General Method in Synthesis of Pass-Transistor Circuits

Dejan Marković, Borivoje Nikolić,
and Vojin G. Oklobdžija, *Fellow, IEEE*

Abstract -- **A new pass-transistor circuit synthesis method is presented in this paper. Several pass-transistor logic families were introduced recently, but no systematic synthesis method is available that takes into account impact of signal arrangement on circuit performance. The method is applied to generation of basic two-input and three-input logic gates in CPL, DPL and DVL, but it is general and can be expanded to synthesis of a random pass-transistor circuit.**

I. Introduction

An extensive consideration was given to the use of pass-transistor logic networks in the design of high-speed digital systems during the last decade, [1]-[12]. However, no systematic approach in the synthesis of pass-transistor circuits exists, that takes into account the impact of circuit technology and signal arrangements on circuit performance. Traditional approach to pass-transistor circuit synthesis was based on binary decision diagrams (BDD), which uses both NMOS and PMOS devices in circuit implementation. BDD approach does not guarantee full swing of output voltage – it can be degraded by the threshold voltage of NMOS or PMOS devices for some input test vectors [6]. Another commonly used design practice was library-based design, with library cells realized as combination of simple logic gates. Therefore, more attention should be directed to optimization of pass-transistor logic gates, which is often overlooked.

In the next section, we give brief overview of previously known pass-transistor logic techniques. Section III presents the rules for algorithmic and systematic synthesis of complementary and dual logic functions, with emphasis on different signal arrangements.

II. Pass-Transistor Logic Families

A. Complementary Pass-Transistor Logic (CPL)

Complementary pass-transistor logic (CPL) [7] consists of complementary inputs/outputs, an NMOS pass-transistor network, and CMOS output inverters. The circuit function is implemented as a tree consisting of pull-down and pull-up branches. Since the high level of pass-transistor output nodes is degraded by the threshold voltage of NMOS transistor, output signals have to be restored by CMOS inverters. CPL has traditionally been applied to the full adders in the multiplier circuits and it has been shown to result in high-speed due to its low input capacitance and high logic functionality.

B. Double Pass-Transistor Logic (DPL)

To avoid problems of reduced noise margins in CPL, twin PMOS transistor branches are added to N-tree in double pass-transistor logic (DPL) [8]. This addition results in increased input capacitance but its symmetrical arrangement and double-transmission characteristics compensate for the speed degradation arising from increased input capacitance. The full swing operation improves circuit performance at reduced supply voltage.

C. Dual Value Logic (DVL)

The main drawback of DPL is its redundancy, i.e. more transistors than actually needed for the realization of a function. To overcome the problem of redundancy a new logic family, dual value logic (DVL) [9], [10], is derived from DPL. It preserves the full swing operation of DPL with reduced transistor count. As explained in [9], DVL circuit can be derived from DPL circuit in three steps, consisting of:

- Elimination of redundant branches,
- Signal rearrangement (resize),
- Selection of the faster halves.

However, the load on different input signals is not necessarily symmetric and circuit optimization requires more design effort.

III. General Synthesis Method

The logic gate design presents a systematic implementation of a logic function. When new pass-transistor families were introduced, [7], [8], the emphasis was given on their suitability for block design, and less attention was paid to tradeoffs in the design of basic logic gates. The method presented here is based on Karnaugh-map coverage and circuit transformations as an approach

Dejan Marković is with the Dept. of Electrical Engineering and Computer Sciences, University of California, Berkeley, CA 94720-1772, USA, e-mail: dejan@eecs.berkeley.edu

Borivoje Nikolić is with the Dept. of Electrical Engineering and Computer Sciences, University of California, Berkeley, CA 94720-1770, USA, e-mail: bora@eecs.berkeley.edu

Vojin G. Oklobdžija is with Integration Corp., Berkeley, CA 94708, USA, e-mail: vojin@nuc.berkeley.edu

0-7803-5235-1/99/$10.00 © 1999 IEEE

to logic gate synthesis. In this section, we present algorithmic method for synthesis of basic logic gates in three conventional pass-transistor techniques.

A. CPL

Logic function can be implemented in CPL following the algorithm given below:

1. Cover the Karnaugh-map with largest possible cubes (overlapping allowed)
2. Express the value of the function in each cube in terms of input signals
3. Assign one branch of transistor(s) to each of the cubes and connect all the branches to one common node, which is the output of NMOS pass-transistor network

Complementary function can be implemented from the same circuit structure by applying complementarity principle, as given below. By applying duality principle, a dual function is synthesized.

Complementarity Principle: Using the same circuit topology, with pass signals inverted, complementary logic function is constructed in CPL.

Duality Principle: Using the same circuit topology, with gate signals inverted, dual logic function is constructed. Following pairs of basic functions are dual:

- AND-OR (and vice-versa)
- NAND-NOR (and vice-versa)
- XOR and XNOR are self-dual (dual to itself)

Proof of complementarity principle is trivial since the pass variables are directly passed from inputs to the outputs, so for obtaining complementary function an inversion of pass variables is needed.

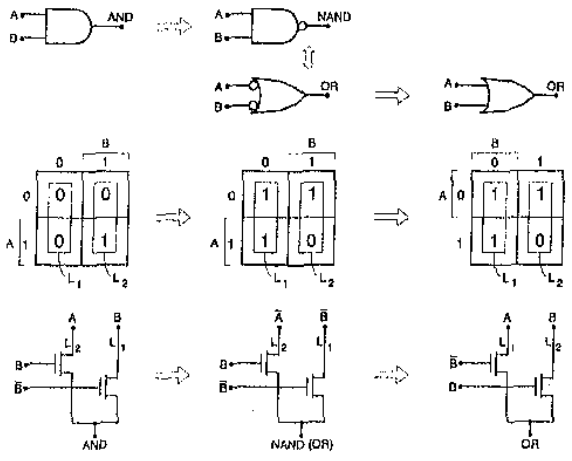

Fig. 1. Construction of dual logic function

Proof of duality principle follows from De Morgan rules and it is illustrated on the example how OR gate can be obtained from AND gate, Fig. 1.

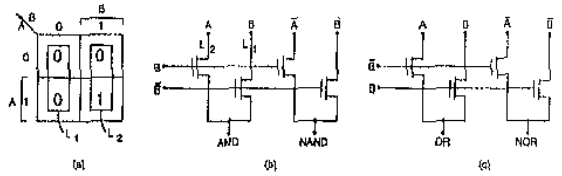

Fig. 2. Two-input CPL logic: (a) AND function Karnaugh map, (b) AND/NAND circuit, (c) OR/NOR circuit

The logic gate synthesis procedure is shown on two-input AND example, Fig. 2*a*, *b*. All the input vectors of Fig. 2*a* are covered with cubes L_1 and L_2. The value of the function covered with cube L_1 is equal to B, which becomes pass signal at the source of transistor branch, representing L_1. Branch L_1 is driven with $\overline{B}$ on its gate. Therefore B is passed when $\overline{B}$ is high. Similarly for the cube L_2 pass signal is input variable A, driven by gate signal B (A is passed when B is high). These two cubes are represented with two NMOS transistor branches, implementing the desired function. By applying complementarity principle on AND circuit we obtain NAND circuit, Fig. 2*b*. Furthermore, by applying duality principle on AND we synthesize two-input OR function. NOR circuit is generated from OR (complementarity) or from NAND (duality), Fig. 2*c*.

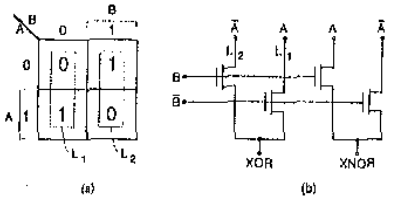

Fig. 3. Two-input CPL logic: (a) XOR function Karnaugh map, (b) XOR/XNOR circuit

Implementation of 2-input XOR/XNOR function is shown in Fig. 3.

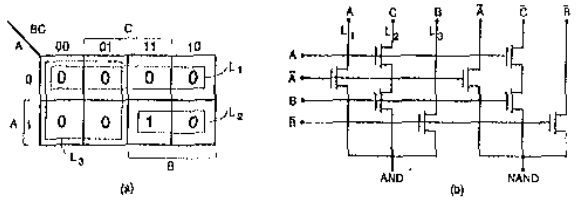

Fig. 4. Three-input CPL logic: (a) AND function Karnaugh map, (b) AND/NAND circuit

The synthesis of three input AND function is shown in Fig. 4. The cubes L_1 and L_3 are overlapped. It has a consequence that the corresponding branches are active for

the input vectors at which the cubes are overlapped. That saves the circuit area.

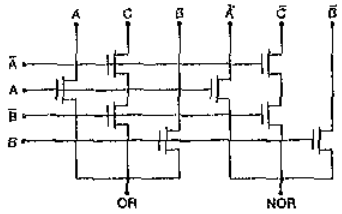

Fig. 5. Three-input CPL logic: OR/NOR circuit

Three-input OR/NOR circuit is shown in Fig. 5. Three-input XOR/XNOR circuit in CPL is implemented as cascade of two-input XOR/XNOR modules [7].

B. DPL

Random logic function in DPL can be synthesized from Karnaugh-map as follows:

1. Two NMOS branches can not be overlapped covering logic 1s. Similarly, two PMOS branches *can not be overlapped covering logic 0s.*
2. Pass signals are expressed in terms of input signals or supply. Every input vector has to be covered with exactly two branches.

At any time, excluding transitions, exactly two transistor branches are active (any of the pairs NMOS/PMOS, NMOS/NMOS and PMOS/PMOS are possible), i.e. they both provide output current.

Complementarity Principle: Complementary logic function in DPL is generated after the following modifications:

- Exchange PMOS and NMOS devices
- Invert all pass and gate signals

Duality Principle: Dual logic function in DPL is generated when PMOS and NMOS devices are exchanged, and signals V_{DD} and *GND* are exchanged.

In obtaining complementary function it is necessary to invert pass signals and gate signals also, since we exchanged PMOS and NMOS devices. Proof of duality principle is similar to that in CPL.

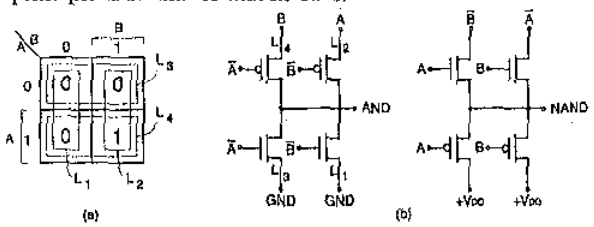

Fig. 6. Two-input DPL logic: (a) AND function Karnaugh map, (b) AND/NAND circuit

The synthesis method is illustrated on the example of two-input AND function, Fig. 6. Cube L_1 of Fig. 6*a* is represented with NMOS transistor, with the source connected to *GND* and the gate connected to $\overline{A}$. PMOS branch of L_2 passes B, when gate signal A is low etc. Complementary circuit (NAND), Fig. 6*b*, is generated from AND, according to complementarity principle.

According to duality principle OR circuit is formed from AND circuit, its dual counterpart, Fig. 7.

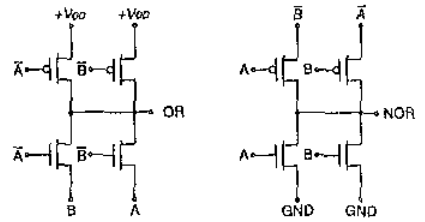

Fig. 7. Two-input DPL logic: OR/NOR circuit

Different two-input XOR/XNOR circuit arrangements are shown in Fig. 8 and Fig. 9. Mapping from Fig. 8 results

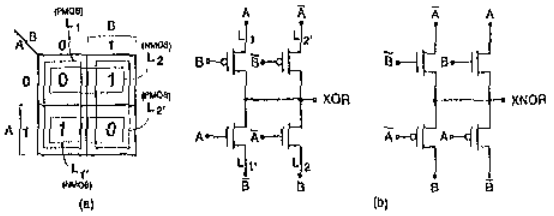

Fig. 8. Two-input DPL logic: (a) Mapping strategy I, *(b) XOR/XNOR circuit I*

in balanced load on both true and complementary input signals. But, it is also possible to cover map as shown in Fig. 9.

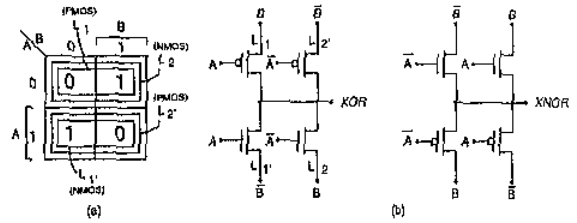

Fig. 9. Two-input DPL logic: (a) Mapping strategy II, (b) XOR/XNOR circuit II

This mapping strategy results in different load on input signals A and B, which could be advantageous in case when switching probability of signals A and B is different.

Three-input functions in DPL are implemented as cascaded combinations of two-input DPL modules.

C. DVL

Random logic function in DVL can be synthesized from Karnaugh-map according to the algorithm given below:

1. Cover all input vectors, which produce 0 at the output, with largest possible cubes (overlapping allowed) and represent those cubes with NMOS devices, which sources are connected to *GND*
2. Repeat step 1 for input vectors, which produce 1 at the output, and represent those cubes with PMOS devices, which sources are connected to V_{DD}
3. Finish with mapping input vectors, not mapped in steps 1 and 2 (overlapping with cubes from steps 1 and 2 allowed), which produce 0 or 1 at the output. Represent this cubes with parallel NMOS (good pull-down) and PMOS (good pull-up) branches, which sources are connected to one of the input signals

Mapping strategy and circuit implementation in DVL is shown on three-input AND example, Fig. 10.

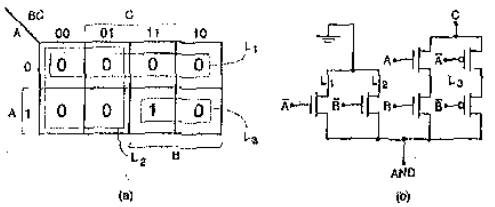

Fig. 10. Three-input DVL logic: (a) AND function Karnaugh map, (b) AND circuit

From Fig. 10 we see that by overlapping cubes L_1 and L_2 we save the area, which would be otherwise wasted on additional transistor in L_1 or L_2. That allows wider transistors of cube L_3. OR/NOR circuit, which is directly generated from AND circuit, is shown in Fig. 11.

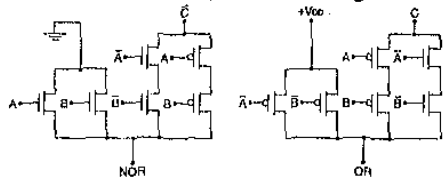

Fig. 11. Three-input DVL logic: OR/NOR circuit

Complementarity and duality principles in DVL are the same as in DPL (both are CMOS structures).

IV. Conclusion

General rules for synthesis of pass-transistor gates in three representative conventional pass-transistor techniques were established in this paper. From those rules an algorithmic way for generation of various circuit topologies (complementary and dual circuits), is established. This lays the foundation for development of computer aided design (CAD) tools capable of generating fast and power-efficient pass-transistor logic.

References

[1] T.Kuroda and T.Sakurai, "Overview of Low-Power ULSI Circuit Techniques," *IEICE Transactions on Electronics*, E78-C, no.4, pp.334-44, April 1995. INVITED PAPER, Special Issue on Low-Voltage Low-Power Integrated Circuits.

[2] K.Yano, Y.Sasaki, K.Rikino, and K.Seki, "Top-down pass-transistor logic design," *IEEE Journal of Solid-State Circuits*, vol.31, no.6, pp.792-803, June 1996.

[3] J.H.Pasternak, A.S.Shubat, and C.A.T.Salama, "CMOS differential pass-transistor logic design," *IEEE Journal of Solid-State Circuits*, vol.SC 22, no.2, pp.216-22, April 1987.

[4] J.H.Pasternak and C.A.Salama, "Design of Submicrometer CMOS Differential Pass-Transistor Logic Circuits", *IEEE Journal of Solid-State Circuits*, vol.26, no.9, pp.1249-58, Sept. 1991.

[5] I.S.Abu-Khater, A.Bellaouar, and M.I.Elmashry, "Circuit techniques for CMOS low-power high-performance multipliers," *IEEE Journal of Solid-State Circuits*, vol.31, no.10, pp.1535-46, Oct. 1996.

[6] P.Buch, A.Narayan, A.R.Newton, and A.Sangiovanni-Vincentelli, "Logic Synthesis for Large Pass Transistor Circuits," *Proceeding of the IEEE International Conference on Computer-Aided Design (ICCAD)*, pp.633-670, Nov. 1997.

[7] K.Yano et al., "A 3.8 ns CMOS 16x16-b Multiplier Using Complementary Pass-Transistor Logic," *IEEE Journal of Solid-State Circuits*, vol. 25, no.2, pp.388-95, April 1990.

[8] M.Suzuki, N.Ohkubo, T.Shinbo, T.Yamanaka, A.Shimizu, K.Sazaki, and Y.Nakagome, "A 1.5ns CMOS 16x16 Multiplier Using Complementary Pass-Transistor Logic," *IEEE Journal of Solid-State Circuits*, vol.28, no.11, pp.599-602, Nov. 1993.

[9] V.G.Oklobdžija and B.Duschene, "Pass-Transistor Dual Value Logic for Low-Power CMOS," *Proceedings of the 1995 International Symposium on VLSI Technology*, Taipei, Taiwan, May-June, 1995.

[10] V.G.Oklobdžija and B.Duchene, "Synthesis of high-speed pass-transistor logic," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol.44, no.11, pp. 974-6, Nov.1997.

[11] P.Y.K.Cheung et al., "High Speed Arithmetic Design Using CPL and DPL Logic," *Proceedings of the 23rd European Solid-State Circuits Conference*, pp.360-3, Sept. 1997.

[12] R.Zimmermann, W.Fichtner, "Low-Power Logic Styles: CMOS Versus Pass-Transistor Logic," *IEEE Journal of Solid-State Circuits*, vol.32, no.7, pp.1079-90, July 1997.