NPCPL : Normal Process Complementary Pass Transistor Logic for Low Latency, High Throughput Designs

Debabrata Ghosh S.K. Nandy K. Parthasarathy V.Visvanathan
Dept. of EE Dept. of ECE Dept. of EE Dept. of ECE
Indian Institute of Science, Bangalore 560 012, India.

Abstract

High throughput and low latency designs are required in modern high performance systems, especially for signal processing applications. Existing logic families cannot provide both of them simultaneously. We propose a Normal Process Complementary Pass Transistor Logic (NPCPL) which can be used as a universal logic to provide finest grain pipelining without affecting overall latency or increasing the area. It does not require any special process steps and hence, can be realised in a normal process technology as against the CPL proposed by Yano et al *[2] which uses threshold voltage adjustment of selected devices. The design procedure is described for (a)low latency, (b)high throughput and (c)low area requirements. In addition to the various advantages, it is envisioned that NPCPL designs can also be used to build ultra-high speed pipelined system without pipelining latches, viz., wave pipelined digital systems, where the throughput achievable is beyond that permitted by the delay of a pipeline stage.*

1 Introduction

High performance systems and a variety of real-time Digital Signal Processing systems derive their performance from VLSI solutions. Since fast arithmetic units are critical to all such high-performance applications, we focus attention to a logic family that realises adders and multipliers for the range of latency and throughput requirements appropriate for a particular DSP application. Addressing this issue, Yano *et al* [2] have offered CPL as a high-speed logic family to realise high-peformance arithmetic units. Their approach, however, is constrained by the requirement of a specialised process, which in general may not always be easily accessible to a common designer. In this paper we take the cue from CPL and extend the logic family to support a variety of arithmetic and logic units which can be realised in a normal process. One novel feature of our approach is that the proposed logic family can be exploited to support designs with both low latency and high throughput simultaneously. Pipelining can be introduced to the finest grain without any significant area and latency overhead. This is in contrast to the conventional approach to design of high throughput systems where latency and area are traded off for high throughput and vice versa.

The rest of the paper is organised as follows. Section 2 gives an overview of CPL, section 3 explains the Normal Process Complementary Pass Transistor Logic and section 4 compares it with other logic. The paper is concluded in section 5.

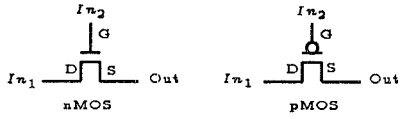

Figure 1: Two Types of Pass Transistors

2 Complementary Pass Transistor Logic (CPL)

NMOS pass transistor logic offers advantages of all the three performance metrics of VLSI, *viz.*, area, speed and power dissipation. In a pass transistor network, an input x_i is steered through a chain of (n-1) pass transistor under the control of inputs $x_1, ..., x_{i-1}, x_{i+1}, ..., x_n$ to perform an n input function $f(x_1, x_2, ..., x_n)$. This, along with low gate capacitances, reduces the delay. However, in pass transistor logic, degraded voltage level, and hence, reduced noise margin is prohibitive. Pasternak *et al* [3] and Jayasumana *et al* [12] have reported the use of pass transistor logic and attempted to solve the problems of degraded voltage level and noise margin. But none of these two methods is efficient and can be used extensively in general.

In pass transistor logic, the basic building blocks are nMOS and pMOS transistors as in Figure 1. An nMOS(pMOS) transistor is a four-terminal device with terminals source, drain, gate and bulk. An input In_1 at the drain is steered to the source by the input In_2 at the gate. In nMOS(pMOS), the source and the drain potentials V_S and V_D respectively are related by $V_D \geq V_S$ ($V_D \leq V_S$). In nMOS(pMOS), when In_1 and In_2 are at logic '1'(logic '0'), the logic level of the output, Out, at the source terminal is degraded to $V_{DD} - V_{Th_n}$ ($\mid V_{Thp} \mid$), where V_{Th_n} (V_{Th_p}) is the threshold voltage of the body-effected nMOS (pMOS) transistor.

K. Yano *et al* [2] describe an excellent method of exploiting the advantages of pass transistors and surmounting the associated problems. Their methodol-

0-8186-3180-5/92 $3.00

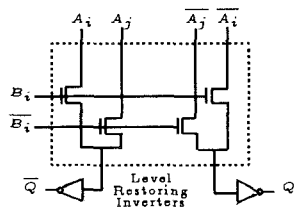

Figure 2: Basic Building Blocks

Function	A_i	A_j	B_i	O/P Q	O/P $\overline{Q}$	Delay
AND/NAND	A	V_{SS}	B	A.B	$\overline{A.B}$	0.6 ns
OR/NOR	A	V_{DD}	$\overline{B}$	A+B	$\overline{A+B}$	0.6 ns
XOR/XNOR	A	$\overline{A}$	$\overline{B}$	$A \oplus B$	$\overline{A \oplus B}$	0.6 ns

Table 1: Realising Different Boolean Functions with the Basic Cell

ogy allows the pass transistor logic to be used universally for the entire data-path with the help of some static CMOS inverters only. This logic family is called Complementary Pass Transistor Logic (CPL). Generically, CPL consists of
1. True and complementary pass variables
2. True and complementary control variables
3. nMOS pass transistor network
4. CMOS level shifting inverters
CPL performs the logic function using nMOS pass transistors only, and the degraded logic '1' is restored by static CMOS inverters.

CPL offers several advantages. The threshold modified CPL reported in [2] has delays 2-2.5 times lower than fully complementary CMOS (FCC). The power dissipation is approximately 30% lower than that of FCC. However, CPL requires threshold voltage adjustment of the devices which is the key to CPL design. Adjustment of threshold voltage for selected devices and maintaining an accurate threshold voltage requires specialised fabrication processes. In general, for a common designer it is difficult to provide easy access to such a process. Hence Fang Lu and H. Samueli[11] contend that CPL may not be useful for general purpose design. They propose a complex solution by way of an adaptively biased pseudo-nMOS logic (APNL).

In the following we demonstrate that CPL can indeed be used under normal process conditions, *i.e.*, without threshold adjustment. We call this logic family NPCPL. Compared to CPL, NPCPL has a degradation of performance (in terms of speed and noise margin), yet NPCPL outperforms any of the other logic families. As we will elucidate later, NPCPL is best suited for both low latency and high throughput applications.

3 NPCPL : Normal Process CPL

In CPL, a basic building block is a two-input pass transistor logic block which can be configured as AND/NAND, OR/NOR and XOR/XNOR modules. When such modules are combined to form arbitrary boolean functions, data lines are loaded heavily. While we retain the basic topology of CPL building blocks in NPCPL, we address the issue of loaded data lines. Figure 2 gives a schematic of such a module, and the various configurations are given in Table 1. (All delay figures in this table are for a 1.6μ, n-well, double metal, C3TU, ES2 process).

Note that Table 1 has V_{DD} and V_{SS} as entries in the AND/NAND and OR/NOR configuration. These modifications to CPL configurations relieve the load on data line B as illustrated in Table 1 and in turn enhances the speed of operation of a NPCPL building block. The degraded logic '1' level at the output of the nMOS pass transistor block is restored by specially designed static CMOS inverters.

The logic threshold voltage of the inverter is given by the expression [1]

$$V_{TH} = \frac{\sqrt{K}(V_{DD} - \mid V_{Th_p} \mid) + V_{Th_n}}{1 + \sqrt{K}} \qquad (1)$$

$$\text{where } K = \beta_p/\beta_n = (W_p\mu_p)/(W_n\mu_n).$$

V_{TH} can be controlled by adjusting the β_n/β_p ratio (or the W_n/W_p) ratio of the devices in the inverters. Since the voltage swing at the output of the nMOS is from 0 to $(V_{DD} - V_{Th_n})$, we set the logic threshold V_{TH} of the inverter at $\frac{1}{2}(V_{DD} - V_{Th_n})$ to ensure equal noise margin for low-to-high and high-to-low transitions. This is the key in NPCPL design. Without this adjustment, the noise margin high, NM_H is severely degraded and may be reduced even to zero.

In CPL, threshold adjustment was necessary to handle degradation of voltage levels which affected the static power dissipation, speed of operation and noise margin. In the following we show how NPCPL can be used for high throughput and low latency applications while doing away with threshold voltage adjustment of selected devices. We give a design methodology by which NPCPL design can be steered to function optimally to meet the design objectives of
1. Low Latency
2. High Throughput
3. Low Area
We revisit the issues of static power disipation and noise margin in the context of NPCPL design for low latency and high throughput application in the subsequent sections. The issues of speed of operation, noise margin, static power dissipation etc. are related to the degraded voltage level for a logic '1' at the output of the nMOS pass transistor block. The proposed design procedures for NPCPL centres around preventing this voltage level from degrading any further beyond $(V_{DD} - V_{Th_n})$.

The optimisation, however, is process sensitive. For the rest of the paper all optimisation steps discussed are specific to the process we use (where

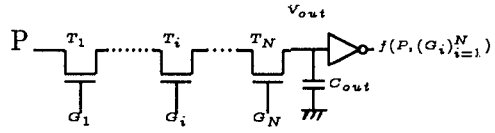

Figure 3: N Pass Transistors in the Critical Path of a Circuit

$V_{Th_n(0)} < \mid V_{Th_p(0)} \mid$). In particular, a process with $V_{Th_n(0)} < \mid V_{Th_p(0)} \mid$ is useful for decreasing static power dissipation as illustrated later. However, our approach is general enough to be exploited in any process.

3.1 NPCPL for Low Latency Application

In a pass transistor configuration the delay increase is a quadratic function of the the number of series pass transistors[13]. Hence, the logic should be partitioned into smaller blocks and the voltage level restored at intermediate points by inverters. We have experimentally found a critical path consisting of two pass transistors followed by an inverter to be optimal for each building block. This restricts the number of input variables for each block to three. This is sufficient to span a class of arithmetic functions. For example, in arithmetic circuits the basic building block is the full adder and it is a three input logic function.

A general pass transistor network consisting of N nMOS transistors (followed by the level restoring inverter) in the critical path is shown in Figure 3.

The voltage level of a logic '1' at the output of the nMOS pass block is given by

$$V_{out} = V_{DD} - V_{Th_n}. \quad (2)$$

Cascading many pass transistors in series causes the delay to increase drastically. On the other hand, if we restrict to just one pass transistor, the overhead of the level restoring inverter may offset the advantages anticipated. Through simulation, we determined the optimal number of pass transistors in series chain as that of two. The size of the pass transistors is to be determined through proper simulation since increasing the width of the transistors decreases the resistance but increases the capacitance. The performance is layout-specific too.

A degraded voltage level implies higher delay. It is because the output voltage reaches $(V_{DD} - V_{Th_n})$ only asymptotically, the n-transistor in the output inverter would not be fully turned 'ON' (may enter linear region instead of being in saturation). It then has to be provided with more drive, *i.e.*, its width is to be increased compared to that of the p-transistor. (This is synonymous to shifting the threshold voltage of the inverter towards the left of the point $V_{DD}/2$ in the DC transfer characteristics of the output inverter fed with output of the pass network.) It is usual to have W_p larger than minimum feature size (unit size transistor) to account for low mobility of p-transistor. In addition, in NPCPL, it is necessary to maintain $W_n > W_p$. Consequently, gate capacitances increase, and hence, contribute to delay and power dissipation. The issues of delay and power dissipation are discussed in detail through the following sections. In summary, logic level degradation can be minimised through a design practice as enumerated below:

1. Restrict number of series pass transistor to two,
2. Use properly sized pass transistors,
3. Draw smallest diffusion lines while forming transistors,
4. Route gate signals in metal and change to polysilicon near the transistors.

Design practice (2) strikes a balance between internal capacitances to be charged and the resistances of the charging path, thereby optimising the delay. Design practices (3) and (4) help reducing the overall delay. The effect of parasitics is to degrade the voltage level at the output of the pass block, and hence, increase the time taken to reach the level $(V_{DD} - V_{Th_n})$. The above thumb–rules ensure that this effect is minimised.

3.1.1 Static Power Dissipation

Static power dissipation is present in NPCPL designs. Careful design can however render keeping the dissipation within tolerable limits. The logic level '1', at the output of a pass block (feeding the level restoring inverter) is degraded to 4 volts, resulting in a power dissipation of 5 μWatt per NPCPL building block and 10 μWatt per full adder in the standby mode. It therefore follows that an NPCPL design with a logic complexity of 10,000 two input logic gates (consisting of 80,000 transistors) will suffer a static power dissipation of 50 mW and that with 10,000 full adders (consisting of 2,80,000 transistors) will dissipate 100 mW. For good NPCPL designs, however, the overall power dissipation is low compared to FCC.

3.1.2 Noise Margin

The noise margin of the NPCPL is less than that of the FCC by $\frac{1}{2}V_{Th_n}$. However, price paid in terms of noise margin for the choice of a simple process (that does not require threshold adjustment) is an acceptable and affordable trade-off. The overall degradation in the noise margin is not disastrous. The methodology followed here ensures that the degradation does not lead to malfunctioning. A full adder (in C3TU), which is a complex three input NPCPL gate, clocked at 500 MHz exhibits a noise margin of 1V which is an acceptable figure. The effect of a number of series pass transistors is important in the context of noise margin. Too many pass transistors in series decrease the noise margin further. With proper design, noise margin is kept within tolerable limit.

3.2 NPCPL for High Throughput Application

High throughput applications derive their performance from pipelining. However the extent of pipelining is often restricted because in a highly pipelined system the area overhead of pipelining latches and their associated delays far exceeds the area and the latency

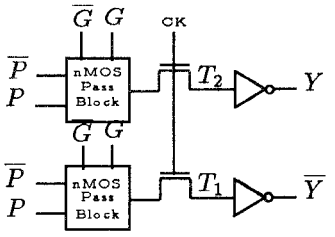

Figure 4: Pipelining the Basic Blocks : Fine–Grain Pipelining

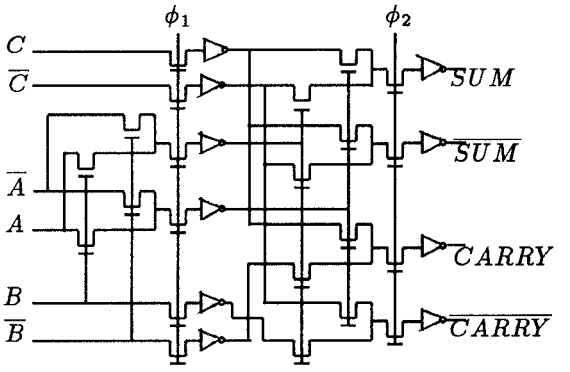

Figure 5: A Full Adder for High Throughput Application

of the unpipelined system. This has been further elucidated in the literature [14]. In NPCPL, it is possible to design each pipeline stage as elementary as a generic NPCPL building block. Thus NPCPL permits us to exploit fine grain pipelining at no extra overhead of area and latency due to the latches. The overheads of the latches are subsumed in the inverters as detailed below. In C3TU, 1.6μ process, it is possible, in principle, to have pipeline stage delay as low as 0.6ns and have an overall pipelined system operating at 800MHz using two–phase clocking.

When maximising the throughput is the main criterion, as in a typical signal processing environment, the logic depth has to be minimum. Hence, two input NPCPL logic blocks, with just one pass transistor followed by another clocked pass transistor in series feeding the level restoring inverter for the combinational block, are most suitable. As seen from Figure 4, two clocked pass transistors T_1, T_2 have been introduced between the two-input logic block and the inverter so that the transistor along with the level restoring inverter can serve as a dynamic latch as shown in Figure 4. Thus a conventional two-phase clocking scheme (pp. 207-209 of [1]) can be used to clock the pipelining stages.

Since the combinational pass block generates differential outputs f and $\overline{f}$, no additional inverters are necessary.

Observe that in this approach the additional overhead of pipelining is that of a single clocked transistor at the end of every pass block. In contrast, fine grain pipelining in other logic families are prone to both area and latency penalty close to 50% of their unpipelined counterparts [14].

Figure 5 gives the schematic of a full adder, which is the basic building block for all types of arithmetic circuits, for high-throughput applications. It has two stages each having a critical path of one pass transistor followed by a level restoring inverter. Introducing a clocked transistor before the inverter will render each stage into a pipeline stage.

3.2.1 Static Power Dissipation

In the high throughput NPCPL system, the number of transistors in the series pass chain is two – one for implementing the combinatorial function and the other for clocking. Static power dissipation is comparable to that of NPCPL design for low latency applications and follows from section 3.1.1.

3.2.2 Noise Margin

Two pass transistors in series provides enough noise margin and follows directly from the discussion in section 3.1.1 and 3.1.2.

3.3 Optimisation for Area

When the goal is to optimise area, pass transistors are to be used extensively. The functions are implemented using a complex pass network, using either a tabular method or the Karnaugh map, following the methodology of Damu Radhakrishnan *et al* [13]. At the end of this network the level restoring inverter is added. Given sufficient time, the pass network output reaches $(V_{DD} - V_{Th_n})$, which can then be restored to true logic values. This ensures low static power dissipation and adequate noise margin as illustrated before.

Generally NPCPL employs two pass blocks to generate differential outputs. However, for low area NPCPL designs, it is advisable to use a single pass block to generate complemented(true) output and invert it by the level–restoring inverter to get the true(complemented) output. Additional inverters are required for obtaining the complemented(true) outputs. A full adder which optimises area using the methodology stated above is shown in Figure 6. Note that it accepts differential inputs and generates differential outputs while using one pass block only for each of the sum and carry parts. The extra overhead of routing dual–rail signals is amortised by the low area requirement of a single pass block.

It takes an area of $63 \times 57\mu^2$, which is approximately 40% of that occupied by a high performance full adder in FCC and approximately 50% of that occupied by a low area full adder in FCC; yet it outperforms both the full adder circuits in FCC. It consumes $0.9mW$ of power at 100MHz which is less than 50% of the power consumed by the FCC counterpart. Its static

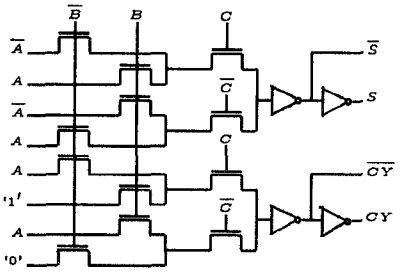

Figure 6: A Full Adder for Low Area and Low Power Dissipation

power dissipation is also very low – 1.7 μWatt. Note that the static power dissipation here is determined statistically, *viz.*, depending upon the probability of SUM and CARRY being '1'.

4 NPCPL vs Other Logic Families

NPCPL is an ideal logic family for high performance designs. It incorporates all the advantages of the DCVS family and yet is free from their drawbacks. Other high performance logic styles like DCVS family (including the Sample Set Differential Logic-SSDL[6], Enable-Disable CMOS Differential Logic-ECDL[7], Multiple Output Domino Logic-MODL[9], Latched CMOS Differential Logic-LCDL[8]), NORA, modified NORA[4], DSL[5] etc. have several disadvantages. These are
1. Charge sharing (common to all)
2. Static power dissipation (in DSL)
3. More dynamic power consumption (in NORA and sometimes in DCVS also)
4. Accidental discharge due to race (in modified NORA) etc. [4].
5. The Adaptively Biased Pseudo-nMOS Logic-APNL [11], is good for non-pipelined cases, but for pipelining it is not useful.
The DSL, uses short channel nMOS devices and hence, requires specialised process technology [11]. Its speed is achieved from this factor also. The pass transistor logic in NPCPL provides speedup from the inherent structure of logic realisation while keeping the area low. It is absolutely free from the charge sharing problem which is a potential hazard in all dynamic logics. Its dynamic power dissipation is also lower than those of the others. The methodology proposed above allows to use NPCPL under normal process condition, thereby eliminating the specialised process requirements of CPL [11]. Note that threshold–adjusted CPL[2] dissipates substantial static power due to subthreshold leakage current[10] which is insignificant in NPCPL. In summary, NPCPL offers the following advantages :

1. **Speed of Operation :** The NPCPL, to the best of our knowledge, is the fastest of all logic barring CPL. A full adder implemented in NPCPL for low latency applications has a worst-case delay of 1.2ns whereas the same takes 2.7ns in FCC (in the 1.6μ C3TU process). The simple yet effective method of voltage level restoration induces minimum delay and makes NPCPL faster than any other nMOS pass transistor logic implementation (excepting of course CPL). Use of minimum sized transistors as pass elements decreases gate capacitance. All these contribute to the higher speed of operation of NPCPL.

2. **Low Latency and High Throughput Operation :** NPCPL can be employed to exploit low latency and high throughput simultaneously. Pipelining can be introduced to the finest grain without any significant increase in overall latency. This, along with the low delay of the logic blocks makes NPCPL ideal for both low latency and high throughput applications simultaneously as against any other logic where a high throughput is beset with high latency.

3. **Wave Pipelining :** We can go beyond fine grain pipelining as discussed in section 3.2 if we exploit wave pipelining in NPCPL. Wave pipelining is a clock-free pipelining technique where the frequency of operation can be increased enormously. In wave pipelining, multiple coherent waves of data are sent through a combinational logic block by applying new inputs faster than the delay through the logic. This technique necessitates that all the paths from the input to the output have almost equal delay[15]. This is a serious constraint and often prohibits the use of wave pipelining in general. However, NPCPL, by virtue of its symmetric structure, is a promising logic family to be exploited for wave pipelining.

4. **Area :** NPCPL for low area applications offers significant advantage of area as mentioned in section 3.3. In this case also the delay is less than that of FCC circuits as may be seen from the Table 2.

   The NPCPL designs for high-speed operations, offer moderate advantages in terms of area. Area optimisation is not so drastic in this case as delay because of extra interconnections required in dual-rail logic. A full adder implemented in NPCPL takes an area of $92 \times 90\mu^2$ while the same in FCC takes $100 \times 100\mu^2$.

5. **Higher Logic Functionality :** Because of the dual-rail logic, NPCPL has balanced delay among different propagation paths. This is essential for achieving high throughput, and a desirable feature for a combinatorial design. NPCPL can implement the XOR function very efficiently, and hence, is ideal for arithmetic circuits.

6. **Sea-of-Gates Approach :** NPCPL can be easily realized on sea-of-gates technology. Using the same generic NPCPL building block, by proper configuration of inputs, different logic functions are easily realised. Hence, in sea-of-gates approach, basic NPCPL generic building blocks can form the basic cells arrayed out on a master die.

	FCC1 (high speed)	NPCPL1 (high speed)	FCC2[16] (low area)	NPCPL2 (low area)
Power @ 100MHz	2.3mW	1.8mW	-	0.9mW
Area	100 X 100μ^2	92 X 90μ^2	93 X 81μ^2	63 X 57μ^2
Delay	2.7ns	1.2	3.4ns	1.6ns

Table 2: Comparison of Area, Power and Delay of Different Logic Styles

7. **Power Dissipation :** As seen from the discussion in section 3.1.1, the static power dissipation in NPCPL is not a serious concern. Because of the low gate capacitances and low voltage swing (0 to $(V_{DD} - V_{Th_n})$) at the internal nodes, NPCPL has a lower dynamic power dissipation than that of FCC. The contribution of static power dissipation to the overall power dissipation is well amortised by the low dynamic power dissipation figure of NPCPL.

The only disadvantage of NPCPL is its relatively low noise margin as compared to static CMOS. However, all the high-speed logic families have noise margin less than that of FCC. Hence, an acceptable degradation in noise margin is a reasonable trade-off for speed of operation and elegant pipelining methods. It is a general observation that all the high-speed logic families have noise margin less than that of CMOS and NPCPL is no exception. We summarise the area, power and speed of full adders in NPCPL and FCC in 1.6μ C3TU process in Table 2.

5 Conclusions and Further Research Directions

In this paper we have presented a Normal Process Complementary Pass Transistor Logic (NPCPL) for low latency and high throughput applications. We have shown that NPCPL offers the best speed of operation comparable to CPL. It permits pipelining to the finest grain with negligible overhead of area and latency as opposed to other logic families where an increase in pipelining throughput is encumbered with heavy area and latency penalty. With NPCPL, it is possible to exploit both latency and throughput simultaneously to the maximum realisable extent. Because of its modularity and higher logic functionality, NPCPL bears the potential for a sea-of-gates realisation. As a further research direction we are working towards exploiting NPCPL design for wave pipelining.

Acknowledgments

The authors wish to thank Ms. S. Sheila and Mr. S. Balakrishnan for their help at different stages of this work. This work is supported in part by DoE under sponsored project DE - NMC/SP - 013.

References

[1] N. Weste and K. Eshragian, *Principles of CMOS VLSI Design: A Systems Perspective*,Addison-Wesley.

[2] K.Yano *et al*, "A 3.8 ns 16 x 16-b Multiplier Using Complementary Pass-Transistor Logic, ". *IEEE Jl. of Solid-State Circuits, SC-25, April 1990*, pp.388-395.

[3] J.H.Pasternak *et al*, "CMOS Differential Pass Transistor Logic", *IEEE Jl. of Solid-State Circuits, SC-22, April 1987*, pp. 216-222.

[4] K. M. Chu *et al*, "A Comparison of CMOS Circuit Techniques : Differential Cascode Voltage Switch Logic vs. Conventional Logic", *IEEE Jl. of Solid-State Circuits,SC-22*, August 1987, pp. 528-532.

[5] L.C.M.G. Pfennings *et al*, "Differential Split-level CMOS Logic for Subnanosecond Speeds". *IEEE Jl. of Solid-State Circuits, SC -20, October 1985*, pp. 1050-1055.

[6] T. A. Grotjohn *et al*, "Sample-Set Differential Logic (SSDL) for Complex High Speed VLSI", *IEEE Jl. of Solid-State Circuits,SC-21, April 1986*, pp. 367-369.

[7] Shih-Lien Lu, "Implementation of Iterative Networks with CMOS Differential Logic." *IEEE Jl. of Solid-State Circuits,SC-23, August 1988*, pp. 1013-1017.

[8] Cheng-Yu Liu *et al*, "Latched CMOS Differential Logic for Complex High-Speed VLSI", *IEEE Jl. of Solid-State Circuits, SC-26, September 1991*, pp. 1324-1328.

[9] I.S.Hwang *et al*, "Ultrafast Compact 32–bit CMOS Adders in Multiple-Output Domino Logic(MODL)", *IEEE Jl. of Solid-State Circuits, SC-24, April 1989*, pp.358-369.

[10] A.P. Chandrakashan *et al*, "Low-Power CMOS Digital Design", *IEEE Jl. of Solid-State Circuits, SC-27, April 1992, pp. 473-483.*

[11] Fang Li *et al*, "Adaptively Biased PseudonMOS Logic ", *ISCAS 90 vol. 1*, pp. 562-565.

[12] Anura P. Jayasumana *et al*, "Logic Design Using Using Pass Transistors", *VLSI System Design, 1990, India.*

[13] Damu Radhakrishnan *et al*, "Formal Design Procedures for Pass Transistor Switching Circuits", *IEEE Jl. of Solid-State Circuits, SC-20, April 1985*, pp. 531-536.

[14] C.Asato *et al*, "A Data-Path Multiplier with Automatic Insertion of Pipeline Stages", *IEEE Jl. of Solid-State Circuits, SC-25, April 1990.*

[15] Derek Wong *et al*, "Design of High-Performance Digital Circuits Using Wave Pipelining", *VLSI 1989.*

[16] S. Ramanathan *et al*, A Methodology to Generating Application Specific Tree Multipliier", *Procedings of VLSI Design '93.*