Master Thesis

Optimal Test Signal Generation for

Analogue Circuits

Date: 08 September 2005
Author: Tobias Völkel
Born 29 March 1978
in Oldenburg

Supervisors: Prof. Dr. Ing. Wolfgang Mathis
Dr. Ir. Jan ter Maten
Ir. Guido Gronthoud

Project Leader: Ing. Shaji Krishnan
Abstract

Tobias Völkel

“Optimal Test Signal Generation for Analogue Circuits”

The quality standards in the development and manufacturing of integrated circuits in the present and the future put higher demands on the test of those circuits. A wide range of test and test preparation methods exist for digital circuits. For analogue integrated circuits those methods are still limited or under development. In the past analogue circuits were tested using the specification oriented approach. With the increasing complexity the test costs for analogue circuits became a key part of the overall manufacturing costs. To reduce the test complexity defect oriented test methods are under development. Due to the higher complexity of analogue circuits there are no general tools for automated analogue test design available. Analogue circuits are continuous in time and values, which makes the definition of analogue fault models much more complicate than it is for digital circuits. The same is true for the development of automated test pattern generators for analogue test. One part of analogue ATPG’s is the automated generation of optimum test signals.

This presentation will deal with automated test signal generation techniques to obtain optimum test signals. This test method is based on transient time domain test using the defect-oriented approach. The faults are modelled through component variations or changes of the circuit’s structure by adding opens or bridges. A systematic and efficient procedure to determine the optimal transient stimuli to test analogue circuits is described. The advantage of this method over existing ones is the determination of the optimal signal without employing any computationally intensive optimisation techniques. The overall reduction in computation time over comparable techniques is about 60%. The obtained test signal is a piecewise constant signal (binary signal), which discriminates the difference between the good circuit and the faulty one.
1. Table of Contents

1. **Table of Contents** .................................................................................................................... I
   1.1. Figures ................................................................................................................................. II
   1.2. Tables ................................................................................................................................. IV

2. **Introduction** ........................................................................................................................... 1
   2.1. Test of integrated circuits ................................................................................................. 2
   2.2. Goals of this work ............................................................................................................... 4

3. **Theoretical Background** ...................................................................................................... 6
   3.1. Defect-oriented test of analog circuits .............................................................................. 6
   3.2. Fault simulation .................................................................................................................. 8
   3.3. Test Signal Generation ..................................................................................................... 10
   3.4. Theory developed by Bernhard Burdiek ........................................................................... 11

4. **Comparison of TORAD to PStar simulator** ................................................................. 14

5. **Test Signal Generation with Philips programs** ......................................................... 18
   5.1. Implementation of the binary signal ................................................................................. 20
   5.2. Optimisation with Adapt using a random start signal .................................................. 27
   5.3. Different start signals ....................................................................................................... 28
   5.4. Intermediate Results ........................................................................................................ 30
   5.5. Example: Continuous Time Band Pass Filter .............................................................. 30
   5.6. Example: Low-Noise-Amplifier ...................................................................................... 37

6. **Equidistant binary signal** .................................................................................................. 39
   6.2. Frequency sweep for the equidistant binary signal ...................................................... 41
   6.3. Example: Continuous Time Band Pass Filter .............................................................. 42
   6.4. Simulation Time ................................................................................................................ 48
   6.5. Example: Low-Noise-Amplifier ...................................................................................... 49
   6.6. Example: Folded Operational Amplifier ............................................................................ 51

7. **Summary and Conclusions** ............................................................................................... 52

8. **Future Work** ..................................................................................................................... 53

9. **Appendix** ............................................................................................................................ 55
1.1. Figures

Figure 3-1: Fault Classification ................................................................. 8
Figure 3-2: Fault Simulation ................................................................. 9
Figure 3-3: Goal of the Test Signal Generation ......................................... 10
Figure 3-4: Qualitative form of a binary signal ........................................ 13
Figure 4-1: Transient simulation of the CTBP filter: Input signals for TORAD and PStar .... 15
Figure 4-2: Transient simulation of the CTBP filter: Output signals for TORAD and PStar .... 15
Figure 4-3: Transient simulation of the CTSV filter: Input signals for TORAD and PStar .... 16
Figure 4-4: Transient simulation of the CTSV filter: Output signals for TORAD and PStar .... 16
Figure 5-1: Comparison of TORAD and Adapt optimization ........................... 18
Figure 5-2: Abstract flow to optimise the test signal using Adapt .................... 19
Figure 5-3: Min. merit function: trust-region approach ................................ 20
Figure 5-4: The general pulse function .................................................... 21
Figure 5-5: Single pulses added to one pulse model .................................... 22
Figure 5-6: Simulation time limits the number of pulses .............................. 23
Figure 5-7: Example for the implementation of the pulse model (pulse duration) ........ 23
Figure 5-8: Pulse model with steps ........................................................ 24
Figure 5-9: Example for the implementation of the pulse model (Switch times) ........ 25
Figure 5-10: Pulse model with fixed pulses ............................................. 26
Figure 5-11: Definition of the sinsq function ............................................... 27
Figure 5-12: Example of a sinsq waveform .............................................. 27
Figure 5-13: Extended flow of the optimization with Adapt ........................... 28
Figure 5-14: Different start signals for the optimization with Adapt .................. 29
Figure 5-15: Schematic for Continuous Time Band Pass Filter ................... 31
Figure 5-16: Frequency response for the fault free circuit $V_{out}(j\omega)$ .............. 31
Figure 5-17: Input signals of TORAD and Adapt (TORAD start signal) ............. 32
Figure 5-18: Comparison $|V_{out}^{R} - V_{out}^{L}|$ of TORAD and Adapt (TORAD start signal) .... 33
Figure 5-19: CTBP Filter: Adapt Optimization (Start signal: one pulse) ......................................................... 33
Figure 5-20: Comparison (|Δout|): TORAD and Adapt (one pulse) .............................................................. 34
Figure 5-21: Comparison (Input): TORAD and Adapt (five pulses) .............................................................. 34
Figure 5-22: Comparison (|Δout|): TORAD and Adapt (five pulses) .............................................................. 35
Figure 5-23: CTBP Filter: PStar simulation (five pulses) .............................................................. 35
Figure 5-24: Comparison (|Δout|): TORAD and PStar (five pulses) .............................................................. 36
Figure 5-25: Work flow for the LNA ........................................................................................................ 37
Figure 5-26: Test flow for the low noise amplifier .................................................................................. 38
Figure 6-1: Flow-chart of the optimization approaches ............................................................................. 39
Figure 6-2: Flow chart to find the frequencies for each fault ...................................................................... 42
Figure 6-3: Maximum difference of the sinusoidal transient simulation .................................................. 43
Figure 6-4: Maximum difference of the transient simulation with a binary signal .................................... 44
Figure 6-5: Binary signal with \( f_1 = 7.08 \text{ kHz} \) ........................................................................ 45
Figure 6-6: Binary signal with \( f_2 = 11.56 \text{ kHz} \) ........................................................................ 46
Figure 6-7: Input at frequency \( f_1 \) and output for the good circuit and the faulty circuit \( F_1 \) ............. 47
Figure 6-8: Difference between the good circuit and fault \( F_1 \) for frequency \( f_1 \) and \( f_2 \) ............. 47
Figure 6-9: Input at frequency \( f_2 \) and output for the good circuit and the faulty circuit \( F_2 \) ............. 48
Figure 6-10: Difference between the good circuit and fault \( F_2 \) for frequency \( f_1 \) and \( f_2 \) ........... 48
Figure 6-11: LNA comparison frequency sweep of sine wave to binary signal \( F_{35} \) ....................... 50
Figure 6-12: LNA comparison frequency sweep of sine wave to binary signal \( F_{56} \) ....................... 50
Figure 6-13: Folded OpAmp comparison frequency sweep of sine wave to binary signal ............ 51
Figure 9-1: Results LNA; Bridge 0 ........................................................................................................ 62
Figure 9-2: Results LNA; Bridge 3 ........................................................................................................ 63
Figure 9-3: Results LNA; Bridge 15 ...................................................................................................... 64
Figure 9-4: Results LNA; Bridge 46 ...................................................................................................... 65
Figure 9-5: Results LNA; Bridge 53 ...................................................................................................... 66
Figure 9-6: Results LNA; Bridge 55 ...................................................................................................... 67
Figure 9-7: Results LNA; Bridge 56 ...................................................................................................... 68
Figure 9-8: Results LNA; Bridge 60 ...................................................................................................... 69
Figure 9-9: Results LNA; Bridge 65 ...................................................................................................... 70
Figure 9-10: Results LNA; Bridge 71 .................................................................................................... 71
Figure 9-11: Results LNA; Bridge 81 .................................................................................................... 72
Figure 9-12: Results LNA; Bridge 88 .................................................................................................... 73
Figure 9-13: Results LNA; Bridge 89 .................................................................................................... 74
Figure 9-14: Results LNA; Bridge 104 ................................................................................................ 75
Figure 9-15: Results LNA; Bridge 119 ................................................................................................ 76
Figure 9-16: Results LNA; Bridge 129 ................................................................................................ 77
1.2. Tables

Table 5-1: Variables and constrains of the pulse model ................................................... 20
Table 5-2: Comparison of the merit functional for the different test signals ......................... 36
Table 6-1: Injected faults and values .................................................................................. 42
Table 6-2: Calculation Time for the simulations ................................................................. 44
Table 6-3: Quantitative description of the binary signals .................................................... 45
Table 6-4: Calculation Time for the simulations ................................................................. 49
Table 9-1: single_pulse_model_table.txt ........................................................................... 57
Table 9-2: tb_sine_sweep_table.txt ............................................................................... 58
Table 9-3: tb_pulse_sweep_table.txt ............................................................................... 58
Table 9-4: Corresponding frequency to the number of pulses ........................................... 61

Remarks:
For the better readability the absolute function is always written in the form $y = |f(x)|$. This is correct for discrete values. If $f(x)$ is a continuous function the absolute function has to be changed to $y = +\sqrt{(f(x))^2}$. 
2. Introduction

Since the beginning of integrated circuits in the end of the fifties we can find them now in nearly every part of our life. Today they are one of the most important parts of the electronic industry. The integration of digital and analog components on one single chip belongs to the standard development and production techniques of high integrated circuits. One reason for the increase of analog circuits is the development of faster and more complex analog circuit design tools. Furthermore the technology to build mixed-signal IC increased the number of analog components. The advantages of analog and mixed signal circuits offering complex functionality with a low computation time and often fewer area requirements than a digital circuit with the same functionality raised the number of analog circuits as well. Hence a lot of effort has taken place to boost the use of analog and mixed-signal circuits.

In the beginning of integrated circuits, testing was not the most important problem due to their low complexity and they were mainly digital. Meanwhile the complexity raised in a way that testing became to one of the major problems of integrated circuits. A lot of research and development has been done according to digital circuit testing and it is still very important whereas the testing of analog and mixed-signal circuits got more and more important in the last years. With the high integration and complexity the test problem gets more difficult because of the loss of observability and controllability for the internal nodes and components [1]. The complexity of analog circuits is much higher than digital circuits [2]. Today there are hardly any industrial tools for automated analog testing. This is one main reason why the test costs of analog components increase more than their complexity increases. Even in mixed-signal circuits the test costs are dominated by the analog functions [3]. One possibility to reduce the test time and in that way the test costs is to find suitable test signals. This work is about finding good analog test signals.
2.1. Test of integrated circuits

Testing plays one of the major roles in microelectronics. Even during the development of a new circuit it needs to be tested. Those tests include problems like logic, speed and functionality. As well there are some optimizations and tests according to the power consumption, needed area and speed. Those questions are answered through knowledge and simulations. As soon as the circuit is in production you need to test the circuit. At the end of the manufacturing you have to figure out if the circuit is fault free.

The fault identification can be reduced to a diagnosis problem [4]. All parameters of the circuit have to be identified through measurements at the inputs and outputs. But the test of an integrated circuit by identifying all circuit parameters can be impossible due to limited observability and controllability or it is too expensive because of the high complexity [5], [6]. For that reason many different strategies have been developed. One of the main strategies is the defect-oriented test of digital, analog and mixed-signal circuits. Another idea is to add structures to the circuit that improve the testability; like scan paths or extra test nodes. This approach is called “Design for Testability” (DFT).

The reason for testing integrated circuits is to find physical defects that occur during manufacturing. Those defects can cause the circuit to have a faulty behaviour. Some reasons for those defects are dust, human errors or defects and recalibrations of the machines. For a defect-oriented test you need suitable test signals based on the fault models. Test signal generation deals with the problem to calculate a test signal to detect faults in a circuit. An automated flow to generate those test signals is called “Automated Test Pattern Generation” (ATPG). The fault simulation belongs to the important tools for circuit testing and diagnosis. It helps to classify the quality of the test signal; especially what fault coverage can be achieved. The test itself is done on an external tester which automatically performs the needed test at the circuit. Those machines and the needed environment is called “Automated Test Equipment” (ATE). It is also possible to add structures to the circuit that are designed to test the circuit or parts of it without an external tester. They are called Build-In-Self-Tests.
(BIST). The test costs for a test are calculated as the costs for the test preparation (off-line) and the test itself (on-line) using the circuit and the ATE [7], [8].

To investigate the electrical behaviours of physical defects of a circuit fault models are used. Those fault models map a physical defect on an electrical behaviour of the circuit. In most cases the defects are caused by random dust particles. They appear in variations of component parameters or in shorts and opens [9]. This leads to the different types of testing a circuit. You can divide them into two main groups:

1. Defect-oriented Test
2. Specification-oriented Test

The specification-oriented test uses the input - output behaviour of the circuit. This test is used to identify the circuit’s specifications and does not use any information of the internal structure of the circuit. Hence you need to have a lot of knowledge about the circuit itself to test its function. Since different specifications are tested in different manners, analogue functional testing is costly and time consuming.

The defect-oriented test approach uses the internal information about the components and the connection network. This description of the circuit can be on different abstraction levels. Digital circuits are often described on the gate level. Analogue circuits are described on the transistor level (discrete components like transistors, resistors, capacitors, inductors, …). The defect-oriented approach is used in this work.

The main goal of test signal generation is the automated generation of a test signal that is suitable to detect the possible defects in a layout. To find the defects the response of the good (fault free) circuit is compared with the responses of the faulty circuits. In most cases the responses are obtained through simulation of the circuit or characterisation of good devices. For the comparison you have to take into account that the good circuit can have process variations within the tolerances. The measured circuit responses are compared to a pass / fail limit. (Chapter 3)
Different test and diagnosis methods have been developed. In the work of Duhamel [10] you can get an overview of the different test methods of the 60s and 70s. At that time mainly fault diagnosis and the identification of the circuit’s parameters and specifications have been developed. In the 90s the development of methods for self-test, fault simulation and different defect-oriented test methods started. These test methods are divided in three groups. The first one is the DC test method. For this test a time constant voltage or current is applied to the circuit. You need to have enough nodes for applying and measuring the voltage and current. The VDD can also be as an input node [11]. With those time-invariant signals it is impossible to catch all faults, especially faults due to dynamic components like capacitors and inductors. The second class is called AC test methods. For this test a sine wave signal is applied to the linearized circuit. Different test signal generation methods have been suggested. As well as for the DC test you have to take the settling time for the circuit and the test environment into account. This leads to longer test times. In addition to that the test does not work for non-linear circuits, which cannot be linearized at the operating point or the operation point may not be the best frequency to test the circuit. The AC test only works for the small signal behaviour.

The last class is the method that deals with transient dynamic behaviour. Transient test signal have more and better possibilities to detect faults in a circuit. Another advantage of transient test signals is that they do not need to take care of the settling time, which leads to shorter test times. The dynamic behaviour and faults in dynamic components are included in the test method. Burdiek [12] could determine in his work the qualitative form of an optimal test signal for the transient test of analog circuits. An optimum test stimulus to test an analog circuit must be a piecewise constant signal, called binary signal.

### 2.2. Goals of this work

This work is based on the dissertation of Bernhard Burdiek. He points out the advantages of the transient test and calculates the required input signal for a
set of faults. The goal is to validate his results on Philips industrial circuits with the help of Philips tools like PStar and Adapt. In the End there should be a workflow how to find the optimal signal.

In fact it is very difficult to find an optimal signal for analog circuit test without the internal knowledge of the circuit. The Philips optimiser Adapt need the right start signal for the optimization to find a comparable good signal as Torad (This is a program developed by Burdiek during his dissertation) does. For the first steps the examples of Burdieks dissertation were tested with the Philips tools. During that work the needed start signal turns out to be a very good test signal. The test signals found for the initialization of Adapt are equidistant binary pulses.

After this validation of Burdieks theory some Philips industrial circuits were tested with that developed workflow. The problem of finding a suitable start signal for the optimisation with Adapt led to difficulties. To solve this problem a new algorithm was developed to cope with that problem. It uses the idea that all faults are frequency dependent and have at least one frequency at which the fault is discriminated the most. Using the equidistant pulse as input function and sweeping over a frequency range is possible but has a high computational effort. This could be reduced by using a sinusoidal signal.
3. Theoretical Background

Some general information on analog testing and signal generation is presented in this chapter. It starts with explanations about defect-oriented testing and their fault models. These fault models help to perform a fault simulation to figure out how the faulty circuit responds to a test signal. The fault simulation is an important tool for test generation. To get good fault coverage the test method need to have a suitable test signal. Some background information about test signal generation is presented in Chapter. The test signal generation method of Bernhard Burdiek is the basis of this work and a brief overview of that method is described in the end of this chapter.

3.1. Defect-oriented test of analog circuits

In the past analog circuits were mainly tested trough specification-oriented tests [10]. To determine if an analog circuit is defect free the specifications of that circuit are validated if they are within a certain tolerance of the specifications. The specifications have been defined during the circuit development. Those specification-oriented tests are very complex and time consuming, thus they are very expensive. Looking at big analog circuits with many blocks the problems get worse due to the lack of observability. For those reasons the defect-oriented test approach gets more important. It takes the internal structure in account and uses this information to build fault models, which are mapped to the physical defects. The problem with that kind of test is that consumers are still used to get the measured specifications like gain or noise ratio of the device even if it has never been measured.

During manufacturing different defects in the circuit can occur. Some defects are caused by dust, particles form the environment or by defects of the machines like mask displacement or temperature variations. Fault models are used to map all these defects to the layout. Defect-oriented fault models are described on the transistor level (discrete components). At that level defects are
modelled on their physical occurrence like bridges, opens and parameter changes of single components. These fault models characterize the electrical behaviour of faults and can be used to simulate a faulty circuit. The simulated behaviour of a faulty circuit with the use of a fault model should correlate with the electrical behaviour of a real defect. The inductive fault analyzes (IFA) [13] extracts a fault list using the layout and technology of the circuit. The IFA can also calculate the likelihood of a defect.

The stuck-at fault model is the most commonly used logic-level fault model in the digital domain. A defect causes a line (connection between gates) to have a permanent logic 1 or 0 value. With that model the faulty logic function can easily be calculated. For analog circuits no established fault model exists. One reason is that digital circuits can be described in Boolean logic and analog circuits use differential equations. For the digital domain it is sufficient to test discrete states but for analog test you have to test in the time and value continuous domain and tolerances has to be taken into account. Analog fault models need to be more complex in their description because analogue defects have different impacts on the faulty behaviour. In the literature different fault classifications are presented [14]. One possibility is to describe defects according to their impact size like:

1. Hard-faults
2. Soft-faults

Hard-faults are defects like a high resistive open or a low resistive bridge. Normal values for a hard fault are $1 \, \Omega$ for a bridge and $1 \, \text{M\Omega}$ for an open. A bridge with a low resistant or an open with a high resistant is called soft fault (See Figure 3-1). The transition between a hard-fault and a soft fault is smooth. Variations of components can be hard-faults or soft-faults depending on their impact.
Another classification is to describe the defect type. If it changes the circuit structure we speak of structural faults like opens and bridges. All component variations not caused by a structural fault are called parametric faults. To define a structural fault a resistance is injected in the circuit.

### 3.2. Fault simulation

With the help of the fault simulation the impact of faults and their electrical behaviour can be examined. The fault simulation calculates the circuit response of the good device (fault free) and all different faulty devices. A faulty device is created by injecting one fault from the given fault set. To test the whole fault set every fault is injected one by one sequentially. This is shown in Figure 3-2. In this work only single faults are assumed to occur.
The fault simulation determines if the difference between the good and the faulty device is large enough to recognize the fault. All devices are stimulated with the same input signal $u(t)$. The components in a good circuit can have deviations from the nominal values. These variations cause the good circuit to have a more than one good response. All responses within those deviations build a tolerance band. To identify a faulty circuit the output is compared to the tolerance band. Using Monte Carlo simulations or real measurements the tolerance band is defined for the circuit test. If the Monte Carlo simulation is too time consuming worst/best case simulations can be done, but they are more pessimistic.

To rate the quality of a test method the fault coverage is defined. According to the literature there are different definitions of the fault coverage [15], [16]. The most common one is: The fault coverage ($FC$) of a given test method, is defined as the ratio of faults detected ($FC_d$) over faults simulated ($FC_s$). If faults are weighted by there likelihood $p_i$, then it is:

$$FC = \frac{\sum_{i \in FC_d} p_i}{\sum_{i \in FC_s} p_i}$$
3.3. Test Signal Generation

For digital circuits many ATPG methods have been invented during the 70th and 80th. In the following years these methods were improved and are now used in commercial programs and test architectures. Hence the test methods for analog circuit testing are still in development. Many of the analog test algorithms have strong limitations for their use like linearity or a special class of designs so they are not suitable for industrial use. Even the existence of linear industrial circuit is questionable or the question if they stay linear in case of a fault. In result there is no common accepted and practical solution to test analog circuits in industrial production. Another problem to development analog ATPGs is the lack of a calculation base. For digital circuits it is possible to calculate the test signals by propagating the faults to the inputs and outputs. For analogue circuits this is very difficult because of feedbacks, loops and non-linearity.

The goal of the test signal generation for analog circuits is defined like:

"Determine an optimum test stimulus for an analog circuit that maximizes the detection of all faults from a given fault set." (Figure 3-3)

\[ \text{Injected fault set} = \{F_1, F_2, \ldots, F_m\} \]

\[ v(t) \]

\[ C_g \] \hspace{1cm} \[ C_{F_x} = C_g + \{F_n\} \]

\[ \Psi^{F_x}(t) \]

\[ \text{Difference responses} \]

\[ \int_0^{T_{inj}} \left[ v_{out}^g(t) - v_{out}^{F_x}(t) \right] dt = \Psi^{F_x}; \quad \text{Maximise} \sum_{n=1}^{m} \Psi^{F_n} \]

\[ \max \left[ v_{out}^g(t) - v_{out}^{F_x}(t) \right] = \Psi^{F_x}; \quad \text{Maximise} \sum_{n=1}^{m} \Psi^{F_n} \]

Figure 3-3: Goal of the Test Signal Generation
The merit functional or goal function can be defined in different ways. The general goal function is shown in (3.3.1). The integral of the absolute difference between the good output and the bad output has to be maximized. The IFA can weight the defects by their likelihood. This weight is injected in the merit functional by adding a factor in front of the integral (3.3.2). To calculate a merit functional for a set of faults all merit functionals for each fault are added (3.3.3). Another merit functional is to use the maximum of the absolute difference between the good and the faulty response (3.3.4). The value of the maximum difference also gives a lot of information how good this fault will be measurable. In this work in most cases the maximum goal function is used because the value of the maximum difference gives a good indication if this fault will be detectable.

\begin{align*}
(3.3.1) \quad J_{f_s} &= \int_0^{T_{\text{tot}}} \left| v_{\text{out}}^g - v_{\text{out}}^f \right| dt \\
(3.3.2) \quad J_{f_s}^w &= w_s \int_0^{T_{\text{tot}}} \left| v_{\text{out}}^g - v_{\text{out}}^f \right| dt \\
(3.3.3) \quad J_{f_s}^{w,\text{all}} &= \sum_{n=1}^{k} J_{f_s}^w \\
(3.3.4) \quad J_{f_s} &= \max \left| V_g - V_{f_s} \right|
\end{align*}

The merit functional (3.3.1) is integrated over the simulation time. To compare different simulation times the merit functional need to be normalised. One normalisation is to divide by simulation time.

### 3.4. Theory developed by Bernhard Burdiek

At the Institute of Electromagnetic Theory and Microwave Technique (University of Hannover, Germany), Bernhard Burdiek developed a theory, how to find an optimum test stimulus for a transient test [12]. The goal function of this optimization is to maximize the integral of the absolute difference between the
good and the faulty response for each fault of the fault set. He uses the integral goal function defined in (3.3.3). With that goal function his optimization algorithm tries to find an optimum input signal for a given simulation time. During his studies he ascertained, that the optimal test stimuli has to be a piecewise constant signal called binary signal [17]. He programmed an application called TORAD to do all needed simulations and optimizations.

The work of Bernhard Burdiek shows an automated test signal generation algorithm which calculates for a given fault set of a circuit an optimal transient test signal. In his work the task of calculating a transient test signal, which maximises the number of detected faults from a given fault set, is formulated as an optimal control problem. The goal function of the optimum control problem, which is a measurement of the fault detection capability of the test signal, is based on the integral distance between the good and all faulty test responses. This goal function enables Burdiek to get a solution for the optimum control problem. He could prove the qualitative form of an optimum test signal with the help of optimum control theory methods like the maximum principle of Pontrjagin, which are based on the calculus of variations [18].

One of the main results of his thesis is stated as followed (Quote Bernhard Burdiek [12]):

"Unter der Voraussetzung, dass die Beschreibungsgleichungen der zu testenden Schaltung ein Index-1-System repräsentieren, welches sich durch Variablenreduktion und algebraische Umformungen in ein System der Form:

\[
\begin{pmatrix}
1 & 0 \\
0 & 0 
\end{pmatrix}
\frac{d}{dt} \begin{pmatrix}
\overline{z} \\
\overline{x}
\end{pmatrix} = \Psi, (\overline{z}, \overline{x}, u, t) = \begin{pmatrix}
f_r(\overline{z}, \overline{x}, t) + \overline{B}, u \\
g_r(\overline{z}, \overline{x}, t)
\end{pmatrix}
\]

überführen lässt, also ein System bei dem sämtliche Testsignalquellen affin an die differentiellen Gleichungen angekoppelt sind, handelt es sich bei Lösung des Problems (TP) um ein binäres Testsignal."

Translation:

"Under the precondition that the description equations of the circuit to test are an index-1 system, which can be translated through variable reductions and
algebraic conversion into a system of the form (3.4.1) thus a system in that all sources are coupled affine to the differential equations, the solution of the test problem is a binary signal.”

Bernhard Burdiek transferred the optimal test signal generation to an optimum control problem. Using the calculus of variations and he could prove mathematically that the test signal that can maximise a general analogue circuit has to be a binary signal. This signal has the qualitative form shown in Figure 3-4.

Figure 3-4: Qualitative form of a binary signal

This signal is switching from the maximum voltage to the minimum voltage. The quantitative form is described by the max and min values, the simulation time and the switching times.
4. Comparison of TORAD to PStar simulator

Burdiek implemented his own simulator in TORAD. To validate the results from his TORAD simulator it is compared with the results from the PStar simulator. Two circuits were compared, one circuit is a continuous time band-pass (CTBP) filter and the other one is a continuous time state variable (CTSV) filter. These circuits were implemented by Burdiek in the TORAD simulator. Other circuits could not be implemented in TORAD because there is no documentation about the used language. For the comparison the circuits are applied to TORAD and as well to the PStar. We use the transient test method so we needed to compare the transient simulations. The input signal for the transient simulation is a binary signal because this type of signal is more difficult to simulate due to the infinite number of harmonic frequencies and the edges of the signal. After the simulation we compare the output curves and make a qualitative comparison. Some of the results are shown below.

In the following examples the results of the CTBP (Layout Chapter 5.5) are shown. The broken lines are obtained by the TORAD simulator and the continuous lines are from PStar. Figure 4-1 shows the input signals for the CTBP.
Figure 4-1: Transient simulation of the CTBP filter: Input signals for TORAD and PStar

The next figure (Figure 4-2) is the comparison for the output of Torad and PStar.

Figure 4-2: Transient simulation of the CTBP filter: Output signals for TORAD and PStar

The next two figures are the results from the CTSV. In the first graph (Figure 4-3) is the comparison of the inputs (A_y1 = TORAD, B_VN(IN) = PStar).
Figure 4-3: Transient simulation of the CTSV filter: Input signals for TORAD and PStar

Figure 4-4 shows the two outputs obtained from the simulation with TORAD (A_y3) and PStar (B_VN(HPO)).

According to the outputs of TORAD and PStar the results are comparable. They differ about 0.5 % for the CTBP and 0.07 % for the CTSV. So we could verify
the comparability between the two simulators and we can trust the results of the TORAD simulator.
5. Test Signal Generation with Philips programs

The next step to find the optimal test signal was to find the same optimum input with the optimiser from Philips called Adapt. This optimum signal should be the same as it was calculated by TORAD (See Figure 5-1). A circuit is implemented in TORAD and in Adapt. Both programs try to find an optimum input signal by maximising the merit functional.

![Diagram of TORAD and Adapt optimization process](attachment:figure_5_1.png)

Figure 5-1: Comparison of TORAD and Adapt optimization

Adapt is a general circuit optimizing software made to optimise circuits according to layout and speed and other variables and not programmed for signal optimization. But it should be possible to validate the results of Burdieks theory and to find a way to obtain a good test signal. For that comparison we need to implement the same circuits in both programs. The first circuits are examples from TORAD. They need to be implemented to Adapt. The circuit description is written in PStar language because the optimiser uses the PStar simulator. Adapt itself needs a problem file, which describes how Adapt has to optimise the circuit. In this file we need a model for the input signal, where we define the variables for Adapt (Chapter 5.1). This input model is for adapt the
CUT (circuit under test). The circuit needs to be defined and the injected faults. The goal function (merit functional) combines the circuit with the input function so that Adapt can optimise the input function. Adapt is trying to find a maximum for the merit functional. Figure 5-2 shows an abstract flow of the optimisation using Adapt.

Adapt performs the simulations of the good and the faulty circuits. Then it calculates the merit functional. In the next step Adapt changes the predefined optimisation variables. In our case these variables define the binary input signal and their values at the optimization start are defining the start signal. With this new input signal Adapt redoes all the simulations and calculates again the merit functional. Adapt calculates \( n+3 \) points (\( n \) = dimension of the optimisation problem; number of variables) and fit them in a linear model using for example the uniform design. With that model the next point is approximated and the corresponding simulations are performed. With the increasing number of points
Adapt increases the complexity of the used model up to a quadratic model. As Adapt is using the start values and the next iteration results the start values have a great influence on the calculated models hence on the optimisation results. One part of the Adapt optimisation is shown in Figure 5-3 [19].

![Figure 5-3: Min. merit function: trust-region approach](image)

5.1. Implementation of the binary signal

The implemented input model is defined as a function of pulses to build a binary signal, which is close to the TORAD signal. This model has to have all needed degrees of freedom and constrains to validate the results from Burdiek. The variables and constrains are shown in Table 5-1:

<table>
<thead>
<tr>
<th>Variables</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simulation time</td>
<td>$T_{sim}$</td>
</tr>
<tr>
<td>Max pulse amplitude</td>
<td>$V_{max}$</td>
</tr>
<tr>
<td>Min pulse amplitude</td>
<td>$V_{min}$</td>
</tr>
<tr>
<td>Number of pulses</td>
<td>$N_{pulse}$</td>
</tr>
<tr>
<td>Amplitude of Pulse n</td>
<td>$x_n$</td>
</tr>
<tr>
<td>Duration of pulse n</td>
<td>$t_n$</td>
</tr>
</tbody>
</table>
With this variables and constrains we can formulate a general pulse function shown in Figure 5-4. The simulation time, the maximum and minimum voltages are constrained. The number of pulses, their duration and their amplitude are variables. With this information any arbitrary piecewise constant signal can be defined.

![Diagram of general pulse function](image)

**Figure 5-4: The general pulse function**

The model for the general pulse function is made of single pulses combined to a continuous binary signal. There is a function implemented in PStar, which is called pulse model. The definition of that model is shown in Chapter 9.3. This function is used to construct a model. Each pulse is one half wave of the binary signal.

The binary model is made by fitting many of those pulses together. In that way every single pulse can be defined for itself by changing the values of that pulse (Start time, start value, pulse duration, rise time and fall time, delay, amplitude). The single pulses added together build the pulse model (See Figure 5-5).
The first pulse starts at zero simulation time. The start time of the second pulse is the end time of the previous pulse. The end time is defined through the start time plus the pulse duration. To change the number of pulses, the model consists of many single pulses. The number of pulses can be defined through different ways. The number of pulses is a variable and it has to be defined for every simulation. Or the number of pulses exceeds the simulation time. Only those pulses, which are within the simulation time, have an effect on the simulation. In this way the number of pulses is changed implicit through all other pulses. The variables for this pulse model are the duration time and amplitude of each pulse Figure 5-6.

Figure 5-5: Single pulses added to one pulse model

\[
pulse \text{ duration} = t^1_{\text{duration}} = t^1 \\
pulse \text{ start} = t^1_{\text{start}} = 0 \\
pulse \text{ end} = t^1_{\text{end}} = t^1_{\text{start}} + t^1 \\
pulse \text{ amplitude} = x^1 = V_{\text{max}}
\]

\[
\begin{align*}
t^2_{\text{duration}} &= t^2 \\
t^2_{\text{start}} &= t^1_{\text{end}} \\
t^2_{\text{end}} &= t^1_{\text{end}} + t^2 \\
x^2 &= V_{\text{min}}
\end{align*}
\]

\[
\begin{align*}
t^3_{\text{duration}} &= t^3 \\
t^3_{\text{start}} &= t^2_{\text{end}} \\
t^3_{\text{end}} &= t^1_{\text{end}} + t^3 \\
x^3 &= V_{\text{max}}
\end{align*}
\]

Pulse model Containing Pulse A, B and C
We tested different kinds of implementations for the optimization with Adapt. For the first tests the pulse model had all degrees of freedom. The values of each
pulse could be changed by Adapt to find an optimum. With that many freedoms Adapt was not able to find any kind of optimum so we had to make some further restrictions. We figured out, that the optimum signal will always be a signal changing between $V_{\text{max}}$ and $V_{\text{min}}$ so the next pulse model had fixed voltage values.

The problem with the pulse model of Figure 5-6 is that changing the duration for one pulse will affect all following pulses. For the optimiser this effect makes it difficult to build an accurate model. To avoid this effect, a different pulse model was programmed. Within this model the switching times, when the binary signal changes from $V_{\text{max}}$ to $V_{\text{min}}$ or from $V_{\text{min}}$ to $V_{\text{max}}$, are defined. In this way Adapt only changes two pulses with one variable Figure 5-8.

Figure 5-8: Pulse model with steps
Figure 5-9: Example for the implementation of the pulse model (Switch times)

Another model was also tested (Figure 5-10), where the time steps of the pulses are fixed and the amplitude can be changed to any value between $V_{\text{max}}$ and $V_{\text{min}}$. To have a good resolution you need to have many pulses but Adapt is restricted to around 200 variables. In this model the number of pulses defines the maximal resolution.
This model is a good and general approach but it is too complex. You need too many single pulses to have a satisfying resolution which leads to a high number of variables. In addition to that, Adapt does not have discrete values for the variables so that the pulse amplitude cannot be constrained to $V_{\text{max}}$ and $V_{\text{min}}$. So the known assumption, that the pulses can only be $V_{\text{max}}$ or $V_{\text{min}}$, cannot be taken in account. As well those pulses must not have a rise and a fall time, which is not very realistic.

To have a more realistic binary signal the pulse function was changed to the $\text{sinsq}$ function. The definition of that function is shown in Figure 5-11 and an example waveform is shown in Figure 5-12.
5.2. Optimisation with Adapt using a random start signal

The first simulations were to validate the test signals obtained by the optimisation of TORAD. The switching times of the binary signal of TORAD are applied to Adapt, which now tries to optimise this signal. If it is the optimal test signal adapt will not find a better stimulus. Only slightly changes should occur and they can be explained from the results of the comparison of PStar with TORAD. In addition to that we search for an optimum test signal for a set of
faults. So TORAD and Adapt have to make a kind of trade-off. By changing the input signal, one fault maybe better detectable but another one is getting less detectable. So the algorithm has to decide which fault is having the higher priority. Indeed Adapt could not improve the input signal. It was only changing some values slightly but was not really changing the signal and finding a better solution.

The next step to approach the optimization with Adapt was to find the optimum input signal from any start signal. Adapt was not able to find the optimum signal with a random start signal so the optimization was extended. Different start signals were applied to Adapt to figure out if this helps to find the same input signal as TORAD found it. The changed flow is shown in Figure 5-13.

![Flowchart](image)

Figure 5-13: Extended flow of the optimization with Adapt

### 5.3. Different start signals

The Figure 5-14 shows the different approaches to find a start signal. Adapt should be able to find the same solution (for the given fault set) as TORAD
does. The different approaches are described in this chapter. Some of the results are presented in chapter 5.5.

![Start Signals Diagram]

Figure 5-14: Different start signals for the optimization with Adapt

The initial try was just one step because this signal is good enough for TORAD to find a first solution of the optimum control problem. Taking just one step as start signal for Adapt did not give the solution. Changing the voltage did not change the optimization results of Adapt. Indeed the optimised input signal is a binary signal switching between $V_{\text{max}}$ and $V_{\text{min}}$. The input signal gained from the Adapt optimization is not as good as the signal gain from TORAD. In the next simulations the start signal was changed and the different implementations for the binary model were tested. To achieve a good approximation to the TORAD signal the start signal should be close to the optimum test signal. During the simulations we could observe that an equidistant binary signal seems to be a good start signal. This signal can be found by changing the number of pulses within the simulation time.

The simulations were done on a Continuous Time Band Pass Filter and a Continuous Time State Variable Filter, which had been tested with the TORAD program by Burdiek before (5.5). The Philips industrial circuit was a Low Noise Amplifier (LNA). This circuit has been tested with the VDD-Ramp Method [20]. The results of the LNA using binary signals are presented in 5.6.
5.4. Intermediate Results

The results of the TORAD optimisation could be validated. Adapt could not find a signal optimise the TORAD signal further. We also found out that Adapt cannot find the optimal test signal for a set of faults without a suitable start signal. After numerous simulations with different start signals we obtained some results about the needed start signal. The start signal has to be very close to the optimum solution of TORAD. A good approximation of this start signal is an equidistant binary signal. Even the equidistant binary signal itself seems to be a good test signal for detecting single faults. To find the right quantitative form of the equidistant binary signal the number of pulses within the simulation time is changed. The number of pulses can be related to the frequency of the equidistant binary signal. The number of pulses at which the merit functional is maximised is the start signal we were trying to find.

These results led to the idea to use the equidistant binary signal itself as test signal. In fact the problem is now shifted to find the right number of pulses and the right simulation time to discriminate the faulty output to the good one. Changing the number of pulses within a fixed simulation time of the test signal is the same as changing the frequency of the equidistant binary signal. The next goal is to figure out which frequency of the binary signal maximises the difference between a good and a faulty circuit response.

5.5. Example: Continuous Time Band Pass Filter

This example is a continuous time band pass filter containing three operational amplifiers with resistors and capacitors in the feedback loops. The schematic is shown in Figure 5-15. It has one input node called IN and one output node called OUT. The maximum voltage that can be applied to the input is ±5 volts.
Optimal Test Signal Generation for Analog Test
Diploma Work of Tobias Völkel

The characteristic frequency response for the AC simulation is shown in Figure 5-16. For this simulation the input amplitude was 1 V at the IN node. The centre pass frequency of this band pass circuit is at 10 kHz.

\[ V_{\text{OUT}} \text{[V]} \]

\[ f \text{[Hz]} \]

In this example we show the results for one fault. The resistor R6 is faulty and has a deviation of 10%. Figure 5-17 shows the input signals of the TORAD optimisation and the optimisation with Adapt using the optimised TORAD signal.
as start signal. The continuous line is the original TORAD input signal and the broken line is the result from the optimization with Adapt. As expected they do not differ that much.

Figure 5-17: Input signals of TORAD and Adapt (TORAD start signal)

Figure 5-18 shows two difference outputs $|v_{\text{out}}^G - v_{\text{out}}^I|$ of the CTBP filter. The broken line is the response of the further optimised test signal and the continuous line is the difference response of the circuits when applying the original TORAD test signal. The variation can be explained with the results from the previous test with the different simulators in chapter 4. In fact the simulators are not equal so the results are as well not the same but the variations are marginal.
After we could prove the results of TORAD we tried to obtain the same result with Adapt by applying a random start signal. The simulation time is fixed to the time we know from the TORAD optimisation. The result of the optimisation with a single pulse as start signal is shown in Figure 5-19. This result is not as expected. Only two pulses build the input signal.
Comparing the difference output of this signal with the difference output of the TORAD test signal (Figure 5-20) gives a better view how far away this solution is from the optimum.

Figure 5-20: Comparison ($|\Delta out|$): TORAD and Adapt (one pulse)

Now a new idea was tested. The start signal for the optimisation with Adapt was in the next example five equidistant pulses (Figure 5-21). The new input signal is much closer to the reference TORAD signal.

Figure 5-21: Comparison (Input): TORAD and Adapt (five pulses)
In Figure 5-22 we compare the quality of these test signals by looking at the difference outputs. According to the results equidistant pulses seems to be good start signals for the optimisation.

For a better understanding of the equidistant pulse signal it is compared with the TORAD signal in Figure 5-23.
Figure 5-24 compares the difference output of the equidistant binary signal with the TORAD signal. The two outputs are very close so it seems that the equidistant binary signal is a suitable signal.

![Graph comparing two signals](image)

**Figure 5-24: Comparison ($|\Delta \text{out}|$): TORAD and PStar (five pulses)**

<table>
<thead>
<tr>
<th>Input Signal</th>
<th>Merit functional</th>
</tr>
</thead>
<tbody>
<tr>
<td>Torad</td>
<td>0.170012</td>
</tr>
<tr>
<td>Adapt (Torad)</td>
<td>0.173571</td>
</tr>
<tr>
<td>Adapt (one pulse)</td>
<td>0.119271</td>
</tr>
<tr>
<td>Adapt (5 pulses)</td>
<td>0.173243</td>
</tr>
<tr>
<td>5 pulses</td>
<td>0.166375</td>
</tr>
</tbody>
</table>

Using a random start signal for the optimisation with Adapt does not give a good solution. Only if the applied start signal is close to the optimum. Adapt can calculate an optimum test signal. The equidistant binary signal seems to be a good solution for the task to find this start signal. Chapter 6 deals with this problem.
5.6. Example: Low-Noise-Amplifier

To prove the theory that an equidistant binary signal can be a good test signal a Philips industrial circuit was tested and analysed. This circuit is a Low-Noise-Amplifier (LNA) [21], [22], which is typically the first stage of a receiver. Its main function is to provide enough gain to overcome the noise of subsequent stages like a mixer. This amplifier is a differential one and operating at the frequency interval of [2.4GHz, 2.5GHz]. This operating frequency is too high to use common digital test equipment so we tried to find a lower frequency up to a maximum of 200MHz. For this example no knowledge form previous TORAD tests were available so we needed to determine the right simulation time as well as the number of pulses within this simulation time. This flow is shown in Figure 5-25.

![Flow diagram for LNA testing](image)

Figure 5-25: Work flow for the LNA

From the previous work on the LNA using the VDD-Ramp method an extracted fault list with 146 faults exists. 11 faults of that set of faults could not be detected by VDD-Ramp. Those faults were injected to figure out if the transient dynamic test with an equidistant binary test signal can detect those faults.
As shown in Figure 5-26 all faults could be detected by the transient test using an equidistant binary test signal. In addition to the binary signal the VDD voltage was changed to examine the impact. The highest voltage was the best to test the circuits. Different VDDs could not change the detectability like it is for the VDD-Ramp method.

![Diagram showing the test flow for the low noise amplifier](image)

Figure 5-26: Test flow for the low noise amplifier

These simulations could prove the idea that an equidistant pulse can be good enough to test detect the faults in this circuit. To find the right simulation time and the number of pulses was very difficult and time consuming. In addition to that the calculation time was very high for the binary signal. In fact we only looked at windows in the solution space. So we cannot assume that those results are the optimal or best signals. For that reason some of the results are only shown in the appendix (9.4). Some more work has been done on the LNA later and is described in more detail in chapter 0.
6. Equidistant binary signal

According to the results from the previous chapter the equidistant binary signal itself is a good test signal. As shown in Figure 6-1 the equidistant binary is another approach for the test signal. It is suitable to do further optimisations with Adapt.

![Flow-chart of the optimization approaches](image)

Figure 6-1: Flow-chart of the optimization approaches

As a result of the previous work we figured out, that the observability of a fault is frequency dependent. Multi-frequency techniques already have been employed to model the faulty behaviour of an analogue circuit [23], [24]. The next goal is to figure out the right frequency, which maximises the difference of a faulty to the good circuit response. For that reason we take a closer look at the binary signal. Making a Fourier analyse shows the frequency content of piecewise constant signal. Disregarding the switching time the Fourier analyses looks like (6.1.1).
\[ y = \frac{4a}{\pi} \left( \sin x + \frac{\sin 3x}{3} + \frac{\sin 5x}{5} + \ldots \right) \]

\[ = \frac{4a}{\pi} \sum_{k=0}^{\infty} \frac{\sin((2k+1)x)}{2k+1} \quad \text{for } x \neq \pm \pi, \pm 3\pi, \ldots \]

The equidistant binary signal contains the main frequency \( f \) and has an infinite number of sub frequencies: \( 3f, 5f, 7f, \ldots \)

Comparing this signal to a simple sine wave the commonness is the main frequency. So it should be possible to find the right binary frequency with a simple sine wave. The advantage of the sine wave is the faster computation time for the simulation. Reducing the problem to find an optimal input signal to the search for the right frequency for every fault reduces the complexity of the problem.

In addition to the idea of finding the frequency the problem of the unknown simulation time can be solved as well. In the transient test only the settling time is taken in account for the test. With that information the simulation time can be fixed to a number of periods for each frequency. The used numbers of periods in this work were between 4 and 10 periods. Determining the right number of periods depends on the circuit behaviour. If it is a very fast circuit with a small settling time the needed numbers of periods are just a couple. If the circuit has different stages the numbers of periods are going to be bigger to obtain a good circuit response.

With this knowledge about the circuit behaviour we can assume that every faulty circuit has at least one frequency at which the difference to the good circuit response can be maximised. With different industrial circuits from Philips we could validate this theory. One circuit was the LNA and in addition to that circuit some simulations were made on a folded operational amplifier.

To obtain the maximising frequency for each fault an AC simulation was tested as well but the results from these simulations cannot be used to get the frequency [25]. This can be explained by looking at the differences between an AC simulation and a transient simulation. Both simulations calculate in the first step the operation point by performing a DC analysis. From this initial state the
transient analyse is started. For the AC analyse a linear small signal model is calculated. This model is only valid for very small signals. To have a good discrimination the amplitude of the input signals should be the maximum amplitude that can be applied to the circuit. Hence the linearized model is not anymore valid. These results we could prove in practical simulations.

6.2. Frequency sweep for the equidistant binary signal

As discussed above, every fault $F_n$ in an analogue circuit has at least one frequency $f_n$ at which the difference between the faulty and the good circuit response is maximised. Using a sinusoidal signal (amplitude $V$, the maximum input voltage) as stimuli is a fast and cheap technique, to perform the time domain simulations to determine the frequency $f_n$ that maximises the difference between the good and the faulty circuit. At every iteration one single fault $F_n$ from the given fault set $F$, is used to determine the maximising frequency $f_n$. Assigning this frequency $f_n$ to the binary signal will maximise the fault detection capability of the given fault $F_n$. Hence a binary signal that switch states between $\pm V$, at every $1/2f$ seconds is the optimal test vector to detect fault $F_n$ and is one test vector element of test set $T$. This procedure is repeated until all faults from the fault set are simulated. The above procedure is shown in Figure 6-2.

We have chosen sinusoidal test signal to iteratively determine the frequency at which the observability of the fault is maximum, due to the fact that a single sinusoidal wave has only one fundamental frequency and hence sinusoidal time domain simulations are faster than any other periodic waveform.
6.3. Example: Continuous Time Band Pass Filter

Two components R6 and C1 (Ref: Figure 5-15) have been chosen as the faulty components with 200% and 60% deviations from their respective nominal values. Two faulty circuits with faults $F_1$, $F_2$ have been constructed by injecting the above faults to the fault free circuit.

<table>
<thead>
<tr>
<th>Fault</th>
<th>Component</th>
<th>Nominal Value</th>
<th>Faulty Value</th>
<th>Deviation in % from nominal</th>
</tr>
</thead>
<tbody>
<tr>
<td>$F_1$</td>
<td>R6</td>
<td>10k Ω</td>
<td>20k Ω</td>
<td>200%</td>
</tr>
<tr>
<td>$F_2$</td>
<td>C1</td>
<td>1.59n F</td>
<td>0.954n F</td>
<td>60%</td>
</tr>
</tbody>
</table>

Figure 6-3 shows the simulation results in time domain, the absolute difference of the responses $\left| v_{out}^F(t) - v_{out}^{F_1}(t) \right|$ for the 2 faulty circuits, with sinusoidal stimuli.
of amplitude 5 Volts over a frequency range from 4 kHz to 16 kHz with a step size of 50 Hz.

![Graph showing difference between responses]

Figure 6-3: Maximum difference of the sinusoidal transient simulation

For the first fault $F_1$, the frequency $f_1 = 7.08 \, \text{kHz}$ maximises the difference to the good circuit. $f_2 = 11.56 \, \text{kHz}$ maximises the discrimination of the second fault $F_2$.

The frequencies $f_1$ and $f_2$ determined from the time domain sinusoidal simulations correspond to the frequencies of the binary signals that maximizes the difference between the response of the good circuit and the faulty circuits with the faults $F_1$ and $F_2$ respectively.

For comparative purposes Figure 6 shows the simulation results in time domain, the absolute difference of the responses $\|v_{out}^g(t) - v_{out}^F(t)\|$, with a binary test signal $[\pm 5]$ over a frequency range from 4 kHz to 16 kHz.
Table 6-2 compares the computation time of the binary signal with the sinusoidal signal over the frequency range from 4 kHz to 16 kHz with step size of 50 Hz. The calculation time is the total time including the good circuit and the two faulty circuits. The table indicates that we have determined the optimal frequency of the binary signals that can maximize the discrimination of the good circuit and the faulty circuits through a cheaper sinusoidal simulation. The speed up that we have obtained in some other experimental examples is about 4 to 8 times.

Table 6-2: Calculation Time for the simulations

<table>
<thead>
<tr>
<th></th>
<th>Binary simulation</th>
<th>Sinusoidal simulation</th>
<th>Speed-up</th>
</tr>
</thead>
<tbody>
<tr>
<td>Calculation Time</td>
<td>117.08 s</td>
<td>14.50 s</td>
<td>8x</td>
</tr>
</tbody>
</table>

Table 6-3 shows the test set for the faults $F_1$ and $F_2$. The quantitative description of each test vector (binary signal) is also noted. With the frequency ($f$) determined from a sinusoidal transient (time domain) simulation the switching times for the binary signal can easily be computed as ($T_{sw} = \frac{1}{2f}$).
Table 6-3: Quantitative description of the binary signals

<table>
<thead>
<tr>
<th>Fault Number</th>
<th>Test Frequency</th>
<th>Switching Times (µ sec)</th>
<th>Simulation Time (µ sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$F_1$</td>
<td>7.08k Hz</td>
<td>70.62</td>
<td>564.96</td>
</tr>
<tr>
<td>$F_2$</td>
<td>11.56k Hz</td>
<td>43.25</td>
<td>346.0</td>
</tr>
</tbody>
</table>

The corresponding binary test signal for fault $F_1$ is shown in Figure 6-5 and for $F_2$ it is Figure 6-6.

Figure 6-5: Binary signal with $f_1 = 7.08\ kHz$
The input signal for $f_i$ with the corresponding responses for the good and the faulty circuit $F_i$ is shown in Figure 6-7. Figure 6-9 shows the outputs for the fault $F_2$ with the test signal at $f_2$. To compare the two test signals $V_1$ and $V_2$ Figure 6-8 shows the corresponding absolute difference of the responses $\left| v_{out}^g(t) - v_{out}^f(t) \right|$ for fault $F_1$ and Figure 6-10 shows the similar graphs for $F_2$. It can be seen from these figures that the observation of fault $F_1$ is maximized with application of test vector $V_1$ and for fault $F_2$ the test vector $V_2$ maximises the difference.
Figure 6-7: Input at frequency $f_1$ and output for the good circuit and the faulty circuit $F_1$.

Figure 6-8: Difference between the good circuit and fault $F_1$ for frequency $f_1$ and $f_2$. 

Optimal Test Signal Generation for Analog Test
Diploma Work of Tobias Völkel
- 47 -
6.4. Simulation Time

In Table 6-4 the different simulation times are compared. For each circuit the number of simulated circuits, the frequency range and the number of periods are the same.
Table 6-4: Calculation Time for the simulations

<table>
<thead>
<tr>
<th>Circuit</th>
<th>Binary simulation</th>
<th>Sinusoidal simulation</th>
<th>Speed-up</th>
</tr>
</thead>
<tbody>
<tr>
<td>CTBP</td>
<td>117.08 s</td>
<td>14.50 s</td>
<td>8x</td>
</tr>
<tr>
<td>Folded OpAmp</td>
<td>886.46 s</td>
<td>177.28 s</td>
<td>5x</td>
</tr>
<tr>
<td>LNA</td>
<td>7022.19</td>
<td>1913.20 s</td>
<td>3.7x</td>
</tr>
</tbody>
</table>

In summary the speed-up is between 4 and 8 time. The LNA is a bit special, because it had a lot of simulation problems.

6.5. Example: Low-Noise-Amplifier

The LNA was as well tested to prove the frequency dependency of a fault. The circuit is described in chapter 5.6. As an example for the frequency sweep the fault $F_{33}$ (a bridge between the nodes SUBBIAS and INP) and fault $F_{50}$ (a bridge between the nodes SUBBIAS and INN) are shown in Figure 6-11 and Figure 6-12. The bridges have a resistance of 100 $\Omega$. The Input is applied at the INP node and the circuit response is measured at the OUTFP node. The graphs show the output of the merit functional over the frequency from 10 MHz to 200 MHz. In these two examples the broken lines are from the equidistant binary input signals and the continuous line represents the sinusoidal input signals.
The test frequency of the sine wave sweep is nearly the same as it is for the equidistant binary signals. Only a small shift can happen due to the multiple frequencies of the binary signal. Looking at the numbers we get a much better test signal by just applying the frequency from the sine wave to the binary pulse.
6.6. Example: Folded Operational Amplifier

Another example is the differential amplifier used in a sample-hold circuit. To test this stage of the circuits we implemented a feedback loop and grounded one path of the differential amplifier. Now having a single ended amplifier the input sine wave was applied to the input. To compare the results from the simulation with different frequencies the results are shown in Figure 6-13. The broken line is the merit functional over the frequency from 1 MHz to 250 MHz for the sinusoidal input signal and the continuous line is the corresponding graph for the equidistant binary signal.

The frequency from the sinusoidal input signal is again transferable to the frequency of the equidistant binary signal. The binary signal discriminates the fault much more than the sinusoidal signal.
7. Summary and Conclusions

This work was about finding an optimum test signal for transient test of analogue circuits. In the start we took the optimisation algorithm of Burdieks dissertation and validated his results. In fact the optimal signal to maximise the difference between the good and the faulty circuit response is a piecewise constant signal called binary signal. We developed a work flow to obtain similar signals using Philips tools. For that flow a suitable start signal as a first approximation of the optimisation problem has to be found. This signal is an equidistant binary signal. Applying this signal to Adapt, this optimiser can find a solution of the optimisation problem.

During the work we achieved the results that the equidistant binary signal itself is a good test signal. Every fault has at least one frequency at which the discrimination of the faulty response to the good one can be maximised. Using a sinusoidal transient signal to determine the frequency for each fault reduces the simulation time. A binary signal (piecewise constant signal) that switches states twice the frequency of above sinusoidal frequency is the optimal test signal that maximizes the detection of a fault. The frequency of the test signal is often lower than the operating frequency of the circuit, which puts fewer demands on the production test. The signature responses from an arbitrary device can be used to classify the device using any signature analyser method or through predetermined tolerance bands.

Even the sinusoidal signal can have good results but the corresponding equidistant binary signal discriminates the fault better. The next benefit of the binary signal is, that it is much easier to apply on a digital tester because they are designed to apply a binary test signal. No special hardware like expensive function generators are needed for the test with binary signals.
**8. Future Work**

With this new approach to find an optimal test signal for every fault the basis for new methods has been created. At first there is the possibility to program an analogue ATPG. This can be done by using a sinusoidal transient simulation to calculate the test frequency for each fault. This frequency can be applied by using the equidistant binary signal to a digital tester. The evaluation of the test output can happen in various ways. The easiest and fastest evaluation would be to build the integral and compare it to a tolerance band. This would be a kind of pass and fail test. Or the output can be sampled at predetermined time samples and again compared to a tolerance band. This test would give more information about the type of the fault and is one of the first steps to perform a fault classification. This classification can be done by a signature analysis. To do all these evaluations it should be investigated how the circuit output behaves according to process variations.

If we try to improve the test generation it should be investigated if Adapt can be changed to find the optimal test frequency for each fault. In this work the used number of periods were chosen manual. This could be another task to define rules and constrains that this number can be calculated as well. One approach could be to use the PSS (Periodic Steady State) function of PStar. It could be possible that the number of periods needed to reach the steady state is the number of periods that should be used for the transient dynamic test with binary signals.

Test pattern compression techniques can be used as well for this test method. From the general work flow we get for every fault one frequency. Some of the faults will have the same test frequency. It should be possible to build test groups that the number of different frequencies can be reduced. As well it is possible that the frequencies for the most difficult faults are used for testing the circuit and the other faults will be identified as well, even if the test signal is not optimal for them. The most interesting solution would be to use an algorithm that can calculate a binary signal containing all the test frequencies. This can be
done for example by pulse-width-modulation or using other more advanced algorithms like the SB-ZePoC [26]. In that way it should be possible to generate one or maybe just a couple of test signals that can test a set of faults.

Adding to the circuit a binary signal generator and predefine the frequencies could be a suitable approach to build a BIST. The binary signal can be generated by a digital circuit. Then the output could be integrated and compared with a reference value. This idea of a BIST can be combined with the idea of calculating a binary signal with multiple frequencies.

After some more investigations with simulations it should be tested on a real circuit with a binary tester. For some circuits the responses obtained from the simulations did not appear valid and they were not taken in account for this work. As an example the LNA started to oscillate for some frequencies of the binary pulse but not using the sinusoidal signal. But there is no knowledge about the behaviour of the LNA for low frequencies. So maybe the simulation results are false or the LNA really behaves like this. As a conclusion of this observation circuits should be tested with binary signals in the development too if they are going to be tested with the binary signal.
9. Appendix

9.1. Directory Structure

The main directory is “nlv12752\work”. The work directory has the following subdirectories.

The “documents” folder contains all data that was written by myself and used for this project. In the “IP_Proposal” folder all versions of the IP proposal
documents are stored. The other directories are self-explanatory. The folder “misc” contains files that have no other place to be.

The different circuit directories have the subfolders shown below.

```
  “circuit”
   |       adapt
   |       pstar
   |       sources
   |       torad
```

For the “pstar” folder there is another subdivision:

```
  pstar
   |       ac
   |       pulse_sweep
   |       sine_sweep
   |       sources
   |       misc
```

The “AC” folder includes all AC simulations. In the folders “pulse_sweep” and “sine_sweep” are the source files and results from the simulations with the sine wave and the binary signal. For some circuits there is also a “torad” directory in which all files concerning the comparison of PStar to Adapt can be found.

Some extra subfolders are within the “pstar” and “adapt” folders. They are from the previous work I did on those circuits.


9.2. Scripts

I wrote a couple of scripts to automate the use of the different methods. All scripts work the same. In a text file is a tab separated table. All information needed for the execution of the script are written inside. The use of the scripts are described in the following paragraphs.

1. single_pulse_model.pl

This script writes a file with a pulse model defined according to the information in Table 9-1:

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pulse_Model_Filename</td>
<td>pulse_model.net</td>
<td>This is the filename of the pulse model. Path information can be added as well.</td>
</tr>
<tr>
<td>Pulse_Type</td>
<td>sinsq</td>
<td>Possible values are sinsq or pulse</td>
</tr>
<tr>
<td>Pulse_Frequency</td>
<td>100k</td>
<td>The frequency used for the pulses</td>
</tr>
<tr>
<td>Puls_Number</td>
<td>20</td>
<td>The number of single pulses the model is made of</td>
</tr>
<tr>
<td>Pulse_VMax</td>
<td>0.8</td>
<td>Maximum voltage of the pulse</td>
</tr>
<tr>
<td>Pulse_VMax_Rise</td>
<td>0.02n</td>
<td>Rise time of the max voltage pulses</td>
</tr>
<tr>
<td>Pulse_VMax_Fall</td>
<td>0.02n</td>
<td>Fall time of the max voltage pulses</td>
</tr>
<tr>
<td>Pulse_Min</td>
<td>-0.8</td>
<td>Minimum voltage of the pulse</td>
</tr>
<tr>
<td>Pulse_Min_Rise</td>
<td>0.02n</td>
<td>Rise time of the min voltage pulses</td>
</tr>
<tr>
<td>Pulse_Min_Fall</td>
<td>0.02n</td>
<td>Fall time of the min voltage pulses</td>
</tr>
</tbody>
</table>

Executing the perl script will generate a file with a single pulse model as defined in the table.

The next two scripts generate a PStar file to make a frequency sweep with a sine wave (in the first script) and with the equidistant binary signal (in the second script). For both of them all needed information has to be in the table. Those tables are described below.

2. tb_sine_sweep.pl
Table 9-2: tb_sine_sweep_table.txt

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>TB_Filename</td>
<td>tb_sine_sweep.pstar</td>
<td>This is the filename of the PStar file. Path information can be added as well.</td>
</tr>
<tr>
<td>Circuit_Good_Filename</td>
<td>../circuit/good.net</td>
<td>Relative path and filename of the good circuit and setup</td>
</tr>
<tr>
<td>Circuit_Good_Modul_Name</td>
<td>good</td>
<td>Name of the model for the good circuit</td>
</tr>
<tr>
<td>Circuit_Bad_Filename</td>
<td>../circuit/bad.net</td>
<td>Relative path and filename of the bad circuits</td>
</tr>
<tr>
<td>Circuit_Bad_Modul_Name</td>
<td>bad</td>
<td>Name of the model for the bad circuits</td>
</tr>
<tr>
<td>Circuit_Bad_No</td>
<td>2</td>
<td>Number of bad models</td>
</tr>
<tr>
<td>No_of_Sine_IN</td>
<td>2</td>
<td>Number of inputs the circuits have</td>
</tr>
<tr>
<td>Circuit_No_OUT</td>
<td>2</td>
<td>Number of outputs the circuits have</td>
</tr>
<tr>
<td>Amplitude_of_Sine</td>
<td>0.4</td>
<td>Amplitude of the sine wave</td>
</tr>
<tr>
<td>No_of_Periods</td>
<td>4</td>
<td>Number of periods used for the simulation</td>
</tr>
<tr>
<td>Phase_Shift_Sine</td>
<td>0, 180,</td>
<td>List where the different phase shifts of the inputs can be defined. [0…360]</td>
</tr>
<tr>
<td>Start_Freq</td>
<td>100k</td>
<td>Start frequency of the sweep</td>
</tr>
<tr>
<td>End_Freq</td>
<td>100mg</td>
<td>End frequency of the sweep</td>
</tr>
<tr>
<td>Freq_Mode</td>
<td>gn</td>
<td>The mode for the steps (an, as, gn, gs, en)</td>
</tr>
<tr>
<td>Mode_Steps</td>
<td>10</td>
<td>The number for the steps (number or size)</td>
</tr>
</tbody>
</table>

3. tb_pulse_sweep.pl

Table 9-3: tb_pulse_sweep_table.txt

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>TB_Filename</td>
<td>tb_pulse_sweep.pstar</td>
<td>This is the filename of the PStar file. Path information can be added as well.</td>
</tr>
<tr>
<td>Pulse_Filename</td>
<td>pulse_model.net</td>
<td>This is the filename for the generated pulse model.</td>
</tr>
<tr>
<td>Circuit_Good_Filename</td>
<td>../circuit/good.net</td>
<td>Relative path and filename of the good circuit and setup</td>
</tr>
<tr>
<td>Circuit_Good_Modul_Name</td>
<td>good</td>
<td>Name of the model for the good circuit</td>
</tr>
<tr>
<td>Circuit_Bad_Filename</td>
<td>../circuit/bad.net</td>
<td>Relative path and filename of the bad circuits</td>
</tr>
<tr>
<td>Circuit_Bad_Modul_Name</td>
<td>bad</td>
<td>Name of the model for the bad circuits</td>
</tr>
<tr>
<td>Circuit_Bad_No</td>
<td>2</td>
<td>Number of bad models</td>
</tr>
</tbody>
</table>

Optimal Test Signal Generation for Analog Test
Diploma Work of Tobias Völkel
- 58 -
<table>
<thead>
<tr>
<th><strong>No_of_Pulse_IN</strong></th>
<th>2</th>
<th>Number of inputs the circuits have</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Circuit_No_OUT</strong></td>
<td>2</td>
<td>Number of outputs the circuits have</td>
</tr>
<tr>
<td><strong>Amplitude_of_Pulse</strong></td>
<td>0.8</td>
<td>Amplitude of the Input</td>
</tr>
<tr>
<td><strong>Rise_Time</strong></td>
<td>0.2n</td>
<td></td>
</tr>
<tr>
<td><strong>Fall_Time</strong></td>
<td>0.2n</td>
<td></td>
</tr>
<tr>
<td><strong>No_of_Periods</strong></td>
<td>6</td>
<td>Number of periods used for the simulation</td>
</tr>
<tr>
<td><strong>Phase_Shift_Pulse</strong></td>
<td>90, 270,</td>
<td>List where the different phase shifts of the inputs can be defined. [0…360]</td>
</tr>
<tr>
<td><strong>Start_Freq</strong></td>
<td>200k</td>
<td>Start frequency of the sweep</td>
</tr>
<tr>
<td><strong>End_Freq</strong></td>
<td>200mg</td>
<td>End frequency of the sweep</td>
</tr>
<tr>
<td><strong>Freq_Mode</strong></td>
<td>gn</td>
<td>The mode for the steps (an, as, gn, gs, en)</td>
</tr>
<tr>
<td><strong>Mode_Steps</strong></td>
<td>10</td>
<td>The number for the steps (number or size)</td>
</tr>
</tbody>
</table>

4. Circuits

To use the scripts the circuits have to be defined in a special way. I tried to make it as general as possible. The two files (the good and the bad files) are included so they should have the normal PStar syntax. The interface for the test bench is made by using models. The good and the bad circuits should be implemented in a model defining the in and outputs. First comes the inputs and then the outputs. To any input will be a source connected. If you do not want to use an input, don’t put it on the interface of the model. Any additional changed can be done inside the models. Only the in and outputs have to be defined always the same. The model name for the good circuit can be defined in the table as well as for the bad circuits. But there can be n number of bad circuit models. The all need to have the same name followed by an increasing number. For example: bad1, bad2, bad3, badn. These are the only restrictions to the circuits. Example circuits are in the circuits directory of the scripts.
9.3. Definition of the PStar pulse model

PULSE function

- **Purpose**
  - To assign a pulse waveform to circuit components.

- **Syntax**
  - `pulse (v1, v2, tds, tr, tw, tf, [period])`

  The options are defined below:

  - `<v1>`: First value.
  - `<v2>`: Second value.
  - `<tds>`: Delay time.
  - `<tr>`: Rise time.
  - `<tw>`: Width of pulse.
  - `<tf>`: Fall time.
  - `<period>`: Time between the start of each pulse.

  These parameters are not allowed to be `<EV expr>` which can depend on independent parameters.

![Pulse waveform diagram](image)

*Figure 23: The pulse function.*

- **Usage and notes**
  - **General**
    - The `pulse` statement is typically used to assign a pulse waveform to sources in a transient analysis. During an (implicit) DC analysis, the source has the value `<v1>`. A negative pulse is defined if `<v2>` is smaller than `<v1>`, and `<tr>` then defines the fall time.

  - If `<period>` is not defined, only one pulse is generated.

  - A squared sine wave is also available. See `SIN function on page 405` for more details.

- **Example**

  ```
  o_2+pulse (0.5, 0.3ml, 1ml, 10ml, 1ml, 15ml);
  transient;
  t=auto(0, 5*period);
  o_1+sin(0, 5, 0, period/10, 4*period/10, period/10, period);
  period=1ml, 5ml, 10ml;
  ...
  ```
9.4. Results of the Low-Noise-Amplifier

On the following pages some of the results from the first simulations with the LNA are shown. The simulation time is fixed to 100 ns. The fault location is written on the top of each graph. The x-axis is the number of pulses within the simulation time. The z-axis shows the applied VDD voltage from 0 to \( V_{DD_{max}} = 1.96 \text{ V} \). The y-axis is the maximum difference between the good and the faulty output. For some faults the maximum is shown but for other faults you can get an image of how detectable the fault will be. To understand the number of pulses better Table 9-4 shows the corresponding frequencies.

Table 9-4: Corresponding frequency to the number of pulses

<table>
<thead>
<tr>
<th>Number of Pulses</th>
<th>Frequency [MHz]</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>4</td>
<td>20</td>
</tr>
<tr>
<td>6</td>
<td>30</td>
</tr>
<tr>
<td>8</td>
<td>40</td>
</tr>
<tr>
<td>10</td>
<td>50</td>
</tr>
<tr>
<td>12</td>
<td>60</td>
</tr>
<tr>
<td>14</td>
<td>70</td>
</tr>
<tr>
<td>16</td>
<td>80</td>
</tr>
<tr>
<td>18</td>
<td>90</td>
</tr>
<tr>
<td>20</td>
<td>100</td>
</tr>
<tr>
<td>22</td>
<td>110</td>
</tr>
<tr>
<td>24</td>
<td>120</td>
</tr>
</tbody>
</table>
Brg_0 (VB, N95) + (N95, N68) *** NOT DETECTED BY IDD ***
(Brg = 100 Ohm, SimTime = 100ns)

MaxDiff(V(Outp_f) - V(Outp_g))

Figure 9-1: Results LNA; Bridge 0

Optimal Test Signal Generation for Analog Test
Diploma Work of Tobias Völkel
- 62 -
Figure 9-2: Results LNA; Bridge 3
Brg_15 (N107, OUTP) *** NOT DETECTED BY IDD ***
(Brg = 100 Ohm, SimTime = 100ns)

Figure 9-3: Results LNA; Bridge 15

Optimal Test Signal Generation for Analog Test
Diploma Work of Tobias Völkel
- 64 -
Figure 9-4: Results LNA; Bridge 46
Figure 9-5: Results LNA; Bridge 53
Figure 9-6: Results LNA; Bridge 55
Figure 9-7: Results LNA; Bridge 56
Figure 9-8: Results LNA; Bridge 60
Figure 9-9: Results LNA; Bridge 65
Figure 9-10: Results LNA; Bridge 71
Figure 9-11: Results LNA; Bridge 81

MaxDiff(V(Outp_f)-V(Outp_g))

V(INP) No of Pulses

VDD

Optimal Test Signal Generation for Analog Test
Diploma Work of Tobias Völkel
- 72 -
Figure 9-12: Results LNA; Bridge 88

MaxDiff(V(Outp_f) - V(Outp_g))

Brg_88 (VB, N68) *** NOT DETECTED BY IDD ***
(100 Ohm, SimTime 100ns)
Figure 9-13: Results LNA; Bridge 89
Figure 9-14: Results LNA; Bridge 104
Figure 9-15: Results LNA; Bridge 119
Brg_129 (OUTP, OUTN) *** NOT DETECTED BY IDD ***
(Brg = 100 Ohm, SimTime = 100ns)

MaxDiff(V(Outp_f) - V(Outp_g))

Figure 9-16: Results LNA; Bridge 129
Figure 9-17: Results LNA; Bridge 145
10. Acknowledgment

I want to thank Philips for the opportunity to make my Master Thesis at the NatLab in Eindhoven possible. And a special thank you to Maurice Lousberg as the department manager for supporting me. As well I want to thank Wolfgang Mathis as my professor from university in Hannover. He gave me the possibility to make an internship abroad and supported me with his knowledge. Guido Gronthoud had been a very nice project supervisor who supported me and always had an open mind for my questions. Thank you for everything. Another big thank you belongs to Jan ter Maten as my second supervisor and as well my second examiner for my master thesis. And of course an essential help was provided by Shaji Krishnan. He was my project leader and put a lot of effort in this project. Thank you very much for your help. I also want to thank Jo Gunnes who was my office neighbour. We had nice lunch discussions to free our minds again. Another thank you belongs to the ED&T helpdesk who helped me to get along with the Philips programs. I also want to mention some people who helped me during my master thesis. Jose Pineda de Gyvez, Amir Zjajo, Marq Kole, Ahmed El Guennouni, Henk van de Donk, Jacqueline Geerings, Achie Lin, Bratislav Tasic. Thank you.

Finally I want to thank my family, my parents for their patience and their support to give me the ability to study and my girlfriend Eva-Maria for her love and power to succeed.

Thank you to all who supported me and I forgot to mention.

Tobias Völkel
11. Literature


