TM-0083

VLSI配線エキスパートシステム：WIREX

森　啓，光本圭子，後藤　敏
（日本電気）
岩田裕之
（日本電気情報サービス）

November, 1984

ＶＬＳＩ配線エキスパートシステム　：ＷＩＲＥＸ

Wiring Design Expert System for VLSI : WIREX

森　啓+，若田　裕之++，光本　圭子+，後藤　敏+

Hajimu Mori, Hiroyuki Wakata, Keiko Mitsumoto and Satoshi Goto

+日本電気（株）Ｃ＆Ｃシステム研究所，++日本電気情報サービス（株）

C & C Systems Research Laboratories NEC Corporation   NEC Information Service, LTD.

Abstract

This paper describes a new interactive routing system for VLSI layout design based on artificial intelligence techniques. In the VLSI routing problem, most of the time is spent on the interactive design operations to get the complete connection after automatic routing. An expert designer has to delete, modify or draw routing patterns on the display terminals by exploiting his own specific knowledge and experience. This interactive process is done in a trial and error manner, and usually takes a lot of design time.

The system proposed here, adopts a knowledge-based method and relieves a designer from the burden of complicated design operations. This system accepts the designer's knowledge as a set of rules represented in Prolog language. The Prolog interpreter interprets the rules and makes inference on the knowledge. This knowledge-based routing system has been developed to solve a large scale real problem. It has been applied to design custom VLSIs with several thousand gates and has shown quite promising result.

1. Introduction

Recent advances in microelectronics technology have required high density layout design techniques for VLSI with complicated design rules. This makes it necessary to establish a highly effective layout design system with automatic and interactive function.

A routing problem in VLSI layout design is to determine a connecting path meeting all physical and electrical constraints for each given net. From a computational point of view, a routing problem is considered to be a hard combinatorial problem. It is far too difficult to complete the whole design by only using automatic programs with deterministic algorithms. The total design process consists of automatic and manual design. In most practical cases, a large portion of the design time is spent on manual design, such as drawing, error checking and correction after automatic design.

In these years, highly interactive CAD systems have been developed to cope with this problem [1,2,3] and the layout design time has been reduced in a great deal. However, its successful operation is fully dependent on the designer's ability. The designer carries out the design by exploiting his own specific knowledge on the layout. The expert designer must examine the wiring patterns carefully on the graphic display to modify or create wiring patterns for complete net connection. This is time consuming and error prone.

In the design process, an essential difference between human beings and computer is an intuitive ability for reaching an appropriate goal. The ability is considered to be based on a mechanism for applying a "rule of thumb". Recently, several new CAD systems, based on artificial intelligence techniques, have been reported [4,5,6,7]. However, it seems that there remain some difficulties to solve the large scale real problems. The system proposed here is focused on applying the designer's specific knowledge into the design process for practical routing design problems. The designer's knowledge specifies a certain operation in accordance with a specific situation, which is represented in a form of the design objects, their properties, and logical relations among them. In this system, the knowledge is expressed in a set of rules and written in Prolog language [8] and stored in the knowledge database. The Prolog interpreter interprets the rules and makes inference on the knowledge to perform layout design task.

In order to make a computer an expert designer's assistant, this paper presents a new interactive routing system which treats with the designer's specific knowledge for a practical design problem.

2. AI Approach to VLSI Routing

A VLSI routing problem is to find a connecting path meeting all phisical and electrical constrains for each given signal net. The connecting path is made by metal wires of two different layers and via holes for interlayer connections. A signal net is a connection requirement to be electrically equivalent for a set of pins. To simplify layout design, grid lines are introduced in the chip vertically and horizontally. Wiring patterns connecting pins can be routed only on these grid lines. Fig. 1(a) shows connection requirement for the nets A, B, C, and one of the solutions is shown in Fig. 1(b).

From the computational complexity point of view, the routing problem is considered to be a hard combinatorial problem, in the sense that the computation time required to obtain the real optimum solution increases in exponential order, when the problem size, i.e., number of signal nets, increases. Hence, the routing algorithms have been based on heuristic rationales. Although those heuristic algorithms have been improved year by year, they have not succeeded yet to give a complete design in practical application.

The final routing goal is to achieve complete net connectivity in as short a design time as possible. To cope with this problem, interactive editing systems have been practically used in the wiring layout design [1,2,3]. When 100% routing is not achieved by automatic programs, the incomplete routing result is transmitted to the graphic display station for further completion. The designer has to create or modify wiring patterns on CRT in order to achieve 100% routing. This is time consuming and error prone.

In the above interactive design process, the designer effectively exploits his knowledge aquired through his experience or intuition and solves the problem. Therefore, if a part of the interactive design operations is replaced by automatic computer programs, the design time is expected to be extremely reduced.

Until now, artificial intelligent techniques have been actively developed and there arise wishes among CAD engineers to use them for their own purpose. In this routing problem, the following key problems have to be made clear.

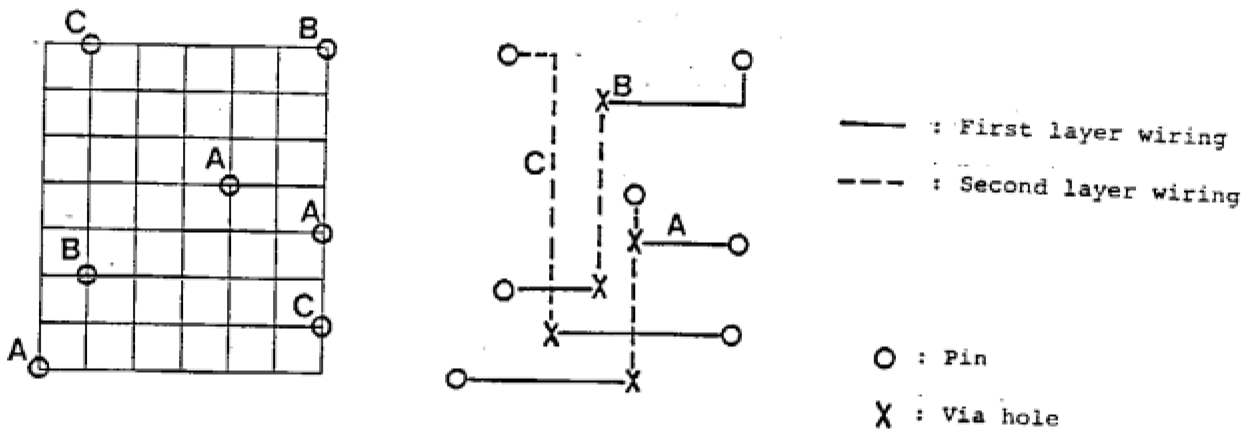
(1) Knowledge acquisition : What operations will the designer do to solve a problem in a specific situation ?

(2) Knowledge representation : How to represent the designer's operation in knowledge-based form by programming language ?

(3) Knowledge utilization : How to utilize or exploit the knowledge database to solve a specific problem ?

We have analyzed the designer's operations for many cases and formalized them as a set of rules. This work is fully done by manuals right now, and we want to make it in an automatic way and establish an automated knowledge acquisition system in a year or so.

As a language to represent and utilize the knowledge, we adopt Prolog language. However, Prolog language



(a) Connection requirement (b) Wiring layout

Fig.1 Routing problem

requires too extravagant memory and runs too slow to be used practically in VLSI routing problem. Therefore, we have developed a new Prolog interpreter which has a function to link with FORTRAN programs.

3. System Configuration

The Figure 2 shows a system configuration. The system includes two kinds of databases, knowledge database and CAD database. The knowledge database stores a certain amount of design knowledge about specific routing situations : properties of design objects, relations among the objects, and rules for guiding problem solving. The CAD database, which is a conventional database, contains physical information about the design objectives.

At present, the inference system is Prolog interpreter. Prolog language has inference mechanism in itself. And the designer's knowledge is written directly in Prolog language. In this language, rules can be considered as a set of knowledge and they are represented as a sequence of predicates. In order to have a linkage to the existing CAD system, our Prolog interpreter allows predicates that correspond to procedures in the CAD system. The procedure is itself written in FORTRAN statements and manipulate physical data on the CAD database.

Rules in Prolog programs have a hierarchical structure, and rules at the top level of the hierarchy can be considered as commands to the system. As the designer inputs a rule name, the interpreter interprets the program and produce a certain procedure sequence.

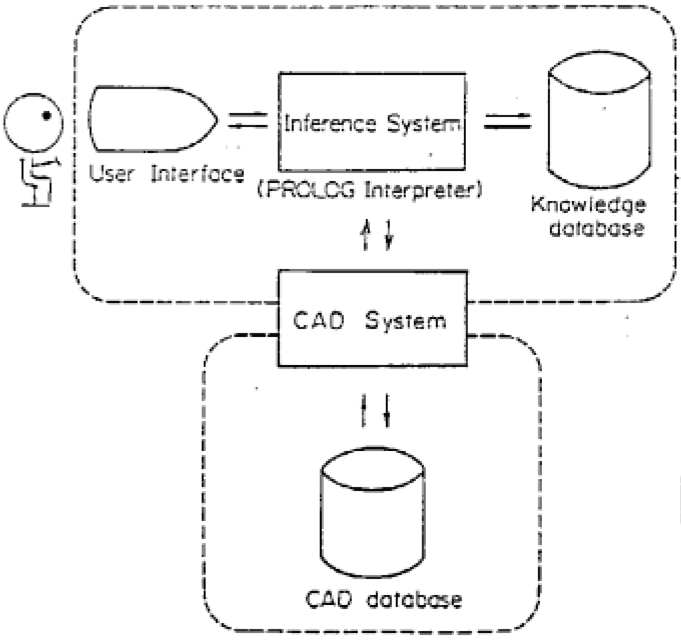


Fig.2 System configuration

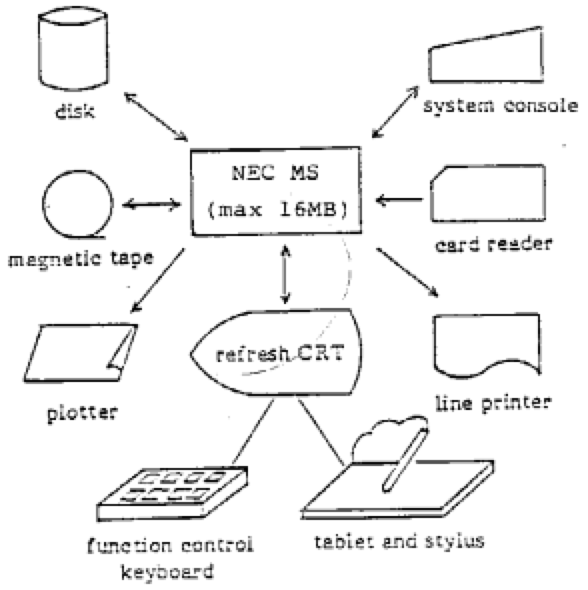


Fig.3 Hardware configuration

This sequence represents the designer's operation on the display, and its execution can simulate the way of design method. These mechanisms enable the designer to solve a practical routing problem within a reasonable computer time and memory.

The system hardware configuration is shown in Fig. 3 and its picture in Fig. 4. The system operates on NEC/MS minicomputer, which has up to 16M byte memory capacity and 200M byte disk storage capacity, coupled with 21-inch beam-directed refresh graphic display. The display terminal includes a graphics tablet and a function control keyboard with 2048 by 2048 point address space.

Fig.4 WIREX system

4. Prolog Interpreter

Prolog is one of languages suitable for representing knowledge and developing rule-based systems. Prolog is characterized by the following three points : the first point is to have a basic inference mechanism in itself. The second point is to be able to manipulate structured data objects, such as lists and trees, by applying pattern matching. The third point is to access program and data in the same way and allow them to be mixed together.

In general, the mechanism of a rule-based system is considered to be a simple repetition of selecting and executing rules. The rule-based system must operate pattern matching process whenever such a repetition would occur. Therefore, the more rules the rule-based system has in the knowledge database, the slower it becomes in inference. In Prolog programming, data are expressed by clauses with empty body (unit clauses). The rule-based system has to store programs and many rules in the knowledge database to solve problems with a large amount of data. This makes the inference speed slower.

To cope with this problem, procedural knowledge should be written in more adequate language such as FORTRAN [7]. A lot of knowledge in the design process is procedural knowledge, such as, how to manipulate physical data and which design primitives should be executed. We have developed Prolog interpreter which has the additional function to link with FORTRAN language. Design data in the CAD database of an existing CAD system can be accessed by FORTRAN routines. This is an efficient method to take advantage of accumulated programs in an existing CAD system. Predicates corresponding to FORTRAN routines can be defined in the Prolog programs. Those routines are procedures in the CAD system, and they can access data in the CAD database. These predicates are called external functional predicates. This additional function enables the system to realize high speed processing with less computer memory.

An external functional predicate corresponds to a FORTRAN subroutine, which recovers data in the CAD database if the unification is failed in the following predicate. This recover function is introduced to keep the design result consistent with the CAD database, when the backtrack occurs in the Prolog clause.

5. Knowledge-Based Routing

The designer's operations are done on the basis of his knowledge acquired through his experience. In order to

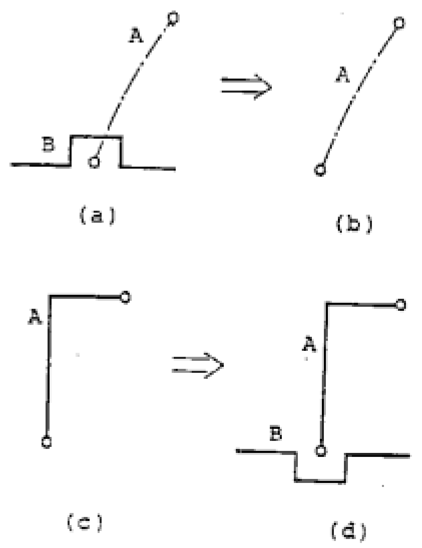


Fig.5 Designer's operation

represent the designer's knowledge as a set of rules, we must analyze the designer's operation. We will show it by a simple example.

Figure 5 shows one of the interactive operations. Net A cannot be connected, since it is blocked by a wiring pattern of net B. The designer will easily solve this problem, i.e., connect both net A and net B on the CRT display by the following operations.

(STEP1) Find a net B, which blocks the connection of net A.

(STEP2) Delete the wiring pattern of net B.

(STEP3) Connect net A.

(STEP4) Connect net B.

These operations, shown in Fig. 5, are considered to be the designer's knowledge to solve this specific problem. This knowledge is described in Prolog statements, called "RULE30", as follows.

```
RULE30(*NET_A) :-
    BLOCKING(*NET_A,*NET_B),
    $DELETE(*NET_B),
    $CONNECT(*NET_A),
    $CONNECT(*NET_B).
```

Variables are represented by letters headed by '*', and the external functional predicates are headed by '$'. The predicate "BLOCKING" is also represented in details as follows.

```
BLOCKING(*NET_A,*NET_B) :-
   GETPIN(*NET_A,*LAY,*XS,*YS),
   EQ(*LAY,1),
   DIRECTION(*YS,*DIR),
   $GRIDSEARCH(*DIR,*LAY,*XS,*YS,*NET_B).
```

The GETPIN predicate takes out a coordinate (*XS,*YS) and a layer number *LAY of *NET_A. The EQ predicate checks whether the layer number is 1 or not. *DIR is a direction, which have four values (up, down, left and right), and the value is set after executing DIRECTION predicate by unification and $GRIDSEARCH predicate denotes a procedure which finds a blocking net *NET_B by searching *LAY layer area toward *DIR direction around (*XS,*YS) coordinate. If one of the blocking nets is found, the next predicate $DELETE is tried. Otherwise, another blocking net is tried to find by executing BLOCKING rule clause.

As the designer gives the following goal caluse to the system,

:- RULE30(NET_A). ,

each predicate of the rule clause RULE30 is executed step by step sequentially, and is backtracked if it fails. It does exactly simulate the designer's operations to achieve complete net connectivity, shown in Fig. 5.

Several examples are tried to evaluate the rules and inference mechanism. Figure 6 shows a routing result on the display, where two nets are left unconnected, as shown by chain lines. Figure 7 shows the result, where one net is succeeded in connecting by executing a goal clause to Fig. 6 routing situation. For 2100 gates gate-array LSI with about 1400 connections, this system could make the complete net connection automatically within half an hour by applying several rules, whose paths could not be found by automatic routers. For the same problem, it took more than one hour for an expert designer to complete the design on the conventional interactive design system.

By accumulating more rules based on the designer's knowledge, more complex and difficult routing problem will be solved in a short time.

We have prepared about 200 rules to the problem. These rules are mainly classified into the following categories.

(1) Blocking area : A signal net is blocked near the pin or far from the pin in the routing area.

(2) Location of signal net pins : A signal net pin whose net is blocked by other nets is located on the edge of a chip or inside a chip.

(3) Number of blocking nets : A signal net is blocked by only one net or more than that.

(4) Layer of blocking nets : A signal net is blocked by the first layer wiring pattern or the second layer wiring pattern.

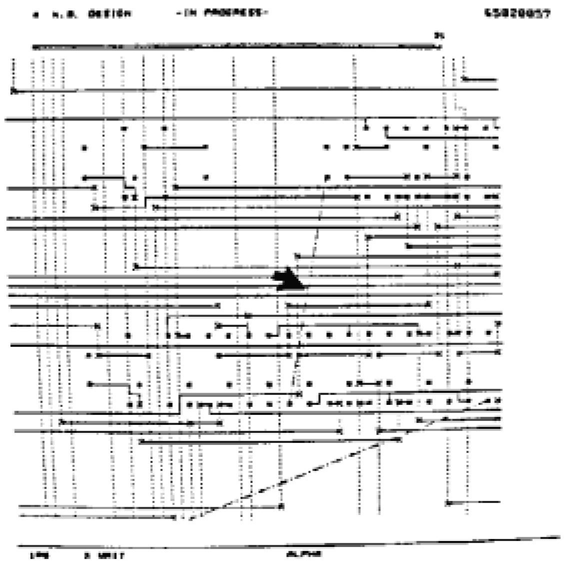
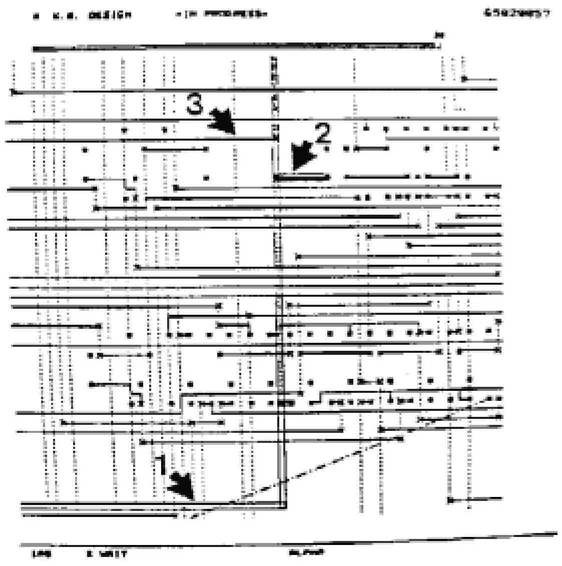


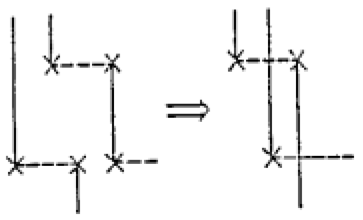
Fig.6 Unconnected net

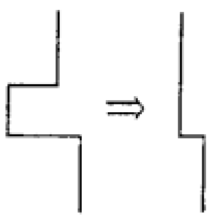


1. blocking wiring pattern
2. connecting path of an unconnected path
3. connecting path of the deleted net

Fig.7 Execution result

(5) Improvement of the pattern shape : Wiring patterns are modified to have better shape by eliminating useless patterns. See Fig. 8.


(c) altering wire routes


(a) removing detour pattern

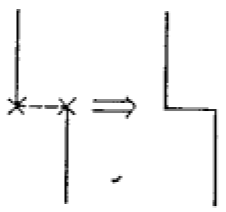

(b) eliminating via holes

Fig.8 Modifying wiring segments

Each rule is constructed by the combination of the above cases and stored in the knowledge database. And also, the designer can add new rules to the knowledge database and use them at the design stage, when he finds a good strategy to solve the situation.

6. Conclusion

This paper presents a new interactive routing system based on artificial intelligence technique. The expert designer can store his own specific knowledge in the system and use them in the design process. The proposed knowledge-based system achieved acceptable response time for the interactive operations and a reasonable result. By accumulating more rules, the system will become more intelligent and be able to solve more difficult problems.

There are several problems which we are now attacking to have more intelligent system. They are

(1) automatic or semi-automatic knowledge acquisition system.

(2) meta-rule system.

(3) meta-inference mechanism.

An essential difference between human beings and a computer program is considered to be an intuitive ability for reaching an appropriate goal. In order to achieve fast and high quality design, a CAD system should have a mechanism with the designer's knowledge.

Aknowledgement

The authors would like to thank Dr. K. Furukawa of the Institute for New Generation Computer Technology for his valuable comments and suggestions. They are also greatly indebted to Drs. Y. Kato, T.Mikami and M.Naniwada for their helpful encouragement.

Reference

[1] F.D.Kinner,"Interactive wiring system", Proc. of 17th Design Automation Conference, pp.296-308, 1980.

[2] H.Mori, T.Fujita, M.Annaka, S.Goto and T.Ohtsuki, "Advanced Interactive Layout Design System for Printed Wiring Boards", in "Hardware and Software Concepts in VLSI", ed. by G. Rabbat, pp.495-523, Van Nostrand Reinhold Company Inc., 1983.

[3] S.Goto, T.Matsuda, K.Takamisawa, T.Fujita, H.Mizumura, H.Nakamura and K.Kitajima, "LAMBDA : an integrated master- slice LSI CAD system", INTEGRATION, vol.1, no.1, pp.53-69, North Holland Pub. Co., 1983.

[4] G.L.Steel, Jr. and G.J.Sussman, "Constraints", Artificial Intelligence, vol. 14, pp.1-39, 1980.

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

[6] H.Brown and M.Stefik, "Palladio : An Expert Assistant for Integrated Circuit Design", Heuristic Programming Project Report HPP-82-5, Stanford, 1982.

[7] T.Fujita and S. Goto,"A Rule-based Routing System", Proc. of IEEE International Conference on Computer Design, pp.451-454, 1983.

[8] W.F.Clocksin and C.S.Mellish, "Programming in Prolog", Spring-verlag, 1981.