Comparison between nMOS Pass Transistor logic style vs. CMOS Complementary Cells*

Rakesh Mehrotra, Massoud Pedram
Dept. of E.E.-Systems
University of Southern California
Los Angeles, CA 90089, USA

Xunwei Wu
Dept. of Electronic Eng.
Hangzhou University
Hangzhou, Zhejiang 310028, China

Abstract

This paper compares three different logic styles for implementing arbitrary Boolean functions of upto three inputs in terms of their layout area, delay and power dissipation. The three styles are nMOS pass transistor based design, NAND gate based design, and CMOS complementary logic design. Results of the comparison show that pass transistor based design is superior to NAND based design, but loses to CMOS complementary logic design.

I. Introduction

Delay and power dissipation have emerged as the major concerns of designers. The gate delay depends on the capacitive load of the gate. The dominant term in power dissipation of CMOS circuits is the power required to charge or discharge the capacitance in the circuit. Thus by reducing capacitance we can decrease the circuit delay and power dissipation. Capacitance is in turn a function of logic cells being used in the design.

There is currently increased interest in nMOS pass transistor based cells because they appear to reduce the capacitance compared to their static CMOS counterparts. There are indeed recent reports [1] based on a full adder circuit comparison which show that nMOS pass transistor logic to be more efficient than CMOS complementary logic. However some key questions remain unanswered. One of these questions is how the nMOS pass transistor based cell compares to the NAND-based cell or the CMOS complementary cell (also referred to custom CMOS in the sequel) in terms of its performance characteristics.

In this paper we use the pass transistor based cell Y1 in [1] to construct a basic three input cell: NMUX2 (a multiplexer followed by an inverter). Next we implement and compare 12 PNN-complete three-input functions which cover all possible boolean functions of less than three-inputs by doing the physical layout using the NMUX2 cell, using the three input NAND gate, and finally using CMOS complementary logic gate.

Our results show that using the custom CMOS cells yields the best performance characteristics in terms of area, delayand power dissipation. The NMUX2-based design ranks second whereas the NAND3 based design is a distant third. This conclusion should however be viewed in light of the fact that the current technology mapping methods (e.g.,[7]) perform best when mapping is done to CMOS cells

However there is room for improvement by using BDD based techniques to directly synthesize pass transistor based logic. For examples see [5, 6].

This paper is organized as follows. In the next section we propose a basic three-input pass transistor cell NMUX and compare it with a three-input NAND gate. In section III we provide a comprehensive comparison between the three different logic styles: NMUX based logic design, NAND gate based logic design and CMOS complementary logic based design. Section IV describes our conclusions.

II. Pass Transistor NMUX cell vs. CMOS NAND Gate

A cell library is proposed in [1], which includes four CMOS inverters with different driving capabilities and three pass-transistor based cells, Y1, Y2, Y3 as shown in Fig.1(a). The output inverters marked by a large dot in these cells are composed of five or three MOS transistors as shown in Fig.1(b). It is seen that a feedback inverter and a pull up pMOS transistor, both consisting of minimum-size MOSFET's are included in the left configuration to avoid DC leakage current in the CMOS inverter. If the configuration with five transistors is used, the timing of the feed back signal is stable no matter how large the load capacitance is. However, when the design process is such that the designer can ensure that the load capacitance of a cell remains within an allowable range, the right configuration in Fig.1(b) with three transistors may be used.

We chose cell Y1 in Fig.1(a) and a conventional CMOS cell of similar size for comparison. If we use conventional NAND gates in the design, examination of real designs indicates that the average fanin count of a NAND gate is about 2.7 [2]. Thus, we chose the three-input NAND gate shown in Fig.2(a) for the purpose of comparison. Since the candidate realization using pass transistor based cells should have the same number of transistors as that of a three input NAND cells, we compose cell Y1 with the three transistor inverter in Fig.1(b) and use a standard inverter to generate the complement control signal $\overline{C}$ as shown in Fig.2(b).

Thus, both cells in Fig.2 have three-input terminals and nearly the same number of transistors. (Cell in Fig.2(b) has one extra minimum size pull up pMOS transistor). The cell in Fig.2(b) can be named NMUX

* This work was supported in part by DARPA under contract #F33615-95-C-1627.

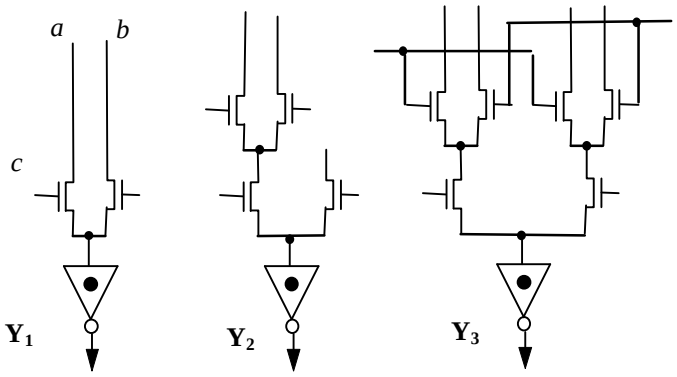
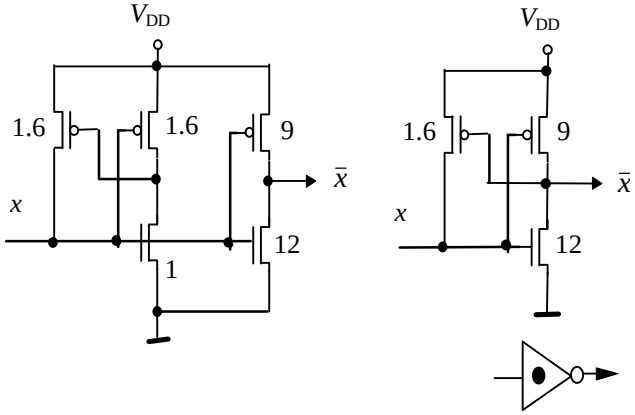


Fig.1(a) Pass Transistor Based Cell



(b) output inverter

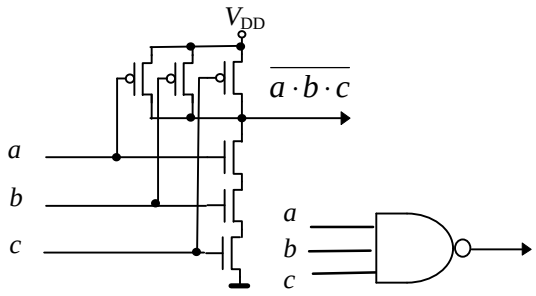
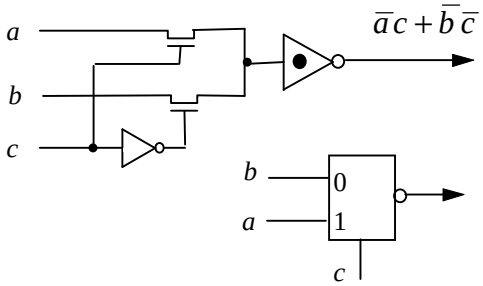


Fig.2(a) Three input NAND gate



(b) Three input pass-transistor based NMUX cell

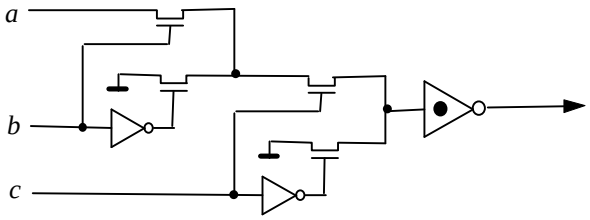
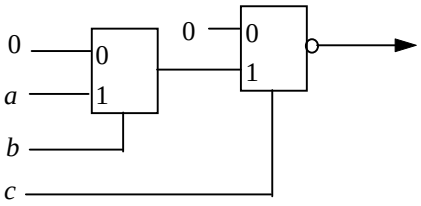


Fig.3(a) Pass-transistor based three input NAND gate



(b) logic structure with MUX

because it is a multiplexer (MUX gate) followed by an inverter (NOT gate).

Comparison between two basic cells (which have equal number of input terminals) may include the following:

1. Silicon area.
2. number of cells required to realize any function of *k* variables
3. signal propagation delay through the cell

4. power dissipation of each cell for a typical input vector sequence.

Criteria (1) and (2) determine chip area whereas criteria (3) and (4) affect the circuit timing and power dissipation.

Table 1. Comparison between CMOS NAND and Pass Transistor Based NAND

Three-input NAND gate	Fig.2(a): Conventional	Fig.3(a): Pass-transistor-based
Tr. Count	6 (1)	13 (2.17)
Area	329μm^2 (1)	579μm^2 (1.75)
Delay	295ps (1)	465ps (1.58)
Power	0.91μW/MHz (1)	0.96μW/MHz (1.05)

In [1], Y_2 cell and two inverters are used to form a pass-transistor based three-input NAND gate, as shown in Fig.3(a). If we use the MUX legend to express the circuit, we obtain its corresponding logic structure as shown in Fig.3(b). Comparison between the pass-transistor based three-input NAND gate and the conventional CMOS NAND gate is given in Table 1. The area, delay and power are evaluated based on a 0.6 *μm* technology. The gate width of the output inverters in Y_2 cell is shown in Fig.1(b). The load capacitance is set to 100 *fF* for the purpose of delay calculation. This table shows that the pass-transistor based NAND has more transistors (2.17 times), larger area (1.75 times), and higher delay (1.58 times). However, its power dissipation is better than expected (only 1.05 times worse).

In fact, we should compare the three-input CMOS NAND gate of Fig.2(a) with the three-input pass-transistor based NMUX cell of Fig.2(b). This is because these two gates are the building blocks of the circuits designed using NAND-based logic and pass transistor based logic. It should be clear that the area and propagation delay of the cells in Fig.2(a) and 2(b) are nearly the same, however, the latter cells should have lower power dissipation. This is because Fig.3(a) and Fig.2(a) have comparable power dissipation, but Fig.2(b) clearly has lower power dissipation than Fig.3(a). In the following discussion, however, we will be conservative and assume that both cells in Fig.2(a) and Fig.2(b) have comparable area, delay and power dissipation.

Fig.3 and Table 1 show that if we use the conventional CMOS NAND gate and pass-transistor based NMUX cell to realize $\overline{a \cdot b \cdot c}$, the CMOS NAND gate is much better. However, this example cannot be used to prove that the CMOS NAND gate is better in designing other circuits. For example, consider the three-input pass-transistor based NMUX cell shown in Fig.2(b). If we use the three-input NAND gate and inverter to realize the same function we will need at least three NAND gates and two inverters [2]. Consequently, the NAND and NMUX cells have to be compared by implementing a large number of functions and averaging results over these functions as shown next and in Table 5. Consider all functions of *n* input. Because of the rapid increase in the number of functions as *n* rises, we confine our investigation herewith to $n \le 3$. This has the further merit that detailed statistics are already available on the realization of $n \le 3$ functions using three-input NAND gates (and three-input NOR gates) [2]. Here we report and compare the statistics for the three-input pass-transistor based cell to the NAND results.

For convenience, the 256 functions of $n \le 3$ variables are classified into 80 representative *Permute(P)-complete* functions; the 256 functions are obtained from these 80 functions by input permutations. Furthermore, we use the decimal number function identification system of [3] to take the output vector of the function truth table and re-express it in decimal notation. For example, the above function $\overline{a} \cdot \overline{c} + \overline{b} \cdot c$ has an output vector of 00100111 (where the output value of minterms $\overline{a} \cdot \overline{b} \cdot \overline{c}$ through $a \cdot b \cdot c$ are read from right to left), which in decimal notation becomes function 039. An example of a *P-complete* transformation would be the transformation of the above function $\overline{a} \cdot \overline{c} + \overline{b} \cdot c$ into function $\overline{b} \cdot \overline{c} + \overline{a} \cdot c$ by permuting *a and b.* The *80 P-complete* functions can be further classified into *22 PN (Permute-Negate)* equivalent functions [3] Every function of a given *PN* class can be transformed into another function in the same class by input permutation and/or input negation. There are *22 PN-complete functions* for $n \le 3$ functions. As an example function $\overline{a} \cdot \overline{c} + \overline{b} \cdot c$ can be transformed into $a \cdot \overline{c} + b \cdot c$ by negating inputs *a* and *b*

The 22 *PN-complete* functions, can be further classified into 14 representative *Permute-Negate-Negate (PNN) complete* functions. Every function of a given class can be transformed to other functions in the same class by input-negation, input permutation and output negation. For example the above function $\overline{a} \cdot \overline{c} + \overline{b} \cdot c$ can be transformed into function $\overline{a} \cdot \overline{b} + \overline{a} \cdot \overline{c} + \overline{b} \cdot c$ by negating inputs *a* and *b* and by negating the output. The number of functions in each class is shown in Table 4. (The trivial classes where the functions are either a constant or single variable functions are shown with a *.)

Minimum realizations by using inverters and NMUX gates for the 12 non-trivial *PNN-complete* functions (one from each class) are tabulated in Table 2. Data corresponding to NAND realization (taken from [2]) are also listed in the table for comparison). For each circuit we report the number of inverters and NMUXs used (n_{inv}, n_{nmux}), the number of input connections (n_c) for all devices used, and the number of device cascade levels (n_l). We also report the corresponding numbers given in [2] when the inverter and three-input NAND gate are used instead. From Table 2, it can be seen that if we use the NMUX2 and inverter in design, we will save an average of 38% cells and 73% inverters. There will also be a 55% reduction in the number of connections and a 28% reduction in the number of cascade levels. Obviously, area, delay and power dissipation of these circuits will significantly be improved since both 3-input pass-transistor based cell and CMOS NAND gate have approximately the same area, delay, and power dissipation.

We take the full adder as a practical design example to compare both cells. The sum and the carry-out outputs are functions (150) and (232) respectively:

$$S = \overline{a} \cdot (b \oplus c) + a \cdot \overline{b \oplus c},$$

$$C_{\text{out}} = a \cdot (b \oplus c) + b \cdot \overline{b \oplus c}.$$

The corresponding design with NMUX2 is shown in Fig.4(a), where only three cells and two inverters are used. However, the S output alone needs six 3-input NAND gates, as shown in Fig.4(b).

III. Pass Transistor Based NMUX cell vs. CMOS Complementary Logic cell

Although pass transistor based design yields better results compared to NAND gate based design, the comparison is not that useful as logic design is almost never done by using NAND gates exclusively. A realistic comparison would be between logic design done using a standard cell library (CMOS complementary cells) and pass transistor based design . Thus we have taken one function from each of the 12 PNN non-trivial classes and implemented it using Y1 cells and custom CMOS cells. Here, we assume that the standard cell library contains at least one function of each PNN-complete class. We use 0.6um technology for laying out all the 12 functions. We use the MAGIC layout editor and use the HP-CMOS14B process parameters for extraction. The HP-CMOS14B Level-39 FET models were used for HSPICE [4] simulations. The load capacitance for each of the circuits was set to 100 *fF* for circuit delay characterization. Worst case delay measurements were made for each function implemented using Y1-cells, NAND gates and custom CMOS cells by analyzing the cell structure and finding the slowest input to output path change. We used the same input vectors to calculate the power dissipation for each design. A software program was used to calculate the energy used by each circuit for a time period of 100 *ns* during which time the output switched exactly twice between different voltage levels. The program calculates a discrete version of the *V.I.dt* integral when given the supply current at intervals of 1ns for 100 *ns*. The results are shown in Table 5.

IV. DISCUSSION

The tabulated results shown above indicate that layout using CMOS complementary cells (column 3) yields the best performance characteristics in terms of delay, power dissipation and area. This characterization was done to set a reference point which pass transistor logic would have to improve upon to be viable for use in the future. If the cells were only mapped to NAND cells, then the Y1-cell pass transistor implementation would be superior for the 12 PNN - complete functions. This is however not a common practice.

The purpose of this paper was to investigate the viability of Y1-cell pass transistor logic to synthesize circuits as opposed to 3-input NAND gates and custom CMOS cells. From the above table, it can be seen that the Y1-cell implementation yields lower areas for a given delay when compared to NAND gate implementation. However Y1-cell based implementations compared with custom layout implementations tend to be inferior. It can be seen that the Y1-cell implementation yields circuits of smaller areas than the NAND gate implementation in 10 out of the 12 cases, but when compared to custom layout, they are larger in 12 out of 12 cases. As for energy consumption under the same input vector sequence, the Y1-cell implementation yields better results in only 5 cases when compared to NAND cells and compared to custom CMOS cells Thus in terms of power dissipation the results are somewhat uncertain.

In summary, the Y1-cell implementation seems to yield better results for delay and area measurements when compared with NAND gate based implementation but is inferior to CMOS complementary cells. Our conclusion is however based on the premise that the same synthesis and mapping technology is used for both standard cell based design and pass transistor based designs. If the synthesis and mapping scripts are changed it is possible that pass transistor logic style would be a better choice than standard cell logic. Our paper does neither support nor reject this possibility. Furthermore we assumed that the ASIC library is PNN complete with respect to all 2 and three input functions.

REFERENCES

[1] K.Yano, Y. Sasaki, K.Rikino and K. Seki, "Top-down pass-transistor logic design," *IEEE J. Solid-state Circuits*, Vol.SC-31, pp.792-803, 1996.

[2] L.Hellerman, "A catalogue of three-variable OR-Invert and AND-Invert logic circuits," *IEEE Trans. Electron Computers*, Vol.E-12, pp198-223, 1963.

[3] G. J. Klir, *Introduction to the Methodology of Switching Circuits*, D. Van Nostrand Company, 1972.

[4] HSPICE refrence manuals by Meta Software.

[5] P. Buch, A. Narayan, A. R.Newton, A. Sangiovanni-Vincentelli, "On synthesizing pass-transistor networks", *International Workshop on Logic Synthesis*, 1997.

[6] V.Bertacco, S.Minato, P.Verplaetse, L.Benini, G. De icheli, "Decision diagrams and pass transistor logic synthesis", *International Workshop on Logic Synthesis* ,1997.

[7] K.Keutzer, "Technology mapping and local optimization", Proc. of the 24th Design Automation Conference, pp 341-347, 1987.

Table 2:Fourteen representative function for n<4

Function (decimal)	Functional expression	NMUX and inverter				NAND and inverter			
		n_{inv}	n_{nmux}	n_c	n_l	n_{inv}	n_{nand}	n_c	n_l
(127)	$(\overline{a} + \overline{b}) + \overline{c}$	0	2	4	2	0	1	3	1
(126)	$\overline{a} \cdot \overline{\overline{b} \cdot \overline{c}} + a \cdot \overline{b \cdot c}$	1	3	8	3	0	5	12	3
(111)	$\overline{a} + (b \oplus c)$	1	2	6	3	0	4	9	3
(063)	$\overline{a} + \overline{b}$	0	1	2	1	0	1	2	1
(151)	$\overline{a} \cdot (\overline{b \cdot c}) + a \cdot \overline{b \oplus c}$	1	3	9	3	0	6	14	3
(152)	$\overline{b} \cdot \overline{\overline{a} + c} + b \cdot c$	1	2	6	2	2	3	9	3
(047)	$\overline{a} + \overline{b + \overline{c}}$	0	2	4	2	1	2	5	3
(232)	$a \cdot (b \oplus c) + b \cdot \overline{b \oplus c}$	2	2	8	3	0	4	9	2
(202)	$\overline{\overline{a} \cdot \overline{c} + a \cdot \overline{b}}$	1	1	4	2	1	3	7	3
(120)	$\overline{\overline{c \cdot b} \oplus a}$	1	2	6	2	0	4	10	3
(105)	$\overline{a \oplus (b \oplus c)}$	2	2	8	3	0	7	16	5
(153)	$b \cdot c + \overline{b} \cdot \overline{c} = \overline{b \oplus c}$	1	1	4	2	2	3	8	3

Table 3: NMUX vs. NAND statistics for n = 2,3 functions

Type of circuit cell	3-input NMUX	3-input NAND
Number of input terminals per cell	3 (1)	3 (1)
Total number of devices required for cell functions	593 (0.53)	1115 (1)
Average inverter count per function	0.51 (0.36)	1.40 (1)
Average cell count per function	1.80 (0.58)	3.09 (1)
Average input connection count per function	5.03 (0.44)	11.40 (1)
Average cascade level count per function	2.15 (0.70)	3.08 (1)

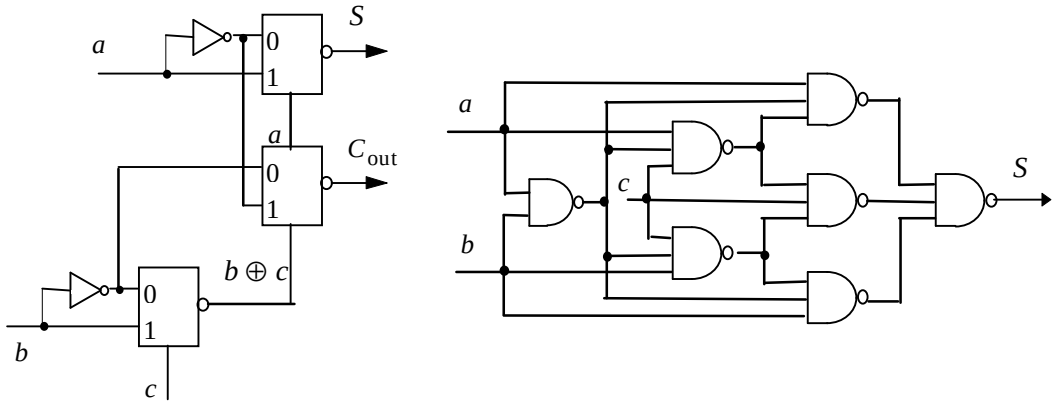


Fig.4(a) Full Adder by using NMUX2 (b) Sum realizations by using three input NAND gate

Table 4: Classification of n<4 functions into 14 PNN classes

Class	I	II	III	IV	V	VI	VII	VIII	IX	X	XI	XII	XIII	XIV
Function in (Decimal)	0 *	127	126	111	063	151	152	047	240 *	232	202	120	105	153
Number of functions	2	16	8	24	24	16	48	48	6	8	24	24	2	6

Table 5: Detailed Comparison			
FUNCTION (Decimal) AREA(um^2) DELAY(ns) ENERGY(pJ)	TWO && THREE INPUT NAND GATE	CMOS COMPLEMENTARY LOGIC	PASS TRANSISTOR LOGIC
127: AREA: DELAY: ENERGY:	 521 0.79 0.499	 147.6 0.81 0.3859	 605.52 0.87 4.63
126 AREA: DELAY: ENERGY:	 2493.18 0.5 0.09506	 627.48 0.661 0.75839	 1119.4 0.63 0.701
111 AREA: DELAY: ENERGY:	 1104.48 0.92 0.787	 421.2 1.09 0.7312	 605.52 0.90 0.748
063 AREA: DELAY: ENERGY:	 490 0.71 0.775	 120.04 0.85 0.9582	 401.94 0.88 0.713
151 AREA: DELAY: ENERGY:	 1552.8 1.07 0.812	 900.9 1.28 6.142	 1122.8 0.99 0.996
152 AREA: DELAY: ENERGY:	 843.07 0.981 0.8133	 772.92 0.981 0.30282	 822.96 0.92 9.66
047 AREA DELAY ENERGY:	 440 0.78 0.380	 110.06 0.78 0.4887	 580.74 0.80 0.951
232 AREA: DELAY: ENERGY:	 1202.67 1.01 0.92639	 1188 1.30 0.733	 1383.5 1.25 0.869
202 AREA: DELAY: ENERGY:	 1038.33 1.11 0.6865	 460 0.61 0.477	 822.96 1.19 10.1
120 AREA: DELAY: ENERGY:	 1083.15 1.02 0.8661	 392.04 0.82 0.8182	 610.74 0.94 0.757
105 AREA: DELAY: ENERGY:	 1410 0.67 0.91318	 1039.5 0.71 1.0007	 1139.3 0.62 0.704
153 AREA: DELAY: ENERGY:	 907 1.05 0.61671	 370.16 1.07 0.62747	 527.22 1.10 1.04