$

Substituting the above relations into the variational form yields an equivalent system of equations

$$\begin{bmatrix} A & B & D \\ B^T & C & 0 \\ D^T & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} U \\ \Phi \\ \Lambda \end{bmatrix} = \begin{bmatrix} G_1 \\ F \\ G_2 \end{bmatrix} \tag{4-76}$$

where

$$[A]_{ij} = \int_{\Omega} (a_{\alpha\beta} v^i_{\beta})\, v^j_{\alpha} \mathrm{d}^3x \qquad [B]_{ij} = -\int_{\Omega} (\partial_{\alpha} v^i_{\alpha})\, \psi^j \mathrm{d}^3x \tag{4-77a}$$

$$[C]_{ij} = -\int_{\Omega} c\psi^i \psi^j \mathrm{d}^3x \qquad [D]_{ij} = \sum_{S_k \in S^h} \int_{\partial S_k} (n_{\alpha} v^i_{\alpha})\, \mu^j ds \tag{4-77b}$$

$$[G_1]_i = -\int_{\partial\Omega_D} g_D v^i_{\alpha} n_{\alpha} ds \qquad [G_2]_i = \int_{\partial\Omega_N} g_N \mu^i ds \qquad [F]_i = -\int_{\Omega} f\psi^i \mathrm{d}^3x \tag{4-77c}$$

The important difference between the mixed-hybrid formulation (above) and the mixed formulation is that in this case the matrices A and B are block diagonal. Hence, we can eliminate U from equation (4-76) by means of static condensation, which results in

$$\begin{bmatrix} (B^T A^{-1} B - C) & (B^T A^{-1} D) \\ (D^T A^{-1} B) & D^T A^{-1} D \end{bmatrix} \cdot \begin{bmatrix} \Phi \\ \Lambda \end{bmatrix} = \begin{bmatrix} (B^T A^{-1} G_1 - F) \\ (D^T A^{-1} G_1 - G_2) \end{bmatrix} \tag{4-78}$$

where U can be calculated using the relation

$$U = A^{-1} (G_1 - B\Phi - D\Lambda) \tag{4-79}$$

Note that the above condensation as well as the calculation of U can be carried out at the element level by virtue of the block diagonality of A. This implies that the inverse of A can be constructed from the individual element contributions, which means that the above system of equations also can be constructed from the individual element contributions. This is not the case in the original mixed formulation, where the condensation must be carried out at the global level.

Also eliminating Φ from equation **(4-78)** results in

$$\tilde{A} \cdot \Lambda = \tilde{F} \qquad \textbf{(4-80a)}$$

with

$$\tilde{A} = D^T (A^{-1} - (A^{-1}B)\,(B^T A^{-1} B - C)^{-1}\,(B^T A^{-1}))\,D \qquad \textbf{(4-80b)}$$

$$\tilde{F} = (D^T A^{-1})\,(G_1 - B\,(B^T A^{-1} B - C)^{-1}\,(B^T A^{-1} G_1 - F)) - G_2 \qquad \textbf{(4-80c)}$$

where Φ and U can be calculated afterwards using the relations

$$\Phi = (B^T A^{-1} B - C)^{-1}\,[B^T A^{-1}\,(G - D\Lambda) - G] \qquad \textbf{(4-81a)}$$

$$U = A^{-1}\,(G_1 - B\Phi - D\Lambda) \qquad \textbf{(4-81b)}$$

The above condensation as well as the calculation of Φ and U can again be carried out at the element level because the block diagonality of B implies that $B^T A^{-1} B$ is also block diagonal. Moreover, since C is diagonal the entire matrix $(B^T A^{-1} B - C)$ is block diagonal and its inverse can be constructed from the individual element contributions. This implies that the above system of equations can also be constructed from the individual element contributions.

4.3.3 Element Contributions

The element contributions to the global system of equations can be calculated in a similar fashion to that described in Section 4.2.5. We shall not go through the procedure again, instead we just simplify equations **(4-77a)** to **(4-77c)**.

In the case of the contributions A_k, B_k and C_k of an element S_k to the matrices A, B and C we may use the results from Section 4.2.5. However, we should keep in mind that by virtue of the hybrid method, the element matrices A_k and B_k do not overlap (block diagonality), that is, they do not share degrees of freedom.

This leaves us to discuss the contribution of D_k of an element S_k to the submatrix D. From equation **(4-77b)** we find

$$D_k = \int_{E_i^{S_k}} n_\alpha v^j|_{S_k}\, ds = \delta_{ij} \qquad \textbf{(4-82)}$$

The element contributions to the right-hand side of equation (4-76) are given as

$$G_1^k = -\frac{\int\limits_{E_k \subset E_D^h} g_D ds}{\int\limits_{E_k \subset E_D^h} ds} \qquad G_2^k = \int\limits_{E_k \subset E_N^h} g_N ds \qquad F^k = -\int\limits_{S^k} f dx \tag{4-83}$$

Using the above element contributions the element matrices $\tilde{A}_k$ and $\tilde{F}_k$ can be calculated, from which the global matrices $\tilde{A}$ and $\tilde{F}$ stated in equations (4-80b) and (4-80c) can be constructed.

4.3.4 Post-Processing

The post-processing technique was first introduced in [Arnold 1985] and was intended to be used to obtain an improved approximation to the potential. In our case it is used in a different context, because we use it to add upwinding characteristics to the discretization (cf. Chapter 7), and to improve the order of the prolongation operator in the multigrid method (cf. Chapter 5). The technique is based on the fact that the Langrange multipliers are good approximations ($O(h^2)$) to the potentials at the midpoints of the edges of a triangle or quadrilateral. Here we discuss the case of the lowest order Raviart-Thomas elements and restrict ourselves to the practical results. For more theoretical details we refer to [Arnold 1985]. First, the triangular case and, second, the quadrilateral case is discussed.

Triangles

The basic procedure is to transform the sub-triangle, constructed by connecting the midpoints of the edges of the triangle shown in Figure 4-3(a), to the

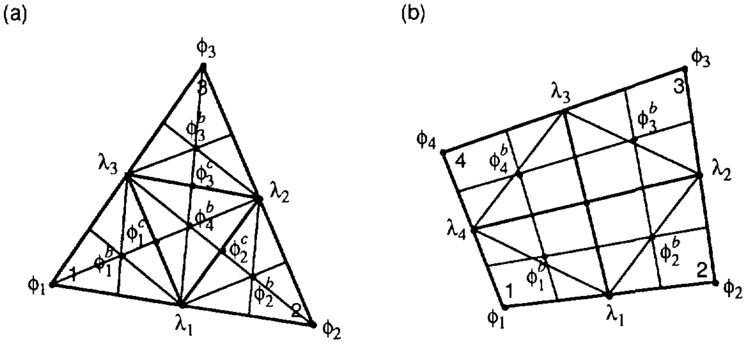

Figure 4-3 Improving the piecewise constant approximation for the potentials to a piecewise linear approximation by means of the Lagrange multipliers at the midpoints of the edges.

reference triangle (cf. Figure 4-1). On the reference triangle we construct the piecewise linear function

$$\phi^h(\hat{x}, \hat{y}) = \lambda_1 + (\lambda_2 - \lambda_1)\hat{x} + (\lambda_3 - \lambda_1)\hat{y} \tag{4-84}$$

such that at its vertices we have $\phi^h(\hat{x}_i, \hat{y}_i) = \lambda_i$. The affine transformation is given by (cf. Appendix C)

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} (\bar{x}_2 - \bar{x}_1) & (\bar{x}_3 - \bar{x}_1) \\ (\bar{y}_2 - \bar{y}_1) & (\bar{y}_3 - \bar{y}_1) \end{bmatrix} \begin{bmatrix} \hat{x} \\ \hat{y} \end{bmatrix} + \begin{bmatrix} \bar{x}_1 \\ \bar{y}_1 \end{bmatrix} \tag{4-85}$$

where the $(\bar{x}_i, \bar{y}_i)$ are the midpoints of the edges on which the λ_i are defined. Hence, by using the inverse of the above transform and equation **(4-84)**, the value of the potential at an arbitrary location of the triangle in Figure 4-3(b) can be evaluated in terms of the Lagrange multipliers. The approximation is linear, however, it is discontinuous (non-conforming) at the vertices of the triangle.

We now calculate the potentials at some special locations of the triangle shown in Figure 4-3(a). Skipping the tedious details, we end up with some particularly simple results. The potentials at the vertices are given by

$$\begin{aligned} \phi_1 &= \lambda_1 - \lambda_2 + \lambda_3 \\ \phi_2 &= \lambda_1 + \lambda_2 - \lambda_3 \\ \phi_3 &= \lambda_2 + \lambda_3 - \lambda_1 \end{aligned} \tag{4-86}$$

The potentials at the barycenters of the sub-triangles in Figure 4-3(a) are given by

$$\begin{aligned} \phi_1^b &= \frac{1}{3}[2\lambda_1 - \lambda_2 + 2\lambda_3] \\ \phi_2^b &= \frac{1}{3}[2\lambda_1 + 2\lambda_2 - \lambda_3] \\ \phi_3^b &= \frac{1}{3}[2\lambda_2 - \lambda_1 + 2\lambda_3] \\ \phi_4^b &= \frac{1}{3}[\lambda_1 + \lambda_2 + \lambda_3] \end{aligned} \tag{4-87}$$

The potentials at the circumcenters of the inner triangle in Figure 4-3(a) are given by

$$\begin{aligned} \phi_1^c &= \frac{1}{2}\,[\lambda_1 + \lambda_3] \\ \phi_2^c &= \frac{1}{2}\,[\lambda_2 + \lambda_1] \\ \phi_3^c &= \frac{1}{2}\,[\lambda_3 + \lambda_2] \end{aligned} \qquad \textbf{(4-88)}$$

Quadrilaterals

In the case of a generic quadrilateral the situation gets severely complex because of the non-linear character of the affine transformation. We restrict ourselves to the special case of parallelograms, rectangles and squares, for which the transformation is linear (cf. Appendix C). The basic procedure is to transform the parallelogram, constructed by connecting the midpoints of the edges of the parallelogram shown in Figure 4-3(b), to the reference square (cf. Figure 4-2). On the reference square we construct the piecewise bilinear function

$$\phi^h(\hat{x}, \hat{y}) = \lambda_1 + (\lambda_2 - \lambda_1)\,\hat{x} + (\lambda_4 - \lambda_1)\,\hat{y} + (\lambda_1 - \lambda_2 + \lambda_3 - \lambda_4)\,\hat{x}\hat{y} \qquad \textbf{(4-89)}$$

such that at the vertices we have $\phi^h(\hat{x}_i, \hat{y}_i) = \lambda_i$. The affine transformation is given by (cf. Appendix C)

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} (\bar{x}_2 - \bar{x}_1) & (\bar{x}_4 - \bar{x}_1) \\ (\bar{y}_2 - \bar{y}_1) & (\bar{y}_4 - \bar{y}_1) \end{bmatrix} \begin{bmatrix} \hat{x} \\ \hat{y} \end{bmatrix} + \begin{bmatrix} \bar{x}_1 \\ \bar{y}_1 \end{bmatrix} \qquad \textbf{(4-90)}$$

where the $(\bar{x}_i, \bar{y}_i)$ are the midpoints of the edges on which the λ_i are defined. Hence, by using the inverse of the above transform and equation **(4-89)**, the value of the potential at an arbitrary location of the parallelogram in Figure 4-3(b) can be evaluated in terms of the Lagrange multipliers. The approximation to the potential is now "bilinear", however, as in the triangular case, it is discontinuous (non-conforming) at the vertices of the quadrilateral.

The values of the potentials at the special locations of the quadrilateral shown in Figure 4-3(b) are listed below. At the vertices we have

$$\begin{aligned} \phi_1 &= \frac{1}{4}\,[3\lambda_1 - \lambda_2 - \lambda_3 + 3\lambda_4] \\ \phi_2 &= \frac{1}{4}\,[3\lambda_2 - \lambda_3 - \lambda_4 + 3\lambda_1] \\ \phi_3 &= \frac{1}{4}\,[3\lambda_3 - \lambda_4 - \lambda_1 + 3\lambda_2] \\ \phi_4 &= \frac{1}{4}\,[3\lambda_4 - \lambda_1 - \lambda_2 + 3\lambda_3] \end{aligned} \qquad \textbf{(4-91)}$$

At the barycenters of the sub-quadrilaterals in Figure 4-3(b) we have

$$\begin{aligned}\phi_1^b &= \frac{1}{2}[\lambda_1 + \lambda_4] \\ \phi_2^b &= \frac{1}{2}[\lambda_2 + \lambda_1] \\ \phi_3^b &= \frac{1}{2}[\lambda_3 + \lambda_2] \\ \phi_4^b &= \frac{1}{2}[\lambda_4 + \lambda_3]\end{aligned} \tag{4-92}$$

Although not shown in Figure 4-3(b), for parallelograms the circumcenters ϕ_i^c of the sub-parallelogram, constructed by connecting the midpoints of the edges, (cf. Figure 4-3(b)) coincide with the barycenters given in equation (4-92).

Conclusions

Essentially, we now have available the necessary techniques to improve the approximation to the potential. As a result, we are able to add upwinding characteristics to the discretization and to improve the prolongation operator in the multigrid method (cf. Chapter 5). The technique is applicable to triangles and parallelograms. The case of generic quadrilaterals was omitted, because of the complex and (yet) impractical results.

4.4 Time Discretization

In Chapter 3, it was shown that the time dependence of the thermoelectric model equations is of the parabolic type. In this section, for the sake of completeness, we briefly discuss a well-known time discretization for this type of time dependence and adapt it to the discrete mixed formulation. In Section 4.4.1 some general aspects are discussed and in Section 4.4.2, we specifically deal with the mixed case.

4.4.1 Parabolic in Time Operators

The general form of a parabolic in time PDE can be stated as below

$$\partial_t u(x;t) + L(x)u(r;t) = f(x;t) \tag{4-93}$$

Where $L(x)$ is a, possibly non-linear, elliptic operator containing the spatial part of the PDE and $u(x;t)$ a tensor of arbitrary rank representing the unknown physical variable. A very popular method to discretize the above equation in time is the generalized trapezoidal method [Hughes 1987]. The generalized trapezoidal method can be made either implicit, explicit or a combination of both (Crank-Nicholson), by means of an adjusting parameter α. The generalized trapezoidal method can be formulated as

$$(1 + \alpha\Delta t_n L)\, u(t_{n+1}) = f(t_n + \alpha\Delta t_n)\Delta t_n + (1 - (1-\alpha)\,\Delta t_n L)\, u(t_n) \tag{4-94}$$

As is well known the easiest scheme is obtained by taking $\alpha = 0$. In this case a fully explicit scheme is obtained for which the solution may be advanced in time without the need of matrix inversion. For $\alpha \neq 0$ an implicit scheme is obtained. Note that only for $\alpha \geq 1/2$ the scheme is unconditionally stable. In the case $\alpha < 1/2$, it requires the condition $\Delta t < 2/(1-2\alpha)\lambda_{\max}$ to be satisfied, where $\lambda_{\max}$ is the maximum eigenvalue of the discrete operator L, which is usually of order $O(h^{-2})$. Further, the local truncation error of the scheme behaves as $O(h)$ except in the case that $\alpha = 1/2$ for which it behaves as $O(h^2)$.

4.4.2 Time Dependent Mixed Problems

In this section, we discuss in particular how to discretize in time the semi-discrete equations as stated in Section 4.2.3. For this purpose we use the generalized trapezoidal rule discussed in the previous section. We start by recapitulating the semi-discrete equations

$$\begin{bmatrix} A & B \\ B^T & (C + D\partial_t) \end{bmatrix} \cdot \begin{bmatrix} U \\ \Phi \end{bmatrix} = \begin{bmatrix} G \\ F \end{bmatrix} \qquad \textbf{(4-95)}$$

$$D\partial_t \Phi - [B^T A^{-1} B - C]\,\Phi = F - B^T A^{-1} G \qquad \textbf{(4-96)}$$

Now, by means of a straightforward generalization of the trapezoidal rule to the mixed case we obtain the following general matrix-vector recursion

$$\tilde{A}(k-1)U(k) = \tilde{B}(k-1)U(k-1) + \tilde{G}(k) \qquad \textbf{(4-97a)}$$

with

$$\tilde{U}(k) = \begin{bmatrix} U(k) \\ \Phi(k) \end{bmatrix} \qquad \textbf{(4-97b)}$$

$$\tilde{G}(k) = \begin{bmatrix} \alpha G_k + (1-\alpha)\,G_{k-1} \\ \alpha F_k + (1-\alpha)\,F_{k-1} \end{bmatrix} \cong \begin{bmatrix} G(t_{k-1} + \alpha\Delta t_{k-1}) \\ F(t_{k-1} + \alpha\Delta t_{k-1}) \end{bmatrix} \qquad \textbf{(4-97c)}$$

$$\tilde{A}(k-1) = \left[\begin{bmatrix} 0 & 0 \\ 0 & D\Delta t_{k-1}^{-1} \end{bmatrix} + \begin{bmatrix} \alpha_1 & 0 \\ 0 & \alpha_2 \end{bmatrix} \begin{bmatrix} A & (B_1 + B_2) \\ B_1^T & C \end{bmatrix}\right] \qquad \textbf{(4-97d)}$$

$$\tilde{B}(k-1) = \left[\begin{bmatrix} 0 & 0 \\ 0 & D\Delta t_{k-1}^{-1} \end{bmatrix} - \begin{bmatrix} (1-\alpha_1) & 0 \\ 0 & (1-\alpha_2) \end{bmatrix} \begin{bmatrix} A & (B_1 + B_2) \\ B_1^T & C \end{bmatrix}\right] \qquad \textbf{(4-97e)}$$

Since the boundary conditions G and the source terms F are assumed to be known functions of time, the above formulation can be used as a guideline for a direct implementation of the time-dependent mixed formulation as a sequence of elliptic problems.

We may also choose to eliminate the flux variables in the global system of equations. With $\alpha_1 = \alpha_2 = \alpha$ we then arrive at the following matrix-vector recursion

$$[\Delta t_{k-1}^{-1} D + \alpha\tilde{A}]\,\Phi_k = [\alpha\tilde{F}_k + (1-\alpha\tilde{F}_{k-1})] + [\Delta t_{k-1}^{-1} D - (1-\alpha)\tilde{A}]\,\Phi_{k-1} \tag{4-98a}$$

with

$$\begin{aligned}\tilde{A} &= (B_1^T A^{-1}(B_1+B_2) - C)\\ \tilde{F}_k &= F_k - B_1^T A^{-1} G_k\end{aligned} \tag{4-98b}$$

In the case of the mixed-hybrid form the procedure is analogous, except for the fact that the entire matrix-vector recursion can be build from the individual contributions at the element level.

4.5 Non-Linear Systems of Equations

The thermoelectric model as discussed in Chapter 3 in principle is a non-linear problem. Using the quasi-linear approach as discussed in the previous sections results in a large system of quasi-linear algebraic equations. In principle such a system cannot be solved directly because the matrix entries depend on the solution vector, which is not yet available. Such a system of equations must be solved by means of an iterative technique. In Section 4.5.1, some basic iterative methods are discussed. In Section 4.5.2, we discuss iterative methods for the non-linear mixed model.

4.5.1 Basic Iterative Methods

In this section we summarize the basic principles for solving non-linear systems of equations. For more background information we refer to [Rheinboldt 1970][Rheinboldt 1974][Sherman 1978][Selberherr 1984].

Suppose that the discretized equations are given by the following matrix equation

$$F(x) = 0 \tag{4-99}$$

Assume that the above equation has a unique solution for x^*, then the iterative method is based on the construction of a contractive mapping with fixed-point x^*

$$x^{m+1} = M(x^m) \tag{4-100}$$

such that

$$\lim_{m \to \infty} \|x^m - x^*\| = 0 \qquad \textbf{(4-101)}$$

The question is how to construct such a mapping and how to specify its initial guess x^0. A sufficient condition for the mapping to be contractive is to assume the existence of the Frechet derivative $M'(x)$ at the fixed-point x^* and that the modulus of its eigenvalues are all less than one. Usually, the mapping is set up as follows

$$x^{m+1} = x^m - (B(x^m)^{-1} \cdot F(x^m)) \qquad \textbf{(4-102)}$$

Now we can evaluate the required properties of $B(x)$ such that the mapping is contractive. Taking the Frechet derivative of the right-hand side of the above equation we arrive at

$$M'(x) = I - ((B(x)^{-1})' \cdot F(x)) - (B(x)^{-1} \cdot F'(x)) \qquad \textbf{(4-103)}$$

At the fixed-point we have

$$M'(x^*) = I - (B(x^*)^{-1} \cdot F'(x^*)) \qquad \textbf{(4-104)}$$

From this equation we deduce that a variety of operators $B(x^*)$ exist such that the mapping is contractive.

Next, we discuss four basic iteration methods that more or less fit into the above theory. First, we discuss the successive substitution method, followed by Picard iteration, Newton iteration and Gauss-Seidel iteration.

Successive Substitution

If the non-linearity of the PDE is restricted to non-linear coefficients, the simplest iteration strategy follows from the natural form of the system of equations that results after discretization, that is

$$F(x) = A(x) \cdot x - b(x) \qquad \textbf{(4-105)}$$

Choosing $B(x) = A(x)$ the contractive mapping can be stated as

$$A(x^m) \cdot \delta x^{m+1} = -F(x) \qquad \textbf{(4-106)}$$

Provided the eigenvalues of $A^{-1}F'$ evaluated at the fixed-point are larger than zero, the method is convergent.

Picard Iteration

Picard's method is defined by choosing B in equation **(4-102)** the identity operator I, that is

$$B(x) = I \qquad \textbf{(4-107)}$$

This yields the following iterative scheme

$$x^{m+1} = x^m - F(x^m) \tag{4-108}$$

According to equation **(4-104)**, the Picard scheme is locally convergent if the eigenvalues of the Frechet derivative of the operator *F(x)* are larger than zero when evaluated at the fixed-point x^*. Further, if the initial x^0 guess is sufficiently close to the solution x^*, the scheme results in linear convergence.

Newton Iteration

Newton's method is defined by taking

$$B(x) = F'(x) \tag{4-109}$$

This results in the following iterative scheme

$$F'(x^m) \cdot \delta x^{m+1} = -F(x^m) \tag{4-110}$$

For Newton's method we immediately see that the modulus of all eigenvalues of *M'* in equation **(4-104)** is zero, which gives quadratic convergence for an initial x^0 guess sufficiently close to the solution x^*. Unfortunately, in most cases the convergence range of the Newton method is not very large, choosing a damped Newton scheme might cure the problem, for instance

$$B(x) = \lambda \cdot F'(x) \tag{4-111}$$

were λ is the damping factor. Here the Frechet derivative evaluates to

$$M'(x) = \left(1 - \frac{1}{\lambda}\right) \cdot I \tag{4-112}$$

The eigenvalues are readily obtained as $(1-1/\lambda)$ and thus a locally convergent iteration scheme is obtained when $\lambda \in (0.5, \infty)$.

Block Gauss-Seidel Iteration

In many cases the physical properties of the problem to be solved suggest a grouping of the degrees of freedom into blocks, for example

$$\begin{aligned} F_1(x_1, x_2, x_3) &= 0 \\ F_2(x_1, x_2, x_3) &= 0 \\ F_3(x_1, x_2, x_3) &= 0 \end{aligned} \tag{4-113}$$

Now each block is solved for its primary variables by assuming the secondary variables are known, e.g, we can solve the first block in equation **(4-113)** for x_1 by assuming that an initial guess for x_1, x_2 and x_3 is available. In order to do so we may either use Picard or Newton. We then proceed to the next block and solve for x_2, however, with the initial guess for x_1 replaced by the value of x_1 obtained from solving the previous block. A solution to the entire system is obtained by repeating the above process until a certain accuracy is obtained.

Conclusions

From the above discussion, it is obvious that the convergence properties are significantly influenced by the choice of the contractive mapping and the available initial guess, however, a "best" choice is difficult to give and usually trial and error are necessary to achieve the best results. From experience, it is known that, although Picard's method does not converge as fast as Newtons's method, the convergence range is usually larger, which makes it a somewhat more robust iteration method. A good strategy is to use a few Picard iterations to get the initial guess close to the actual solution and then to proceed with Newton iterations. The block Gauss-Seidel method proves to be a robust iteration method, however, quadratic convergence as in the Newton method cannot be expected. In isothermal semiconductor modeling the block Gauss-Seidel method is also known as Gummel's method. Here, the first block corresponds to the discretized Poisson equation and the second block corresponds to the discretized particle balance equations. For the thermoelectric problem Gummel's method can obviously be extended to the thermal domain by taking the discretized heat balance equation as the third block.

In the next section we discuss the essential non-linearities of the thermoelectric model equations in the mixed formulation. We get back to these non-linear aspects in Chapter 5, where we describe a multigrid solution method using local linearization instead of global linearization.

4.5.2 Non-Linear Mixed Problems

We now deal with the essential non-linearities of the model problem stated in equation (4-1a). As discussed in the previous sections, the basic tool for dealing with non-linear problems is linearization and iteration. The linearized problem should be formulated such that it fits into the framework of the mixed method. Let us first recapitulate the mixed problem, making explicit the essential non-linearities that appear in the thermoelectric model equations stated in Section 3.5

$$\begin{bmatrix} a_{\alpha\beta}(x;\phi) & \partial_\alpha \\ \partial_\beta & c(x) \end{bmatrix} \cdot \begin{bmatrix} u_\beta(x) \\ \phi(x) \end{bmatrix} = \begin{bmatrix} 0 \\ u_\alpha(x)\partial_\alpha\phi(x) + f(x;\phi) \end{bmatrix} \quad x \in \Omega \qquad \text{(4-114)}$$

Clearly, we recognize the non-linearity of the transport parameters and the non-linearity of the "thermal" and "electrical" source terms. The only non standard term is the first term of the bottom entry of the right-hand side. We observe that this term in principle belongs to the lower right entry of the left-hand side matrix. Hence, we can rewrite equation (4-114) as

$$\begin{bmatrix} a_{\alpha\beta}(x;\phi) & \partial_\alpha \\ \partial_\beta & c(x) - u_\alpha(x)\partial_\alpha \end{bmatrix} \cdot \begin{bmatrix} u_\beta(x) \\ \phi(x) \end{bmatrix} = \begin{bmatrix} 0 \\ f(x;\phi) \end{bmatrix} \qquad x \in \Omega \qquad (4\text{-}115)$$

However, in formulating a modified mixed variational statement (cf. Section 4.2.2 and Section 4.2.3), the lower right entry of the left-hand side matrix must be treated with care, as we show below.

The discrete variational formulation of equation (4-112) reads (cf. equation (4-10))

$$\begin{bmatrix} A & B \\ B^T & (C + C_1) \end{bmatrix} \cdot \begin{bmatrix} U \\ \Phi \end{bmatrix} = \begin{bmatrix} G \\ F \end{bmatrix} \qquad (4\text{-}116)$$

The additional contribution C_1 can be calculated from the variational form (mind the sign reversal)

$$c_1(\phi, \psi) = \int_\Omega (u_\alpha(x)\partial_\alpha\phi(x))\, \psi(x) dx \qquad (4\text{-}117)$$

where it is understood that $u_\alpha(x)$ is a given vector function. Clearly, if we use the lowest order Raviart-Thomas discretization (cf. Section 4.2.4), where the potential is approximated by a piecewise constant function, the gradient of the potential equates to zero. This problem can be avoided by first using Green's theorem, that is

$$c_1(\phi, \psi) = \int_{\partial\Omega} \phi(x)\psi(x)u_\alpha(x)n_\alpha ds - \int_\Omega \phi(x)\partial_\alpha\,(u_\alpha(x)\psi(x))\, dx \qquad (4\text{-}118)$$

Using the piecewise constant approximation for the potential ϕ and test function ψ (cf. Section 4.2.4.1), given by

$$\phi(x) = \sum_i \Phi_i\xi^i(x) \qquad \psi(x) = \sum_i \Psi_i\xi^i(x) \qquad (4\text{-}119)$$

and splitting the integrals in equation (4-118) into the element contributions yields

$$\begin{aligned} c_1(\phi, \psi) = & \sum_i\sum_j \Phi_i\Psi_j \sum_k \int_{(\partial S_k \cap \partial\Omega)} \xi^i(x)\xi^j(x)u_\alpha(x)n_\alpha ds \\ & - \sum_i\sum_j \Phi_i\Psi_j \sum_k \int_{S_k} \xi^i(x)\partial_\alpha\,(u_\alpha(x)\xi^j(x))\, dx \end{aligned} \qquad (4\text{-}120)$$

The fact that the test functions $\xi^i(x)$ are piecewise constant and unity yields

$$c_1(\phi, \psi) = \sum_i \sum_j \Phi_i \Psi_j \left[\int_{(\partial S_i \cap \partial\Omega)} u_\alpha(x) n_\alpha ds - \int_{S_i} \partial_\alpha u_\alpha(x) dx \right] \delta_{ij} \qquad \textbf{(4-121)}$$

Since the variational statement must hold for any choice of $\psi(x)$, the matrix C_1 is given by

$$[C_1]_{ij} = \left[\int_{(\partial S_i \cap \partial\Omega)} u_\alpha(x) n_\alpha ds - \int_{S_i} \partial_\alpha (u_\alpha(x))\, dx \right] \delta_{ij} \qquad \textbf{(4-122)}$$

which can easily be evaluated by using the piecewise linear approximation of the fluxes as discussed in Section 4.2.4.

The above modification of the discrete variational formulation can conveniently be used in the successive substitution iteration method or the Picard iteration method (cf. Section 4.5.1).

In the case we want to make use of the Newton iteration method we first must evaluate the Frechet derivative or Jacobian of the differential operator (cf. equation **(4-1a)**). A formal application of the Newton method yields

$$F'(z^m)\delta z^{m+1} = -F(z^m) \qquad \textbf{(4-123)}$$

with

$$z^m = \begin{bmatrix} u^m_\beta \\ \phi^m \end{bmatrix} \qquad \delta z^m = \begin{bmatrix} \delta u^m_\beta \\ \delta\phi^m \end{bmatrix} \qquad z^{m+1} = z^m + \delta z^{m+1} \qquad \textbf{(4-124)}$$

and

$$F(z^m) = \begin{bmatrix} a_{\alpha\beta}(x;\phi^m) & \partial_\alpha \\ \partial_\beta & c(x) - u^m_\alpha(x)\partial_\alpha \end{bmatrix} \cdot \begin{bmatrix} u^m_\beta(x) \\ \phi^m(x) \end{bmatrix} - \begin{bmatrix} 0 \\ f(x;\phi^m) \end{bmatrix} \qquad \textbf{(4-125)}$$

and the Frechet derivative or Jacobian given by

$$F'(z^m) = \begin{bmatrix} a_{\alpha\beta}(x;\phi^m) & \left(\partial_\alpha + \left.\dfrac{\partial a(\phi)}{\partial\phi}\right|_{\phi^m} u^m_\alpha(x) \right) \\ (\partial_\beta - \partial_\beta \phi^m(x)) & \left(c(x) - u^m_\alpha(x)\partial_\alpha - \left.\dfrac{\partial f(\phi)}{\partial\phi}\right|_{\phi^m} \right) \end{bmatrix} \qquad \textbf{(4-126)}$$

Assuming linear Dirichlett and Neumann boundary conditions, the resulting discrete variational problem, which is no longer symmetric, now reads

$$\begin{bmatrix} A & (B+B_1) \\ (B^T+B_2^T) & (C+C_1+C_2) \end{bmatrix} \cdot \begin{bmatrix} \delta U^{m+1} \\ \delta\Phi^{m+1} \end{bmatrix} = \begin{bmatrix} G \\ F \end{bmatrix} - \begin{bmatrix} A & B \\ B^T & (C+C_1) \end{bmatrix} \cdot \begin{bmatrix} U^m \\ \Phi^m \end{bmatrix} \quad \textbf{(4-127)}$$

where A, B, C, F and G are given by equation (4-11).The additional entries B_1, B_2, C_1 and C_2 are given by

$$[C_1]_{ij} = \left[\int_{(\partial S_i \cap \partial\Omega)} u_\alpha^m(x) n_\alpha ds - \int_{S_i} \partial_\alpha (u_\alpha^m(x))\, dx \right] \delta_{ij} \quad \textbf{(4-128)}$$

$$[C_2]_{ij} = \left[\int_{S_i} \left(\frac{\partial f(\phi)}{\partial \phi} \bigg|_{\phi^m} \right) dx \right] \delta_{ij} \quad \textbf{(4-129)}$$

$$[B_2^T]_{ij} = \sum_k \left[\int_{(\partial S_k \cap \partial\Omega)} \phi^m(x) \xi^i(x) v_\alpha^j(x) n_\alpha ds - \int_{S_k} \phi^m(x) \xi^i(x) \partial_\alpha v_\alpha^j(x) dx \right] \textbf{(4-130)}$$

$$[B_1]_{ij} = \sum_k \int_{S_k} \left(\frac{\partial a(\phi)}{\partial \phi} \bigg|_{\phi^m} u_\alpha^m(x) \right) v_\alpha^i \xi^j dx \quad \textbf{(4-131)}$$

which can easily be evaluated by using the piecewise linear approximation of the fluxes and piecewise constant approximation of the potentials, as discussed in Section 4.2.4.

4.6 Concluding Remarks

In this chapter a discretization method for a non-linear parabolic partial differential equation, representative for the thermoelectric problem as defined in Chapter 3, was presented. The discretization method was based on the mixed formulation of the problem. The mixed discrete equations were obtained by using the lowest order Raviart-Thomas element, which was discussed for triangular and parallelogramic elements. Moreover, all the necessary tools for the successful application of the mixed discretization method to the thermoelectric problem were discussed, including time-dependence and non-linearity. The main reason for using the mixed discretization method is that it naturally fits the form in which the problem is represented. Since we now have at our disposal all the necessary tools for the discretization of the thermoelectric problem, the following chapter must deal with an effective numerical solution method for the resulting discrete model equations. We do not wish to use the standard solution methods, because these have proven to be rather ineffective. Instead, we focus on the formulation of a multigrid method that can be used with the mixed discrete equations.

Let $\| \cdot \|$ *be some quality measure and let* A *and* B *be some achievements, then* $A \cup B = C \Rightarrow \| A \cup B \| = \| C \|$ *but if* $A \cap B \neq 0$ *then* $\| A \| \cup \| B \| \leq \| C \|$.

Chapter 5

An Adaptive Multigrid Solution Method for the Thermoelectric Problem

5.1 Introduction

In this chapter, we deal with the solution method for the discrete form of the thermoelectric problem, as presented in the previous chapter. Essentially, the problem is a system of equations of the form

$$Au = b \qquad (5\text{-}1)$$

where, $A \in R^n \times R^n$ is the discretized differential operator, $u \in R^n$ the discrete solution and $b \in R^n$ the right-hand side. Essentially, there are two more or less classical methods for solving such systems: direct methods and iterative methods. In a direct method, depending on the specific properties of the matrix A, some variant of the Gaussian elimination algorithm is used. In an iterative method, we rewrite the system as a contractive mapping with a fixed-point equal to the solution of the original system. We then start with an initial guess and successively improve this guess by iteration. For more detailed information with respect to iterative methods we refer to [Young 1971][Hackbush 1991b]. Table 5-1 shows the relative merits of some direct and some iterative methods in respect to computational effort and storage requirement. In Table 5-1, the first two methods are direct methods and the remaining ones are iterative methods. The constant $\varepsilon < 1$ is a factor with which the initial error is required to be reduced after one iteration. The size of the matrix is assumed to be $n \times n$. The Gaussian method assumes no special properties with respect to the matrix A. The Band-Gauss method assumes that A is a sparse band matrix. Clearly, the more advanced iterative methods, such as SSOR-CG (Symmetric Successive Over Relaxation Conjugate Gradient) perform very well. However, in our case, we are particularly interested in the last iterative method listed in Table 5-1, which is the multigrid or multilevel method (MGM). For large problems (n large), the multigrid method outperforms all other methods. Since the numerical solution of the full thermoelectric problem is extremely demanding from the computational point of view

[Polak 1988], the effort to investigate its application to the thermoelectric problem seems worthwhile.

Table 5-1 Upper bounds in computational effort and storage requirements for some direct and some iterative solution methods applied to a linear self-adjoint elliptic boundary value problem with a large matrix of size $n \times n$ (e.g. the Poisson equation) in 2D.

method:		computational effort:	storage:
direct:	Gaussian	$c_0 n^3$	$s_0 n^2$
	Band-Gauss	$c_1 n^2$	$s_1 n^{1.5}$
iterative:	Gauss-Seidel	$-c_2 \ln(\varepsilon) n^2$	$s_2 n$
	SOR	$-c_3 \ln(\varepsilon) n^{1.5}$	$s_3 n$
	CG	$-c_4 \ln(2\varepsilon) n^{1.5}$	$s_4 n$
	SSOR-CG	$-c_5 \ln(2\varepsilon) n^{1.25}$	$s_5 n$
	MG	$-c_6 \ln(\varepsilon) n$	$s_6 n$

Besides the benefits in computational effort for linear problems, there is yet another important reason for using the multigrid method. Let us therefore consider some non-linear problem. When using a direct or iterative method, the problem must first be globally linearized (cf. Chapter 4). Subsequently, in order to obtain a solution to the non-linear problem, the linearized equations need to be resolved many times, each time using the previous solution as an initial guess. Despite many clever schemes that have been developed over the past decades [Rheinboldt 1970][Rheinboldt 1974][Sherman 1978], this clearly involves a lot of computational work. However, in contrast to the other methods, the multigrid method is directly applicable to non-linear problems, in the sense that no global linearization of the system of equations is required, instead, a local linearization can be used. The benefit of this feature is that both linear and non-linear problems can be treated with the same efficiency. In other words, the computational effort needed to solve a non-linear problem is still $O(n)$ [Hackbush 1985].

According to [Brandt 1979] .[McCormick 1989] the multigrid method is also perfectly suited for adaptive local mesh refinement. In this case we start with a coarse partitioning of the computational domain (cf. Chapter 4), which is subsequently locally refined at those places where it is needed. Local refinement is achieved by subdivision of an element into a number of congruent child elements. This way the computational effort is automatically directed to those places of the computational domain where the solution shows sharp shifts. Clearly for such problems a significant additional improvement in accuracy and computational effort can be expected.

There are many ways to employ the multigrid method, i.e. in [Brandt 1977] multigrid is used in the context of the finite difference method, and in [Wesseling 1992] multigrid is used in the context of the finite volume method. Since we have already invested much time and effort to apply the mixed finite element method to the thermoelectric problem [Duyn 1991][Duyn 1992], this chapter builds upon the knowledge collected in Chapter 4. This approach is also followed in [Schmidt 1988] where multigrid is applied to reservoir simulation, and in [Molenaar 1992] where multigrid is applied to the semiconductor problem. Essentially the thermoelectric problem is a superset of the semiconductor problem and therefore we may build upon the research results presented in [Molenaar 1992]. Although we discuss the multigrid method in the context of the thermoelectric problem it is emphasized that in principle it can also be used for many other problems arising in semiconductor device physics, i.e. the simulation of device processing and anisotropic etching. Also more complex physical models such as the hydrodynamic model and the Monte Carlo model lend themselves for multigrid.

The structure of this chapter is as follows. In Section 5.2 a general introduction to the multigrid method is presented. In particular, we introduce the basic multigrid cycling algorithms together with their key components: nested grids, relaxation, prolongation, restriction and interpolation. The treatment presented here is not rigorous and is intended to instruct the reader who is not familiar with the multigrid principle. For more background information we refer to [Brandt 1984][Hackbush 1985][Briggs 1987][Wesseling 1992]. In subsequent sections the key components of the multigrid method in the context of the mixed formulation are discussed. The aim is to arrive at a framework that can be used to implement a fast adaptive multigrid solution method for the mixed equations resulting from the discretization of the thermoelectric problem. Consequently, as much as possible we try to arrive at results that are close to the actual implementation. Moreover, in the derivations of the results we keep in mind an object-oriented implementation strategy (cf. Chapter 6).

We first discuss, in Section 5.3, the multigrid method on composite nested grids. In Section 5.4, we discuss the relaxation method. Section 5.5 deals with the intergrid transfer operators: prolongation, restriction and interpolation. Next, in Section 5.6 we derive the error estimators that can be used for multigrid on adaptive composite grids. Finally, in Section 5.7, we close the chapter with some concluding remarks.

5.2 Basic Multigrid Techniques

Ever since the development of the first multilevel solvers for linear elliptical partial differential equations, multilevel techniques have been developed for the solution of a wide variety of other scientific problems, not only in the field of partial differential equations, but also in fields such as image processing, statistical physics and optimization problems. For an overview of recent developments in the field we refer to [McCormick 1987][Hackbush 1991a][Brandt 1992]. This section specifically deals with the basic theory and techniques for the fast solution of non-linear elliptic PDEs.

5.2.1 Relaxation

To explain the basic idea behind the multigrid method, we proceed as follows. Let us assume that we are using some iterative method (e.g. 1-point Gauss-Seidel) to solve the problem stated in equation (5-1). This means that the matrix equation representing the discrete problem is rewritten as a contractive mapping with a fixed-point equal to the solution of the original system. We then start with a global initial guess for the solution and successively improve this guess according to the rule characteristic for the contractive mapping. This sequence is repeated until an approximation of the solution is obtained that satisfies some prescribed tolerance.

However, it can be shown that for many iterative processes, the asymptotic speed of convergence is small. Typically, the error reduction per iteration is $O(1-h^2)$. Consequently, for small h, which means a large number of finite elements, many iterations are necessary to solve the problem to the required degree of accuracy. For simple problems[1] it can be shown, by means of Fourier analysis, that the slow convergence is caused by the fact that the iterative method is only efficient in removing the high frequency error components[2], whereas it has severe problems with the removal of the low-frequency error components [Briggs 1987]. This can clearly be seen from Figure 5-1 which plots the *amplification factor* μ of the Fourier components of the error, as a function of the spatial frequencies θ_1 and θ_2. Note that the amplification factor is defined as the reduction of a Fourier component after one iteration. The figure shows that $\mu \to 1$ as $(\theta_1,\theta_2) \to (0,0)$, hence, the low frequencies are not significantly reduced. A measure of the effectivity of the iteration to remove the high-frequency components is given by the *smoothing factor* $\bar{\mu}$. The smoothing factor is defined as the maximum amplification factor for those Fourier error components θ_R that cannot be resolved on the coarse grid, that is,

$$\bar{\mu} = \sup \{ \mu(\theta) | \theta \in \theta_R \} \tag{5-2}$$

In the case of the above example $\bar{\mu} = 0.5$, this means that after three iterations the high frequencies are reduced by almost one order of magnitude. In other words, the error is smooth with respect to the grid size h and contains only frequency components with periods which are large in comparison to the mesh size h.

[1] For example, the 2D Poisson problem on a square grid using the usual 5-point FD stencils.

[2] In this context, high frequency means those frequencies that cannot be accurately resolved on a grid with mesh size h.

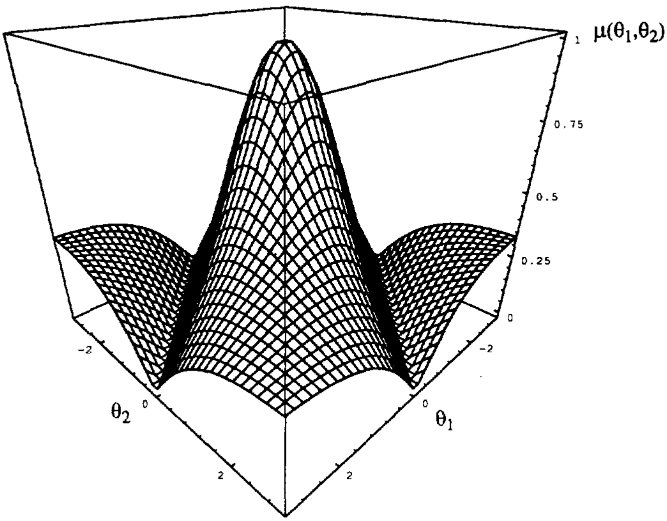

Figure 5-1 Error amplification factor for a 1-point Gauss-Seidel relaxation on a 2D Poisson problem on a square domain.

The basic idea behind the multigrid method is to approximate the smooth error on a coarser grid with grid size $2h$. Note that the definition of a coarse-grid size of $2h$ (standard coarsening) immediately defines the high-frequency error components θ_H on the fine grid. This can be argued as follows; according to the Nyquist sampling theorem the largest spatial frequency θ that can be resolved on the fine grid is $|\theta| < \pi/h$. Since the sampling frequency is effectively halved on the coarse grid, the largest spatial frequency that can be resolved on the coarse grid is $|\theta| < (1/2)\ (\pi/h)$. Hence, the spatial frequencies $|\theta| > (1/2)\ (\pi/h)$ on the fine grid cannot be resolved on the coarse grid, these must therefore be defined as the high-frequency components θ_H on the fine grid. The remaining frequencies, that is, $|\theta| < (\pi/2h)$ are defined as the low-frequency components. Note that a similar argument holds for a 2D problem. Obviously, on a coarser grid the error appears less smooth, and we may apply the same iteration strategy to reduce its high-frequency components. Hence, after a few iterations, the error is also smooth with respect to the grid size $2h$. By going to coarser and coarser grids we eventually reach the state where also the low-frequency components in the error are effectively reduced. Once an accurate approximation to the error is obtained, it is used to correct the solution on the fine grid.

The process of reducing the high-frequency components at each grid is called *relaxation* or *smoothing*. Most basic iterative methods can be used for this purpose, however, we must keep in mind that in the context of multigrid it is the smoothing property of the relaxation method which is important, not the overall convergence properties of the iterative method. Or to put it more plainly, a good iteration method is not necessarily also a good smoothing method. The schemes used to correct the solution on the fine grid can be grouped into three variants, the correction scheme (CS) [Brandt 1984], the full approximation scheme (FAS) [Brandt 1984] and the non-linear multigrid scheme (NMGS) [Hackbush 1985]. The CS is only applicable to linear problems, whereas the other two are also applicable to non-linear problems. Both non-linear algorithms (FAS and NMGS) are very similar, although the NMGS has two extra parameters which can be adapted to the specifics of the non-linear problem to be solved. So far, our experience is limited to the use of the CS and the FAS, therefore, we restrict the discussion to these types. Note that for the multigrid solution of the thermoelectric problem, we must use the FAS. However, to introduce the FAS we first discuss the CS.

5.2.2 Correction Scheme

The correction scheme can best be explained by assuming that we are dealing with two grids, the fine grid G^k with mesh size h^k and the coarse grid G^{k-1} with mesh size h^{k-1}. Let the problem to be solved on G^k be given as

$$L^k u^k = b^k \tag{5-3}$$

After a few relaxations of the above equation we obtain an approximation $\tilde{u}^k$ to the exact solution u^k. Let us define the error in the solution as

$$e^k = u^k - \tilde{u}^k \tag{5-4}$$

Moreover, let us define the residue as

$$r^k = b^k - L^k \tilde{u}^k \tag{5-5}$$

Clearly, by virtue of the linearity of the operator L^k, the equation for the error can be written as

$$L^k e^k = r^k \tag{5-6}$$

As stated in the previous section, after a few relaxations the error e^k will appear smooth compared to the mesh size h^k. It may therefore be more economically represented and solved on the coarser grid G^{k-1}. This can symbolically be written as

$$L^{k-1} e^{k-1} = R_k^{k-1} r^k \tag{5-7}$$

where, L^{k-1} is the matrix representation of the differential operator on the coarse grid, R_k^{k-1} is a restriction operator transferring the residue from the fine grid to the coarse grid. The operator L^{k-1} can be obtained completely analogous to the way L^k was obtained, in our case by means of the finite element method (cf. Chapter 3). Clearly, the above equation can be solved for

e^{k-1} once the restriction operator is defined, the result may then be used to correct the solution on the fine grid

$$u^k = \tilde{u}^k + I_{k-1}^k e^{k-1} \tag{5-8}$$

where, I_{k-1}^k is an interpolation operator transferring the error from the coarse grid to the fine grid. The above line of reasoning may be summarized in the linear two-grid algorithm shown in Table 5-2.

Table 5-2 Linear two-grid algorithm.

```
{
    Set the initial guess: $\tilde{u}^k$
    Relax $L^k u^k = b^k$ using $\tilde{u}^k$ as the initial guess
    Calculate the residue: $r^k = b^k - L^k \tilde{u}^k$
    Calculate the right-hand side of the coarse-grid problem: $r^{k-1} = R_k^{k-1} r^k$
    Solve the coarse-grid problem: $L^{k-1} e^{k-1} = r^{k-1}$
    Correct the fine-grid solution: $u^k = \tilde{u}^k + I_{k-1}^k e^{k-1}$
    Relax $L^k u^k = b^k$ using $u^k$ as the initial guess
}
```

5.2.3 Full Approximation Scheme

Let the non-linear problem to be solved on G^k be given as

$$L^k(u^k) = b^k \tag{5-9}$$

Here L^k is a non-linear operator and the correction scheme can no longer be used, since we no longer can express the error as in equation (5-6). To remedy this problem let us start from equation (5-5). Substituting equation (5-3) into equation (5-5) we obtain a relation for calculating the residue

$$r^k = L^k(u^k) - L^k(\tilde{u}^k) \tag{5-10}$$

Using equation (5-4) we may also write

$$L^k(\tilde{u}^k + e^k) = L^k(\tilde{u}^k) - r^k \tag{5-11}$$

Now defining the coarse-grid variable as

$$\hat{u}^{k-1} = \tilde{R}_k^{k-1} \tilde{u}^k + e^{k-1} \tag{5-12}$$

the coarse-grid representation of equation (5-11) reads

$$L^{k-1}(\hat{u}^{k-1}) = L^{k-1}(\tilde{R}_k^{k-1}\tilde{u}^k) + R_k^{k-1}r^k \tag{5-13}$$

where, $\tilde{R}_k^{k-1}$ is a restriction operator to transfer the solution from the fine grid to the coarse grid. Note that it may be chosen different from R_k^{k-1}. Clearly, the above equation can be solved for $\hat{u}^{k-1}$ from which we then can construct an approximation to the error $\tilde{e}^{k-1}$ on the coarse grid

$$\tilde{e}^{k-1} = \hat{u}^{k-1} - R_k^{k-1}\tilde{u}^k \tag{5-14}$$

This approximation is then used to correct the fine-grid solution according to

$$u^k = \tilde{u}^k + I_{k-1}^k\tilde{e}^{k-1} \tag{5-15}$$

We may summarize the procedure in the non-linear two-grid algorithm shown in Table 5-3.

Table 5-3 Non-Linear two-grid algorithm.

```
{
    Set the initial guess: $\tilde{u}^k$
    Relax $L^k(u^k) = b^k$ using $\tilde{u}^k$ as the initial guess and store the result in $\tilde{u}^k$
    Calculate the residue: $r^k = b^k - L^k(\tilde{u}^k)$
    Calculate the restriction of the residue: $r^{k-1} = R_k^{k-1}r^k$
    Calculate the restriction of the solution: $\tilde{u}^{k-1} = \tilde{R}_k^{k-1}\tilde{u}^k$
    Calculate the right-hand side of the coarse-grid problem: $b^{k-1} = L^{k-1}(\tilde{u}^{k-1}) + r^{k-1}$
    Solve the coarse-grid problem: $L^{k-1}(u^{k-1}) = b^{k-1}$
    Calculate the coarse-grid error: $e^{k-1} = u^{k-1} - \tilde{u}^{k-1}$
    Correct the fine-grid solution: $u^k = \tilde{u}^k + I_{k-1}^k e^{k-1}$
    Relax $L^k(u^k) = b^k$ using $u^k$ as the initial guess and store the result in $u^k$
}
```

The next step is to extend the algorithms presented in Table 5-2 and Table 5-3 to the case where multiple grids are involved.

5.2.4 Recursive Multigrid Algorithm

As discussed in the previous sections, applying the two-grid algorithm to the problem on the fine grid effectively reduces only the high-frequency error components. As pointed out in the previous sections, we must, therefore, represent and solve the error on the coarse grid and use the result to correct the solution on the fine grid. However, the number of elements on the coarse grid may still be so large that the overall convergence stalls after a few relaxations. The problem is the same as that on the fine grid, that is, only the coarse-grid high-frequency components are effectively reduced. Obviously, the solution to this problem is to apply the same strategy as applied on the fine grid, that is, after convergence stalls, we must represent and solve the error on an even coarser grid. Moreover, this process of coarsening can be applied recursively until a grid is reached where the number of elements is so small that the problem can be solved very economically by means of a direct method or in a few relaxations by means of an iterative method.

To describe the recursive multigrid algorithm, we assume that we have a sequence of nested grids G^k with grid sizes $h^k = (1/2)\,h^{k-1}$, where $k = 1$ indicates the coarsest grid and $k = K$ indicates the finest grid. Since we are dealing with a non-linear problem, we only discuss the recursive non-linear multigrid algorithm. The basic recursive non-linear multigrid algorithm follows from the non-linear two-grid algorithm by replacing the coarse-grid solution statement by γ multigrid iterations. We then obtain the algorithm described in Table 5-4.

Table 5-4 Recursive Non-Linear Multigrid Algorithm for *V*, *W* and *F* cycles.

```
Non_Linear_Multigrid( initial guess: $\tilde{u}^k$, result: $u^k$, right-hand side: $b^k$, level: $k$ )
{
    if ($k = 1$)
    {
        Solve the problem exactly by means of a direct method, or approximately by applying
        $\nu_0$ relaxation sweeps
        if (cycle = 'F' ) $\gamma = 1$
    }
    else
    {
        Relax $L^k(u^k) = b^k$ $\nu_1$ times, using $\tilde{u}^k$ as the initial guess
        Calculate the residue: $r^k = b^k - L^k(u^k)$
        Calculate the restriction of the residue: $r^{k-1} = R_k^{k-1} r^k$
        Calculate the restriction of the solution: $\tilde{u}^{k-1} = \tilde{R}_k^{k-1} u^k$
        Calculate right-hand side of the coarse-grid problem: $b^{k-1} = L^{k-1}(\tilde{u}^{k-1}) + r^{k-1}$
        for ($i = 1$ ; $i \le \gamma$ ; $i = i + 1$)    // note: $\gamma_l = 1$ by definition
        {
            Non_Linear_Multigrid( $\tilde{u}^{k-1}$, $u^{k-1}$, $b^{k-1}$, $k - 1$ )
            $\tilde{u}^{k-1} = u^{k-1}$
        }
        Calculate the coarse-grid error: $e^{k-1} = u^{k-1} - \tilde{R}_k^{k-1} u^k$
        Correct the fine-grid solution: $u^k = u^k + I_{k-1}^k e^{k-1}$
        Relax $L^k(u^k) = b^k$ $\nu_2$ times using $u^k$ as the initial guess
        if ( $k = K$ && cycle = 'F' ) $\gamma = 2$
}
```

The relaxation sweep counts ν_1 and ν_2 are usually either 0,1 or 2 whereas ν_0 in general is taken larger. The order in which the grids (levels) are visited is determined by the cycle parameter γ. The multigrid cycle for $\gamma = 1$ is usually referred to as the $V(\nu_1,\nu_2)$-cycle and in the case where $\gamma = 2$, it is referred to as the $W(\nu_1,\nu_2)$-cycle. Moreover, in the above algorithm an extra scheduling parameter "cycle" was included. By setting "cycle" equal to F we obtain the $F(\nu_1,\nu_2)$-cycle. Note that the F-cycle modifies the parameter γ during execution. An example of the three cycle types for $K = 4$ is shown in Figure 5-2 (a,b,c). A driver routine for the recursive non-linear multigrid algorithm is shown in Table 5-5.

Table 5-5 Driver routine for the Non-Linear Multigrid Algorithm to carry out *V*, *W* or *F* cycles.

```
Driver( )
{
    Choose the initial guess on the finest grid $K$: $\tilde{u}^K$
    if ( cycle = 'W' || cycle = 'F' ) { $\gamma = 2$ } else { $\gamma = 1$ }
    for ($i = 1$ ; $i \le n_K$ ; $i = i + 1$)
    {
        Non_Linear_Multigrid( $\tilde{u}^K$, $u^K$, $b^K$, $K$ )
        Let: $\tilde{u}^K = u^K$
    }
}
```

5.2.5 Full Multigrid or Nested Iteration

The idea of nested iteration [Brandt 1984] is motivated by the following notion. When no *a priori* information about the solution is available to guide us towards a choice of the initial guess $\tilde{u}^K$ on the finest grid G^K, it is obviously wasteful to start the computation on the finest grid and successively work our way to the coarsest grid. It might even be the case that with an unfortunate choice of $\tilde{u}^K$, the algorithm diverges for a non-linear problem. Since computing on a coarse grid is much cheaper, it is much better to use a coarse grid to provide an initial approximation to a fine grid. This first approximation is subsequently improved by multigrid cycles. Applying the idea recursively yields a multigrid algorithm which is commonly referred to as the Full Multigrid algorithm (FMG) or Nested Iteration. The algorithm is summarized in Table 5-6. Figure 5-2 (d) shows an FMG cycle for the case where $K = 4$, $n_K = 1$ and $\gamma = 1$. Note that the FMG algorithm starts at the coarsest grid and provides an initial guess to a finer grid by prolongation of the coarse-grid solution to a finer grid.

Table 5-6 The Full Multigrid Algorithm.

```
Full_Multigrid ( )
{
    Choose the initial guess on the coarsest grid: $\tilde{u}^1$
    Solve the coarse-grid problem exact by means of a direct method, or approximately by
    applying $\nu_0$ relaxation sweeps
    for ($k = 2$ ; $k \le K$ ; $k = k + 1$)
    {
        Prolongate: $\tilde{u}^k = P^k_{k-1}\tilde{u}^{k-1}$
        for ($i = 1$ ; $i \le n_k$ ; $i = i + 1$)
        {
            Non_Linear_Multigrid( $\tilde{u}^k$, $u^k$, $b^k$, $k$ )
            $\tilde{u}^k = u^k$
        }
    }
}
```

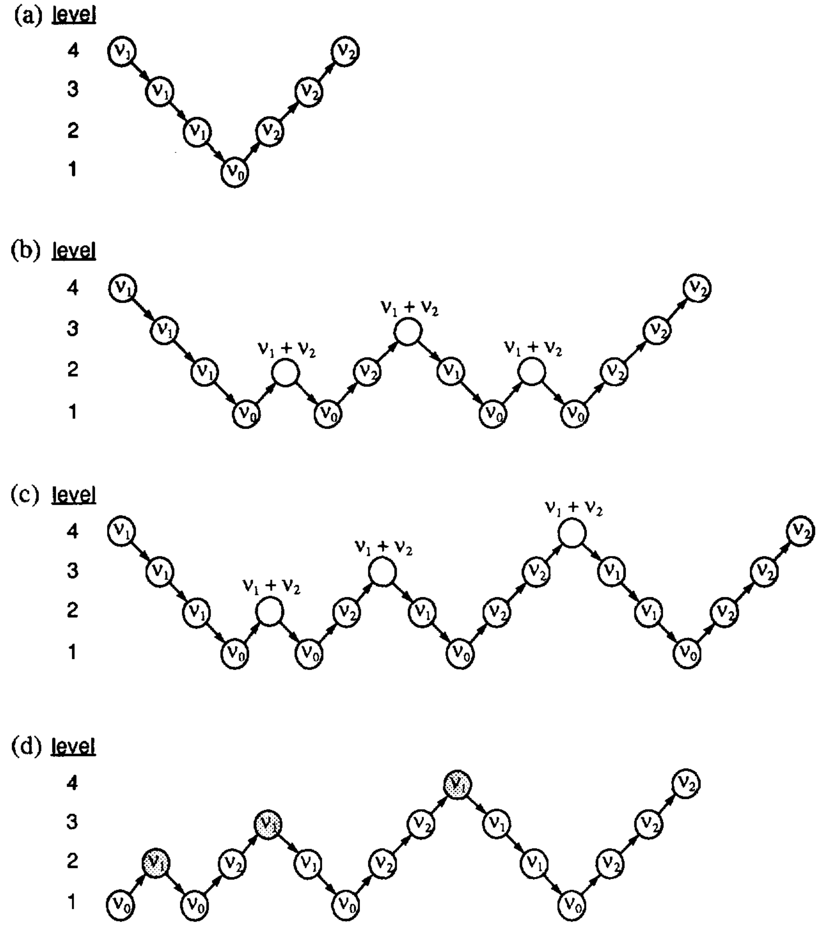

Figure 5-2 Examples of multigrid cycles: (a) *V*-cycle, (b) *W*-cycle, (c) *F*-cycle and (d) FMG V-cycle.

5.2.6 Intergrid Transfer Operators

In the preceding sections we introduced three types of intergrid transfer operators: restriction, interpolation and prolongation. Moreover, in the non-linear algorithm we defined different restriction operators for transferring the residue and the solution. Although the intergrid operators we actually use are somewhat more complicated, it is instructive to briefly discuss the essential idea for a regular one-dimensional grid using finite difference discretization. In the case a finite element discretization is used the situation is somewhat different, because then we usually use the properties of the approximation spaces (cf. Chapter 4) to design the intergrid transfer operators. This case will be dealt with later on in this chapter.

5.2.6.1 Restriction

The restriction operators $\tilde{R}_k^{k-1}$ and R_k^{k-1} are defined to transfer the fine-grid solution and the fine-grid residue to the next coarser grid. Formally, the restriction operator is defined to transfer a fine-grid function to a coarse-grid function, that is, if u^k is a fine-grid function then $R_k^{k-1}u^k$ is a coarse-grid function. The most straightforward restriction operator is denoted as *injection*. In this case, the value of $u^{k-1} = R_k^{k-1}u^k$ in a coarse-grid point is simply the value of u^k in the fine-grid point coinciding with the coarse-grid point. Another method to obtain the value $R_k^{k-1}u^k$ in a coarse-grid point is to use a weighted average of the values u^k in the coinciding fine-grid point and its nearest neighbors. This variant is usually called*full weighting*. Both variants are exemplified for the one-dimensional case in Figure 5-3.

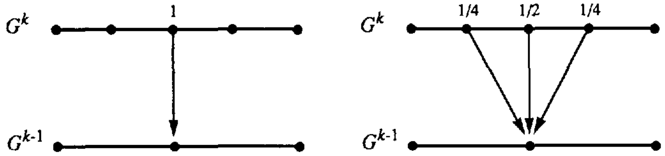

Figure 5-3 Restriction operators for the one-dimensional case.

For regular grids, usually a stencil notation is used to characterize the restriction operator, for example the stencils for the case of injection and full weighting in the one dimensional case are

$$R_k^{k-1} = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} \qquad \text{(5-16a)}$$

$$R_k^{k-1} = \frac{1}{4}\begin{bmatrix} 1 & 2 & 1 \end{bmatrix} \qquad \text{(5-16b)}$$

The extension to two-dimensional regular grids is obvious and the injection and full weighting operators read

$$R_k^{k-1} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \qquad \text{(5-17a)}$$

$$R_k^{k-1} = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \qquad \text{(5-17b)}$$

5.2.6.2 Interpolation

The interpolation operator I_k^{k+1} is defined to transfer the coarse-grid error to the next finer grid, hence, if e^{k-1} represents the coarse-grid error then

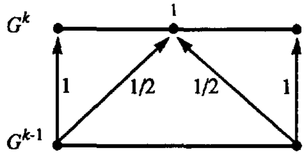

Figure 5-4 Linear interpolation for the one dimensional case.

$I_{k-1}^{k}e^{k-1}$ is the fine-grid error. As such, it performs the opposite function of the restriction operator. The interpolation is usually carried out according to a multi-polynomial interpolation of some specified order, e.g., linear interpolation in the case of a one-dimensional problem. An example of linear interpolation on a regular one-dimensional grid is shown in Figure 5-4. Note that apart from a constant factor the interpolation operator is the transpose of the restriction operator given in equation (5-16b). This will appear to be a general feature of the (Galerkin) finite element approximations we use.

5.2.6.3 Prolongation

The prolongation operator P_k^{k+1} essentially performs the same operation as the interpolation operator, however, it is only used in the nested multigrid method for transferring a converged coarse-grid solution to the next finer grid. The reason for defining a separate prolongation operator is that it is sometimes advisable to choose the order of the prolongation higher than the order of the interpolation, for instance, linear interpolation and cubic prolongation. Choosing a higher-order prolongation reduces the introduction of high-frequency error components on the next finer grid.

5.2.6.4 Accuracy Conditions

In order to assure the mesh-size independent convergence rates, it has been shown that the orders of the transfer operators must satisfy the following condition [Brandt 1984][Wesseling 1992]

$$m_I + m_R > 2m \tag{5-18}$$

where, m_I is the order of the interpolation operator, m_R is the order of the restriction operator and $2m$ is the order of the partial differential equation. The order of an interpolation operator is defined as the highest degree plus one, of polynomials that are interpolated exactly. For a restriction operator, the order is similarly defined for the transpose of the restriction operator. For example, in the case of a linear interpolation operator, the order is $m_i = 2$. For a restriction operator using full weighting the order is $m_R = 2$, because the transpose of the restriction operator is identical to linear interpolation (apart from a constant factor). Note that the transpose of the restriction operator using injection is not a well-defined interpolation, however, its order is $m_R = 1$.

In [Brandt 1984][Wesseling 1992] it is argued that it advisable to choose the order of the prolongation operator such that

$$m_p > m_c \tag{5-19}$$

where m_p is the order of the prolongation operator and m_c is the order of consistency of the discretization. When the above condition is met it can be shown that the full multigrid method reduces the algebraic error (cf. Section 5.2.7) to well below the truncation error in a small number of work units [Wesseling 1992]. In the case a less accurate prolongation operator is used, e.g. $m_p = m_c$ it is still useful to apply the full multigrid, however, there is no guarantee that the algebraic error will be much smaller than the discretization error. In such cases we may improve the solution further by appending the full multigrid iteration with a few more plain multigrid iterations.

5.2.7 Convergence Criterium

To determine the convergence properties of the multigrid algorithm, we should keep in mind the following definitions. First, the *truncation error* is the norm of the difference in the exact solution of the original continuous problem and the exact solution of the approximate difference problem on the finest grid, that is

$$e_t^h = \|e_t^h\| = \|u(x_i) - u_i^h\| \le Ch^p \tag{5-20}$$

Note that depending on the type of discretization chosen, the truncation error is of $O(h^p)$, as is also indicated in equation (5-20). The truncation error measures how well the solution of the discrete problem approximates the exact solution of the original continuous problem.

Obviously, we cannot solve the discrete problem exactly by means of an iterative method such as the multigrid method. For this purpose we must define another quantity called the *algebraic error*

$$e_a^h = \|e_a^h\| = \|u^h - \tilde{u}^h\| \tag{5-21}$$

The algebraic error measures how well the iterative solution $\tilde{u}^h$ to the differential problem approximates the exact discrete solution u^h. Obviously, it makes no sense to try to get the algebraic error way below the truncation error. Consequently, any approximation that differs by less than the truncation error from the exact solution of the discretized problem is as good an approximation to the solution of the continuous differential problem as the exact solution of the discretized problem itself. Hence, an iterative algorithm has converged when the algebraic error is of the order of the truncation error. Here we say that we have converged up to the level of truncation.

The property of the multigrid algorithm to reduce the algebraic error up to the level of truncation has a few interesting consequences. If we take a look at the FMG cycle in Figure 5-2 (d) and assume that at each shaded circle the algorithm has converged up to the level of truncation (if not we may increase the number of cycles per level, e.g., choose $n_k = 2$ for each k) we can easily check the order of convergence of the discrete approximation to the continuous problem. For this purpose let us define an error measure

$$\mathrm{ERR}(k, k-1) = \left\| \tilde{u}^{k-1} - \tilde{R}_k^{k-1} \tilde{u}^k \right\| \qquad \textbf{(5-22)}$$

measuring the difference in the converged solutions on two consecutive grids. The ratios $\mathrm{ERR}(k, k-1)/\mathrm{ERR}(k-1, k-2)$ should approach 2^{-p} for increasing mesh size.

5.2.8 Performance

To estimate the computational work needed to perform one multigrid iteration, we introduce the concept of a work unit (WU). Following [Brandt 1984] a WU is defined as the amount of work involved to perform one relaxation sweep on the finest grid. Assuming standard grid coarsening, a relaxation sweep on grid G^k then involves an amount of work equal to

$$W_k = 2^{d(k-K)}\mathrm{WU} \qquad \textbf{(5-23)}$$

where d is the dimension of the problem. Using some elementary theory of geometrical series, assuming that the cost of the intergrid transfers and the calculation of the solution on the coarsest grid can be neglected, it follows that the total amount of work W involved to carry out one $V(\nu_1,\nu_2)$-cycle ($\gamma = 1$) or $W(\nu_1,\nu_2)$-cycle ($\gamma = 2$) is bounded by

$$W \leq \left[\frac{\nu_1 + \nu_2}{1 - \gamma 2^{-d}} \right] \mathrm{WU} \qquad (\gamma < 2^d) \qquad \textbf{(5-24)}$$

Using similar arguments the total amount of work involved to carry out one $F(\nu_1,\nu_2)$-cycle is bounded by

$$W \leq \left[\frac{\nu_1 + \nu_2}{(1 - 2^{-d})^2} \right] \mathrm{WU} \qquad \textbf{(5-25)}$$

Finally, the total amount of work involved to carry out one FMG(ν_1,ν_2, γ)-cycle is bounded by

$$W \le \left[\frac{(\nu_1 + \nu_2)}{(1 - \gamma 2^{-d})\,(1 - 2^{-d})}\right] \mathrm{WU} \qquad (\gamma < 2^d) \tag{5-26}$$

The work involved in carrying out the various multigrid cycles is shown in Table 5-7.

Table 5-7 The work involved to carry out *V*, *W* and *F* cycles (normalized to ($\nu_1 + \nu_2$) WU).

d	1	2	3
V-cycle	2	4/3	8/7
F-cycle	4	16/9	64/49
W-cycle	x	2	4/3
FMG V-cycle	2	16/9	64/49
FMG W-cycle	x	8/3	48/21

Note that for 1D problems ($d = 1$), it usually does not pay off to use multigrid, except when we specifically want to make use of the additional advantages of multigrid, such as local grid refinement and non-linear problems. Also note that the work estimate given in equation (5-24) is out of range in the case of a *W*-cycle applied to a 1D problem (cf. Table 5-7). It can be shown that for such cases the amount of work becomes super-linear in the number of unknowns [Wesseling 1992]. In practice, such situations should be avoided. The important conclusion that can be drawn from the above theory is that the total amount of work involved to carry out one multigrid cycle is proportional to one WU, which in its turn is proportional to the number of elements n in the finest grid. Hence, provided the convergence of the multigrid algorithm is such that it reduces the algebraic error to the level of truncation in $O(1)$ multigrid cycles, the algorithm is of $O(n)$. This leads us to a discussion on the convergence properties of the multigrid algorithm.

As should be clear from the preceding sections, an important step in developing multigrid solvers is the design of an interior relaxation scheme with a high smoothing factor $\bar{\mu}$. Note that the reduction of the non-smooth error components is basically a local task, that is, it can be done in a certain neighborhood independently of other parts of the domain. This is the reason why it can efficiently be done by relaxation, which essentially is a local process. An estimate for the error reduction that is possible with a multigrid cycle can be obtained by using local mode analysis [Brandt 1984][Wesseling 1992]. In a local mode analysis, one assumes that the problem is posed in a homogeneous unbounded domain. Hence, the effect of non-uniformities at boundaries and interfaces are neglected. Usually, the smoothing factor deteriorates in the vicinity of non-uniformities and must be compensated for by adding extra relaxations. Since a boundary or interface is on a lower dimensional manifold, the extra amount of work involved in relaxing a boundary or interface is still small compared to the work involved in relaxing the interior. Note that in the case of problems with variable coefficients, the smoothing factor $\bar{\mu}$ is position dependent. Non-linear problems also can be treated by applying local mode analysis to the linearized problem. For problems with weakly non-linear coefficients, a simple Picard linearization usually suffices, however, in the case of exponential dependence, as for the problem defined in Chapter 3, Newton linearization or some other variant is necessary.

The important conclusion to be drawn from the preceding discussion is that, assuming the low-frequency components in the error are solved on coarser grids, the error reduction is approximately determined by the efficiency of the relaxation process to reduce the high-frequency components. Hence, the error reduction factor per multigrid cycle is approximately,

$$\bar{\mu}^{(\nu_1 + \nu_2)} \tag{5-27}$$

where $\bar{\mu}$ is the asymptotic smoothing factor, defined in Section 5.2.1, and ν_1, ν_2 respectively are the number of pre and post relaxation sweeps. For the example given in Section 5.2.1, the smoothing factor is equal to 0.5, hence, a single multigrid cycle with $\nu_1 = 2$ and $\nu_2 = 1$ gives an error reduction of almost one order of magnitude, independent of the number of elements in the finest grid.

Bearing the above in mind, we are now in a position to estimate the performance of the various multigrid cycles. For example, by applying a *V*-cycle to an initial approximation of order $O(1)$, one needs $O(\ln n)$ cycles to reduce the algebraic error to the level of truncation $O(h^p)$. Hence, the multigrid algorithm using *V*-cycles and solving up to the level of truncation is of $O(n \ln n)$. A similar argument holds for *W*-cycles. However, if we consider the FMG algorithm, where a converged solution of the discrete problem on a coarser grid is used as a first approximation, only $O(1)$ multigrid cycles are needed. Effectively, this means that the FMG algorithm is indeed of $O(n)$.

5.2.9 Conclusions

In this section we have tried to justify the fairly large amount of effort that needs to be spent in order to design and implement a suitable multigrid method that can be used for solving discrete problems obtained by application of the mixed discretization method. Although the multigrid method is fairly intricate, its computational benefits are simply too significant for not trying to apply it. Once a suitable version is developed it can be applied, with some minor adjustments, to the thermoelectric problem. In subsequent sections the key components of the multigrid method in the context of the mixed formulation are discussed. The aim is to arrive at a framework that can be used to implement a fast adaptive multigrid solution method for the mixed equations resulting from the discretization of the thermoelectric problem. Consequently, as much as possible we try to arrive at results that are close to the actual implementation. Moreover, in the derivations of the results we keep in mind an object-oriented implementation strategy (cf. Chapter 6).

5.3 Composite Grids

From singular perturbation analysis it can be shown that the semiconductor problem is singularly perturbed [Markowich 1983][Markowich 1990]. Since the thermoelectric problem is a superset of the semiconductor problem it must also be singularly perturbed. A singularly perturbed problem features sharp regions of strong and of weak variation of its solution. In order to properly resolve the regions of strong variation (boundary layers and junctions), these regions need a much finer grid than the regions of weak variation. In this section we therefore discuss the idea of composite grids [McCormick 1989] and investigate how it relates to the mixed formulation of the problem to be solved on the computational domain. It turns out that the mixed formulation possesses some convenient properties with respect to the use of composite grids.

In contrast to the treatment of composite grids in [McCormick 1989], where only rectangular domains regularly partitioned into square elements are considered, we allow any shape of domain as long as it can be represented by a quasi-regular partitioning into triangles and quadrilaterals. In principle the partitioning should be chosen as crude as the irregularity of the computational domain admits. However, at curved boundaries we need many elements in the coarse grid in order to properly resolve the curvature of the boundary. Unfortunately, for the multigrid method to be effective, we need coarse grids in the proper sense. This means that the element sizes are not allowed to vary rapidly over the coarse grid. However, if we use an adaptive grid strategy anyway, an appealing solution to this problem is to use a partitioning that not exactly matches the actual computational domain. We may organize the grid adaptation in such a way that during the solution procedure the boundary is resolved up to a specified accuracy[3].

The multigrid algorithms discussed in Section 5.2 assume that so-called homogeneous grid refinement is used. In that case we start with a coarse grid G^0 which is a rather crude quasi-regular partitioning of the domain Ω into triangles and/or quadrilaterals (cf. Chapter 4). In order to obtain finer grids each coarse-grid element is partitioned into a number of congruent sub-elements, that is, a triangle into four congruent sub-triangles and a quadrilateral into four congruent sub-quadrilaterals[4] (cf. Figure 5-5). This process is

[3] In this dissertation we have not yet fully investigated the algorithmic aspects of this method and therefore leave it for future research.

called basic refinement. The additional elements created upon refining an element are called the child elements and the element generating the childs is called a parent. The subdivision of elements can (recursively) be repeated and in fact generates a sequence of nested grids $\{G_k\}_{k \in M}$, with $M = \{0, \ldots, K-1\}$ and k the level of refinement. Note that this type of grid refinement reduces the average element size by a factor of two for each additional level of refinement, hence the corresponding sequence of grid sizes is given by $\{h_k = h_0 2^{-k}\}_{k \in M}$.

When refining a grid G_k it is not necessary to refine each element belonging to that grid, instead, we may use a suitable error criterium to decide whether an element needs to be refined. This process is called adaptive refinement and results in nested locally refined grids. The set of elements created by refinement of grid G_k is grouped into a set of disjoint grid patches $\{P^l_{k+1}\}_{l \in N}$, with $N = \{0, \ldots, L-1\}$ and l the patch number. The union of the patches P^l_{k+1} to grid G_k constitutes grid G_{k+1}. An example of the above procedure is given in Figure 5-6. The type of grids obtained in this way are usually referred to as composite grids. Note that in contrast to more conventional strategies of grid refinement we do not demand inter element continuity at the artificial boundary of a grid patch. Instead, each grid patch is considered as an independent grid together with boundary conditions, on which a solution must be calculated. This greatly simplifies the grid data structures that are needed to represent the grid.

The entire grid is organized as a multiple-rooted tree data structure (forest) called the grid tree, where each root represents a coarse-grid element. The nodes of the tree represent the elements, and the edges represent the parent-

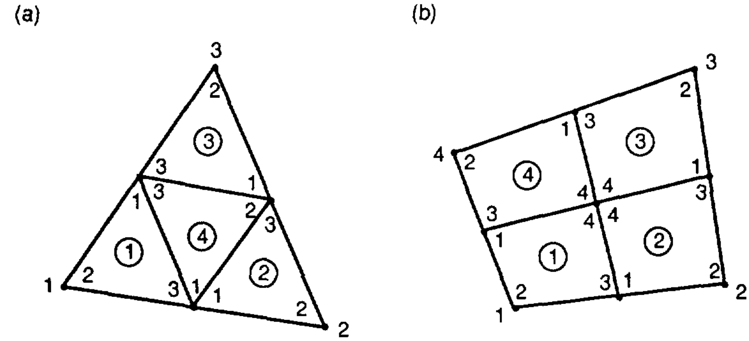

Figure 5-5 Basic refinement of (a) triangles and (b) quadrilaterals. A triangle is refined into four congruent child triangles by connecting the midpoints of the edges of the triangle. A quadrilateral is refined into four child quadrilaterals by connecting the midpoints of opposite edges. The numbering of the elements, edges and vertices is also shown.

4. Higher order partitioning is also conceivable, for instance, the second order regular partitioning of a triangle generates nine congruent sub-triangles.

child relationship. The leaves of the tree at a certain level represent the unrefined elements of that level. In the case of 1D grids the tree is a binary tree, in 2D grids it is a quad tree, and in 3D grids it is an octal tree. Also at each level k of the grid tree the topology of grid G_k is stored, which means that each node of the tree contains pointers to its nearest neighbors at the same grid level. Essentially, this additional feature makes the tree a graph with the original tree as a minimum spanning tree. Note that a grid scanning algorithm can be based upon the stored topology (cf. Chapter 6).

In order to manage the grid patches, we introduce an additional tree data structure which is called the patch tree. The root of the patch tree contains a pointer to the seed element of the coarse grid. Each node at a certain level in the patch tree contains a pointer to an element at the corresponding level in the grid tree. This element acts as a seed element for that grid patch. Note that because the topology of the grid is stored, it is sufficient to point to the seed element of a patch, the remaining elements of that patch can be found by scanning. The edges of the patch tree represent the parent-child relationship between the patches, expressing the fact that the region occupied by a child patch is embedded in the region occupied by its parent patch.

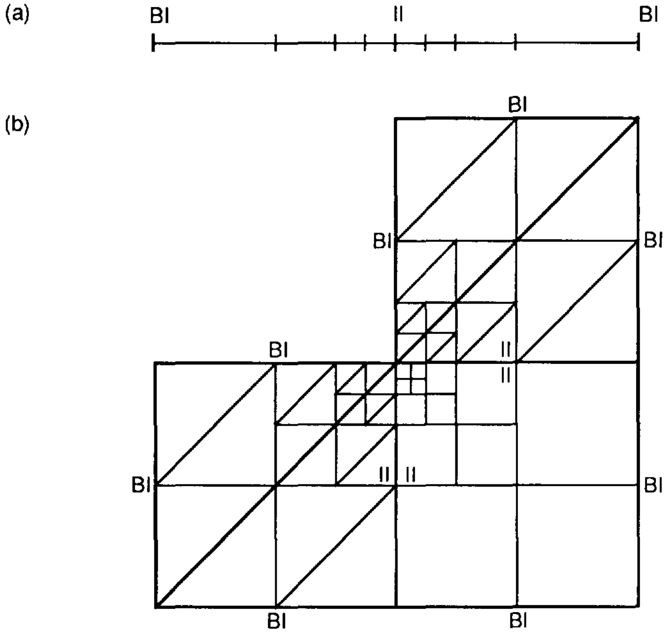

Figure 5-6 Two simple examples of locally refined computational domains: (a) 1D configuration with four levels, (b) 2D configuration with four levels.

A more detailed description of the way the composite grids are organized and how grids are scanned can be found in Chapter 6.

Obviously, if we use the above procedure the locally refined grids that are obtained are discontinuous, in the sense that there are interfaces that have two fine-grid cells at one side and a coarse-grid cell at the other side (cf. Figure 5-6). Such interfaces are called green interfaces. In the case a conforming finite element method is used, as in [Bank 1990] and [McCormick 1989], we face the problem of having discontinuous potentials at the green interfaces. To get around this problem in [Bank 1990] and [McCormick 1989] the concept of temporary green interface elements was used (cf. Figure 5-7). However, if the mixed finite element discretization is used the situation is rather different. Basically, the benefit is that we do not need to use the temporary green interface elements which greatly simplifies the data structures and algorithms needed to manage the composite grid (cf. Chapter 6). How to treat grid patches in the context of the mixed finite element discretization is discussed in the following section.

5.3.1 Nested Grid Patches

In order to calculate the solution on a grid patch we proceed as follows. Basically, we want to treat the grid patch as an ordinary grid for which we apply the mixed discretization method. However, we then face the problem of what boundary conditions to specify at the interface between the grid patch and the coarse-grid cells (cf. Figure 5-8). Following [Bank 1990], the coarse and fine-grid edges that coincide with the interface between the grid patch and the coarse-grid cells are called the "green" edges. Formally, at the green coarse-grid edges we have available the values of the fluxes. We could therefore take the linearly interpolated values of the fluxes at the green coarse-grid edges as Neumann boundary conditions at the green fine-grid edges. However, if the grid patch does not touch a part of the Dirichlett boundary of the computational domain, we obtain a singular Neumann problem for which

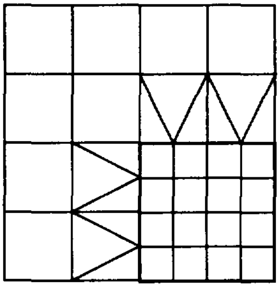

Figure 5-7 An example of the treatment of locally refined grids with square elements by means of temporary triangular green interface elements.

the solution is only determined up to an arbitrary constant. Clearly, this is not acceptable, because there is no way of determining this constant.

Fortunately, the mixed-hybrid discretization method (cf. Section 4.3) offers a very elegant solution to this problem. Since the Lagrange multipliers are a good approximation to the potentials at the edges, we can use these Lagrange multipliers as Dirichlett boundary conditions at the green fine-grid edges. To be more precise, we take the value of a Lagrange multiplier at a coarse-grid green edge as the Dirichlett boundary condition for the two green fine-grid edges that coincide with the green coarse-grid edge. In principle, the coarse-grid Lagrange multipliers can be calculated by means of equations (4-80a)-(4-80c). However, this procedure is a little cumbersome, because we need to keep track of the entire mixed-hybrid formulation. In the next section where we discuss a relaxation method for the mixed equations, we also discuss a procedure to calculate the Lagrange multipliers from the results of the relaxation. Note that the accuracy of the Lagrange multipliers at the edges is of the same order $O(h^2)$ as the approximation to the fluxes, hence the use of the Lagrange multipliers does not at all involve an accuracy penalty.

5.4 Relaxation

A basic component in the multi-level method is the relaxation of the system of equations on a particular grid level (cf. Section 5.2.1). Basically, a relaxation method should act as an error smoother. It is known that all basic iterative methods (Jacobi, Gauss-Seidel, etc.) can be used as an error smoother, provided that proper under or over-relaxation is used [Wesseling 1992]. In this case the relaxation method is based on the mixed formulation introduced in the previous chapter. This method was proposed in a different form by [Vanka 1986] to solve the Navier-Stokes equations on a rectangular staggered grid by means of finite differencing. In [Schmidt 1988] a superbox relaxation method was used to solve the mixed equations for reservoir simulation by means of multigrid. However, in [Molenaar 1992] it was shown that the Van-

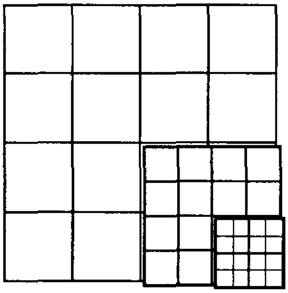

Figure 5-8 The boundary conditions at the "artificial" boundary of a grid patch.

ka type of relaxation is more efficient. In this section we will investigate if the Vanka relaxation can also be used to resolve the peculiarities of the thermoelelectric problem.

In Section 5.4.1 we discuss some notational concepts. In Section 5.4.2 - Section 5.4.4, we consider the case of mixed finite elements on (mixed) triangular and quadrilateral grids in 2D, for which we will derive a very compact formulation of the relaxation operator. Next, Section 5.4.5 deals with the treatment of the boundary conditions in the relaxation operator. In order to get some impression of the obtainable efficiency of the relaxation operator, in Section 5.4.6 a local mode analysis for the case of square grid cells is presented. Whether these results also approximately hold for the more general case of triangular and quadrilateral grids is difficult to answer *a priori*, because the usual tools for checking the convergence of a relaxation method do not generalize to quasi unstructured grids. However, for grids with triangles and quadrilaterals that are not too weirdly shaped, likely similar convergence results can be expected. If some regions behave badly we may add extra relaxation to these (localized) regions. Next, Section 5.4.7 deals with a nonlinear relaxation method. Finally, in Section 5.4.8, we discuss the calculation of the Lagrange multipliers.

5.4.1 Preliminaries

Essentially a Vanka relaxation is a block Gauss-Seidel iteration applied to the system of equations represented in equation (4-10), and arises naturally from the mixed formulation (cf. Chapter 4). Basically, it means that when an element is visited the fluxes and potentials are relaxed simultaneously. In order to do so we must express the values of the unknowns in an element in terms of the (known) values of the nearest neighbors. In what follows we shall use the following labeling (cf. Figure 5-9 and Figure 5-10).

The element containing the unknown degrees of freedom is denoted as S_0. The potential[5] on this element is denoted as Φ_0. The edges of S_0 are then labeled as, $E_{01}, \ldots, E_{0N_0}$ with N_0 the number of edges of S_0. The corresponding fluxes are denoted as $U_{01}, \ldots, U_{0N_0}$ and are positive in outward direction. The nearest neighbor element bordering edge E_{0k} then is denoted as S_k and the potential on this element is denoted as Φ_k. The edges or faces of S_k starting with the one that borders S_0 are labeled sequentially starting from one, that is, $E_{k1}, \ldots, E_{kN_k}$, where N_k is the number of edges of S_k. The corresponding fluxes are denoted as $U_{k1}, \ldots, U_{kN_k}$ and are positive in outward direction. At first sight this labeling may seem a little complex, but it has the advantage of making the derived relations very compact and close to an actual implementation.

[5] This may actually be a vector of potential-like degrees of freedom.

5.4.2 Grids with Triangles

In Figure 5-9 we show the basic relaxation subdomain in the case of a 2D triangular grid. The goal is to express the degrees of freedom of element S_0 in terms of the known degrees of freedom of the nearest neighbor elements S_k, the source terms, and if present the boundary conditions.

For this purpose we may regard the relaxation domain as a tiny computational domain for which we assemble the system of equations according to the principles discussed in Section 4.2.5. This gives us a system of equations in the well known mixed form

$$\begin{bmatrix} A & B \\ B^T & C \end{bmatrix} \cdot \begin{bmatrix} U \\ \Phi \end{bmatrix} = \begin{bmatrix} G \\ F \end{bmatrix} \tag{5-28}$$

with

$$U = [U_{01}, \ldots, U_{33}]^T \qquad \Phi = [\Phi_0, \ldots, \Phi_3]^T \tag{5-29}$$

The matrix entries can be calculated according to

$$[A]_{ij} = \int_\Omega (a_{\alpha\beta} w^i_\beta) w^j_\alpha dx \qquad [B]_{ik} = -\int_\Omega (\partial_\alpha w^i_\alpha) \xi^k dx$$
$$[C]_{kl} = -\int_\Omega c\xi^k\xi^l dx \tag{5-30}$$

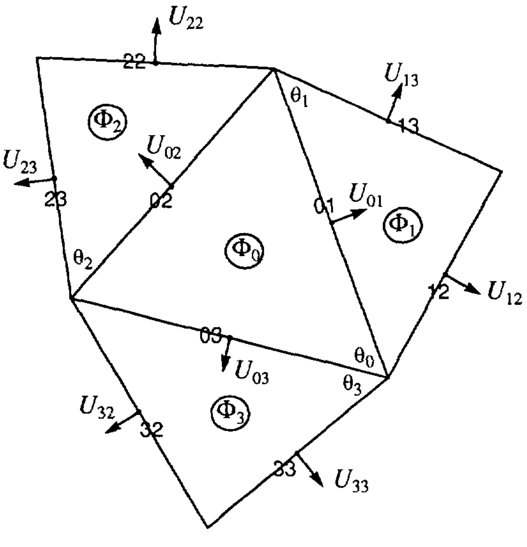

Figure 5-9 Relaxation subdomain on a triangular grid.

and the right hand sides as

$$[G]_i = -\int_{\partial\Omega_D} g_D w^i_\alpha n_\alpha ds \qquad [F]_k = -\int_{\Omega} f\xi^k dx \tag{5-31}$$

where according to Figure 5-9 the indices (i,j) can take the values {01, 02, 03, 11, 12, 13, 21, 22, 23, 31, 32, 33} and the indices (k,l) can take the values {0, 1, 2, 3}.

Note 5-1: In calculating the above matrix entries one must take great care with respect to orientation of the fluxes associated with the edges. In principle the same sign conventions hold as discussed in Section 4.2.5. In short, the direction of the flux depends on with which sign the element uses the edge. If the sign is positive the flux is directed outwards and if the sign is negative the flux is directed inwards. Since elements sharing an edge always use that edge with opposite signs, the orientation of the fluxes is always consistent.

Since only the degrees of freedom on element S_0 are assumed to be unknown, we may remove the relations associated with the other degrees of freedom from equation (5-28). Expanding the remaining equations then gives

$$\begin{gathered}\sum_j [A]_{0i,j}[U]_j + \sum_k [B]_{0i,k}[\Phi]_k = [G]_{0i} \qquad i = \{1,2,3\} \\ \sum_j [B]_{j,0}[U]_j + \sum_k [C]_{0,k}[\Phi]_k = [F]_0\end{gathered} \tag{5-32}$$

where the index j runs over all edges with index {01, 02, 03, 11, 12, 13, 21, 22, 23, 31, 32, 33} and the index k runs over all elements with index {0, 1, 2, 3} of the relaxation subdomain. Now we may use equations (5-30) and (5-31) to simplify equation (5-32) by eliminating the terms that are always zero in the summations, e.g. edge E_{01} does not couple with edge E_{12}. This yields

$$\begin{gathered}\sum_{j=1,2,3} [A]_{0i,0j}[U]_{0j} + \sum_{j=2,3} [A]_{0i,ij}[U]_{ij} + \sum_{k=0,i} [B]_{0i,k}[\Phi]_k = [G]_{0i} \\ \sum_{j=1,2,3} [B]_{0j,0}[U]_{0j} + [C]_{0,0}[\Phi]_0 = [F]_0\end{gathered} \tag{5-33}$$

where index i can take the values {1, 2 ,3}. Next, we transfer all known degrees of freedom in the left hand side to the right hand side and replace the summation indices by the ones indicated by Figure 5-9. This yields

$$\begin{gathered}\sum_{j=1}^{N_0} [A]_{0i,0j}[U]_{0j} + [B]_{0i,0}[\Phi]_0 = [G]_{0i} - \sum_{j=2}^{N_i} [A]_{0i,ij}[U]_{ij} - [B]_{0i,i}[\Phi]_i \\ \sum_{j=1}^{N_0} [B]_{0j,0}[U]_{0j} + [C]_{0,0}[\Phi]_0 = [F]_0\end{gathered} \tag{5-34}$$

where index i can take the values $\{1, \dots, N_0\}$.

Next, we calculate explicit formulas for the remaining matrix entries in the above equation. Starting with the simplest one, we observe that the entries of the matrix B are +1 or -1 depending on the sign of the edge to which the matrix entry belongs. Referring to Figure 5-9 we, for example, have

$$[B]_{01,0} = -1 \qquad [B]_{01,1} = 1 \tag{5-35}$$

Evaluating the entry of the matrix C is trivial and follows from

$$[C]_{0,0} = -\int_{S_0} c(x)dx \tag{5-36}$$

Likewise the entries of the vectors G and F are trivial and follow from

$$[G]_{0i} = -\int_{E_{0i}} g_D w_\alpha^{0i} n_\alpha ds \qquad [F]_0 = -\int_{S_0} f(x)dx \tag{5-37}$$

The entries of the matrix A are a little more difficult, because several elements may contribute to an entry. According to equation **(5-34)** we only need the entries $[A]_{i,j}$ with indices $(i,j) \in \{01, 02, 03\}$. We therefore write the submatrix $\tilde{A}$ of A as

$$\tilde{A} = \tilde{A}_0 + \sum_{k=1}^{N_0} \tilde{A}_k \tag{5-38a}$$

with

$$\tilde{A}_0 = \int_{S_0} (a_{\alpha\beta} w_\beta^i) w_\alpha^j dx \qquad (i,j) \in \{01, 02, 03\} \tag{5-38b}$$

$$\tilde{A}_k = \int_{S_k} (a_{\alpha\beta} w_\beta^i) w_\alpha^j dx \qquad (i,j) \in \{01, 02, 03\} \tag{5-38c}$$

Assuming $a_{\alpha\beta} = a\delta_{\alpha\beta}$, equation **(5-38b)** is recognized as equation **(4-46)** in Section 4.2.5.1, which in this case can be written as

$$\tilde{A}_0 = \frac{\hat{a}_0}{12J_0}\begin{bmatrix} (l_{01}^2+3l_{03}^2-3L_0^2) & (l_{01}^2-l_{03}^2-L_0^2) & (-l_{01}^2-l_{03}^2+3L_0^2) \\ (l_{01}^2-l_{03}^2-L_0^2) & (l_{01}^2+l_{03}^2+L_0^2) & (-l_{01}^2+l_{03}^2-L_0^2) \\ (-l_{01}^2-l_{03}^2+3L_0^2) & (-l_{01}^2+l_{03}^2-L_0^2) & (3l_{01}^2+l_{03}^2-3L_0^2) \end{bmatrix} \qquad \textbf{(5-39)}$$

The matrices $\tilde{A}_k$ in equation **(5-38c)** only contribute to the diagonal entries of $\tilde{A}$; to be more precise $\tilde{A}_k$ contributes only to $[A]_{k,k}$. The result can be written as

$$[\tilde{A}_1+\tilde{A}_2+\tilde{A}_3]_{i,j} = \left(\frac{\hat{a}_i}{12J_i}(l_{i1}^2+3l_{i3}^2-3L_i^2)\right)\delta_{ij} \qquad (i,j) \in \{1,2,3\} \qquad \textbf{(5-40)}$$

where, l_{ij} is defined as the length of the corresponding edge E_{ij}, and the parameter $L_k^2 = l_{k0}l_{k3}\cos(\theta_k)$. Furthermore, J_k is the Jacobian of element S_k and $\hat{a}_k$ is a suitable average of a on element S_k. Note that the inclusion of the contribution of an antisymmetric part of the tensor $a_{\alpha\beta}$ to A is straightforward (cf. Section 4.2.5.1).

In principle the preceding discussion provides sufficient information to solve for the unknown degrees of freedom on element S_0. However, it is cheaper to first eliminate the fluxes from equation **(5-34)**. For this purpose we rewrite equation **(5-34)** (again) as a matrix-vector relation

$$\begin{bmatrix} \tilde{A} & \tilde{B} \\ \tilde{B}^T & \tilde{C} \end{bmatrix} \cdot \begin{bmatrix} U \\ \Phi \end{bmatrix} = \begin{bmatrix} \tilde{G} \\ \tilde{F} \end{bmatrix} \qquad \textbf{(5-41)}$$

with $\tilde{A}$ a 3 × 3 matrix given by equations **(5-39)** and , $\tilde{B}$ a 3 × 1 matrix with entries equal to +1 or -1 depending on the sign of the edge use, $\tilde{C}$ a 1 × 1 matrix given by equation **(5-36)**, $\tilde{F}$ a 1 × 1 vector given by equation **(5-37)** and $\tilde{G}$ a 3 × 1 vector given by

$$[\tilde{G}]_{0i} = [G]_{0i} - \sum_{j=2}^{N_i} [A]_{0i,ij}[U]_{ij} - [B]_{0i,i}[\Phi]_i \qquad i = 1,2,3 \qquad \textbf{(5-42)}$$

The fluxes may then be eliminated in the usual manner resulting in an equation explicit in the unknown potential

$$(\tilde{B}^T\tilde{A}^{-1}\tilde{B} - \tilde{C})\Phi = \tilde{B}^T\tilde{A}^{-1}\tilde{G} - \tilde{F} \qquad \textbf{(5-43)}$$

The fluxes can be calculated afterwards according to

$$U = \tilde{A}^{-1}(\tilde{G} - \tilde{B}\Phi) \qquad \textbf{(5-44)}$$

5.4.3 Grids with Quadrilaterals

In Figure 5-10 we show the basic relaxation subdomain in the case of a 2D quadrilateral grid. For simplicity we have shown the case of square elements. Not surprisingly the procedure is entirely analogous to the one shown in the preceding section. Carrying out this procedure again results in equation (5-34) which we subsequently may simplify for this specific case. The difficulty in this case is that it is rather difficult to obtain explicit results for the generic quadrilateral case. This is because of the non-linear character of the affine transformation (cf. Appendix C) for the case of quadrilaterals. However, we are able to obtain explicit results for the case of grids with square, rectangular and parallelogramic elements.

Next, we calculate explicit formulas for the matrix entries in equation (5-34) for the case of parallelograms. The results for squares and rectangles follow as simplifications. We only explicitly discuss the coefficients that are significantly different from the triangular case. In the case of the entries of the matrix B the situation is identical to the triangular case. Also the calculations of the matrix C and the vectors G and F are identical, so equations (5-36) and (5-37) may be used for this purpose. The difference is in the calculation of the matrix A. According to equation (5-34) we only need the entries $[A]_{i,j}$ with indices $(i,j) \in \{01, 02, 03, 04\}$. We therefore write the submatrix $\tilde{A}$ of A as

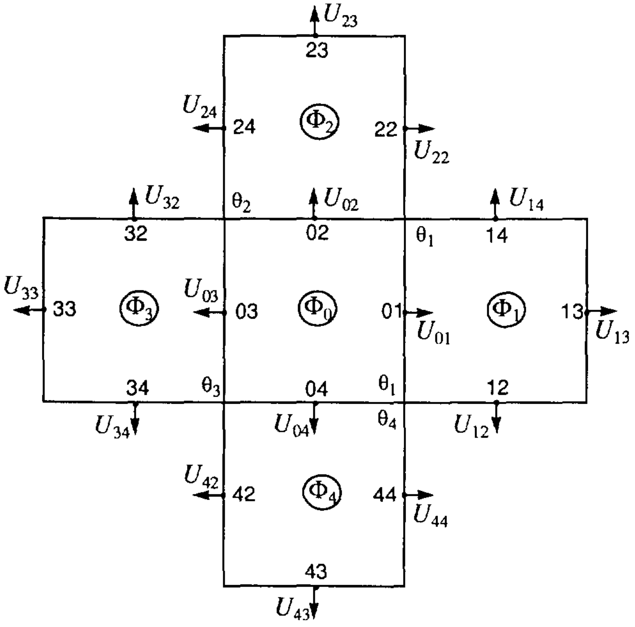

Figure 5-10 Relaxation subdomain on a "quadrilateral" grid.

$$\tilde{A} = \tilde{A}_0 + \sum_{k=1}^{N_0} \tilde{A}_k \tag{5-45a}$$

with

$$\tilde{A}_0 = \int_{S_0} (a_{\alpha\beta} w^i_{\beta})\, w^j_{\alpha} dx \qquad (i,j) \in \{01, 02, 03, 04\} \tag{5-45b}$$

$$\tilde{A}_k = \int_{S_k} (a_{\alpha\beta} w^i_{\beta})\, w^j_{\alpha} dx \qquad (i,j) \in \{01, 02, 03, 04\} \tag{5-45c}$$

Assuming $a_{\alpha\beta} = a\delta_{\alpha\beta}$, equation **(5-38b)** is recognized as equation **(4-60)** in Section 4.2.5.2, which in this case can be written as

$$\tilde{A}_0 = \frac{\hat{a}_0}{24J_0}\begin{bmatrix} 8l_{04}^2 & -6L_0^2 & -4l_{04}^2 & 6L_0^2 \\ -6L_0^2 & 8l_{01}^2 & 6L_0^2 & -4l_{01}^2 \\ -4l_{04}^2 & 6L_0^2 & 8l_{04}^2 & -6L_0^2 \\ 6L_0^2 & -4l_{01}^2 & -6L_0^2 & 8l_{01}^2 \end{bmatrix} \tag{5-46}$$

The matrices $\tilde{A}_k$ in equation **(5-38c)** only contribute to the diagonal entries of $\tilde{A}$; to be more precise $\tilde{A}_k$ contributes only to $[A]_{k,k}$. The result can be written as

$$[\tilde{A}_1 + \tilde{A}_2 + \tilde{A}_3 + \tilde{A}_4]_{i,j} = \left(\frac{\hat{a}_i}{24J_i}(8l_{i4}^2)\right)\delta_{ij} \qquad (i,j) \in \{1, 2, 3, 4\} \tag{5-47}$$

where, l_{ij} is defined as the length of the corresponding edge E_{ij}, and the parameter $L_k^2 = l_{k1}l_{k4}\cos(\theta_k)$. Furthermore, J_k is the Jacobian of element S_k and $\hat{a}_k$ is a suitable average of a on element S_k. Just as in the triangular case the inclusion of the contribution of an antisymmetric part of the tensor $a_{\alpha\beta}$ to A is straightforward (cf. Section 4.2.5.1). In a similar fashion we can eliminate the fluxes, the results are identical to the triangular case.

The results get particularly simple if square elements are used. In that case we may set $L_k = 0$ and $l_{ij} = h$, and consequently the following entries of the matrix A are zero

$$\begin{aligned} [A]_{01,02} = [A]_{01,04} = [A]_{02,01} = [A]_{02,03} = 0 \\ [A]_{03,02} = [A]_{03,04} = [A]_{04,01} = [A]_{04,03} = 0 \\ [A]_{01,12} = [A]_{01,14} = [A]_{02,22} = [A]_{02,24} = 0 \\ [A]_{03,32} = [A]_{03,34} = [A]_{04,42} = [A]_{04,44} = 0 \end{aligned} \tag{5-48}$$

5.4.4 Mixed Grids

Using the previous results it is not difficult to set up the equations in the case of a mixed grid, for instance a triangular cell with two triangular and one quadrilateral nearest neighbors. However, instead of deriving results for all particular cases we shall give a heuristic description of an algorithm that can be used to build the system of equations that needs to be solved when an element is visited. Since our grids are indeed of mixed type this procedure is also followed in practical cases. Again the starting point is equation (5-34) which actually provides sufficient algorithmic detail for all encountered cases. The information provided by the visited element[6] and its nearest neighbors, in combination with equation (5-34) is then used to calculate the entries of the vector-matrix equations shown in equation (5-41). These results are subsequently used to solve equation (5-43). Next the fluxes on the edges of the element are calculated by means of equation (5-42).

Note 5-2: All results of the relaxation are stored locally, that is, the potential is stored in the element itself and the fluxes are stored in the edges. In other words we do not build a global solution vector. More on this topic will be discussed in Chapter 6.

5.4.5 Boundary Conditions

Until now we have not discussed how to incorporate the Dirichlett and Neumann boundary conditions into the relaxation scheme. We may distinguish three basis cases: (a) none of the edges in the relaxation subdomain coincide with a boundary, (b) one or more of the nearest neighbor elements have edges that coincide with the boundary, and (c) the visited element has edges that coincide with the boundary. Note that it is possible that these cases occur simultaneously. Since the derivation of equation (5-34) is completely general we may also use it for these cases. However, there are a few details that need to be taken care of.

case (a):

Since we do not have to take into account any boundary conditions we may set vector $[G]_{0i}$ to zero in equation (5-34).

case (b):

Let us first assume that an edge of a neighbor coincides with a Dirichlett boundary. Again we may set vector $[G]_{0i}$ to zero, because it only relates to element S_0, which is assumed not to touch any boundary. No further actions need to be taken. Next we assume that an edge of a neighbor coincides with a Neumann boundary. Again we set vector $[G]_{0i}$ to zero. The flux associat-

[6] Each element is an object that encapsulates sufficient information to be "self supporting" (cf. Chapter 6).

ed with the edge is now evaluated from the Dirichlett boundary condition instead of being generated by the previous relaxation sweep. For instance, if edge E_{01} coincides with the Neumann boundary we may set

$$[U]_{01} = -\int_{E_{01}} g_N ds \cong -\hat{g}_N^{01} l_{01} \qquad \textbf{(5-49)}$$

In principle however nothing changes because we always assume the correct value of the flux to be stored in the edge, which in this case is the Dirichlett boundary condition.

Note 5-3: Care must be taken with respect to the appropriate sign of $[U]_{0i}$. As in previous cases the sign is determined by the sign with which the element uses the edge. If the sign is positive then the flux associated with that edge is directed outwards and equation **(5-49)** has the appropriate sign. However, in case the sign is negative the flux is directed inwards and as a result equation **(5-49)** needs a minus sign. Note that by definition the flux g_D is always directed outwards.

case (c):

First we assume that an edge of the visited element coincides with a Dirichlett boundary. In this case the vector $[G]_{0i}$ becomes important. The effect of the boundary condition can be accounted for by calculating the appropriate entry of $[G]_{0i}$. For instance, if edge E_{01} coincides with the Dirichlett boundary we may set

$$[G]_{01} = -\int_{E_{01}} g_D w_{\alpha}^{01} n_{\alpha} ds = -\frac{\int_{E_{01}} g_D ds}{\int_{E_{01}} ds} \cong -\hat{g}_D^{01} \qquad \textbf{(5-50)}$$

Moreover, since the edge has no neighbor we may leave out the remaining terms of the right hand side of equation **(5-34)** for this edge. In the case the edge coincides with a Neumann boundary we simply “remove” the corresponding equation and transfer the known value of the flux at that edge to the right hand side. The flux can be evaluated according to equation **(5-49)**.

Note 5-4: Also in this case special care needs to be taken with respect to the sign of $[G]_{0i}$. If the sign of the edge is positive the equation **(5-50)** has a proper sign. However, in case the edge is negative the sign of equation **(5-50)** needs to be reversed.

5.4.6 Local Mode Analysis

The reduction of the non-smooth error components is basically a local task, that is, it can be done in a certain neighborhood independently of other parts of the domain. This is the reason why it can be done by means of relaxation, which is in fact is a local smoothing process. The efficiency of the relaxation method can be approximated by means of local mode analysis which aims to produce an estimate of the smoothing factor $\bar{\mu}$ (cf. Section 5.2.1). Once the smoothing factor is known the efficiency of the multigrid method can be estimated as discussed in Section 5.2.8. Possible causes that deteriorate the efficiency, such as boundary conditions and local singularities, can be taken care of by adding extra local relaxation at these areas. As long as these areas are on a lower dimensional manifold the extra work involved is insignificant. In this section we will present a local mode analysis of the relaxation operator, derived in the previous sections. We only carry out the analysis for grids with square elements, since the analysis is virtually impossible to carry out for the irregular finite element grids we actually use. The analysis of the convergence of the relaxation operator on these types of grids uses tools that go beyond the present scope of this dissertation.

Discrete Fourier Transform

As usual in local mode analysis it is assumed that the problem is defined on an unbounded rectangular domain with square grid cells and that the coefficients in the PDEs are frozen. Moreover, for non-linear problems the analysis is carried out for the (Newton) linearized problem. Basically, the goal is to relate the frequency components in the error before and after a relaxation sweep. Our starting point is Figure 5-10 and equation (5-34). Next, we define a rectangular grid with grid cells twice as fine as shown in Figure 5-10 and expand the solution (u_x, u_y, ϕ) into a discrete Fourier series

$$u_x(n, m) = \sum_{k=0}^{N_x-1} \sum_{l=0}^{N_y-1} a^x_{k,l} \exp\left(j\frac{k2\pi}{N_x}n\right) \exp\left(j\frac{l2\pi}{N_y}m\right) \tag{5-51a}$$

$$u_y(n, m) = \sum_{k=0}^{N_x-1} \sum_{l=0}^{N_y-1} a^y_{k,l} \exp\left(j\frac{k2\pi}{N_x}n\right) \exp\left(j\frac{l2\pi}{N_y}m\right) \tag{5-51b}$$

$$\phi(n, m) = \sum_{k=0}^{N_x-1} \sum_{l=0}^{N_y-1} a^\phi_{k,l} \exp\left(j\frac{k2\pi}{N_x}n\right) \exp\left(j\frac{l2\pi}{N_y}m\right) \tag{5-51c}$$

Note 5-5: Since the Fourier transform is defined on a grid twice as fine as the grid used for relaxation, all indices used in equation (5-34) can be related to the Fourier grid by means of the following correspondence

$$\begin{array}{cccc} 01 \to i+1,j & 02 \to i,j+1 & 03 \to i-1,j & 04 \to i,j-1 \\ 13 \to i+3,j & 23 \to i,j+3 & 33 \to i-3,j & 43 \to i,j-3 \end{array}$$
$$0 \to i,j \quad 1 \to i+2,j \quad 2 \to i,j+2$$
$$3 \to i-2,j \quad 4 \to i,j-2 \tag{5-52}$$

Relaxation Operator

In order to proceed we shall make the following assumption with respect to the order in which the grid cells are scanned. If we use the wavefront scanning method, to be discussed in Chapter 6, the cells in the neighborhood of the relaxation domain are scanned in the following order: 4 → 3 → 0 → 1 → 3. As a result the fluxes are updated in the order: (04), (03), (01,02,03,04), (01) and (02) (cf. Figure 5-10). Hence, in a single relaxation sweep the fluxes of a grid cell are effectively updated twice. When estimating the smoothing factor we must take this behavior into account. Using these notions and the results from Section 5.4.3 we may now rewrite the relaxation operator in equation **(5-41)** to a form

$$Du = h(u) \tag{5-53a}$$

with

$$h = \begin{bmatrix} (-\frac{1}{6})\,\hat{a}_1 u_x^{m-1}(i+3,j) + \phi^{m-1}(i+2,j) \\ (-\frac{1}{6})\,\hat{a}_2 u_y^{m-1}(i,j+3) + \phi^{m-1}(i,j+2) \\ (-\frac{1}{6})\,\hat{a}_3 u_x^{m}(i-3,j) - \phi^{m}(i-2,j) \\ (-\frac{1}{6})\,\hat{a}_4 u_y^{m}(i,j-3) - \phi^{m}(i,j-2) \\ 0 \end{bmatrix} \tag{5-53b}$$

and

$$D = \begin{bmatrix} \frac{\hat{a}_0+\hat{a}_1}{3} & 0 & \frac{\hat{a}_0}{6} & 0 & 1 \\ 0 & \frac{\hat{a}_0+\hat{a}_2}{3} & 0 & \frac{\hat{a}_0}{6} & 1 \\ \frac{\hat{a}_0}{6} & 0 & \frac{\hat{a}_0+\hat{a}_3}{3} & 0 & -1 \\ 0 & \frac{\hat{a}_0}{6} & 0 & \frac{\hat{a}_0+\hat{a}_4}{3} & -1 \\ 1 & 1 & -1 & -1 & -\hat{c}_0 \end{bmatrix} \tag{5-53c}$$

and

$$u = \left[\bar{u}_x^m(i+1,j)\ \bar{u}_y^m(i,j+1)\ u_x^m(i-1,j)\ u_y^m(i,j-1)\ \phi^m(i,j)\right]^{\mathrm{T}} \qquad \textbf{(5-53d)}$$

where it is understood that $\{\bar{u}_x(i,j), \bar{u}_y(i,j), u_x(i,j), u_y(i,j), \phi(i,j)\}$ refers to the error in the solution instead of the actual solution, the superscript m refers to the relaxation sweep count, that is, m-1 means a value from the previous sweep, and the overbar is used to indicate a quantity that has not yet been seen by the current relaxation sweep (m).

Error Amplification Matrix

Substituting equations **(5-51a)-(5-51c)** in equation **(5-53a)** and simplifying, results in a Fourier transformed relaxation operator

$$\tilde{D}\tilde{u} = \tilde{h} \qquad \textbf{(5-54)}$$

with

$$\tilde{h} = \begin{bmatrix} \left(-\frac{\hat{a}_1}{6}\right) a_{k,l}^{x,m-1} e^{j\frac{3\theta_x}{2}} + a_{k,l}^{\phi,m-1} e^{j\theta_x} \\ \left(-\frac{\hat{a}_2}{6}\right) a_{k,l}^{y,m-1} e^{j\frac{3\theta_y}{2}} + a_{k,l}^{\phi,m-1} e^{j\theta_y} \\ \left(-\frac{\hat{a}_3}{6}\right) a_{k,l}^{x,m} e^{-j\frac{3\theta_x}{2}} - c_{k,l}^{\phi,m} e^{-j\theta_x} \\ \left(-\frac{\hat{a}_4}{6}\right) a_{k,l}^{y,m} e^{-j\frac{3\theta_y}{2}} - c_{k,l}^{\phi,m} e^{-j\theta_y} \\ 0 \end{bmatrix} \qquad \textbf{(5-54a)}$$

and

$$\tilde{D} = \begin{bmatrix} \left(\frac{\hat{a}_0+\hat{a}_1}{3}\right) e^{j\frac{\theta_x}{2}} & 0 & \left(\frac{\hat{a}_0}{6}\right) e^{-j\frac{\theta_x}{2}} & 0 & 1 \\ 0 & \left(\frac{\hat{a}_0+\hat{a}_2}{3}\right) e^{j\frac{\theta_y}{2}} & 0 & \left(\frac{\hat{a}_0}{6}\right) e^{-j\frac{\theta_y}{2}} & 1 \\ \left(\frac{\hat{a}_0}{6}\right) e^{j\frac{\theta_x}{2}} & 0 & \left(\frac{\hat{a}_0+\hat{a}_3}{3}\right) e^{-j\frac{\theta_x}{2}} & 0 & -1 \\ 0 & \left(\frac{\hat{a}_0}{6}\right) e^{j\frac{\theta_y}{2}} & 0 & \left(\frac{\hat{a}_0+\hat{a}_4}{3}\right) e^{-j\frac{\theta_y}{2}} & -1 \\ e^{j\frac{\theta_x}{2}} & e^{j\frac{\theta_y}{2}} & -e^{-j\frac{\theta_x}{2}} & -e^{-j\frac{\theta_y}{2}} & -\hat{c}_0 \end{bmatrix} \qquad \textbf{(5-54b)}$$

and

$$\tilde{u} = \left[\bar{a}^{x,m}_{k,l}\ \bar{a}^{y,m}_{k,l}\ a^{x,m}_{k,l}\ a^{y,m}_{k,l}\ a^{\phi,m}_{k,l}\right]^{\mathrm{T}} \tag{5-54c}$$

After elimination of $\{\bar{a}^m_{k,l}, \bar{b}^m_{k,l}\}$ and solving for $\{a^{x,m}_{k,l}, a^{y,m}_{k,l}, a^{\phi,m}_{k,l}\}$ we obtain a system of equations of the form

$$\begin{bmatrix} a^{x,m}_{k,l} \\ a^{y,m}_{k,l} \\ a^{\phi,m}_{k,l} \end{bmatrix} = S(\theta_x, \theta_y) \cdot \begin{bmatrix} a^{x,m-1}_{k,l} \\ a^{y,m-1}_{k,l} \\ a^{\phi,m-1}_{k,l} \end{bmatrix} \tag{5-55}$$

where S is defined as the error amplification matrix relating the Fourier error components before and after a relaxation sweep. Following [Brandt 1984][Stüben 1984] the smoothing factor is defined as

$$\bar{\mu} = \sup\{\rho(S(\theta_x, \theta_y))|\ (\theta_x, \theta_y) \in \theta_R\} \tag{5-56}$$

with $\rho(\cdot)$ the spectral radius, and the set of "rough" frequencies θ_R defined as (cf. Section 5.2.1)

$$\theta_R = T_h \backslash T_H \qquad T_h = (-\pi, \pi]^2 \qquad T_H = \left(-\frac{\pi}{2}, \frac{\pi}{2}\right]^2 \tag{5-57}$$

Next, we discuss some conclusions that can be drawn from the previous results.

Conclusions

In Table 5-8 the numerically evaluated behavior of the smoothing factor $\bar{\mu}$ is shown for a number of values of the ratio $\hat{c}_0/\hat{a}$. We observe that in the case $\hat{c}_0 = 0$ the Vanka relaxation is a good smoother and for increasing values of the source term the smoothing properties get better.

Table 5-8 Smoothing factor of the Vanka relaxation for a varying linear source term.

$\hat{c}_0/\hat{a}$	0	0.1	1	10	100
$\bar{\mu}$	0.58722	0.58137	0.53232	0.25586	0.09091

Whether the above results also approximately hold for the more general case of triangular and quadrilateral grids is difficult to answer. However, for grids with not too weirdly shaped elements, likely similar convergence results can be expected. If some regions behave badly we may add extra relaxations to these (localized) regions.

The above proposed relaxation method for the thermoelectric problem could be improved somewhat by the introduction of under-relaxation for the flux variables. In that case we make the following replacements in equation **(5-53a)**

$$\begin{aligned} \bar{u}^m_{i,j} &= \alpha^{-1}\left(\bar{u}^m_j - (1-\alpha)\,u^{m-1}_j\right) \\ u^m_{i,j} &= \alpha^{-1}\left(u^m_{i,j} - (1-\alpha)\,\bar{u}^m_{i,j}\right) \end{aligned} \qquad \textbf{(5-58)}$$

We may then carry out a similar analysis obtaining the smoothing factor as a function of the damping parameter α. For specific cases the optimum value of α can be numerically evaluated, however, because of its limited general scope we prefer not to use this technique.

5.4.7 Non-Linear Relaxation

In the case the set of equations to be solved is non-linear the relaxation method needs to be extended to the non-linear case. The principle for a non-linear relaxation method is simple [Brandt 1984](pp. 36); each time an element is visited a small system of non-linear equations of the form of equation **(5-34)** needs to be solved. For this purpose the iterative techniques described in Section 4.5 can be used. So, basically the relaxation equations are linearized with respect to the unknown quantities on the element, in our case the four fluxes at the edges and the potential at the barycenter, and subsequently solved by iteration. Note that the FAS multigrid rate of convergence is not constrained by the (global) convergence of the iteration method. It is still mainly determined by the interior smoothing rate. The effectivity of the non-linear relaxation method to reduce the high frequency error components can be studied by means of a local mode analysis of the linearized relaxation equations [Brandt 1984](pp. 84).

Note that the local linearization employed in the relaxation method has nothing to do with the global linearization usually used to deal with non-linear problems. It is by virtue of the multigrid method that such a complicated global linearization of the problem is not necessary. As long as the smoothing rate of the non-linear relaxation method is sufficient, non-linear problems are solved as efficient as the corresponding linear problem.

For problems with weakly non-linear coefficients often the successive substitution strategy (principal linearization) gives good results. However, for the semiconductor problem the exponential non-linearity of the coefficients (cf. Chapter 3) is expected to pose severe problems [Brandt 1984](pp. 36). For the relaxation of the thermoelectric equations we use a more sophisticated iteration method which is a combination of the Newton and the Gauss-Seidel method. Basically, we first perform one Newton step on the non-linear relaxation equations arising from the Poisson equation to update the electrostatic potential in the barycenter and the dielectric flux densities at the edges of the visited element. Subsequently, the result is used to initialize a Newton step on each of the non-linear relaxation equations arising from the carrier continuity equations to update the Fermi-potentials at the barycenter and the carrier fluxes at the edges. This result is on its turn used to initialize a Newton step on the non-linear relaxation equations arising from the heat balance equation to update the temperature at the barycenter and the heat fluxes at the edges. This process is repeated until a certain (local) accuracy is achieved, after which the next element is visited.

5.4.8 Calculation of the Lagrange Multipliers

In this section we derive a formula that can be used to calculate the Lagrange multiplier λ_i on an edge E_j. As discussed in Section 5.3.1, the purpose of the Lagrange multipliers is to serve as fine-grid Dirichlett boundary conditions when prolongating the solution to a fine-grid patch. In order to make the calculations strictly local, we express the Lagrange multiplier on an edge E_{01} in terms of the fluxes and potentials on the elements S_0 and S_1 neighboring that edge (cf. Figure 5-11).

The starting point is the mixed hybrid formulation discussed in Section 4.3. We proceed in a similar way as with the derivation of the relaxation operator, that is, we regard the relaxation domain as a tiny computational domain S for which we apply the mixed-hybrid discretization method. On the edges of the

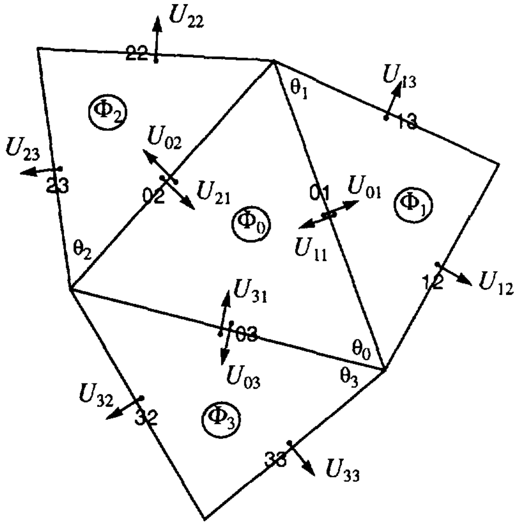

Figure 5-11 Domain for calculating the Lagrange multiplier on an internal edge. The numbers located on the edges denote the identification number of the edge, whereas the numbers shown off the edge denote the identification numbers of the element vectorial basis functions.

domain S, the vector of Lagrange multipliers Λ is formally given by (cf. equation (4-79))

$$D\Lambda = G_1 - B\Phi - AU \tag{5-59}$$

with

$$\begin{aligned} \Phi &= [\phi_0, \phi_1, \phi_2, \phi_3] \\ \Lambda &= [\lambda_{01}, \lambda_{02}, \lambda_{03}, \lambda_{12}, \lambda_{13}, \lambda_{22}, \lambda_{23}, \lambda_{32}, \lambda_{33}] \\ U &= [U_{01}, U_{02}, U_{03}, U_{11}, U_{12}, U_{13}, U_{21}, U_{22}, U_{23}, U_{31}, U_{32}, U_{33}] \end{aligned} \tag{5-60}$$

and D, B, A and G_1 given by equations (4-77a)-(4-77c).

If we assume that the solution vectors U and Φ on domain S are known vectors obtained from the previous relaxation sweep, we can solve (5-59) for the vector of Lagrange multipliers Λ. However, we observe that the matrix D is not a square matrix, hence its inverse cannot be calculated. In order to get around this problem, equation (5-59) is multiplied by the transpose of D. Now the matrix $D^T D$ is square and invertible, hence

$$\Lambda = (D^T D)^{-1} D^T [G_1 - B\Phi - ADU] \tag{5-61}$$

By expanding the above equation we can evaluate the Lagrange multipliers at the internal edges of the relaxation domain S

$$\lambda_{0k} = \frac{1}{2}\left[[\Phi]_0 - \sum_{i=1}^{N_0} [A]_{0k,0i}[U]_{0i} + [\Phi]_k - \sum_{i=1}^{N_k} [A]_{k1,ki}[U]_{ki} \right] \tag{5-62}$$

where the index k can take the values $\{1, \ldots, N_0\}$ and it is understood that the value of each flux is taken according to the sign with which the element uses the edge on which the flux is defined. For example, if element S_0 uses edge E_{01} with a positive sign and element S_1 uses edge E_{01} with a negative sign, then

$$U_{01} = -U_{11} = \text{value of flux stored in edge} \tag{5-63}$$

It is not difficult to show that equation (5-62) is also valid for a quadrilateral type of relaxation domain. Moreover, it can also be used on relaxation domains consisting of triangles and quadrilaterals.

Note 5-6: In the special case where the edge on which we wish to evaluate the Lagrange parameter is a Dirichlett edge, we can simply use the value of the Dirichlett boundary condition at that edge, and in the case the edge is a Neumann edge we can use

$$\lambda_{0k} = \left[[\Phi]_0 - \sum_{i=1}^{N_0} [A]_{0k,0i} [U]_{0i} \right] \quad \textbf{(5-64)}$$

5.5 Grid Transfer Operators

As discussed in Section 5.2.6 we need operators to transfer grid functions between the coarse and the fine grids, that is, interpolation to transfer the correction from the coarse to the fine grid, restriction to transfer the residuals and solutions from a fine to a coarse grid, and prolongation to transfer the solution from a coarse to a fine grid. In this section we will deal with these operators in the context of the mixed formulation. In doing so there are a few important considerations that need to be taken into account: (a) the intergrid operators should be such that the accuracy conditions are satisfied (cf. Section 5.2.6.4), and (b) the flux conserving properties of the discretization should not be spoiled by the intergrid operators. In the remainder of this section we will first discuss the interpolation operators for triangles and parallelograms. Next, we will discuss the restriction operators for the above mentioned cases. Finally, we discuss the prolongation operators.

5.5.1 Preliminaries

In principle we are free to choose the intergrid operators as long as they are accurate enough, however, we notice that the fine grid is obtained from the coarse grid by element-wise refinement as shown in Figure 5-5. This way nested grids are obtained and as a result the Raviart-Thomas approximation spaces (cf. Section 4.2.4.1) for the fine and coarse grid are also nested, i.e. $V^H(\Omega) \subset V^h(\Omega)$ and $W^H(\Omega) \subset W^h(\Omega)$. This means that we may use the intergrid operators that are suggested by the discretization method. For the Raviart-Thomas elements of lowest order this comes down to piecewise constant interpolation for the potential and piecewise linear interpolation for the fluxes. The restriction operators can be obtained by transposing the interpolation operators, because the test and trial functions in the mixed formulation are identical (Galerkin method).

Since the mixed formulation essentially consists of two coupled first order PDEs the accuracy conditions (cf. Section 5.2.6.4) can be formulated as

$$\begin{aligned} &m_I^u + m_R^u > 1 \quad && m_i^u = 2, m_R^u = 2 \\ &m_I^\phi + m_R^\phi > 1 \quad && m_I^\phi = 1, m_R^\phi = 1 \end{aligned} \quad \textbf{(5-65)}$$

Hence, in principle the natural interpolation and restriction operators satisfy the accuracy conditions. Moreover, for a proper full multigrid algorithm we at least need

$$m_p^u \geq m_c^u = 2$$
$$m_p^\phi \geq m_c^\phi = 2 \tag{5-66}$$

Hence, for the prolongation of the fluxes we can use the natural interpolation operator, however, for the prolongation of the potentials we need to construct a higher order operator. For this purpose we use the post processing technique discussed in Section 4.3.4.

5.5.2 Interpolation Operators

We now discuss the natural interpolation operators suggested by the Raviart-Thomas approximation spaces in more detail. We use these operators to transfer the error on the coarse grid to the fine grid (cf. equation (5-15)). Since the potentials are approximated by piecewise constant functions and the fluxes by piecewise linear functions we may use piecewise constant interpolation for the potentials and piecewise linear interpolation for the fluxes. First, we discuss interpolation on triangular elements and, second, the interpolation on rectangular elements.

Interpolation on Triangular Elements

It suffices to consider a coarse-grid reference triangle refined into four congruent fine-grid triangles. Figure 5-12 shows such a refinement together with the local numbering of the elements, edges, fluxes and potentials. We first

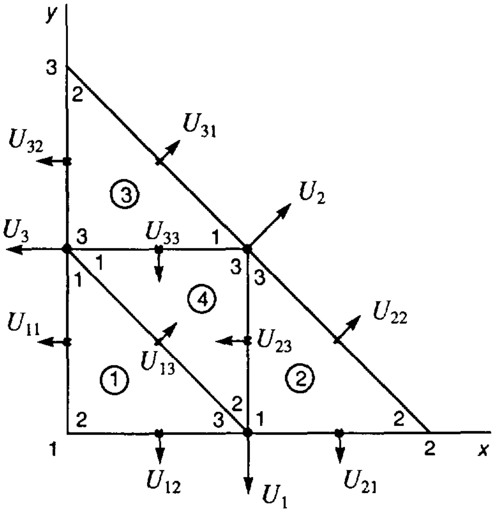

Figure 5-12 Interpolation on a triangle, coarse nodal points are indicated with a dot and fine nodal points are indicated with a cross. Also the local node numbering of the coarse and fine-grid triangles are shown.

discuss the interpolation operator for the flux part of the error, and, second, the interpolation operator for the potential part of the error.

To design a proper interpolation operator I_H^h for the flux part of the error we note that for the lowest-order Raviart-Thomas approximation (cf. Section 4.2.4.1) the flux $u^H(x, y)$ on a coarse-grid reference triangle can be written as a linear combination of its three vectorial basis functions

$$u^H(x, y) = U_1^H \begin{bmatrix} x \\ y-1 \end{bmatrix} + U_2^H \begin{bmatrix} x \\ y \end{bmatrix} + U_3^H \begin{bmatrix} x-1 \\ y \end{bmatrix} \qquad \text{(5-67)}$$

where the U_k^H represent the net fluxes leaving the edges of the coarse triangle on which they are defined. The interpolated value of the flux at edge E_{ij} of a fine-grid triangle can be calculated by projecting the coarse-grid flux on to the normal vector n^{ij} of the edge E_{ij} followed by integration along the edge, hence

$$U_{ij}^h = \int_{E_{ij}} (u^H \cdot n^{ij})\, ds \qquad \text{(5-68)}$$

where i indicates the child element and j indicates the edge. Evaluating the above integral for all the child edges yields the following relation between the coarse-grid and fine-grid fluxes

$$\begin{bmatrix} U_{11}^h \\ U_{12}^h \\ U_{13}^h \\ U_{21}^h \\ U_{22}^h \\ U_{23}^h \\ U_{31}^h \\ U_{32}^h \\ U_{33}^h \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1/2 \\ 1/2 & 0 & 0 \\ -1/4 & 1/4 & -1/4 \\ 1/2 & 0 & 0 \\ 0 & 1/2 & 0 \\ -1/4 & -1/4 & 1/4 \\ 0 & 1/2 & 0 \\ 0 & 0 & 1/2 \\ 1/4 & -1/4 & -1/4 \end{bmatrix} \cdot \begin{bmatrix} U_1^H \\ U_2^H \\ U_3^H \end{bmatrix} \qquad \text{(5-69)}$$

The element-wise interpolation operator for the flux part of the correction follows immediately from equation (5-69) and is given by

$$I_H^h = I_k^{k+1} = \begin{bmatrix} 0 & 0 & 1/2 \\ 1/2 & 0 & 0 \\ -1/4 & 1/4 & -1/4 \\ 1/2 & 0 & 0 \\ 0 & 1/2 & 0 \\ -1/4 & -1/4 & 1/4 \\ 0 & 1/2 & 0 \\ 0 & 0 & 1/2 \\ 1/4 & -1/4 & -1/4 \end{bmatrix} \tag{5-70}$$

Note 5-7: The above definition of the interpolation operator is not only valid for the reference triangle but also for general triangles.

Note 5-8: The above definition of the interpolation operator clearly preserves the current conservation property when going from the coarse grid to the fine grid, because the total flux through the fine-grid edges that have a coarse-grid edge in common equals the total flux through the coarse-grid edge.

To find the interpolation operator for the potential part of the error we note that the Raviart-Thomas approximation to the potential $\Phi^H(x, y)$ on the coarse-grid triangle is piecewise constant. This suggests the following relation between the coarse-grid and fine-grid potentials

$$\begin{bmatrix} \Phi_1^h \\ \Phi_2^h \\ \Phi_3^h \\ \Phi_4^h \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \cdot \begin{bmatrix} \Phi^H \end{bmatrix} \tag{5-71}$$

Hence, the element-wise interpolation operator I_H^h for the potential part of the error is given by

$$I_H^h = I_k^{k+1} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \tag{5-72}$$

Interpolation on Square Elements

It suffices to consider a square coarse-grid reference element refined into four congruent fine-grid squares. Figure 5-13 shows such a refinement together with the local numbering of the elements, edges, fluxes and potentials. The lowest order Raviart-Thomas approximation of the flux on the coarse-grid element is given by

$$u^H(x,y) = U_1^H\begin{bmatrix}0\\y-1\end{bmatrix} + U_2^H\begin{bmatrix}x\\0\end{bmatrix} + U_3^H\begin{bmatrix}0\\y\end{bmatrix} + U_4^H\begin{bmatrix}x-1\\0\end{bmatrix} \tag{5-73}$$

where the U_k^H, $(k = 1, 2, 3)$ represent the net fluxes leaving the edges of the coarse-grid element. In a similar fashion as in the triangular case we obtain for the relation between the coarse-grid and fine-grid fluxes

$$\begin{bmatrix}U_{11}^h\\U_{12}^h\\U_{13}^h\\U_{21}^h\\U_{22}^h\\U_{23}^h\\U_{31}^h\\U_{32}^h\\U_{33}^h\\U_{41}^h\\U_{42}^h\\U_{43}^h\end{bmatrix} = \begin{bmatrix}0 & 0 & 0 & 1/2\\ 1/2 & 0 & 0 & 0\\ 0 & 1/4 & 0 & -1/4\\ 1/2 & 0 & 0 & 0\\ 0 & 1/2 & 0 & 0\\ -1/4 & 0 & 1/4 & 0\\ 0 & 1/2 & 0 & 0\\ 0 & 0 & 1/2 & 0\\ 0 & -1/4 & 0 & 1/4\\ 0 & 0 & 1/2 & 0\\ 0 & 0 & 0 & 1/2\\ 1/4 & 0 & -1/4 & 0\end{bmatrix} \cdot \begin{bmatrix}U_1^H\\U_2^H\\U_3^H\\U_4^H\end{bmatrix} \tag{5-74}$$

Hence, the element-wise interpolation operator for the flux part of the error I_H^h is given by

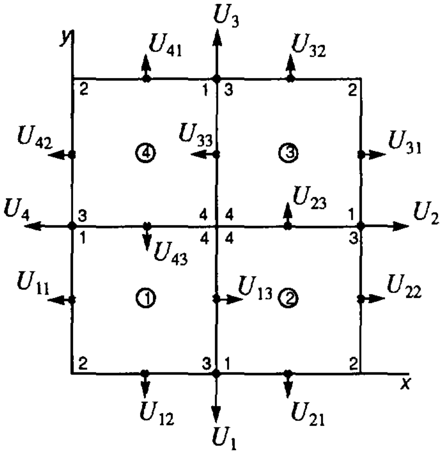

Figure 5-13 Interpolation on square elements, coarse nodal points are indicated with a dot and fine nodal points are indicated with a cross.

$$I_H^h = I_k^{k+1} = \begin{bmatrix} 0 & 0 & 0 & 1/2 \\ 1/2 & 0 & 0 & 0 \\ 0 & 1/4 & 0 & -1/4 \\ 1/2 & 0 & 0 & 0 \\ 0 & 1/2 & 0 & 0 \\ -1/4 & 0 & 1/4 & 0 \\ 0 & 1/2 & 0 & 0 \\ 0 & 0 & 1/2 & 0 \\ 0 & -1/4 & 0 & 1/4 \\ 0 & 0 & 1/2 & 0 \\ 0 & 0 & 0 & 1/2 \\ 1/4 & 0 & -1/4 & 0 \end{bmatrix} \qquad \text{(5-75)}$$

Note 5-9: The above definition of the interpolation operator is also valid for rectangles and parallelograms provided the basic refinement as shown in Figure 5-5 is used.

Note 5-10: The above definition of the interpolation operator also preserves the current conservation property.

For the interpolation of the potentials the result is the same as in the triangular case, that is, we can use equations (5-71) and (5-72).

5.5.3 Restriction Operators

In order to transfer fine-grid residuals, right-hand sides and solutions to the coarse grid (cf. equation (5-13)), restriction operators need to be defined. The natural restriction operators that come with the mixed discretization are in principle, apart from a constant scaling factor, given by the transpose of the interpolation operators. This property follows from the Galerkin type weak formulation (cf. Section 4.2.2) [Wesseling 1992]. However, some care must be taken because we cannot simply take the transpose of the local interpolation operators as given in the previous section. For a proper definition of the restriction operators we also must take into account the contributions of the neighboring elements. For this purpose it suffices to consider the relaxation subdomains as shown in Figure 5-9 and Figure 5-10. First, we discuss the restriction on triangular elements and, second, the restriction on square elements.

Restriction on Triangular Elements

Before the restriction operators R_h^H and $\tilde{R}_h^H$ can be defined, we need a convention for the numbering of the coarse-grid and fine-grid elements, edges and vertices. For this purpose we consider the relaxation subdomain in Figure 5-9, for which we assume that each element is refined as shown in Figure 5-12. We then proceed by constructing the interpolation operator for the entire relaxation subdomain. From the transpose of this interpolation operator we can extract the natural restriction operator that comes with the mixed discretization. Using a scaling factor of 2 to make the fine-grid and coarse-grid approximation to the flux consistent the natural restriction operator reads

$$R_h^H : \begin{cases} U_{01}^H = U_{12}^{h,0} + U_{21}^{h,0} - \frac{1}{2}\left[U_{13}^{h,0} + U_{23}^{h,0} - U_{33}^{h,0} - U_{13}^{h,1} - U_{23}^{h,1} + U_{33}^{h,1}\right] \\ U_{02}^H = U_{22}^{h,0} + U_{31}^{h,0} - \frac{1}{2}\left[U_{13}^{h,0} + U_{23}^{h,0} - U_{33}^{h,0} - U_{13}^{h,2} - U_{23}^{h,2} + U_{33}^{h,2}\right] \\ U_{03}^H = U_{32}^{h,0} + U_{11}^{h,0} - \frac{1}{2}\left[U_{13}^{h,0} + U_{23}^{h,0} - U_{33}^{h,0} - U_{13}^{h,3} - U_{23}^{h,3} + U_{33}^{h,3}\right] \end{cases} \tag{5-76}$$

where the superscripts and subscripts must be interpreted as follows. Coarse-grid quantities are indicated by a superscript H. Similarly, fine-grid quantities are indicated by a superscript h. For the subscripts the first digit indicates the element number and the second digit indicates the edge number. The additional digit in the superscript for the fine-grid quantities indicates the parent coarse-grid element.

We use the above defined restriction operator to transfer the residuals associated with the fluxes from the fine grid to the coarse grid. However, if the same restriction operator is used to transfer the flux from the fine grid to the coarse grid, it is important to notice that the restriction operator as proposed above is not current preserving in the sense that the net flux through a coarse-grid edge equals the net flux through its fine-grid child edges. This is only the case if the second part of each of the right-hand sides of equation (5-76) vanishes, which holds for sufficiently smooth fluxes. For the choice of the restriction operator $\tilde{R}_h^H$ for the vector part of the solution u^h, we use the following heuristic argument. Recapitulating the construction of the right-hand side in the coarse-grid correction (cf. equation (5-13))

$$L^H(u^H) = L^H(\tilde{R}_h^H \tilde{u}^h) + R_h^H r^h \tag{5-77}$$

we observe that at convergence the residue is approximately zero and we must have

$$u^H = \tilde{R}_h^H \tilde{u}^h \tag{5-78}$$

So, if we define $\tilde{R}_h^H$ by

$$\tilde{R}_h^H : \begin{cases} U_{01}^H = U_{12}^{h,0} + U_{21}^{h,0} \\ U_{02}^H = U_{22}^{h,0} + U_{31}^{h,0} \\ U_{03}^H = U_{32}^{h,0} + U_{11}^{h,0} \end{cases} \tag{5-79}$$

then at convergence the net flux through a coarse-grid edge equals the net flux through its corresponding fine-grid child edges. This choice of $\tilde{R}_h^H$ also ensures current conservation at the green edges (cf. Section 5.3.1) of a grid patch by ensuring that at a coarse grid edge that has two green child edges, the net flux through the coarse-grid edge equals the net flux through its green child edges.

The derivation of the restriction operators for the potential and the corresponding residue is trivial and follows by transposing equation (5-71). Referring to Figure 5-12 the restriction operator is given by

$$R_h^H, \tilde{R}_h^H : \Phi^H = \frac{1}{4}\begin{bmatrix}1 & 1 & 1 & 1\end{bmatrix} \cdot \begin{bmatrix}\Phi_1^h \\ \Phi_2^h \\ \Phi_3^h \\ \Phi_4^h\end{bmatrix} \tag{5-80}$$

where the factor 1/4 is a suitable scaling factor. Note that we use the above restriction for both the potentials and the corresponding residuals.

Restriction on Square Elements

As in the triangular case we need a convention for the numbering of the coarse-grid and fine-grid elements, edges and vertices. For this purpose we consider the relaxation subdomain in Figure 5-10, for which we assume that each element is refined as shown in Figure 5-13. In a similar fashion as in the triangular case the natural restriction operator R_h^H that comes with the mixed discretization reads

$$R_h^H : \begin{cases} U_{01}^H = U_{12}^{h,0} + U_{21}^{h,0} - \frac{1}{2}U_{23}^{h,0} + \frac{1}{2}U_{43}^{h,0} - \frac{1}{2}U_{43}^{h,1} + \frac{1}{2}U_{23}^{h,1} \\ U_{02}^H = U_{22}^{h,0} + U_{31}^{h,0} - \frac{1}{2}U_{33}^{h,0} + \frac{1}{2}U_{13}^{h,0} - \frac{1}{2}U_{43}^{h,2} + \frac{1}{2}U_{23}^{h,2} \\ U_{03}^H = U_{32}^{h,0} + U_{41}^{h,0} - \frac{1}{2}U_{43}^{h,0} + \frac{1}{2}U_{23}^{h,0} - \frac{1}{2}U_{43}^{h,3} + \frac{1}{2}U_{23}^{h,3} \\ U_{04}^H = U_{42}^{h,0} + U_{11}^{h,0} - \frac{1}{2}U_{13}^{h,0} + \frac{1}{2}U_{33}^{h,0} - \frac{1}{2}U_{43}^{h,4} + \frac{1}{2}U_{23}^{h,4} \end{cases} \tag{5-81}$$

where the notation is similar to the one used in the triangular case. The above restriction operator is used to transfer the vector residuals from the fine grid to the coarse grid.

We use the above restriction operator to transfer the residue associated with the flux to the coarse grid. To define the restriction operator for the vector part of the solution we may proceed in a similar fashion as in the triangular case. Skipping the details, the restriction operator $\tilde{R}_h^H$ for the flux is defined by

$$R_h^H : \begin{cases} U_{01}^H = U_{12}^{h,0} + U_{21}^{h,0} \\ U_{02}^H = U_{22}^{h,0} + U_{31}^{h,0} \\ U_{03}^H = U_{32}^{h,0} + U_{41}^{h,0} \\ U_{04}^H = U_{42}^{h,0} + U_{11}^{h,0} \end{cases} \tag{5-82}$$

The restriction for the potential and the corresponding residue is the same as in the triangular case, hence, we can use equation (5-80).

5.5.4 Prolongation Operators

Taking into account the accuracy conditions discussed in Section 5.5.1, we observe that the order of the natural interpolation operation for the potential is too low. Hence, in contrast to the natural interpolation operator for the flux, the natural interpolation operator for the potential cannot be used as a prolongation operator. So, for the prolongation of the fluxes, we simply use the natural interpolation operator that comes with the mixed discretization. However, for the prolongation of the potentials we propose the following construction.

An effective way to increase the order of the interpolation for the potential is to use the post-processing technique introduced in Section 4.3.4. For the triangular case, we calculate the Lagrange multipliers defined on the edges of the triangle (cf. Section 5.4.8) and subsequently use these to construct a piecewise linear approximation of the potential on the element (cf. Section 4.3.4). The potentials in the fine-grid elements can then be obtained by evaluating the piecewise linear potential at the barycenter of each of the child elements. Using the results from Section 4.3.4 we define the prolongation operator for the potential on triangular elements as

$$P_H^h : \begin{cases} \Phi_1^h = \frac{1}{3}\,[2\lambda_1^H - \lambda_2^H + 2\lambda_3^H] \\ \Phi_2^h = \frac{1}{3}\,[2\lambda_1^H + 2\lambda_2^H - \lambda_3^H] \\ \Phi_3^h = \frac{1}{3}\,[2\lambda_2^H - \lambda_1^H + 2\lambda_3^H] \\ \Phi_4^h = \frac{1}{3}\,[\lambda_1^H + \lambda_2^H + \lambda_3^H] \end{cases} \tag{5-83}$$

For the quadrilateral case a bilinear approximation of the flux is constructed by using the Lagrange multipliers at the edges. The potentials in the fine-grid elements can again be obtained by evaluating the piecewise bilinear potential at the barycenter of each of the child elements. Using the results from Section 4.3.4 we define the prolongation operator for the potential on quadrilateral elements as

$$P_H^h: \begin{cases} \Phi_1^h = \frac{1}{2}[\lambda_1^H + \lambda_4^H] \\ \Phi_2^h = \frac{1}{2}[\lambda_2^H + \lambda_1^H] \\ \Phi_3^h = \frac{1}{2}[\lambda_3^H + \lambda_2^H] \\ \Phi_4^h = \frac{1}{2}[\lambda_4^H + \lambda_3^H] \end{cases} \tag{5-84}$$

This concludes the section on the intergrid operators. Basically, we now have available a consistent and accurate set of intergrid operators on grids with triangles and parallelograms.

5.5.5 Damping of the Restriction Operator

According to the Multigrid Guide [Brandt 1984] in some special situations the dominant solution-dependent term in the coefficients of the PDE may have the form $g(u)$, where g is a sensitive function, that is, large variations in g are caused by more-or-less normal changes in u over a meshsize. In such cases the restriction operator should have the special form

$$u^H = g^{-1}(R_h^H g(u^h)) \tag{5-85}$$

where R_h^H is the restriction operator. The above situation obviously applies to the thermoelectric problem because the coefficients depend on the carrier densities n^e and n^h, which respectively are exponentially varying functions of the Fermi-levels E_f^e and E_f^h.

Effectively, equation **(5-85)** proposes to apply the restriction operator to the carrier densities instead of the Fermi-levels. The Fermi-levels on the coarse grid are then reconstructed by applying the inverse transform. However, for some yet unknown reason this method does not seem to work properly for the thermoelectric problem. A better method aiming at a damping of the restricted residuals is described in [Molenaar 1992].

5.6 Adaptive Multigrid

There are several issues related to adaptive multigrid and in the literature there seems to be some diversification with respect to this subject. In general an adaptive multigrid algorithm may support a combination of (a) adaptive grids, (b) adaptive cycling, (c) adaptive relaxation, and (d) adaptive order of discretization. The grid adaptivity (a) refers to the local refinement of a grid by using a suitable error criterium. The adaptive cycling (b) feature refers to the capability of the algorithm to adapt its cycling parameters to the specifics of the problem to be solved. The adaptive relaxation (c) feature refers to the capability of the algorithm to (1) decide if the convergence rate of the relaxation slows down and (2) to decide if certain areas of the domain need additional relaxation. Finally, the adaptive order of discretization (d) refers to the capability of the algorithm to choose higher order discretizations in regions where the solution is sufficiently smooth. In this section we are only concerned with the subject of adaptive grids.

Note that in the case of singular perturbation problems, such as the thermoelectric problem, the minimum amount of adaptivity that should be supported, in order to achieve the optimal convergence bounds, is grid adaptivity [Brandt 1979] .. The general guiding factor in adaptivity is that we must question how much effort it takes to achieve a certain local accuracy. In other words, in the case the local problem is already accurate the algorithm should not spend a lot of effort to make it even more accurate, instead it should pay attention to regions which are not yet accurate enough. The difficulty is in finding suitable criteria that provide estimates for the already attained local accuracy.

Although the adaptive cycling strategy potentially makes the method very robust, it in general is a nuisance when checking the performance of the individual multigrid components. Only after the various multigrid components are tested for their performance we can think of adaptive cycling. We, therefore, consider fixed cycling strategies and leave the adaptive cycling strategies for future investigations.

Moreover, we only deal with a special kind of adaptive grids which are called composite grids (cf. Section 5.3). In contrast to the treatment of composite grids in [McCormick 1989], where rectangular domains regularly partitioned into square elements are considered, we allow any shape of domain as long as it can be represented by a quasi-regular partitioning into triangles and quadrilaterals (cf. Section 5.3).

The structure of this section is as follows. In Section 5.6.1, we discuss the extension of the basic full multigrid algorithm (cf. Table 5-6) such that it supports composite grids.

5.6.1 Adaptive Local Grid Refinement

In this section we discuss an extension to the basic multigrid algorithm such that it supports composite grids. From the algorithmic perspective the basic question to be answered is “when and where” to refine the grid at a certain grid level k. The “when” refers to the actual place in the multigrid cycle where the refinement should take place. The “where” refers to a suitable local error criterium that we can use to decide whether an element needs to be refined or not. The starting point of our discussion is the full multigrid algorithm presented in Table 5-6. In Section 5.6.1.1, we discuss how to extend this algorithm to support a given composite grid. Next, in Section 5.6.1.2, an extension to the algorithm is discussed such that it also creates the composite grid during execution. In Section 5.6.1.3, the error estimators in the context of the mixed discretization are derived. Finally, in Section 5.6.1.4, some concluding remarks are given.

5.6.1.1 Fixed Composite Grids

In this section we extend the algorithm presented in Table 5-6 such that it calculates a solution on a given composite grid. Such a composite grid could be obtained by a careful analysis of the truncation error of the expected singularities (e.g. right-hand side singularities, structural singularities and boundary layers) in the model equations [Bai 1987]. The extension to self-adaptation is discussed in the following section. If we think about this problem various possible ways, each with its specific advantages and disadvantages, can be devised. In fact, we enter a rather difficult research field in computational mathematics where many basic issues have not been fully explored. This fact makes it very difficult to select the most appropriate algorithm. However, since it is not our intention to provide an optimal algorithm, we follow an approach that is led by heuristics rather than mathematical rigor.

We first define the basic notations that we need. Let the sequence of nested grids be given by $\{G_k\}_{k \in M}$, with $M = \{0, \ldots, K-1\}$ and k the level of refinement. Let the set of disjoint grid patches of grid level k be given by $\{P_k^l\}_{l \in N}$, with $N = \{0, \ldots, L-1\}$ and l the patch number. Obviously, the union of the patches P_{k+1}^l to grid G_k constitutes grid G_{k+1}, that is

$$G_{k+1} = \bigcup_{l=0}^{L-1} P_{k+1}^l \tag{5-86}$$

Moreover, the grid G_k^l obtained by restriction of grid G_k to the region covered by a patch P_{k+1}^l of grid G_{k+1} is defined as

$$G_k^l = G_k \cap P_{k+1}^l \tag{5-87}$$

Obviously, grid G_k^l contains those grid cells of G_k that lie in the region covered by patch P_{k+1}^l. Further, the part of grid G_k not covered by grid patches from the set $\{P_{k+1}^l\}_{l \in N}$ is denoted as

$$\overline{G}_k = G_k \backslash (G_k \cap G_{k+1}) \tag{5-88}$$

The most obvious and perhaps the "safest" way of implementing the treatment of the grid patches is to deal with all patches of a given grid level on an equal footing[7]. This means that the prolongation, restriction and interpolation operators effectively operate on all patches of a grid level simultaneously. The modification of the algorithm in Table 5-6 for the above case is

[7] This method is used by most other approaches dealing with grid adaptivity.

straightforward; all it needs is a redefinition on the grid transfer operators: prolongation, restriction and interpolation.

Prolongation

The prolongation operator P_k^{k+1} must be redefined such that it prolongates the converged solution at the subgrids G_k^l of grid G_k to each of the grid patches P_{k+1}^l, hence

$$u_l^{k+1} = \tilde{P}_k^{k+1} u^k \qquad \forall G_k^l \tag{5-89}$$

Note that the prolongation operator also should provide the boundary conditions on the boundary of a grid patch. In Section 5.3.1 we discussed the use of the prolongated coarse-grid Lagrange multipliers as Dirichlett boundary conditions at those parts of the fine-grid patch boundaries which do not coincide with the true boundary of the computational domain. For those parts of the patch boundary the prolongation operator for the Lagrange multipliers is defined by a straightforward piecewise constant interpolation, similar to the prolongation operator for the potentials (cf. Section 5.5.4). At the part of the patch boundary which coincides with the true boundary of the computational physical domain we simply prolongate the values of the specified coarse-grid boundary conditions. For the Dirichlett as well as the Neumann boundaries we use piecewise constant interpolation[8].

The implementation is straightforward and is achieved by looping over the elements of each of the fine-grid patches and carrying out the usual prolongation (cf. Section 5.5.4) for each element. Note that the coarse-grid data needed for the prolongation is available through a pointer that points to the parent element. Moreover, for each fine-grid element that has green edges we also calculate the Lagrange parameter on the corresponding coarse-grid edge and subsequently prolongate it to the green edges of the fine-grid element. In a similar fashion, when the fine-grid element has (true) boundary edges, the appropriate boundary conditions are prolongated.

Restriction

The restriction operators $\tilde{R}_{k+1}^k$ and R_{k+1}^k must be redefined such that they restrict the solution and the residue on each of the fine grid patches G_{k+1}^l to the corresponding G_k^l of the coarse grid G_k. Moreover, the coarse grid problem needs to be redefined because no restrictions are available at the coarse-grid region $\overline{G}_k$. Therefore at the coarse-grid region $\overline{G}_k$ the role of the coarse-grid problem must be changed to the calculation of a solution instead of a correction. We now redefine the coarse-grid problem as

[8] For the prolongation of the Neumann boundary conditions we must keep a scaling factor of 1/2 in mind in order to maintain current conservation. This fact is directly related to the fact that a flux degree of freedom at an edge represents the net current through that edge.

$$L^k(u^k) = \begin{bmatrix} f^k \quad \text{on } \overline{G}_k \\ L^k(\tilde{R}^k_{k+1}u^{k+1}) + R^k_{k+1}r^{k+1} \quad \text{on each } G^l_k \end{bmatrix} \tag{5-90}$$

Clearly, on the parts G^l_k of G_k we have the familiar correction scheme and on $\overline{G}_k$ we solve the plain coarse grid problem.

The implementation is straightforward and is achieved by looping over the elements of each of the fine-grid patches and carrying out the usual restrictions (cf. Section 5.5.3). We can then relax on the coarse grid in the usual manner by applying equation (5-34). However, in equation (5-34) the term corresponding to the right-hand side of the coarse-grid problem should be chosen in accordance with equation (5-90), that is, if the element which is currently relaxed does not have childs we choose the upper term, and if the element does have childs we choose the lower term of the right-hand side of equation (5-90).

Coarse Grid Correction

The interpolation operator I^{k+1}_k in the coarse-grid correction step must be redefined such that it provides an improved initial guess to the solution on a fine-grid patch. Again the implementation is straightforward; we simply loop over the elements of each of the fine-grid patches and carry out the coarse-grid correction by means of usual interpolation operators (cf. Section 5.5.2).

Note 5-11: The above schemes can operate very efficiently by virtue of the advanced data structures we use to represent the grid levels and grid patches (cf. Chapter 6).

5.6.1.2 Adaptive Composite Grids

In this section we extend the algorithm discussed in the previous section such that it also creates the composite grid during execution. For many practical cases this is very convenient, because the order and location of the singularities are not known in advance, or because their behavior is of dynamic nature. For these cases we need to extract criteria from the emerging solution in order to decide when and where to create new grid patches. To be more specific, we need some method that estimates the local error on an element.

In sophisticated traditional adaptive finite element methods the local error is based on a residual analysis, that is, for a calculated discrete solution we seek a posteriori global and local error bounds for the solution in terms of computable element residuals. The grid refinement is then organized such that the local error on each element is smaller than some prescribed tolerance regardless of the amount of work needed to achieve it [Carey 1984]. It is also possible to relate the error and the work needed to achieve it, and formulate the refinement scheme such that it minimizes the error for a given amount of expendable work [Bai 1987].

An important advantage of the multigrid method is that it enhances the choice of conventional error estimators; in addition to using the data on the present level of refinement to solve the local error approximation equations, the approximations on the various refinement levels can be used to quantify the known form of the error. One could say that this method of local error estimation is more or less natural to the multigrid method. For example, if the order of consistency (truncation error) of the discretization is of the form ch^2, with c an unknown constant, then approximations with two different mesh sizes can be used to determine the constant c and, hence, the error on the finest level. In the case that the truncation error is of the general form ch^p with c and p unknown, three levels can be used for the determination of c and p. As pointed out in [McCormick 1989], care must be taken with this approach since in practice the error may not behave according to the form ch^p, even when the convergence properties of the discretization indicate that the truncation error is bounded by such a form. This is especially true on very coarse grids. In [McCormick 1989] it is proposed to use the multi-level error estimation not as a replacement but rather as a supplement to the conventional local error estimators.

However, in practice we have obtained satisfactory results with the multi-level error estimator, and we therefore confine the discussion to this type of error estimation. An investigation towards the possible benefits of the use of local error estimators is left for future research.

The multi-level error estimator is based on the dual representation of the full approximation scheme (FAS) (cf. Section 5.2.3). Let us first recapitulate the right-hand side used in the FAS scheme

$$L^k(u^k) = L^k(\tilde{R}^k_{k+1}u^{k+1}) + R^k_{k+1}[f^{k+1} - L^{k+1}(u^{k+1})] \tag{5-91}$$

The dual representation of the FAS scheme can now be obtained by rewriting the above relation as

$$L^k(u^k) = R^k_{k+1}f^{k+1} + \tau^k_{k+1} \tag{5-92}$$

with τ^k_{k+1} defined by

$$\tau^k_{k+1} = L^k(\tilde{R}^k_{k+1}u^{k+1}) - R^k_{k+1}L^{k+1}(u^{k+1}) \tag{5-93}$$

In order to interpret the quantity τ^k_{k+1} we proceed as follows. First, we observe that equation (5-92) without the τ^k_{k+1} term is the original coarse-grid equation. Second, we observe that the solution u^k is intended to approximate $\tilde{R}^k_{k+1}u^{k+1}_{new}$, and at convergence $\tilde{R}^k_{k+1}u^{k+1}$. Hence τ^k_{k+1} is a fine-to-coarse defect-correction, that is, a correction to make the solution of the coarse-grid equations coincide with the fine-grid solution. According to [Brandt 1984] this leads to a dual interpretation of the FAS scheme where, instead of regarding the coarse grid as a device for accelerating convergence on the fine grid, the fine grid is viewed as a device for calculating a correction τ^k_{k+1} to the coarse-grid equations.

In the FAS-FMG algorithm the fine-to-coarse defect-correction τ^k_{k+1} is used as a local error estimator which is motivated as follows. Basically, the fine-to-coarse defect-correction is an approximation to the local truncation error τ^k on the coarse grid defined by

$$\tau^k = L^k(\tilde{R}^k u) - R^k L(u) \tag{5-94}$$

where u is the true solution of the differential equation, and $\tilde{R}^k$, R^k are restriction operators that transfer the continuum solution to the coarse grid. Essentially, τ^k is a defect-correction that makes the solution of the coarse-grid problem coincide with the true solution of the differential equation. Note the analogy between equations (5-93) and (5-94). According to [Brandt 1984] we must have

$$\tau^k \equiv \tau^{k+1} + \tau^k_{k+1} \tag{5-95}$$

up to high order terms in h. In the case the order of consistency of the discretization is given by ch^p, i.e. $\tau^k \approx ch^p_k$, we can write

$$\tau^{k+1} = \left[\frac{1}{2^p - 1}\right]\tau^k_{k+1} \tag{5-96}$$

In the case of the mixed discretization we have $p = 2$. Because of the analogy to the local truncation error, τ^k_{k+1} is commonly referred to as the relative local truncation error, that is, the local truncation error of the coarse grid relative to the fine grid.

Clearly, τ^k_{k+1} is conveniently obtained as a by-product of the FAS scheme and can be calculated on the coarse grid in order to determine where the fine grid needs to be refined. We use the following refinement condition

$$\left|\tau^{k-1}_k(x, y)\right| \le 2^{-p}\tau_{sup} \qquad \tau_{sup} = \sup_{(x, y) \in G_{k-1}} \left|\tau^{k-1}_k(x, y)\right| \tag{5-97}$$

where p is the order of consistency of the discretization, which in our case is 2. Clearly, if equation (5-97) is violated for some $\tau^{k-1}_k(x, y)$ the corresponding location (x, y) should be within a fine-grid refinement patch.

Effectively, we now have answered the question where to refine the grid which leaves us to answer the question when to refine the grid. This question can easily be answered by the following argument. The relative local truncation error τ^k_{k+1} only approximates the actual truncation error τ^k if it is calculated by means of a converged fine-grid solution. Note that in this context converged means an algebraic error well below the discretization error. In the FAS-FMG scheme this is achieved[9] just before the solution is prolongated to the next finer grid. This means that the prolongation to the next finer grid G_{k+1} must be preceded by a calculation of the relative truncation error τ_k^{k-1} of the coarse grid G_{k-1}. The relative truncation error τ_k^{k-1} is then used to decide where grid G_k needs to be refined.

Note 5-12: The refinement of the coarse grid must be homogeneous because at the coarse grid the relative local truncation error is not yet available.

This leaves us the task of the calculation of the relative truncation error in the context of the mixed formulation of the problem.

5.6.1.3 Error Criteria Obtained from the Mixed Formulation

This section specifically deals with the calculation of the relative truncation errors in the context of the mixed formulation. Clearly, for the mixed formulation two relative truncation errors can be constructed; one for the potential and one for the flux. The relative truncation error for the potential is associated with the interior of an element and the relative truncation error for the flux is associated with the edges of an element. Clearly, the relative truncation error of the potential can be used to decide whether the interior of an element needs to be refined, whereas the relative truncation error of the flux can be used to decide whether an edge needs to be refined. The refinement of an edge is implemented by refinement of the elements neighboring that edge. Moreover, when several potentials and fluxes are defined, for each a relative truncation error is constructed. The overall refinement criterium for the elements is defined by a linear combination of the relative truncation errors of the potentials, and the refinement criterium for the edges is defined by a linear combination of the relative truncation errors of the fluxes.

We now calculate the relative truncation errors resulting from the mixed formulation. We assume the mixed formulation on grid level k has the following non-linear appearance

$$M^k(U^k, \Phi^k) \equiv \begin{bmatrix} A^k & B_1^k \\ B_2^k & C^k \end{bmatrix} \begin{bmatrix} U^k \\ \Phi^k \end{bmatrix} - \begin{bmatrix} G^k \\ F^k \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{5-98}$$

where the matrix and vector entries possibly depend on the solution vector. As in the case of the derivation of the relaxation operator, we take the subdomains as shown in Figure 5-9 and Figure 5-10 as the starting point. After a complete relaxation sweep the equations for the relative truncation errors on

[9] If this is not achieved the FAS-FMG algorithm is not properly used and we need to increase the number of cycles per level.

the elements $\tau^{\phi,k}_{k+1}$ and the edges $\tau^{u,k}_{k+1}$ of the relaxation subdomain can be written as

$$\begin{bmatrix}\tau^{u,k}_{k+1}\\ \tau^{\phi,k}_{k+1}\end{bmatrix} = \left(\begin{bmatrix}A_k & B^1_k\\ B^2_k & C_k\end{bmatrix}\begin{bmatrix}\tilde{R}^{u,k}_{k+1} & 0\\ 0 & \tilde{R}^{\phi,k}_{k+1}\end{bmatrix}\begin{bmatrix}U^{k+1}\\ \Phi^{k+1}\end{bmatrix} - \begin{bmatrix}G^k\\ F^k\end{bmatrix}\right) -$$

$$\begin{bmatrix}\tilde{R}^{u,k}_{k+1} & 0\\ 0 & \tilde{R}^{\phi,k}_{k+1}\end{bmatrix}\left(\begin{bmatrix}A_{k+1} & B^1_{k+1}\\ B^2_{k+1} & C_{k+1}\end{bmatrix}\begin{bmatrix}U^{k+1}\\ \Phi^{k+1}\end{bmatrix} - \begin{bmatrix}G^{k+1}\\ F^{k+1}\end{bmatrix}\right) \quad \textbf{(5-99)}$$

From which the relative truncation errors on the interior and edges of element S_0 can easily be extracted.

5.6.1.4 Conclusions

Although the adaptive multigrid algorithm, described in the previous sections, is quite satisfactory (cf. Chapter 7) from the practical point of view, it still has room for improvement. One obvious problem can be exemplified by observing that when the FMG scheme is used for solving a regular problem on a sequence of grids, the work invested on the coarse grids is usually smaller then the work invested at the finest grids, despite the fact that the coarse grids are relaxed many more times. This is because of the fact that the coarse grids contain much less elements than the fine grids. For the FMG scheme it can then be proven that the total amount of work needed to reduce the error to below the truncation error is proportional to the total number of elements in the computational domain. However, when local levels of refinement are used, with finer levels covering much smaller subdomains, the number of elements on coarser levels is not necessarily small in comparison. The usual FMG scheme which makes a cycle (V,W,F) through all the coarser grids per additional level of refinement, no matter how small the refined domain is, may easily end up in investing work which is no longer proportional to the total number of elements. Especially, for semiconductor problems, where the patches are strongly localized to the junctions, this is the case (cf. Chapter 7). Although the total amount of work is less in comparison to the case that local refinement is used, it clearly defeats the basic principle of the multigrid method which is to ensure the linear relationship between the total number of elements and the amount of work needed to reduce the algebraic error to below the truncation error. Although from the practical point of view one can be perfectly satisfied with the adaptive algorithm as presented in the previous sections, from the theoretical point of view we know that in the case of strongly localized patches we can do better.

A rather intuitive remedy to this problem is not to treat the patches simultaneously, as in the previous sections, but rather to start a localized FMG-cycle on each individual grid patch of a certain grid level. In this case the localized cycle does not reach grid levels coarser than the level on which it was started, which alleviates the problem. In fact this idea can be applied recursively and basically a nested FMG algorithm is obtained. Note that a localized FMG-cycle returns to the patch on which it was triggered with either a converged solution or an improved solution, which is then used in the FMG-cycle that triggered the localized cycle. An advantage of this type of algorithm is that the grid patches of a certain grid level can be processed in parallel, which is advantageous on multi-processor computers. In this thesis we have not yet fully investigated this type of algorithm[10], and therefore leave it for future research.

5.7 Concluding Remarks

In this chapter we have discussed an adaptive multigrid solution method which can be used to solve the discrete problems obtained from the mixed discretization method discussed in Chapter 4. We thoroughly discussed the Vanka relaxation operator for which we calculated its smoothing factor by means of local mode analysis. It was concluded that the Vanka relaxation is an effective error smoother. We also discussed the intergrid operators, for which we concluded that the natural intergrid operators suggested by the mixed discretization are sufficiently accurate, except for the prolongation operator for the potentials. An improvement for this prolongation operator was proposed based on a post-processing technique. Also the concept of adaptive composite grids was thoroughly discussed and in particular the multigrid error estimators in the context of the mixed discretization were derived.

[10] As far as we know this type of multigrid algorithm has not yet been investigated in the literature.

Library design is language design,

Bell Labs proverb

and vice versa.

A.R. Koenig

Chapter 6

Issues Related to an Object-Oriented Implementation Strategy

In the previous chapters, the emphasis was mainly on the algorithms and methods used. In this chapter the focus is on some of the more interesting basic implementation issues. During the past 20 years, a strong tradition in the "coding" of the finite element method, based on the procedural programming paradigm, has evolved[1]. For example [Becker 1981][Carey 1983][Segal 1984a][Hughes 1987]. Languages support this paradigm by providing facilities for passing arguments to functions and returning values from functions, for example: Fortran 77, Algol 68, Pascal and C. A consequence of this is that the focus is on processing, that is, the algorithms needed to perform the desired computations. In procedural languages, no specific tools are offered with respect to managing complexity. Especially, when one has to deal with intricate problems, this is a severe limitation. Various suggestions to improve the quality of software have been proposed over the past ten years. Perhaps the most powerful one, and certainly the most popular one, is the Object-Oriented Programming paradigm (OOP) with as examples, the programming languages SmallTalk and C++ [Bourne 1992][Stroustrup 1991][Lippman 1991][Barkakati 1991]. In the past four years the application of OOP has received a lot of attention in the scientific literature. However, to date OOP has mainly been a university research vehicle and has not been widely accepted as an industrial programming standard.

In this chapter, we depart from the "classical" view of finite element coding by putting the entire scheme in the realm of OOP. In particular, we investigate an efficient implementation of the low-level FEM features using OOP. However, due to space limitations, we only describe some of the more interesting low-level features. As is well known, OOP may significantly enhance software development speed, ease of maintenance, reliability and reusability, subjects which are also of major importance in the coding of a new generation of finite element programs, which are of a highly dynamic nature. However, using the OOP technique requires a way of thinking about the problem

[1] For a prototype of a "classical" finite element program the reader may consult [Hughes 1987].

to be implemented which substantially differs from the one used in more traditional programming strategies.

The structure of this chapter is as follows. In Section 6.1, we discuss some of the essential ideas of Object-Oriented Programming. In Section 6.2, we apply OOP to the design of object-oriented finite element data structures. Finally, in Section 6.3, some concluding remarks are stated.

6.1 The Object-Oriented Programming Method

This section briefly recapitulates some of the essential ideas of OOP. The focus is on the C++ language, however, any language providing the essential OOP features can be used. For a clear introduction to the OOP philosophy and terminology we refer to [Stroustrup 1991][Lippman 1991][Barkakati 1991].

6.1.1 Objects

The major difference between OOP and the more traditional procedural oriented programming techniques is that in object-oriented design, a program conceptually consists of a collection of interacting *objects*, each encapsulating data known as the object's *members*, and functions known as the object's *methods*. The methods are used to manipulate the members. Part of the object's methods can be made publically available, forming a well-defined interface to the object. A publically available method is invoked by sending a *message* to the object, hence, an object that needs a service from another object may achieve this by sending the appropriate message to the other object. This technique is commonly referred to as the *client-server* model. At each time instant, the *state* of an object is determined by the current actual values of the object's members. Moreover, publically available methods can be provided to alter the state of the object. In this sense, the messages the object responds to, drive the objects apparent behavior. More loosely, an object can be characterized as a device capable of performing predefined actions, such as storing and retrieving information, performing work, or providing access to another object.

It goes without saying that the above point of view has a major impact on program organization and that the application of this technique is in most cases a non-trivial exercise.

6.1.2 Classes

Objects are created from object templates called *classes* in C++, that is, each object is an instance of a class. A class is in fact a user-defined abstract data type and is used to group related data and methods. In C++ these are called the class members and we may distinguish *member data* (attributes) and *member functions* (methods). Moreover, since a class encapsulates its class members, it can be used to set access privileges for the class members, restricting the access to only those member functions which are designed to interface to the class. These interface member functions may, for instance, interrogate the state of the object or ask the object to perform a certain action. Effectively, the interface member functions are invoked through the message mechanism. From the preceding, it should be clear that in OOP the class and not the function is the primary unit of system organization.

6.1.3 Class Hierarchies

Note that the above features, with a little discipline, are also conceivable in ordinary programming languages. However, the language becomes object oriented when such concepts as *inheritance* and *polymorphism* are supported. The inheritance mechanism supports one of the most powerful intellectual tools for managing complexity, which is hierarchical ordering. Hence, we have the ability to organize related concepts into a tree structure with the most general concept at the root of the tree (base class). In C++ this is effected by allowing a class to inherit features of one or more base classes, and it is usually called a class hierarchy tree. Note that the inheritance mechanism supports the re-use of code since newly to be added classes can inherit from already existing classes.

6.1.4 Polymorphism

The polymorphism feature is even more important in OOP, because this allows objects of different type to activate different methods when sent the same message. The idea is that the object itself binds the received message to one of the methods it encapsulates. We can distinguish between polymorphism using static or dynamic binding. The difference is that with static binding, the method to bind to the received message is known at compile time, in contrast to dynamic binding where the method is determined at run time using a virtual function table (C++). The dynamic binding feature truly supports object-oriented programming, because each object can be made self supporting at run time. This allows us to define various objects that represent the same concept, however, with some slight differences in behavior. Since all objects represent a similar concept, the messages to invoke the methods can be made the same. Sending messages to these objects to perform actions is completely transparent, in the sense that each object itself knows how to respond to the message received. Hence, low-level complexity can effectively be hidden. Later on in this chapter, we will show that by using this technique we can achieve transparency with respect to the space dimension of a problem to be solved, that is, at a certain level of abstraction it does not matter if we are solving a 1D, 2D or 3D problem.

Note that there is a run-time penalty on the excessive use of polymorphism in combination with dynamic binding. This is because each method bound to a received message must be invoked through a function call and cannot be expanded inline by the compiler. Hence, dynamic binding always involves a speed penalty with respect to static binding. Only in the case where the function call overhead is negligible, in comparison with the work done by the function, can the speed penalty be neglected. However, it is the authors experience that the dynamic binding mechanism implemented through a virtual function table significantly outperforms the alternative implementation of polymorphism by using a series of case statements.

6.1.5 Object Hierarchies

In OOP there is yet another way to express hierarchic relationship, called an object hierarchy. Here we speak of a *member* relationship. Opposed to inheritance, which is a *"is a"* relationship, membership is a *"has a"* relationship. A simple guideline in the use of inheritance or membership is to investigate whether the concept one wants to model is similar to a previously modeled concept, or whether it only needs an instance of a previously modeled concept.

The member relationship can be enforced by three methods: inclusion, reference and pointer. When using inclusion, an instance of a class is created within the object representing the concept to be modeled. In other words an object contains another object. The other two methods are very similar, with the difference that the object is not actually included, only a pointer or a reference to the object is stored. Note that in this case the object pointed to must be explicitly initialized, opposed to the inclusion mechanism where the included object is automatically initialized by calling the default initializer for that object. The use of a reference or a pointer is very similar, the difference being that a reference to an object needs to be initialized whereas a pointer may be left uninitialized.

The above technique to create object hierarchies is, in a way, complementary to the class hierarchy method, the difference being that a class hierarchy is always static, whereas an object tree using pointers to other objects can be of a dynamic nature. For instance, if the object pointed to is the root of a class hierarchy with each of the leaves modeling a similar concept, we may, at run time, bind the object pointer to each of the leaves of the class hierarchy. The polymorphism and virtual function mechanism can then be used to identify the object pointed to at run time. To give an example, suppose we have designed a class modeling a generic semiconductor, which, amongst other things, contains a pointer to a generic class modeling the doping configuration. In order to be able to choose between several different doping configurations (at run time) we make the class *doping* the root of a class hierarchy where each of the leaves models a different doping configuration. This allows us to bind a specific doping configuration to the class semiconductor at run time[2]. The polymorphism and virtual function mechanism ensure the proper handling of the messages send to the class *doping*.

6.2 Object-Oriented Finite Element Data Structures

In the terminology set out in Chapter 2, a model of a certain problem was assumed to consist of a computational domain Σ, a set of model equations in the form of a PDE, a set of boundary conditions and (if necessary) a set of interface conditions. It seems logical to take an object of type *Model* as the root of the object hierarchy. The data encapsulated by the class *Model* is the data base representing the model in its discrete form.

The basic messages we can send to an object of type *Model* are: (1) initialize, (2) solve and (3) show. When sending the message "initialize" to an empty instance of an object of type *Model*, it takes a data file containing the specification of the model, parses this file and initializes the data base accordingly, that is, it builds the required object hierarchy. Similarly, sending the message "solve" causes the object of type *Model* to determine the solution of the problem it models. In a similar fashion, the message "show" causes the object to display its solution. Note that when sending a message to an object, the object itself knows how to take care of this message. In fact, this takes place by sending messages to other objects at a lower level in the object hierarchy. Again each of the objects knows locally how to deal with the message it receives.

In this section, we are mainly concerned with that part of the class *Model* which deals with the object-oriented representation of the computational domain. Effectively this comes down to the definition of a suitable object hierarchy. Moreover, once the object hierarchy is defined, the representation of the model equations, boundary and interface conditions is more or less self-evident. Since it is impossible to list all details, only the general principles are outlined below. For the actual implementation details (C++) we refer to the software distribution (cf. Chapter 1).

6.2.1 Representation of a Computational Domain

The basic class for this purpose is the class *Structure*. Starting from this class, the question is, of course, how we can set up the appropriate class and/or object hierarchy. In doing so, we may follow the general principle of the FEM, however, in addition, we must also take into consideration the use of mixed-hybrid finite elements and the finite element based multigrid method as discussed in Chapters 4 and 5. For instance, mixed-hybrid finite elements need to have degrees of freedom defined in the interior and on the edges of the elements, and the multigrid method needs sophisticated data structures to

[2] This means we do not have to hard-code the doping model into the semiconductor model.

support locally refined grids and efficient grid-scanning algorithms. These features pose additional constraints on the design and implementation of the class and object hierarchy.

In Chapter 2, Section 2.2, we argued that a convenient strategy towards the modeling of an actual transducer configuration is to divide the computational domain into K disjoint simply connected[3] subdomains $\overline{\Omega}^k$ with $(k = 1, \ldots, K)$, separated from each other by internal interfaces, and from the outside world by boundary interfaces (cf. Figure 6-1). Note that the interfaces between subdomains are defined by

$$I_{kl}^I = \overline{\Omega}^k \cap \overline{\Omega}^l \tag{6-1a}$$

and boundaries by

$$I_k^B = \overline{\Omega}^k \cap (U \backslash \Sigma) \tag{6-1b}$$

where U is the one-, two-, or three-dimensional universe and $\Sigma \subset U$ represents the space occupied by the transducer configuration. A convenient strat-

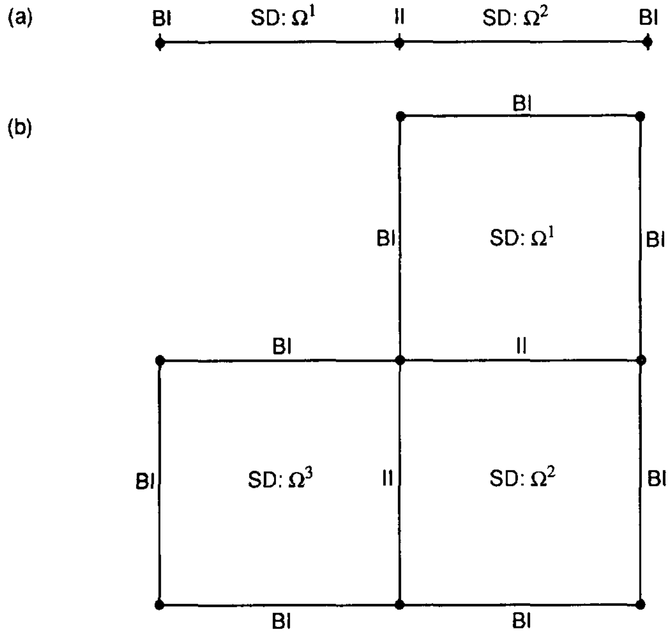

Figure 6-1 Two simple examples of a computational domain divided into a number of subdomains, separated from each other by internal interfaces (II) and from the outside world by boundary interfaces (BI): (a) a 1D configuration, (b) a 2D configuration.

[3] Note that we require a subdomain to be simply connected, however, we may always construct a non simply connected domain by patching together simply connected subdomains.

egy to define the class *Structure* is to introduce two additional classes, that is, the class *Domain* and the class *Interface*. Now a class *Domain* can uniquely be defined by means of an ordered list of pointers to objects of type *Interface*. To be more precise, the class interface acts as the root of two derived classes, which are the class *InternalInterface* and the class *BoundaryInterface*. We may use polymorphism to distinguish between objects of type *InternalInterface* and type *BoundaryInterface* at run time.

In order to define the problem to be solved on a domain, each domain also contains a pointer to an object of type *Problem*. In its turn, the class *Problem* is the root of a class hierarchy defining various types of problems. In general, the class *Problem* should, for example, provide methods to calculate material parameters, set up the local element matrix and the right-hand side vector, or in the case of multigrid, perform a local (Vanka) iteration. Note that this setup allows a different model to be defined on each domain and that we may bind a model to a domain at run time. For instance, referring to Figure 6-1, we might want to solve just the Poisson equation on subdomain Ω^1, and the full semiconductor equations on subdomains Ω^2 and Ω^3.

Similarly, the classes *BoundaryInterface* and *InternalInterface* should provide methods for calculating the boundary conditions and (if applicable) the interface conditions. One way to provide flexibility in this respect is not to hard code the methods for calculating the boundary and interface conditions, but rather to use pointers to user-definable functions. This is a standard procedure, well known to C and C++ programmers, so for details we refer to [Barkakati 1991]. The UNIX (SVR4) operating system provides facilities to compile these functions at run time and link them into the program by using its dynamic linking features.

In order not to be burdened by too many implementation details, we impose the restrictive condition that each domain can be represented by a geometrically "simple" shape, that is, a straight line for 1D models, a convex polygonal surface for 2D models, or a polyhedronal volume for 3D models[4]. Effectively, this means that the boundary interfaces and internal interfaces of the domain are piecewise "flat". For 1D domains the interfaces are points in 1D space, for 2D domains the interfaces can be constructed as lists of straight line segments in 2D space, and for 3D domains the interfaces can be constructed from "flat" polygonal surfaces in 3D space. In its turn, a polygonal surface in 3D can be represented by a list of straight line segments in 3D. Finally, any straight line segment can be represented by the two nodes it connects.

[4] In the case of 2D and 3D models this a rather restrictive approximation, however, most planar silicon structures can adequately be modeled in this way. Moreover, the possibility of curved geometrical objects can be added at a later stage.

So basically, at the lowest level of the object hierarchy we need objects of type *Node* located in 1D, 2D or 3D space. For this purpose we define a class hierarchy with a generic class *Node* at the top of the tree. The 1D, 2D, 3D variants of a node are derived from the root of the tree by inheritance. These derived classes are called *NodeR1*, *NodeR2* and *NodeR3*. An object of type *Line* can then be constructed by using pointers to the objects of type *Node* it connects. Thus again, we define a class hierarchy with a generic class *Line* at the top of the tree, the 1D, 2D and 3D variants are derived from the root by inheritance and are called *LineR1*, *LineR2* and *LineR3*. By using the inheritance mechanism it is also relatively easy to later on incorporate curved line types in 2D and 3D space. Note that by definition the direction of a line is from the first referenced node to the second referenced node.

Similarly, an object of type *Surface* can be constructed by (signed) pointers to an ordered set of objects of type *Line*. Note that the sign of a pointer must be such that it corrects the orientation of a line, when the original orientation of the line is not in correspondence with the ordering of the lines defining the surface. The orientation of a surface is always taken according to the right-hand rule when traversing the boundary of the surface in the direction specified by the ordering of the lines defining the boundary. Note that we can have surfaces in two- and tree-dimensional space, the class *Surface* therefore again is the root of a class hierarchy with classes *SurfaceR2* and *SurfaceR3*. Note that we may later on add the possibility of curved surfaces in three-dimensional space.

Finally, an object of type *Volume* can be constructed by (signed) pointers to a set of objects of type *Surface* which enclose the volume. Again, note that the sign of the pointers is necessary to define the orientation of the surfaces with respect to the volume, here we have the rule that the orientation of each surface multiplied by the sign of the pointer must be directed outwards from the volume.

Perhaps a last remark concerning the role of the various geometrical objects defined above is in place. In the case of one-dimensional models, the basic geometrical objects are the *NodeR1* and the *LineR1*. Here the object *LineR1* represents the domain and the object *NodeR1* represents an internal interface or a boundary interface. In the case of a two-dimensional model the basic geometrical objects are the *NodeR2*, the *LineR2* and the *SurfaceR2*. Here the object *SurfaceR2* represents a domain and the object *LineR2* represents an internal interface or a boundary interface. Finally, for three-dimensional models the basic geometrical objects are the *NodeR3*, the *LineR3*, the *SurfaceR3* and the *VolumeR3*. Here the object *VolumeR3* represents a domain and the object *SurfaceR3* represents an internal interface or a boundary interface.

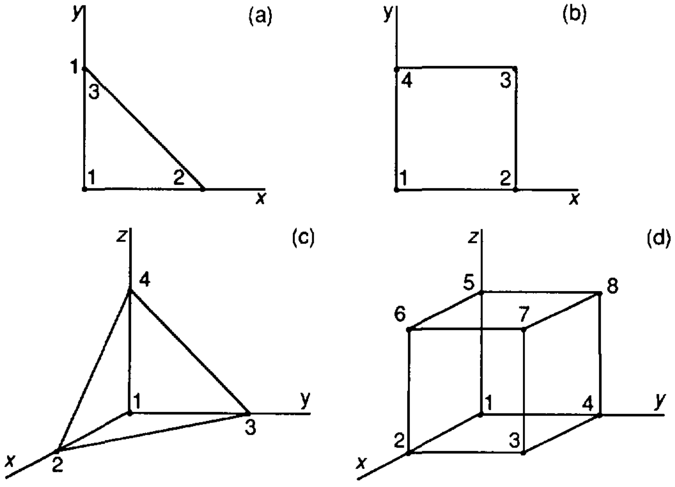

Figure 6-2 Basic 2D and 3D finite element shapes: (a) triangle, (b) quadrilateral, (c) tetrahedron and (d) hexahedron.

6.2.2 Representation of the Coarse Grid

The coarse grid is defined by a rather crude quasi-regular partitioning of the domains Ω^k into elements. For this purpose, a generic class *Element* is introduced, which is discussed in the following section. For each domain, the coarse grid elements are stored in a linear element list which is pointed to by the object of type *Domain*. For 1D domains the elements are straight line segments. For 2D and 3D domains, the topology of the elements are restricted to those shown in Figure 6-2. Note that Figure 6-2 shows the reference elements, meaning that all vertices of the element are taken at unit coordinates. Elements with distorted[5] geometry (not topology) can be obtained by means of an affine 2D or 3D transformation (cf. Appendix C). This way the domains and interfaces are constructed by patching together simple geometrical objects.

6.2.3 Representation of an Element

Not surprisingly, the finite element is taken as the basic processing unit. The general procedure is to loop through the list of elements and process each element according to its specific properties. As discussed in the previous section, in 1D models the element is a straight line segment called an edge, in

[5] The distortion is not entirely free, for instance, we should avoid triangles with obtuse angles or quadrilaterals which are too weirdly shaped.

2D models it is a (flat) surface called a face and in 3D models it is a volume called a body. Therefore, the class *Element* is taken as the root of a tree of derived classes, which are the classes *EdgeR1*, *FaceR2* and *BodyR3*, and polymorphism is used to determine the space dimension of the element. The elements are indicated by the shaded entries in Table 6-1. In a similar fashion as the definition of the domains, the classes *EdgeR1*, *FaceR2* and *BodyR3* can also be defined by means of pointers to a few primitive geometrical objects as indicated in Table 6-1. This table indicates, for each space dimension of the model, the object hierarchy when reading from right to left. For example, in the case of a 3D domain, an object of type *BodyR3* is defined by pointers to a set of objects of type *FaceR3*. In their turn the objects of type *FaceR3* are defined by pointers to an ordered set of objects of type *EdgeR3*. The objects of type *EdgeR3* are in their turn defined by pointers to objects of type *NodeR3*. Similar arguments hold for the 2D and 1D case.

Now, in order to enable the use of geometrically different types of elements for 2D and 3D models[6], the classes *FaceR2* and *BodyR3* are taken as the roots of class hierarchies as indicated in Table 6-2. Again polymorphism is used to distinguish between the various types of elements at run time. An example of the resulting object hierarchy for a tetrahedron is given in Figure 6-3. Similar object hierarchies can be drawn for the other 2D and 3D element types.

Table 6-1 Table indicating the hierarchy of geometrical objects for a 1D, 2D and 3D domain.

space	node	edge	face	body
R1	NodeR1	EdgeR1		
R2	NodeR2	EdgeR2	FaceR2	
R3	NodeR3	EdgeR3	FaceR3	BodyR3

Table 6-2 Table indicating the different types of 2D and 3D elements.

FaceR2	QuadrilateralR2	TriangleR2	
BodyR3	HexahedronR3	WedgeR3	TetrahedronR3

6.2.4 Grid Refinement and Unrefinement

Obviously, the spatial discretization of the domain, as outlined in the previous section, is usually too crude to be very accurate. Therefore, each element may on its turn be partitioned into a number of congruent sub-elements, that is, a triangle into four congruent sub-triangles, a quadrilateral into four con-

[6] Obviously, in the 1D case only one type of element is possible, which is an edge.

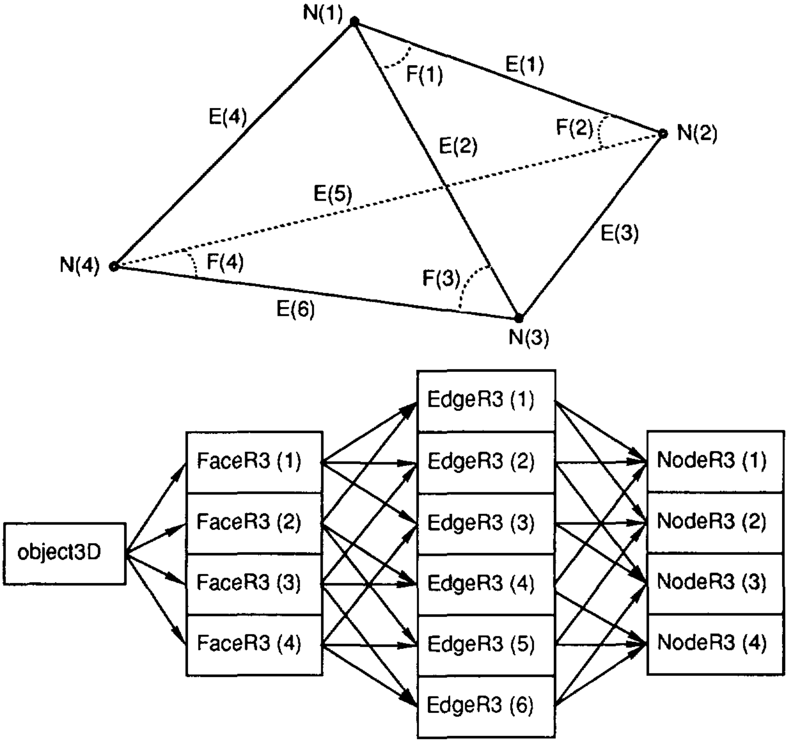

Figure 6-3 Abstract representation of a tetrahedron.

gruent sub-quadrilaterals, a tetrahedron into eight congruent sub-tetrahedrons, a wedge into eight congruent sub-wedges and a hexahedron into eight congruent sub-hexahedrons[7]. This process is called basic refinement. The additional elements created upon refining an element are called the child elements and the element generating the children is called a parent. The subdivision of elements can (recursively) be repeated and in fact generates a sequence of nested grids G^l with $(l = 0, \ldots, L-1)$ indicating the level of refinement.

This feature can easily be implemented by adding pointers to objects of type *Element* to the classes *EdgeR1*, *FaceR2* and *BodyR3*. In the 1D case we obviously need two pointers, in the 2D case we need four pointers and in the 3D case we need eight pointers. The entire grid may then be organized in a multiple-rooted tree data structure (forest), where each root represents a domain, the nodes of the tree represent the elements and the edges of the tree represent the child-parent relations. The leaves of the tree at a certain level

[7] Higher order partitionings are also conceivable, for instance, the second order regular partitioning of a triangle generates nine congruent subtriangles.

represent the unrefined elements of that level. In the case of 1D grids the tree is a binary tree, in 2D grids it is a quad tree, and in 3D grids it is an octal tree. Note that arranging the grid levels this way allows easy implementation of the intergrid transfer operators, because for each element the child-parent relation is known an we may easily locally switch between a coarse-grid and a fine-grid element.

Note that the process of grid refinement and unrefinement is a local process and can be dealt with at the element level. Thus in order to refine the grid at grid level *l*, each element at that level is sent the message “Refine”, upon receiving this message the element itself knows how to create and initialize its child elements. Similarly, upon receiving the message “Unrefine" the element destroys its child elements. It is perhaps superfluous, but also in this case we use polymorphism to properly handle the “Refine” and “Unrefine” messages for the various types of elements at run time. Note that in order to refine a grid at level *l*, we scan the grid elements at level *l*, however, in order to unrefine the grid at level *l*, the grid is scanned at level *l*-1.

It is not necessary to refine each element of a specific grid level. In order to decide whether it needs refinement, the element first evaluates a local error criterium (cf. Chapter 5). In other words, an element only needs to refine itself if the approximation to the “true” solution on that element is “not good enough”. Similarly, if the local error criterium is below a certain threshold, the element may decide to unrefine itself. This process is called adaptive refinement and unrefinement. For this purpose we may introduce the messages “AdaptiveRefine” and “AdaptiveUnrefine”. Just as in the previous case, the element itself knows how to deal with these messages. For instance, if we send the message “AdaptiveRefine” or “Refine” to an element which is already refined, it simply ignores the message. However, if the message “AdaptiveUnrefine” or “Unrefine” is sent to an element, it simply destroyes its children, which then succesively destroy their children and so on.

We thus obtain locally refined grids and the set of elements created by refinement of level *l* can be split into a number of disjunct patches to the grid at level *l*. The union of all the patches constitutes the grid at level *l*+1. To keep track of the patches at a certain grid level, we introduce the class *GridPatch* which holds a pointer to the first element of a disjoint patch to the grid at level *l*-1, and the number of elements in the patch. Instances of the class *GridPatch* themselves are managed by yet another class, the *GridPatchList* class. This class is implemented as a doubly linked list of objects of type *GridPatch*. Instances of the class *GridPatchList* are in their turn organized by a class *GridPatchArray*, which, in fact, is itself a doubly linked list of objects of type *GridPatchList*. A method to maintain this data structure is discussd in the following section.

In order for this scheme to work, we need to store the topology of the grid at each level. This means that each object of type *Element* needs to be able to store pointers to the neighbors located at its nodes, edges, or faces, depending on whether the element is embedded in 1D, 2D or 3D space. Since for each patch, a pointer to an element of the patch and the topology of the patch is available, it can be scanned by a scanning algorithm based on the topology of the grid (cf. Section 6.2.5). Note that in this sense the coarse grid may be viewed as a single patch to the union of the domains.

Note that each grid patch may have a boundary which not necessarily coincides with the true internal and boundary interfaces of the domain. However, when scanning a patch by using its topology, the patch boundary can easily be detected because at this artificial boundary the elements do not have a neighbor.

6.2.5 Grid Scanning

In this section, we discuss an algorithm for scanning the elements in quasi-structured grids. For 1D grids the scan is trivial, however, for 2D and 3D grids the situation is a lot more difficult. The easiest approach is to use an hierarchical scan. In this method, we process each of the elements in the coarsest grid recursively. This type of scan could be used to carry out the intergrid transfer operators, however, for the relaxation operator this type of scan is not very suitable for reasons of convergence. A better method, which can be used on quasi structured grids of the type described in the previous section, is described below.

This type of scanning strongly resembles the evolution of a *wave front* in space-time. To make this type of scan possible, the topology of each level of the grid must be available. The algorithm operates as follows. To scan the grid at a certain level, or better, to scan a certain patch, it is first given a seed element[8]. The pointers to the seed elements are stored in the patch lists. The seed element is then pushed on to an element processing queue (FIFO)[9] and a local element flag is set to mark the element as being on the queue. The next step is to pop the next-to-be-processed element from the queu and set a local element flag to mark the element as being processed. A message is sent to this element to push, in the order determined by the local numbering of the edges[10], those pointers to it neighbors which are not already processed or not already on the queue. Obviously, when an edge has no neighbor, which means that it is a boundary edge, nothing needs to be pushed with respect to this edge. We may then send a message to the element to process itself according to some internal method, e.g. relaxation, refinement, prolongation, etc. Upon finishing this method, the next element is popped from the queue

[8] The seed is not restricted to a single element, in 2D we may also give the elements lying at a boundary as a seed.

[9] The fact that we use a queue is important here, if we use a stack instead, the algorithm behaves in a totally different way.

[10] For the 1D and 3D case we should read "nodes" and "faces".

and the same procedure is repeated. Thus a wave front of processed elements cells travels through the domain until there are no more elements left to process. For uniform rectangular grids, this manner of scanning the grid is identical to a forward or backward diagonal ordering of the elements as described in [Wesseling 1992]. Note that when we replace the queue in the scanning algorithm with a stack, the scanning pattern is identical to forward lexicographical ordering of the elements as described in [Wesseling 1992].

In fact, the grid at a certain level needs to be scanned many times. So, in order to speed up this process, only the first scan of a grid at a certain level is scanned by using the above method. During the first scan, a scan list is build which is implemented as a doubly linked list. The doubly linked list enables fast forward and backward traversing of the elements in a certain grid. The link fields are implemented locally, that is, in the element itself and therefore we do not have to maintain an additional data structure. Moreover, for a 2D grid each link field needs at most two bits to indicate which neighbor is next in the list. Remember, the pointer to the neighbor is already available in the data structure, hence we only need to indicate which one to follow.

The entire scheme is implemented in a class *GridScan*. In order to scan the grid at a certain level, for whatever purpose, e.g. relaxation, interpolation, we just instantiate an object of type *GridScan* with the appropriate level number and grid data structure (tree). We then send messages to this object to request the next element. Effectively, this class hides the implementation details, because outside the object we do not know wether the scan is made by using the wavefront method or by traversing the scan list.

6.2.6 Patch Identification

The same scanning algorithm can be used when scanning the grid at level l for the purpose of grid refinement. However, note that in an adaptive grid refinement setting, we need to identify newly created grid patches. To achieve this we use the following method. We start by scanning each patch of the previously locally refined grid level l by traversing the previously built scan list. Note that the patches to be scanned belong to level l, however, the newly to be identified patches belong to level l+1. Each element scanned this way is checked if it is already flagged as visited. If this is not the case, we check whether it has children and if it does we need to check if its children are candidates for a new grid patch at level l+1. In order to do so, we apply the wave front scanning method with the given element as a seed.

Hence, the seed element is pushed on to the element processing queue (FIFO) and a local element flag is set to mark the element as being on the queue. The next step is to pop the next-to-be-processed element from the queue and set a local element flag to mark the element as visited. A message is sent to this element to examine its neighbors as to whether they should be pushed on to the queue. Only those neighbors which have children, and have not already been visited or on queue are pushed on to the queue. Neighbors which do not have children and have not yet been flagged as visited are flagged as visited. Then the next element is popped from the queue and the same procedure is repeated until the queue is empty. Thus all elements that have children and are connected to the seed element are marked as visited. Now a new object of type *GridPatch* can be created. Note that the stored topology ensures that we can find all elements by using the seed element stored in the object of type *GridPatch.*

We then proceed to the next element and repeat the procedure. In this way we detect all seed elements for the grid patches in time proportional to the number of elements in grid level *l*.

6.2.7 Allocation of Objects

As we have argued in the previous sections, the OOP approach to the coding of the FEM significantly enhances software development speed, ease of software maintenance, reliability and reusability. Its application to technical software has increased significantly over the past few years. However, there is also a serious drawback with respect to OOP. As pointed out earlier, a FEM program in the context of OOP is a collection of objects encapsulating their associated data and methods, and communicating with each other through messages. Although each model is represented in terms of the same set of objects, the number of each type of object can be different for each model. Moreover, if we consider the local refinement or unrefinement of the grid, we observe that it is highly desirable to be able to create and delete objects on the fly. Or to put it a little differently, the data structures are required to be highly dynamic. For instance, the OOP features of C++ provide the "new" and "delete" operators for this purpose. Clearly, the use of these operators is very convenient because memory is only consumed when an object is created and is immediately available for reuse when the object is destroyed. However, as a fact of experience, in the case of a large number of small objects, as in our case, the standard "new" and "delete" operators of C++ provide too much overhead. The reason for this is that the entire operating system machinery must be put in motion for each object to be created or destroyed.

Most of the existing FEM programs written in Fortran-77 manage the free store by means of user-written routines. Here the free store is a large block of memory defined at compile time which cannot be changed at run time. Although this technique is not very flexible, it does allow the memory management routines to be optimized with respect to the application. A similar approach could be used in C by using the memory management facilities of the underlying operating system (UNIX). In this way we may allocate a large block of memory at run-time, and again manage the block of memory by user-written routines. Clearly, this enhances speed, however, usually, it is difficult to a priori estimate the amount of memory the program needs and, hence we have to use some ad hoc size, which clearly can be a waste. What we actually need is a mechanism offering a compromise between speed and the amount of memory wasted.

To achieve this, we make use of the fact that we only have a limited number of different objects of which we may take advantage to speed up the memory allocation process. In short, this means that the first time an object of a certain type is created, a memory pool (heap) capable of containing a certain number of this type of object is created. Each next time an object is created, it is put in to its corresponding memory pool. Since the memory in the pool was previously allocated, this process can be very fast. As soon as the pool is filled, a new pool is created which is linked to the previously created pool. Hence, the memory pool is effectively resized to twice the basic pool size. Note that such a pool is maintained for each type of object created. An important parameter is the choice of the base size of each pool. Obviously, the larger the pool size the faster the overall memory allocation can take place. However, choosing the pool size too large is a waste of memory. Also there exists a relationship between the number of allocated objects of each type. For instance, in a 2D grid consisting of N triangles, the approximate number of edges and nodes resp. is $3/2N$ and $1/2N$ respectively. Obviously, the relative basic pool sizes of the corresponding objects should be chosen accordingly.

In C++ terminology, the above is achieved by overloading the “new” and “delete” operators. The implementation of equivalent operators using the C operators “calloc” and “free” is straightforward. In principle, the equivalent operator must check as to whether a memory pool already exists for the requested type of object. If not, a memory pool is created by means of the “calloc” operator. The memory pool can be managed by a pool header with pointers to the bottom, top and next free entry of the pool. The entries which become available after deletion of objects can be managed by adding them to a (singly) linked list which is pointed to from the pool header. As soon as the pool is filled, a second one is created and is added to the pool header in a similar fashion. As soon as a pool is empty it is destroyed[11].

6.2.8 Representation of the Global Solution

Some remarks on the representation of the local discrete solution are in order at this point. As we have already pointed out, the approximate solution on an element is defined by an expansion in terms of the element (local) basis functions. Depending on how the basis functions are defined, the expansion

[11] Obviously, there should be some hysteresis in this process, because the number of objects needed might approximately be equal to the memory pool size. This will cause excessive pool creation and deletion. A 10 % hysteresis seems to do well.

coefficients can be associated with element vertices, the element edges, the element faces or the element body. For this purpose, it is best to define an object of type *DofArray* (degree of freedom) which can be pointed to from the appropriate object. To give an example, in a 1D element, the element is an object of type *EdgeR1* with two pointers to objects of type *NodeR1*. Now, the expansion coefficients defined on the element edge are stored in to an object of type *DofArray* which is pointed to from the object *EdgeR1*. The expansion coefficients defined at each of the two nodes of the edge are also stored in an object of type *DofArray*, however, these objects are pointed to from the objects of type *NodeR1*.

6.3 Concluding Remarks

The main purpose of this chapter was to present some guidelines in implementing the FEM, featuring mixed hybrid finite elements and the multigrid method, by means of an object-oriented programming language (OOP). A brief introduction to OOP was presented and it was argued that the use of the OOP features of the C++ languages can conveniently be used to construct an elegant implementation of the FEM. It is the author's experience that the effort needed to write a similar program in a traditional language such as Fortran-77 is significantly greater. The OOP paradigm was subsequently applied to the problem of the design of an object-oriented representation of the computational domain, the grid levels, the elements and efficient grid scanning algorithms. We emphasize that the approach forces strict locality, meaning that no global storage is used. In other words solutions, residuals and stiffness matrices are all stored element wise and are accessed by sending messages to objects of type element. Obviously, there are many more FEM features which lend themselves to an object-oriented implementation, however, for the sake of brevity these have to be omitted.

So eine Arbeit wird eigentlich nie fertig, mann muss sie für fertig erklären, wenn man nach Zeit und Umständen das Möglichste getan hat.

J.W. Goethe

Chapter 7

A Glimpse of the Results

In Chapters 4, 5 and 6, we have discussed a general framework for the numerical modeling of systems of (elliptic) partial differential equations in mixed form. In this chapter we wish to show a glimpse of the results, in particular we show that by using the proposed framework, that is, a combination of mixed finite element discretization and multigrid acceleration, the expected h-independent convergence can indeed be realized for the thermoelectric problem. A typical problem of moderate apparent complexity that can be used for this purpose is the 2D quarter diode test problem. Although this problem is rather specific, we emphasize that the method in principle also is applicable to more intricate semiconductor problems.

The structure of this chapter is as follows. In Section 7.1, the quarter diode test problem is discussed. In Section 7.2, we discuss the general mixed formulation of the thermoelectric problem. Next, in Section 7.3, the discrete mixed variational form of the thermoelectric problem is discussed. A strategy for solving the coarse grid problem is discussed in Section 7.4. In Section 7.5 some convergence results are discussed. Finally, in Section 7.6 some final concluding remark are stated.

7.1 Quarter Diode Test-Problem

In this section we discuss a simple test configuration dealing with the self-heating effects in a forward biased diode. For simplicity we restrict ourselves to the two-dimensional problem. Moreover, for reasons of symmetry the configuration is reduced to the one shown in Figure 7-1. For the material properties we assume $\varepsilon = 1.036\times10^{-12}$, $\mu^e = 700$, $\mu^h = 700$, $P^e = 0.5\times10^{-3}$, $P^h = 0.5\times10^{-3}$ and $\kappa_t = 1.5$. For the generalized SRH model we assume that the trap level is at the midpoint of the band gap and that $\tau^e = \tau^h = 10\times10^{-6}$.

With respect to the thermoelectric powers we should remark that the above values are not very realistic because if we examine equations **(3-50a)** and **(3-50b)** we observe that in P-type material P^e is rather large whereas P^h is very small. In N-type material the opposite is true. However, for the test problem we are considering, the thermoelectric powers do not have a significant influence on the final results, so we do not use the more realistic model for the thermoelectric powers. The use of more realistic models for the material parameters is not expected to have much effect on the convergence results.

The doping profile is modeled as an abrupt junction

$$n^a(x) = \begin{cases} 1\times10^{18}, & \|x\| < 0.5\times10^{-3} \\ 0, & \|x\| \geq 0.5\times10^{-3} \end{cases} \qquad \textbf{(7-1a)}$$

$$n^d(x) = \begin{cases} 0, & \|x\| \leq 0.5\times10^{-3} \\ 1\times10^{18}, & \|x\| > 0.5\times10^{-3} \end{cases} \qquad \textbf{(7-1b)}$$

From the physical point of view this type of doping profile is not very realistic, however, from the numerical point of view it is quite a challenge. We can safely state that once the method is capable of solving the problem with abrupt junctions it is also capable of solving the problem with more realistic doping profiles stated in terms of Gaussian distribution functions.

For the electrical Dirichlett boundary conditions we take $E_f^n = E_f^p = -V_a$ and $E_f^n = E_f^p = 0$, respectively at the top and bottom contact, where V_a is the applied bias voltage. At both contacts the corresponding electrostatic potential is, for each bias condition, obtained by (iteratively) solving φ from the zero space charge condition: $\xi(\varphi, E_f^n, E_f^p, T) = 0$. This value serves as the Dirichlett boundary condition for the Poisson equation. At the non-contacted parts of the boundary homogeneous electrical Neumann boundary conditions $d_\alpha \cdot n_\alpha = j_\alpha^e \cdot n_\alpha = j_\alpha^h \cdot n_\alpha = 0$ are assumed. For the thermal Dirichlett boundary conditions we take $T = T_a = 300$ at the bottom contact. At the remaining part of the boundary (including the top contact) we take homogeneous thermal Neumann boundary conditions $j_\alpha^h \cdot n_\alpha = 0$.

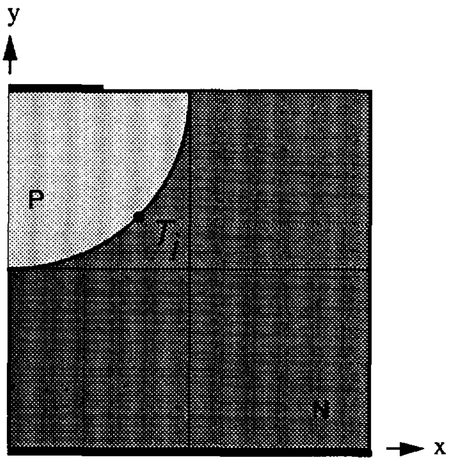

Figure 7-1 Configuration for calculating the self-heating effect of a forward biased diode.

For the above problem we calculate the temperature T_j at the midpoint of the junction as a function of the bias voltage (cf. Figure 7-1). Note, however, that in the present context we are more interested in the convergence behavior of the solution method than the actual solution.

7.2 Thermoelectric Problem in Mixed Form

The only step that separates us from the application of the mixed discretization method, as discussed in Chapter 4, is the reformulation of the thermoelectric model to the mixed form. For this purpose, we take the scaled model equations, discussed in Section 3.5.1, as the starting point.

Clearly, for the thermoelectric problem we have four sets of potentials and fluxes; a set corresponding to the Poisson equation, a set corresponding to the electron continuity equation, a set corresponding to the hole continuity equation, and a set corresponding to the heat balance equation. In principle, the derivation of the corresponding mixed formulation is straightforward, however, we make the following simplifying assumption

$$\kappa_t \gg q n^e \mu^e (P^e)^2 T \qquad \kappa_t \gg q n^h \mu^h (P^h)^2 T \tag{7-2}$$

which is valid for not too high carrier densities (cf. Section 3.4.2.1.2). The advantage of the above assumption is that the inverse of the transport matrix (cf. equation (3-44)) has a simple and very appealing form

$$\Gamma^{-1} = \begin{bmatrix} \left(-\frac{1}{\mu^e n^e}\right) & \left(-\frac{P^e P^h T}{\kappa_t}\right) & \left(\frac{P^e}{\kappa_t}\right) \\ \left(\frac{P^e P^h T}{\kappa_t}\right) & \left(\frac{1}{\mu^h n^h}\right) & \left(-\frac{P^h}{\kappa_t}\right) \\ \left(\frac{P^e T}{\kappa_t}\right) & \left(\frac{P^h T}{\kappa_t}\right) & \left(-\frac{1}{\kappa_t}\right) \end{bmatrix} \tag{7-3}$$

which makes it very easy to add exponential fitting characteristics (upwinding) to the discretization by means of the inverse type of averaging of the matrix entries $1/n^e$ and $1/n^h$ [Brezzi 1989a]. In the case the assumption in equation (7-2) is violated the inverse of the transport matrix is more complex, however, in that case the entire model needs to be revised in order to also include heavy doping effects.

Using the above result it is not very difficult to cast the entire thermoelectric model in (scaled) mixed form, that is

$$\begin{bmatrix} A & B \\ B^T & C \end{bmatrix} \begin{bmatrix} \Lambda_u \\ \Lambda_\phi \end{bmatrix} = \begin{bmatrix} 0 \\ F \end{bmatrix} \tag{7-4}$$

with the vector of unknown potentials and fluxes respectively given by

$$\Lambda_\phi = \left[\varphi\; E_f^e\; E_f^h\; T\right]^T \qquad \Lambda_u = \left[d_\alpha\; n_\alpha^e\; n_\alpha^h\; h_\alpha\right]^T \tag{7-5}$$

and the vector of source terms given by

$$F = \left[\xi\; \dot{n}^e\; \dot{n}^h\; \dot{h}\right]^T \tag{7-6}$$

The matrices A, B and C are respectively given by

$$A = \begin{bmatrix} \frac{1}{\varepsilon} & 0 & 0 & 0 \\ 0 & (\frac{1}{n^e\mu^e}) & \left(\frac{P^eP^hT}{\kappa_t}\right) & \left(\frac{-P^e}{\kappa_t}\right) \\ 0 & \left(-\frac{P^eP^hT}{\kappa_t}\right) & (-\frac{1}{n^h\mu^h}) & \left(\frac{P^h}{\kappa_t}\right) \\ 0 & \left(-\frac{TP^e}{\kappa_t}\right) & \left(-\frac{TP^h}{\kappa_t}\right) & (\frac{1}{\kappa_t}) \end{bmatrix} \tag{7-7}$$

and

$$B = \begin{bmatrix} \partial_\alpha & 0 & 0 & 0 \\ 0 & \partial_\alpha & 0 & 0 \\ 0 & 0 & \partial_\alpha & 0 \\ 0 & 0 & 0 & \partial_\alpha \end{bmatrix} \tag{7-8}$$

and

$$C = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & (\dot{n}^e + n_\alpha^e\partial_\alpha) & -(\dot{n}^h + n_\alpha^h\partial_\alpha) & 0 \end{bmatrix} \tag{7-9}$$

Note that in the case $P^h = P^e = 0$ the submatrix A reduces to a diagonal matrix. Clearly in this case the coupling between the electrical and thermal domain is eliminated.

7.3 Mixed Discretization of the Thermoelectric Problem

By using the methods described in Chapter 4, the reformulation of the above system of equations into a discrete variational statement is straightforward. The result is given by

$$\begin{bmatrix} \begin{bmatrix} A_{\varphi\varphi} & 0 & 0 & 0 \\ 0 & A_{ee} & A_{eh} & A_{nT} \\ 0 & A_{he} & A_{hh} & A_{hT} \\ 0 & A_{Tn} & A_{Tp} & A_{TT} \end{bmatrix} & \begin{bmatrix} B_{\varphi\varphi} & 0 & 0 & 0 \\ 0 & B_{ee} & 0 & 0 \\ 0 & 0 & B_{hh} & 0 \\ 0 & 0 & 0 & B_{TT} \end{bmatrix} \\ \begin{bmatrix} B^T_{\varphi\varphi} & 0 & 0 & 0 \\ 0 & B^T_{ee} & 0 & 0 \\ 0 & 0 & B^T_{hh} & 0 \\ 0 & 0 & 0 & B^T_{TT} \end{bmatrix} & \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & C_{Te} & C_{Th} & 0 \end{bmatrix} \end{bmatrix} \begin{bmatrix} \begin{bmatrix} U_\varphi \\ U_e \\ U_h \\ U_T \end{bmatrix} \\ \begin{bmatrix} \Phi_\varphi \\ \Phi_e \\ \Phi_h \\ \Phi_T \end{bmatrix} \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} G_\varphi \\ G_e \\ G_h \\ G_T \end{bmatrix} \\ \begin{bmatrix} F_\varphi \\ F_e \\ F_h \\ F_T \end{bmatrix} \end{bmatrix} \qquad \text{(7-10)}$$

where U_k and Φ_k respectively represent the vector of fluxes and the vector of potentials associated with degrees of freedom $k = \{\varphi, E^e_f, E^h_f, T\}$. The correspondence between equations (7-4) and (7-10) is evident. Each of the submatrices A_{kl}, B_{kl}, C_{kl}, G_k and F_k can be calculated according to the method described in Chapter 4. However, special care must be taken in the evaluation of the submatrices A_{nn} and A_{pp}.

At this stage we wish to emphasize that the implicit use of Fermi statistics in the thermoelectric model poses some additional problems, which can be identified as: (a) the efficient evaluation of the Fermi-Dirac integrals, (b) the upwind characteristics of the discretization, and (c) the treatment of the nonlinearity of the model equations. An elegant solution to problem (a) can be found in Appendix D. However, problems (b) and (c) remain to be discussed.

7.3.1 Exponential Fitting and Upwinding

In principle the set of equations (cf. equation (7-4)), apart from the non-linearity, is well behaved in the sense that a straightforward application of the mixed discretization method leads to a stable discretization. No problems are to be expected with respect to advection dominated flows, which need upwind discretization to produce stable results. However, we do face the problem that the coefficients of the PDE vary strongly with respect to the mesh size. This results in a significant loss in accuracy which can only be resolved by choosing the mesh-size extremely small. In order to achieve reasonable accuracy without the need for excessively fine meshes we may resort to the exponential fitting technique. Note that exponential fitting in a way is equivalent to upwinding [Polak 1988]. Provided a special quadrature rule is used the mixed discretization method introduces a natural upwind effect because of the inverse averaging of the matrix entries $1/n^e$ and $1/n^h$ in equation (7-7) [Brezzi 1989a]. Without getting into too much details we can state that the inverse type of averaging introduces an artificial diffusion length of the order of the mesh size h. Moreover, if the mesh size is sufficiently related to the spatial variation in the potential-like degrees of freedom, the effect of the artificial diffusion is negligible. Hence the upwind effect adapts itself to the spatial variations in the potential-like degrees of freedom.

7.3.2 Non-Linearity

For the sake of simplicity we only linearize with respect to the principal non-linearity of the model which reflects itself through the explicit dependence of the matrix elements of equations (7-7) and (7-9) upon the quantities n^e, n^h and T. For simplicity we do not linearize with respect to the non-linearity of the material parameters. Note that in the present setting the use of weakly non-linear material parameters is not expected to pose any problems, however, the results might be suboptimal with respect to the convergence properties of the non-linear iteration.

In principle we may follow two basic strategies. In the first one we do not decouple the equations and apply a straightforward Newton linearization to the entire system of equations. In principle this results in quadratic convergence when the initial guess is sufficiently close to the true solution. However, the Newton method can be rather unpredictable when used to solve the coarse-grid problem. A more robust approach is to first decouple the system of equations (cf. equation (7-10)). Various decoupling strategies come into mind and at this stage it is rather unclear which one is the best possible choice. However, if we assume that the thermoelectric powers are small which implies that the coupling between the electrical and thermal domain is rather weak, equation (7-7) approximates a diagonal matrix. This suggests a straightforward Gauss-Seidel type decoupling method where we repeatedly solve the following sequence of non-linear systems

$$\begin{bmatrix} \begin{bmatrix} A_{\varphi\varphi} \end{bmatrix} & \begin{bmatrix} B_{\varphi\varphi} \end{bmatrix} \\ \begin{bmatrix} B^T_{\varphi\varphi} \end{bmatrix} & \begin{bmatrix} 0 \end{bmatrix} \end{bmatrix} \begin{bmatrix} \begin{bmatrix} U_{\varphi} \end{bmatrix} \\ \begin{bmatrix} \Phi_{\varphi} \end{bmatrix} \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} G_{\varphi} \end{bmatrix} \\ \begin{bmatrix} F_{\varphi} \end{bmatrix} \end{bmatrix} \qquad \text{(7-11a)}$$

$$\begin{bmatrix} \begin{bmatrix} A_{ee} \end{bmatrix} & \begin{bmatrix} B_{ee} \end{bmatrix} \\ \begin{bmatrix} B^T_{ee} \end{bmatrix} & \begin{bmatrix} 0 \end{bmatrix} \end{bmatrix} \begin{bmatrix} \begin{bmatrix} U_e \end{bmatrix} \\ \begin{bmatrix} \Phi_e \end{bmatrix} \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} G_e - A_{eh}U_h - A_{eT}U_T \end{bmatrix} \\ \begin{bmatrix} F_e \end{bmatrix} \end{bmatrix} \qquad \text{(7-11b)}$$

$$\begin{bmatrix} [A_{hh}] & [B_{hh}] \\ [B_{hh}^T] & [0] \end{bmatrix} \begin{bmatrix} [U_h] \\ [\Phi_h] \end{bmatrix} = \begin{bmatrix} [G_h - A_{he}U_e - A_{hT}U_T] \\ [F_h] \end{bmatrix} \qquad \textbf{(7-11c)}$$

$$\begin{bmatrix} [A_{TT}] & [B_{TT}] \\ [B_{TT}^T] & [0] \end{bmatrix} \begin{bmatrix} [U_T] \\ [\Phi_T] \end{bmatrix} = \begin{bmatrix} [G_T - A_{Te}U_e - A_{Th}U_h] \\ [F_T - C_{Te}\Phi_e - C_{Th}\Phi_h] \end{bmatrix} \qquad \textbf{(7-11d)}$$

Each decoupled system is linearized by means of the Newton method and subsequently solved by iteration. The solution of each solved system is used to update the matrix entries and the right-hand-side of the next-to-be solved system. The procedure is repeated until we have convergence.

7.4 Solution Method

Before we can apply the FMG algorithm it is necessary that we first obtain an accurate solution to the non-linear coarse-grid problem. For this purpose we use the Newton iteration method (cf. Section 4.5) in combination with a continuation method. The continuation method is used to enhance the robustness of the iteration method. Basically, this means that we gradually step up from a simple problem which is easy to calculate, to the final problem by means of incrementing a continuation parameter, thereby using the result of the previous continuation step as an initial guess to the current continuation step. Note that in this respect the time dependent problem as discussed in Section 4.4 can be regarded as an almost perfect continuation method, where the time discretization index acts as a continuation parameter. For the thermoelectric problem the following coarse grid solution method is used. This method can either be used to solve the time independent thermoelectric problem, or to calculate the initial condition on the coarse grid needed in the time dependent thermoelectric problem.

7.4.1 Coarse-Grid Problem

The following continuation strategy for the coarse grid problem has proven to be effective. We start by setting the Dirichlett as well as the Neumann boundary conditions to zero. This means that at the electrical Dirichlett boundaries the electrostatic potential is set to the built-in potential and the Fermi-levels are set to zero. At the thermal Dirichlett boundaries the temperature is set to a reference temperature (e.g. the ambient temperature T_a). At the electrical Neumann boundaries the normal components of the dielectric flux density, the electron and hole fluxes are set to zero. At the thermal Neumann boundaries the normal component of the heat flux is set to zero. This action forces the device into the state of thermodynamic equilibrium, hence, all thermodynamic potentials (E_f^e, E_f^h, T) are constant throughout the device. This means that we have zero electrical and thermal currents, and as a result only the non-linear Poisson equation needs to be solved to obtain the electrostatic potential at thermodynamic equilibrium.

According to singular perturbation theory [Markowich 1983][Markowich 1990] the built-in potential φ_{bi} can be used as a reasonable approximation to the actual solution of the Poisson equation provided the spatial variation in the doping and band gap is not too strong. We therefore use the built-in electrostatic potential φ_{bi} as an initial guess for solving the non-linear Poisson equation. The built-in potential is found from the zero space charge condition. In the case Boltzmann statistics and completely ionized shallow localized states are assumed the built-in potential is found to be[1]

$$\varphi_{bi}^0 = \left(\frac{kT}{q}\right) \ln\left[\frac{1}{2}\left(\left(\frac{n^d - n^a}{N^{cb}}\right) e^{\frac{E^{gp}}{2kT}} + \sqrt{\left(\frac{n^d - n^a}{N^{cb}}\right)^2 e^{\frac{E^{gp}}{kT}} + 4\left(\frac{N^{vb}}{N^{cb}}\right)}\right)\right] \quad \textbf{(7-12)}$$

Note that for uniform doping the above built-in potential is indeed the trivial solution of the Poisson equation. So only close to a junction this approximation will not hold. For the (full) thermoelectric model using Fermi-Dirac statistics, and assuming incompletely ionized shallow localized states, the above defined built-in potential φ_{bi}^0 does not hold. In principle, we can use equation **(7-12)** as an initial guess to the Poisson equation, however, it is cheaper to first calculate a better approximation to the built-in potential. Since in this case an analytical expression is not available an iterative method must be used. We use a pointwise Newton method with equation **(7-12)** as the initial guess, to solve the zero space charge condition

$$\xi(\varphi_{bi}) = q\left[n^h(\varphi_{bi}, T_a) - n^e(\varphi_{bi}, T_a) + n^{d^+}(\varphi_{bi}, T_a) - n^{a^-}(\varphi_{bi}, T_a)\right] = 0 \quad \textbf{(7-13)}$$

for φ_{bi} at the barycenter of each element[2]. To obtain double-precision accuracy $(\delta \cong 2.2\times10^{12})$ approximately 16 iterations are necessary at $T_a = 50$, approximately 4 at $T_a = 77$ and at room temperature and above the Newton algorithm converges immediately.

We then have available the thermodynamic equilibrium solution on the coarse grid. Next, the Dirichlett and Neumann boundary conditions are set to their desired values and the corresponding globally linearized problem is formulated with the thermodynamic equilibrium solution as the initial guess. We solve for the state corresponding to the actual boundary conditions by starting a global iteration where each iteration step uses the result of the pre-

[1] For the numerical evaluation of this expression consult Section 3.5.2.

[2] We also need to solve it at the electrical Dirichlett boundaries in order to find the correct boundary condition for the electrostatic potential.

vious step as the initial guess. If the iteration fails to converge within 10 steps, it is aborted and the algorithm starts an iteration for the simpler problem where the (actual) boundary conditions are multiplied by a factor $\alpha=1/2$. In general, each time the iteration fails to converge the multiplication factor α for the boundary conditions is halved and a new iteration is started. As soon as we have a convergent iteration the boundary conditions are set to their desired values, however, the last multiplication factor α is stored and if the iteration fails to converge the boundary conditions are multiplied by ($\alpha + 1/2(1-\alpha)$). This process is repeated until we have convergence for the desired boundary conditions.

The global linearization of the system of equations uses a combination of the Gauss-Seidel decoupling method (cf. Section 7.3) and the Newton linearization method. Basically, we first solve the global non-linear (mixed) Poisson equation by means of a Newton iteration where each iteration step is solved by the repeated application of (linear) Vanka relaxation sweeps (cf. Section 5.4). The result is then used to initialize a Newton iteration on the electron and hole continuity equations, where again each Newton step is solved by repeated application of Vanka relaxation sweeps. These results, on their turn, are used to initialize a Newton iteration on the heat balance equation. The process is repeated until a prescribed accuracy is achieved. Note that this iteration process is global and bears no immediate resemblance to the non-linear relaxation method discussed in Section 5.4.7.

7.4.2 Multigrid Iteration

In order to reduce the algebraic error well below the truncation error on the finest grid we use a Full Multigrid V-cycle as shown in Figure 5-2(d). In the FMG-V cycle we use $\nu_1 = \nu_2 = 1$ which means that before each restriction, and after each interpolation (prolongation), a single symmetric non-linear Vanka relaxation sweep is applied. Moreover, each coarse-grid problem (except the first one) is approximately solved by applying $\nu_0 = 2$ non-linear Vanka relaxation sweeps. Upon completion of the FMG-V cycle we continue by applying F-cycles (cf. Figure 5-2(c)) until the scaled residuals on the finest grid are sufficiently small. The residuals are scaled by the corresponding diagonal entry of the Jacobian of the differential operator, hence the scaled residuals approximately represent the corrections necessary to make the fine grid solution exact.

For the thermoelectric problem the non-linear Vanka-type relaxation results, for each element, in a small (20x20) non-linear system of equations. As in the calculation of the coarse-grid problem we use a Gauss-Seidel decoupling method in combination with Newton iteration to solve this system of equations. Note however, that each decoupled equation now is only linearized with respect to the local unknown degrees of freedom. Clearly, this type of linearization is local and not global as in the calculation of the coarse-grid solution.

7.5 Convergence Results

In this section we analyze the convergence behavior of the solution procedure as described in the previous sections. To demonstrate the mesh-size independent performance of the multigrid method for the test problem proposed in Section 7.1 we solve the problem three times, the first time with a finest grid with 16x16 elements (four grid levels), the second time with a finest grid with 32x32 (five grid levels) elements and the third time with a finest grid with 64x64 elements (six grid levels). In each case the coarsest grid only consists of 2x2 elements. We calculate the (scaled) logarithm of the max-norm of the residuals on the finest grid after completion of the FMG-V cycle and after completion of each F-cycle. In Figure 7-2 the residuals are shown for the electrostatic potential, the electron and hole Fermi-levels and the temperature. Note that the residuals are scaled by the corresponding diagonal entry of the Jacobian of the differential operator, hence the scaled residuals approximately represent the corrections necessary to make the fine-grid solution exact.

For all three cases the behavior of the residuals is very similar (the resolution of the graph is too small to show the differences). This clearly demonstrates the mesh-size (*h*) independent convergence rate of the method. Consequently, because the number of cycles needed is independent of the finest grid size and the amount of work involved in carrying out a single F-cycle is of $O(n)$ (cf. Section 5.2.8), the total amount of work needed to solve the problem increases linear with the number of cells in the finest grid. Note that in Figure

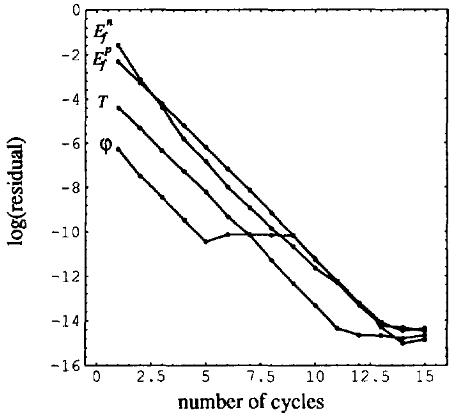

Figure 7-2 Plot of the logarithm of the (scaled) residuals corresponding to the electrostatic potential, the Fermi-levels and the temperature, as a function of the number of cycles.

7-2 we continue to apply F-cycles until machine accuracy is achieved, however, in practice we may settle for much less accuracy, e.g. 10^{-5}. In this case we already have convergence after approximately four additional F-cycles.

Finally, (as promised) we calculate the temperature T_j at the midpoint of the junction for several bias conditions (cf. Table 7-1). For this case we use nine grid levels in order to assure sufficient accuracy in the neighborhood of the junction. However, in order to keep the total number of cells limited we now use the adaptive grid refinement criterium, as discussed in Chapter 5, which uses the fine-to-coarse grid defect correction of the electrostatic potential to locally refine the grid. Basically, before each prolongation is carried out in the FMG-V cycle, the fine-to-coarse grid defect correction is calculated and each cell for which the defect correction is above the threshold value (0.1) is locally refined. This way we obtain the locally refined grid as shown in Figure 7-3.

Table 7-1 Temperature at the midpoint of the junction for several values of the bias voltage.

V_a	0.0	0.2	0.4	0.6	0.7	0.8	0.9	1.0
T_j	300	300	300	300	305	320	350	405

7.6 Concluding Remarks

The major conclusion of this chapter is that the multigrid method in combination with the mixed discretization method in principle is a very effective method for solving the thermoelectic problem in semiconductor materials, in

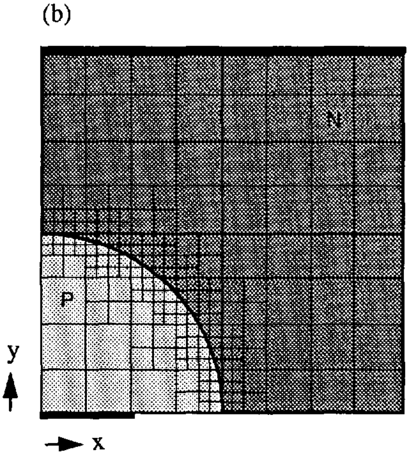

Figure 7-3 Quarter diode test problem with locally refined grid.

a sense that mesh-size independent convergence rates are obtained. So potentially we have at our disposal a method that solves these problems optimally efficient. This clearly is demonstrated by calculating the self-heating effects for the quarter diode problem on a simple computational domain with square grid cells.

It goes without saying that there are many more features that should be studied before the proposed method can withstand the demands that arise in practical computations. This is inherent to the use of the multigrid method which needs fine tuning for each type of problem to be solved. Therefore the characteristic features of each type of problem must be extracted and tested for their effect on the efficiency. In this respect we have shown the basic applicability for the thermoelectric problem, however, we have not yet addressed problems such as interface and surface charge, inhomogeneous Neumann boundary conditions, small Neumann contacts that cannot be resolved by the coarsest grid, non-linear Dirichlett boundary conditions, inversion layers, low and high temperature behavior, etc. Future research obviously needs to deal with these cases in a structured approach in order to obtain a simulator which besides being efficient is also generally applicable.

Although the proposed method is optimally efficient, in a sense that the amount of work needed to solve the problem is $C \cdot n$, we have not discussed the size of the proportionality factor C. Clearly, we should pay sufficient attention to the optimization of C. We can think of improving the local non-linear Vanka relaxation which at present uses a Gauss-Seidel decoupling method which was chosen for its robustness and not for its good convergence rate. It is very likely that the multigrid iteration tracks the actual solution sufficiently close so that we can apply a straightforward Newton linearization and hence obtain quadratic convergence in solving the local non-linear equations. Also the effectivity of the relaxation process could be improved by using different relaxation ordering schemes. It is well known that a red-black ordering of the grid cells shows much better smoothing factors. However, on quasi-unstructured grids this type of relaxation is hard to realize and it seems that on these types of grids the grid scanning algorithms as presented in Chapter 6 are optimal.

In principle the method also works on quasi-unstructured domains with triangular and parallelogramic elements. However, for these types of domains it is hard to employ relatively coarse grids. The solution to this problem must be found in the inherent use of grid adaptivity with respect to the irregular internal interfaces and boundaries. This way we may still use relatively coarse grids which not necessarily accurately approximate the computational domain. During the actual solution process the coarse grid is refined in such a way that it more and more approximates the true computational domain. It is also possible to use a coordinate transformation which transforms the computational domain of a problem to a square computational domain. This way the actual solver can be kept very simple, however, at this time it is unclear if this approach is truly beneficial.

Appendix A

Cartesian Tensor Notation

This appendix summarizes the most important issues related to Cartesian tensor notation. First, a few general remarks are in place. The main advantage of (Cartesian) tensor notation is its ease of algebraic manipulation. However, the pure algebraic nature of tensor notation deprives us of the introduction and use of high-level (symbolic) operators that could be treated as mathematical objects. In respect to the implementation, one could say that tensor notation is close to the procedural programming paradigm (cf. Chapter 6), in the sense that the notation leads directly to a computer algorithm. However, in an object-oriented environment, we rather wish to identify a well-chosen set of high-level mathematical operators which are then treated as objects. The mathematical object then may hide their procedural characteristics. If these objects are available as programming tools, a more compact high-level formulation of the problem is possible.

A-1 The Summation Convention

Repeated Greek subscripts imply summation, for example

$$a_\alpha b_\alpha \rightarrow \sum_\alpha a_\alpha b_\alpha \qquad \textbf{(A-1)}$$

A-2 The Kronecker Delta Tensor

The definition of the Kronecker delta is as follows

$$\delta_{\alpha\beta} = \begin{cases} 1, i = j \\ 0, i \neq j \end{cases} \qquad \textbf{(A-2)}$$

A-3 The Levi-Civita Tensor

The definition of the Levi-Civita tensor of rank N is as follows

$$\epsilon_{\alpha_1,\ldots,\alpha_N} = \begin{cases} +1, \text{even permutation of the subscripts} \\ -1, \text{odd permutation of the subscripts} \\ 0, \text{repeated subscripts} \end{cases} \qquad \textbf{(A-3)}$$

The relation between the Kronecker delta and Levi-Civita tensor is given by the relation

$$\epsilon_{\alpha\beta\gamma}\,\epsilon_{\alpha\eta\mu} = \delta_{\beta\eta}\delta_{\gamma\mu} - \delta_{\beta\mu}\delta_{\lambda\eta} \tag{A-4}$$

A-4 Determinant of a Tensor of Rank Two

The determinant of a tensor of rank two is defined as

$$\mathrm{Det}(a_{\alpha\beta})\,\epsilon_{\alpha_1....\alpha_N} = a_{\alpha_1\beta_1}\cdots a_{\alpha_N\beta_N}\,\epsilon_{\beta_1....\beta_N} \tag{A-5}$$

where N is the dimension of the tensor $a_{\alpha\beta}$. In two dimensional space $N = 2$, and in three dimensional space $N = 3$.

A-5 Symmetric and Anti-Symmetric Tensors

It is usually convenient to split a tensor of rank two into its symmetric and anti-symmetric part, that is

$$\sigma_{\alpha\beta} = \sigma^s_{\alpha\beta} + \sigma^a_{\alpha\beta} \tag{A-6}$$

where

$$\begin{aligned} \sigma^s_{\alpha\beta} &= \frac{1}{2}(\sigma_{\alpha\beta} + \sigma_{\beta\alpha}) \\ \sigma^a_{\alpha\beta} &= \frac{1}{2}(\sigma_{\alpha\beta} - \sigma_{\beta\alpha}) \end{aligned} \tag{A-7}$$

A further decomposition can be obtained by extracting the trace from the symmetric part of the tensor, that is

$$\sigma_{\alpha\beta} = \sigma^t_{\alpha\beta} + \sigma^{s,\,0}_{\alpha\beta} + \sigma^a_{\alpha\beta} \tag{A-8}$$

where

$$\begin{aligned} \sigma^t_{\alpha\beta} &= \left(\frac{1}{3}\sigma_{\mu\eta}\delta_{\mu\eta}\right)\delta_{\alpha\beta} \\ \sigma^{s,\,0}_{\alpha\beta} &= \frac{1}{2}(\sigma_{\alpha\beta} + \sigma_{\beta\alpha}) - \left(\frac{1}{3}\sigma_{\mu\eta}\delta_{\mu\eta}\right)\delta_{\alpha\beta} \\ \sigma^a_{\alpha\beta} &= \frac{1}{2}(\sigma_{\alpha\beta} - \sigma_{\beta\alpha}) \end{aligned} \tag{A-9}$$

From the above decomposition, it can be observed that a physical quantity, determined by the nine independent elements of a tensor of rank two, can also be characterized by a scalar equal to one third of the trace of the tensor, by five independent elements of the symmetric part (after substraction of the trace), and by the three independent elements of the antisymmetric part.

A-6 Laminar and Solenoidal Vector Fields

Solenoidal and laminar vector fields are defined by the following property

$$\partial_\alpha F^{sol}_\alpha = 0$$

$$\epsilon_{\alpha\beta\gamma}\partial_\beta F^{lam}_\gamma = 0 \tag{A-10}$$

This means that the divergence of a solenoidal vector field is zero, and the rotation of a laminar vector field is zero. Obviously, this allows us to write a solenoidal field as the rotation of a vector field and a laminar field as the gradient of a scalar field

$$F^{sol}_\alpha = \epsilon_{\alpha\beta\gamma}\partial_\beta\Phi_\gamma$$

$$F^{lam}_\alpha = \partial_\alpha\Phi \tag{A-11}$$

A-7 Cartesian Tensor Notation vs. Dyadic Tensor Notation

A table giving the correspondence between the (symbolic) dyadic notation and the Cartesian tensor notation is given below.

Table A-1 The correspondence between dyadic and Cartesian tensor notation.

description	dyadic notation	cartesian notation
scalar	a	a
vector	$\boldsymbol{a}$	a_α
dyadic	$\boldsymbol{ab}$	$a_\alpha b_\beta$
scalar product	$\boldsymbol{a}\cdot\boldsymbol{b}$	$a_\alpha b_\alpha$
cross product	$\boldsymbol{a}\times\boldsymbol{b}$	$\epsilon_{\alpha\beta\gamma}a_\beta b_\gamma$
gradient of scalar	∇a	$\partial_\alpha a$
gradient of vector	$\nabla\boldsymbol{a}$	$\partial_\alpha a_\beta$
gradient of dyadic	$\nabla\boldsymbol{ab}$	$\partial_\alpha a_\beta b_\gamma$
divergence of vector	$\nabla\cdot\boldsymbol{a}$	$\partial_\alpha a_\alpha$
divergence of dyadic	$\nabla\cdot\boldsymbol{ab}$	$\partial_\alpha a_\alpha b_\beta$
Laplacian of scalar	$\nabla^2 a$	$\partial_\alpha\partial_\alpha a$
Laplacian of vector	$\nabla^2\boldsymbol{a}$	$\partial_\alpha\partial_\beta a_\beta$
curl of vector	$\nabla\times\boldsymbol{a}$	$\epsilon_{\alpha\beta\gamma}\partial_\beta a_\gamma$

Appendix B

Functional Spaces

In this appendix, we briefly discuss the functional spaces as used in the mixed variational formulation as discussed in Chapter 4. The functional spaces, to be discussed below, are used to classify the behavior of the solution expressed by the variational form of the PDE. We assume that the PDE is defined on an open and simply connected computational domain Ω in R^d (d=1,2,3) with a smooth boundary $\partial\Omega$. The boundary consists of a part $\partial\Omega_D$, on which Dirichlett boundary conditions are prescribed, and a part $\partial\Omega_N$, on which Neumann boundary conditions are prescribed. Moreover, we assume that $\partial\Omega_D \cap \partial\Omega_N = 0$ and $\partial\Omega_D \cup \partial\Omega_N = \partial\Omega$. Further, the vector $\boldsymbol{n}$ is defined as the outward normal vector on the boundary $\partial\Omega$.

The Lebesque space $L^2(\Omega)$ of square-integrable scalar functions on Ω is defined as

$$L^2(\Omega) = \{\phi : \Omega \rightarrow \mathrm{R} \mid \int_{\Omega} \phi^2 d\Omega < \infty \} \tag{B-1}$$

The Lebesque space $\boldsymbol{L}^2(\Omega)$ of square-integrable vector functions on Ω is defined as

$$\boldsymbol{L}^2(\Omega) = \{\boldsymbol{u} : \Omega \rightarrow \mathrm{R}^d \mid \int_{\Omega} |\boldsymbol{u}|^2 d\Omega < \infty \} \tag{B-2}$$

The Sobolev space $H^1(\Omega)$ containing the square-integrable functions of which the gradients are also square integrable is defined as

$$H^1(\Omega) = \{\phi \in L^2(\Omega) \mid \nabla\phi \in \boldsymbol{L}^2(\Omega) \} \tag{B-3}$$

For vector functions a similar space is defined, however, the notation is a little different

$$H(\mathrm{div};\Omega) = \{\boldsymbol{u} \in \boldsymbol{L}^2(\Omega) \mid \nabla \cdot \boldsymbol{u} \in L^2(\Omega) \} \tag{B-4}$$

The above spaces are defined on Ω, similar Sobolev spaces can be defined on $\partial\Omega$. If the trace operator γ_D is defined to filter the values of ϕ at the Dirichlett boundary from ϕ, the following subspace can be defined

$$H^{1/2}(\partial\Omega) = \{\gamma_{\mathrm{D}}\phi \,|\, \phi \in H^1(\Omega)\} \qquad \textbf{(B-5)}$$

Similarly, a Sobolev space can be defined for the Neumann part of the boundary

$$H^{-1/2}(\partial\Omega) = \{\gamma_{\mathrm{N}}\boldsymbol{u} \,|\, \boldsymbol{u} \in H(\mathrm{div};\Omega)\} \qquad \textbf{(B-6)}$$

It will prove to be convenient to define some subspaces in the above Sobolev spaces given in Equations (B-3) to (B-6). Imposing zero boundary conditions on the solutions we get the following subspaces defined on Ω

$$H^1_{\mathrm{D}}(\Omega) = \{\phi \in H^1(\Omega) \,|\, \gamma_{\mathrm{D}}\phi = 0\} \qquad \textbf{(B-7)}$$

$$H_{\mathrm{N}}(\mathrm{div};\Omega) = \{\boldsymbol{u} \in H(\mathrm{div};\Omega) \,|\, \boldsymbol{n} \cdot \gamma_{\mathrm{N}}\boldsymbol{u} = 0\} \qquad \textbf{(B-8)}$$

and on the boundary $\partial\Omega$

$$H^{1/2}_{\mathrm{D}}(\partial\Omega) = \{\lambda \in H^{1/2}(\partial\Omega) \,|\, \lambda = 0 \text{ on } \partial\Omega_{\mathrm{D}}\} \qquad \textbf{(B-9)}$$

$$H^{-1/2}_{\mathrm{N}}(\partial\Omega) = \{\mu \in H^{-1/2}(\partial\Omega) \,|\, \mu = 0 \text{ on } \partial\Omega_{\mathrm{N}}\} \qquad \textbf{(B-10)}$$

Finally, the linear varieties, which define the solution spaces satisfying the boundary conditions can be defined as

$$H^1_*(\Omega) = \{\phi \in H^1(\Omega) \,|\, \gamma_{\mathrm{D}}\phi = g_{\mathrm{D}} \,|\, g_{\mathrm{D}} \in H^{1/2}_{\mathrm{D}}(\partial\Omega)\} \qquad \textbf{(B-11)}$$

$$H_*(\mathrm{div};\Omega) = \{\boldsymbol{u} \in H(\mathrm{div};\Omega) \,|\, \boldsymbol{n} \cdot \gamma_{\mathrm{N}}\boldsymbol{u} = g_{\mathrm{N}} \,|\, g_{\mathrm{N}} \in H^{-1/2}(\partial\Omega)\} \qquad \textbf{(B-12)}$$

This concludes the discussion on some of the basic notions of functional analysis with respect to the mixed variational formulation.

Appendix C

Affine Element Transformations

Calculations on finite elements can be substantially simplified if these are made on a reference or master element S'. To achieve this, an affine transformation $\boldsymbol{F} : S \rightarrow S'$ is used to project an element S in the mesh on to the reference element S'; then the calculations are carried out on this reference element. In this appendix, we consider such a transformation for triangles and quadrilaterals, moreover, we also address the issue of how the operators and the physical quantities on which they operate transform when going from an element in the mesh to the reference element.

C-1 Transformations for Triangles

The reference triangle (cf. Figure C-1) is defined by its vertices at (0,0) (1,0) and (0,1). An arbitrary element in the mesh can be described by the position vectors of its vertices, which we denote as: $\boldsymbol{x}^1$, $\boldsymbol{x}^2$ and $\boldsymbol{x}^3$. Note that we use the convention to specify the vertices in an anticlockwise fashion.The transformation $\boldsymbol{F(x')} \rightarrow \boldsymbol{x}$ carrying any point $\boldsymbol{x}' \in S'$ on to $\boldsymbol{x} \in S$ can be constructed by means of the following rule

$$\boldsymbol{x} = \sum_{i=1}^{3} \boldsymbol{x}^i \psi'_i(x', y') \qquad \textbf{(C-1)}$$

where the ψ'_i are the linear shape functions on the reference triangle, that is

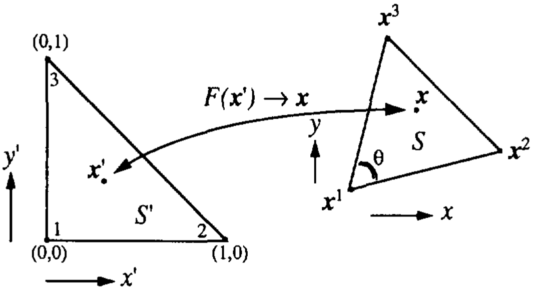

Figure C-1 An arbitrary triangular element S and its transformation to a master element S'.

$$\begin{aligned} \psi'_1(x', y') &= 1 - x' - y' \\ \psi'_2(x', y') &= x' \\ \psi'_3(x', y') &= y' \end{aligned} \qquad \textbf{(C-2)}$$

Using equations **(C-1)** and **(C-2)** we may write the mapping *F(x')* → ***x*** as

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x'(x^2 - x^1) + y'(x^3 - x^1) \\ x'(y^2 - y^1) + y'(y^3 - y^1) \end{bmatrix} + \begin{bmatrix} x^1 \\ y^1 \end{bmatrix} \qquad \textbf{(C-3)}$$

That the above transformation indeed maps the triangle in the mesh to the reference triangle can be easily verified by inspection. The functional matrix b, the metric g and the functional determinant J of the transformation are found to be

$$\begin{aligned} b_{\alpha\beta} &= \frac{\partial x_\alpha(\boldsymbol{x}')}{\partial x'_\beta} = \begin{bmatrix} x^2 - x^1 & x^3 - x^1 \\ y^2 - y^1 & y^3 - y^1 \end{bmatrix} \\ g_{\alpha\beta} &= b_{\tau\alpha} b_{\tau\beta} \\ J &= \det(b_{\alpha\beta}) = \sqrt{\det(g_{\alpha\beta})} \end{aligned} \qquad \textbf{(C-4)}$$

Using the following rules, which are valid for an arbitrary triangle in the mesh

$$\begin{aligned} l_1 &= |\boldsymbol{x}^2 - \boldsymbol{x}^1| \\ l_2 &= |\boldsymbol{x}^3 - \boldsymbol{x}^2| \\ l_3 &= |\boldsymbol{x}^1 - \boldsymbol{x}^3| \\ L &= \sqrt{l_1 l_3 \cos\theta} \end{aligned} \qquad \textbf{(C-5)}$$

the metric and the determinant can be written as

$$g_{\alpha\beta} = \begin{bmatrix} l_1^2 & L^2 \\ L^2 & l_3^2 \end{bmatrix} \qquad J = l_1 l_3 \sin\theta \qquad \textbf{(C-6)}$$

Note that using the above notations we may write the transformation as a linear map, that is

$$x_\alpha = b_{\alpha\beta} x'_\beta + b_\alpha \qquad b_\alpha = (x_1, y_1) \qquad \textbf{(C-7)}$$

Also note that when the vertices of the triangle in the mesh are numbered counterclockwise and the triangle is non-degenerate (all angles < π), the Jacobian (J) of the transformation is always positive, hence the transformation is invertible. According to equation (C-7) the inverse transformation can then be stated as

$$x'_{\alpha} = b^{-1}_{\alpha\beta} \cdot (x_{\beta} - b_{\beta}) \tag{C-8}$$

C-2 Transformations for Quadrilaterals

The reference quadrilateral (cf. Figure C-2) is defined by its vertices at (0,0) (1,0) (1,1) and (0,1). An arbitrary element in the mesh can be described by the position vectors of its vertices, which we denote as: $\boldsymbol{x}^1$, $\boldsymbol{x}^2$, $\boldsymbol{x}^3$ and $\boldsymbol{x}^4$. The transformation can be constructed by means of the following rule

$$\boldsymbol{x} = \sum_{i=1}^{4} \boldsymbol{x}^i \psi'_i(x', y') \tag{C-9}$$

where the ψ_i are the standard bilinear shape functions on a quadrilateral, that is

$$\begin{aligned} \psi'_1(x', y') &= (1-x')\,(1-y') \\ \psi'_2(x', y') &= x'\,(1-y') \\ \psi'_3(x', y') &= x'y' \\ \psi'_4(x', y') &= (1-x')\,y' \end{aligned} \tag{C-10}$$

We may then calculate the mapping $F(x') \rightarrow x$ as

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x'\,(x^2-x^1) + y'\,(x^4-x^1) + x'y'\,(x^1-x^2+x^3-x^4) \\ x'\,(y^2-y^1) + y'\,(y^4-y^1) + x'y'\,(y^1-y^2+y^3-y^4) \end{bmatrix} + \begin{bmatrix} x^1 \\ y^1 \end{bmatrix} \tag{C-11}$$

That the above transformation indeed maps the quadrilateral in the mesh to the reference square can easily verified by inspection. The functional matrix

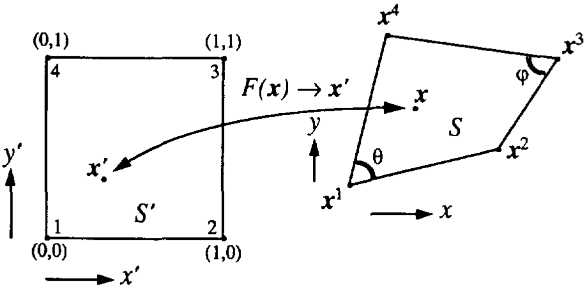

Figure C-2 An arbitrary quadrilateral element *S* and its transformation to a master element *S'*.

b, the metric g and the functional determinant J of the transformation are found to be

$$b_{\alpha\beta} = \frac{\partial x_\alpha(\boldsymbol{x}')}{\partial x'_\beta} = b^1_{\alpha\beta} + b^2_{\alpha\beta}$$

$$b^1_{\alpha\beta} = \begin{bmatrix} (x^2 - x^1) & (x^4 - x^1) \\ (y^2 - y^1) & (y^4 - y^1) \end{bmatrix}$$

$$b^2_{\alpha\beta} = \begin{bmatrix} (x^1 - x^2 + x^3 - x^4)\,y' & (x^1 - x^2 + x^3 - x^4)\,x' \\ (y^1 - y^2 + y^3 - y^4)\,y' & (x^1 - x^2 + x^3 - x^4)\,x' \end{bmatrix}$$

$$g_{\alpha\beta} = b_{\tau\alpha} b_{\tau\beta}$$

$$J = \det(b_{\alpha\beta}) = \sqrt{\det(g_{\alpha\beta})} \qquad \textbf{(C-12)}$$

When the vertices of the quadrilateral in the mesh are numbered anticlockwise and all its interior angles are less than π, the Jacobian can be proven to be positive for any $\boldsymbol{x}' \in S'$, which means that the transformation is invertible. However, stating the inverse transformation is, in general, a difficult task because of the non-linear character of the transformation.

Note that in the case of general quadrilaterals, opposed to the triangular case, the transformation cannot be written as a linear mapping. Further, the functional matrix, the metric and the functional determinant depend on $\boldsymbol{x}'$. The dependence on $\boldsymbol{x}'$ can be removed by restricting the type of quadrilaterals to either parallelograms, rectangles or squares. Now the second part of the functional matrix $b^2_{\alpha\beta}$ vanishes and as a result the transformation becomes linear. Using the following rules, which are valid for an arbitrary paralellogram in the mesh

$$\begin{aligned} l_1 &= |\boldsymbol{x}^2 - \boldsymbol{x}^1| \\ l_2 &= |\boldsymbol{x}^3 - \boldsymbol{x}^2| \\ l_3 &= |\boldsymbol{x}^4 - \boldsymbol{x}^3| \\ l_4 &= |\boldsymbol{x}^1 - \boldsymbol{x}^4| \\ L &= \sqrt{l_1 l_4 \cos\theta} \end{aligned} \qquad \textbf{(C-13)}$$

the metric and the determinant can be written as

$$g_{\alpha\beta} = \begin{bmatrix} l_1^2 & L^2 \\ L^2 & l_4^2 \end{bmatrix} \qquad J = l_1 l_4 \sin\theta \qquad \textbf{(C-14)}$$

The transformation and its inverse can now be written as linear maps just as in the triangular case

$$x_\alpha = b^1_{\alpha\beta} x'_\beta + b_\alpha \qquad b_\alpha = (x_1, y_1) \tag{C-15}$$

$$x'_\alpha = \left(b^1_{\alpha\beta}\right)^{-1} \cdot (x_\beta - b_\beta) \tag{C-16}$$

Now that the transformations are completely defined, the question remains how operators and the quantities on which they operate transform from an arbitrary element in the mesh to the reference element. In the remainder of this appendix, we assume that the quantities and operators are known on the reference element.

C-3 Transformation of Scalars, Vectors and Tensors

Following the transformation rules corresponding to the covariant and contravariant vector formalism [Weinberg 1972][Spiegel 1974] we may state the following rules (using tensor notation):

A scalar quantity ϕ transforms as

$$\phi(x) \leftrightarrow \phi'(x') \tag{C-17}$$

A vector quantity u_α transforms as

$$u_\alpha(x) \leftrightarrow J^{-1}(x')\,(b_{\alpha\beta}(x') \cdot u'_\beta(x')) \tag{C-18}$$

A tensorial quantity $c_{\alpha_1 \ldots \alpha_N}$ of rank N transforms as

$$c_{\alpha_1 \ldots \alpha_N}(x) \leftrightarrow J^{-N}(x')\,(b_{\alpha_1\beta_1} \ldots b_{\alpha_N\beta_N}(x') \cdot c'_{\beta_1 \ldots \beta_N}(x')) \tag{C-19}$$

C-4 Transformation of Differential Operators

Using the coordinate free formulations of the covariant gradient, divergence and rotation operators [Weinberg 1972][Spiegel 1974] the transformed operators can be written as

$$\partial_{x_\alpha} \phi(x) \leftrightarrow b^{-1}_{\alpha\beta}(x') \partial_{x'_\alpha}\, \phi'(x') \tag{C-20}$$

$$\partial_{x_\alpha} u_\alpha(x) \leftrightarrow J^{-1}(x') \partial_{x'_\alpha}\, u'_\alpha(x') \tag{C-21}$$

$$\in_{\alpha\beta\gamma} \partial_{x_\beta} u_\gamma \leftrightarrow J^{-1}(x)\, \in_{\alpha\beta\gamma} \partial_{x'_\beta}\, u'_\gamma(x') \tag{C-22}$$

C-5 Transformation of Integral Operators

Volume elements, surface elements and line elements in R^2 transform according to

$$\begin{aligned} dV &\leftrightarrow J(x')dV' \\ n_\alpha dS &\leftrightarrow J(x')b^{-1}_{\alpha\beta}(x')n'_\alpha \, dS' \\ \tau_\alpha dL &\leftrightarrow b_{\alpha\beta}(x')\tau'_\beta \, dL' \end{aligned} \tag{C-23}$$

Similar rules can be derived in R^3. Using the above rules, the various element integrals as occurring in the finite element formulations in Chapter 3 transform according to

$$\int_V \phi(x)dV = \int_{V'} \phi'(x')J(x')dV' \tag{C-24}$$

$$\int_V \phi(x)\,\partial_{x_\alpha} u_\alpha(x)dV = \int_{V'} \phi'(x')\;\partial_{x'_\alpha} u'_\alpha(x')dV' \tag{C-25}$$

$$\int_S \phi(x)u_\alpha(x)n_\alpha dS = \int_{S'} \phi'(x')u'_\alpha(x')n'_\alpha dS' \tag{C-26}$$

$$\int_V c_{\alpha\beta}(x)u^i_\beta(x)u^j_\alpha(x)dV = \int_{V'} J^{-1}\,(b_{\alpha\sigma}b_{\beta\tau}\hat{c}_{\alpha\beta}(x'))\,u'^i_\sigma\,(x')u'^j_\tau(x')dV' \tag{C-27}$$

Equations **(C-24)-(C-26)** are more or less self-evident, however, equation **(C-27)** requires some explanation. The tensor $\hat{c}_{\alpha\beta}(x')$ is obtained by the following rule

$$\hat{c}_{\alpha\beta}(x') = c_{\alpha\beta}(F(x')) \tag{C-28}$$

hence, this is not a usual tensor transformation but rather the transformation resulting from the transformation of the integral. We could have followed the transformation formalism more strictly, however, here there is no need to, because it is on the element in the mesh that we know $c_{\alpha\beta}(x)$, not on the reference element.

Appendix D

Numerical Evaluation of the Fermi-Dirac Integrals

In this section, we discuss a very fast and accurate method to evaluate the Fermi-Dirac integrals of arbitrary order. The proposed method is very attractive in comparison to the complicated analytical approximations that are usually used.

The Fermi-Dirac integrals are defined by

$$F_s(x) = \frac{1}{\Gamma(s+1)}\int_0^{\infty}\left[\frac{y^s}{1+\exp(y-x)}\right]dy \qquad s \in \{\ldots,-1,0,1,\ldots\} \tag{D-1}$$

We are specifically interested in the integrals $F_{1/2}(x)$ and $F_{-1/2}(x)$ which are approximated as follows

$$F_{-1/2}(x) = \begin{cases} \exp(x) & (x < -7) \\ \text{Lookup Table} & (-7 \le x < 11.5) \\ \left(\frac{4}{3}\right)\sqrt{\pi}\left[x^2 + \frac{\pi^2}{8}\right] & (x \ge 11.5) \end{cases} \tag{D-2}$$

$$F_{1/2}(x) = \begin{cases} \exp(x) & (x < -6) \\ \text{Lookup Table} & (-6 \le x < 10) \\ \left(\frac{8}{3}\sqrt{\pi}\right)x & (x \ge 10) \end{cases} \tag{D-3}$$

To construct the look-up table we make use of the following property

$$\frac{\partial F_s(x)}{\partial x} = F_{s-1}(x) \tag{D-4}$$

Using this property, the integrals can be written as

$$F_{-1/2}(x+\delta x) = F_{-1/2}(x) + F_{-3/2}(x)\delta x + 0.5F_{-5/2}(x)\delta x^2 \tag{D-5}$$

$$F_{1/2}(x+\delta x) = F_{1/2}(x) + F_{-1/2}(x)\delta x + 0.5F_{-3/2}(x)\delta x^2 \tag{D-6}$$

Now, in order to allow a fast and accurate calculation of $F_{1/2}(x)$ and $F_{-1/2}(x)$, we tabulate the values $F_{-5/2}(x_k)$, $F_{-3/2}(x_k)$, $F_{-1/2}(x_k)$ and $F_{1/2}(x_k)$ in the range of $-10 \le x \le 15$ with step size $\delta x = 0.1$. Such a table can easily be generated by the simple MATHEMATICA program shown in Figure D-1.

```
(* make_frm_table.m *)

F[s_,x_] := N[Re[Exp[x]LerchPhi[-Exp[x],s+1,1] ],12]

s[m_] := StringForm[“`` `` `` ``”, F[-5/2,m], F[-3/2,m],
                                  F[-1/2,m], F[+1/2,m]]

OpenWrite[“frmtab.dat”]

For[m=-10.0, m<=15.0, m+=0.1,
                    WriteString[“frmtab.dat”,s[m],”\n”]]

Close[“frmtab.dat”]

(* end *)
```

Figure D-1 Mathematica program to calculate the Fermi-Dirac table.

The idea is to round the value of x to the nearest tabulated x_k value and extract the corresponding values of $F_{-5/2}(x_k)$, $F_{-3/2}(x_k)$, $F_{-1/2}(x_k)$ and $F_{1/2}(x_k)$ from the table. By using the remainder $\delta x = x - x_k$ and equation **(D-5)** or **(D-6)** the values of $F_{1/2}(x)$ and $F_{-1/2}(x)$ are obtained with less than 1 % error, when the table step size $\delta x = 0.1$.

The entire scheme is implemented as a C++ class. Creating an instance of the class automatically reads the table into core and we may then send messages to the object to instruct it to calculate and return the desired value of a Fermi integral. The destruction of the object automatically frees the storage occupied by the table.

List of References

[Arnold 1985] D.N. Arnold and F. Brezzi, Mixed and Nonconforming Finite Element Methods: Implementation, Postprocessing and Error Estimates, Mathematical Modelling and Numerical Analysis, vol. 19, pp. 7-32, 1985.

[Ashcroft 1976] N.W. Ashcroft and N.D. Mermin, Solid State Physics, Saunders College, Philadelphia, 1976.

[Babuska 1978] I. Babuska and W.C. Rheinboldt, Error Estimates for Adaptive Finite Element Computations, SIAM J. Numer. Anal., vol. 15, pp. 736-754, 1978.

[Bai 1987] D. Bai and A. Brandt, Local Mesh Refinement Multilevel Techniques, Siam J. Sci. Stat. Comput., vol. 8, pp. 109-134, 1987.

[Baliga 1980] B.R. Baliga and S.V. Patankar, A new finite-element formulation for convection-diffusion problems, Numer. Heat Transfer, vol.3, pp. 393-409, 1980.

[Bank 1990] R.E. Bank, PLTMG: A Software Package for Solving Elliptic Partial Differential Equations, SIAM, Philadelphia, 1990.

[Barber 1967] H.D. Barber, Effective Mass and Intrinsic Concentration in Silicon, Solid-State Electronics, vol. 10, pp. 1039-1051, 1967.

[Barkakati 1991] N. Barkakati, Object-Oriented Programming in C++, SAMS, Carmel, 1991.

[Barrie 1987] R. Barrie and P.C.W. Holdsworth, Hopping Conductivity for Localized Electronic States, J. Phys. C: Solid State Phys., vol. 20, pp. 2219-2229, 1987.

[Barzel 1992] R. Barzel, Physically-Based Modeling for Computer Graphics: A Structured Approach, Academic Press, Boston, 1992.

[Becker 1981] E.B. Becker, G.F. Carey and J.T. Oden, Finite Elements: An Introduction (Volume I), Prentice-Hall, New Yersey, 1981.

[Bladau 1974] W. Bladau, A. Onton and W. Heinke, Temperature Dependence of the Bandgap of Silicon, Journal of Applied Physics, vol. 45, pp. 1846-1848, 1974.

[Blakemore 1962] J.S. Blakemore, Semiconductor Statistics, Pergamon Press, Oxford, 1962.

[Bourne 1992] J.R. Bourne, Object-Oriented Engineering: Building Engineering Systems Using Smalltalk-80, R.D. Irwin, Inc., and Aksen Associates, Inc., 1992.

[Brandt 1977] A. Brandt, Multi-Level Adaptive Solutions to Boundary-Value Problems, Mathematics of Computation, vol. 31, pp. 333-390, 1977.

[Brandt 1979] A. Brandt, Multi-Level Adaptive Techniques (MLAT) for Singular-Perturbation Problems, in P.W. Hemker and J.J. Miller (eds.), Numerical Analysis of Singular Perturbation Problems, Academic Press, London, 1979, pp. 53-142.

[Brandt 1984] A. Brandt, Multigrid Techniques: 1984 Guide with Applications to Fluid Dynamics, GMD, Bonn, 1984.

[Brandt 1992] A. Brandt, Multiscale Computational Methods: Research Activities, unpublished, 1992.

[Brezzi 1985] F. Brezzi, J. Douglas and L.D. Marini, Two Families of Mixed Finite Elements for Second Order Elliptic Problems, Numer. Math, vol. 47, pp. 217-235, 1985.

[Brezzi 1989a] F. Brezzi, L.D. Marini and P. Pietra, Mixed Exponential Fitting Schemes for Current Continuity Equations, in Proceedings of the Sixth International Nasecode Conference, ed. J.J.H. Miller, Dublin, Ireland, July 11-14, pp. 546-555, 1989.

[Brezzi 1989b] F. Brezzi, L. D. Marini and P. Pietra, Two-Dimensional Exponential Fitting and Applications to Drift-Diffusion Models, SIAM J. Numer. Anal, vol. 26, pp. 1342-1355, 1989.

[Brezzi 1990] F. Brezzi and K.J. Bathe, A Discourse on the Stability Conditions for Mixed Finite Element Formulations, Computer Methods in Applied Mechanics and Engineering, vol 82, pp. 27-57, 1990.

[Briggs 1987] W.L. Briggs, A Multigrid Tutorial, SIAM, Philadelphia, 1987.

[Bringer 1990] A. Bringer and G. Schön, Extended Moment Equations for Electron Transport in Semi-Conducting Submicron Structures, Journal of Applied Physics, vol. 188, pp. 19xx.

[Callen 1948] H.B. Callen, The Application of Onsager's Reciprocal Relations to Thermoelectric, Thermomagnetic and Galvanomagnetic Effects, Physical Review, vol. 73, pp. 1349-1358, 1948.

[Callen 1960] H.B. Callen, Thermodynamics, Wiley, New York, 1960.

[Carey 1983] G.F. Carey and J.T. Oden, Finite Elements: A Second Course (Volume II), Prentice-Hall, New Yersey, 1983.

[Carey 1984] G.F. Carey and J.T. Oden, Finite Elements: Computational Aspects (Volume III), Prentice-Hall, New Yersey, 1984.

[Carey 1986] G.F. Carey and J.T. Oden, Finite Elements: Fluid Mechanics (Volume VI), Prentice-Hall, New Yersey, 1986.

[Caughey 1967] D.M. Caughey and R.E. Thomas, Carrier Mobilities in Silicon Empirically Related to Doping and Field, Proc. IEEE 52, pp. 2192-2193, 1968.

[Chen-To Tai 1991] Chen-To Tai and Nenghang Fang, A Systematic Treatment of Vector Analysis, IEEE Transactions on Education, vol. 34, pp. 167-174, 1991.

[Chou 1992] So-Hsiang Chou and Qian Li, Error Estimates for Mixed Finite Element Methods for Nonlinear Parabolic Problems, Numerical Methods for Partial Differential Equations, vol. 8, pp. 395-404, 1992.

[Cristina 1987] M. Cristina and J. Squeff, Superconvergence of Mixed Finite Element Methods for Parabolic Equations, Mathematical Modelling and Numerical Analysis, vol. 21, pp. 327-352, 1987.

[Dawson 1976] T.H. Dawson, Theory and Practice of Solid Mechanics, Plenum Press, New York, 1976.

[Dawson 1990] T.H. Dawson, Theory and Practice of Solid Mechanics, Plenum Press, New York, 1990.

[Duyn 1990] D.C. van Duyn and S. Middelhoek, Information Transduction in Solid-State Transducers: A General Thermodynamic Systems Approach, Sensors and Actuators, vol. A21-A23, pp. 25-32, 1990.

[Duyn 1991] D. C. van Duyn and P. de Vries, Finite Element Modeling of Thermoelectric Materials, Proc. Materials Research Society Symposium: Modern Perspectives on Thermoelectrics and Related Materials (volume 234), May 1-2, 1991, Anaheim, CA, U.S.A.

[Duyn 1992] D.C. van Duyn and P.J.A. Munter, Finite Element Modeling of Thermoelectric Materials and Devices, Sensors and Actuators A, vol. 32, pp. 413-418, 1992.

[Engl 1981] W.L. Engl and H. Dirks, Models of Physical Parameters, in Introduction to the Numerical Analysis of Semiconductoir Devices and Integrated Circuits, pp. 42-46, Boole Press, Dublin, 1981.

[Farlow 1982] S.J. Farlow, Partial Differential Equations for Scientists and Engineers, Jon Wiley, New York, 1982.

[Geballe 1955] T.H. Geballe and G.W. Hull, The Seebeck Effect in Silicon, Physical Review, vol. 98, pp. 940-947, 1955.

[Glassbrenner 1964] C.J. Glassbrenner and G.A. Slack, Thermal Conductivity of Silicon and Germanium from 3 K to the Melting Point, Physical Review, vol. 134, pp. 1058-1069, 1964.

[Groot 1969] S.R. de Groot and P. Mazur, Non-equilibrium Thermodynamics, North-Holland, Amsterdam, 1969.

[Guerrieri 1983] R. Guerrieri and M. Rudan, Optimization of the Finite-Element Solution of the Semiconductor-Device Poisson Equation, IEEE Transactions on Electron Devices, vol. ED-30, pp. 1097-1103, 1983.

[Gyarmati 1970] I. Gyarmati, Non-equilibrium Thermodynamics, Springer, Berlin, 1970.

[Hackbush 1985] W. Hackbusch, Multi-Grid Methods and Applications, Springer, Berlin, 1985.

[Hackbush 1991a] W. Hackbusch and U. Trottenberg (eds.), Multigrid Methods III, Birkhäuser, Basel, 1991.

[Hackbush 1991b] W. Hackbush, Iterative Lösung großer Schwachbesetzter Gleichungssysteme, Teubner, Stuttgart, 1991.

[Hemker 1980] P.W. Hemker, On the Structure of an Adaptive Multi-Level Algorithm, BIT, vol. 20, pp. 289-301, 1980.

[Hiratsuka 1991] R. Hiratsuka, D.C. van Duyn, T. Otaredian and P. de Vries, A Novel Accelerometer based on a Silicon Thermopile, Tech. Digest, 6th Int. Conf. Solid-State Sensors and Actuators (Transducers '91), San Francisco, CA, USA, June 24-27, pp. 420-423, 1991.

[Hiratsuka 1992] R. Hiratsuka, D.C. van Duyn, T. Otaredian and P. de Vries, Design Considerations for the Thermal Accelerometer, Sensors and Actuators A, vol. 32, pp. 380-385, 1992.

[Holt 1965] E.H. Holt and R.E. Haskel, Plasma Dynamics, The Macmillian Company, New York, 1965.

[Hoop 1986] A.T. de Hoop, Radiation and Scattering of Electromagnetic Waves, Laboratory of Electromagnetic Research, Department of Electrical Engineering, TUD, Delft, 1986.

[Hughes 1987] T.J.R. Hughes, The Finite Element Method, Prentice-Hall, New Jersey, 1987.

[Huijben 1987] A.J.M. Huijben, Een hybride gemengde eindige-elementen methode met post-processing, toegepast op het Hall-probleem, Masters Thesis, Eindhoven University of Technology, 1987.

[Johnson 1981] C. Johnson and V. Thomée, Error Estimates for some Mixed Finite Element Methods for Parabolic Type Problems, R.A.I.R.O Numerical Analysis, vol. 15, pp. 41-78, 1981.

[Kaasschieter 1990] E.F. Kaasschieter, Preconditioned Conjugate Gradients and Mixed-Hybrid Finite Elements for the Solution of Potential Flow Problems, Ph.D. Thesis, Delft University of Technology, 1990.

[Kirschner 1991] I. Kirschner, S. Leppävuori and A. Haász, An Irreversible Thermodynamic Theory of Measuring Sensors, Sensors and Actuators, vol. A25-A27, pp. 677-682, 1991.

[Kirschner 1992] I. Kirschner, S. Leppävuori and A. Haász, Non-Equilibrium, Irreversibility, Non-Linarity and Instability in the Operation of Sensors, Sensors and Actuators, vol. A31, pp. 275-282, 1992.

[Kubo 1985] R. Kubo, M. Toda, N. Hashitsume, Statistical Physics II, Springer-Verlag, Berlin, 1985.

[Landsberg 1991] P.T. Landsberg, Recombination in Semiconductors, Cambridge University Press, Cambridge, 1991.

[Li 1962] J.C.M. Li, Stable Steady State and the Thermokinetic potential, The Journal of Chemical Physics, vol. 37, pp. 1592-1595, 1962.

[Lippman 1991] S.B. Lippman, C++ Primer, Addinson-Wesley, NewYork, 1991.

[Markowich 1983] P.A. Markowich, C.A. Ringhofer, S. Selberherr and M. Lentini, A singular perturbation Approach for the Analysis of the Fundamental Semiconductor Equations, IEEE Transactions on Electron Devices, vol. ED-30, pp. 1165-1180, 1982.

[Markowich 1990] P.A. Markowich, C.A. Ringhofer and C. Schmeiser, Semiconductor Equations, Springer, Wien, 1990.

[Marshak 1989] A.H. Marshak, Modeling Semiconductor Devices with Position-Dependent Material Parameters, IEEE Transactions on Electron Devices, vol. 36, pp. 1764-1772, 1989.

[Matsuno 1988] K. Matsuno and H.A. Dwyer, Adaptive Methods for Elliptic Grid Generation, Journal of Computational Physics, vol. 77, pp. 40-52, 1988.

[McCormick 1987] S.F. McCormick (ed.), Multigrid Methods, SIAM, Philadelphia, 1987.

[McCormick 1989] S.F. McCormick, Multilevel Adaptive Methods for Partial Differential Equations, SIAM, Philadelphia, 1989.

[Middelhoek 1981] S. Middelhoek and D.J.W. Noorlag, Three-dimensional Representation of Input and Output Transducers, Sensors and Actuators, vol. 2, pp. 29-41, 1981.

[Middelhoek 1986] S. Middelhoek and A.C. Hoogerwerf, Classifying Solid-State Sensors: The Sensor Effect Cube, Sensors and Actuators, vol. 10, pp. 1-8, 1986.

[Middelhoek 1989a] S. Middelhoek and D.C. van Duyn, Classification of Physical Effects used in Sensors, Proc. AIM Conference, November 14-17, 1989, Adelaide, Australia.

[Middelhoek 1989b] S. Middelhoek and S.A. Audet, Silicon Sensors, Academic Press, London, 1989.

[Miller 1974] D.G. Miller, The Onsager Relations; Experimental Evidence, Proc. Symp. Foundations of Continuum Thermodynamics, Bussaco, Portugal, pp. 185-214, 1974.

[Miller 1981a] K. Miller and R.N. Miller, Moving Finite Elements I, SIAM J. Numer. Anal., vol. 18, pp. 1019-1032, 1981.

[Miller 1981b] K. Miller, Moving Finite Elements II, SIAM J. Numer. Anal., vol. 18, pp. 1033-1057, 1981.

[Moglestue 1992] C. Moglestue, Monte Carlo Simulation of Semiconductor Devices, James & James Science Publishers, London, 1992.

[Molenaar 1992] H. Molenaar, Multigrid Methods for Semiconductor Device Simulation, PhD thesis, CWI, Amsterdam, 1992.

[Münster 1970] A. Münster, Classical Thermodynamics, Wiley, New York, 1970.

[Munter 1992] P.J.A. Munter and D.C. van Duyn, Spatial Symmetry of Transduction Effects in Hall Plates, Sensors and Actuators A, vol. 32, pp. 206-209, 1992.

[Mur 1985] G. Mur, A Finite Element Method for Computing Three-Dimensional Electromagnetic Fields in Inhomogeneous Media, IEEE Transactions on magnetics, vol. 21, pp. 2188-2191, 1985.

[Mur 1988] G. Mur, Optimum Choice of Finite Elements for Computing Three-dimensional Electromagnetic Fields in Inhomogeneous Media, IEEE Transactions on magnetics, vol. 24, pp. 330-333, 1988.

[Mur 1990] G. Mur, The Finite Element Modeling of Three Dimensional Time-Harmonic Electromagnetic Fields in Anisotropic and Strongly Inhomogeneous Media, COMPEL, vol. 9, pp. 83-86, 1990.

[Nag 1972] B.R. Nag, Theory of Electrical Transport in Semiconductors, Pergamon Press, Oxford, 1972.

[Nag 1980] B.R. Nag, Electron Transport in Compound Semiconductors, Springer, Berlin, 1980.

[Nakwaski 1983] W. Nakwaski, The Peltier Coefficient in Degenerate and Non-Degenerate Semiconductors, Electron Technology, vol. 14, pp. 81-99, 1983.

[Nedelec 1980] J.C. Nedelec, Mixed Finite Elements in R3, Numer. Math., vol. 35, pp. 315-341, 1980.

[Nedelec 1986] J.C. Nedelec, A New Family of Finite Elements in R3, Numer. Math., vol. 50, pp. 58-81, 1986.

[Oden 1983] J.T. Oden and G.F. Carey, Finite Elements: Mathematical Aspects (Volume IV), Prentice-Hall, New Yersey, 1983.

[Oden 1984] J.T. Oden and G.F. Carey, Finite Elements: Special Problems in Solid Mechanics (Volume V), Prentice-Hall, New Yersey, 1984.

[Otaredian 1992] T. Otaredian, Contactless Microwave Lifetime Measurement, Ph.D. Thesis, Delft University of Technology, 1992.

[Oudheusden 1992] B.W. van Oudheusden, Silicon Thermal Flow Sensors, Sensors and Actuators A, vol. A30, pp. 5-26, 1992.

[Penman 1986] J. Penman, Dual and Complementary Variational Techniques for the Calculation of Electromagnetic Fields, Advances in Electronics and Electron Physics, vol. 70, pp. 315-364, 1986.

[Polak 1987] S.J. Polak, C. den Heijer, W.H.A. Schilders and P. Markowich, Semiconductor Device Modelling from the Numerical Point of View, International Journal for Numerical Methods in Engineering, vol. 24, pp. 763-838, 1987.

[Polak 1988] S.J. Polak, W.H.A. Schilders and H.D. Couperus, A Finite Element Method with Current Conservation, in Simulation of Semiconductor Devices and Processes, vol. 3, eds. G. Baccarani and M. Rudan, Bologna, Italy, September 26-28, pp. 453-462, 1988.

[Prakash 1978] C. Prakash, Thermal Conductivity Variation of Silicon with Temperature, Microelectr. Reliab., vol. 18, pp. 333, 1978.

[Raviart 1977] P.A. Raviart and J.M. Thomas, Primal Hybrid Finite Element Methods for 2nd Order Elliptic Equations, Mathematics of Computation, vol. 31, pp. 391-413, 1977.

[Rheinboldt 1970] W.C. Rheinboldt, Iterative Solution of Nonlinear Equations in Several Variables, Academic Press, New York, 1970.

[Rheinboldt 1974] W.C. Rheinboldt, Methods for Solving Systems of Nonlinear Equations, SIAM, Philadelphia, 1974.

[Rheinboldt 1980] W.C. Rheinboldt, On a Theory of Mesh-Refinement Processes, SIAM J. Numer. Anal., vol. 17, pp. 766-778, 1980.

[Roosbroeck 1950] W.V. van Roosbroeck, Theory of Flow of Electrons and Holes in Germanium and other Semiconductors, Bell Syst. Techn. J., vol. 19, pp. 560-607, 1950.

[Schmidt 1988] G.H. Schmidt and F.J. Jacobs, Adaptive Local Grid Refinement and Multi-grid in Numerical Reservoir Simulation, Journal of Computational Physics, vol. 77, pp. 140-165, 1988.

[Schumacher 1985] B. W. Schumacher, Dimensional Terms for Energy Transport by Radiation and for Electromagnetic Quantities: Comments on the SI system, Advances in Electronics and Electron Physics, vol. 65, pp. 229-294, 1985.

[Segal 1984a] G. Segal, SEPRAN Programmers Manual, SEPRA, Leidschendam, 1984.

[Segal 1984b] G. Segal, SEPRAN Users Manual, SEPRA, Leidschendam, 1984.

[Segal 1984c] G. Segal, SEPRAN Standard Problems Manual, SEPRA, Leidschendam, 1984.

[Selberherr 1984] S. Selberherr, Analysis and Simulation of Semiconductor Devices, Springer-Verlag, New-York, 1984.

[Selberherr 1989] S. Selberherr, MOS Device Modeling at 77 K, IEEE Transactions on Electron Devices, vol. 36, pp. 1464-1474, 1989.

[Sherman 1978] A.H. Sherman, On Newton-Iterative Methods for the Solution of Systems of Nonlinear Equations, Siam J. Mumer. Anal., vol. 15, pp. 755-771, 1978.

[Spiegel 1974] M.R. Spiegel, Vector Analysis, McGraw-Hill, 1974.

[Stroustrup 1991] B. Stroustrup, The C++ Programming Language, Addinson-Wesley, NewYork, 1991.

[Stüben 1984] K. Stüben and U. Trottenberg, Multigrid Methods: Fundamental Algorithms, Model Problem Analysis and Applications, GMD, Bonn, 1984.

[TAHD 1978] W. Morris (ed.), The American Heritage Dictionary of the English Language, Houghton Mifflin, Boston, 1978.

[Toda 1983] M. Toda, R. Kubo, N. Saito, Statistical Physics I, Springer-Verlag, Berlin, 1983.

[Tonti 1972] E. Tonti, On the Mathematical Structure of a Large Class of Physical Theories, Acad. Naz. Dei, Lincei., Lii, Series III, pp. 48-56, 1972.

[Vanka 1986] S.P. Vanka, Block-Implicit Multigrid Solution of Navier-Stokes Equations in Primitive Variables, Journal of Computational Physics, vol. 65, pp. 138-158, 1986.

[Wachutka 1990] G. Wachutka, Rigorous Thermodynamic Treatment of Heat Generation and Conduction in Semiconductor Device Modeling, IEEE Transactions on Computer Aided Design, vol. 9, pp. 1141-1149, 1990.

[Weinberg 1972] S. Weinberg, Gravitation and Cosmology: Principles and Applications of the General Theory of Relativity, Wiley, New York, 1972.

[Wesseling 1992] P. Wesseling, An Introduction to Multigrid Mcthods, Wiley, New York, 1992.

[White 1985] J. White, F. Odeh, A.S. Sangiovanni-Vincentelli and A. Ruehli, Waveform Relaxation: Theory and Practice, Memorandum No. UCB/ERL M85/65, Electronics Research Laboratory, College of Engineering, University of Califomia, Berkeley, 1985.

[Ylilammi 1989] M. Ylilammi, Thermodynamics of Sensors, Sensors and Actuators, vol. 18, pp. 167-178, 1989.

[Young 1971] D.M. Young, Iterative Solution of Large Linear Systems, Academic Press, New York, 1971.

List of Symbols

This section first explains the main notational conventions used for the various physical quantities. Next, the main intensive, extensive and flux state variables and constitutive parameters are listed for each energy domain. This is done by grouping the table entries of a table; the first group represents the intensive state variables, the second group the extensive state variables and the third group the flux state variables. Finally, a fourth group represents the constitutive parameters which can be either explicit or implicit.

Some effort has been made towards a unified notation of the various physical quantities. The notation is somewhat different from the conventional one, however, we think the present one is more productive. For instance, energy densities are denoted by the symbol u and the energy of a component k is represented by u^k. In order to refer to a specific type of energy, e.g. electrical energy, an additional superscript is used, e.g. $u^{k,\,th}$. The following three tables summarize the conventions used for the notation of various density, flow and source quantities.

Density:

Particle	Mass	Charge	Energy	Heat	Entropy
n^k	$\rho^k = m^k n^k$	$\xi^k = q^k n^k$	u^k	h^k	s^k

Flux:

Particle	Mass	Charge	Energy	Heat	Entropy
$n^k_\alpha = n^k v^k_\alpha$	$\rho^k_\alpha = \rho^k v^k_\alpha$	$\xi^k_\alpha = \xi^k v^k_\alpha$	u^k_α	h^k_α	s^k_α

Source:

Particle	Mass	Charge	Energy	Heat	Entropy
$\dot{n}^k$	$\dot{\rho}^k = m^k \dot{n}^k$	$\dot{\xi}^k = q^k \dot{n}^k$	$\dot{u}^k$	$\dot{h}^k$	$\dot{s}^k$

Next, we list the main state variables and constitutive parameters for the electromagnetic (radiant), electrical and thermal energy domains.

Electromagnetic:

Symbol	Description	SI Unit
e_α	electric field strength	[V·cm^{-1}]
h_α	magnetic field strength	[A·cm^{-1}]
d_α	electric flux density	[A·s·cm^{-2}]
b_α	magnetic flux density	[V·s·cm^{-2}]
p_α	electric polarization	[C·cm^{-2}]

m_α	magnetic polarization	[A·cm^{-1}]
φ	scalar electric potential	[V]
a_α	vector magnetic potential	[V·s·cm^{-1}]
ξ_α	electric current density	[A·cm^{-2}]
ξ	electric charge density	[C·cm^{-3}]
ξ_α^{ext}	external electric current	[A·cm^{-2}]
K_α^{ext}	external magnetic current	[V·cm^{-2}]
$\varepsilon_{\alpha\beta}$	permittivity tensor	[-]
$\mu_{\alpha\beta}$	permeability tensor	[-]

Electrical:

Symbol	Description	Unit
E_f^e	Fermi energy of the electrons in the conduction band	[J]
E_f^h	Fermi energy of the electrons in the valence band	[J]
E_i	intrinsic Fermi energy	[J]
$\tilde{\nu}^e$	chemical potential of the electrons in the conduction band	[J]
$\tilde{\nu}^h$	chemical potential of the holes in the valence band	[J]
ν^e	electro-chemical potential of the electrons in the conduction band	[J]
ν^h	electro-chemical potential of the holes in the valence band	[J]
ϕ^e	quasi Fermi-potential of the electrons in the conduction band	[V]
ϕ^h	quasi Fermi-potential of the holes in the valence band	[V]
$E_f^{d_m}$	Fermi energy of the electrons in a donor-like localized state	[J]
$E_f^{a_n}$	Fermi energy of the electrons in an acceptor-like localized state	[J]
ϕ^{d_m}	quasi Fermi-potential of the electrons in a donor-like localized state	[V]
ϕ^{a_n}	quasi Fermi-potential of the holes in an acceptor-like localized state	[V]
n^e	electron density	[cm^{-3}]
n^h	hole density	[cm^{-3}]
n^{d_m}	donor-like localized state density	[cm^{-3}]
n^{a_n}	acceptor-like localized state density	[cm^{-3}]
$n^{d_m^+}$	ionized donor-like localized state density	[cm^{-3}]

$n^{a_n^-}$	ionized acceptor-like localized state density	[cm^{-3}]
$n^{d_m^\times}$	neutral donor-like localized state density	[cm^{-3}]
$n^{a_n^\times}$	neutral acceptor-like localized state density	[cm^{-3}]

n_α^e	electron flow	[s^{-1}·cm^{-2}]
n_α^h	hole flow	[s^{-1}·cm^{-2}]
ξ_α^e	electron contribution to charge flow	[C·s^{-1}·cm^{-2}]
ξ_α^h	hole contribution to charge flow	[C·s^{-1}·cm^{-2}]
ξ_α	total charge current density	[C·s^{-1}·cm^{-2}]
$\dot{n}^{e \to h}$	net recombination rate of electrons from the conduction band to the valence band	[s^{-1}·cm^{-3}]
$\dot{n}^{e \to d_m}$	net recombination rate of electrons from the conduction band to a donor-like localized state	[s^{-1}·cm^{-3}]
$\dot{n}^{e \to a_n}$	net recombination rate of electrons from the conduction band to an acceptor-like localized state	[s^{-1}·cm^{-3}]
$\dot{n}^{h \to d_m}$	net recombination rate of holes from the valence band to a donor-like localized state	[s^{-1}·cm^{-3}]
$\dot{n}^{h \to a_n}$	net recombination rate of holes from the valence band to an acceptor-like localized state	[s^{-1}·cm^{-3}]
$\dot{n}^{d_m \to e}$	net recombination rate of electrons from a donor-like localized state to the conduction band	[s^{-1}·cm^{-3}]
$\dot{n}^{d_m \to h}$	net recombination rate of holes from a donor-like localized state to the valence band	[s^{-1}·cm^{-3}]
$\dot{n}^{a_n \to e}$	net recombination rate of electrons from an acceptor-like localized state to the conduction band	[s^{-1}·cm^{-3}]
$\dot{n}^{a_n \to h}$	net recombination rate of holes from an acceptor-like localized state to the valence band	[s^{-1}·cm^{-3}]

E^{gp}	band gap	[J]
E^{cb}	conduction band edge	[J]
E^{vb}	valence band edge	[J]
E_0^{cb}	conduction band edge at zero electrostatic potential	[J]
E_0^{vb}	valence band edge at zero electrostatic potential	[J]
χ	electron affinity	[J]
q^e	electron charge	[C]
q^h	hole charge	[C]
m^0	electron rest mass	[kg]
m^{cb}	density-of-states effective mass of the conduction band	[kg]

m^{vb}	density-of-states effective mass of the valence band	[kg]
$m^{e}_{\alpha\beta}$	effective mass tensor of the electrons in the conduction band	[kg]
$m^{h}_{\alpha\beta}$	effective mass tensor of the electrons in the valence band	[kg
μ^{e}	electron mobility	[cm^2/Vs]
μ^{h}	hole mobility	[cm^2/Vs]
P^{e}	electron thermoelectric power	[V/K]
P^{h}	hole thermoelectric power	[V/K]

Thermal:

Symbol	Description	Unit
T^{e}	absolute temperature of the electron system	[K]
T^{h}	absolute temperature of the hole system	[K]
T^{l}	absolute temperature of the lattice system	[K]
T	absolute temperature	[K]
T_o, T_a	reference temperature or ambient temperature	[K]
s^{e}	entropy density in the electron system	[J·K^{-1}·cm^{-3}]
s^{h}	entropy density in the hole system	[J·K^{-1}·cm^{-3}]
s^{l}	entropy density in the lattice system	[J·K^{-1}·cm^{-3}]
s	total entropy density	[J·K^{-1}·cm^{-3}]
h^{e}	thermal energy density in the electron system	[J·cm^{-3}]
h^{h}	thermal energy density in the hole system	[J·cm^{-3}]
h^{l}	thermal energy density in the lattice system	[J·cm^{-3}]
h	total thermal energy density	[J·cm^{-3}]
h^{e}_{α}	heat flow in the electron system	[J·s^{-1}·cm^{-2}]
h^{h}_{α}	heat flow in the hole system	[J·s^{-1}·cm^{-2}]
h^{l}_{α}	heat flow in the lattice system	[J·s^{-1}·cm^{-2}]
κ_t	total thermal conductivity	[J·s^{-1}·cm^{-1}·K^{-1}]
P^{e}	electron thermoelectric power	[V/K]
P^{h}	hole thermoelectric power	[V/K]
Θ_D	Debye temperature	[K]
c^{e}	heat capacity of the electron system	[J·K^{-1}·cm^{-3}]

c^h	heat capacity of the hole system	$[J \cdot K^{-1} \cdot cm^{-3}]$
c^l	heat capacity of the lattice system	$[J \cdot K^{-1} \cdot cm^{-3}]$

Physical Constants:

Symbol	Description	Value	Unit
c	velocity of light	$3 \cdot 10^8$	[m/s]
ε_0	permittivity of vacuum	$8.85 \cdot 10^{-12}$	$[A \cdot s \cdot V^{-1} \cdot m^{-1}]$
μ_0	permeability of vacuum	$4\pi \cdot 10^{-7}$	$[V \cdot s \cdot A^{-1} \cdot m^{-1}]$
q	electron charge	$1.60219 \cdot 10^{-19}$	[C]
m	electron rest mass	$9.1095 \cdot 10^{-31}$	[kg]
k	Boltzmann constant	$1.3807 \cdot 10^{-23}$	[J/K]
h	Planck constant	$6.6262 \cdot 10^{-34}$	$[J \cdot s]$

Mathematical:

Symbol	Description
δ_{ij}	Dirac delta function
$\in_{ijk}$	Levi-Civita tensor of rank three
h^k	average grid size for grid k
μ	error amplification factor
$\bar{\mu}$	smooting factor
u^H	coarse-grid solution
u^h	fine-grid solution
u^k	solution on grid level k
r^H	coarse-grid residue
r^h	fine-grid residue
r^k	residue on grid level k
$u^{l,k}$	solution on patch l at level k
$\tilde{u}^k$	approximate solution on grid level k
$\tilde{u}^{l,k}$	approximate solution on patch l at level k
G_k	grid at level k
P^l_k	grid patch l of grid level k
L^k	discretized differential operator at level k
$\tilde{R}^k_{k+1}, \tilde{R}^H_h$	restriction operators for tranferring solutions
R^k_{k+1}, R^H_h	restriction operators for tranferring residuals
P^{k+1}_k, P^h_H	prolongation operators
I^{k+1}_k, I^h_H	interpolation operators

List of Symbols

Greek Alphabet:

Symbol	Description
α,A	alfa,ALPHA
β,B	bèta,BÈTA
γ,Γ	gamma,GAMMA
δ,Δ	delta,DELTA
ε,E	epsilon,EPSILON
ζ,Z	zêta,ZÊTA
η,H	êta,ÊTA
θ,Θ	thêta,THÊTA
ι,I	iota,IOTA
κ,K	kappa,KAPPA
λ,Λ	lambda,LAMBDA
μ,M	mu,MU
ν,N	nu,NU
ξ,Ξ	xi,XI
o,O	omnikron,OMNIKRON
π,Π	pi,PI
ρ,P	rho,RHO
σ,Σ	sigma,SIGMA
τ,T	tau,TAU
υ,Y	upsilon,UPSILON
ϕ,Φ	phi,PHI
χ,X	chi,CHI
ψ,Ψ	psi,PSI
ω,Ω	omega,OMEGA

List of Abbreviations

BFEM	Boundary Finite Element Method
BI	Boundary Interface
CG	Conjugate Gradient
CH	Chemical Domain
CS	Correction Scheme
EL	Electrical Domain
FAS	Full Approximation Scheme
FDM	Finite Difference Method
FEM	Finite Element Method
FVEM	Finite Volume Element Method
FVM	Finite Volume Method
FMG	Full Multigrid
IC	Interface Condition
II	Internal Interface
MA	Magnetic Domain
ME	Mechanical Domain
MG	Multigrid
OOP	Object-Oriented Programming
PDE	Partial Differential Equation
RA	Radiant Domain
SOR	Successive Over Relaxation
SSOR-CG	Symmetric Successive Over-relaxation Conjugate Gradient
TH	Thermal Domain
TIP	Thermodynamics of Irreversible Processes
TSM	Time-Stepping Method
WRM	Waveform Relaxation Method
WU	Work Unit
1D	One-Dimensional
2D	Two-Dimensional
3D	Three-Dimensional

Summary

Chapter 1

In Chapter 1 a general introduction is presented which may help to put the subject into perspective. In particular we motivate the choice of irreversible thermodynamics as a *consistent* framework for the *physical modeling* of solid-state transducers, and the use of the mixed discretization method and the multigrid method for the *numerical modeling* of solid-state transducers.

Chapter 2

In Chapter 2 we present various aspects that are helpful to come to a unified abstract description of a solid-state transducer configuration. In particular, we discuss an abstract geometrical representation of a generic transducer configuration in terms of regions, interfaces and boundaries. In order to describe their characteristic (physical) features appropriate model equations are linked to each of the regions, interfaces and boundaries. With respect to the construction of explicit model equations we argue and exemplify that the thermodynamics of irreversible processes provides a powerful and consistent physical modeling paradigm which can be applied to a variety of physical situations, including the physical modeling of solid-state transducers. We also show that the entire field can conveniently be organized by means of the signal or energy domains, with the tacit assumption that a certain region of a transducer configuration operates in one or more energy domains. We also discuss a generic abstract mathematical representation of the model equations and argue that, for the sake of simplicity, it is convenient to represent apparently different physical models in the same abstract (canonical) mathematical form. Also some attention is paid to the general treatment of interface and boundary conditions.

Chapter 3

In Chapter 3 the objective is to derive the model equations for a region of the transducer configuration operating in the thermal and/or electrical energy domain. By using the principles of the thermodynamics of irreversible processes a closed mathematical model, in terms of balance equations, equilibrium equations of state, and non-equilibrium equations of state is derived. The derivation uses the following simplifying assumptions,

- fixed chemical composition
- isostress
- mechanical rigidity
- mechanical equilibrium
- zero magnetic field
- electrostatic conditions

Where appropriate, we also compile several useful (parametrized) constitutive models from the available literature, however, it should be mentioned that most of these models are not tested for their compatibility with the available fabrication process. Essentially, this means that for each semiconductor fabrication process the parameters of each constitutive model should be calibrated to that particular process. This is an aspect often neglected in device modeling.

Future research in this field should obviously be directed towards the relaxation of the various simplifying conditions. This way we hopefully evolve towards a complete modular computer implementation of the model, which can be applied to various practical cases. As argued earlier the implementation of such a model should be modular, in a sense that the apparent complexity of the model should be configurable. The rather new object-oriented programming techniques can conveniently be used for this purpose. Also the use of high-level compilers that compile the actual simulation program from a high level problem definition language are expected to be useful. Despite these interesting aspects we digress from the present subject, because we feel that more substantial knowledge is necessary with respect to the implementation of the models. Therefore, in the following chapters the objective is to define a convenient framework for the discretization, solution and implementation of the thermoelectric model equations. The proposed framework can later on be extended to cope with extended physical models.

Chapter 4

In Chapter 4 the objective is to present a convenient framework for the discretization of the thermoelectric model equations. The discretization method is based on the mixed formulation of the problem. The main reason for using the mixed discretization method is that it naturally fits the form of the models resulting from thermodynamic systems theory. In particular, we discuss the mixed discretization method for a non-linear parabolic partial differential equation, representative for the thermoelectric problem as defined in Chapter 3. The mixed discrete equations are obtained by using the lowest-order Raviart-Thomas element, which is discussed for the triangular and parallelogramic case. Also the problem of the time-dependence and non-linearity of the problem are discussed.

At this stage we have at our disposal all the necessary tools for the discretization of the thermoelectric problem, the following chapter must deal with an effective numerical solution method for the resulting discrete model equations. We do not wish to use a standard solution method, because these have proven to be rather ineffective. Instead, we focus on the formulation of a multigrid method that can be used with the mixed discrete equations.

Chapter 5

In Chapter 5 we discuss an adaptive multigrid solution method which can be used to solve the discrete problems obtained from the mixed discretization method as discussed in Chapter 4. Besides the more intricate details, we also present a fairly complete introduction to the essential features of the multigrid method. With respect to the mixed discretization the Vanka-type relaxation operator and the intergrid operators are discussed. By means of local mode analysis we show that the Vanka-type relaxation is an effective error smoother. With respect to the intergrid operators we conclude that the natural operators suggested by the mixed discretization method are sufficiently accurate, except for the prolongation operator for the potentials. An improvement for this prolongation operator is proposed based on a post-processing technique which improves the prolongation operator to a piecewise linear instead of a piecewise constant prolongation. Also the concept of adaptive composite grids is discussed in great detail and in particular the multigrid error estimators in the context of the mixed discretization are derived.

Chapter 6

In Chapter 6 we present some guidelines for implementing the FEM, featuring mixed finite elements and multigrid, by means of an object-oriented programming language (OOP). A brief introduction to OOP is presented and it is argued that the use of the OOP features of the C++ languages can conveniently be used to construct an elegant implementation of the FEM. It is the author's experience that the effort needed to write a similar program in a traditional language such as Fortran-77 is significantly greater. The OOP paradigm is subsequently applied to the design of an object-oriented representation of the computational domain, the grid levels, the elements and efficient grid scanning algorithms. We emphasize that the approach forces strict locality, meaning that no global storage is used. In other words solutions, residuals and stiffness matrices are all stored element wise and are accessed by sending messages to objects of type *element*. Obviously, there are many more FEM features which lend themselves to an object-oriented implementation, however, for the sake of brevity these have to be omitted.

Chapter 7

In Chapter 7 we present a glimpse of the results. In particular, we show that the optimal multigrid efficiency can indeed be obtained for the thermoelectric problem. For this purpose we consider a simple test case dealing with the self-heating characteristics of a forward biased diode. The mixed formulation of the thermoelectric problem is discussed together with the application of the mixed discretization method. The resulting discrete equations are solved by means of the multigrid method and for the forward biased case the convergence results are discussed. The results clearly show the grid-size independent convergence behavior of the method. Clearly, many more features need to be treated, however, this is left for future investigations.

Samenvatting

Hoofdstuk 1

In hoofdstuk 1 wordt een algemene introductie gegeven welke behulpzaam is bij het in het perspectief plaatsen van het onderzoek. In het bijzonder motiveren we het gebruik van de irreversibele thermodynamica als een consistent raamwerk ten behoeve van het *fysisch modelleren* van vaste stof transducenten. Ook motiveren we het gebruik van de gemengde eindige discretizatie methode en de multirooster methode voor het *numeriek modelleren* van vaste stof transducenten.

Hoofdstuk 2

Hoofdstuk 2 behandelt verschillende aspecten welke van belang zijn bij het construeren van een abstracte beschrijving van een vaste stof transducent. In het bijzonder wordt een abstracte geometrische beschrijving van een generieke "transducer" configuratie beschreven in termen van "regions", "interfaces" en "boundaries". Om de karakteristieke fysische eigenschappen van de "regions", "interfaces" en "boundaries" te beschrijven wordt aan elk van de "regions", "interfaces" en "boundaries" een geschikt model gekoppeld. Met betrekking tot de constructie van expliciete modelvergelijkingen propageren we het gebruik van de irreversibele thermodynamica, welke een krachtig en consistent raamwerk voor het fysisch modelleren vormt en gebruikt kan worden in diverse fysische configuraties, waaronder ook het fysisch modelleren van vaste stof transducenten. Ook laten we zien dat het gehele veld handig onderverdeeld kan worden in termen van energiedomeinen waartussen informatie uitgewisseld wordt. Tevens beschrijven we een generieke abstracte mathematische representatie van de modelvergelijkingen en beargumenteren dat het om uniformiteitsredenen praktisch is om ogenschijnlijk verschillende fysische modellen in dezelfde abstracte mathematische representatie te formuleren. Tot slot wordt nog enige aandacht aan de algemene behandeling van de "interface" en "boundary" condities besteed.

Hoofdstuk 3

Het doel van hoofdstuk 3 is het opstellen van de modelvergelijkingen die een "region" van de "transducer" configuratie beschrijven welke in het thermische en/of elektrische energiedomein opereert. Op basis van de irreversibele thermodynamica stellen we een gesloten mathematisch model op, in termen van balansvergelijkingen, evenwichts toestandsvergelijkingen, en niet evenwichts toestandsvergelijkingen. Hierbij worden de onderstaande simplificerende aannamen gebruikt:

- niet veranderlijke chemische samenstelling
- uniforme stress

- mechanisch onvervormbaar
- mechanisch evenwicht
- geen magnetisch veld
- electrostatische condities

Tevens worden waar nodig verschillende (geparametrizeerde) constitutieve modellen gecompileerd uit de beschikbare literatuur. Hierbij moet vermeld worden dat deze modellen niet getest zijn op hun compatibiliteit met het beschikbare fabricageproces. In feite betekent dit dat voor elk fabricageproces de parameters van elk constitutief model geijkt moeten worden aan dat specifieke proces. Dit is een aspect wat vaak verwaarloosd wordt.

Toekomstig onderzoek moet zich uiteraard richten op het elimineren van de simplificerende aannamen. Op deze manier moet uiteindelijk een complete en modulaire computerimplementatie tot stand gebracht worden, welke in diverse praktische situaties gebruikt kan worden. Het is gewenst dat de implementatie modulair is, d.w.z. de ogenschijnlijke complexiteit van het model moet configureerbaar zijn. De object georiënteerde programmeermethode kan hier heel geschikt voor zijn. Ook hogere generatie compilers welke het simulatieprogramma compileren aan de hand van een probleemdefinitie taal kunnen uitkomst bieden.

Ondanks deze interessante aspecten verlaten we het huidige onderwerp, omdat op dit moment meer kennis nodig is met betrekking tot de implementatie van fysische modellen. In de volgende hoofdstukken is daarom het doel een geschikt raamwerk te formuleren ten behoeve van het discretizeren, oplossen en implementeren van de thermo-elektrische modelvergelijkingen. Dit raamwerk kan in een latere fase uitgebreid worden met meer geavanceerde fysische modellen.

Hoofdstuk 4

Het doel van hoofdstuk 4 is om een geschikt raamwerk te formuleren voor de discretizatie van de thermo-elektrische modelvergelijkingen. De discretizatiemethode is gebaseerd op de gemengde formulering van het probleem. De belangrijkste reden voor het gebruik van de gemengde discretizatiemethode is het feit dat deze nauw aansluit bij de modellen welke met behulp van de irreversibele thermodynamica verkregen zijn. Hier beschrijven we de gemengde discretizatiemethode voor het geval van een niet-lineaire parabolische partiële differentiaalvergelijking, welke representatief is voor het thermo-elektrische probleem. De discrete gemengde vergelijkingen worden verkregen met behulp van het Raviart-Thomas element van de laagste orde, welke beschreven wordt voor een driehoek en een parallellogram. Ook wordt aandacht besteed aan de tijdsafhankelijkheid en het niet-lineaire gedrag van het probleem.

In deze fase zijn alle benodigde gereedschappen ten behoeve van de discretizatie van het thermo-elektrische probleem beschikbaar. Het volgende hoofdstuk zal zich bezig houden met een effectieve numerieke oplosmethode voor de discrete thermo-elektrische modelvergelijkingen. Hiervoor willen we geen standaard oplosmethode gebruiken, omdat gebleken is dat deze niet geschikt zijn.

Hoofdstuk 5

In hoofdstuk 5 behandelen we een adaptieve multirooster methode welke gebruikt kan worden om de met behulp van de gemengde discretizatie methode verkregen problemen effectief op te lossen. Voor de lezer die niet bekend is met de multirooster methode wordt een introductie gegeven. In relatie tot de gemengde discretizatie methode bespreken we de Vanka relaxatie operator en de interrooster operatoren. Met behulp van "local mode analysis" laten we zien dat de Vanka relaxatie een zeer efficiënte fout "smoother" is. Met betrekking tot de interrooster operatoren laten we zien dat de natuurlijke interrooster operatoren van de gemengde discretizatie voldoende nauwkeurig zijn, behalve de prolongatie operator voor de thermodynamische potentialen. Op basis van een "post-processing" techniek wordt de orde van deze prolongatie operator verbeterd van stuksgewijs constant naar stuksgewijs lineair. Ook wordt aandacht besteed aan adaptieve "composite" roosters, in het bijzonder worden de multirooster foutschatters in de context van de gemengde discretizatie methode behandeld.

Hoofdstuk 6

Hoofdstuk 6 behandelt enkele richtlijnen voor de implementatie van de eindige elementen methode, in combinatie met gemengde eindige elementen en de multirooster methode, met behulp van een object georiënteerde programmeertaal. Een korte introductie tot object georiënteerd programmeren (OOP) wordt gegeven. We demonstreren dat de eigenschappen van een object georiënteerde programmeertaal zeer goed gebruikt kunnen worden om een elegante implementatie van de eindige elementen methode te realizeren. Het is de ervaring van de auteur dat de moeite om een gelijkwaardig programma in een traditionele taal te formuleren beduidend hoger ligt. De OOP methode wordt daarna gebruikt voor het ontwerpen van de object georiënteerde representaties van het "computational domain", de "grid levels", de elementen en efficiënte grid-scanning algorithmen. We benadrukken dat deze aanpak strikte localiteit forceert, hetgeen betekent dat geen globale opslag gebruikt wordt. Met andere woorden; oplossingen, residuen en stijfheidsmatrices worden elementsgewijs opgeslagen en geadresseerd door boodschappen naar objecten van het type element te zenden. Daar bij object georiënteerd programmeren de standaard methode voor het alloceren van geheugen voor de objecten tamelijk veel overhead kost worden ook mogelijkheden gegeven om dit process efficiënter te maken.

Hoofdstuk 7

In hoofdstuk 7 lichten we een tipje van de sluier op met betrekking tot de resultaten. In het bijzonder laten we zien dat de optimale multirooster efficiëntie inderdaad verkregen wordt voor het thermo-electrische probleem. Hiervoor beschouwen we een testgeval welke het zelfverwarmingseffect van een voorwaarts ingestelde diode modelleert. We formuleren het thermo-electrische probleem in de gemengde vorm en passen de gemengde discretizatie methode toe. De resulterende vergelijkingen worden numeriek opgelost met behulp van de multirooster methode. Voor het genoemde testprobleem worden de convergentieaspecten van de methode belicht. De resultaten demonstreren duidelijk de rooster onafhankelijke convergentie eigenschappen van de methode. Uiteraard zijn er veel meer eigenschappen te onderzoeken, maar dat is werk voor toekomstig onderzoek.

Acknowledgements

It is a great pleasure for me to thank Professor S. Middelhoek for giving me the opportunity to work on solid-state transducer modeling in the Laboratory for Electronic Instrumentation of the Electrical Engineering Department of the Delft University of Technology. It was only the academic freedom granted under his supervision that allowed the liberty to pursue the ideas as presented in this dissertation, for this I am truly grateful.

Besides the moral support of Professor Middelhoek, my academic life was made "easy" by a generous grant from FOM (Foundation for Fundamental Materials Research) under project number DEL88-1638. The financial support for traveling abroad received from Shell and NWO was also greatly appreciated.

My thanks go to dr. G. Mur of the Electromagnetics Department of the Delft University of Technology, who in the early stages of my "quest" suggested the mixed finite element discretization method.

I am also greatly indebted to the Applied Mathematics Department of the Delft University of Technology. Especially I would like to thank Professor P. Wesseling for his advice on the multi-grid method, and ir. G. Segal and dr. N. Praagman for introducing me to the complex world of numerical mathematics.

Thanks also to Dr. ir. C.H. Venner and dr. ir. A.A. Lubrecht of the tribology group of the Faculty of Mechanical Engineering of the University of Twente for their help with the application of the multigrid method.

My thanks also go to Mrs. Zaat-Jones for her skillful correction of the English text.

I also wish to thank R. Hiratsuka, a visiting Japanese scientist from Nippon Steel with whom I closely worked on the thermal accelerometer project. He somehow always managed to achieve beyond the call of duty. I wonder if all Japanese are like this, because if they are we are in serious trouble. The help of P.H.S. de Vries in the thermal accelerometer project was greatly appreciated.

Special thanks go to my fellow Ph.D. students T. Otaredian, P.J.A. Munter and S.E. Wouters. In particular my thanks go to Toofan, for always managing to provide me with the drive to proceed with matters that presented a challenge instead of a smooth and prepaved road to "success", Peter, whose help, laughter and enthusiasm always provided a sparkling source of light in the "dark age" of being a Ph.D. student, and Sietse, for being a fine roommate with whom it always was fun to discuss the more serious matters of life.

Finally, I would like to thank Edith, Leendert-Jan and my parents for their moral support and for putting up with me during this last year. If anyone felt neglected I truly apologize, but it had to be done.

About the Author

Dave C. van Duyn was born in Christchurch, New Zealand, on April 29, 1961. His secondary education was at the Pieter Groen College, Katwijk, The Netherlands, where he obtained his HAVO diploma in 1979. In 1983 he received his B.Sc. degree in Electrical Engineering from the Municipal Polytechnical School in the Hague. In 1983 he continued his studies at the Faculty of Electrical Engineering of the Delft University of Technology.
In the first half of 1988 he was on leave at the IBM Thomas J. Watson Research Center, Yorktown Heights, New York, USA, where he was engaged in the experimental and theoretical analysis of mobility models for the simulation of MOSFETs in the IBM "FIELDAY" simulation program. In 1988 he received his M.Sc. (cum laude) in electrical engineering from the Delft University of Technology. In the same year Dave started his Ph.D. research at the Laboratory of Electronic Instrumentation, in the silicon solid-state sensor group under supervision of Prof. dr. ir. S. Middelhoek. His research was concerned with a study towards the modeling and simulation of solid-state silicon sensors, in particular, those operating in the thermal and electrical signal domain. The results of his research are described in this dissertation. He has also written several papers closely related to this work. In addition, he has presented his work at five international conferences. A visit to the MRS conference in Anaheim (USA) was sponsored by Royal Dutch Shell, and a visit to the TRANSDUCERS 91 conference in San Francisco (USA) was sponsored by NWO (the Dutch Association for Scientific Research).
Besides his research activities, Dave's hobbies are reading, listening to classical music, jazz-rock and fusion music. Dave also plays electric guitar in the rock band "Private Eye".

About the Author

List of Publications and Presentations

Publications:

[1] A.W. van Herwaarden, D.C. van Duyn, B.W. van Oudheusden and P.M. Sarro, Integrated Thermopile Sensors, Sensors and Actuators, vol. A21-A23, pp. 621-630, 1989.

[2] D.C. van Duyn and S. Middelhoek, Information Transduction in Solid-State Transducers: A General Thermodynamic Systems Approach, Sensors and Actuators, vol. A21-A23, pp. 25-32, 1990.

[3] S. Middelhoek and D.C. van Duyn, Classification of Physical Effects Used in Sensors, Proc. AIM Conference, November 14-17, 1989, Adelaide, Australia.

[4] D. C. van Duyn and P. de Vries, Finite Element Modeling of Thermoelectric Materials, Proc. Materials Research Society Symposium: Modern Perspectives on Thermoelectrics and Related Materials (volume 234), May 1-2, 1991, Anaheim, CA, U.S.A.

[5] A.W. van Herwaarden, D.C. van Duyn and J. Groeneweg, Small-size Vacuum Sensors Based on Silicon Thermopiles, Sensors and Actuators A, vol. 25-27, pp. 565-569, 1991.

[6] R. Hiratsuka, D.C. van Duyn, T. Otaredian and P. de Vries, A Novel Accelerometer Based on a Silicon Thermopile, Proc. Transducers '91, San Francisco, CA, June 23-27, 1991.

[7] D.C. van Duyn, Finite Element Modeling of Sensors: Thermoelectric Effects, Proc. Transducers '91, San Francisco, CA, June 23-27, 1991.

[8] R. Hiratsuka, D.C. van Duyn, T. Otaredian and P. de Vries, Design Considerations for the Thermal Accelerometer, Sensors and Actuators A, vol. 32, pp. 380-385, 1992.

[9] D. C. van Duyn and P.J.A. Munter, Finite Element Modeling of Thermoelectric Materials and Devices, Sensors and Actuators A, vol. 32, pp. 413-418, 1992.

[10] D.C. van Duyn and S. Middelhoek, Three-Dimensional Representation of Sensors and Actuators, in Thin Film Resistors Sensors, ed. P. Ciureanu, 1992.

[11] P.J.A. Munter and D.C. van Duyn, Spatial Symmetry of Transduction Effects in Hall Plates, Sensors and Actuators A, vol. 31, pp. 206-209, 1992.

Presentations:

[12] D.C. van Duyn, Information Transduction in Solid-State Transducers: A General Thermodynamic Systems Approach, Transducers '89, June 25-30, 1989, Montreux, Switzerland.

[13] D.C. van Duyn, Finite Element Modeling of Thermoelectric Materials, Materials Research Society Symposium, April 29-May 3, 1991, Anaheim, CA, USA.

[14] D.C. van Duyn, Finite Element Modeling of Sensors: Thermoelectric Effects, Transducers '91, June 23-27, 1991, San Francisco, CA, USA.

[15] D.C. van Duyn, Finite Element Modeling of Thermoelectric Materials and Devices, Eurosensors V, September 30-October 2, 1991, Rome, Italy.

[16] D.C. van Duyn, The Thermodynamic Classification of Solid-State Sensors as a Basis for a Finite Element Simulation Tool, The Brussels Workshop on "Fundamental Issues in Sensor Science", September 25-26, 1991, Brussels, Belgium.

[17] D.C. van Duyn, Modeling and Simulation of Solid-State Transducers: The Thermal and Electrical Signal Domains, Multigrid, accepted for presentation at the Eurosensors VII Conference, September 26-29, 1993, Budapest, Hungary.