VLSI DESIGN
1999, Vol. 9, No. 2, pp. 105–117
Reprints available directly from the publisher
Photocopying permitted by license only

Optimal Differential Routing based on Finite State Machine Theory

M. S. KRISHNAMOORTHY[a], JAMES R. LOY[b,*] and JOHN F. McDONALD[c]

[a] *Department of Computer Science, Rensselaer Polytechnic Institute, Troy, N.Y. 12181;*
[b] *Department of EE&CS, West Point, N.Y. 10996;*
[c] *Center for Integrated Electronics, Rensselaer Polytechnic Institute, Troy, N.Y. 12181*

(Received 11 November 1995; In final form 31 March 1997)

Noise margins in high speed digital systems continue to erode. Full differential signal routing provides a mechanism for deferring these effects. This paper proposes a three stage routing process for solving the adjacent placement routing problem of differential signal pairs, and proves that it is optimal. The process views differential pairs as logical nets; routes the logical nets; then bifurcates the result to achieve a physical realization. Finite state machine theory provides the critical theoretical underpinning and formal proof of correctness necessary for linear time bifurcation. Regular expressions map the theoretical solution to an appropriate implementation strategy that employs feature vectors for net recognition.

Keywords: Routing, differential signal, placement problem, feature vector

I. INTRODUCTION

A. Motivation

Current trends in technology indicate that adjacent placement routing of differential signal pairs is gradually gaining acceptance as a mechanism for coping with reduced noise margins. By routing a signal and its complement in adjacent tracks throughout their run from source to terminus, the ability to reject common mode noise is enhanced. Digital Equipment's new macropipelined VAX and IBM's 9000 and AS400 now employ differential signals on critical nets and clock distribution signals [1, 2, 3]. However, differential routing is not without costs. Chip area increases, and interconnect doubles [4]. Until recently, these costs were considered prohibitive.

Our work produces an optimal solution to adjacent placement routing, while only adding a linear time factor to that required to route a single ended design of equivalent complexity. Finite state machine theory (FSM) provides the theoretical basis and formal verification for the bifurcation

* Address for correspondence: Manager of Technology Operations, Tiger Management, L.L.C., 101 Park Ave (47th Floor), New York, N.Y. 10178. Tel.: (212) 984-2453; e-mail: james_loy@tigerfund.com

process. This is crucial because not all nets are bifurcatable, and they must be identified. Using regular expressions as a mapping function, the actual implementation strategy employs feature vectors to accomplish net recognition and identification. A glossary of terms appears at the end of the paper.

B. Results Obtained

Traditional routing topics have been investigated extensively in the technical literature, to include recent articles on mixed signal routing topics [5]. However, there is an absence of information concerning differential routing of integrated circuits. The prevailing approach independently wires each member of a pair in succession with a conventional router. Often, long runs of closely tracking wire pairs can be obtained in this manner. However, inevitably, one member of a pair will encounter an obstacle, forcing the pair to deviate from close traveling.

Our approach insures complete adjacent tracking, with the following significant results:

(1) Provides the optimal solution to the differential routing problem.
(2) Employs FSM theory to generate net recognizers.
(3) Identifies a basis set of bifurcatable net categories.
(4) Uses regular expressions to map the theoretical identification technique into a readily implementable strategy using feature vectors.

The details of these findings will be examined beginning with an analysis of the router flow diagram and a proof of optimality. The focus then shifts to the bifurcation problem. The generalized bifurcation algorithm suffers from exponential running time. Finite state machine analysis revealed a fundamental basis set of bifurcatable net configurations. The actual implementation strategy relies on feature vectors, generated through regular expression mappings, to recognize the bifurcatable nets, and identify those that are not splittable for re-route.

II. DIFFERENTIAL ROUTING PROCESS

A. Overview

Our three stage process is depicted in Figure 1. It begins by preprocessing net lists so that differential pairs are collapsed to logical nets, routing these single logical entities (SLE), and then bifurcating the resulting wires. This guarantees placement of differential signals pairs in adjacent tracks, while at

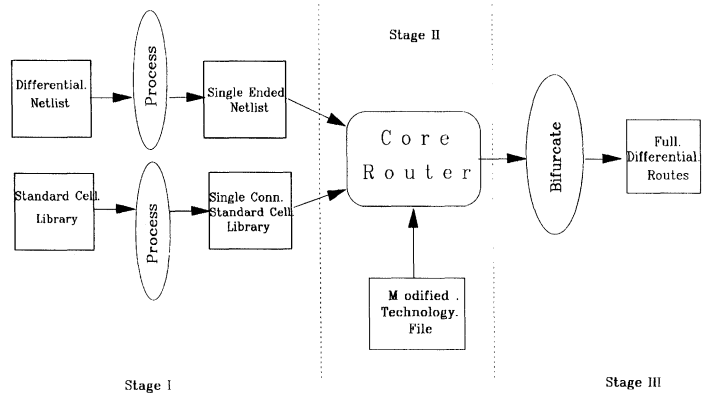


FIGURE 1 Differential router system diagram.

the same time making use of the best available routing tools.

B. Algorithm Analysis by Stage

(1) Stage I generates an image of the standard cell library where each differential port pair on a standard cell is merged into a single logical port as shown in Figure 2. Concurrently, the differential net list is collapsed, so that all differential pair nets are transformed into SLE nets. As the merging progresses, inversions are identified and extracted, since inversion information is not preserved in SLE form. These are all linear time translations.

(2) Stage II uses the best available commercial routing tool. Functioning in a modified environment that consists of – (a) a cell library Image, (b) an SLE net list, and (c) a modified design rule file, the core router routes the layout using large geometry or "fat" wires. The time complexity of Stage II is equivalent to that required to route a single ended design of similar logic complexity.

(3) Stage III is a post processing phase. The "fat" wires are bifurcated, the standard cell images are replaced by the real standard cells, and inversion information is re-introduced. The result is a chip with adjacent track routing of all differential nets.

Using a generalized bifurcation algorithm, the worst case time complexity is exponential as a function of decision points, or "fat" vias that must be resolved. Thus, to split all nets, an upper bound on the computational complexity is given by Eq. (1),

$$\text{Bifurcation\_Time\_Complexity} = \sum_{i=1}^{n} 2^{v_i} \qquad (1)$$

where n represents the total number of nets, and v_i, the number of vias in net i. This upper bound arises, because to properly bifurcate a net you may have to exhaustively explore all configurations before finding a solution, if one exists. We propose a more effective bifurcation alternative based on FSM theory. This idea will lead to a linear time algorithm as we will demonstrate in Section IV.

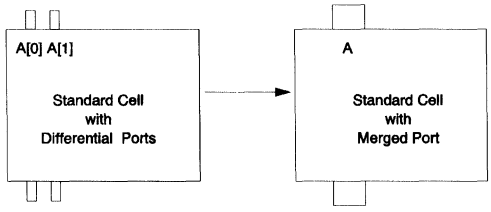


FIGURE 2 Port merge process.

III. OPTIMALITY OF ROUTING PROCESS

If we assume that a linear time solution is achievable for the third stage (bifurcation operation), then the proposed solution is optimal in both computational complexity and physical layout space.

A. Optimality of Solution

Metrics such as chip area, vias, total interconnect and computational complexity define the optimality of the solution. The fundamental routing is accomplished by the core router. Since the pre and post processing stages do not affect the relative ratios of these metrics, the optimality of the fully differential solution will be a direct function of the core router. The only constraint imposed on the router is that it conform to the Manhattan constraint with respect to interconnect layer directions.

With respect to increases in chip area, each differential pair is always routed within the bounds of the fat wire, and generates a minimal block out region at any crossover. This also takes into account the introduction of inversions where necessary, since they will be applied at the "fat" vias. Any approach that routes the differential wires independently will generate crossover block out regions at least as large.

In the area of computational complexity, the fat-wire approach reduces the problem space from size N to size $N/2$, where N represents the total number of nets (counting each wire of a differential pair as an individual net). This is the direct result of the single logical entity (SLE) net concept. Let N represent the total number of thin wires. Now, the run time of a traditional router can be expressed by Eq. (2), where N represents the number of nets and M the available routing tracks [6].

$$T(R)_{\text{Traditional}} \cong M^N \tag{2}$$

If the number of fat wires is $N/2$, then the routing time for our proposed solution is given by Eq. (3).

$$\begin{aligned} T(R)_{\text{Fat\_Wire}} &\cong M^{(N/2)} + T(\text{Bifurcation}) \\ &\cong M^{(N/2)} + C_1 N \\ &\cong \sqrt{T(R)_{\text{Traditional}}} + C_1 N \end{aligned} \tag{3}$$

For actual routing problems where the number of nets is in the thousands, Eq. (4) holds true.

$$T(R)_{\text{Fat\_Wire}} \ll T(R)_{\text{Traditional}} \tag{4}$$

Therefore, if the assumption that bifurcation can be accomplished in linear time is true, then this process will always be equal to or better than a pure differential router, and the solution produced is indeed optimal from both a space and computational complexity standpoint.

IV. FSM BASED BIFURCATION

A. Background

Regardless of bifurcation algorithm selected, certain net topologies are unsplittable in the allocated fat wire tracks, without a local reroute. Examples of two such topologies are provided in Figure 3. By imposing the Manhattan constraint on wiring layers, the class of nets shown in Figure 3(B) can be split, Figure 3(C). This is possible because the constraint guarantees a via at each direction change. Each fat via introduces a degree of freedom for polarity propagation, Figure 4.

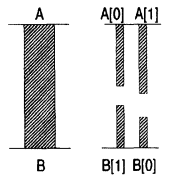


FIGURE 3(A) Unsplittable single metal connection.

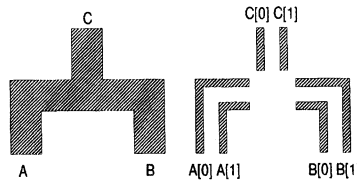


FIGURE 3(B) Single metal unsplittable "Wye" connection.

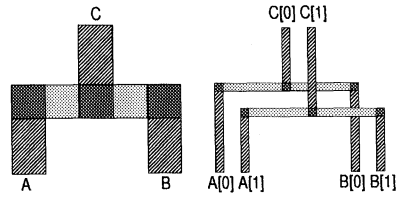


FIGURE 3(C) Splittable "Wye" in two layer metal.

The ability to perform the gross classification of nets into the two categories of bifurcatable and not bifurcatable is crucial to linear time bifurcation. A generalized bifurcation algorithm can be

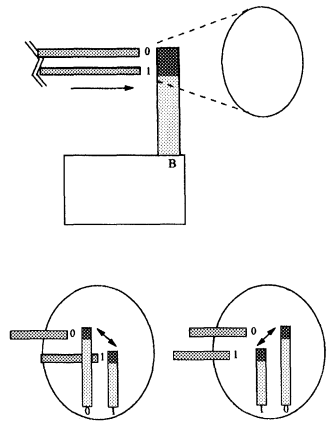


FIGURE 4 Large *via* split represents degree of freedom.

formulated using exhaustive search. It explores all possible via settings and terminates with one of two outcomes: (1) successful bifurcation, or (2) realization that the net is not bifurcatable. Either of these cases may entail exponential running time as a function of vias in the net. Additionally, this technique ignores the inversion problem that must also be handled during bifurcation.

An example of this shortcoming follows. Given the fat net in Figure 5, the algorithm will explore all setting options in an effort to resolve polarities. Ultimately, it will fail to find a solution. Two results are apparent from this exercise. First, even with the Manhattan constraint on routing layers, some nets will not be bifurcatable. Second, without the benefit of "knowing" where to begin in the bifurcation process, the algorithm may have to explore all configurations before determining that a solution does not exist. In the worst case, this is exponential as a function of the vias in the net.

The configuration in Figure 5 that causes the problem is an over-constrained via. Each via has a single degree of freedom and can assure proper connection to a single port pair regardless of pin ordering. When two port segments converge at a via, sufficient freedom does not exist to insure proper connections. A net of this type would have to be designated for local reroute to introduce a jog to offset the segment junctions.

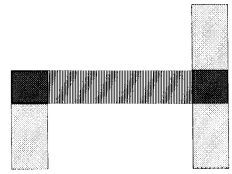


FIGURE 5(A) Over-constrained via.

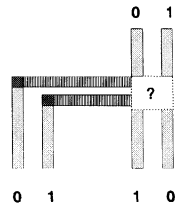


FIGURE 5(B) Unresolvable polarity situation.

Where the generalized algorithm fails, our solution successfully categorizes the net as to its type, and if bifurcatable, successfully splits it, reintroducing inversions where appropriate. Nets that are not bifurcatable are identified as such in linear time and marked for local reroute.

B. Basis Set of Net Categories

An analysis of net structure reveals a major partitioning of bifurcatable and non-bifurcatable, Figure 6(A). Further study revealed the bifurcatable taxonomy of Figure 6(B), which consists of five categories and a group known as multiple backbone [7]. The bifurcatable taxonomy represents a basis set of net categories from which all other nets can be constructed. Multi-backbone nets are merely single backbone nets joined by a vertical segment. The ends of the joining segment terminate in a via on each individual backbone. Passing a "cut line" through the adjoining segment produces a single backbone net and either another

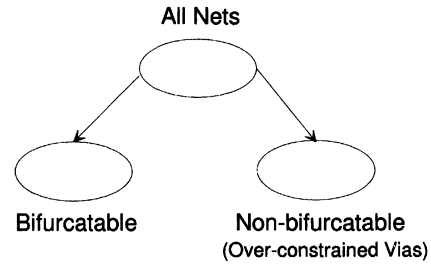


FIGURE 6(A) Gross net taxonomy.

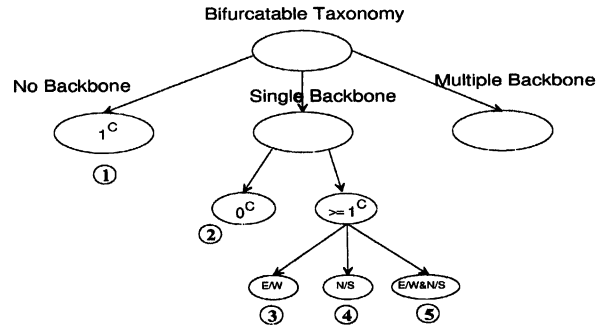


FIGURE 6(B) Bifurcatable net taxonomy.

multiple backbone net, or a final single backbone one. Partitioning continues recursively until the result of the partitioning no longer produces a multi-backbone net as one of the products.

C. Finite State Machine Recognizers

As noted earlier, the critical factor in accomplishing linear time bifurcation is the gross categorization of nets. The most effective way of accomplishing the task in the net list domain is by attempting to "recognize" a net as one of the bifurcatable types. If it is identified, then it is bifurcatable, otherwise it falls into the nonbifurcatable class.

SLE nets can be decomposed into tokens, of which there is a finite set shown in Figure 7. Using the taxonomy as a guide, finite state machines can be constructed to identify or recognize sequences of input tokens. An example of this process is provided in Figure 8. Using the sample SLE net in Figure 8(A), input tokens are generated and depicted graphically in Figure 8(B). Figure 8(C) demonstrates the functioning of the recognizer given the input string.

By constructing FSMs to recognize the bifurcatable categories shown in the taxonomy, arbitrary nets can be "recognized" in linear time. Additionally, failure to be recognized amounts to categorization as non-bifurcatable. This also occurs in linear time regardless of net complexity.

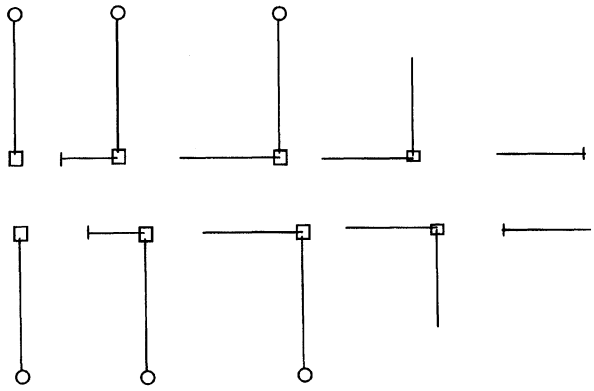


FIGURE 7 SLE net token set.

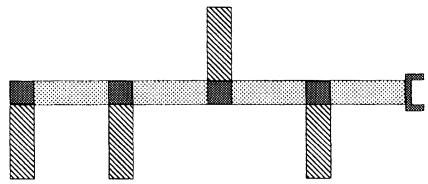


FIGURE 8(A) SLE net.

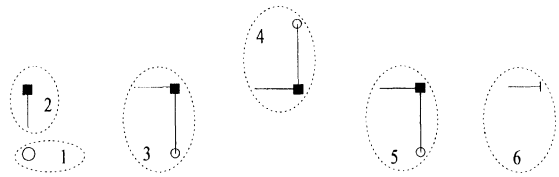


FIGURE 8(B) Net split into graphical input tokens.

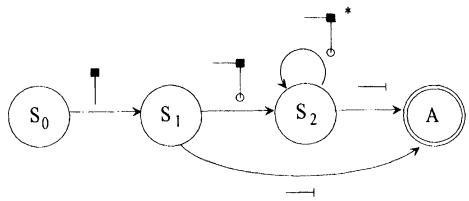


FIGURE 8(C) Functioning FSM recognizer.

There are two very important effects associated with this methodology. First, formal verification of correctness is achieved. For design automation tools, this is of paramount importance due to the nature of the regime. Second, system testing only requires a single net from each category be examined, as opposed to attempting extensive testing of randomly generated nets in an effort to achieve some degree of coverage. These concerns dominate as design automation system complexity continues to grow.

D. Implementation Strategy using Feature Vectors

Although the FSM approach provides an idea theoretical basis for net recognition and ultimate bifurcation, implementation of such a technique is burdensome at best. By mapping the FSM theory into an alternative domain a suitable implementation strategy can be derived.

The field of computer vision has employed feature vectors as a mechanism for recognition

for many years. Let us define a feature vector $\mathbf{F}_i$, which is composed of components reflecting counts of primary net elements, such as number of vias, number of ports, and number of connectors. Each of these characteristics can be counted in time proportional to one scan over the components of the net. The value of i ranges from 1 to R, where R represents the number of net categories. By examining the relationship among the components of the vector, a categorization can be accomplished.

$$\begin{aligned}\mathbf{F}_i = \{&\text{number of vias,}\\ &\text{number of ports,}\\ &\text{number of connectors(E/W),}\\ &\text{number of connectors(N/S),}\\ &\text{number of layers/backbones,}\\ &|(\text{constraints})\\ \}&\end{aligned}$$

The node labels of the tree in Figure 6(B) represent either the number of connectors contained in the net, *e.g.*, 1^C, indicating one connector. Nodes labeled E/W, N/S and E/W and N/S refer to nets with either an East (or) West facing connector, a North (or) South facing connector, or a combination of the two possibilities respectively.

Rewriting the vector components so that the number of vias is X_1, the number of ports is X_2, the number of E/W connectors is X_3, the number of N/S connectors is X_4, and number of layers or backbones is X_5, $\mathbf{F}_i$ can be written as

$$\mathbf{F}_i : \{X_1, X_2, X_3, X_4, X_5 | (\text{constraints})\}$$

where *constraints* represents the conditions imposed on the relations among the vector components. For Category 2 type nets, vector $\mathbf{F}_2$ with explicit conditions is defined as-

$$\begin{aligned}\mathbf{F}_2 :\{&X_1, X_2, X_3, X_4, X_5 | X_1 == X_2,\\ &X_3 == 0, X_4 == 0, X_5 == 1\}\end{aligned}$$

For the net portrayed in Figure 9, its feature vector, $\mathbf{F}_{\text{observed}}$, can be constructed by tallying

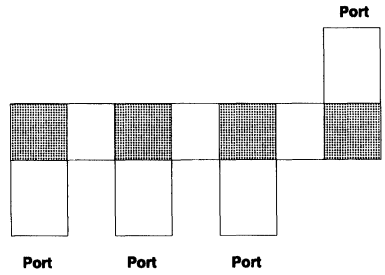


FIGURE 9 SLE net.

components during a scan of the net. The vector would appear as follows:

$$\begin{aligned}\mathbf{F}_{\text{observed}} :\{&X_1, X_2, X_3, X_4, X_5 | X_1 == 4,\\ &X_2 == 4, X_3 == 0,\\ &X_4 == 0, X_5 == 1\}\end{aligned}$$

Since the relationship among the components of $\mathbf{F}_{\text{observed}}$ satisfy the conditions of $\mathbf{F}_2$, the net can be categorized as a Type 2 net and the bifurcation process commenced.

E. FSM – Feature Vector Equivalence

We now demonstrate equivalence between the theoretically sound FSM technique and the empirically derived feature vector approach. Figure 10 shows a generic FSM. The corresponding regular expression is shown in Eq. (5).

$$\textbf{Reg\_Expression} = \mathbf{ab(cb)}^n\mathbf{d} \tag{5}$$

In this form, the strict mathematical relations among the components is established. For Eq. (5) the constraints are:{**a** = = 1}, {**d** = = 1}, {**b** $\geq$ 1}, {#**c** = = #**b**−1} Using the FSM recognizers constructed previously, and relabeling their arcs using

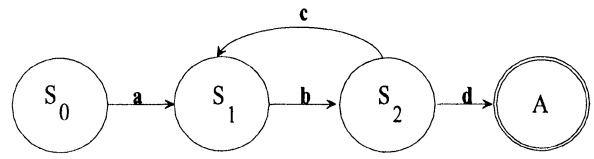


FIGURE 10 Sample FSM to generate regular expression.

formal token symbol labels from Table I, the feature vectors for all categories can be generated directly.

As an example, the process will be applied to the SLE net in Figure 9. Beginning with the farthest left port, the input tokens form the expression shown below-

PVSVHSVPHSVPHSVP

Rewriting this expression, expanding segment names, results in the following form-

P(VSeg)V(HSeg)VP(HSeg)VP(HSeg)VP

Discounting the **VSeg**(vertical segment) and **HSeg**(horizontal segment) entries, the expression can be rewritten as-

PVVPVPVP

which resolves to the regular expression form-

$$\mathbf{PV\{VP\}^*}$$

TABLE I Formal FSM transition label definitions

Symbol	Definition
P	**P**ort Connection Where Bifurcation Begins
VSV	**V**ertical **S**egment to **V**ia
HSVP	**H**orizontal **S**egment to a **V**ia with a **P**ort Junction
HSC(E/W)	**H**orizontal **S**egment to a **C**onnector (**E**ast or **W**est)
HSC(N/S)	**H**orizontal **S**egment to a **C**onnector (**N**orth or **S**outh)
VSVC	**V**ertical **S**egment to **V**ia with a **C**onnector Junction

From this expression, the feature vector conditions for this type net are written directly as-

$$\{\text{Vias} \geq 1, \text{Ports} \geq 1, \text{E/W Conn.} == 0, \text{N/S Conn.} == 0, \text{Layers} == 1, \#\text{Ports} == \#\text{Vias}\}$$

SLE nets can be broken into a number of distinct tokens. These tokens can then be used as input strings to FSM recognizers. A representative Type 3 Net and its corresponding FSM recognizer are given in Figure 11.

Beginning at the start state, $\mathbf{S_{03}}$ (corresponding to the farthest left port), a vertical segment to a via with a connector is encountered (**VSVC**). If there were no additional input tokens, the net would be accepted in state $\mathbf{A_3}$. For this example, there are now three occurrences of horizontal segments to vias with port segments emanating from them. This is represented by the **HSVP** transition to state $\mathbf{A_4}$, and the subsequent cycling back to state $\mathbf{A_4}$. The last token found is the horizontal segment to the East/West connector on the right end (**HSC(E/W)**) that causes the transition to acceptor state $\mathbf{A_5}$. If our example net did not have the left side E/W connector, then the transition path through the state machine would have been across the top path.

The feature vector concept provides the readily implementable, linear time method of categorization. Finite State Machine (FSM) theory provides the theoretical underpinnings of this approach. Regular expressions derived from these recognizers demonstrate equivalence between the theoretical foundation provided by FSMs and the set of feature vectors.

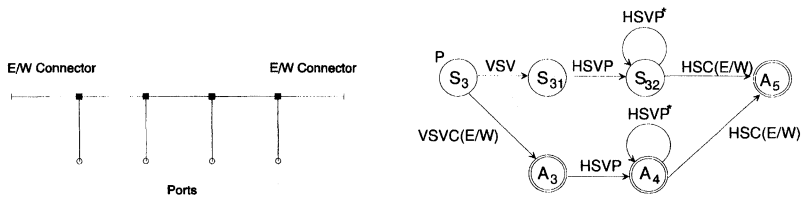


FIGURE 11 Example type 3 SLE net and corresponding FSM recognizer.

F. FSM Based Bifurcation Algorithm (Fig. 12)

```
function bifurcate
    begin bifurcation
        for each fat wire net
            bifurcatable = recognize_fat_net(fat_net)
            if bifurcatable == True then
                split_net
            else
                tag for local reroute
        end for loop
    end bifurcation

function recognize_fat_net(fat_net)
    begin
        scan components of net
        tally components to build feature vector
        test component relations
        if "recognized"  return True
         else
                return False
    end

function split_net
    begin
        duplicate vertical and horizontal segments
        fix horizontal polarities
        match polarity of port segs to horizontal segs through vias,
            re-introducing inversions where necessary using table lookup
            (Table 2 and EQ 3)
    end
```

FIGURE 12 Linear time bifurcation algorithm.

G. Analysis of the Algorithm

By using the underlying FSM theory, SLE wires can be recognized and categorized in linear time by means of a feature vector. Once categorized, the SLE wires can be split in time proportional to scanning the components, replicating those parts necessary to construct the complimentary wire of

the differential pair, and computing the via split decision at each junction. At the same time, inversions which were extracted from the problem space during the collapsing of the net list are reintroduced as part of the process. What remains to be shown is that the via split decision making and re-introduction of inversions can be dealt with in linear time.

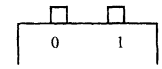
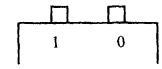
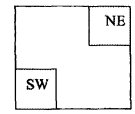
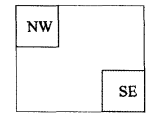
The Manhattan routing layer constraint imposed earlier guarantees that for all nets, other than the degenerate case of a single metal connection crossing a channel and connecting to a port on the opposite side, there will exist at least one horizontal segment and vias connecting that segment to ports. Since a via must exist for each connection from a port to a backbone, polarity matching is accomplished through the two via splitting options of Figure 4. Three variables affect the via splitting decision: (1) the port ordering of the differential signal pair along the edge of the standard cell, (2) cell orientation in the layout, and (3) inversion data extracted during the differential net list collapsing operation.

Since each input variable and the via split operation have two possible configurations, they can be viewed as state variables and represented by binary digits, Table II. By applying a state variable transformation, the logic operation for deciding the proper via cut can be elegantly defined by the concise formula shown in Eq. (6).

$$\textbf{Via\_cut} = (\textbf{Original\_Port\_Polarity}) \oplus (\textbf{Cell\_reflection}) \oplus (\textbf{Inversion}) \tag{6}$$

This methodology facilitates the re-introduction of inversions, and allows linear time decision making when determining the proper cut at each via during the bifurcation process.

TABEL II State variable mapping table

Variable	State	Description
Cell Reflection	0	Original Cell Orientation
	1	Cell Reflected [$x=(-x)$]
Inversion	0	No Inversion
	1	Netlist Inversion at Instance
Instance Ordering	0	
	1	
Via Cut Orientation	0	
	1	

VI. RESULTS AND CONCLUSIONS

When evaluating the results of this research, extreme care must be exercised when comparing the quality metrics generated by this approach and other solutions. In this relatively unexplored routing regime there are no established benchmarks against which one can measure the quality of the approach. For this reason, data collected from our system is only compared to the results produced by the core router (COMPASS Design Automation) routing differential pairs in a non-adjacent manner.

The first production test of the system was provided by an ARPA sponsored RISC project. All standard cell and block routing was performed by our system. Chips are currently undergoing functional testing. Closeup photos of selected chip sections are provided in Figure 13(A and B).

For analysis purposes we believe we have chosen sample routing problems which are representative of the entire class. The results of our experiments, showing total interconnect, vias and circuit area are provided in Table III. Data provided represent percent reductions (improvements) for each metric over the pure differential routing.

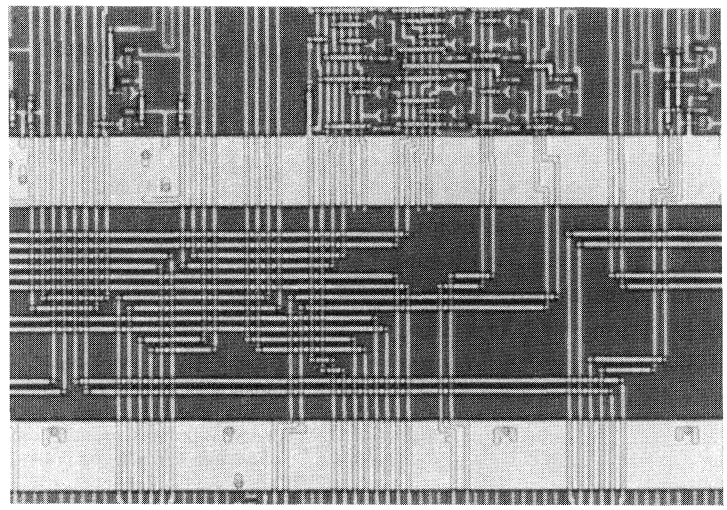


FIGURE 13(A) Standard cell area of test chip.

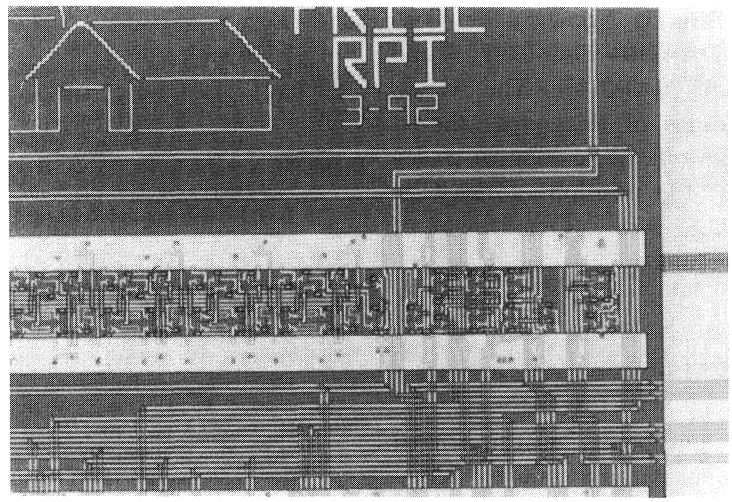


FIGURE 13(B) Block routing area.

Although our approach may appear simplistic, it is the first large scale effort aimed at solving the differential routing problem. An important test of any theory is its capacity to solve actual problems [8], and the results in Table III confirm our theoretical expectations. Our research has produced both theoretical and practical results. The theoretical underpinnings rooted in FSM theory and the basis set of bifurcatable nets are exciting. The FSM-Feature Vector equivalence shown to exist through the regular expression link provides both closure and an ideal implementation medium. The overall concept appeals to intuition and serves

TABLE III Percent reductions in quality metrics

	Nets	Vias	Chip Area	Interconnect
Datapath	972	20%	15%	31%
PCblock	212	−1%	−1%	13%
Testchip	160	12%	2%	14%
Shifter	68	7%	−9%	5%
SmallCkt	52	4%	−9%	−4%

to effectively halve the routing problem size. Finally, it produces a differential solution whose optimality is equivalent to the core router's capability, with only a linear time penalty factor.

However, nets with doglegs may cause some difficulty with our FSM approach. This may be primarily due to the number of cases that doglegs will introduce in FSM states. Our continuing research has extended the chip routing solution into the MCM domain [9]. Along with this effort, other applications of FSM Theory in the recognition arena are being investigated.

Acknowledgements

This research was sponsored in part by the Advanced Research Projects Agency under contract ARPA/ARODAAH04-93-G-0477 and DARPA/ARODAAL03-90-G-0187. The views, opinions, and/or findings of this paper should not be construed as an officail position or policy and do not represent the position of the U.S. Military Academy, U.S. Army, Department of Defense, or U.S. Government.

GLOSSARY OF TERMS

port – connection point between routed interconnect and a standard cell.

fat wire – wire whose geometry encompasses the width of two standard wires and the interwire spacing.

connector – point of connection at the end of a tack to which another block would be connected by the block level router. These can be on the north, south, east, or west edges of the standard cell area.

Manhattan constraint – for our purposes it refers to the sub-constraint that all layer 1 interconnect run vertically, and all layer two interconnect run horizontally.

FSM – Finite State Machine.

bifurcation – conceptual act of splitting or resolving fat wires into their respective two wire components for the physical realization of the layout.

References

[1] Greub, H. J., McDonald, J. F., Creedon, T. and Yamaguchi, T. (1991). "High-Performance Standard Cell Library and Modeling Technique for Differential Advanced Bipolar Current Tree Logic," *IEEE Journal of Solid-State Circuits*, pp. 749–762.

[2] Barish, A. E., Eckhardt, J. P., Mayo, M. D., Svarczkopf, W. A. and Gaur, S. P. (1992). "Improved Performance of IBM Enterprise System/9000 bipolar logic chips," *IBM Journal of Research and Development*, pp. 829–834.

[3] Badeau, R. W. *et al.* (1992). "A 100-MHz Macropipelined VAX Microprocessor," *IEEE Journal of Solid-State Circuits*, pp. 1585–1598.

[4] Bakoglu, H. B. (1990). *Circuits, Interconnections, and Packaging for VLSI*, Addison-Wesley.

[5] Horowitz, E. and Sahni, S. (1984). *Fundamentals of Computer Algorithms*, Computer Science Press.

[6] McNaughton, R. (1982). *Elementary Computability, Formal Languages and Automata*, Prentice-Hall, Inc.

[7] Loy, J. "Differential Routing," *Doctoral Thesis*, RPI, Aug. 1993.

[8] Joy, D. A. and Ciesielski, M. J., "Layer Assignment for Printed Circuit Boards and Integrated Circuits," *Proceedings of the IEEE*, pp. 311–331, February, 1992. Tutorial on routing problem.

[9] Ohtsuki, T. (1986). *Layout Design and Verification*, Elsevter Science Publishing Company.

[10] Asbeck, P. M., Chang, M. F., Higgins, J. A., Sheng, N. H., Sullivan, G. J. and Wang, K. (1989). "GaAlAs/GaAs Heterojunction Bipolar Transistors: Issues and Prospects for Application," *IEEE Trans. on Electron Devices*, pp. 2032–2041.

[11] Philhower, R., Greub, H., VanEtten, J., Loy, J., Kyungsuc, Nah, Garg, A., Tsen, T. and McDonald, J., "An 800-ps 32-bit Adder for a 1.0 GHz RISC Processor," *IEEE JSSC*, pending.

[12] Kyung-suc Nah, Greub, H. and McDonald, J., "A 200-ps Register File for a 1-ns RISC Processor," *IEEE JSSC*, pending.

[13] Hu, T. C. and Kuh, E. S. (1985). "Theory and Concepts of Circuit Layout," *VLSI Circuit Layout: Theory and Design*, IEEE Press.

Authors' Biographies

M. S. Krishnamoorthy received the B.E. degree (with honors) from Madras University in 1969, the M. Tech degree in Electrical Engineering from

the Indian Institute of Technology, Kanpur, in 1971, and the Ph.D. degree in Computer Science, also from the Indian Institute of Technology, in 1976.

From 1976 to 1979, he was an Assistant Professor of Computer Science at the Indian Institute of Technology, Kanpur. From 1979 to 1985, he was an Assistant Professor of Computer Science at Rensselaer Polytechnic Institute, Troy, NY, and since, 1985, he has been an Associate Professor of Computer Science at Rensselaer. Dr. Krishnamoorthy's research interests are in the design and analysis of combinatorial and algebraic algorithms and programming environments.

James R. Loy (M'93-Present) received the B.S. degree from the U.S. Military Academy, West Point, NY, in 1974; the M.S. degree in computer science and the M.E. degree in computer engineering from Rensselaer Polytechnic Institute, Troy, NY, in 1986; the M.A. degree in national security and strategic studies from the U.S. Naval War College in 1987; the Ph.D. degree in computer engineering from Rensselaer Polytechnic Institute in 1993.

After instructing at West Point from 1987–1990, he returned to the Academy in 1993 where he served as an Associate Professor in the Department of Electrical Engineering and Computer Science. His research interests include VLSI routing algorithms, fuzzy logic, and optical electronic devices.

He served as the Chairman of Mid-Hudson Section of the IEEE in 1996. He recently retired from the military and is working as Manager of Technology Operations, Tiger Management, L.L.C. in New York City.

John F. McDonald was born on Jan. 14, 1942 in Narberth, PA. His education included a BSEE at MIT in 1963, followed in 1965 by an MEng. at Yale University in New Haven, Connecticut, and finally by the Ph.D. at Yale in 1969. He spent a brief period as MTS at Bell Laboratories in 1965. He held the position of lecturer at Yale in 1969, and was appointed assistant professor there in 1970. In 1974 he joined the Rensselaer Polytechnic Institute as an associate professor, where he is currently a full professor in the Department of Electrical, Computer, and Systems Engineering. He was one of the founding members of the RPI Center for Integrated Electronics in 1980. His publication record consists of 200 articles approximately a third of which are journal articles. He holds 7 patents, and an additional 8 disclosures. He has supervised 23 Ph.D. dissertations, 90 Masters Thesis projects, and participated in 80 contracts or grants. He is listed in 24 Compilations of Technical Recognition including Who's Who in the World, Who's Who in America, and American Men and Women of Science. He has been past LEOS representative for the I.E.E.E. Solid State Circuits Council, and currently an Associate Editor for I.E.E.E. VLSI Transactions. His background includes work from a wide spectrum of technology ranging from radar/sonar signal processing, communications, controls, through IC fabrication, multilevel interconnections, polymeric dielectrics, microwave circuits, electron and ion beam technology, and GHz testing. His current research interests include VLSI and Computer Design with recent emphasis on very high speed electronic packaging, GaAs HBT/MESFET, InP or SiGe HBT circuits and RISC processors.