A New Algorithm For Signal Flow Determination In CMOS VLSI

Ahmed Riadh Baba-ali

CDTA - Laboratoire de Microelectronique
128, rue Mohamed Gacem EL MADANIA ALGER ALGERIA

-Abstract-

This paper presents a new algorithm that automatically classify transistors as bidirectionals or unidirectionals in any CMOS circuits. This algorithm uses simple rules based on node strength as well as local transistor direction and path informations. Since circuit transistors are processed in linear running time, this approach enables the algorithm to be fast.

1) Introduction

Many MOS static analysis tools such as timing analyzers performs on circuits such tree walk processes. The circuit transistors are explored and parsed many time in order to deduce informations such as delay, strength etc..., or to find subcircuits sharing common properties such as classes [1], or stages [3].

These processes can consume a large amount of the total running time of the analyzer, and can also lead to erroneous results if transistor signal flow are not precisely known.

If a transistor is known to be unidirectional then we need only to explore the source or drain, and hence could cut the search time in half. Another benefit of identifying unidirectional transistors is that many false paths can be eliminated. This lead to a reduced analysis time and an enhanced accuracy.

For this purpose, a pre-procesing of the network is necessary to identify transistor signal flow.

2) Previous approaches

For the Crystal [3] timing analyzer, transistor flows are reported manually during the layout design phase and are propagated through the netlist by a special extractor. This approach has the drawback of needing special tools, and extensive user intervention.

For the TV [2] timing analyzer, transistors ratio rules are used to determine a signal flow. However, this approach is NMOS oriented, and is virtually unusable for CMOS circuits.

For the Pearl [4] timing analyzer, an automatic and more general approach is used. It consists of using complex rules. However, this approach causes circuit transistors to be processed several times and have difficulties to deal with circuits using pass transistors arrays.

3) Graph formulation and definitions

CMOS network are modeled in the litterature [5] as undirected graphs, where each graph node represent a node circuit and where each graph edge represent a transistor (drain-source or channel connection). A network is considered as a set of connected components called transistor groups [1]. For signal flow determination purpose additional features are required which are as follow :

* Power sources and input nodes are represented by what is called source nodes. Output ones as well as nodes with transistors gates are represented by what is called sink nodes.
* Each node has a strength as defined by Bryant [1] with the same relative strengths
(ie : input > pulled >charged > supercharged). These strengths express the electrical fact that strong currents overides weak ones.
However, each strength is newly defined as follow :

- input node : any source
- supercharged : sink node
- pulled : any node connected by a transistor to a source node or could be any node connected to a pulled node.
- charged : any node connected by a transistor to a supercharged or could be any node connected to a charged node.

* A mixed node is a network node where both PMOS and NMOS transistors are connected.

0-8186-2610-0/92 $3.00 © 1992 IEEE

1- If a transistor is directly connected to an input node then the signal flow can only be an output signal from the input node.
2- If only one transistor connected to a node is unknown and all other signal flows are input, then the unknown transistor flow must be an output signal.
3- Given an analysis direction starting from source nodes reaching an unknown transistor, and also if this transistor is connected to a mixed or supercharged node, then its flow is an output signal.
4a- Given an analysis direction reaching an unknown transistor, If all paths coming to this transistor reach only source nodes, then this transistor flow must be an input signal.
4b- Given an analysis direction, reaching an unknown transistor, If all paths coming to this transistor reach only mixed nodes or a supercharged, then this transistor flow must be an output signal.
5- Given an analysis direction starting from sources and reaching an unknown transistor, if all paths coming to this transistor reach source and mixed nodes, and if there exist at least one transistor with output signal flow, then the unknown transistor flow must be an input signal.

Each rule is assigned a decreasing priority, so the algortihm tries to apply rules starting from the first to the last one. However for efficiency purpose, each rule must be applied at the appropriate moment of the analysis as will be explained in the following section.

Note that the backward rules are the duals of the forward ones. In order to obtain the backward rules from the forward ones, the following modifications must be done :
Into rules 1 and 2 :
* Inverse signal flows and path directions.
* Replace source nodes by sink ones.
Into rules 3, 4 and 5 :
* Replace supercharged or mixed nodes by mixed ones with all connected transistors locked.
* Replace source nodes by sink ones.
* Replace signal flows which are in the same direction than analysis direction, by bidirectional flow.

5) Algorithm description

The algorithm assume that all source and sink nodes of each group are identified and stored respectively into lists called forward and backward lists. This process can be done in linear running time with the number of network nodes by exploring each group. One problem could occur with source nodes due to the fact that they are common to all the circuit. Therefore, it is difficult to find all the transistors connected to a source and belonging to the same group. One possible solution is to use the node splitting technique [1], but it has the drawback to need a modification of the network data structure. The solution we have adopted consists in storing directly nodes which are connected to source ones by transistors into the forward list.

We assume within each transistor record, the following fields exist :
- flow : which can be bidirectional, input or output, it is initially set to bidirectional.
- lock : is a flag indicating whether transistor signal flow has been determined. It is initially set to false.

We assume that within each node record, the following fields exist :

- Lock : is a flag indication whether all the paths starting from it, has been explored. It is initially set to false.
- NbTr : indicates the number of transistors connected by diffusion nodes to the current node.
- Nbpaths : indicate the number paths starting from the current node remaining to be explored. It is initially set to NbTr.
- NbIn : indicates the number of input signals (relatively to the current node) reaching the current node. It is initially set to 0.
- NbOut : indicates the number of output signals (relatively to the current node) reaching the current node. It is initially set to 0.
- Visited : is a flag indicating that the current node is being analyzed. This flag is used to detect feed back paths. It is initially set to false.
- Inlist : is a flag indicating that the current node is stored into either forward list or backward list.

The forward pass is decomposed into two phases. The first phase starts from source nodes that have only one transistor connected to them and tries to apply rule 2 as long as possible. Typically, this phase will process serial sequences of transistors starting from sources. During this phase, signal flows are determined directly as long as parallel combinations of transistors are not found. When this happens, the last analyzed node is stored at the end of the forward list and the phase 1 is stoped. Experience with tested circuits has showed us that this phase is sufficient to process the great majority of circuit transistors. However, if transistors remain not processed from the previous phase then the second phase of the forward pass starts from nodes still stored into the forward list and which have not been yet processed. Then it recursively explores all possible transistors as long as possible (ie : until reaching a source or a mixed or a supercharged node).

During the second phase of the forward pass transistor identification is done when all possible paths have been explored (ie : during the backtrack phase), the algoritm tries to apply successively rules 3, 4 and 5. When a transistor signal flow is precisely determined, it is locked (ie : it becomes virtually non-existant). In the same manner, nodes are also locked when all paths have been explored. During exploration, if a sink or source node is encoutered, then this information must be propagated on the previous ones. For this purpose within each node record is provided boolean flags called sinkfound and sourcefound fields. They are updated after each backtrack from a node and therefore path informations are progressly propagated to previous nodes.

In order to avoid the algorithm getting lost in circuits that requires inordinate amount of computations, the number of paths remaining to be analyzed is computed at each time. If this number exceeds a user defined limit, then the

* A neighbour of a node relatively to a transistor is defined as the node which can be reached through this transistor. For example, in Fig 1, the neighbour of node B through transistor 1 is the node A.
* During network analysis for an unknown transistor, signal flow is relatively considered to one of its diffusion node (generally the node being currently analyzed) as :
An input signal if the signal flow goes from the node, and an output signal if the signal flow comes to the node. For example in Fig 1, signal flow of transistor 3 is an output signal relatively to node C and is an input signal from node D.

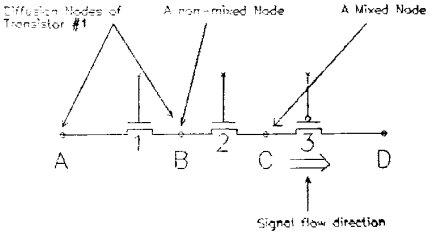

Fig 1: Illustration of transistor terminology

4) Algorithm Principle

The objective of our algorithm is to identify the maximum of unidirectional transistors in CMOS circuits with a minimum time.

In order to improve algorithm performance, circuits are considered to be partitioned into three regions : Pullup, Pulldown and intermediate regions. Pullup and Pulldown are considered to be composed only by the same type of transistors (N or P transistors). Each region is going to be processed separetely. However, the algorithm can even process correctly circuits where this assumption is not true.

The algorithm uses a depth-first approach. Network nodes are processed from source or sink nodes as far as possible in order to find their strength. Two recursive passes are performed : the first one is initiated from source nodes and tries to reach source or sink nodes. The second pass performs the same steps but from sink nodes as long as current node strength is greater or equal than the neighbour's one. Note that during both passes, if a feedback path is detected then the analysis will simply backtrack from it and explore the other paths.

Node strength are determined in order to deduce transistor signal flow , by using the following rules:

a) Information flow from strongest nodes to weakest ones.
b) When nodes have the same strength, additional rules are necessary. For the sake of simplicity, lets consider only the forward pass. Signal flow is deduced by applying one of the following rules (see Fig 2) :

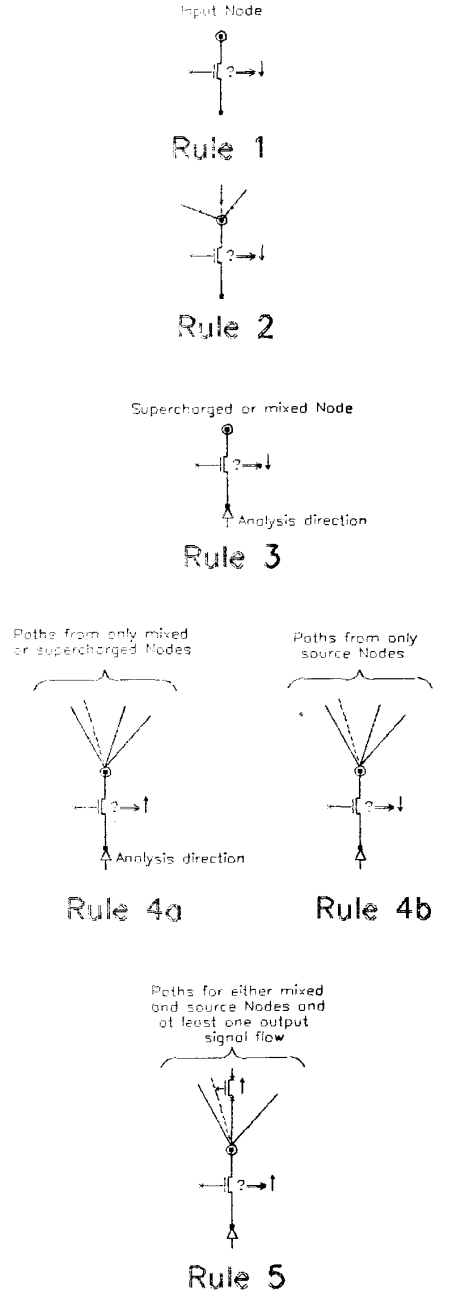

Fig 2: Forward Rules

exploration will stop and this information will be propagated to previous nodes with the same technique used with sourcefound and sinkfound flags.

Once this procedure has been applied to a circuit, generally few transistors remain unprocessed. However, the backward pass is necessary to identify these transistors. This pass can be considered as the dual of the forward one. It uses the backward rules (which are also the duals of the forward ones) and starts from sink nodes rather than source ones. It also tries to reach source or sink nodes, as long as current node strength is little than pulled. (Note that mixed nodes has not been assigned a strength during the forward pass). Moreover, rather to determine signal flow as input or output like in the forward pass, signal flow will be set opposite to the analysis direction if only mixed or source nodes have been found. In the same time, signal flows will be set bidirectionals if only sink nodes are found.

All this procedure must be repeated for all the groups connected to the current one by transistor gates. Then for all connected groups once processed, the procedure is also repeated for all the neighbour's groups, and so on until all circuit transistors are identified.

6) Algorithm analysis

Our algorithm has a linear running time, since even for the worst case (ie : highly parallel circuits), the transistors are processed only two times, once from drain to source and once from source to drain. The linearity of the algorithm is achieved due to the use of flags for network nodes and transistors. Each time node strength and signal flow are precisely identified, then these nodes and transistors are locked avoiding other explorations.

7) Conlusion

This algorithm is part of our local timing analyzer called MOSTAFA [6].
It has been successfully applied on a wide range of circuits using different styles and techniques such as :
- static and dynamic circuits
- complementary, ratioed, precharge and pass logics.

These circuits include :
- A full transmission gate adder based on "Tiny XOR".
- A manchester carry lookahead adder.
- Various barrel shifters.
- A PLA.
- Dynamic and static latches.

This algorithm uses a fast and safe approach, since some unidirectional transistors may be left bidirectionals but in all cases, no bidirectional devices can be marked unidirectional.

Acknowledgments

The author would like to thank Mr A. Belaouar for his comments and suggestions as well as Mrs S. Baba-ali, Mr Y. El haffaf, A. Bouhraoua and Y. Bourai for their help.

Appendix (Main procedure of the forward pass)

```
BEGIN
  Select all transistor connected to source or input nodes
  FOR each transistor t
    cn=neighbour of source node relatively to transistor t
    SetOut (t,source)
    set lock flag of transistor t
    Set InList flag of node cn
    store cn into end of Forward list
    set source flag of cn
  END
  sort forward list in increaising order of NbIn field
  WHILE forward list not empty
    cn = first node of forward list
    remove cn from forward list
   IF cn locked THEN continue
   reset InList of node cn
  IF cn locked THEN continue
  IF cn.NbTr - cn.NbIn = 1
     Phase1Forward (cn)
  ELSE
    FOR each transistor t not locked
        n = neighbour of cn relatively to t
        set Visited flag of n
        Phase2Forward (t,cn,n)
        reset Visited flag of n
    END
  END
END
```

References

[1] Bryant
An algorithm for MOS logic simulation, in Lambda, 4th qtr. pp 46-53, 1980.

[2] Norman P. Jouppi
Derivation of signal flow direction in MOS VLSI, in IEEE Trans. on computer-aided Design vol. CAD-6, No 3, pp 480-490, MAY 87.

[3] John K. Ousterhout
A switch-level timing verifier for digital MOS VLSI, in IEEE Trans. on computer-aided Design Vol CAD-4 No 3 JULY 85.

[4] James J. Cherry
PEARL a CMOS timing analyzer, in 25 th DAC pp 148-153, 1988.

[5] T. Uehara, W.M. Vancleemput
Optimal layout of CMOS functional arrays, in IEEE trans. on computers VOL. C-30 No 5, pp 305-312, MAY 81.

[6] A.R. Baba-ali & C. Benmehrez
MOSTAFA a timing timing analyzer for CMOS digital circuits, in Proceeding ICM'90 Damascus, Oct 90.

[7] R. Tarjan
Depth-first search and linear graph algorithms,
in SIAM computer, Vol. 1 No 2, pp 146-160, June72.