Expanded Final Report for NSF Grant MIP87-07067

Digital Design Derivation

(November 30, 1990)

Steven D. Johnson, P.I.

Computer Science Department
Indiana University
Bloomington, IN 47405-4101 USA
<sjohnson@iuvax.cs.indiana.edu>

Contents

II. Summary of Completed Project

With a growing dependence on computer-aided design (CAD) tools, digital system design benefits from rigorous formalizations of its methodology. Such formalizations are needed not only to secure more confidence in individual designs, but also to provide a basis for classifying design tools. This research employed functional modeling theory to describe logical aspects of design and elements of high-level VLSI synthesis. Software oriented theories of data-type abstraction were adapted to address qualities such as architecture, physical orgainzation and distributed control. Although outstanding issues remain, these results lead to a treatment of abstraction which is more general than that provided by current CAD tools.

The algebra of functional expressions was mechanized in order to explore the formalism at practical levels of detail. A system for digital design derivation was implemented. It is used interactively to transform applicative specifications into hierarchical boolean systems, to which low-level design tools are then applied. Several moderate-scale (20,000–50,000 transistors) designs were derived, and two of these were incorporated in working prototype systems. Experimentation focused on the relationship between automated verification and formalized synthesis. It was demonstrated that the complexity of mechanized correctness proofs is greatly diminished when an integrated combination of reasoning systems is used.

III. Technical Information

Prior to this grant we had developed a general approach to formalizing design methods based on a purely functional modeling language and its algebra. We had begun implementing a transformation system for deriving digital implementations in this mathematical framework. The priority for this project—and the expressed concern of its reviewers—was to demonstrate the practical potential of the approach. We placed equal emphasis on the investigation of mathematical issues and the development of mechanized capabilities to produce working circuitry.

Infrastructure for this research was supported, in part, under NSF grant DCR 85-21497, a CER project. This research project is continuing under NSF grant MIP89-21842, entitled, "Algebra for Digital Design Derivation."

III.1. Formal Description of Digital Systems

One of the main topics for this project was the treatment of data-type hierarchies in hardware description. In [ZJ91, ZJ90, ZJ89] we propose extensions to the classical formulation of first-order algebraic data types to reflect structural aspects of hardware description. The definition of a many sorted algebra is augmented by a formal representation of architectural constraints. We develop the semantics of these extended structures, and we determine conditions under which the conventional notion of data-type implementation transfers to the extended structures. We present a procedure for making control transformations to satisfy an architectural constraint (but also observe that, in general, such transformations may depend on undecidable properties of the underlying equational theories).

Algebra for structural manipulation is reduced to a collection of elementary laws in [J89]. The *system factorizations* discussed in that paper provide for the decomposition of system descriptions into cooperating processes.

This work represents significant progress toward our goal of manipulating data abstraction (hence processes) in hardware specifications. It paves the way for the inclusion of user-defined types in a hardware description language—something no current high-level synthesis system fully supports. However, the important components of timing and protocol have yet to be captured in the formalism. We are working on these problems now.

III.2. System Development

Software sources and documentation developed under this project are publically available by request, provided the rights of its developers are acknowledged. However, since these systems are under continual development, we do not post them to public (FTP) directories. Contact Steven D. Johnson at the address given on the title page for information.

The *DDD transformation system* was developed to help us explore and demonstrate the formalization [DDD, JB90, JB89]. The system is implemented in Scheme, a Lisp dialect. It will probably run on any standard Scheme implementation, but we use Chez Scheme, a product of Cadence Research, Inc. DDD system is used interactively to reduce higher level specifications into hierarchical boolean systems, to which logic synthesis tools (e.g.

the Berkeley CAD suite) are then applied. It is seeing increasing use as a design tool in our department; and as a consequence, its capabilities are rapidly evolving.

We consider algebraic derivation to be a candidate framework for high-level synthesis, but DDD is not a synthesis system in the engineering sense because it does not automatically "explore the design space." It's primary purpose is to secure the sometimes massive transformations needed in manipulating design descriptions. The optimiztions of a synthesis system would be coded at a higher level. Thus, DDD has more in common with a basic theorem proving system, which automates the elementary inference rules of a logical calculus but requires guidance to perform a proof. DDD automates the formula manipulations needed to conduct design by derivation, assuring correct application of the algebra, but it sometimes requires detailed guidance to perform a design.

The most significant experimental application of DDD is to Hunt's mechanically verified FM8501 microprocessor. This is described below in Section III.4.

We have also designed a generic bit-slice data-path for rapid VLSI prototyping [RCWJ, RCW]. The data-path is specified by a *register-transfer table* (RTT)—one of DDD's intermediate representations. It implements registers and common combinational operations with a multiple-bus organization, so that the degree of parallel data transfer is programmable. The registers also have a serial shift capability, which is intended to support a testing discipline. The RTT generater automatically assembles individual bit slices of the VLSI data path and its shared control PLA. Currently, the bit-slice and control modules are connected manually or by an automatic router in `magic`. A test instance of the generic has been fabricated and successfully tested. We have generated RTT data-paths for several of our experimental projects but have not yet fabricated any of these circuits [JB91].

III.3. Other Results

The inherent directionality of functions is a possible disadvantage of using functional calculus as a modeling theory for hardware. At the gate level, bidirectional phenomena (e.g. buses) present little difficulty, while at the switch level, truly bidirectional transistor networks are more of a problem. And for analog circuits a full relational calculus may be necessary. In [JBB87] we present several techniques for modeling near the practical threshold of functional methodology. Animation of the results was done using *Daisy*, a "lazy" functional language developed by Johnson under previous NSF funding [DAISY]. Daisy is implemented in 'C' on a variety of UNIX hosts, and is publically available. Contact Steven D. Johnson at the address given on the title page.

III.4. Experimental Projects

Using DDD, a number of moderate-scale design exercises were done in the course of this project. Two of these projects were realized in VLSI technologies. Two others were implemented with PLD devices. The remaining projects were reduced to VLSI layouts but not fabricated. A consolidation is in progress of the draft project reports cited below. Appendix B gives more details about the first two projects.

- *A Stop-and-Copy Garbage Collector.* Two version were derived. The first, realized using PLD and MSI components, was integrated for testing purposes with a production implementation of Scheme, and functioned correctly on initial startup [JBB87]. A second, more serial version, was realized in VLSI using PLA technology. This set of MOSIS chips did not function correctly, owing to an electrical problem that we were unable to isolate. A complete switch-level simulation showed that the problem was *not* in the logical design generated by DDD [BJ89], but was due either to problems with lower level tools or (more likely) fabrication.

- *A simple LISP computer*, based on a published description of the SECD processor, together with its garbage collector and serial I/O subsystem, was derived in DDD, and realized using PLD and MSI components. The result was a working computer system running compiled Lisp code [W89].
- In [JBW90], the DDD system was applied to Hunt's mechanically verified FM8501 microprocessor description (Warren A. Hunt, Jr., *FM8501: A Verified Microprocessor*, Ph.D. Thesis, Univ. of Texas at Austing, 1985). This article illustrates a fruitful interaction between mechanical reasoning and mechanical synthesis, made possible by the fact that the two systems operate on the same formal representation. Such an interaction is already practical for boolean level tools, but at higher levels of abstraction it has hardly been studied. The FM8501 derivation was reduced to PLA layouts but was not fabricated. In a subsequent project, PLA and standard-cell layouts of a more advanced design, Hunt's FM8502, were generated [BCHS]. A VLSI realization of FM8502, or a later version, will be done in conjunction with Bose's Ph.D. research.
- *The LIBRA Prolog processor*, designed by colleague Jonathon Mills, was reduced to PLA layouts as a seminar project [CRWZ].
- *A prototype SCHEME processor*, designed by David Boyer, was reduced to PLA layouts as a seminar project [TMB].
- *A programmable data-path generic.* See the description in Section III.2.

III.5. Impact on Educational Infrastructure

This project has been instrumental in enhancing the infrastructure for hardware related research at Indiana. It has been the primary impetus for the acquisition of CAD tools for logic synthesis. It has provided the motivation for a large number of projects in system design. During this grant period, seven graduate students were introduced to the area of formal methods through seminar participation; and of these it appears that at least four wish to continue working in the area of hardware. We have also contributed to two collaborative projects, Mill's LIBRA design, mentioned above, and a specialized heap-memory project directed by colleague David Wise. Finally, researchers in this project were the first to use a new generation of the Logic Engine—a hardware system prototyping vehicle developed at Indiana.

Refereed and Invited* Publications

Listed articles published after August 1990 represent work which was done, in whole or in part, during the grant period funded under MIP87-07067.

[JB91] STEVEN D. JOHNSON AND BHASKAR BOSE. "DDD – A System for Mechanized Digital Design Derivation," accepted for presentation at the ACM/SIGDA International Workshop on Formal Methods in VLSI Design, Miami, Florida, January 9–11, 1991.

[ZJ91] ZHENG ZHU AND STEVEN D. JOHNSON. "An Example of Interactive Hardware Transformation," accepted for presentation at the ACM/SIGDA International Workshop on Formal Methods in VLSI Design, Miami, Florida, January 9–11, 1991.

[ZJ90] ZHENG ZHU AND STEVEN D. JOHNSON. "An Algebraic Framework for Data Abstraction in Hardware Description," to appear, G. Jones and M. Sheeran (eds.), proceedings of the Oxford PRG workshop on *Designing Correct Circuits (DCC90),* Oxford, September 1990.

[JBW90] STEVEN D. JOHNSON, BHASKAR BOSE AND ROBERT M. WEHRMEISTER. "On the Interplay of Synthesis and Verification, Experiments with the FM8501 Microprocessor Description," in: L. Claesen (ed.), *Applied Formal Methods for Correct VLSI Design* (Proceedings of the IMEC-IFIP International Workshop, Leuven, Belgium, November 1989), North-Holland, 1990.

[ZJ89] ZHENG ZHU AND STEVEN D. JOHNSON. "An Algebraic Characterization of Structural Synthesis for Hardware," in: L. Claesen (ed.), *Applied Formal Methods for Correct VLSI Design* (Proceedings of the IMEC-IFIP International Workshop, Leuven, Belgium, November 1989), North-Holland, 1990.

[JB89] STEVEN D. JOHNSON AND BHASKAR BOSE. "A system for Digital Design Derivation," Indiana University Computer Science Dept. Technical Report No. 289 (August 1989). Refereed abstract presented at the 1989 IEEE High-level Synthesis Workshop, Kennebunkport, Maine.

[J89] STEVEN D. JOHNSON. "Manipulating Logical Organization with System Factorizations*," invited (nonrefereed) paper in: M. Leeser and G. Brown (eds.) *Hardware Specification, Verification and Synthesis: Mathematical Aspects,* (Proceedings of the Mathematical Sciences Institute Workshop, Cornell University, Ithaca, New York, USA, July 1989, Volume 408, *Lecture Notes in Computer Science*, series editors G. Goos and J. Hartmanis, Springer, Berlin, 1990.

[BJ89] C. DAVID BOYER AND STEVEN D. JOHNSON. "Using the Digital Design Derivation System: Case Study of a VLSI Garbage Collector Implementation, in J. Darringer and F. Ramming (eds.), *Computer Hardware Description Languages and their Applications*, Proceedings of the IFIP WG 10.2 Ninth International Symposium, North Holland, 1990.

[JB88] STEVEN D. JOHNSON AND C. DAVID BOYER. "Modeling Transistors Applicatively," in George J. Milne (ed.), *The Fusion of Hardware Design and Verification,* North-Holland, 1988, 397–420 (proceedings of the IFIP WG10.2 working conference on hardware, Glasgow, July, 1988).

[JBB87] STEVEN D. JOHNSON, BHASKAR BOSE, AND C. DAVID BOYER. "A Tactical Framework for Digital Design*," invited (nonrefereed), in *VLSI Specification Verification and Synthesis,* Graham Birtwistle and P. A. Subramanyam (eds.), Kluwer, 1987, pp. 349–384. *This paper was accepted prior to the proposal.*

Nonrefereed Publications and Reports

[RCWJ] KAMLESH RATH, IGNACIO CELIS, ROBERT M. WEHRMEISTER AND STEVEN D. JOHNSON. "RTBA: An Open Ended Bus Architecture for VLSI Implementation of Register Transfer Equations," submitted.

[W89] ROBERT M. WEHRMEISTER. "Derivation of an SECD Machine: Experience with a Transformational Approach to Synthesis," Indiana University Comupter Science Dept. Technical Report No. 290 (September 1989).

Draft Reports

[DDD] BHASKAR BOSE. DDD – A Transformation System for Digital Design Derivation, User Manual Draft program documentation. The DDD system is available for public distribution. Contact Steven D. Johnson, Computer Science Department, Indiana University, Bloomington, Indiana, USA.

[DAISY] STEVEN D. JOHNSON. The Daisy Programming Manual Draft program documentation. The Daisy/DSI system is available for public distribution. Contact Steven D. Johnson, Computer Science Department, Indiana University, Bloomington, Indiana, USA.

[RCW] KAMLESH RATH, IGNACIO CELIS, AND ROBERT M. WEHRMEISTER. An Experiment in VLSI Implementation of Register Transfer Equations using Register Transfer Bus Architecture, draft project report, May 1990.

[CRWZ] IGNACIO CELIS, KAMLESH RATH, ROBERT M. WEHRMEISTER, AND ZHENG ZHU. Systematic Derivation of LIBRA using DDD, draft project report, December 1989.

[BCHS] BHASKAR BOSE, MANUEL CORDERO, BRIAN HECK, AND WEIWEI SUN. A Derivation of the FM8502: A Verified Microprocessor, draft project report, December 1989.

[TMB] ROBERT TRUEL, JAMES MARSHALL, AND C. DAVID BOYER. Scheme Machine Derivation, draft project report, December 1989.

Appendix A

Other Sponsored Activities

Personnel Summary

Supported Personnel

Steven D. Johnson (Principal Investigator), four summer months.
David E. Winkel (Co-Investigator), two summer months.
Bhaskar Bose (Research Assistant), three full years, August 1987 to May 1989. Nominated to Ph.D. candidacy, August 1990.
C. David Boyer (Research Assistant), two academic years plus on summer, August 1987 to May 1989.
Zheng Zhu (Research Assistant), one full year, August 1989 to July 1990. Nominated to Ph.D. candidacy, July 1990.

Other Personnel

Robert M. Wehrmeister, Member of Technical Staff, Indiana University Computer Science Department.

Research Presentations

1. Zheng Zhu, "An Algebraic Framework for Data Abstraction in Hardware Description," the Oxford Workshop on *Designing Correct Circuits,* Oxford, September 1990.
2. Steven D. Johnson, "On the Interplay of Synthesis and Verification, Experiments with the FM8501 Microprocessor Description," the IMEC-IFIP International Workshop on *Applied Formal Methods for Correct VLSI Design*, Leuven, Belgium, November 1989.
3. Zheng Zhu, "An Algebraic Characterization of Structural Synthesis for Hardware," Poster presentation at the IMEC-IFIP International Workshop on *Applied Formal Methods for Correct VLSI Design*, Leuven, Belgium, November 1989.
4. Steven D. Johnson, "A system for Digital Design Derivation," at the IEEE High-level Synthesis Workshop, Kennebunkport, Maine, September 1989.
5. Steven D. Johnson, "Manipulating Logical Organization with System Factorizations," at the Cornell Mathematical Sciences Institute Workshop on *Hardware Specification, Verification and Synthesis: Mathematical Aspects,* Ithaca, New York, July 1989.
6. Steven D. Johnson, "Digital Design Derivation," at a working meeting of the IFIP WG2.8, Mystic Connecticut, May, 1989.
7. C. David Boyer "Using the Digital Design Derivation System: Case Study of a VLSI Garbage Collector Implementation," CHDL 1989, Washington, D.C.
8. C. David Boyer "Using the Digital Design Derivation System: Case Study of a VLSI Garbage Collector Implementation," departmental colloquium, Indiana University Computer Science Department, June 1989.

9. C. David Boyer and Bhaskar Bose, "Digital Design Derivation—An Engineering Perspective," departmental colloquium, Indiana University Computer Science Department, January 27, 1989.
10. Steven D. Johnson, "Modeling Transistors Applicatively," at the IFIP WG10/2 working conference on *The Fusion of Hardware Design and Verification,* Glasgow, Scotland, July 1988.
11. Steven D. Johnson, "The DSI Model of Concurrent Graph Processing," The workshop on the *Implementation of Lazy Functional Languages,* Aspenäs, Sweden, September 1988, sponsered by The University of Göteborg and Chalmers University of Technology.
12. Steven D. Johnson, "Modeling Transistors Applicatively," departmental colloquium, Indiana University Computer Science Department, June 1988.

Supported External Speakers

Contribution of grant MIP87-07067 is shown in parentheses.

1. Mary Sheeran, "The Ruby Algebra," Indiana University Computer Science Department Colloquium, May 11, 1988 ($300).
2. Ganesh Gopalakrishnan, "HOP—a Model for Concurrent Synchronous Processes," Indiana University Computer Science Department Colloquium, April 5, 1990. Gopalakrishnan visited the department for four days ($150).

Related Seminars

1. The *Indiana Hardware Methods Research Group* is a continuing seminar of the Indiana University Computer Science Department.' It meets weekly. Participants include all members of this research project, and others from hardware and architecture research in this department.
2. "Hardware Derivation" (1989), conducted by Steven D. Johnson. Nine participants. Several projects in digital design derivation were done in this seminar.
3. "Applicative System Description" (1987), conducted by Steven D. Johnson. Aspects of an algebraic formalism of digital design synthesis.

Supported Travel

The grant provided partial support for travel by research assistants, with the department and other sources also contributing a portion.

1. 1990 IEEE/ACM Design Automation Conference, Orlando, Florida, June 24 to June 26. *Johnson.*
2. 1989 IFIP/IMEC International Workshop on Applied Formal Methods for Correct VLSI Design, Brussels, Belgium, November 11 to November 18. *Johnson, Zhu.*
3. 1989 IEEE International Workshop on High-Level Synthesis, Kennebunkport, Maine October 14 to October 18. *Johnson, Bose.*
4. 1989 Cornell Mathematical Sciences Institute workshop on Hardware Specification, Verification, and Synthesis: Mathematical Aspects, Ithaca, New York, July 4 to July 7. *Johnson.*

5. 1989 IFIP WG10.2 Ninth International Symposium on Computer Hardware Description Languages and their Applications, Washington, D.C., June 19 to June 21. *Johnson, Bose, Zhu, and Boyer.*
6. 1989 IFIP WG2.8 (Functional and Applicative Programming) Meeting in Mystic, Connecticut, May 1 to May 7. Johnson attended as an observer (not funded by this grant).
7. 1988 Workshop on the Implementation of Lazy Functional Languages, Aspenäs, Sweden (sponsered by Chalmers University and the University of Göteborg), September 1988. *Johnson* (not funded by this grant).
8. 1988 IFIP WG10.2 Working Conference on the Fusion of Hardware Design and Verification, Glasgow, Scotland, July 2 to July 9. *Johnson.*
9. 1988 Banff Hardware Verification Workshop, Calgary, Alberta, Canada, June 12 to June 18. *Johnson.*
10. 1988 IEEE VLSI Workshop, Clearwater Beach, Florida, February 28 to March 2. *Johnson.*

Appendix B

Design Derivation Projects, 1986–1989

[*Excerpt of a University report*]

These photographs and drawings show three hardware prototype projects undertaken as part of digital synthesis research [under MIP87-07067]. The designs were done in order to explore and demonstrate the methods developed in my research. While they are not the primary results of the research, they are essential evidence that the methodology and supporting tools work. In that sense, these systems are what the research is about. Two of the projects are fully functional systems. The third, which involved fabrication of the first VLSI chip for this research group (and the second for the department) is mostly operational.

Each of the projects involved the same collection of tools, including three separate systems developed at Indiana.. Circuit descriptions were synthesized using DDD, a design system developed by my research group. DDD is written in the Chez Scheme programming language, which was created by colleague Kent Dybvig. Parts of the descriptions are compiled into physical circuitry by CAD tools obtained through the CER project. Other parts are realized using off-the-shelf integrated circuit components. The prototypes are built on the Logic Engine development system, a device created for this purpose by Franklin Prosser and David Winkel of Indiana. The Logic Engine provides secure power and clocking, a large prototyping area, and a panel of display lights and switches. It has other, more advanced, capabilities as well, including on-board computers for 'microprogram' development. However, these facilities were not used in any of these projects.

A synopsis of these projects can be found in the *Annual Progress Report for NSF Grant MIP 8707067: 1987–1988*,

1. An SECD Computer

The SECD prototype is a complete general-purpose computer, derived from description of a Lisp language processor first published by Peter Landin in 1964. The "SECD machine" is something of a benchmark in certain branches of programming language research. It has also been used as a case study in hardware verification research. The prototyping project was headed by Robert Wehrmeister, a member of the department's technical staff.

Photograph 1. This photograph shows the completed project. The prototype can be seen on the vertical panel, which is the project area of the Logic Engine development system. The box labeled "Logic Engine" houses the power supply, display panel, a microprocessor, and two disk drives. Also shown are the terminal display and keyboard for the SECD machine.

Photograph 2. This is the project area. The central region of the circuit, consisting of seven chips mounted in plastic housings and four that are not, comprise the SECD processor. The bank of labeled chips across the top implement a *heap management unit,* which maintains the memory of the computer. The bank of eight chips between are the computer's memory. The subsystem at the lower left is an interface to the computer terminal. The processor and heap manager were derived using the DDD system. The memory and interface were built manually.

Photograph 3. This is the rear view of the SECD prototype. The discrete components plugged into the project panel are connected by wires (about 3,000) which are manually wrapped about terminal posts. All of the prototypes are similarly "wire wrapped."

Photographs 4 and 5. The SECD prototype is built from two kinds of integrated circuits. The components held in plastic housings, and those covered by white labels, are programmable logic devices (PLDs). These are circuits that can be electrically reconfigured by software. The DDD system synthesized the 'programs' (actually boolean equations) used to configure the PLDs. The remaining components on the panel are standard medium-scale integrated circuits (MSI). Photograph 4 is the same as Photograph 2. Photograph 5 shows one of the PLD devices uncovered, an Altera EP1800. The grey area within the hole is the PLD chip. It implements several hundred boolean equations.

2. A PLD Garbage Collector

[*This project was completed prior to MIP87-07067 funding, but is discussed in several subsequent publications. See also the PLA version, shown next—SDJ.*]

Symbolic processing computers operate against a memory that is functionally different from that of a conventional computer. Such memories require their own "front end" processor to manage a complex space of symbolic data. Among other tasks, this processor recycles data that has been discarded by the computer; hence it is called a "garbage collector." This prototype, which was our first large-scale design, implements a garbage collector in PLD technology.

The device was tested against a production Scheme language implementation. It ran at about sixty times the rate of the software benchmark. The project leader was David Boyer, one of my doctoral students.

Photograph 6. This photograph shows the prototype assembled on the design panel of a Logic Engine. The terminal shown in the picture represents the workstation that hosted the software benchmark. The collector was tested by halting the work station program whenever it was about to reclaim data, copying an image of the memory over to the prototype, reclaiming the data, copying the resulting image back to the work station, and resuming execution there.

Photograph 7. Here, the project panel is shown removed from the Logic Engine base unit. The dual bank of PLD devices across the top of the panel and the bank of MSI components across the bottom comprise the collector prototype. The array of chips center-left are the memory on which the collector operates. The remaining components on the panel implement the interface for transporting memory images to and from the work station.

3. A VLSI Garbage Collector

Again using the DDD system, Boyer retargeted his garbage collector design to VLSI. In order to do it, he had to make substantial changes to the architecture of the design. However, he was able to derive these changes in architecture starting from the same specification used for the previous project. The resulting three-chip design was functionally correct, as determined by extensive simulation, but the chips suffer a minor electrical problem.

PHOTOGRAPH 8. This picture shows the VLSI version of the garbage collector, mounted at the top of the design panel of a Logic Engine. None of the other circuitry shown is involved in the prototype; however, a conventional memory for the system remains to be built.

PHOTOGRAPHS 9 AND 10. Photograph 9 is a close-up of the VLSI prototype. These three chips account for approximately thirty-four of devices on the PLD version. Photograph 10 shows the chip packages removed from their sockets. Each of the chips has eighty four pins.

DRAWINGS 11 AND 12. These two drawings are computer generated plots of the VLSI structures on the surface of the chip. Drawing 11 shows the entire layout and Drawing 12 shows the detail of one of the sixteen rectangular "bit slices." These components are Programmed Logic Arrays (PLAs); each is roughly equivalent to one of the PLDs on the earlier design. The derived circuit is on the interior; the elements around the border of the drawing are pin pads, to which the external pins are connected.

PHOTOGRAPH 13. This figure is a photomicrograph of the actual circuit, which is roughly 40 mm^2 in total area. This is a moderately large chip area.