Abstract

This paper describes the implementation of a knowledge-based system that provides VLSI design advice in the areas of performance, manufacturability, testability, and overall design quality, from a netlist description of an application specific integrated circuit (ASIC) design. The NCR DesignAdvisor™ addresses key issues that may plague the designer during actual design and schematic capture of an ASIC. In turn, it reduces the necessity for tedious work with conventional tools and potentially eliminates costly redesign. In its first application, it is used to automate much of the semicustom design review process, which was previously a painstaking manual effort.

Introduction

NCR Microelectronics provides the necessary software, service, and manufacturing capability to support IC designers in the creation of Application Specific Integrated Circuits (ASICs). These ASICs are often designed by engineers with varying levels of experience, from novice to expert, and with varying backgrounds, from IC expert to system designer.

The knowledge required to design and implement a successful ASIC design is distributed among diverse experts who perform specific tasks related to ASIC design and manufacture. This includes expertise from design engineers, product and test engineers, and quality engineers, all with specific knowledge of various aspects of IC design and manufacture. However, this expertise is often unavailable to the designer during the specification and design of a component, and only becomes available at the time of a design review or during prototype testing. Changes to a design at this stage of the process are very costly in terms of time and money; therefore, it is desirable to correct problems as early as possible in the design cycle.

The DesignAdvisor provides a mechanism for capturing this distributed knowledge, representing years of experience, and allows the designer to apply it to a design. The system provides suggestions regarding specific aspects of IC design quality while the design is still in progress and the cost of implementing changes is relatively low. The system analyzes partial or complete designs, pointing out critical problems that may cause the device to fail or to be difficult to manufacture and test, and provides helpful guidelines and information to improve performance and density.

This is a departure from most current research in applying AI technology to problems in the VLSI design domain, which has typically focused on...
some form of design synthesis or decomposition [Mostow85]. Integrated CAD systems that automate design synthesis from a high level design description into a pre-fabrication database will be the elite CAD tools of the future. However, until the difficult problems facing such a system, i.e. low level analysis of performance, redundancy minimization, and intercommunication among synthesis tools [Gu & Smith86], are solved, much can be gained through intermediate knowledge-based applications like the DesignAdvisor [Steele87].

The DesignAdvisor is in commercial use and has provided valid advice that designers have chosen to follow for each design on which it was run. Payoffs for the designer include increased productivity (a correct design much sooner), lower design costs, first pass success (advice uncovers fatal errors), and higher quality, more consistent designs.

Description of the DesignAdvisor System

As an ASIC vendor, we are exposed to a wide variety of designs with widely varying style, complexity and function. With this variance comes a need for increased sophistication in analysis. The DesignAdvisor analyzes performance, testability, and manufacturability of CMOS semi-custom VLSI designs implemented using a library of digital and analog cells. Input to the system consists of a netlist (text) description of a circuit logic schematic. Once the circuit information is represented within the knowledge base, additional information is obtained when needed through interactive dialogue. Hierarchical design methodology, a "must" for managing complexity in VLSI design, is preserved by running the system on netlists for design modules as well as for complete designs. The DesignAdvisor system runs on the design engineer's workstation (currently on the Mentor Graphics/Apollo DN3000/DN4000 series and the Sun 3) and is integrated into the designer's accustomed environment where it highlights problem areas within the schematic capture system.

An initial hierarchy of design attributes, which serves as a framework for the system's recommendations, was compiled by studying the major problems discovered in commercial designs over the past several years [Figure 1]. An analysis of problem data was conducted to pinpoint typical mistakes that had caused unsatisfactory silicon prototypes. It was found that the attributes of the DesignAd-
visor [Figure 1] address a high percentage of these problems. Sixty-six percent of the design problems in first pass prototypes would have been detected by a system possessing the attributes of the DesignAdvisor. Seventy percent of the timing related problems alone could have been avoided. This sometimes subtle area accounted for the majority of the observed problems. The skeleton hierarchy has expanded throughout the development of the system to encompass attributes in greater detail.

The system was implemented using a prototype hybrid expert system development tool, called Proteus [Petrie,Russinoff & Steiner86]. The tool integrates both known and novel AI programming techniques into a robust vehicle for the implementation of knowledge based systems. Proteus integrates a logic-based rule mechanism (Horn clauses), including both forward and backward chaining rules, with a frame-based object representation and a fully integrated Truth Maintenance System (TMS) based on the work of Jon Doyle [Doyle79]. LISP s-expressions may be used in both the antecedents (pre-conditions) and the consequents (resulting actions) of rules.

A TMS Based Default Reasoning Approach To Design

Design systems typically reason with large amounts of incomplete knowledge. This form of knowledge is inherently intractable and cannot be applied in reasonable resource bounds [Levesque86]. By representing and reasoning about partial knowledge in a principled way, it can be made tractable. Forcing a more complete representation of the knowledge domain can provide a context for guiding search and limiting combinatorial explosion. Incomplete knowledge can be made more manageable by using default reasoning to fill in missing detail, and by using closed-world assumptions to define clear boundaries for the domain.

Consider a scenario where one is seeking technical advice from a knowledgeable, but possibly imperfect, design consultant. The exchange would probably involve an explanation of the existing design and maybe some additional “question-and-answer” dialogue. Having obtained the facts and recognized how the design related to past experience, the consultant may then offer some advice on how to improve the design. In general, this is an effective way for obtaining feedback on the evaluation and refinement of a design approach. The exchange can allow for communication of concerns and viewpoints to the consultant for use in focusing the analysis. The focus can then be targeted for the designer’s accustomed fashion of design, rather than forcing overly restrictive design practices. The educational and productive exchange may also force consideration of critical trade-offs in the design and point out possible deficiencies in the design methodology. In addition, the advice may be overridden with knowledge contrary to that of the consultant.

This is the model of interaction with the NCR DesignAdvisor and illustrates the use of default reasoning for an advisory approach to design [Steele88]. Normal and plausible design practices can be addressed first, using simplifying closed-world assumptions for domain guidance, while more complex situations can be handled upon contradiction of default advice, using domain guided dependency-directed backtracking (DDB). This approach has shown the effectiveness and limitations of using a justification based TMS for problems with large search spaces where several applicable solutions must be found. In contrast to an ATMS1, the control provided by DDB can be used to focus the search on subsets of the solution space in order to avoid needless inefficiency. By using this form of control, the system can be directed toward the efficient detection of constraint violations and the subsequent identification of faulty assumptions. The strategy has proven to be effective for reasoning about the design process in general, because it corresponds very closely to the designer’s own thought process.

A Novel Application Of Contradiction Resolution

In many cases, the system advises on problem areas using incomplete knowledge, making reasonable default assumptions based on typical logic design techniques. Initially, the system considers only a small number of exceptions to the default assumptions in each advice area. Less common exceptions, that apply to specialized cases, are considered only when a designer disagrees with the advice given. If the designer contradicts the default advice, the system can present known, less common, alternatives to the designer to determine if

1Assumption-based TMS approaches have advantages over the Doyle-style TMS for certain types of problems, but ATMS control mechanisms are weak [deKleer86].
a justification exists for retracting the advice. The integrated TMS provides the mechanism for identifying and retracting faulty assumptions [Petrie85] in order to resolve the contradiction of a default design rule violation. The contradiction resolution itself is performed using the DDB control mechanism. Frequent criticisms, involving lack of control of this technique, have been overcome by allowing the contradiction resolution to be guided by domain knowledge [Petrie87].

The system is prepared to provide exceptions to each advice category, although these exceptions may or may not cover the user's particular case. If an appropriate exception cannot be determined by the system, the user may still oppose the advice, but the user is prompted to explain the exception in order to provide a record of the decision made in the session transcript. This helps the designer to communicate the deviation from typical practice to himself and to his colleagues, and provides an opportunity for the knowledge engineer to continually improve the knowledge base through utilization of the recorded sessions and consultation with expert designers. This method allows for rapid inclusion of additional exceptions to each area of advice and therefore minimizes the knowledge acquisition bottleneck.

Knowledge Base Examples

Expertise Modules

The knowledge base is broken down into modules that correspond to specific areas of expertise. This knowledge was elicited from interviews with senior designers from among our more experienced customers, and interviews and case studies with NCR design, product, and test engineers. These experts brought their experience to bear on five broad categories: timing and clocking, speed performance, system interface (I/O) specifications, general knowledge about good design practice, and testability. Output of each module is in the form of advice, concerning a number of aspects of the design. The designer then has the option to seek further explanation about a piece of advice, to reason with the system regarding possible exceptions to the advice, or to simply accept the suggestion and make the recommended changes.

We believe that much of the value of the system's advice lies in its availability to the designer during actual design and schematic capture of the ASIC, independent of simulation. This may reduce the necessity for tedious work with conventional tools, and potentially eliminate costly redesign, thus substantially cutting the time to market. The following examples are drawn from recent DesignAdvisor sessions on designs in progress, and serve to illustrate key capabilities of the system.

Timing/Clocking Expertise Module

The Timing/Clocking Expertise Module prompts the designer for the names of the system clocks and their periods. Derived clock signals and clock periods are then determined and automatically propagated throughout the design. The system then checks for problems such as clock skew, glitch-prone and risky gated clocks, insufficiently buffered clocks, and excessive clock speed. Risky asynchronous practices are advised against due to a preference for synchronous design. In addition, information is provided about critical paths and routing efficiency factors.

Phase-skew-use-PCL2

ERROR — The clock phases N-3055 and CLXX3 (to the DFFR.I-802) are skewed. No more than an inverter difference is allowed. A PCL2 should be used in this situation.

In this instance, the system warns the designer that the possibility of clock skew exists. This was determined by looking at the relationship of the signals driving the clock inputs of the clocked logic cell, and noting that more than one inverting buffer lies in the path between the clock and clock-bar signals. This heuristic allows the system to quickly identify potential clock skew situations.

Performance Expertise Module

The Performance Expertise Module gives advice on topics such as optimal buffer configurations (serial vs. parallel), placement or removal of buffers to speed up critical paths, loading of buffers that can improve routing inefficiency, and optimization of buffer placement when complemented signals are available. High-drive cells (logic cells identical to standard logic gates but capable of driving larger capacitive load) are recommended when appropriate. These analyses eliminate tedious manual calculations, and reduce the need for guesswork or re-
peated simulations in order to determine the optimum buffering of signals in the design.

**Replace-buf-cell**

NOTE — The buffered cell NOR2H I-12 provides no speed-up for the signal N-32.
— The buffered cell should be replaced with the non-buffered equivalent NOR2.

This example indicates a situation where the designer placed a high-drive cell (NOR2H — a 2 input inverting logical OR) on a path where it was not actually needed. Circuit paths without a large amount of capacitance are actually slower if driven by a buffered cell. The system can determine the best way to correct the situation, recommending the use of a smaller-drive cell in this case. Similarly, the system can recommend use of buffered-cells or buffers on paths that drive large loads, always with the goal of optimizing the area/speed tradeoff. This knowledge module uses a complex tree-pruning algorithm as it seeks to meet highly interactive constraints.

**System Interface Expertise Module**

The System Interface Expertise Module verifies proper input/output pad and buffer combinations to meet system requirements. When evaluating system requirements, several attributes of each signal, such as noisy inputs and tristated outputs, are considered. Also, the power needs of the entire design are considered in order to verify, and if necessary improve, the designer's choice of the number and placement of power and ground pins. This is an area where many designers make common mistakes that may go unnoticed until prototypes fail to function within the intended system.

**General Expertise Module**

The General Expertise Module identifies poor design practices such as feedback loops (using combinational logic cells) and delay logic. These practices are especially dangerous in standard cell design — due to the use of automated routing, there is more uncertainty in predicting delays than in full custom layout. Other errors checked by this module include floating tristate buses, unconnected input signals, and incorrect usage of pullup and pulldown elements and other special cells.

In this example [Figure 2], the DesignAdvisor looks at the connections of each cell, checking for common design errors related to the intended use of that cell. This example illustrates a rule that may not be obvious to a new designer, and may even be overlooked by an expert, but is absolutely critical to the success of a CMOS design. Tristate buses allow two or more devices to drive the same signal at different times, dependent on the state of the control signals to those devices. Thus, not driving a bus signal, or allowing it to float to an undetermined value will cause any fault on that bus to be undetectable.

In this situation, the designer needs to add pullup or pulldown devices to each floating bus. These sorts of problems may be considered minor in terms of how little trouble they are to correct, but are increasingly more annoying to find as the complexity of designs increases. In the past, these errors may have been caught by peer reviews of a design, or at a structured design review involving the Design Engineer and NCR engineers respon-
sible for manufacturing and testing the completed IC. The DesignAdvisor system is intended to serve as an automated design review, thus reducing the burden of managing the complexity of large designs.

Testability Expertise Module

The Testability Expertise Module currently contains knowledge pertaining to the controllability of clocked logic elements. Designs are more easily testable when each clocked logic element can be conveniently set or reset. For example, given the names of all external reset signals, the system traces internal resets and then looks for deeply imbedded resets, embedded non-resettable clock elements, and invalid resets that are not related to an external reset signal.

Indirect Reset

WARNING — Up to ten clock cycles are required to set or reset the clocked logic cell I-468. A direct set or reset enhances testability.

In this case [Figure 3], the worst case path to set or reset this particular element I-468 may require ten clock cycles, meaning that the clocked logic cell is potentially difficult to control. An overall system reset should be able to put the entire design in a known state; this reduces test time for an individual circuit, and increases the overall testability of the circuit, thus reducing the overall cost of the part to the customer.

As an example of the default reasoning capability of the system, consider the situation where the designer wishes to contradict this advice. While the above advice, based upon heuristic knowledge about controllability in typical IC designs, is valid in a local sense, the designer may have global knowledge of the design that renders the advice invalid. In this case, the system will attempt to resolve the contradiction by seeking some new information about this exception, responding with a query:

Direct control of reset signals is preferred. An excessive number of states may be required to control the gated reset signal — Can the signal be controlled functionally with only a few input patterns?

Suppose that the designer knows that the element can easily be set or reset by directly loading in a value during test mode, and responds positively to the query. The system would then change the state of the database to indicate that it no longer believes that this particular piece of advice is true, justified by the user's explicit knowledge about the situation.

If this is not the case, and the designer accepts the DesignAdvisor's recommendation, then the circuit should be redesigned to more easily reset this clocked element.

The advice provided by the DesignAdvisor's five expertise modules is currently of commercial quality and has been used by a number of customers. Additionally, the DesignAdvisor system is a flexible foundation for building enhancements into the overall knowledge structure. Enhancements can be made as new information is received from users (both internal to NCR and our external customers) and from systematic knowledge acquisition in new areas of expertise.

Development History

A large scale prototype was designed at NCR, Fort Collins, Colorado by the Advanced Development group. The Software Development group, in a joint effort with Advanced Development, subsequently generated a product from the prototype. During product generation, top down software development methodologies were employed [Mintz89].

The initial prototype system was developed on a Symbolics LISP machine and required 3.5 man-years of effort. The product development effort which followed the advanced prototype system required an additional 5 man-years of effort, and was accomplished in one year by utilizing the original Advanced Development engineers plus 3 Software Development engineers. The Software Development staff served internships in Advanced Development in order to be trained effectively and to create a cooperative team for the intense product development effort which followed.

The system has been used internally on real designs since October 1987. The system was run for customers by NCR staff for a period of nine months prior to an external product offering in July 1988. All new internal designs done within the Microelectronics Division now undergo a DesignAdvisor analysis.
Figure 3: The DesignAdvisor system is coupled with the schematic capture environment so that problem areas can be pointed out visually.
Commercial Success and Payoff: Designer Feedback

The DesignAdvisor has been used on many designs and in each case the designer received valuable advice that was taken into account prior to device prototyping. In 40% of commercial designs analyzed internally, first pass failure was avoided by following the advice given by the DesignAdvisor. This represents a significant savings in both time (about 12 weeks) and cost (on the order of 10's of thousands).

Other measures of success for the customer include improved quality of design, more consistent design practices, tutorial advice, and improved documentation of the critical, esoteric portions of the design. Other measures of success for our organization include a position of industry leadership, in providing state-of-the-art technological solutions to meet customer needs, and in providing a more cost effective environment for designing ASICs. This tool is showing financial impact well beyond the revenue generated through software sales and maintenance by enhancing NCR’s leadership position in the ASIC industry.

Acknowledgements

The authors would like to thank the MCC Expert Systems Group, headed by Charles Petrie, and Dr. Dan Ellsworth (NCR) for their consultation and support. Special thanks also to Eric Mintz (NCR), Chris Whitley (NCR), and Dr. Andrew Ressler (Listz Programming Inc.).

References
