FORMAL ANALYSIS OF HOMOGENEOUS LINEAR DIFFERENTIAL EQUATIONS USING THEOREM PROVING

By

Muhammad Usman Sanwal

2010-NUST-MS-CS&E-10

A thesis submitted in partial fulfillment of the requirements for the degree of Masters of Science in Computational Science and Engineering

Research Centre for Modeling and Simulation,
National University of Sciences and Technology (NUST),
Islamabad, Pakistan.
(October 2012)

Dedicated To My Father

Bashir Ahmad

Acknowledgements

In the name of ALLAH, the Most Gracious and the Most Merciful Alhamdulillah, all praises to ALLAH for the strengths and His blessings in completing this thesis.

I am immensely pleased to express my profound gratitude and heartfelt thanks to my advisor Dr. Osman Hasan for his excellent guidance, strong support, expert advice and providing me the life time chance for doing research under his supervision even though I had no research experience. I would also like to thanks him for supporting me by providing Research Assistantship. I attribute the level of my Masters degree to his encouragement and effort and without him this thesis, too, would not have been completed or written. His regularity, punctuality, prompt response of emails, his patience regarding my mistakes were all excellent. I could not have wished for a better thesis supervisor.

I offer my thank to Dr. Sohail Iqbal for taking time out of his busy schedule to serve as my external examinar. I would like to thank Dr. Meraj Mustafa Hashmi for his helpful suggestions, comments and for serving on my thesis committe. I sincerely thank to Engr. Sikander Hayat Mirza for his unconditional help and encourgment for doing my thesis in SAVE Lab. I would also like to thank Dr. Khalid Pervez who guided me on the various aspects of my research thesis.

I offer my special thank to Umair Siddique, my friend and a former member of SAVE Lab for his motivation and support. Without his help, I would not be where I am today. Many thanks to my colleagues at RCMS and all members of SMART Lab for their support and nice company. I would also like to thank Muhammad Wisal, Nadeem Iqbal, Binyameen, Ahmad, Faiq, Waqar Ahmed, Aqib Chisthy, Saqib Khan and Amjad Hussain for their support and encourgment.

Lastly but not least, I will always be grateful to my mother for her prayers, love and support. I wish to thank all my family members, specially my elder brother Sohail Ahmad for his support and motivation for higher studies.

Abstract

Traditionally, differential equations are solved using paper-and-pencil proof methods, computer based numerical methods or computer algebra systems. All these methods are error-prone and thus the analysis cannot be termed as accurate, which poses a serious threat to the accuracy of the safety-critical systems that involve differential equations. To guarantee the correctness of analysis, we propose to use higher-order-logic theorem proving to reason about the correctness of solutions of differential equations. This thesis presents a formalization framework to express homogeneous linear differential equation of arbitrary order and formally verify their solutions within the sound core of a higher-order-logic theorem prover HOL4. In order to illustrate the usefulness of the proposed formalization, we utilize it to formally verify the solutions of a couple of safety-critical biomedical systems, namely a heart pacemaker and a fluid-filled catheter, which are one of the most safety-critical systems as their bugs could eventually result in the loss of human lives. We also show the utilization of our work by formally verifying Analog and Mixed Signals (AMS) designs which are widely being used in many integrated circuits. Due to their continuous nature, their analysis has primarily been done with informal techniques, like simulation, or formal methods with abstracted discrete models. This fact makes AMS designs error prone, which may lead to disastrous consequences given the safety and financial critical nature of their applications. Leveraging upon the high expressiveness of higher-order logic, we propose to use higher-order-logic theorem proving to analyze continuous models of AMS designs. To take an example of our foundational AMS design analysis formalization, we present the formal analysis of a classical RLC circuit using the HOL4 theorem prover.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Differential Equations and their Historical Background

An equation involving one dependent variable and its derivatives with respect to one or more independent variables, is called a differential equation. Differential equations differ from algebraic equations of mathematics because in addition to variables and constants they also contain derivatives of one or more of the variables involved. Scientists and mathematicians discovered differential equations in the middle of the 17th century. It was Isaac Newton an english physicist, who solved his first differential equation in 1676 and was working with what he called ”fluxional equations”. In 1693, German mathematician Gottfried Leibniz solved his first differential equation and that same year Newton published the results of previous differential equation solution methods, a year that is said to mark the inception for the differential equations as a distinct field in mathematics.

1.2 Motivation

Most engineering and physiological systems exhibit a deterministic relationship between continuously changing quantities and their rates of change. Such systems are mathematically analyzed by first capturing their behaviors by differential equations [40], where a variable's rate of change is modeled by an appropriate derivative function. These differential equations are then solved to obtain interesting design parameters for the underlying physical system. Solving a differential equation means to find the set of values that, when substituted into the equation, satisfy it. Differential equations may have more than one solution or no solution at all. However, it is a general consensus that a correctly specified engineering or a physiological system always has a unique solution, which is usually obtained using initial conditions for each order of an ordinary differential equation [11].

1.3 Techniques used for Solving Differential Equations

1.3.1 Paper-and-Pencil Method

Traditionally, differential equations are solved using paper-and-pencil proof methods. However, considering the complexity of present age engineering and physiological systems, such kind of analysis is notoriously difficult, if not impossible, and is quite error prone due to the human error factor. Moreover, it is quite often the case that mathematicians forget to pen down all the required assumptions that are required for the validity of their analysis. This fact may lead to erroneous designs as well. For example, one of the recent paper-and-pencil based analysis bugs can be found in [7] and its identification and correction is reported in [30].

1.3.2 Computer based Softwares for solving Differential Equations

With the advent of computers, many computer based software tools based on the principle of numerical methods have been introduced for finding the solutions of differentials equations [3, 4]. Due to the reliable and efficient bookkeeping characteristic of computers, much larger systems can be analyzed using these methods. However, these methods cannot attain 100% accuracy due to the associated high memory and computation requirements for analyzing real-world systems and the inherent usage of finite precision computer arithmetic. Similarly Constraint Logic programming (CLP), which is an interval-based constraint language, has been used to model differential equations and find their solutions [20]. However, due to the usage of interval arithmetics, this method cannot guarantee the absolute accuracy of the results. Another alternative to determine the solutions of differential equations is a computer algebra system [13, 35], which is very efficient for computing mathematical solutions symbolically, but it is also unreliable [16] due to the presence of unverified huge symbolic manipulation algorithms in its core, which are quite likely to contain bugs. Thus, these traditional techniques should not be relied upon for the analysis of systems involving differential equations, especially when they are used in safety-critical areas, such as medicine and transportation, where inaccuracies in the analysis could result in system design bugs that in turn may even lead to the loss of human lives in worst cases.

1.3.3 Formal Methods

In the past couple of decades, formal methods have emerged as a successful verification technique for both hardware and software systems. The rigorous exercise of developing a mathematical model for the given system and analyzing this model using mathematical reasoning usually increases the chances for catching subtle but critical design errors that are often ignored by traditional techniques like paper-and-pencil based proofs or numerical methods. Given the extensive usage of differential equations in safety-critical systems, there is a dire need of using formal methods support in this domain. However, due to the continuous nature of the analysis and the frequent involvement of transcendental functions, pure automatic state-based approaches, like model checking [24], cannot be used in this domain. In order to overcome this limitation, hybrid model-checking and theorem proving based approaches, e.g, [1] have been used for the verification of hybrid systems having both discrete and continuous components. Moreover, safety properties of hybrid systems have also been formally verified using differential invariants [32, 33] based on fixed point algorithms. The higher-order logic theorem prover PVS has also been used in the context of verifying parallel composition of hybrid systems [2]. Similarly, the Coq theorem prover has been used to formally verify the convergence of numerical solutions for a widely used partial differential wave equation [5]. However, to the best of our knowledge, none of these existing formal approaches allow us to verify the solutions of differential equations. We believe that higher-order-logic theorem proving [14, 17] offers a promising solution for overcoming this limitation. The high expressiveness of higher-order logic can be leveraged upon to essentially model any system that can be expressed in a closed mathematical form. In fact, most of the classical mathematical theories behind elementary calculus, such as differentiation, limit, etc., and transcendental functions, which are the most fundamental tools for solving differential equations, have been formalized in higher-order logic [16].

1.4 Related Work

Formal methods are widely being used these days for the verification of hardware and software systems due to their extensive usage in safety-critical applications, such as medicine and transportation. However, to the best of our knowledge, the usage of formal methods for analyzing dynamic systems, whose behavior can be modeled by differential equations, is very rare. The main challenge being the continuous nature of the system behavior, which cannot be modeled precisely in computer-arithmetic systems like floating point numbers.

Traditionally, differential equations are solved using paper-and-pencil proof methods. However, considering the complexity of present age engineering and physiological systems, such kind of analysis is notoriously difficult, if not impossible, and is quite error prone due to the human error factor. Moreover, it is quite often the case that mathematicians forget to pen down all the required assumptions that are required for the validity of their analysis. This fact may lead to erroneous designs as well. For example, one of the recent paper-and-pencil based analysis bugs can be found in [7] and its identification and correction is reported in [30]. With the advent of computers, many computer based software tools based on the principle of numerical methods have been introduced for finding the solutions of differentials equations [3, 4]. Due to the reliable and efficient bookkeeping characteristic of computers, much larger systems can be analyzed using these methods. However, these methods cannot attain 100% accuracy due to the associated high memory and computation requirements for analyzing real-world systems and the inherent usage of finite precision computer arithmetic. Similarly Constraint Logic programming (CLP), which is an interval-based constraint language, has been used to model differential equations and find their solutions [20]. However, due to the usage of interval arithmetics, this method cannot guarantee the absolute accuracy of the results. Another alternative to determine the solutions of differential equations is a computer algebra system [13, 35], which is very efficient for computing mathematical solutions symbolically, but it is also unreliable [16] due to the presence of unverified huge symbolic manipulation algorithms in its core, which are quite likely to contain bugs. Thus, these traditional techniques should not be relied upon for the analysis of systems involving differential equations, especially when they are used in safety-critical areas, such as medicine and transportation, where inaccuracies in the analysis could result in system design bugs that in turn may even lead to the loss of human lives in worst cases.

The main principle behind formal analysis of a system is to construct a computer based mathematical model of the given system and formally verify, within a computer, that this model meets rigorous specifications of intended behavior. Two of the most commonly used formal verification methods are model checking [24] and higher-order-logic theorem proving [14]. Model checking is an automatic verification approach for systems that can be expressed as a finite-state machine. Higher-order-logic theorem proving, on the other hand, is an interactive approach but is more flexible in terms of tackling avariety of systems.

In model checking the basic idea is to represent the system behavior as a finite-state concurrent model and the system properties as temporal logic formulas. Then, efficient symbolic algorithms are used to automatically traverse the finite-state model of the system and check if a given temporal logic specification holds or not. However, model checking is not very suitable for analyzing the pure continuous models of a system. The biggest limitation associated with model checking in this regard is the state-space explosion problem [24]. The state space of a system with continuous components is usually very very large or infinite. Thus, at the outset, it becomes impossible to explore the entire state space with limited resources of time and computer memory.

Higher-order logic is a system of deduction with a precise semantics and is expressive enough to be used for the specification of almost all classical mathematics theories. Interactive theorem proving is the field of computer science and mathematical logic concerned with precise computer based formal proof tools that require some sort of human assistance. Due to the high expressive nature of the underlying logic, higher-order-logic theorem proving can be utilized to analyze any system, irrespective of its continuous nature, as long as it can be expressed in a closed mathematical form. In fact, most of the classical mathematical theories, such as real numbers, differentiation, limits [16], which are the most fundamental tools for analyzing continuous systems, have already been formalized in higher-order-logic. Given the availability of this foundational formalization, we utilize higher-order-logic theorem proving for developing a formal framework to reason about homogeneous linear differential equations. In particular, we use the HOL4 theorem prover, which is an LCF style theorem prover mainly to have access to Harrison's seminal work on real analysis[16] .

Other notable higher-order-logic formalizations related to differential equations include verification of the convergence of numerical solutions for differential equations [5] and the approximate numerical solution of ordinary differential equations using the one-step method [23]. However, to the best of our knowledge, none of these existing formal approaches allow us to verify the solutions of differential equations that are obtained via traditional informal analysis techiques. In this thesis, we provide this capability by leveraging upon the high expressiveness of higher-order logic. The main idea is to construct pure continuous models of differential equations and their solutions as functions in higher-order logic and formally verify their correspondence using theorem proving [14, 17]. This way the exact solutions obtained from traditional techniques can be verified and the error-bounds on the approximate solutions can also be formally judged. Moreover, the current thesis presents the formal verification of solutions of differential equations of arbitrary order, which to the best of our knowledge is a novelty.

1.5 Proposed Methodology

The main purpose of this thesis is to formally verify the solutions of homogeneous linear differential equation with constant coefficients. Particularly, developed framework is characterized as :

1. The formal definitions of the derivative, n-order derivative and differntial equation in HOL theorem prover.
2. The ability to formally verify the theorems and properties of the homogeneous linear differential equations in a higher-order logic theorem prover. These properties play a very important role for the formal analysis of the systems that involve differential equations.
3. The ability to formally reason about the theorems, formalized in step 2, using a theorem prover.
4. The ability to utilize the above mentioned capabilities to formally model and reason about real world problems that can be modeled using differential equations.

The proposed methodology, given in Figure 1.1, outlines the main idea behind the theorem proving based analysis of systems involving differential equations. The grey shaded boxes in this figure represent the key contributions of the thesis that serve as basic requirements for conducting formal analysis of the systems involving differential equations in a theorem prover. The input to this framework, depicted by two rectangles with curved bottoms, is the formal description of the system involving differential equations that needs to be analyzed and a set of constraints (properties) that are required to be checked for the given

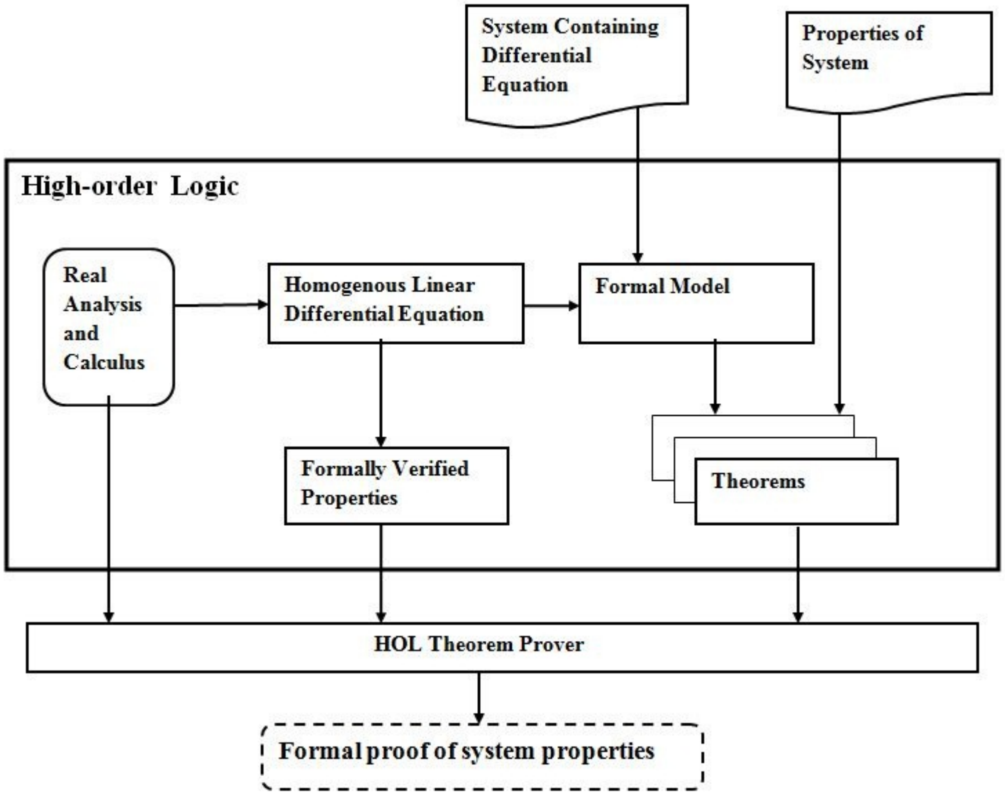
System Containing Differential Equation
Properties of System
High-order Logic
Real Analysis and Calculus
Homogenous Linear Differential Equation
Formal Model
Formally Verified Properties
Theorems
HOL Theorem Prover
Formal proof of system properties

Figure 1.1: Proposed Methodology

system. The first step in conducting formal analysis of the systems involving differential equations is to construct a formal model of the given system in higher-order-logic. For this purpose, our formalization primarily builds upon the higher-order-logic formailzation of the derivative function and its associated propertties. Our work is based on Harrison's formalization [16] that is available in HOL4 theorem prover. The second step is to represent the interesting properties of the system as higher-order-logic proof goals.

The third step for conducting analysis of system involving differential equations in a theorem prover is to formally verify the higher-order-logic theorems developed in the previous step using a theorem prover. For this verification, it would be quite handy to have access to a library of some pre-verified theorems corresponding to some commonly used properties of the differential equations. To fullfill this requirment, this thesis presents the formal verification of properties related to differential equations such as linearity for n-order derivative and homogeneous property. Building on such a library of theorems would minimize the interactive verification efforts and thus speed up the verification process. Finally, the output of the theorem proving based framework of the system involving differential equations, depicted by the rectangle with dashed edges, is the formal proofs of system properties that certify that the given system properties are valid for the given system involving differential equations.

1.6 Thesis Contributions

This thesis presents a set of formal definitions that allow us to formalize any homogeneous linear differential equation. We also provide a couple of formally verified properties that facilitate reasoning about the solutions of these differential equations within the sound core of a higher-order-logic theorem prover HOL4. The almost automatic reasoning process makes our methodology very useful for industrial usage as its users do not have to very proficient with the cumbersome real-theoretic formal reasoning process. More specifically, this thesis makes the following contributions:

1. It provides the formal definitions related to solution of general order homogeneous linear differential equations.

2. The above definitions provide a framework to formally describe and verify solutions of homogeneous linear differential equation.

3. In this thesis, For illustration purpose, we applied the proposed methodology for verifying the solutions of differential equations related to a couple of biomedical systems and Analog and Mixed Signal designs. To the best of our knowledge, this is the first formal reasoning support for verifying solutions of homogeneous linear differential equations that has been reported in the open literature.

1.7 Organization of the Thesis

The rest of the thesis is organized as follows: Chapter 2 presents a brief introduction to the HOL4 theorem prover and its formalization of the derivative function. In Chapter 3, we present the formalization of the second order homogeneous linear differential equation and then we utilize this formalization to formally verify its general solution. Chapter 4 describes the formalization of the general order homogeneous linear differential equation and similarly as in chapter 3, we utilize this formalization to formally verify its general solution. The analysis of the couple of biomedical applications and AMS designs such as series RLC circuit is presented in Chapter 5. Finally, Chapter 6 concludes the thesis and gives directions for some future work.

Chapter 2

Preliminaries

In this chapter, we give a brief introduction to the HOL4 theorem prover and the formalization of the derivative function in HOL4 function to facilitate the understanding of the rest of the thesis.

2.1 HOL4 Theorem Prover

HOL4 is an interactive theorem prover developed by Mike Gordon at the University of Cambridge for conducting proofs in higher-order logic. It utilizes the simple type theory of Church [6] along with Hindley-Milner polymorphism [29] to implement higher-order logic. HOL4 has been successfully used as a verification framework for both software and hardware as well as a platform for the formalization of pure mathematics.

2.2 Theorem Proving

In order to ensure secure theorem proving, the logic in the HOL4 system is represented in the strongly-typed functional programming language ML [31]. An ML abstract data type is used to represent higher-order logic theorems and the only way to interact with the theorem prover is by executing ML procedures that operate on values of these data types. The HOL4 core consists of only 5 basic axioms and 8 primitive inference rules, which are implemented as ML functions.

2.3 Terms

There are four types of HOL4 terms: constants, variables, function applications, and lambda-terms (denoted function abstractions). Polymorphism, types containing type variables, is a special feature of higher-order logic and is thus supported by HOL4. Semantically, types denote sets and terms denote members of these sets. Formulas, sequences, axioms, and theorems are represented by using terms of Boolean types.

2.4 Theories

A HOL4 theory is a collection of valid HOL4 types, constants, axioms and theorems, and is usually stored as a file in computers. Users can reload a theory in the HOL4 system and utilize the corresponding definitions and theorems right away. The concept of HOL4 theory allows us to build upon existing results in an efficient way without going through the tedious process of regenerating these results using the basic axioms and primitive inference rules.

HOL4 theories are organized in a hierarchical fashion. Any theory may inherit types, definitions and theorems from other available HOL4 theories. The HOL4 system prevents loops in this hierarchy and no theory is allowed to be an ancestor and descendant of a same theory. Various mathematical concepts have been formalized and saved as HOL4 theories by the HOL4 users. These theories are available to a user when he first starts a HOL4 session. We utilized the HOL4 theories of Booleans, lists, positive integers and *real* analysis in our work. In fact, one of the primary motivations of selecting the HOL4 theorem prover for our work was to benefit from these built-in mathematical theories.

2.5 Writing Proofs

HOL4 supports two types of interactive proof methods: forward and backward. In forward proof, the user starts with previously proved theorems and applies inference rules to reach the desired theorem. In most cases, the forward proof method is not the easiest solution as it requires the exact details of a proof in advance. A backward or a goal directed proof method is the reverse of the forward proof method. It is based on the concept of a *tactic*; which is an ML function that breaks goals into simple sub-goals. In the backward proof method, the user starts with the desired theorem or the main goal and specifies tactics to reduce it to simpler intermediate sub-goals. Some of these intermediate sub-goals can be discharged by matching axioms or assumptions or by applying built-in decision procedures. The above steps are repeated for the remaining intermediate goals until we are left with no further sub-goals and this concludes the proof for the desired theorem.

The HOL4 theorem prover includes many proof assistants and automatic proof procedures to assist the user in directing the proof. The user interacts with a proof editor and provides it with the necessary tactics to prove goals while some of the proof steps are solved automatically by the automatic proof procedures.

2.6 Derivatives in HOL4

Harrison [16] formalized the *real number theory* along with the fundamentals of calculus, such as real sequences, summation series, limits of a function and derivatives and verified most of their classical properties in HOL4. The limit of a function f, which takes a real number and returns a real number, is defined in HOL4 using the operator → as follows [16]:

Definition 1: *Limit of a Function*

```
⊢ ∀ f y0 x0.  (f → y0)(x0) =
    ∀e.  0 < e ⇒
      ∃d.  0 < d ∧ ∀x.  0 < |x - x0| ∧ |x - x0| < d ⇒
        |f(x) - y0| < e
```

where `(f → y0)(x0)` can be written mathematically as $lim_{(x\rightarrow x0)} f(x) = y0$, i.e., the function `f` approaches `y0` as its real number argument approaches `x0`. Based on this definition, the derivative of a function f is defined as follows [16]:

Definition 2: *Derivative of a Function (Relational Form)*

```
⊢ ∀ f l x.  (f diffl l) x = ((λ h.(f (x + h) - f x) / h) → l) (0)
```

Definition 2 provides the derivative of a function f at point x as the limit value of $\frac{f(x+h)-f(x)}{h}$ when h approaches 0, which is the standard mathematical definition of the derivative function. Now, the differentiability of a function f is defined as the existence of its derivative [16].

Definition 3: *Differentiability of a Function*

```
⊢ ∀ f x.  f differentiable x = ∃l.  (f diffl l) (x)
```

A functional form of the derivative, which can be used as a binder, is also defined using the Hilbert choice operator `@`as follows [16]:

Definition 4: *Derivative of a Function (Functional Form)*

```
⊢ ∀ f x.  deriv f x = @l.  (f diffl l) x
```

The function `deriv` accepts two parameters f and x and returns the derivative of function f at point x.

The above mentioned definitions associated with the derivative function have been accompanied by the formal verification of most of their classical properties, such as uniqueness, linearity and composition [16].

2.6.1 HOL Notations

Table 2.1 provides the mathematical interpretations of some frequently used HOL symbols and functions in this thesis.

Table 2.1: HOL Symbols and Functions

HOL Symbol	**Standard Symbol**	**Meaning**
/\	and	Logical *and*
\/	or	Logical *or*
$\sim$	not	Logical *negation*
==>	$\longrightarrow$	Implication
<==>	$=$	Equality
`!x.t`	$\forall x.t$	for all $x : t$
`?x.t`	$\exists x.t$	for some $x : t$
λ`x.t`	$\lambda x.t$	Function that maps x to $t(x)$
`num`	$\{0, 1, 2, \ldots\}$	Positive Integers data type
`real`	All Real numbers	Real data type
`suc n`	$(n + 1)$	Successor of natural number
`ln x`	$log_e(x)$	Natural logarithm function
`abs x`	$\vert x\vert$	Absolute function
`min x y`	$min(x, y)$	Minimum of x and y
`max x y`	$max(x, y)$	Maximum of x and y
`FACT n`	$n!$	Factorial of n
`inv x`	$1/x$	Inverse of x
SUC n	$n + 1$	Successor of a *num*
$m \ \ast\ast\ n$	m^n	*num* m raised to *num* exponent n
inv x	x^{-1}	Multiplicative inverse of a *real* x
$\lambda x.t$	$\lambda x.t$	Function that maps x to $t(x)$
$lim(\lambda n.f(n))$	$\lim\limits_{n\to\infty} f(n)$	Limit of a *real* sequence f
$\{x\vert P(x)\}$	$\{\lambda x.P(x)\}$	Set of all x that satisfy the condition P
(a, b)	a x b	A mathematical pair of two elements

Chapter 3

Formalization of Second-order Homogeneous Linear Differential Equation in HOL4

This chapter presents the higher-order-logic formalization of second-order homogenous linear differential equation and the formal verification of its general solution using the HOL4 theorem prover. This chapter consists of two sectioms. In the first section we have verified solution of second-order homogenous linear differential equation with real and distinct roots while in second section we have verified solution of second-order homogenous linear differential equation with real and repeated roots

3.1 Second-order Homogeneous Linear Differential Equations

A second-order homogeneous linear differential equation can be mathematically expressed as follows:

$$p_2(x)\frac{d^2y(x)}{dx} + p_1(x)\frac{dy(x)}{dx} + p_0(x)y(x) = 0 \tag{3.1}$$

where terms p_i represent the coefficients of the differential equation defined over a function y. The equation is linear because (i) the function y and its derivatives appear only in their first power and (ii) the products of y with its derivatives are also not present in the equation. By finding the solution of the above equation, we mean to find functions that can be used to replace the function y in the above equation and satisfy it.

We proceed to formally represent the above equation by first formalizing an n$^{\mathtt{th}}$-order derivative function as follows:

Definition 1: *N*$^{\mathtt{th}}$*-order Derivative of a Function*

```
⊢ (∀ f x.  n_order_deriv 0 f x = f x) ∧
  (∀f x n.n_order_deriv (n+1) f x = n_order_deriv n (deriv f x) x)
```

The function `n_order_deriv` accepts an integer n that represents the order of the derivative, the function f that represents the function that needs to be differentiated, and the variable x that is the variable with respect to which we want to differentiate the function f. It returns the n$^{\mathtt{th}}$-order derivative of f with respect to x. Now, based on this definition, we can formalize the left-hand-side (LHS) of an n$^{\mathtt{th}}$-order differential equation in HOL4 as the following definition.

Definition 2: *LHS of a N*$^{\mathtt{th}}$*-order Differential Equation*

```
⊢ ∀ P y x.  diff_eq_lhs P y n x =
    sum(0,n)(λm.(EL m P x) * (n_order_deriv m y x))
```

The function `diff_eq_lhs` accepts a list P of coefficient functions corresponding to the p_i's of Equation (3.1), the differentiable function y, the order of differentiation n and the differentiation variable x. It utilizes the HOL4 functions `sum (0,n) f` and `EL n L`, which correspond to the summation ($\sum_{i=0}^{n-1} f_i$) and the $\mathtt{n}^{\mathtt{th}}$ element of a list L_n, respectively. It generates the LHS of a differential equation of n$^{\mathtt{th}}$ order with coefficient list P. The second-order differential equation of Equation (3.1) can now be formally modeled by instantiating variable n of Definition 2 by number 3, since n = 0 returns 0.

3.1.1 Solution of Second-order Homogeneous Linear Differential Equations with Real and Distinct roots

If the coefficients p_i's of Equation (3.1) are constants then, using the fact that the derivative of the exponential function $y = e^{rx}$ (with a constant r) is a constant multiple of itself $dy/dx = re^{rx}$, if the roots of Equation (3.1) are real and distinct then we can obtain the following solution:

$$Y(x) = c_1 e^{r_1 x} + c_2 e^{r_2 x} \tag{3.2}$$

where c_1 and c_2 are arbitrary constants and r_1 and r_2 are the real and distinct roots of the auxiliary equation $p_2 r^2 + p_1 r^1 + p_0 = 0$ [40]. In this section, we formally verify this result which plays a key role in formal reasoning about the solutions of second-order homogeneous linear differential equations in a higher-order-logic theorem prover.

Theorem 1: *General Solution of a Second-order Homogeneous Linear Differential Equation with real and distinct roots*

```
⊢ ∀ a b c c1 c2 r1 r2 x.
  (c + (b * r1) + (a * (r1 pow 2)) = 0) ∧
  (c + (b * r2) + (a * (r2 pow 2)) = 0) ⇒
  (diff_eq_lhs (const_list [c; b; a])
    (λx.  c1 * (exp (r1 * x)) + c2 * (exp (r2 * x))) 3 x = 0)
```

where $[c; b; a]$ represents the list of constants corresponding to the coefficients p_0, p_1 and p_2 of Equation (3.1), $r1$ and $r2$ represent the roots of the corresponding auxiliary equation as given in the assumptions, $c1$ and $c2$ are the arbitrary constants and x is the variable of differentiation. The function `const_fn_list` used in the above theorem transforms a constant list to the corresponding constant function list recursively as follows:

Definition 3: *Constant Function List*

```
⊢ (const_fn_list [] = []) ∧
(∀ h t.  const_fn_list (h::t) = (λ(x:real).  h) ::  (const_fn_list t))
```

The function diff_eq_lhs permits coefficients that are functions of the variable of differentiation but Theorem 1 is valid only for constant coefficients. Thus, using const_fn_list we provide the required type for the coefficient list of the function diff_eq while fulfilling the requirement of Theorem 1. The formal reasoning about Theorem 1 is primarily based on differential of exponential function which has been verified in Theorem 3 and the linearity property of higher-order derivatives, which has been verified in our work for *class* C^n functions, i.e., the functions for which the first n derivatives exist for all x as the following higher-order-logic theorem:

Theorem 2: *Linearity of n^{th}-order Derivative*

```
⊢ ∀ f g x a b.
  (∀m x.  m ≤ n ⇒ (λx.  n_order_deriv m f x) differentiable x) ∧
  (∀m x.  m ≤ n ⇒ (λx.  n_order_deriv m g x) differentiable x) ⇒
    (n_order_deriv n (λx.  a * f x + b * g x) x =
      a * n_order_deriv n f x + b * n_order_deriv n g x)
```

where variables a and b represent constants with respect to variable x. The formal reasoning about Theorem 2 involves induction on variable n, which represents the order of differentiation, and is primarily based on the linearity property of the first order derivative function [16]. The derivatives of some commonly used transcendental functions have also been verified. For example, the derivative of the Exponential function has been verified as follows:

Theorem 3: *Differential of the Exponential Function*

```
⊢ ∀ g m x.  ((g diffl m) x ⇒
    ((λ x.  exp (g x)) diffl (exp (g x) * m)) x)
```

where exp x represents the exponential function e^x and $(\lambda \mathtt{x}.\mathtt{f}(\mathtt{x}))$ represents the lambda abstraction function which accepts a variable x and returns $f(x)$.

Similarly, the Gauge integral has been formalized as a function Dint $(\mathtt{a},\mathtt{b})$ f k [16], which mathematically describes $\int_{\mathtt{a}}^{\mathtt{b}} \mathtt{f(x)\ dx} = \mathtt{k}$. The corresponding function form is

Definition 4: *Integral of a Function (Functional Form)*

```
⊢ ∀ a b f.integral (a,b) f = @k.Dint (a,b) f k
```

Many interesting properties of integration have been formally verified in [16] and, for the current work, we also extended the formal reasoning support for integrals as part of the reported work. One of the most useful formally verified property in our development being the first fundamental theorem of calculus (FTC) [15], according to which if f is a continuous real-valued function defined on a closed interval $[a, b]$ and

$$F(x) = \int_a^x f(t)\,dt \texttt{ then } \frac{dF(x)}{dx} = f(x) \tag{3.3}$$

for all x in (a, b). We verified this in HOL4 as follows:

Theorem 4: *First Fundamental Theorem of Calculus*

```
⊢ ∀ f a b x.  (a < x) ∧ (x < b) ∧ (∀ y.  (a≤y) ∧ (y≤b)
⇒ (g y = integral (a,y) f)) ∧ (∀ z.  (a≤z) ∧ (z≤b)
⇒ (f contl z)) ⇒ ((g diffl (f x)) x)
```

We build upon the above mentioned results to develop the higher-order-logic foundations for AMS circuit analysis.

3.1.2 Solution of Second-order Homogeneous Linear Differential Equations with Real and Repeated roots

If the roots of Equation (3.1) are real and repeated then we can obtain the following solution:

$$Y(x) = c_1 e^{rx} + c_2 x e^{rx} \tag{3.4}$$

where c_1 and c_2 are arbitrary constants and r is the real and repeated root occurring twice of the auxiliary equation $p_2 r^2 + p_1 r^1 + p_0 = 0$ [40]. Just like Theorem 3, we also formally verified that this solution satisfies Equation (3.1).

Theorem 5: *General Solution of a Second-order Homogeneous Linear Differential Equation with real and repeated roots*

```
⊢ ∀ a b c c1 c2 r1 r2 x.
  (c + (b * r) + (a * (r pow 2)) = 0) ∧
  ((b pow 2) - (4 * a * c) = 0) ∧ b <> 0
  ⇒ (diff_eq_lhs (const_list [c; b; a])
     (λx.  c1 * (exp (r * x)) +
                        c2 * (x * exp (r * x))) 3 x = 0)
```

where r represents the real and repeated root of the corresponding auxiliary equation and the rest of the variables are the same as Theorem 1. The formal reasoning about Theorem 4 is also mainly based on Theorems 2 and 3 and the well-known quadratic formula which we formally verified as follows:

Theorem 6: *Quadratic Formula*

```
⊢ ∀ a b c x.
((a*(x pow 2)) + (b*x)+c = 0) ∧ a <> 0 ⇒
   (x=(-b+sqrt(b pow 2 - 4*a*c))/(2 * a)) ∧
                (x=(-b-sqrt(b pow 2 - 4*a*c))/(2 * a))
```

Chapter 4

Formal Verification of Solutions of General-order Homogeneous Linear Differential Equation in HOL4

In the last chapter we have formally verified solutions of second-order homogeneous linear differential equations. In this chapter we have enhanced our work to the general order and formally verified solutions of homogeneous linear differential equations of general order. This chapter also consists of two sections. In the first section we have formally verified solutions of homogeneous linear differential equations of general order with real and distinct roots while in second section we have formally verified solutions of homogeneous linear differential equations of general order with real and repeated roots.

4.1 Homogeneous Linear Differential Equations

An n$^{\mathtt{th}}$-order homogeneous linear differential equation can be mathematically expressed as follows:

$$p_n(x)\frac{d^n y(x)}{dx} + p_{n-1}(x)\frac{d^{n-1} y(x)}{dx} + \cdots + p_0(x)y(x) = 0 \tag{4.1}$$

where terms $p_i(x)$ represent the coefficients of the differential equation defined over a function y. The equation is linear because (i) the function y and its derivatives appear only in their first power and (ii) the products of y with its derivatives are also not present in the equation. By finding the solution of the above equation, we mean to find functions that can be used to replace the function y in Equation (4.1) and satisfy it. In the next section, we utilize Definition 2 and Theorem 2 which are explained in chapter 3 to verify generic solutions of Equation (4.1).

4.2 Solution of Homogeneous Linear Differential Equations

It is a well-known mathematical fact that if $y_1(x), y_2(x), \cdots, y_n(x)$ are independent solutions of Equation (4.1) then their linear combination

$$Y(x) = c_1y_1(x) + c_2y_2(x) + \cdots + c_ny_n(x) \tag{4.2}$$

also forms a solution of Equation (4.1), where $c_1, c_2, \cdots, c_n$ are arbitrary constants [40]. This result plays a vital role in solving differential equations as it allows us to find the solution of a differential equation if its n independent solutions are known.

We formalized the first property, corresponding to Equation (4.2), as the following higher-order-logic theorem:

Theorem 1: *General Solution of a Homogeneous Linear Differential Equation*

```
⊢ ∀ Y C P x.
  (n_order_differentiable_fn_list Y (LENGTH P)) ∧
  (n_order_diff_eq_soln_list Y P x) ⇒
    (diff_eq_lhs P (λx.  linear_sol C Y x) x = 0)
```

where Y represents the list of solutions $y_1(x), y_2(x), \cdots, y_n(x)$ of the given differential equation, C represents the list of arbitrary constants $c_1, c_2, \cdots, c_n$, P represents the list of functions corresponding to the coefficients $p_1(x), p_2(x), \cdots, p_n(x)$ of the differential equation and x is the variable of differentiation. The first predicate in the assumptions of

Theorem 1, i.e, `n_order_differentiable_fn_list`, ensures that each element of the list Y is n^{th}-order differentiable, where n ranges from 0 to `LENGTH P`. It is defined in HOL4 recursively as follows:

Definition 1: *N^{th}-order Differentiable List of Functions*

```
⊢ (∀ n.  n_order_differentiable_fn_list [] n = True) ∧
  ∀ h t n.  n_order_differentiable_fn_list (h::t) n =
    (∀m x.  m ≤ n ⇒ (λx.  n_order_deriv m h x) differentiable x) ∧
      n_order_differentiable_fn_list t n
```

where `::` represents the list *cons* operator in HOL4.

The second predicate in the assumptions of Theorem 1, i.e., `n_order_diff_eq_ soln_list`, ensures that each element of the list Y is a solution of the given differential equation with coefficients P. This predicate is recursively defined in HOL4 as follows:

Definition 2: *List of Solutions of a N^{th}-order Differential Equation*

```
⊢ (∀ P x.  n_order_diff_eq_soln_list [] P x = True) ∧
  ∀ h t P x.  n_order_diff_eq_soln_list (h::t) P x =
(diff_eq_lhs P h x = 0) ∧ n_order_diff_eq_soln_list t L x
```

Finally the function `linear_sol`, used in the conclusion of Theorem 1, models the linear solution represented by Equation (4.2) using the lists of solution functions Y and arbitrary constants C as follows:

Definition 3: *Linear Combination of Solutions*

```
⊢ (∀ C x.  linear_sol C [] x = 0) ∧
  ∀ C h t x.  linear_sol C (h::t) x =
    EL (LENGTH C - LENGTH (h::t)) C * h x + linear_sol C t x
```

The looping variable of the above definition is instantiated with the list Y in Theorem 1 and the expression `EL (LENGTH C - LENGTH (h::t)) C` picks the corresponding constant from list C for each y_i. Thus, using the functions `linear_sol` and `diff_eq_lhs`, we have formally specified the intended property in the conclusion of Theorem 1.

We verified Theorem 1 by performing induction on the the variable Y. The proof is primarily based on the linearity properties of the n$^{\texttt{th}}$-order derivative, given in Theorem 2 of chapter 3, the linearity properties of the summation function along with arithmetic reasoning [34].

4.3 Solution of Homogeneous Linear Differential Equation with Real and Distinct Roots

A particular case of interest arises when the coefficients p_i's of Equation (4.1) are constants in terms of the differentiation variable x. In this case, using the fact that the derivative of the exponential function $y = e^{rx}$ (with a constant r) is a constant multiple of itself $dy/dx = re^{rx}$, we can obtain the following solution of Equation (4.1):

$$Y(x) = c_1 e^{r_1 x} + c_2 e^{r_2 x} + \cdots + c_n e^{r_n x} \tag{4.3}$$

where $c_1, c_2, \cdots, c_n$ are arbitrary constants and $r_1, r_2, \cdots, r_n$ are the real and distinct roots of the auxiliary equation

$$p_n r^n + p_{n-1} r^{n-1} + \cdots + p_0 = 0 \tag{4.4}$$

with constant p_i's [40]. The above mentioned results play a key role in solving homogeneous linear order differential equations and the main focus of this thesis is the formal verification of these results, which in turn would facilitate formal reasoning about the solutions of differential equations in a higher-order-logic theorem prover.

The second property of interest, described using Equation (4.3), can be expressed in HOL4 as the following theorem:

Theorem 2: *General Solution of a Homogeneous Linear Differential Equation with Real and Distinct Roots*

```
⊢ ∀ C P R x.
  (aux_eq_roots_list R (const_fn_list P) x) ⇒
    (diff_eq_lhs (const_fn_list P)
      (λx.  linear_sol C (exp_list R) x) x = 0)
```

where C represents the list of arbitrary constants $c_1, c_2, \cdots, c_n$, P represents the list of constants corresponding to the coefficients $p_1, p_2, \cdots, p_n$ of the differential equation, R represents the list of roots $r_1, r_2, \cdots, r_n$ of the auxiliary equation, given in Equation (5.1), and x is the variable of differentiation. The function `const_fn_list` used in the above theorem transforms a constant list to the corresponding constant function list recursively as follows:

Definition 4: *Constant Function List*

```
⊢ (const_fn_list [] = []) ∧
  (∀ h t.const_fn_list (h::t) = (λ(x:real).h) ::  (const_fn_list t))
```

The function `diff_eq_lhs` permits coefficients that are functions of the variable of differentiation but Theorem 2 is valid only for constant coefficients. Thus, using `const_fn_list` we provide the required type for the coefficient list of the function `diff_eq_lhs` while fulfilling the requirement of Theorem 2.

The assumption predicate, i.e, `aux_eq_roots_list`, ensures that each element of the list R is a valid root of the auxiliary equation, like the one given in Equation (5.1), with constant coefficients given by list P. It is defined in HOL4 recursively as follows:

Definition 5: *Auxiliary Equation Roots*

```
⊢ ∀ P r x.  aux_eq_root P r x =
    (sum(0,LENGTH P)(λn.((EL n P x)) * (r pow n)) = 0)
    (∀ P x.  aux_eq_roots_list [] P x = True) ∧
    (∀ h t P x.  aux_eq_roots_list (h::t) P x =
                (aux_eq_root P h x) ∧ (aux_eq_roots_list t P x))
```

The first function aux_eq_root ensures that its argument r is a valid root of the auxiliary equation formed by coefficients given in list P. The function aux_eq_roots_list recursively calls function aux_eq_root for each entry of the looping variable and thus ensures that all the entries of the looping list are valid roots of the auxiliary equation formed by coefficients given in list P.

Finally, the function exp_list is used in Theorem 2 to model a list of exponential functions that are used to form the solution of the main differential equation, like the one given in Equation (4.3). This function is defined as follows:

Definition 6: *List of Exponential Functions*

```
⊢ (exp_list [] = []) ∧
  (exp_list (h::t) = (λx.  exp (h * x)) ::  (exp_list t))
```

It is important to note that the function linear_sol is used to express the conclusion of Theorem 2 as has been then case for Theorem 1. This way, the formally verified result of Theorem 1 can be used in formally verifying Theorem 2. The formal reasoning about Theorem 2 is conducted by performing induction on variable Y and the reasoning is primarily based on Theorem 2 and the following lemma that allows us to express the left-hand-side of the step case subgoal of Theorem 2 in terms of real summation [34].

Lemma 1:

```
⊢ ∀ P h x.  (diff_eq_lhs P (λx.  exp (h * x)) x =
  (exp (h * x) * (sum (0,LENGTH P) (λn.  EL n P x * h pow n))))
```

4.4 Solution of Homogeneous Linear Differential Equation with Real and Repeated Roots

Now, If the roots of an auxiliary equation are real and repeated then the solution of Equation (4.1) can be written as

$$Y(x) = c_1 e^{rx} + c_2 x e^{rx} + \cdots + c_n x^{n-1} e^{rx} \tag{4.5}$$

where $c_1, c_2, \cdots, c_n$ are arbitrary constants and r is the real and repeated root of the auxiliary equation given below

$$p_n r^n + p_{n-1} r^{n-1} + \cdots + p_0 = 0 \tag{4.6}$$

The solution of Equation (4.1) , described using Equation (4.5), can be expressed in HOL4 as the following theorem:

Theorem 3: *Solution of a Homogeneous Linear Differential Equation with Real and Repeated Roots*

```
⊢ ∀ C R r. (∀n. EL n R = r) ∧
  (∀m. m < LENGTH R ⇒
  (diff_eq_lhs (const_fn_list C) (λx. x pow m * exp (r * x)) x =0))
   ⇒ (diff_eq_lhs (const_fn_list C)
      (λx. linear_sol C (polynomial_function R) x) x = 0)
```

Where C and R are lists of constants and roots, respectively, just like the distinct roots theorem.

The assumptions of Theorem 3 ensure that all the roots of the auxiliary equation are the same and equal to r and $e^{rx}, xe^{rx}, x^2e^{rx}, \cdots, x^{LENGTH\ R\ -1}e^{rx}$ are all solutions of the given differential equation. The conclusion of the theorem specifies that Equation 5 is a solution of the given differential equation using the functions polynomial_function and linear_sol, which are defined as follows::

Definition 7: *List of Polynomial Functions*

```
⊢ (polynomial_function [] = []) ∧
  (polynomial_function (h::t) n =
  (λx.(x pow (LENGTH t)) * exp (h * x)) ::  (polynomial_function t))
```

Definition 8: *Linear Combination of Solutions*

```
⊢ (∀ C x.  linear_sol C [] x = 0) ∧
  ∀ C h t x.  linear_sol C (h::t) x =
    EL (LENGTH C - LENGTH (h::t)) C * h x + linear_sol C t x
```

The formal reasoning about Theorem 3 is conducted by performing induction on variable R and the reasoning is primarily based on Theorem 1 and the following lemma that tells us that all derivatives of exponential with multiple of increasing power of x are differentiable.

Lemma 2:

```
⊢ ∀ R n h x.
  (λx.  n_order_deriv n (λx.  x pow LENGTH R * exp(h * x)) x)
  differentiable x
```

Besides the above mentioned key results, we also verified the famous quadratic formula, which plays a vital role in reasoning about the auxiliary equations of second degree and also provides some support for reasoning about auxiliary equations of higher order. The quadratic formula is verified as the following theorem in our development.

Theorem 4: *Quadratic Formula*

```
⊢ ∀ a b c x.  (a ≠ 0) ∧ (4 * a * c < b pow 2) ⇒
aux_eq_roots_list [((-b + sqrt (b pow 2 - 4 * a * c)) / (2 * a));
    ((-b - sqrt (b pow 2 - 4 * a * c)) / (2 * a))]
      (const_fn_list [a; b; c]) x
```

where the functions `sqrt` and `pow` represent the square-root and square of a real number, respectively. The theorem essentially says that the roots of the auxiliary equation ax^2+bx+c are given by the first list argument of the function `aux_eq_roots_list`. The assumption `(4 * a * c < b pow 2)` guarantees that the roots are always real.

Chapter 5

Applications

In order to illustrate the usefulness of the proposed formalization, we show usefulness of our work by formally verifying analog and mixed signal (AMS) designs such as classical RLC circuit and an second-order Op Amp circuit. We utilize it to reason about the correctness of a couple of safety-critical biomedical systems, namely a heart pacemaker and a fluid-filled catheter.

5.1 Analog and Mixed Signal Circuits Applications

The analog and mixed signal (AMS) circuits [26] are characterized by having both analog and digital components on a single semiconductor die and are usually used to connect the electronic systems with their continuous real-world environments. These days, AMS circuits are widely used in many applications ranging from smart sensors, low data rate RF devices, medical monitoring devices, cellular telephones, software radios and network routers. The continuous nature of the AMS circuits makes their design and analysis quite challenging as we have to guarantee that the circuit generates a correct and stable output for all continuously changing inputs, in-die variations and noise parameters. AMS circuits are usually analyzed by first capturing their behaviors by appropriate differential equations [40] and then solving these differential equations to obtain the required design constraints.

Traditionally, AMS circuits are analyzed using paper-and-pencil proof methods. However, considering the complexity of present age AMS circuits, such kind of analysis is notoriously difficult, if not impossible, and is error prone due to the human error factor. Moreover, it is quite often the case that many key assumptions of the results obtained using paper-and-pencil proof methods are in the mind of the mathematician and are not documented. These missing assumptions may also lead to erroneous AMS designs. With the advent of computers, many computer-aided design tools based on the principles of numerical methods and simulation have been introduced for finding the solutions of differential equations and have been used to analyze AMS designs (See e.g. [28]). Due to the reliable and efficient bookkeeping characteristic of computers, complex AMS circuits can be analyzed using these methods. However, these methods cannot attain 100% accuracy due to the memory and computation limitations. Another alternative for analyzing AMS circuits is computer algebra systems [27, 21]. These methods are very efficient for computing mathematical solutions symbolically, but they are not reliable as well [18] due to the presence of unverified huge symbolic manipulation algorithms in their core, which are quite likely to contain bugs. Thus, given the above mentioned inaccuracies, these traditional techniques should not be relied upon for the analysis of AMS circuits, especially when they are used in safety-critical areas, such as medicine and transportation, where inaccuracies in the analysis could result in system design bugs that in turn may even lead to the loss of human life.

In the past couple of decades, formal methods have been successfully used for the precise analysis of a variety of digital circuits. The rigorous exercise of developing a mathematical model for the given system and analyzing this model using mathematical reasoning usually increases the chances for catching subtle but critical design errors that are often ignored by traditional techniques like paper-and-pencil based proofs or numerical methods. Given the extensive usage of AMS circuits in safety-critical systems, formal methods have also been utilized in this domain [38]. However, due to the continuous nature of the analysis and the frequent involvement of transcendental functions, automatic state-based approaches, like model checking [24], and automatic theorem provers [17] can only be used to analyze abstract discrete models of AMS circuits (See e.g., [10, 8]). In order to overcome this limitation, we propose to use higher-order-logic theorem proving [14] for conducting the formal analysis of AMS circuits since the high expressiveness of higher-order logic can be leveraged upon to model elements of continuous nature and thus have a more realistic formal analysis. The formalization of circuit analysis fundamentals, i.e., KVL, KCL and basic circuit components, is provided here.

5.1.1 Kirchhoff's Voltage and Current Laws

Kirchhoff's Voltage law (KVL) and Kirchhoff's current law (KCL) [9] form the most foundational circuit analysis laws. The KVL and KCL state that the directed sum of all the voltage drops around any closed network (loop) of an electrical circuit and the directed sum of all the branch currents leaving an electrical node is zero, respectively. Mathematically:

$$\sum_{k=1}^{n} V_k = 0, \sum_{k=1}^{n} I_k = 0 \tag{5.1}$$

where V_k and I_k represent the voltage drops across the k^{th} component in a loop and the current leaving the k^{th} branch in a node, respectively. The formalization is as follows:

Definition 1: *Kirchhoff's Voltage and Current Law*

```
⊢ ∀ V t.  kvl V t =   (∀ x.  0 < x ∧ x < t ⇒
    (sum (0,LENGTH V) (λn.  EL n V x) = 0))
⊢ ∀ V t.  kcl I t =   (∀ x.  0 < x ∧ x < t ⇒
    (sum (0,LENGTH I) (λn.  EL n I x) = 0))
```

The function `kvl` accepts a list V of functions of type (`real` → `real`), which represents the behavior of time-dependant voltages in the given circuit and a time variable t as a *real* number. It return the predicate that guarantees that the sum of all the voltages in the loop is zero for all time instants in the interval $(0, t)$. Similarly, the function `kcl` accepts a list I, which represents the behavior of time-dependant currents and a time variable t and returns the predicate that guarantees that the sum of all the currents leaving the node is zero for all time instants in the interval $(0, t)$.

The AMS circuits are analyzed using their corresponding differential equations. The first step in this regard is to capture the behavior of the given circuit mathematically using KVL or KCL. This mathematical expression may not be in the form of a differential equation as it may contain integrals. Thus, the equation is differentiated as many times as required to obtain an equivalent differential equation. In order to facilitate the formal reasoning about this transformation, we formally verified the following theorem.

Theorem 1: *KVL/KCL to Differential Equation*

```
⊢ ∀ V I t.(∀x.  0 < x ∧ x < t ⇒
 (EVERY (λf.  f differentiable x) V) ∧ (EVERY (λf.  f differentiable x) I) ∧
 (kvl V t) ∧ (kcl I t) ⇒ (kvl(deriv_list V) t) ∧ (kcl(deriv_list I) t)
```

The function `EVERY P L` ensures that every element of list `L` satisfies the predicate `P`. Thus, Theorem 6 allows us to differentiate all the terms in the mathematical equation obtained via KVL or KCL once as long as all the functions in the list of functions are differentiable. The proof of Theorem 6 is primarily based on the linearity property of a differential equation, given in Theorem 4.

Besides the above results, we also formalized the voltage expressions for a resistor, capacitor and inductor, which are the most commonly used analog circuit components as the following higher-order-logic functions:

Definition 2: *Resistor, Inductor and Capacitor*

```
⊢ ∀ R i.resistor_voltage R i = (λt.i t * R)
⊢ ∀ R v.resistor_current R v = (λt.v t / R)
⊢ ∀ L i.inductor_voltage L i = (λt.  L * deriv i t)
⊢ ∀ L v Io.inductor_current = (λt.  Io + 1/L * integral (0,t) v)
⊢ ∀ C i Vo.  capacitor_voltage C i Vo = (λt.  Vo + 1/C * integral (0,t) i)
⊢ ∀ C v.  capacitor_current = (λt.  C * deriv v t)
```

The variables i and v represents the time dependant current and voltage variables, respectively, in the above function definitions. While the variables R, L and C represent the constant resistance, inductance and the capacitance of their respective components, respectively. The variables Io and Vo are used in the definitions of inductance and capacitance to model the initial current in the inductor and the initial voltage across the capacitor [9], respectively. All these functions return a (`real` → `real`) type function that models the corresponding time dependant voltage or current. As a case studies, we present the formal analysis of a classical RLC circuit and an second-order Op Amp circuit.

5.1.2 RLC Series Circuit

Serially connected resistor (R), inductor (L) and capacitor (C), or the RLC, circuit is one of the classical examples of an AMS circuit. It is also widely used in modeling parasitics in the metal interconnect of sub-micrometer ICs [37]. We utilize the foundational formalization for analyzing AMS circuits, described in the last two sections, to formally verify the electrical current flow relationship in the RLC circuit, shown in Figure 1, with the intent to demonstrate the proposed methodology for formally analyzing AMS circuits. The first step in the proposed methodology is to model the behavior of the given circuit in higher-order logic. The behavior of the given circuit can be captured using the KVL as follows:

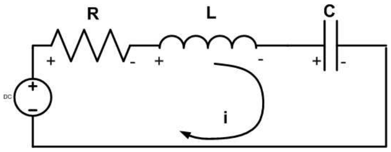

Figure 5.1: RLC Series Circuit with constant Voltage

Definition 3: *RLC Series Circuit Model*

```
⊢∀ R L C V Vo i t.rlc_ckt R L C V Vo i t =
  kvl [resistor_voltage R i; inductor_voltage L i;
                                  capacitor_voltage C i Vo; (λt.  -V)] t
```

The list input of the function `kvl` is composed of all the elements of the circuit that have a voltage drop. The dc voltage source V is modeled in this list as a time independent constant. The next step in the proposed methodology is to obtain a differential equation representation of the given AMS circuit. We formally verified this relationship as follows:

Theorem 2: *Differential Equation for the RLC Circuit*

```
⊢ ∀ R L C V Vo i t y.(0 < y) ∧ (y < t) ∧
(∀x.  0≤x ∧ x≤t ⇒ i differentiable x) ∧
(∀x.  0≤x ∧ x≤t ⇒ ((λt.deriv i t)) differentiable x) ∧
(rlc_ckt R L C V Vo i t) ⇒ (diff_eq_lhs(const_list[1/C;R;L]) i y = 0)
```

The conclusion of Theorem 2 describes the second-order differential equation corresponding to the RLC circuit given in the assumption using the function `rlc_ckt`. The theorem is verified under the assumptions that both the current function i and its first derivative are differentiable. It is also important to note that the theorem is valid for all time y in the interval $(0, t)$, where t represents the upper bound of the time for which the behavior of the function `rlc_ckt` is valid. Theorem 2 has been primarily verified using Theorem1 and Theorems 2 and 4 of Chapter 3.

Now, we have the differential equation, corresponding to our given circuit, and the next step in the proposed theorem proving based analysis of AMS circuits is to formally verify its solution. We consider the real and distinct roots of the given differential equation and verified the following theorem for the current of the given RLC circuit.

Theorem 3: *Current of the RLC Circuit*

```
⊢ ∀ L R C c1 c2 x.(L≠0) ∧ (4*L/C<R^2) ⇒
(diff_eq_lhs (const_list [1/C;R;L])
  (λx.c1*exp((-R+sqrt(R^2-4*L/C))/(2*L)*x)+
                      c2*exp((-R-sqrt(R^2-4*L/C))/(2*L)*x)) x = 0)
```

The conclusion of the above theorem provides us the solution $(c_1e^{-\frac{-R+\sqrt{R^2-\frac{4L}{C}}}{2L}x}+c_2e^{-\frac{-R-\sqrt{R^2-\frac{4L}{C}}}{2L}x})$ of the differential equation obtained from Theorem 2. It is formally verified primarily using Theorem 3 of chapter 3 and some simple arithmetic reasoning. The assumptions of Theorem 3 declare the relationships between the various parameters that are required for the solution to hold. This is one of strengths of the proposed theorem proving based verification as all the assumptions have to be explicitly stated besides the theorem for its formal verification. Thus, there is no chance of missing a critical assumption which often occurs in paper-and-pencil proof methods. It is also important to note the generic and continuous nature of Theorem 3 as all the variables are of type `real` and they are universally quantified. Such results cannot be obtained via state-based formal methods tools.

5.1.3 RLC Parallel Circuit

Now we formally verify the voltage drop in the RLC circuit, shown in Figure 2, with the intent to demonstrate the proposed methodology for formal analyzing AMS circuits. The first step in the proposed methodology is also to model the behavior of the given circuit in higher-order logic. The behavior of the given circuit can be captured using the KCL as follows:

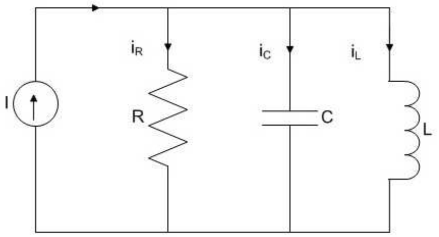

Figure 5.2: RLC Parallel Circuit

Definition 4: *RLC Parallel Circuit Model*

⊢ ∀ R L C I Io v t. rlc_parallel_circuit R L C I Io v t = kcl [resistor current R v; inductor current L v Io;
capacitor current C v; (λt. -I)] t

The list input of the function `kvl` is composed of all the elements of the circuit that have a current flow. The current source I is modeled in this list as a time independent constant. The next step in the proposed methodology is to obtain a differential equation representation of the given AMS circuit. We formally verified this relationship as the following theorem for the RLC circuit.

Theorem 4: *Differential Equation for the Parallel RLC Circuit*

```
⊢ ∀ R L C I Io v t y.  (0 < y ) ∧ (y < t) ∧
(∀x.  0 ≤ x ∧ x ≤ t ⇒ v differentiable x) ∧
(∀x.  0 ≤ x ∧ x ≤ t ⇒ ((λt.  deriv v t)) differentiable x) ∧
(rlc_pararllel_circuit R L C I Io v t) ⇒
(diff_eq_lhs (const_list [(1/L); (1/R); C]) v (3:num) y = 0))
```

The conclusion of the above theorem describes the second-order differential equation corresponding to the Paralllel RLC circuit given in the assumption using the function rlc_parallel_circuit. The theorem is verified under the assumptions that both the voltage drop v and its first derivative are differentiable. It is also important to note that the theorem is valid for all time y in the interval $(0, t)$, where t represents the upper bound of the time for which the behavior of the function rlc_parallel_circuit is valid. Theorem 14 has been verified using Theorems 4 and 11 along with some arithmetic reasoning.

Now, same as like series RLC circuit we have the differential equation, corresponding to our given circuit, and the next step in the proposed theorem proving based analysis of AMS circuits is to formally verify its solution. We verified the following theorem for the voltage of the given RLC circuit.

Theorem 5: *Voltage of the RLC Circuit*

```
⊢ ∀ L R C c1 c2 x.  (L ≠ 0) ∧ (C ≠ 0) ∧
(1/(R pow 2 * C) = 4/L) ⇒
(diff_eq (const_list [1 / (L*C); 1/(R*C); 1])
        (λx.  c1 * exp ((-1/2*R*C) * x) +
                c2 * (x * exp ((-1/2*R*C) * x)) 3 x = 0)
```

The conclusion of the above theorem provides us the solution $(c_1 e^{-\frac{1}{2RC}x} + c_2 x e^{-\frac{1}{2RC}x})$ of the differential equation obtained from Theorem 4. It is verified primarily using Theorem 5 of chapter 3 and some simple arithmetic reasoning. The assumptions of Theorem 5 same as like Theorem 3 declare the relationships between the various parameters that are required for the solution to hold. This is one of strengths of the proposed theorem proving based verification as all the assumptions have to be explicitly stated besides the theorem for its formal verification. Thus, there is no chance of missing a critical assumption which often occurs in paper-and-pencil proof methods where there is no such guarantee that the mathematician who worked out the proof has written down all the assumptions for the readers. It is also important to note the generic and continuous nature of Theorem 5 as all the variables are of type `real` and they are universally quantified. Such results cannot be obtained via state-based formal methods tools.

5.1.4 Second-order Op-Amp

As a second example, consider a second-order op-amp circuit (Figure 2), which has a wide range of applications in oscilators, filters, audio buffers and line drivers [25, 36].

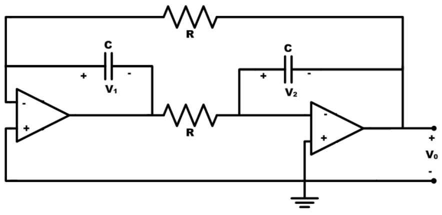

Figure 5.3: Second-order Op-Amp Circuit

The behavior of this circuit can be modeled as follows:

Definition 5: *Second-order Op-Amp Model*

```
⊢ ∀ R1 R2 C1 C2 v t.op_amp_ckt R1 R2 C1 C2 v t =
  (kcl [resistor_current R1 v; capacitor_current C1 v] t)∧
  (kcl [resistor_current R2 v; capacitor_current C2 v] t)
```

The function `kcl` is applied at the input of both op-amp and thus two predicates are required in the above definition, which distinguishes this application from the previous one. The functions `capacitor_current` and `resistor_current` provide the current expressions in terms of voltages for a capacitor and resistor, respectively, as given in Definition 3. The next step in the proposed methodology is to obtain a differential equation representation of the given AMS circuit. While doing so we take into consideration that both the resisters and the capacitors have the same value, respectively, as shown in Figure 2. Moreover, the input terminals of the op-ams are connected to the ground resulting in $v2 = -vo$.

Theorem 6: *Differential Equation for the Second-order OP-Amp*

```
⊢ ∀ R C vo v1 v2 t y.
(vo = -v2) ∧ (op_amp_ckt R R C C v t)⇒
    (diff_eq_lhs (const_list [-1/(R^2 * C^2); 0; 1])vo y = 0)
```

The proof steps of this theorem are quite similar to Theorem 2. The next step is to formally verify its solution, i.e., an expression for the voltage vo, which is verified as the following theorem.

Theorem 7: *Solution of Second-order Op-Amp Circuit Differential Equation*

```
⊢ ∀ R L c1 c2.  (R * C ≠ 0) ⇒
  (diff_eq_lhs (const_list [-1/(R^2 * C^2);0;1])
  (λx.  c1 * exp ((1/R*C) * x) + c2 * (x * exp ((1/R*C) * x)) x = 0))
```

The proof of the above theorem is based on Theorem 5 of chapter 3, which presents the formally verified solution of the homogeneous linear differential equation with real and repeated roots, along with some arithmetic reasoning, which can be done in an automatic manner using the HOL4 arithmetic simplifiers.

5.2 Biomedical Applications

Biomedical applications are one of the most safety-critical systems as their bugs could eventually result in the loss of human lives. Differential equations form the core foundation of modeling almost all biomedical applications [11]. Due to a lack of formal reasoning support for differential equation solutions, most of the biomedical system analysis have been conducted using informal analysis techniques so far. Our work tends to fill this gap and thus facilitates the usage of formal methods in this safety-critical domain. We present two simple case studies, i.e., the analysis of a heart pacemaker and a fluid-filled catheter, to illustrate the usefulness and effectiveness of our work in this section.

5.2.1 Heart Pacemaker

Electronic heart pacemakers are widely used these days for supplementing or replacing heart's natural pacing mechanism in humans. Their main principle is to store electrical energy in a capacitor and then discharge this energy in short pulses through the heart to provide it with the required sudden electrical stimulus. Besides the capacitor, they include a battery, which provides the energy source, and a switch to govern the charging and discharging of the capacitor. Figure 1 illustrates the connections between these main components and their working [11]. The capacitor is charged via the battery when the switch S is moved to position A, while the capacitor provides the short and intense pulses to the heart when the switch S is in position B.

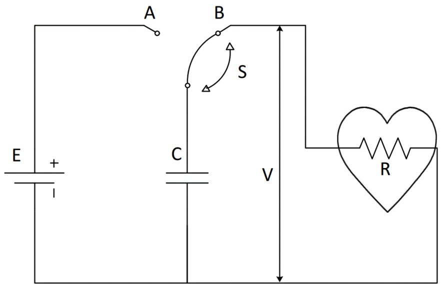

Figure 5.4: Equivalent Circuit of an Electronic Pacemaker

Based on Figure 1, the behavior of an electronic heart pacemaker can be described in terms of the following differential equation [11]:

$$\frac{dV}{dt} + \frac{1}{RC}V = 0,\ V(0) = E \tag{5.2}$$

since the current through the capacitor (CdV/dt) equals the current through the heart (V/R), which behaves as a resistor R, when the switch S is in position B. Moreover, the capacitor is allowed to charge to its full capacity when the switch is in position A and thus we obtain the initial condition $V(0) = E$. This simplistic but realistic mathematical model of a heart pacemaker has been extensively used in the literature to analyze the underlying properties of interest (See e.g., [39, 11]. In this thesis, we utilize our formalization described in the previous two sections to formally reason about the solution of Equation (5.2). The first step in this regard is to specify the theorem stating the solution ($Ee^{-\frac{t}{RC}}$) of Equation (5.2) as follows:

Theorem 1: *Solution of Heart Pacemaker Differential Equation*

```
⊢ ∀ R C V E t.
  (((λx.  linear_sol [C] (exp_list [-(1/(R*C))]) x) 0) = E) ⇒
  (diff_eq_lhs (const_fn_list [(1/(R*C)); 1])
    (λx.  linear_sol [C] (exp_list [-(1/(R*C))]) x) t = 0) ∧
      (C = E)
```

The assumption of the above theorem declares the given initial condition $V(0) = E$. The consequence of the goal is a conjunction of two propositions, where the first one defines the general solution of the given differential equation and the second one provides the value of the constant C for the particular solution.

Our formalized definitions facilitated the formal specification of the above theorem and our formally verified Theorem 4 allowed us to verify the above theorem in a few reasoning steps where we just had to provide the definitions of the functions used in Theorem 6 and some primitive list theory functions, like EL and LENGTH, along with invoking an automatic arithmetic simplifier. The straightforward reasoning process about the correctness of solution of the given differential equation in the sound environment of HOL4 clearly demonstrates the effectiveness of our work.

5.2.2 Fluid-Filled Catheter

As a second case study of our work, consider the dynamic analysis of a fluid-filled catheter, which allows physicians to measure the pressure of the internal organs and fluids of a human body without inserting a pressure transducer in the body. The main idea is to insert a long and small-bore fluid-filled tube or catheter in the body and thus bring the pressure of the pressure measuring site outside and then use a conventional pressure transducer to measure it. However, mechanical parameters like the mass of the catheter fluid and the friction of this fluid with the catheter wall may introduce some discrepancies in the pressure measurements. Therefore, it is very important to analyze the effects of such mechanical parameters on the pressure measurements as a wrong reading may endanger a patient's life. A number of studies, e.g. [22, 12], have analyzed this aspect by considering the following second-order linear differential equation with constant coefficients

$$\frac{1}{\omega_n^2}\frac{d^2p}{dt^2} + \frac{2\zeta}{\omega_n}\frac{dp}{dt} + p = 0 \tag{5.3}$$

where p is the applied pressure, $\omega_n = \sqrt{k/\rho LA}$ represents the undamped natural angular frequency (radians per unit time) in terms of a constant k, catheter fluid density ρ, length L and cross-sectional area A, and $\zeta = c/2\sqrt{1/\rho kLA}$ is the damping factor with a constant c. Equation (5.3) allows us to find the pressure in response to any force function given that the coefficients ω_n and ζ are known. The solution of this equation can be formally verified as the following theorem:

Theorem 2: *Solution of Fluid-Filled Catheter Differential Equation*

```
⊢ ∀ g A L k c C1 C2.
(4 * g * L * A * k < c pow 2 ∧ 0 < g ∧ 0 < L ∧ 0 < A ⇒
(diff_eq_lhs (const_fn_list [k / (g * L * A); c / (g * L * A); 1])
(λx.
 linear_sol [C1; C2]
  (exp_list
   [(-(c / (g * L * A)) +
    sqrt
     ((c / (g * L * A)) pow 2 -
      4 * (k / (g * L * A)))) / 2;
       (-(c / (g * L * A)) -
        sqrt
         ((c / (g * L * A)) pow 2 -
          4 * (k / (g * L * A)))) / 2]) x) x =
           0))
```

The assumptions of the above theorem declare the relationships between the various parameters that are required for the solution to hold. This is one of strengths of the proposed theorem proving based verification as all the assumptions have to be explicitly stated besides the theorem for its formal verification. Thus, there is no chance of missing a critical assumption which often occurs in paper-and-pencil proof methods where there is no such guarantee that the mathematician who worked out the proof has written down all the assumptions for the readers.

Formal reasoning about Theorem 2 is primarily based on Theorems 2 and 3 of chapter 4 along with some arithmetic reasoning, which can be done in an automatic manner using the HOL arithmetic simplifiers. The straightforward proof scripts for of Theorems 1 and 2 are available at [34] and clearly indicate the usefulness of our foundational formalization presented in Chapter 3 and Chapter 4 of this thesis. Just like these case studies our formalization results can be utilized to formally reason about solution of any homogeneous linear differential equation and the results would be guaranteed to be correct due to the inherent soundness of theorem proving.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

In this thesis, we presented a set of formal definitions that allow us to formalize any homogeneous linear differential equation. We also provide a couple of formally verified properties that facilitate reasoning about the solutions of these differential equations within the sound core of a higher-order-logic theorem prover HOL4. The almost automatic reasoning process makes our methodology very useful for industrial usage as its users do not have to very proficient with the cumbersome real-theoretic formal reasoning process. For illustration purpose, we applied the proposed methodology for verifying the solutions of differential equations related to a couple of biomedical systems. To the best of our knowledge, this is the first formal reasoning support for differential equations that has been reported in the open literature.

we also propose to use higher-order-logic theorem proving to analyze AMS circuits. Due to the high expressiveness of the underlying logic, we can formally analyze the AMS circuits along with their continuities and the soundness of theorem proving guarantees correctness of results. To the best of our knowledge, these features are not shared by any other existing AMS circuit analysis technique. The main challenge in the proposed approach is the enormous amount of user intervention required due to the undecidable nature of the logic. We propose to overcome this limitation by formalizing AMS circuit foundations and verifying associated properties that aid formal reasoning about the AMS circuit foundations. As a first step towards illustrating the proposed approach, this thesis presents the formalization of homogenous linear differential equations, Kirchhoff's voltage law and some basic components and some of their associated properties. Based on this work, we are able to formally verify the current relationship for a classical RLC circuit in a very straightforward way.

It is important to note that the formal framework, presented in this thesis, can verify solutions of differential equations but cannot guess solutions by itself. So the main objective of the work is to formally verify already obtained solutions via paper-and-pencil proof methods, numerical methods or computer algebra systems. Due to the inherent soundness of mechanical theorem proving this verification is guaranteed to be correct, i.e., a feature that cannot be attained by the above mentioned informal analysis techniques.

6.2 Future Work

Our work opens the doors to many new directions of research. By building upon the reported formalization, we are working on extending the support for formal reasoning about complex roots of homogeneous linear differential equations. Moreover, we also plan to develop reasoning support for non-homogeneous linear differential equations. To broaden the scope of AMS circuit verification and analyze a large variety of AMS circuits, other foundations, like the Kirchhoff's Current Law (KCL), and other frequently used analog components, like diodes and transistors, have to be formalized. The reported formalization can be used in many other domains besides the biomedical systems and Analog and Mixed Signal circuits presented in this thesis. Formal methods have been used in the verification of electronic circuits [38] and optical systems [19] but due to the lack of formal reasoning support for differential equations the solutions of differential equations have been handled by informal techniques in these efforts. Our work can be used to develop a complete formal methods based analysis methodology for these domains. For the cases where closed form mathematical solutions of differential equations cannot be obtained, our formalization can be used to formally verify error bounds for the numerical methods based solutions.

References

[1] E. Abraham-Mumm, M. Steffen, and U. Hannemann. Verification of hybrid systems: Formalization and proof rules in pvs. In *ICECCS*, pages 48–57, 2001.

[2] E. Abraham-Mumm, M. Steffen, and U. Hannemann. Assertion-based analysis of hybrid systems with pvs. In *EUROCAST*, pages 94–109, 2002.

[3] F.M. Ali, R. Nazar, and N.M. Arifin. Numerical Investigation of Free Convective Boundary Layer in a Viscous Fluid. *American Journal of Scientific Research*, 5:13–19, 2009.

[4] A.Mohameden. Ms Excel for higher Maths: Solving Initial Value Second order Ordinary Differential Equations on an Excel Spreadsheet. *MSOR Connections*, 8(4):21–25, 2009.

[5] S. Boldo, F. Clment, J. Fillitre, M. Mayero, G. Melquiond, and P. Weis. Formal proof of a Wave Equation resolution Scheme: The Method Error. In *Interactive Theorem Proving*, volume 6127 of *LNCS*, pages 147–162. Springer, 2010.

[6] A. Church. A Formulation of the Simple Theory of Types. *Journal of Symbolic Logic*, 5:56–68, 1940.

[7] Q. Cheng , T. J. Cui and C. Zhang. Waves in Planar Waveguide Containing Chiral Nihility Metamaterial. *Optics and Communication*, 274:317–321, 2007.

[8] W. Denman, B. Akbarpour, S. Tahar, M. Zaki, and L. C. Paulson. Formal Verification of Analog Designs using MetiTarski. In *Formal Methods in Computer Aided Design*, pages 93–100. IEEE, 2009.

[9] C.A. Desoer and Kuh E.S. *Basic Circuit Theory*. McGraw-Hill, 1969.

[10] G. Frehse, C. Le Guernic, A. Donzé, S. Cotton, R. Ray, O. Lebeltel, R. Ripado, A. Girard, T. Dang, and O. Maler. Spaceex: Scalable Verification of Hybrid Systems. In *Computer Aided Verification*, volume 6806 of *LNCS*, pages 379–395. Springer, 2011.

[11] S. A. Glantz. *Mathematics for Biomedical Applications.* University of California Press, 1979.

[12] S.A. Glantz and J. V. Tyberg. Determination of Frequency Response from Step Response: Application to Fluid-Filled Catheters. *The American Journal of Physiology*, 236:376–378, 1979.

[13] U. Goktas and D. Kapadia. Methods in Mathematica for Solving Ordinary Differential Equations. *Mathematical and Computational Application*, 16(4):784–796, 2011.

[14] M.J.C. Gordon and T.F. Melham. *Introduction to HOL: A Theorem Proving Environment for Higher-Order Logic.* Cambridge Press, 1993.

[15] G.Strang. *Calculus.* Wellesley College, second edition, 2009.

[16] J. Harrison. *Theorem Proving with the Real Numbers.* Springer-Verlag, 1998.

[17] J. Harrison. *Handbook of Practical Logic and Automated Reasoning.* Cambridge University Press, 2009.

[18] J. Harrison and L. Théry. A Skeptic's Approach to combining HOL and Maple. *Journal of Automated Reasoning*, 21:279–294, 1998.

[19] O. Hasan, S. K. Afshar, and S. Tahar. Formal Analysis of Optical Waveguides in HOL. In *Theorem Proving in Higher-Order Logics*, volume 5674 of *LNCS*, pages 228–243. Springer, 2009.

[20] T. J. Hickey. Analytic Constraint Solving and Interval Arithmetic. In *POPL'00*, pages 338–351, 2000.

[21] J. Hospodka and J. Bicak. Symbolic and Semisymbolic Analysis of Electronic Circuit in Maple. pages 128–131, 2011.

[22] J.O. Hougen, S.T. Hougen, and T.J. Hougen. Dynamics of Fluid-Filled Catheter Systems by Pulse Testing. In *Ind. Eng. Chem. Fundam*, volume 25, pages 462–470, 1986.

[23] F. Immler and J. Holzl. Numerical Analysis of Ordinary Differential Equations in Isabelle/HOL. In *Interactive Theorem Proving*, volume 7406 of *LNCS*, pages 377–392. Springer, 2012.

[24] E. M. Clarke Jr., O. Grumberg, and D. A. Peled. *Model Checking.* The MIT Press, 1999.

[25] K.Alexander and M.N. O. Sadiku. *Fundamentals of Electric Circuits.* McGraw-Hill, 2008.

[26] K. Kundert, H. Chang, D. Jefferies, G. Lamant, E. Malavasi, and F. Sending. Design of Mixed-Signal Systems on a Chip. *Computer-Aided Design Integrated Circuits Systems*, 19(12):1561–1571, 2000.

[27] J. Kyncl and M. Novotny. Education of Digital and Analog Circuits Supported by Computer Algebra System. volume 978, pages 341–344, 2011.

[28] T. Mei and J. Roychowdhury. Efficient AC Analysis of Oscillators using Least-Squares Methods. In *Design, Automation and Test in Europe*, pages 263–268, 2006.

[29] R. Milner. A Theory of Type Polymorphism in Programming. *Journal of Computer and System Sciences*, 17:348–375, 1977.

[30] A. Naqvi. Comments on Waves in Planar Waveguide Containing Chiral Nihility Metamaterial. *Optics and Communication*, 284:215–216, 2011.

[31] L.C. Paulson. *ML for the Working Programmer.* Cambridge University Press, 1996.

[32] A. Platzer. Differential dynamic Logics for Hybrid Systems. *Journal of Automated Reasoning*, 41(2):143–189, 2008.

[33] André Platzer and Edmund M. Clarke. Computing differential invariants of hybrid systems as fixedpoints. *Formal Methods in System Design*, 35(1):98–120, 2009.

[34] M.U. Sanwal. Formal Reasoning about Homogeneous Linear Differential Equations. http://save.seecs.nust.edu.pk/students/usman/lde.html, 2012.

[35] T. Stancheva. Differential Equations with MAPLE. In *Applications of Mathematics in Engineering And Economics*, volume 946 of *Conference Proceedings*, pages 176–187, 2007.

[36] W.Jung. *Op Amp Applications Handbook*. Newnes, 2004.

[37] Y.I.Ismail, E.G. Friedman, and J.L. Neves. Figuresof Merit to Characterize the Importance of On-Chip Inductance. *IEEE Transactions on Very Large Scale Integration(VLSI) Systems*, 7(4):442–449, 1999.

[38] M.H. Zaki, S. Tahar, and G. Bois. Formal Verification of Analog and Mixed Signal Designs: A Survey. *Microelectronics Journal*, 39(12):1395–1404, 2008.

[39] H. Zhang, J.H. Liu, and A.V. Holden. Computing the Age-Related Dysfunction of Cardiac Pacemaker. In *Computers in Cardiology*, volume 33, pages 665–668, 2006.

[40] D.G. Zill, W.S. Wright, and M.R. Cullen. *Advanced Engineering Mathematics*. Jones and Bartlett Learning, fourth edition, 2009.