LOGIC SYNTHESIS FOR PASS-TRANSISTOR DESIGN

VOJIN G. OKLOBDZIJA*, B. DUCHÊNE**
*Electrical and Computer Engineering Department
University of California
Davis, USA
vojin@ece.ucdavis.edu
**Alcatel Paris, FRANCE

ABSTRACT

New logic CMOS families using pass-transistor circuit techniques have recently been proposed with the objective of improving speed and power consumption [2-8]. The Double Pass-Transistor Logic, developed by Hitachi [6] in 1993 has proven that in 0.25 μm CMOS technology, DPL full adder is as fast as that of CPL. In this work, a method for synthesis of pass-transistor logic has been developed. It has been shown by simulation that in terms of speed this family outperforms standard CMOS design. The logic developed using described techniques is also advantageous in terms of power as compared to static CMOS. The new logic presented in this paper represents an improvements over DPL, made by elimination of the redundant branches and rearrangement of signals.

NEW LOGIC FAMILY: DVL

New logic gate: DVL (Dual Value Logic) gate was obtained from DPL by eliminating the redundant branches and rearrangement of signals. Signal rearrangement resulted in NAND gate configuration which is faster than DPL (60pS vs. 75pS), whith the AND half being faster. These simplifications, illustrated in Fig. 1, 2 and 3, preserve the advantages of DPL gates.

We chose a faster half (Fig. 3.) from Fig.1. and from Fig.2 to construct DVL gate. The resulting DVL gate contains total of 8 transistors compared to DPL consisting of 4 transistors of each type. There is a total of 9 inputs in DVL versus 12 in DPL resulting in a smaller capacitive load of DVL gate. In DVL, 3 inputs are connected to the transistor source and 6 to the gate (3 to p-type and 3 to n-type). In DPL 4, inputs are connected to the source and 8 to the gate (4 to p-type and 4 to n-type transistors). A similar method can be used to build the NOR/OR gates.

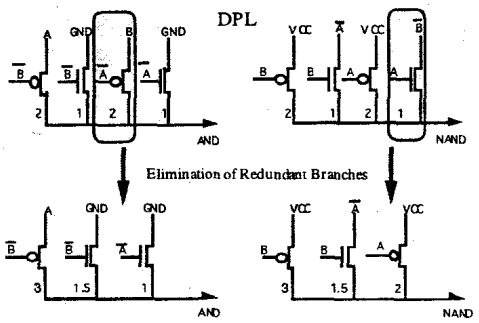

Fig.1. *Elimination of redundant branches*

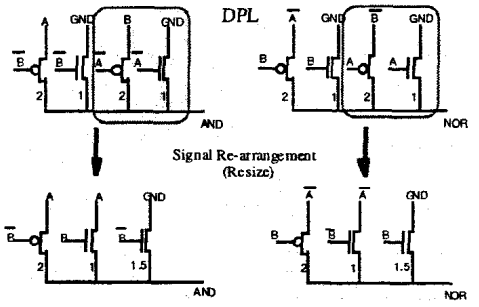

Fig.2. *Signal Re-arrangement*

SYNTHESIS METHOD FOR DVL

There are several methods used to synthesize circuits from a Boolean function, which use basic logic gates, but none achieves optimal results. At present, there is no known algorithm to find minimal multi-stage logic circuits. The proposed method for synthesis of DVL, is based on transistors instead of logic gates. In place of conjointly assembling several basic gates, functions are synthesized at the transistor level. In addition, a program implementing this method has been developed to prove the efficiency of the theory presented.

The key point in DVL synthesis consists of employing Karnaugh-Map at the transistor level. Thus, we are not cascading several logic gate levels (NAND/AND or NOR/OR), but building functions by directly using several transistor levels in series. The choice of *"pseudo Karnaugh-Map"* for illustration purposes was used, because its explanation is simple, but a *"pseudo Quine*

(0-7803-3062-5)

McCluskey" technique has been adopted instead. Usually, the general Karnaugh-Map is covered by loops of "0" or "1", in such a way that a minimized Sum of Products form is obtained. In our case, four classes of loops are allowed (as illustrated in Fig. 4) to directly synthesize the final circuit. Accumulating all the loops necessary to cover the Karnaugh-Map yields the resulting circuit.

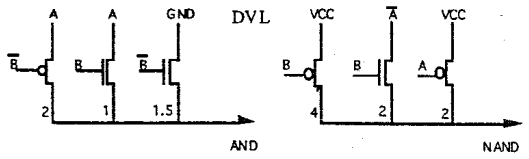

Fig.3. *Resulting DVL Gate*

The synthesis of circuits from functional expressions can be summarized in Fig. 5. which explains the circuit synthesis stages. There are two important steps in this flowchart which are critical. The first step determines how to optimally cover the Karnaugh-Map at the transistor level for an N-input circuit . The second step, based on circuit synthesis rules, deals with determining suitable sizes for transistors in order to minimize circuit delays. In the last phase, reduction of the number of transistors is accomplished from the list of nodes, components, and nets.

RULES FOR CIRCUIT SYNTHESIS

(1) A loop corresponds to one branch.

(2) The loop cover represents the control variables assigned to the gates of transistors in series.

(3) The input to the branch is: (a) GROUND for a "0" loop, (b) V_{CC} for a "1" loop, (c) Variable for "01" or "10" loop

(4) For a "0" loop we use an nMOS transistors.

(5) For a "1" loop we use a pMOS transistors, but add a twin branch assembled with nMOS transistor such that the control variables are of the opposite value to those of the pMOS transistors. This is done in order to avoid speed degradation.

(6) For a "10" or "01" loop, we use nMOS and pMOS branches. This concept has two beneficial results which are the basics for DPL: (a) full swing (b) avoidance of speed degradation by using nMOS path in parallel to pMOS.

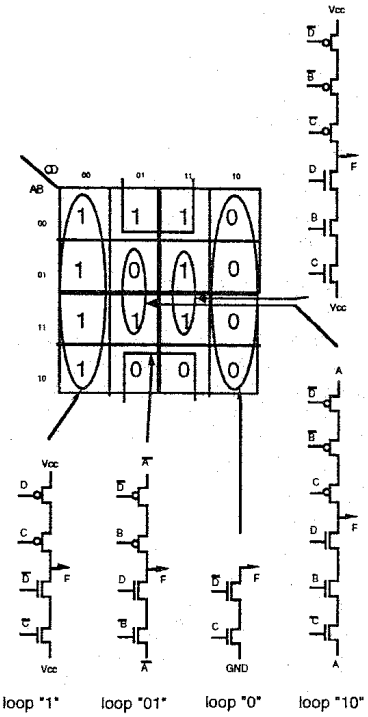

Fig. 4 *Loops allowed for 4 inputs*

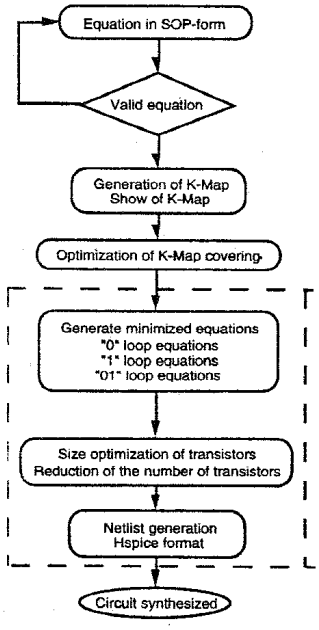

Fig. 5 *General Flowchart*

RULES FOR TRANSISTOR SIZING

In general, transistor sizes are fixed in standard NAND/AND, NOR/OR, XNOR/XOR gate synthesis. However, the transistor size has an effect. Because of undesirable input configurations, there exists a slow path which can not be avoided. Therefore, transistor sizes need to be enlarged to match the speed of the faster paths. By choosing bigger transistors, the input capacitance and the output capacitance of circuit lead to increased loading because the size of these capacitances is increased. Thus, a trade-off occurs in determining optimal transistor sizes for the different branches of the logic network.

ELIMINATION OF REDUNDANT TRANSISTORS

A reduction in the number of transistors is automatically accomplished during the net-list generation by the synthesis program. At this step, we eliminate redundant transistors by examining the resulting net-list. This process involves merging of two pMOS or two nMOS transistors with the same control variable that belong to a parallel branch, as shown in Fig. 6.

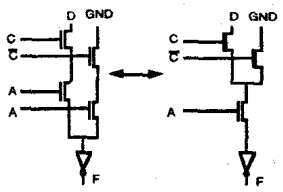

Fig. 6. *Circuit simplification*

DVL SYNTHESIS RESULTS

The results were compared not only to the Conventional CMOS but also to DPL circuits [6], CPL [4], and CPL circuits using lean cells [8]. In comparison with DPL circuits, there is 15% to 50% improvement in speed while in comparison with CPL (using lean cells [8]) this improvement is at least 10%. However, the exact speed improvement is dependent on each particular circuit. In comparison with conventional CMOS our logic shows performance improvement of up to 43% while the improvement in size ranges around 20%. Improvement regarding *the power consumption* is 30% to 50%. The technology used is 1-μm CMOS, and the simulations and tests were performed using H-SPICE.

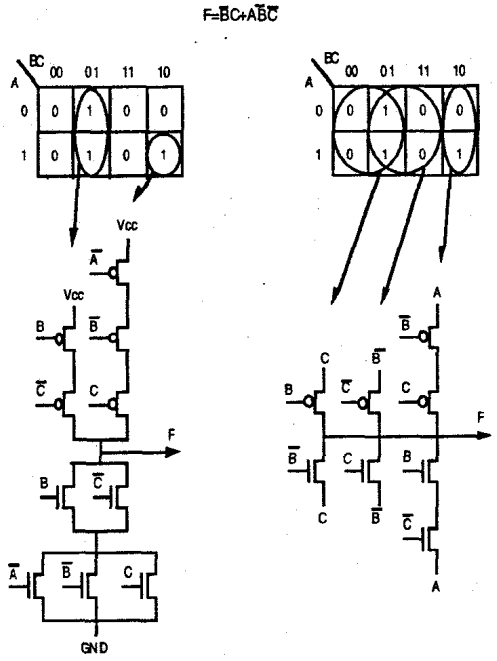

Fig. 7. *Example showing DVL and conventional CMOS*

REFERENCES

[1] L.G. Heller, W.R. Griffin, et al, "Cascode Voltage Switch Logic : A Differential CMOS Logic Family" *1984 IEEE International Solid-State Circuits Conference*, vol. 27, pp. 16-17 February 1984.

[2] KM Chu and DL Pulfrey, "A Comparison of CMOS Circuit Techniques : Differential Cascode Voltage Switch logic Versus Conventional Logic" *IEEE J.Solid State Circuits*, vol. SC-22, pp. 528-532, 1987.

[3] F.S. Lai and W. Hang, "Differential Cascode Voltage Switch with Pass Gate Logic Tree for High Performance CMOS Digital Systems", *1993 International Symposium on VLSI Technology, Systems and Applications*, pp. 358-362, May 1993.

[4] Yano, K, et al, "A 3.8 ns CMOS 16 *16-b Multiplier Using Complementary Pass-Transistor Logic", *IEEE J. Solid State Circuits*, vol. 25, p 388-395, April 1990.

[5] Akilesh Parameswar, et al, "A Swing Restored Pass-Transistor Logic Based Multiply and Accumulate Circuit for Multimedia Applications", *IEEE 1994 Custom Integrated Circuit Conference*, pp. 278-281.

[6] Makoto Suzuki, et al, "A 1.5 nS 32 b CMOS ALU in Double Pass-Transistor Logic", *1993 ISSCC Dig. Tech. Papers*, pp. 90-91, February 1993.

[7] KM Chu and DL Pulfrey, "Design Procedures for Differential Cascode Voltage Switch Circuits", *IEEE J.Solid State Circuits*, vol. SC-21, no 6, December 1986.

[8] Yano, K, et al, "Lean Integration : Achieving a Quantum Leap in Performance and Cost of Logic LSIs", *IEEE 1994 Custom Integrated Circuit Conference*, pp. 603-606.