Konrad Slind
Annette Bunker
Ganesh Gopalakrishnan (Eds.)

LNCS 3223

Theorem Proving in Higher Order Logics

17th International Conference, TPHOLs 2004
Park City, Utah, USA, September 2004
Proceedings

Springer

Konrad Slind Annette Bunker
Ganesh Gopalakrishnan (Eds.)

Theorem Proving in Higher Order Logics

17th International Conference, TPHOLs 2004
Park City, Utah, USA, September 14-17, 2004
Proceedings

Springer

Volume Editors

Konrad Slind
Ganesh Gopalakrishnan
University of Utah
School of Computing
50 South Central Campus Drive, Salt Lake City, Utah, UT84112, USA
E-mail: {slind;ganesh}@cs.utah.edu

Annette Bunker
Utah State University
Electrical and Computer Engineering Department
4120 Old Main Hill, Logan, UT 84341, USA
E-mail: bunker@helios.ece.usu.edu

Library of Congress Control Number: 2004111288

CR Subject Classification (1998): F.4.1, I.2.3, F.3.1, D.2.4, B.6.3

ISSN 0302-9743
ISBN 3-540-23017-3 Springer Berlin Heidelberg New York

Springer is a part of Springer Science+Business Media

springeronline.com

Printed in Germany

Typesetting: Camera-ready by author, data conversion by Olgun Computergrafik
Printed on acid-free paper SPIN: 11319238 06/3142 5 4 3 2 1 0

Lecture Notes in Computer Science 3223

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Preface

This volume constitutes the proceedings of the *17th International Conference on Theorem Proving in Higher Order Logics* (TPHOLs 2004) held September 14–17, 2004 in Park City, Utah, USA. TPHOLs covers all aspects of theorem proving in higher-order logics as well as related topics in theorem proving and verification.

There were 42 papers submitted to TPHOLs 2004 in the full research category, each of which was refereed by at least 3 reviewers selected by the program committee. Of these submissions, 21 were accepted for presentation at the conference and publication in this volume. In keeping with longstanding tradition, TPHOLs 2004 also offered a venue for the presentation of work in progress, where researchers invited discussion by means of a brief introductory talk and then discussed their work at a poster session. A supplementary proceedings containing papers about in-progress work was published as a 2004 technical report of the School of Computing at the University of Utah.

The organizers are grateful to Al Davis, Thomas Hales, and Ken McMillan for agreeing to give invited talks at TPHOLs 2004.

The TPHOLs conference traditionally changes continents each year in order to maximize the chances that researchers from around the world can attend. Starting in 1993, the proceedings of TPHOLs and its predecessor workshops have been published in the Springer Lecture Notes in Computer Science series:

1993 (Canada)	Vol. 780	1999 (France)	Vol. 1690
1994 (Malta)	Vol. 859	2000 (USA)	Vol. 1869
1995 (USA)	Vol. 971	2001 (UK)	Vol. 2152
1996 (Finland)	Vol. 1125	2002 (USA)	Vol. 2410
1997 (USA)	Vol. 1275	2003 (Italy)	Vol. 2758
1998 (Australia)	Vol. 1479	2004 (USA)	Vol. 3223

We would like to thank Amber Chisholm and Perry Hacker of University of Utah Conference Services for their help in many aspects of organizing and running TPHOLs.

Finally, we thank our sponsors: Intel and the National Science Foundation.

June 2004

Konrad Slind,
Annette Bunker,
and Ganesh Gopalakrishnan

Program Committee

Mark Aagaard (Waterloo)
David Basin (Zurich)
Ching-Tsun Chou (Intel)
Peter Dybjer (Chalmers)
Jean-Christophe Filliâtre (Paris Sud)
Mike Gordon (Cambridge)
Elsa Gunter (NJIT)
Jason Hickey (Caltech)
Doug Howe (Carleton)
Bart Jacobs (Nijmegen)
Matt Kaufmann (AMD)
Tom Melham (Oxford)
Tobias Nipkow (München)
Christine Paulin-Mohring (Paris Sud)
Frank Pfenning (CMU)
Sofiène Tahar (Concordia)

Clark Barrett (NYU)
Yves Bertot (INRIA)
Thierry Coquand (Chalmers)
Amy Felty (Ottawa)
Jacques Fleuriot (Edinburgh)
Jim Grundy (Intel)
John Harrison (Intel)
Peter Homeier (DoD, USA)
Paul Jackson (Edinburgh)
Sara Kalvala (Warwick)
Thomas Kropf (Bosch)
César Muñoz (NASA)
Sam Owre (SRI)
Lawrence Paulson (Cambridge)
Konrad Slind (Utah)
Burkhardt Wolff (Freiburg)

Additional Referees

Stefan Berghofer
Sylvain Conchon
Christophe Dehlinger
Lucas Dixon
Alfons Geser
Ali Habibi
Felix Klaedtke
Mohamed Layouni
Nicolas Magaud

Holger Pfeifer
Sylvan Pinsky
Tom Ridge
Norbert Schirmer
Carsten Schürmann
Radu I. Siminiceanu
Laurent Théry
Luca Viganò
Martin Wildmoser

Lecture Notes in Computer Science

For information about Vols. 1–3101

please contact your bookseller or Springer

Vol. 3232: R. Heery, L. Lyon (Eds.), Research and Advanced Technology for Digital Libraries. XV, 528 pages. 2004.

Vol. 3223: K. Slind, A. Bunker, G. Gopalakrishnan (Eds.), Theorem Proving in Higher Order Logic. VIII, 337 pages. 2004.

Vol. 3221: S. Albers, T. Radzik (Eds.), Algorithms – ESA 2004. XVIII, 836 pages. 2004.

Vol. 3220: J.C. Lester, R.M. Vicari, F. Paraguaçu (Eds.), Intelligent Tutoring Systems. XXI, 920 pages. 2004.

Vol. 3208: H.J. Ohlbach, S. Schaffert (Eds.), Principles and Practice of Semantic Web Reasoning. VII, 165 pages. 2004.

Vol. 3207: L.T. Jang, M. Guo, G.R. Gao, N.K. Jha, Embedded and Ubiquitous Computing. XX, 1116 pages. 2004.

Vol. 3206: P. Sojka, I. Kopecek, K. Pala (Eds.), Text, Speech and Dialogue. XIII, 667 pages. 2004. (Subseries LNAI).

Vol. 3205: N. Davies, E. Mynatt, I. Siio (Eds.), UbiComp 2004: Ubiquitous Computing. XVI, 452 pages. 2004.

Vol. 3203: J. Becker, M. Platzner, S. Vernalde (Eds.), Field Programmable Logic and Application. XXX, 1198 pages. 2004.

Vol. 3199: H. Schepers (Ed.), Software and Compilers for Embedded Systems. X, 259 pages. 2004.

Vol. 3198: G.-J. de Vreede, L.A. Guerrero, G. Marín Raventós (Eds.), Groupware: Design, Implementation and Use. XI, 378 pages. 2004.

Vol. 3194: R. Camacho, R. King, A. Srinivasan (Eds.), Inductive Logic Programming. XI, 361 pages. 2004. (Subseries LNAI).

Vol. 3193: P. Samarati, P. Ryan, D. Gollmann, R. Molva (Eds.), Computer Security – ESORICS 2004. X, 457 pages. 2004.

Vol. 3192: C. Bussler, D. Fensel (Eds.), Artificial Intelligence: Methodology, Systems, and Applications. XIII, 522 pages. 2004. (Subseries LNAI).

Vol. 3189: P.-C. Yew, J. Xue (Eds.), Advances in Computer Systems Architecture. XVII, 598 pages. 2004.

Vol. 3186: Z. Bellahsène, T. Milo, M. Rys, D. Suciu, R. Unland (Eds.), Database and XML Technologies. X, 235 pages. 2004.

Vol. 3184: S. Katsikas, J. Lopez, G. Pernul (Eds.), Trust and Privacy in Digital Business. XI, 299 pages. 2004.

Vol. 3183: R. Traunmüller (Ed.), Electronic Government. XIX, 583 pages. 2004.

Vol. 3182: K. Bauknecht, M. Bichler, B. Pröll (Eds.), E-Commerce and Web Technologies. XI, 370 pages. 2004.

Vol. 3181: Y. Kambayashi, M. Mohania, W. Wöß (Eds.), Data Warehousing and Knowledge Discovery. XIV, 412 pages. 2004.

Vol. 3180: F. Galindo, M. Takizawa, R. Traunmüller (Eds.), Database and Expert Systems Applications. XXI, 972 pages. 2004.

Vol. 3179: F.J. Perales, B.A. Draper (Eds.), Articulated Motion and Deformable Objects. XI, 270 pages. 2004.

Vol. 3178: W. Jonker, M. Petkovic (Eds.), Secure Data Management. VIII, 219 pages. 2004.

Vol. 3177: Z.R. Yang, H. Yin, R. Everson (Eds.), Intelligent Data Engineering and Automated Learning – IDEAL 2004. XVIII, 852 pages. 2004.

Vol. 3175: C.E. Rasmussen, H.H. Bülthoff, B. Schölkopf, M.A. Giese (Eds.), Pattern Recognition. XVIII, 581 pages. 2004.

Vol. 3174: F. Yin, J. Wang, C. Guo (Eds.), Advances in Neural Networks - ISNN 2004. XXXV, 1021 pages. 2004.

Vol. 3172: M. Dorigo, M. Birattari, C. Blum, L. M.Gambardella, F. Mondada, T. Stützle (Eds.), Ant Colony, Optimization and Swarm Intelligence. XII, 434 pages. 2004.

Vol. 3170: P. Gardner, N. Yoshida (Eds.), CONCUR 2004 - Concurrency Theory. XIII, 529 pages. 2004.

Vol. 3166: M. Rauterberg (Ed.), Entertainment Computing – ICEC 2004. XXIII, 617 pages. 2004.

Vol. 3163: S. Marinai, A. Dengel (Eds.), Document Analysis Systems VI. XII, 564 pages. 2004.

Vol. 3162: R. Downey, M. Fellows, F. Dehne (Eds.), Parameterized and Exact Computation. X, 293 pages. 2004.

Vol. 3160: S. Brewster, M. Dunlop (Eds.), Mobile Human-Computer Interaction – MobileHCI 2004. XVIII, 541 pages. 2004.

Vol. 3159: U. Visser, Intelligent Information Integration for the Semantic Web. XIV, 150 pages. 2004. (Subseries LNAI).

Vol. 3158: I. Nikolaidis, M. Barbeau, E. Kranakis (Eds.), Ad-Hoc, Mobile, and Wireless Networks. IX, 344 pages. 2004.

Vol. 3157: C. Zhang, H. W. Guesgen, W.K. Yeap (Eds.), PRICAI 2004: Trends in Artificial Intelligence. XX, 1023 pages. 2004. (Subseries LNAI).

Vol. 3156: M. Joye, J.-J. Quisquater (Eds.), Cryptographic Hardware and Embedded Systems - CHES 2004. XIII, 455 pages. 2004.

Vol. 3155: P. Funk, P.A. González Calero (Eds.), Advances in Case-Based Reasoning. XIII, 822 pages. 2004. (Subseries LNAI).

Vol. 3154: R.L. Nord (Ed.), Software Product Lines. XIV, 334 pages. 2004.

Vol. 3153: J. Fiala, V. Koubek, J. Kratochvíl (Eds.), Mathematical Foundations of Computer Science 2004. XIV, 902 pages. 2004.

Vol. 3152: M. Franklin (Ed.), Advances in Cryptology – CRYPTO 2004. XI, 579 pages. 2004.

Vol. 3150: G.-Z. Yang, T. Jiang (Eds.), Medical Imaging and Augmented Reality. XII, 378 pages. 2004.

Vol. 3149: M. Danelutto, M. Vanneschi, D. Laforenza (Eds.), Euro-Par 2004 Parallel Processing. XXXIV, 1081 pages. 2004.

Vol. 3148: R. Giacobazzi (Ed.), Static Analysis. XI, 393 pages. 2004.

Vol. 3146: P. Érdi, A. Esposito, M. Marinaro, S. Scarpetta (Eds.), Computational Neuroscience: Cortical Dynamics. XI, 161 pages. 2004.

Vol. 3144: M. Papatriantafilou, P. Hunel (Eds.), Principles of Distributed Systems. XI, 246 pages. 2004.

Vol. 3143: W. Liu, Y. Shi, Q. Li (Eds.), Advances in Web-Based Learning – ICWL 2004. XIV, 459 pages. 2004.

Vol. 3142: J. Diaz, J. Karhumäki, A. Lepistö, D. Sannella (Eds.), Automata, Languages and Programming. XIX, 1253 pages. 2004.

Vol. 3140: N. Koch, P. Fraternali, M. Wirsing (Eds.), Web Engineering. XXI, 623 pages. 2004.

Vol. 3139: F. Iida, R. Pfeifer, L. Steels, Y. Kuniyoshi (Eds.), Embodied Artificial Intelligence. IX, 331 pages. 2004. (Subseries LNAI).

Vol. 3138: A. Fred, T. Caelli, R.P.W. Duin, A. Campilho, D.d. Ridder (Eds.), Structural, Syntactic, and Statistical Pattern Recognition. XXII, 1168 pages. 2004.

Vol. 3137: P. De Bra, W. Nejdl (Eds.), Adaptive Hypermedia and Adaptive Web-Based Systems. XIV, 442 pages. 2004.

Vol. 3136: F. Meziane, E. Métais (Eds.), Natural Language Processing and Information Systems. XII, 436 pages. 2004.

Vol. 3134: C. Zannier, H. Erdogmus, L. Lindstrom (Eds.), Extreme Programming and Agile Methods - XP/Agile Universe 2004. XIV, 233 pages. 2004.

Vol. 3133: A.D. Pimentel, S. Vassiliadis (Eds.), Computer Systems: Architectures, Modeling, and Simulation. XIII, 562 pages. 2004.

Vol. 3132: B. Demoen, V. Lifschitz (Eds.), Logic Programming. XII, 480 pages. 2004.

Vol. 3131: V. Torra, Y. Narukawa (Eds.), Modeling Decisions for Artificial Intelligence. XI, 327 pages. 2004. (Subseries LNAI).

Vol. 3130: A. Syropoulos, K. Berry, Y. Haralambous, B. Hughes, S. Peter, J. Plaice (Eds.), TeX, XML, and Digital Typography. VIII, 265 pages. 2004.

Vol. 3129: Q. Li, G. Wang, L. Feng (Eds.), Advances in Web-Age Information Management. XVII, 753 pages. 2004.

Vol. 3128: D. Asonov (Ed.), Querying Databases Privately. IX, 115 pages. 2004.

Vol. 3127: K.E. Wolff, H.D. Pfeiffer, H.S. Delugach (Eds.), Conceptual Structures at Work. XI, 403 pages. 2004. (Subseries LNAI).

Vol. 3126: P. Dini, P. Lorenz, J.N.d. Souza (Eds.), Service Assurance with Partial and Intermittent Resources. XI, 312 pages. 2004.

Vol. 3125: D. Kozen (Ed.), Mathematics of Program Construction. X, 401 pages. 2004.

Vol. 3124: J.N. de Souza, P. Dini, P. Lorenz (Eds.), Telecommunications and Networking - ICT 2004. XXVI, 1390 pages. 2004.

Vol. 3123: A. Belz, R. Evans, P. Piwek (Eds.), Natural Language Generation. X, 219 pages. 2004. (Subseries LNAI).

Vol. 3122: K. Jansen, S. Khanna, J.D.P. Rolim, D. Ron (Eds.), Approximation, Randomization, and Combinatorial Optimization. IX, 428 pages. 2004.

Vol. 3121: S. Nikoletseas, J.D.P. Rolim (Eds.), Algorithmic Aspects of Wireless Sensor Networks. X, 201 pages. 2004.

Vol. 3120: J. Shawe-Taylor, Y. Singer (Eds.), Learning Theory. X, 648 pages. 2004. (Subseries LNAI).

Vol. 3118: K. Miesenberger, J. Klaus, W. Zagler, D. Burger (Eds.), Computer Helping People with Special Needs. XXIII, 1191 pages. 2004.

Vol. 3116: C. Rattray, S. Maharaj, C. Shankland (Eds.), Algebraic Methodology and Software Technology. XI, 569 pages. 2004.

Vol. 3115: P. Enser, Y. Kompatsiaris, N.E. O'Connor, A.F. Smeaton, A.W.M. Smeulders (Eds.), Image and Video Retrieval. XVII, 679 pages. 2004.

Vol. 3114: R. Alur, D.A. Peled (Eds.), Computer Aided Verification. XII, 536 pages. 2004.

Vol. 3113: J. Karhumäki, H. Maurer, G. Paun, G. Rozenberg (Eds.), Theory Is Forever. X, 283 pages. 2004.

Vol. 3112: H. Williams, L. MacKinnon (Eds.), Key Technologies for Data Management. XII, 265 pages. 2004.

Vol. 3111: T. Hagerup, J. Katajainen (Eds.), Algorithm Theory - SWAT 2004. XI, 506 pages. 2004.

Vol. 3110: A. Juels (Ed.), Financial Cryptography. XI, 281 pages. 2004.

Vol. 3109: S.C. Sahinalp, S. Muthukrishnan, U. Dogrusoz (Eds.), Combinatorial Pattern Matching. XII, 486 pages. 2004.

Vol. 3108: H. Wang, J. Pieprzyk, V. Varadharajan (Eds.), Information Security and Privacy. XII, 494 pages. 2004.

Vol. 3107: J. Bosch, C. Krueger (Eds.), Software Reuse: Methods, Techniques and Tools. XI, 339 pages. 2004.

Vol. 3106: K.-Y. Chwa, J.I. Munro (Eds.), Computing and Combinatorics. XIII, 474 pages. 2004.

Vol. 3105: S. Göbel, U. Spierling, A. Hoffmann, I. Iurgel, O. Schneider, J. Dechau, A. Feix (Eds.), Technologies for Interactive Digital Storytelling and Entertainment. XVI, 304 pages. 2004.

Vol. 3104: R. Kralovic, O. Sykora (Eds.), Structural Information and Communication Complexity. X, 303 pages. 2004.

Vol. 3103: K. Deb, e. al. (Eds.), Genetic and Evolutionary Computation – GECCO 2004. XLIX, 1439 pages. 2004.

Vol. 3102: K. Deb, e. al. (Eds.), Genetic and Evolutionary Computation – GECCO 2004. L, 1445 pages. 2004.

Table of Contents

Error Analysis of Digital Filters Using Theorem Proving

Behzad Akbarpour and Sofiène Tahar

Dept. of Electrical & Computer Engineering, Concordia University
1455 de Maisonneuve W., Montreal, Quebec, H3G 1M8, Canada
{behzad,tahar}@ece.concordia.ca

Abstract. When a digital filter is realized with floating-point or fixed-point arithmetics, errors and constraints due to finite word length are unavoidable. In this paper, we show how these errors can be mechanically analysed using the HOL theorem prover. We first model the ideal real filter specification and the corresponding floating-point and fixed-point implementations as predicates in higher-order logic. We use valuation functions to find the real values of the floating-point and fixed-point filter outputs and define the error as the difference between these values and the corresponding output of the ideal real specification. Fundamental analysis lemmas have been established to derive expressions for the accumulation of roundoff error in parametric Lth-order digital filters, for each of the three canonical forms of realization: direct, parallel, and cascade. The HOL formalization and proofs are found to be in a good agreement with existing theoretical paper-and-pencil counterparts.

1 Introduction

Signal processing through digital techniques has become increasingly attractive with the rapid technological advancement in digital integrated circuits, devices, and systems. The availability of large scale general purpose computers and special purpose hardware has made real time digital filtering both practical and economical. Digital filters are a particularly important class of DSP (Digital Signal Processing) systems. A digital filter is a discrete time system that transforms a sequence of input numbers into another sequence of output, by means of a computational algorithm [13]. Digital filters are used in a wide variety of signal processing applications, such as spectrum analysis, digital image and speech processing, and pattern recognition. Due to their well-known advantages, digital filters are often replacing classical analog filters. The three distinct and most outstanding advantages of the digital filters are their flexibility, reliability, and modularity. Excellent methods have been developed to design these filters with desired characteristics. The design of a filter is the process of determination of a transfer function from a set of specifications given either in the frequency domain, or in the time domain, or for some applications, in both. The design of a digital filter starts from an ideal real specification. In a theoretical analysis of the digital filters, we generally assume that signal values and system coefficients

K. Slind et al. (Eds.): TPHOLs 2004, LNCS 3223, pp. 1–17, 2004.

are represented in the real number system and are expressed to an infinite precision. When implemented as a special-purpose digital hardware or as a computer algorithm, we must represent the signals and coefficients in some digital number system that must always be of a finite precision. Therefore, arithmetic operations must be carried out with an accuracy limited by this finite word length. There is a variety of types of arithmetic used in the implementation of digital systems. Among the most common are the floating-point and fixed-point. Here, all operands are represented by a special format or assigned a fixed word length and a fixed exponent, while the control structure and the operations of the ideal program remain unchanged. The transformation from the real to the floating-point and fixed-point forms is quite tedious and error-prone. On the implementation side, the fixed-point model of the algorithm has to be transformed into the best suited target description, either using a hardware description or a programming language. This design process can be aided by a number of specialized CAD tools such as SPW (Cadence) [3], CoCentric (Synopsys) [20], Matlab-Simulink (Mathworks) [16], and FRIDGE (Aachen UT) [22].

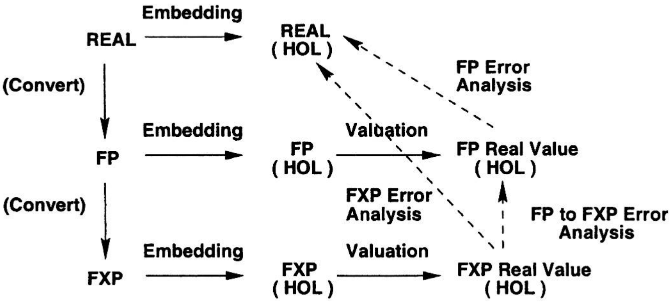

Fig. 1. Error analysis approach

In this paper we describe the error analysis of digital filters using the HOL theorem proving environment [5] based on the commutating diagram shown in Figure 1. Thereafter, we first model the ideal real filter specification and the corresponding floating-point and fixed-point implementations as predicates in higher-order logic. For this, we make use of existing theories in HOL on the construction of real numbers [7], the formalization of IEEE-754 standard based floating-point arithmetic [8, 9], and the formalization of fixed-point arithmetic [1, 2]. We use valuation functions to find the real values of the floating-point and fixed-point filter outputs and define the errors as the differences between these values and the corresponding output of the ideal real specification. Then we establish fundamental lemmas on the error analysis of the floating-point and fixed-point roundings and arithmetic operations against their abstract mathematical counterparts. Finally, we use these lemmas as a model to derive expressions for the accumulation of the roundoff error in parametric Lth-order digital filters, for each of the three canonical forms of realization: direct, parallel, and cascade [18].

Using these forms, our verification methodology can be scaled up to any larger-order filter, either directly or by decomposing the design into a combination of internal sub-blocks. While the theoretical work on computing the errors due to finite precision effects has been extensively studied since the late sixties [15], it is for the first time in this paper, that a formalization and proof of this analysis for digital filters is done using a mechanical theorem prover, here the HOL. Our results are found to be in a good agreement with the theoretical ones.

The rest of this paper is organized as follows: Section 2 gives a review of the related work. Section 3 introduces the fundamental lemmas in HOL for the error analysis of the floating-point and fixed-point rounding and arithmetic operations. Section 4 describes the details of the error analysis in HOL of the class of linear difference equation digital filters implemented in the three canonical forms of realization. Finally, Section 5 concludes the paper.

2 Related Work

Work on the analysis of the errors due to the finite precision effects in the realization of the digital filters has always existed since their early days, however, using theoretical paper-and-pencil proofs and simulation techniques. For digital filters realized with the fixed-point arithmetic, error problems have been studied extensively. For instance, Knowles and Edwards [14] proposed a method for analysis of the finite word length effects in fixed-point digital filters. Gold and Radar [6] carried out a detailed analysis of the roundoff error for the first-order and second-order fixed-point filters. Jackson [12] analyzed the roundoff noise for the cascade and parallel realizations of the fixed-point digital filters. While the roundoff noise for the fixed-point arithmetic enters into the system additively, it is a multiplicative component in the case of the floating-point arithmetic. This problem is analyzed first by Sandberg [19], who discussed the roundoff error accumulation and input quantization effects in the direct realization of the filter excited by a deterministic input. He also derived a bound on the time average of the squared error at the output. Liu and Kaneko [15] presented a general approach to the error analysis problem of digital filters using the floating-point arithmetic and calculated the error at the output due to the roundoff accumulation and input quantization. Expressions are derived for the mean square error for each of the three canonical forms of realization: direct, cascade, and parallel. Upper bounds that are useful for a special class of the filters are given. Oppenheim and Weinstein [17] discussed in some details the effects of the finite register length on implementations of the linear recursive difference equation digital filters, and the fast Fourier transform (FFT) algorithm. Comparisons of the roundoff noise in the digital filters using the different types of arithmetics have also been reported in [21].

In order to validate the error analysis, most of the above work compare the theoretical results with corresponding experimental simulations. In this paper, we show how the above error analysis can be mechanically performed using the HOL theorem prover, providing a superior approach to validation by simulation.

Our focus will be on the process of translating the hand proofs into equivalent proofs in HOL. The analysis we propose is mostly inspired by the work done by Liu and Kaneko [15], who defined a general approach to the error analysis problem of digital filters using the floating-point arithmetic. Following a similar approach, we have extended this theoretical analysis for fixed-point digital filters. In both cases, a good agreement between the HOL formalized and the theoretical results are obtained.

Through our work, we confirmed and strengthened the main results of the previously published theoretical error analysis, though we uncovered some minor errors in the hand proofs and located a few subtle corners that are overlooked informally. For example, in the theoretical fixed-point error analysis it is always assumed that the fixed-point addition causes no error and only the roundoff error in the fixed-point multiplication is analyzed [17]. This is under the assumption that there is no overflow in the result and also the input operands have the same attributes as the output. Using a mechanical theorem prover, we provide a more general error analysis in which we cover the roundoff errors in both the fixed-point addition and multiplication operations. On top of that, for the floating-point error analysis, we have used the formalization in HOL of the IEEE-754 [8], a standard which has not yet been established at the time of the above mentioned theoretical error analysis. This enabled us to cover a more complete set of rounding and overflow modes and degenerate cases which are not discussed in earlier theoretical work.

Previous work on the error analysis in formal verification was done by Harrison [9] who verified the floating-point algorithms such as the exponential function against their abstract mathematical counterparts using the HOL Light theorem prover. As the main theorem, he proved that the floating-point exponential function has a correct overflow behavior, and in the absence of overflow the error in the result is bounded to a certain amount. He also reported on an error in the hand proof mostly related to forgetting some special cases in the analysis. This error analysis is very similar to the type of analysis performed for DSP algorithms. The major difference, however, is the use of statistical methods and mean square error analysis for DSP algorithms which is not covered in the error analysis of the mathematical functions used by Harrison. In this method, the error quantities are treated as independent random variables uniformly distributed over a specific interval depending on the type of arithmetic and the rounding mode. Then the error analysis is performed to derive expressions for the variance and mean square error. To perform such an analysis in HOL, we need to develop a mechanized theory on the properties of random variables and random processes. This type of analysis is not addressed in this paper and is a part of our work in progress. Huhn *et al.* [11] proposed a hybrid formal verification method combining different state-of-the-art techniques to guide the complete design flow of imprecisely working arithmetic circuits starting at the algorithmic down to the register transfer level. The usefulness of the method is illustrated with the example of the discrete cosine transform algorithms. In particular, the authors have shown the use of computer algebra systems like Mathematica or Maple at the algorithmic level to reason about real numbers and to determine certain error bounds for the results of numerical operations. In contrast to [11], we propose an error analysis for digital filters using the HOL theorem prover. Although the computer algebraic systems such as Maple or Mathematica are much more popular and have many powerful decision procedures and heuristics, theorem provers are more expressive, more precise, and more reliable [10]. One option is to combine the rigour of the theorem provers with the power of computer algebraic systems as proposed in [10].

3 Error Analysis Models

In this section we introduce the fundamental error analysis theorems [23, 4], and the corresponding lemmas in HOL for the floating-point [8, 9] and fixed-point [1, 2] arithmetics. These theorems are then used in the next sections as a model for the analysis of the roundoff error in digital filters.

3.1 Floating-Point Error Model

In analyzing the effects of floating-point roundoff, the effects of rounding will be represented multiplicatively. The following theorem is the most fundamental in the floating-point rounding-error theory [23, 4].

Theorem 1: If the real number x located within the floating-point range, is rounded to the closest floating-point number x_R, then

$$x_R = x(1+\delta), \text{ where } |\delta| \leq 2^{-p} \tag{1}$$

and p is the precision of the floating-point format.

In HOL, we proved this theorem in the IEEE single precision floating-point format for the case of rounding to nearest as follows:

```
Lemma 1: FLOAT_ROUND_RELATIVE_ERROR
⊢  normalizes x ⟹  ∃ e. abs (e) < (1 / 2 pow ((fracwidth X) + 1)) ∧
   (Val (float (round X To_nearest x)) = x * (1 + e))
```

where the function *normalizes* defines the criteria for an arbitrary real number to be in the normalized range of floating-point numbers [8], *fracwidth* extracts the fraction width parameter from the floating-point format X, *Val* is the floating-point valuation function, *float* is the bijection function that converts a triple of natural numbers into the floating-point type, and *round* is the floating-point rounding function [9].

To prove this theorem [4], we first proved the following lemma which locates a real number in a binade (the floating-point numbers between two adjacent powers of 2):

```
Lemma 2: REAL_IN_BINADE
⊢  normalizes x ⟹  ∃ j. j ≤ ((emax X) − 2) ∧
   (2 pow (j + 1) / 2 pow (bias X)) ≤ abs x ∧
   abs x < (2 pow (j + 2) / 2 pow (bias X))
```

where the function *emax* defines the maximum exponent in a given floating-point format, and *bias* defines the exponent bias in the floating-point format which is a constant used to make the exponent's range nonnegative. Using this lemma we can rewrite the general floating-point absolute error bound theorem (ERROR_BOUND_NORM_STRONG) developed in [9] as follows:

```
Lemma 3: ERROR_BOUND_NORM_STRONG_NORMALIZE
⊢  normalizes x ⟹
     ∃ j. abs (error x) ≤ (2 pow j / 2 pow (bias X + fracwidth X))
```

which states that if the absolute value of a real number is in the representable range of the normalized floating-point numbers, then the absolute value of the error is less than or equal to $2^j/2^{(bias\ X\ +\ fracwidth\ X)}$. The function *error*, defines the error resulting from rounding a real number to a floating-point value which is defined as follows [9]:

```
⊢def  error x = (Val (float (round X To_nearest x)) − x)
```

Since $(2^{(j+1)}\ /\ 2^{(bias\ X)}) \leq |x|$ for the real numbers in the normalized region as proved in Lemma 2, we have $(|error\ x|\ /\ |x|) \leq (2^j\ /\ 2^{(bias\ X\ +\ fracwidth\ X)})\ /\ (2^{(j+1)}\ /\ 2^{(bias\ X)})$ or $(|error\ x|\ /\ |x|) \leq (1\ /\ 2^{((fracwidth\ X)\ +\ 1)})$. Finally, defining $e = (error\ x\ /\ x)$ will complete the proof of the floating-point relative error bound theorem as described in Lemma 1.

Next, we apply the floating-point relative rounding error analysis theorem (Theorem 1) to the verification of the arithmetic operations. The goal is to prove the following theorem in which floating-point arithmetic operations such as addition, subtraction, multiplication, and division are related to their abstract mathematical counterparts according to the corresponding errors.

Theorem 2: Let $*$ denote any of the floating-point operations +, -, × , /. Then

$$fl\ (x * y) = (x * y)(1 + \delta), \text{ where } |\delta| \leq 2^{-p} \tag{2}$$

and p is the precision of the floating-point format. The notation *fl (.)* is used to denote that the operation is performed using the floating-point arithmetic.

To prove this theorem in HOL, we start from the already proved lemmas on the absolute analysis of rounding error in the floating-point arithmetic operations (FLOAT_ADD) developed in [9]. We have converted these lemmas to the following relative error analysis version, using the relative error bound analysis of floating-point rounding (Lemma 1):

```
Lemma 4: FLOAT_ADD_RELATIVE
⊢  Finite a ∧ Finite b ∧ normalizes (Val a + Val b)
     ⟹ Finite (a + b) ∧ ∃ e. abs e ≤ (1 / 2 pow ((fracwidth X) + 1))
         ∧ (Val (a + b) = (Val a + Val b) * (1 + e))
```

where the function *Finite* defines the finiteness criteria for the floating-point numbers. Note that we use the conventional symbols for arithmetic operations on floating-point numbers using the operator overloading in HOL.

3.2 Fixed-Point Error Model

While the rounding error for the floating-point arithmetic enters into the system multiplicatively, it is an additive component for the fixed-point arithmetic. In this case the fundamental error analysis theorem can be stated as follows [23].

Theorem 3: If the real number x located in the range of the fixed-point numbers with format *X'*, is rounded to the closest fixed-point number x'_R, then

$$x'_R \ = \ x \ + \ \epsilon, \quad \text{where} \ |\epsilon| \ \leq \ 2^{-fracbits\ (X')} \tag{3}$$

and *fracbits* is a function that extracts the number of bits that are to the right of the binary point in the given fixed-point format.

This theorem is proved in HOL as follows [1]:

```
Lemma 5: FXP_ROUND_ABSOLUTE_ERROR_BOUND
⊢  (validAttr X′) ∧ (representable X′ x) ⟹
   abs (Fxp_error X′ x) ≤ (1 / 2 pow (fracbits X′))
```

where the function *validAttr* defines the validity of the fixed-point format, *representable* defines the criteria for a real number to be in the representable range of the fixed-point format, and *Fxp_error* defines the fixed-point rounding error.

The verification of the fixed-point arithmetic operations using the *absolute* error analysis of the fixed-point rounding (Theorem 3) can be stated as in the following theorem in which the fixed-point arithmetic operations are related to their abstract mathematical counterparts according to the corresponding errors.

Theorem 4: Let $*$ denote any of the fixed-point operations +, -, × , /, with a given format *X'*. Then

$$fxp\ (x \ * \ y) \ = \ (x \ * \ y) \ + \ \epsilon, \quad \text{where} \ |\epsilon| \ \leq \ 2^{-fracbits\ (X')} \tag{4}$$

and the notation *fxp (.)* is used to denote that the operation is performed using the fixed-point arithmetic. This theorem is proved in HOL using the following lemma [1]:

```
Lemma 6: FXP_ADD_ABSOLUTE
⊢  (Isvalid a) ∧ (Isvalid b) ∧ validAttr (X′) ∧
   representable X′ (value a + value b) ⟹  (Isvalid (FxpAdd X′ a b)) ∧
   ∃ e. abs e ≤ (1 / 2 pow (fracbits X′)) ∧
   value (FxpAdd X′ a b) = (value a + value b) + e
```

where *Isvalid* defines the validity of a fixed-point number, *value* is the fixed-point valuation, and *FxpAdd* is the fixed-point addition.

4 Error Analysis of Digital Filters in HOL

In this section, the principal results for the roundoff accumulation in digital filters using the mechanized theorem proving are derived and summarized. We