PEARL: AN EXPERT SYSTEM FOR POWER SUPPLY LAYOUT

Edward J. DeJesus, James P. Callan and Curtis R. Whitehead

DIGITAL EQUIPMENT CORPORATION
CAD Systems Engineering
Andover, Massachusetts 01810-1098

ABSTRACT

The use of artificial intelligence (AI) expert systems technology has demonstrated its advantages with many new tools in the computer aided design (CAD) field. This paper describes how domain specific knowledge was integrated with a conventional CAD architecture to develop an expert system. The combination resulted in a tool that provides intelligent assistance to printed wiring board (PWB) layout designers.

This CAD tool focuses entirely on the layout requirements of power supply circuits for PWBs. The system is called PEARL for Power-supply Expert Assisted Rule-based Layout. PEARL acts as an advisor to the layout designer, providing expert assistance with the placement of components on PWBs. It is presently being enhanced to provide etch routing assistance as well.

INTRODUCTION

A survey of the design automation literature [1] reveals that for several decades numerous CAD tools have been successfully developed with conventional programming methods and algorithmic solutions. The last few years have also seen the knowledge-based approach successfully applied to a wide variety of CAD problems (e.g. [2,3,4,5,6,7]), including several in the printed wiring board (PWB) domain (e.g. [8,9,10]). However, both traditional and knowledge-based tools developed in the PWB domain have primarily focused on only the problems of laying out PWBs for digital circuits. Until now little has been done to resolve the PWB layout problems for analog circuits.

Power supply layout is a subset of the larger analog PWB design problem. These layouts are very heavily constrained, and consequently the requirements differ greatly from those for which conventional programming techniques and algorithms alone are sufficient.

Digital Equipment Corporation (DEC), has developed an expert system that addresses just this problem area. The prototype system [11] has been refined and is currently used as a production layout tool in many of DEC's engineering sites. This paper describes the expert tool for power supply layout called PEARL, the factors that influenced its design, its problem solving approach, its flexible functionality, some recent field results, and the plans for its future.

The Problem Domain

Power supply layout is very different from the layout of digital circuits. DEC has traditionally developed CAD tools specifically for digital circuits comprised of dual inline packages (DIPS) and gate arrays. These components have consistent rectangular shapes, and are typically placed in rows and columns. Their interconnections are usually routed with consistent etch widths and spacings. DEC and the rest of the CAD industry have developed very successful algorithms for these digital circuit layout problems [1,12].

However, power supplies are predominantly comprised of discrete components (e.g. capacitors, transformers, diodes, resistors, transistors) in analog circuits. Discretes tend to be irregularly shaped, making the component placement problem more difficult. Component placements are further constrained by a number of factors, including:

- Certain groups of components, called functional circuit groups, must be placed near one another.
- The primary side of the power supply circuit must be kept away from the secondary side.
- Industry safety standards (e.g. UL, VDE) and manufacturing requirements must be met.

Power supply routing requires varying conductor widths and spacings for the differing voltages and currents. Etch widths and paths must also be chosen to prevent problems with inductance and signal noise from inadequate grounding separation. It is also common to see conductors run both horizontally and vertically on the same layer, something that is avoided on digital logic PWBs. These constraints make power supply routing difficult for our present topological router [12].

Before PEARL, power supply designs at Digital were laid out manually or on CAD systems which had no logic information, automatic tools, or analysis capabilities. Several layout passes were made because the prototype boards that were built and tested failed to meet requirements.

A Knowledge-Based Solution

It is difficult to build and support a layout tool that satisfies power supply layout constraints

0738-100X/86/0000/0615$01.00 ©1986 IEEE

using only traditional algorithmic approaches. However, there are several reasons why a knowledge-based approach might succeed. Knowledge-based systems are often applied to situations where the problem is heavily constrained, and the knowledge about how to solve it is scattered or incomplete. Power supply layout fits this description nicely. It is heavily constrained. The knowledge about what constitutes a good layout is distributed between the engineer and the layout designer. The fact that it has been a multi-pass layout process indicates that there are many unknown factors affecting it.

Another advantage of knowledge-based systems is that since the knowledge-base is expected to grow throughout its duration, most systems provide mechanisms to facilitate and manage that growth. As pointed out in [4], this typically means that knowledge-based systems are easy to modify and can explain their reasoning, making it easier for human experts to find weaknesses and suggest improvements during normal interactions with the system.

THE DEVELOPMENT STRATEGY

In order to reduce the complexity of the power supply layout problem, several major decisions were made about the overall design of PEARL.

First, we decided to treat placement and routing separately by focusing on just the needs of discrete mosaic placements that are highly influenced by constraints. This provided a smooth learning curve by allowing us to develop the rule-based system in phases.

Second, trying to solve the problem with one fully-automatic tool was undesirable since automatic approaches often fail to solve all of the problems or allow the input of experienced layout designers. Therefore PEARL is designed to assist and guide the user. It is intended to primarily operate in a highly interactive semi-automatic mode. However, it can also be used in a completely manual or fully-automatic mode.

Third, it was important to offer a prototype tool quickly in order to determine the feasibility of the approach and to elicit user feedback for further development. The assistant or semi-automatic mode became the basis of the tool, creating an environment for observing the user. It also enabled the developers to acquire knowledge about layout problems to formulate rules and strategies for intelligent solutions to them.

Finally, it was necessary to integrate PEARL within the present CAD system, called the VAX Layout System (VLS). The decision to make PEARL a VLS application was significant, because it is important to produce a tool that operates in an environment that the users are familiar with and is part of the mainstream design process.

Just Another VLS Application

VLS is DEC's proprietary CAD system for PWB layout. It is an environment and a set of application programs that share a common data model [13], user interface, and set of utilities. VLS applications include editors for libraries and logic, tools for placement, routing and analysis, and an interface to manufacturing.

PEARL is designed to be "just another VLS application". This was crucial to user acceptance, because VLS was already installed at many engineering sites whose design processes relied upon it. In addition, VLS offers many other functions that are vital to the layout process. For example, an etch editing application provides manual routing capabilities that are used to complete the analog layouts until PEARL's routing functionality becomes available.

VLS can run on any VAX/VMS system. The typical VLS workstation is either a VAX-11/750 or a MicroVAX II with with 8 megabytes of memory, a high resolution graphics display, a graphics tablet, and a terminal. VLS also supports terminals with low resolution graphics capabilities (e.g. VT241) and no graphics capabilities (e.g. VT220). This gives users the option of doing tasks that are not graphics-intensive at their desks.

ARCHITECTURE

Power supply layouts have many of the classic symptoms for which a knowledge-based approach is appropriate. However, knowledge-based systems generally employ symbolic processing methods while CAD systems require large databases. The symbolic processing of a large database could potentially result in a very slow system. Since PEARL is an interactive tool, reasonably fast user response was essential. Consequently PEARL's architecture has two levels to provide adequate speed and take advantage of a knowledge-based approach.

A Two Level System Approach

PEARL is structured in two separate layers. The top layer is a knowledge-based system and the bottom layer is comprised of efficient algorithms and data structures. The top layer acts as a flexible control mechanism, directing the user interface, explanation facility, placement strategies, and overall program control. The bottom layer consists of subroutines that contain the placement algorithms and supporting data structures in addition to providing access to VLS data structures, utilities and graphics (see figure 1). This two level approach gives PEARL the advantage of combining two problem solving methods into one expert system.

The top layer of PEARL was developed in OPS5, a production system language designed for implementing rule-based systems [14]. The bottom layer was developed in BLISS, a procedural language that is similar to the C programming language. BLISS is extremely fast and efficient for traditional algorithmic programming. OPS5 and BLISS were chosen as the implementation languages because they were general enough for the task, while providing the necessary speed.

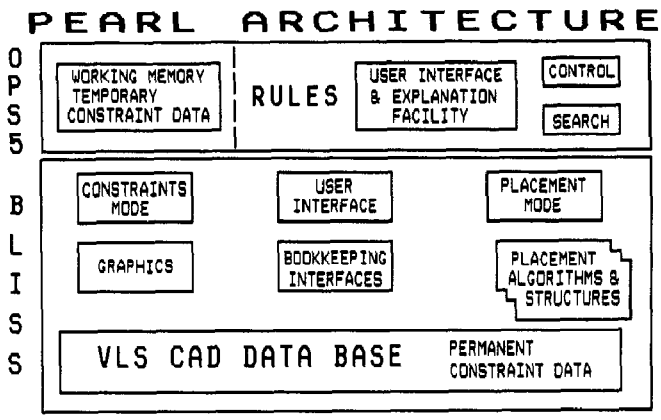

Figure 1. PEARL's two level architecture.

Since VLS is largely written in BLISS, developing large portions of PEARL in BLISS would minimize future support and maintenance issues.

DEC's version of OPS5 is itself written in BLISS. In the CAD domain, OPS5 has been used to develop several successful expert systems (e.g. [2,3,5,6,7,11]). Unlike most programming languages, OPS5 has no underlying notion of sequential execution. Activation of its production rules is dependent upon the contents of OPS5 working memory. Since OPS5 runs much faster when the number of working memory elements is kept to a minimum [15], we limit that area to only what is relevant about a specific goal or problem at hand.

The Rule-Based Level

The OPS5 rules drive the user interface and placement control strategies, enabling the rule-based system to always know what mode and state the user and layout are in. When the user presses a button or enters a command string, an OPS5 working memory element is created and matched against production rules. There are essentially three types of production rules:

- o user interface rules -
  carry out functions chosen by the user, and drive the explanation facility.
- o control rules -
  metarules to establish subgoals for overall program control, and heuristics to handle special layout sequences based upon power supply constraints and layout strategies.
- o component searching rules -
  identify one or more components that satisfy a set of constraints.

PEARL has several different placement strategies to partition the component selection search space. Heuristic rules determine the order in which placement strategies are considered. They insure that the placement algorithms give priority to components within functional circuits and critical groups, and to components that only qualify for placement under very specific conditions.

Within a particular placement strategy, selection rules search for components that satisfy the constraints and placement criteria. The selection rules for a given placement strategy can only be considered when that placement strategy is active.

The advantage of the rule-based system is apparent in the way constraints from the knowledge base are used to control and prune the search for placement candidates. In addition, critical groups or clusters of components are guaranteed to be placed closely to each other. A similar grouping or clustering approach is used in KNOB-PLACE [9], however our methods of capturing the cluster relationships is quite different.

The control rules are maintained by metarules that monitor the status fields of each placement strategy. Control is transferred by the OPS5 inference engine between groups of rules with goals and subgoals that guide the processing through a series of placement strategies. The placement strategies and control rules could easily be altered to accommodate technologies for other PWB layout problems and the BLISS primitives and algorithms would still be effective.

There are presently about 250 rules coded in OPS5. Through refinement many OPS5 rules were condensed down to one or two rules that call external BLISS routines which are more efficient at accomplishing specific tasks. There are well over 500 "if then" situations in the BLISS code that alter various conditions in both the placement algorithms and the geometric primitives.

The Lower Level and Algorithms

When the rule-based system finds several components that it considers equally good placement candidates, it relies on a modified constructive placement algorithm [1] that is implemented in BLISS to choose among them. The component selection criteria is one which chooses components that are most heavily connected to previously placed components, with a negative weight included for interconnections attached to unplaced components. This empirical selection method is similar to many "least commitment" approaches, and is intended to avoid the need for backtracking. PEARL has no backtracking designed into its fully-automatic mode, but a later discussion about its flexible semi-automatic mode does address the problem.

Once a component has been chosen for placement, geometric primitives determine where to position it. The rule-based system may specify that the component must be kept near another component or group of components, but it is up to a placement algorithm to figure out the topology. Typical algorithms, such as nearest neighbor using manhattan and euclidean metrics, are common throughout PEARL and VLS. These algorithms, many of which carry their own constraints in the form of weighting factors, are intended to be generic for solving many classes of PWB layout problems. Component positions are based upon minimization of connection length, and constraints affecting orientation and alignment for polarization and machine insertion.

An important feature of PEARL is its ability to position odd shaped components without overlapping one another. This is a major advantage to designers because it eliminates a tedious and time consuming effort in the layout process. Unlike typical placement algorithms for DIPs, PEARL does not depend on channels or slots to position components in rows and columns. Instead, it positions each component next to another component on a user-defined grid, and adjusts it to prevent overlaps by quickly checking an obstacle list.

To facilitate rapid searching through the obstacles, which consist of points, segments, and rectangles, the obstacles are stored in a data structure which is a variant of a k-d tree [16]. With this structure, additions, deletions, spacing violations, and overlaps can each be determined in O(log(n)) time, where n is the number of obstacles and each obstacle is stored only once.

Focusing The Knowledge

PEARL's knowledge-base contains rules that satisfy many of the constraints that are typically imposed on power supply layouts. Constraints are transformed into physical relationships that address the layout requirements in a form that is suitable for a layout tool and its users. For example, snubber networks contain components which need to be placed near one another to prevent signal radiation problems by avoiding long conductor paths. This constraint requires that the system have knowledge about which components must be placed near one another (called keep near groups by PEARL).

The information that identifies constrained objects is captured interactively in a forms mode, which consists of predefined screens with fill-in-the blank fields (see figure 2). This bridges an important information gap between the power supply engineer and the layout designer by capturing instructions that engineers previously gave to designers verbally or on marked up schematics. In fact, at times the engineers themselves are not aware of all of the critical layout needs and potential problems. This mode insures that both the engineer and the designer are guided into examining and preparing for the typical layout problems of analog designs. These constraints can now be entered and stored with the rest of the design data by either the engineer or the designer. PEARL does not require this data to do a placement, but it will produce much better results if the information is available.

This organization of information could be construed as just another data base, or an extension of an existing data base. In fact, as pointed out in [2], a data base alone will not provide any assistance to human experts with making decisions. However, this critical data combined with rules produces a knowledge-base that focuses attention on the relevant information at each design step and suggests appropriate courses of actions to be taken. This combination is a very powerful aid in decision making.

```
PEARL 4.1(1.53)                CONSTRAINT MODE
Page 2 of 9

                          KEEP NEAR GROUPS

Group 1:   Q1      T3
Group 2:   Q1      R4
Group 3:   T3      C7      R5      R12
Group 4:   D5      C13     R10
Group 5:   D6      C14     R9
Group 6:   D7      C15     R8
Group 7:   J1      VR1
Group 8:   J2      C23     C24
Group 9:   J2      C18     C20     L1
Group 10:  C8      R2
Group 11:  C9      D2
Group 12:  C10     D3
Group 13:  R6      R7      D8      D9      C12
Group 14:

 Enter string name:
```

```
PEARL 4.1(1.53)                CONSTRAINT MODE
Page 3 of 9

                      POWER TRAIN (PRIMARY SIDE)

Input Connectors:  J1

Emi Filter:        C8      C9

Primary Rectifer:  D2

Input Filter:      VR1     R1      D1      C1      C3
                   C4      C2

Inverter:          T2      C5      C6      C7

Remaining Primary
Side Components:   D8      D9      C10     C11     D3
                   D4      R2

Main Transformer:  T3

 Enter string name:
```

Figure 2. Two pages from Constraints mode.

PLACEMENT USER INTERFACE

PEARL's placement tool can be used interchangeably between three modes: manual, semi-automatic, and fully-automatic. All three modes operate with the same user keypad (see figure 3).

The manual mode is provided to give the designer the option of overriding PEARL's suggestions at any time. Keys are available for selecting, moving, rotating and placing components.

The semi-automatic mode is PEARL's default mode and operates as an assistant to the designer. Once the Placement mode is entered, PEARL offers the user up to four candidate components with up

SELECT	BY AREA	UNSELECT ALL	UNSELECT
PLACE	ROTATE	NEXT LOCATION	NEXT CANDI-DATA
MOVE HORIZ ON/OFF	MOVE VERT ON/OFF	BEST CAND & LOCATION	EXPLAIN
START/ STOP	UNPLACE	OPTIONS/ CONS-TRAINTS	HELP
WINDOW		EXIT	

Figure 3. Functions assigned to placement keypad.

to four possible locations per candidate (see figure 4). Designers can examine the candidate components in their various locations using the Next Candidate and Next Location keys. When a candidate is offered in a location, its interconnections are dynamically reconnected to the nearest placed components in the network. This dynamic reconnecting simplifies the picture and illustrates the reason for choosing the different locations.

As each candidate is offered, PEARL displays a reason for its choice. Messages such as "Engineering Constraint: Keep C5 near R4", or "C19 was chosen as part of the secondary side of the power train" are generated by PEARL's explanation facility (see figure 4). Users can accept PEARL's recommendation by marking the candidate as placed via the place key, or create a choice of their own in manual mode.

Keys are provided to place or unplace any candidate (PEARL's or the user's). Once place or unplace steps are taken, the semi-automatic placement immediately continues by offering new candidates and locations.

A Start/Stop key allows the user to change back and forth from the semi-automatic mode to a fully-automatic mode at any time. Fully-automatic mode places the best candidate in its best location, without requesting approval from the user after each choice. As each component is placed, the layout is updated and progress is displayed on the screen until all parts are placed or the user stops it. Stopping it returns PEARL to the semi-automatic mode, where the designer may examine a new list of choices or unplace some of PEARL's previously placed components.

A Flexible Tool

PEARL's interactive nature enables the users to unplace or undo any portion of the layout that may not meet their requirements. This eliminates the need for the fully-automatic tool to have an elaborate backtracking scheme or an iterative algorithm to detect and resolve problems. It also enables designers to include their own creative and experienced contributions in the layout, making them more comfortable with the system.

PEARL provides the flexibility for the user to override its suggestions, yet it still safeguards against violations of constraints. A dynamic checking facility produces warning messages whenever the user moves a constrained component. For example, if component C5 is Selected and Moved, a bell goes off to draw attention to a displayed message of "C5 was placed as part of a keep near group". If the user Selected and Rotated a polarized or machine insertable component, a message like "D5 pin one direction should be up (^)" would appear.

The entire placement process is always restartable. In other words, users can stop, exit PEARL or VLS at any time, and return later to pick up where they left off. This feature is an inherent result of the interactive approach, requiring constant updating of data structures,

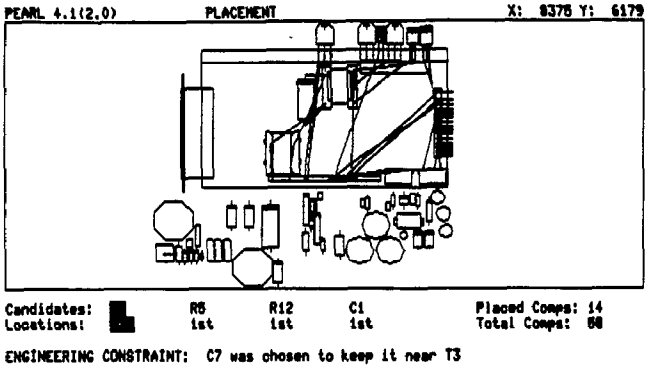

Figure 4. Graphics display in Placement mode showing four candidates, with C7 as the 1st candidate in its 3rd location. Note the explanation of why C7 was chosen.

and the fact that the constraint data is saved with other design data.

Another useful feature is the ability to graphically highlight classes of components based on how they were entered into the system in the constraints mode. A designer can display the components in the primary side of the circuit by typing "primary" and pressing the Select key. This can be used to verify the separation of the power train circuits or to identify groups of components that must be kept together. This feature quickly points out the critical parts of the design and reinforces the confidence in the system with both designers and engineers.

EXPLANATION FACILITY

One aspect of PEARL that is common with expert systems yet unique among CAD tools is that it can explain its reasoning. Explanations indicate whether the user manually placed a component or it was placed by PEARL because of some specific design constraint. The explanation facility provides information in two ways. First, the user can ask for an explanation by pressing the Explain key and typing a component reference designator after the prompt. Second, explanations are generated automatically whenever a component is suggested for placement (note message in figure 4). In either case, an explanation consists of a one line reason why the component was placed or suggested for placement.

The explanation facility increases the user's confidence in PEARL's suggestions, and also serves as a useful debugging tool for developers. It verifies that the knowledge base is accurate and being interpreted correctly. It can also be used to find out what influenced decisions, after the fact. The ability to find out why something was placed has always been desired in our CAD systems, but until now it remained unfulfilled.

FIELD RESULTS AND ENHANCEMENTS

PEARL was released as a placement tool to DEC's design community during the fall of 1985. Performance estimates are difficult to determine

for a tool that offers 3 modes of interaction. Since PEARL is an interactive tool, estimates based upon CPU time are not really relevant. Therefore, the performance estimates given here are based upon real time use, as opposed to CPU time, for a typical VLS workstation.

In its manual mode, PEARL is as fast as most manual placement tools. In its semi-automatic mode, the overhead of suggesting candidates and computing locations is noticeable but not annoying. PEARL can generate a candidate list and display the best candidate in its best location in 1 to 3 seconds of real time, depending on how many previously placed components must be examined. Once the candidates and locations are generated, switching around between them is very fast.

In its fully-automatic mode, PEARL can place an average sized power supply of about 200 components in under 30 minutes. The overhead involved in updating the graphics display slows PEARL down somewhat, but unlike other automatic tools, the user can intervene at any time. An option also exists to turn off the graphics, resulting in a 10 to 15% speed improvement. This same sized 200 component design, using the semi-automatic interactive mode and including the time to enter constraints, can be completed in about two days.

These performance estimates may appear to be slow when compared with other popular placement algorithms that can generate an entire placement in just a few seconds. However, PEARL is designed for power supply placement, a process that was previously done manually and took from several days to several weeks. The significant difference is that the designers feel confident that manufacturing requirements such as machine insertion and polarity constraints have all been satisfied. More importantly, the engineer's most critical concerns have all been adequately addressed in the first pass of the layout.

In our experience users seem quite pleased with the real time performance and the overall quality of the placements. They especially like its flexible human interface, and have requested that many other VLS applications be reimplemented in a similar fashion. As a result PEARL has been enhanced to handle placement of digital circuits as well as power supply layouts.

PEARL was designed to be general enough for these kinds of enhancements. Recent improvements include features to ignore specific components and networks, and allow ranges of keep near groups to represent memory clusters in the constraints mode. Many of the rules and placement primitives are generic, and can effectively address the layout needs and strategies for digital circuits. PEARL was intended to be a specialized tool for power supply placement, however it is now being used to place other types of PWBs as well.

FUTURE DEVELOPMENT

We expect that the placement knowledge base will continue to grow significantly during its next few years in the field. While some of this growth is expected to accommodate the placement needs of other types of PWBs, power supplies will continue to be the primary focus of PEARL. We hope that it ultimately leads to a better understanding of the factors affecting power supply placement, so that fewer prototypes are required.

As mentioned earlier, we initially chose to develop PEARL in phases. The placement phase has been the focus of this paper, but work has recently begun on the routing phase. We envision a routing tool that is similar to the placement tool in its style of user interaction, with manual, semi-automatic and fully-automatic modes.

We are presently developing a manual interactive router with dynamic checking features available as options. We expect it to warn of potential signal management and spacing violations, especially in the critical power train sections of the circuit. This will require knowledge of the varying etch conductor paths and spacings necessary to accommodate high and low voltage networks.

The manual routing mode will later be complemented with the development of a semi-automatic routing mode similar to PEARL's semi-automatic placement mode. We plan for it to suggest which connections to route and offer choices of potential conductor paths for each of them.

Ultimately we intend to provide a fully-automatic routing mode that, like the fully-automatic placement mode, can have its progress viewed on the graphics screen or be run in a batch mode for improved run times. It will be restartable so that designers can interrupt the router, manually alter portions of the design and then continue in any mode. This capability will require knowledge of partially completed routes and rerouting. We hope to address the reroute problem in a fashion similar to the methods used in BLOREC [10]. paragraph Most of the routing issues discussed and addressed in WEAVER [4], are also needs and goals for PEARL. For instance, we believe that the PEARL router must also consider several metrics simultaneously, such as minimum wire length, routing area, and vias. In addition there is an important need to attempt the completion of many networks on a single layer. Networks routed on a single layer are a common occurrence on most power supply layouts that today are predominantly routed manually by experts. In addition to the typical geometry and routing problems, analog designs are dominated by varying conductor widths and spacings, as well as constraints for safety requirements. Consequently, we hope to address these multiple routing problems with a two level approach similar to PEARL's placement architecture. We plan to develop variations of popular routing algorithms (e.g. line search, maze [1]), combined with and controlled by a knowledge-based system.

CONCLUDING REMARKS

The original goal was to provide a CAD tool for power supply layout within the existing CAD system, VLS. PEARL satisfies that goal by providing an excellent power supply placement tool now and the promise of a routing tool with a similar approach in the future.

PEARL utilizes both conventional programming and knowledge-based problem solving methods to focus attention on critical aspects of each design step and suggest appropriate courses of action. The result is an intelligent layout assistant that offers expert advice, explains its reasoning, allows users to explore alternatives, warns of potential violations, and remembers forgotten details. This approach is not necessarily a typical expert system, but it is a knowledge-based CAD system with a flexible interactive approach that offers expert placement choices and produces intelligent results.

PEARL has also proved to be a very valuable learning experience for Digital's CAD Systems Engineering department. It has demonstrated that a knowledge-based system can be successfully applied to a very difficult layout problem. It has also demonstrated that a knowledge-based system can be built within the VLS environment. These are particularly significant results because they open up another programming methodology and approach for CAD software developers.

ACKNOWLEDGMENTS

We want to extend our thanks to several people who helped us with PEARL. Jon Leitch of Medicine Hat College spent his sabbatical developing the placement primitives and manual override functions. Dave Concordia and Dave Wall built the explanation facility while undergraduates at Worcester Polytechnic Institute. Design engineers Anil Ohri and Jim Gregorich, and layout designer Jim McDonough, were the experts in the power supply domain. Finally, Mike Doreau, Woody Greene, Vinay Nulkar and Tom Smith provided valuable comments and suggestions on this paper.

REFERENCES

[1] M. A. Breuer, A. D. Friedman, and A. Iosupovicz, "A Survey of the State of the Art of Design Automation," IEEE Trans. Comput., pp 58-75, Oct, 1981.

[2] J. McDermott, "Domain Knowledge and the Design Process," Proc of the 18th Design Automation Conf., pp 580-588, June, 1981.

[3] W.P. Birmingham, and D.P. Siewiorek, "MICON: A Knowledge Based Single Board Computer Designer," Proc. of 21st Design Automation Conf., June, 1984.

[4] L. I. Steinberg, and T. M. Mitchell, "A Knowledge-Based Approach to VLSI CAD: The Redesign System," Proc of the 21st Design Automation Conf., pp 252-258, June, 1984.

[5] J.H. Kim, J. McDermott, and D.P. Siewiorek, "Exploiting Domain Knowledge in IC Layout," IEEE Design & TEST, pp 52-64, August, 1984.

[6] R. Joobbani, and D. P. Siewiorek, "WEAVER: A Knowledge-Based Routing Expert," Proc of the 22nd Design Automation Conf., pp 266-272, June, 1985.

[7] T. J. Kowalski, and D. E. Thomas, "The VLSI Design Automation Assistant: What's In A Knowledge Base," Proc of the 22nd Design Automation Conf., pp 252-258, June, 1985.

[8] H. Mori, K. Mitsumoto, T. Fujita, and S. Goto, "Knowledge-Based VLSI Routing System - WIREX -," Proc. of the Int'l Conf. on Fifth Generation Computer Systems, 1984.

[9] G. Odawara, K. Iijima, and K. Wakabayashi, "Knowledge-Based Placement Technique for Printed Wiring Boards," Proc of the 22nd Design Automation Conf., pp 616-622, June, 1985.

[10] R. L. Joseph, "An Expert Systems Approach to Completing Partially Routed Printed Circuit Boards," Proc. of the 22nd Design Automation Conf., pp 523-528, June, 1985.

[11] E. J. DeJesus, and J. P. Callan, "PEARL: A Knowledge-Based Expert Assisted CAD Tool," Proc 2nd Conference on Artificial Intelligence Applications, pp 258-263, December, 1985.

[12] R. A. Armstrong, and M. T. Doreau, "A Topological Representation for Single Layer Routing Problems with Variable Sized Objects," Proc. of Int'l Symposium on Circuits and Systems, 1982.

[13] T.R. Smith, "A Data Architecture for an Uncertain Design and Manufacturing Environment," Proc. of 22nd Design Automation Conference, 1985.

[14] C. Forgy, "The OPS5 User's Manual," Technical Report CMU-CS-81-135, Computer Science Dept., Carnegie-Mellon Univ., 1981.

[15] L. Brownston, R. Farrell, E. Kant, and N. Martin, "Programming Expert Systems in OPS5: An Introduction to Rule-Based Programming," Addison-Wesley Publishing Co., Reading, Mass, 1985.

[16] J. L. Bentley, and J. H. Friedman, "Data Structures for Range Searching" Computing Surveys, Vol 11, No 4, pp 397-409, Dec. 1979.