International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 4 Issue 11, November-2015

Design of Various 4 Bit Shifters using CMOS 32nm Technology

Saptarshi Patnaik,
Bachelor of Engineering
Department of Electronics & Telecommunication
K. J Somaiya College of Engineering-Vidyavihar-400077
Mumbai, Maharashtra, India

Shreyas Pandey,
Bachelor of Engineering
Department of Electronics & Telecommunication
K.J Somaiya College of Engineering-Vidyavihar-400077
Mumbai, Maharashtra, India

Sanchit Patil,
Bachelor of Engineering
Department of Electronics & Telecommunication
K.J Somaiya College of Engineering-Vidyavihar-400077
Mumbai, Maharashtra, India

***Abstract*— Bit-wise operations are pivotal when programming hardware registers in embedded systems. There are many instances where we need to change only one bit of a complete register, in this case, changing the whole register value can be time consuming and undesirable. Bit shifting is one of the many useful bit operations one can perform. In this paper we have designed bit-shifter using CMOS 32nm technology that can perform left-shift, right-shift, no-shift and arithmetic shift depending upon what the user demands, and studied its characteristics. The simulation results are carried out using SILVACO software.**

Keywords— CMOS 32nm Technology, Shifter, SILVACO

I. INTRODUCTION

It is the area of hot research nowadays to improve the characteristics of a circuit by reducing its size and power consumption. 32 nanometer" refers to the average half-pitch (i.e., half the distance between identical features) of a memory cell at this technology level. The bit shifters help in shifting the bit (left shift/ right shift) and also store the out-going bit for some required operation. Due to the vital role played by the bit-shifter in many arithmetic as well as logical operations. An optimized design of bit shifter is required to achieve low power, small size and delay. In this paper we have tried to design bit shifters with minimum number of transistors possible. The objective of the design methodologies in this paper is to propose a simple but accurate design of different types of Shifters.

II. BASIC THEORY OF SHIFTERS

There are four types of shifters:

a. Logical Shifter (Shift Right / Shift Left)
b. Arithmetic Shifter
c. Barrel Shifter
d. Funnel Shifter (Combination of the above three Shifters)

➢ *LOGICAL SHIFTER*:

Left Shift and Right Shift Operation

As we can see in the figure, Left shift involves shifting the input bits to the left and 0 is appended at the Least Significant Bit(LSB).The Most Significant Bit(MSB) of the input is discarded or it can be saved for further use. By, shifting the input one time to the left we are actually multiplying the input by the 2

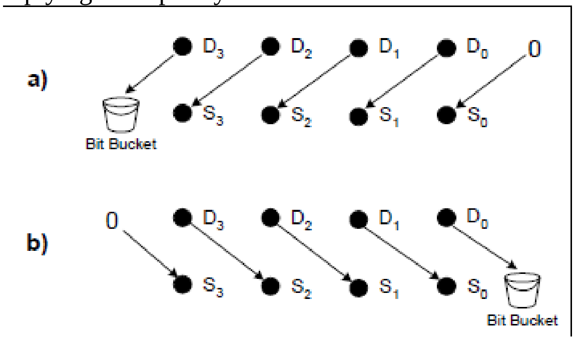


Figure 1: Basic left shift operation

a. Left Shift Operation

b. Right Shift Operation

In Right Shift operation, the input bits are shift right and 0 is appended at the Most Significant Bit(MSB).The Least Significant Bit(LSB) of the input is discarded or stored for any further operations. By, shifting the input right one time we are actually dividing the input by 2.

General Structure for Left and Right Shift Operation

As we can see in the figure below, both Left Shift and the Right Shift is being done in the same circuit depending on one of two control lines. Examine the AND gates at the extreme left and at extreme right, If the LEFT control line is active, the leftmost AND gate sends its output to the “bit bucket.” If the RIGHT control line is active, the other AND gate of the pair sends signal D3 to output S2 via the OR gate. Each of the remaining pairs of AND gates can send their outputs either left or right, depending on which control line is active. Also it is worth noting that we are not saving the LSB, MSB of the input bits for Left, Right Shift operation respectively.

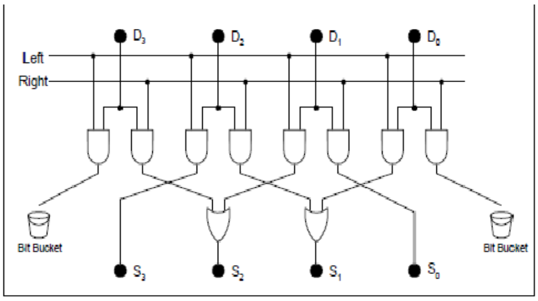


Figure 2 : Modification of General Structure

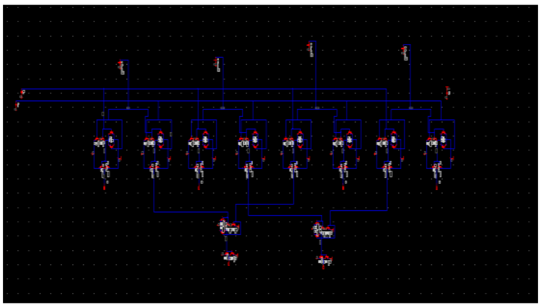


Figure 3 : Schematic of General Structure

A major problem in above structure was that the bit that was shifted out was not stored. However, the bit shifted out is of interest. In a right shift, it is the remainder after division by two. In a shift left, it can be tested for significance so that the results of a left shift can be evaluated for validity. The bits shifted out are available from the AND gates at the extreme left and right of the circuit.

In the below figure, this issue is resolved such that both shifted out bits are given input to the OR Gate.

For the circuit shown, this will be D3 for a left shift or D0 for a right shift. Often the bit shifted out is saved in the CARRY flag.

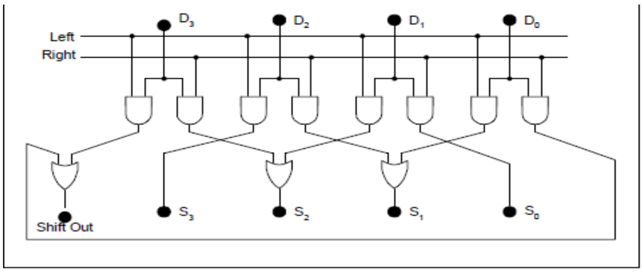


Figure 4: Improvised General Structure

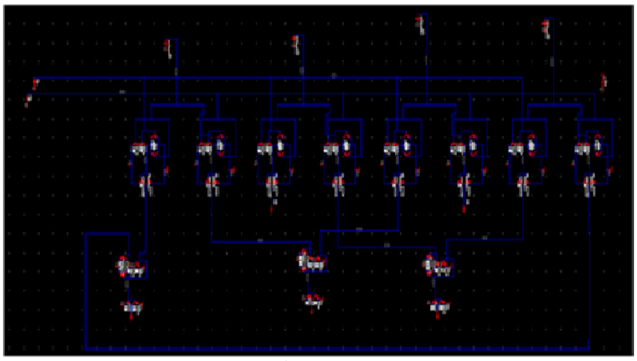


Figure 5: Schematic of Improvised General Structure

If we design a shifter that can shift left or right, data must be routed through the shifter when shifting is needed and around it at other times. This increases the complexity of the control unit. Another alternative is to design the shifter to have three options: shifting left, shifting right, or not shifting at all. This compromise adds two gate delays when shifting is not needed, but at a saving of complexity elsewhere. Adding a “no shift” option is accomplished with a third control line and a third AND gate for each bit. In the below figure an extra option (No Shift) is added. Each input bit now drives three

AND gates, only one of which is selected by one of the three control lines. The middle AND gate sends input to output without shifting

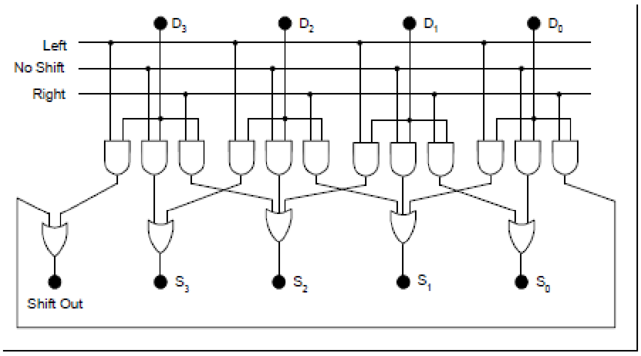


Figure 6: Left/Right/No Shift

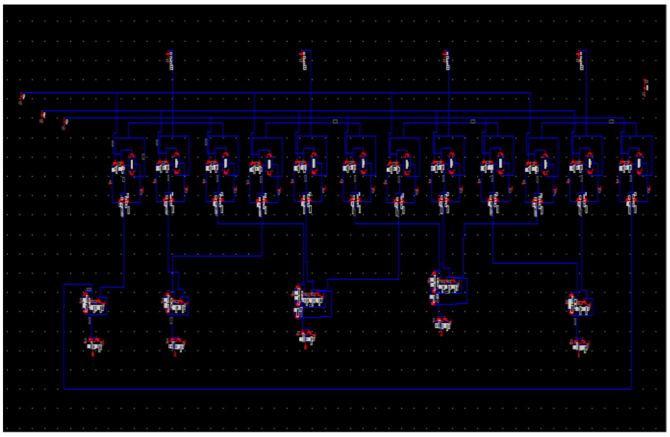


Figure 7 : Schematic

➢ *ARITHMETIC SHIFTER*

So far we have no way to deal with the sign bit when performing a right shift on signed data. In other words, we cannot do an arithmetic right shift. For a logical right shift, a zero is supplied at the left. Each of the circuits above does this. For an arithmetic shift, the leftmost bit of the input is considered the sign bit. It must be shifted to the right *and also* copied to the leftmost bit of the output. The design for such a shifter is shown below. A fourth control line is added for arithmetic right shift. An OR gate drives the RIGHT control line when either an arithmetic or a logical right shift is commanded. A second OR gate drives the copy function for the leftmost bit when an arithmetic right shift is commanded.

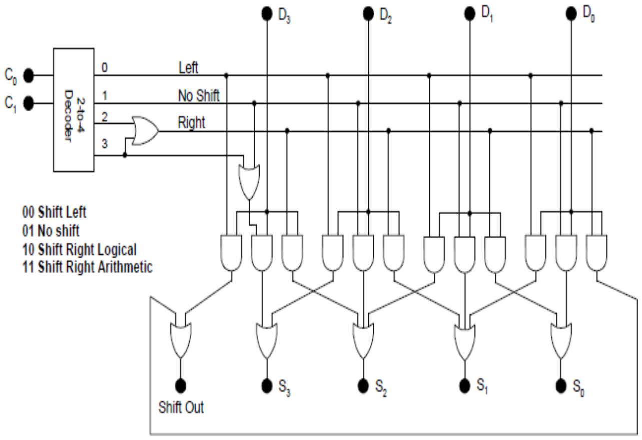


Figure 8 : Final Circuit

A decoder is used to generate the control lines. This means that the control unit has to generate only two bits to specify shifting and guarantees that one and only one set of control lines within the shifter is active at any time. The codes used for the four types of shifts are arbitrary.

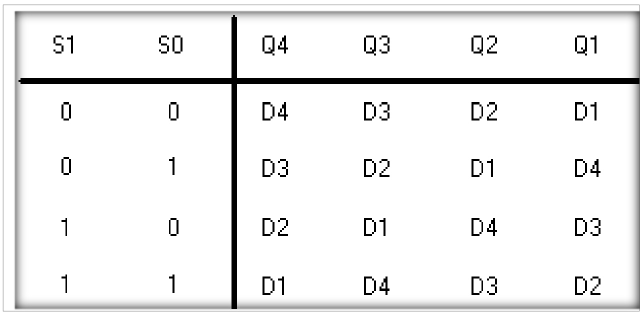
➢ *BARREL SHIFTER*

It is a very important part of a combinational Logic Block. It has the ability to shift data work in a single operation over standard shift left or right registers that utilize more than one clock cycle. It rotates numbers in a cycle such that empty spots are filled with bits shifted of the other end.

Example: Barrel Shift Right - 1100 after shift operation will become 0110

Barrel Shift Left – 1010 after shift operation will become 0101

TABLE 1: Input-Output Table of Barrel Shifter



S1	S0	Q4	Q3	Q2	Q1
0	0	D4	D3	D2	D1
0	1	D3	D2	D1	D4
1	0	D2	D1	D4	D3
1	1	D1	D4	D3	D2

III. SIMULATION RESULTS

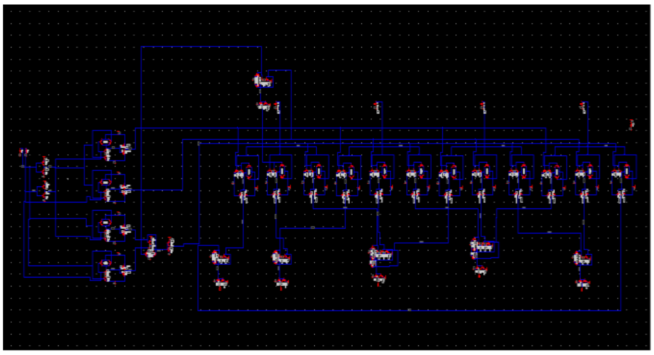


Figure 9

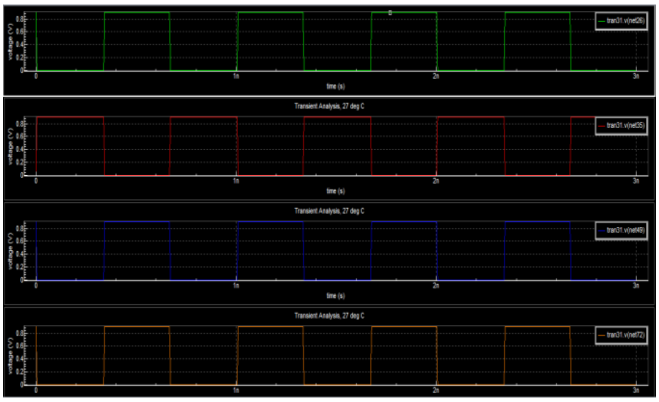


Figure 10 : Input

This is the input that we have given to the shifter. There are 4 inputs namely D0, D1, D2, D3 .Here D0 is assumed to be LSB and D3 is assumed as MSB of the input bits.
Each input is shown in different color such that:
D3-Green
D2-Red
D1-Blue
D0-Yellow

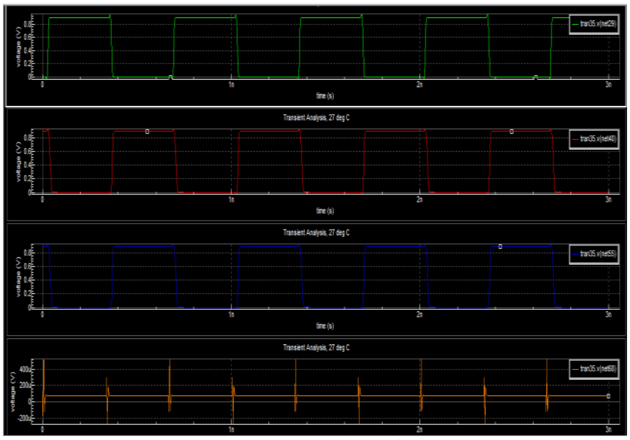


Figure 11 : Left Shift

Now if we select Left Shift as operation using the control bits (0,0) in multiplexer we get the above output.
If S0,S1,S2,S3 are assumed as the outputs, and taking S0 as LSB and S1 as MSB, then in Left Shift S0 is taken as 0 and DO,D1,D2 are passed to S1,S2,S3 respectively. The same we have got in the output.S0 bit has 0 value and all the other bits are just the shifted version of the input.

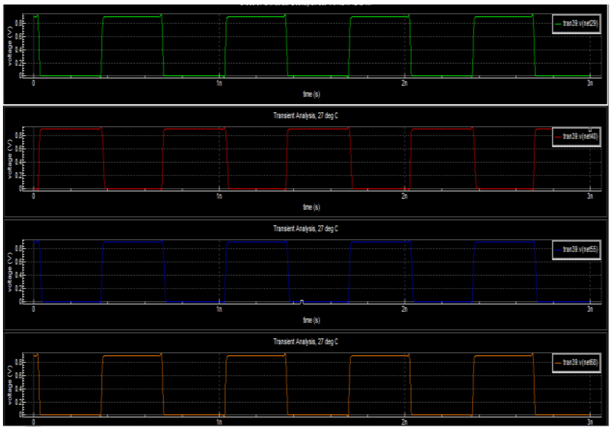


Figure 12 : No shift

Now if we select the control bits in multiplexer as (0, 1) then no shift operation is performed.
As we can see above the output is same as that of the input. D0, D1, D2, D3 is passed to S0, S1, S2, S3 respectively without any shift operation.

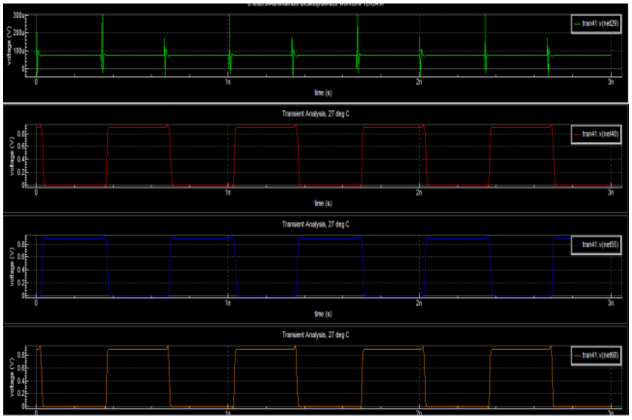


Figure 13 : Right Shift

Now if the control bits are (1, 0) then Right Shift Operation is performed on the input.
0 is passed to S3 and D3, D2, D1 is passed to S2, S1, and S0 respectively. The output of the Right Shift is shown above and as we can see S3 has 0 value and all the other bits are shifted version of the input.

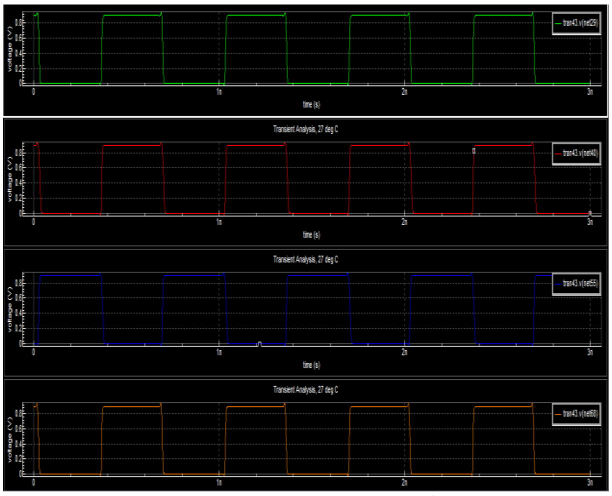


Figure 14: Arithmetic Shift

The arithmetic shift is same as that of the right shift just that the input D3 is passed to S3 as well as to S2.So, the output signal of both S2, S3 is same. The Arithmetic Shift can be done by passing (1, 1) as control bits to the multiplexer.

So, as we can see we can perform Left Shift, No Shift, Right Shift and Arithmetic Shift through this circuit. And since it is made using CMOS so the size as well as complexity of the circuit is negligible

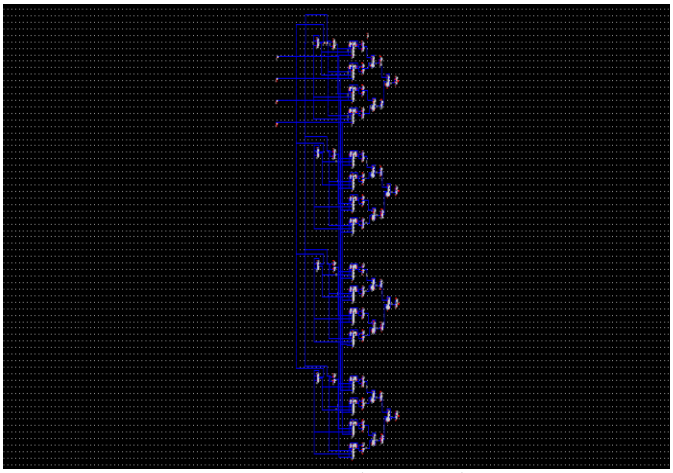


Figure 15: Schematic of Barrel Shifter

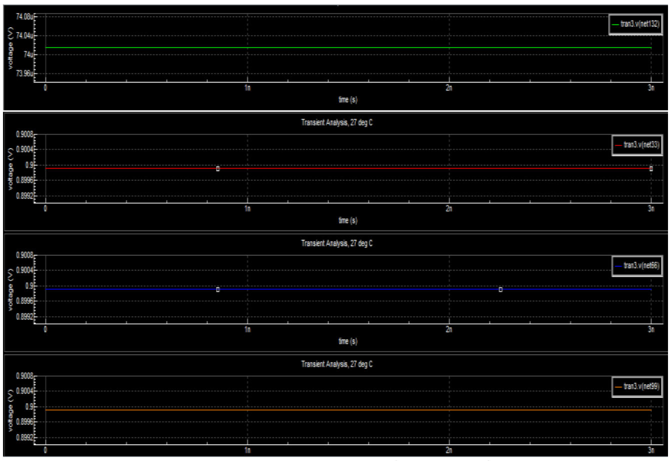


Figure 16: Schematic output of barrel shifter

For an input of 1011, the output comes out to be 0111. Here we can see that no additional 0 has been added. The number has been rotated clockwise 3 times.

TABLE 2: Parameters of Shifters

Types of Shifters	Number of transistors
Left Shift / Right Shift	66
Arithmetic	52
Barrel	216
W/L ratio of pmos	7/1
W/L ratio of nmos	2.4/1
Delay	27.4ps

IV. NCLUSION

Different types of Shifters are studied, designed, analyzed by using SILVACO EDA TOOL full custom using 32nm technology.
The schematic has been implemented using cmos. The W/L ratio of the transistors are chosen so as to get minimum delay possible between the input and output.
The delay in the shifter circuit is 27.4ps.

ACKNOWLEDGMENT

It is our pleasure to refer Microsoft Word exclusive of which the compilation of this project would have been impossible. An assemblage of this nature could never have been attempted with our reference to and inspiration from the works of others whose details are mentioned in references section. We acknowledge our indebtedness to all of them. Last but not the least my sincere thanks to all my friends who have patiently extended all sorts of help for accomplishing this undertaking.

V. EFERENCES

[1] Barry Paton,”Fundamental of Digital Electronics”, Clarkson, March 1998

[2] J. Rabaey, A. Chandrakasan, B. Nikolic, Digital Integrated Circuits, 2nd edition, Prentice Hall, 2003

[3] Liqiong Wei; Roy, K; De, V.K., “Low voltage low power CMOS Design techniques for deep submicron ICs”, VLSI Design.2000. Thirteenth International Conference, Page(s) 24 – 29, 3-7, Jan. 2000.

[4] J. Rocha, M. Santos, J. M. Dores Costa, F. Lima, “High voltage Tolerant level shifters and DCVSL in standard low voltage CMOS Technologies,” in Proc. IEEE Int. Symp. Ind. Electron, pp. 775–780, Jun. 2007

[5] Andrew D. Booth. “A signed binary multiplication technique.” The Quarterly Journal of Mechanics and Applied Mathematics, Volume IV, Pt. 2, 1951 [1]

[6] Q. LI, G. LIANG, and A. BERMAK, “A High-speed 32-bit Signed/Unsigned Pipelined Multiplier,” IEEE 5th Int. Sym-posium Electronic Design, Test and Application, pp.207- 211, Jan. 2010.

[7] Chang, Kou-Chuan. Digital design and modeling with VHDL and Synthesis. IEEE Computer Society Press, 1996.

[8] J.C.Garcia,J.A Montiel –Nelson, S Nooshabadi , “Bootstrapped Power efficient CMOS driver,”IEEE Int Conf on Microelectronics , pp.30-35, Dec 2005.

[9] Kevin Naviand Omid Kavehei“Low-Power and High-Performance1-BitCMOSFull-AdderCell"Journal of Computers, VOL. 3, NO.2, February2008

[10] Hanchate, N.; Ranganathan,N; (2004) “A new technique for leakage Reduction in CMOS circuits using self-controlled stacked Transistors,” 17th International Conference on VLSI Design, pp.228-233.

[11] K Sathyaki and Roy Paily, (2007) “Leakage Reduction by Modified Stacking and Optimum ISO Input Loading in CMOS sevices,” 15th International Conference on Advanced Computing and Communications, pp.18-21.

[12] A. P. Chandrakasan, S. Sheng, and R. W. Brodersen, (1992) “Low-power CMOS digital Design,” IEEE J. Solid-State Circuits, vol. 27, no.4, pp. 473– 484

[13] S. H. Kulkarni and D. Sylvester, "Fast and Energy Efficient Asynchronous Level Converters for Multi VDD Design," IEEE International SOC Conference, 2003. pp. 169-172.