4-bit Brent Kung Parallel Prefix Adder Simulation Study Using Silvaco EDA Tools

Anas Zainal Abidin, Syed Abdul Mutalib Al Junid, Khairul Khaizi Mohd Sharif,
Zulkifli Othman, Muhammad Adib Haron
Faculty of Electrical Engineering
Universiti Teknologi Mara
Shah Alam, 40450, Selangor, Malaysia
e-mail: samaljunid@salam.uitm.edu.my

Abstract — **In IC design environment, the chip performance is influence by design environment, schematic and sizing parameter of the transistor. Therefore, this study is an attempt to investigate the performance of 4-bit Brent Kung Parallel Prefix Adder using Silvaco EDA Tools and targeted to 0.18um Silterra Technology. The objective of this project is to review the performance of the adder by forming different of transistors gate sizing and schematics. Furthermore, the study been carried out by implementing Brent Kung Adder in Basic Logic Gate and Compound Gate, then simulate the design in various sizes of transistors in order to see the effects on propagation delay, power consumption and the number of transistors used. At the end of this paper, evidently the improvement of transistors size contributes reducing the propagation delay and proportionally advances the power consumption. Besides, the Compound Gate takes about 35.58% power consumption decreased, reduced 9.16% of propagation delay and less 96 transistors used rather than Basic Logic Gate. Nevertheless, larger size of buffers required to stable the output consistency in Compound Gate schematic.**

Keywords - Brent Kung Adder, Parallel Prefix Adder

I. INTRODUCTION

An adder is one of the basic building blocks of common data path components, As such, they are of immense importance to designer being so commonly used and such a critical part of the data path. For smaller adders, carry-look ahead, carry skip or carry select will suffice, but as the width of the adder grows, the delay of passing the carry through the stages begin to dominate [1]. Therefore, in current technology, Parallel Prefix Adder are among the best adders, with respect to the area and time (cost: performance ratio), and are particularly good for the high-speed addition of large numbers [2]. Moreover, the requirements of the adder are that it is primarily fast and secondarily efficient in terms of power consumption and chip area [3].

Parallel Prefix Adder as terminology background is describing prefix as the outcome of the execution of the operation depends on the initial inputs. Parallel in this term is defines as the process of involving the execution of an operation in parallel. This is done by segmentation into smaller pieces that computed in parallel [4]. Then all bits of the sum will begin the process concurrently. There are a lot of parallel prefix adders been developed example in 1960: J. Sklansky–conditional adder, 1973: Kogge-Stone adder, 1980: Ladner-Fisher adder, 1982: Brent-Kung adder, 1987: Han Carlson adder and 1999: S. Knowles. Other parallel adder architectures also include H. Ling adder in 1981 and 2001: Beaumont-Smith [4].

Practically, the Brent Kung Parallel Prefix Adder has a low fan-out from each prefix cell but has a long critical path and is not capable of extremely high speed addition [3]. In spite of that, this adder proposed as an optimized and regular design of a parallel adder that addresses the problems of connecting gates in a way to minimize chip area. Accordingly, it considered as one of the better tree adders for minimizing wiring tracks, fan out and gate count and used as a basis for many other networks [1].

II. PRELIMINARY BACKGROUND

In Parallel Prefix adder case, binary addition usually expresses in terms of carry generation signal g_i, carry propagation signal p_i, carry signal c_i, and sum signal s_i, at each bit position ($1 \leq i \leq n$) [6], all these signals can be obtain by regard to the equation below:

$$g_i = a_i + b_i \quad (1)$$

$$p_i = a_i \oplus b_i \quad (2)$$

$$c_i = \begin{cases} g_i \\ g_i + p_i c_{i-1} \end{cases} \quad (3)$$

$$s_i = p_i \oplus c_{i-1} \quad (4)$$

This Adder computes the sum in three stages as shown as the block diagram in Fig. 1.

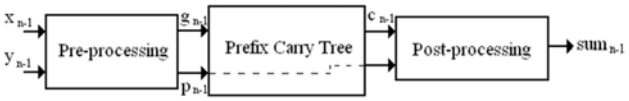


Fig. 1: Block Diagram of Parallel Prefix Adder [5]

A. *Pre-processing stage*

Block diagram illustrated above represent the *n* bit Parallel Prefix Adder operation that begin with Pre-processing stage for generating p_i and g_i as in equation (1) and (2).

B. *Prefix Carry Tree stage*

The signal from the first stage will proceed with the next stage, to yield all carry bits signal. The stage containing three main complex logic cells called as Black cell, Gray cell and buffer cell. Black cell compute both G_{ij} and P_{ij} as define in equation (5) and (6), whereas Gray cell only execute G_{ij} [5]. The stage of Prefix carry tree is a part that differentiate or determine the adder used.

Brent Kung Prefix Tree schematic structure showed in Fig. 2 with the content of three cells at Fig. 3.

$$G_{ij} = G_{ik} + P_{ik} \times G_{k-1j} \quad (5)$$

$$P_{ij} = P_{ik} \times P_{k-1j} \quad (6)$$

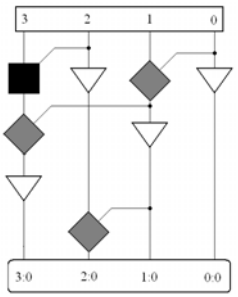


Fig.2: Brent Kung Prefix Tree [5]

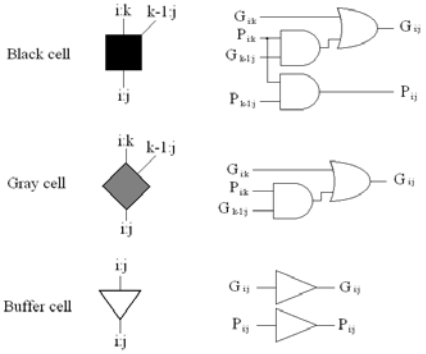


Fig. 3: Complex logic cells inside the Prefix Carry Tree [5]

C. *Post-processing stage*

Complement the overall adder operation, carry bits that produced from the second stage shall pass through the last part known as Post-Processing stage. The procedure to obtain the final adder result is able by the following equation (4).

III. DESIGN METHODOLOGY

Simulation Study of 4-bit Brent Kung Parallel Prefix Adder Using SILVACO EDA Tools develop by following the step that show in the Fig. 4.

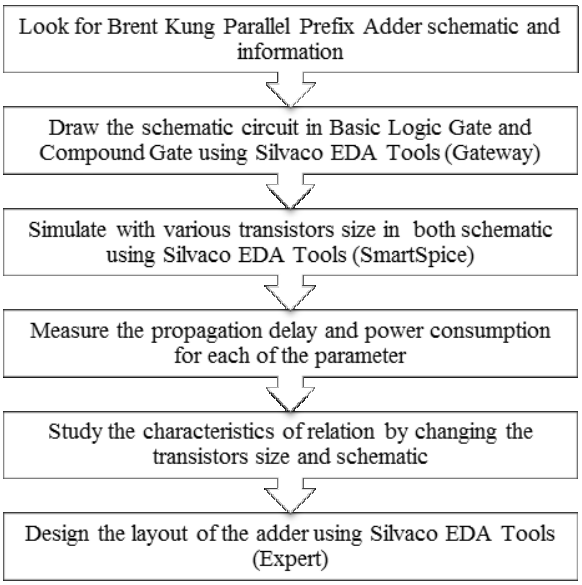


Fig. 4: Project flow chart

A. *Basic Logic Gate*

By using Basic Logic Gate schematic, the project is necessary to design overall transistors circuit in three general gate which are NAND, NOR and inverter such in Fig. 5. Then, to make it systematic in design implementation, simply OR gate, AND gate and XOR gate certainly can be develop as in Fig. 6.

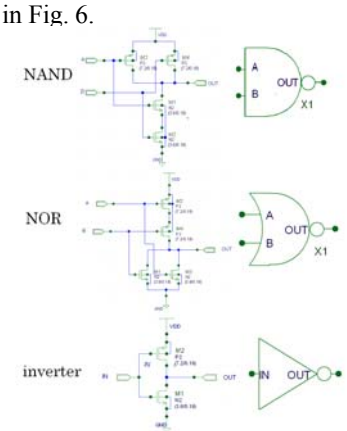


Fig. 5: Schematic and Symbol for NAND, NOR and inverter

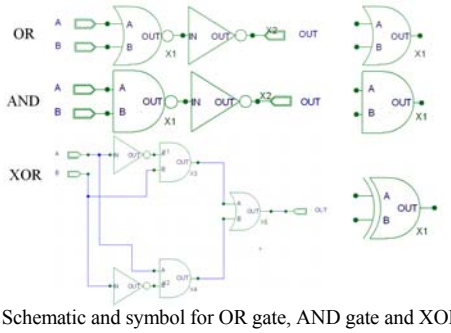


Fig. 6: Schematic and symbol for OR gate, AND gate and XOR gate

Specific into pre-processing stage, it is obviously consumes two input ports a_i and b_i while producing generation signal, g_i and propagation signal, p_i. By refer to the equation (1) and (2), the pre-processing block has managed to create a circuit as shown in Fig. 7. The block of this stage should be put at every single input bits of the adder.

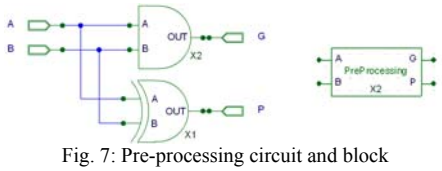


Fig. 7: Pre-processing circuit and block

Based on the basic logic gates created, the tree adder placed in between Pre-processing stage and Post-processing stage was design in accordance with equation (3), (5) and (6) as well in Fig. 2 and Fig. 3 which been discussed previously. Necessarily, Black cell, Gray cell and buffer cell schematic as Fig. 8 should be done before assembling the complete tree for Brent Kung Parallel Prefix Adder type such in Fig. 9 (a) and (b).

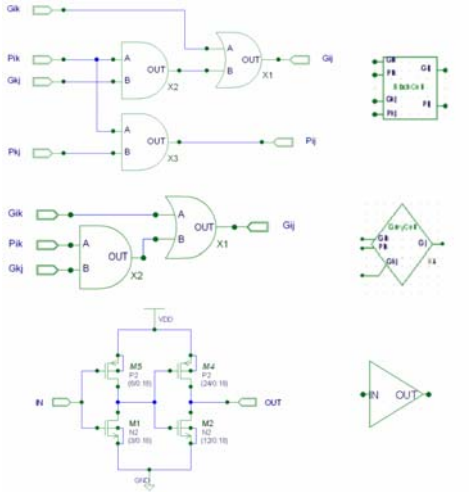
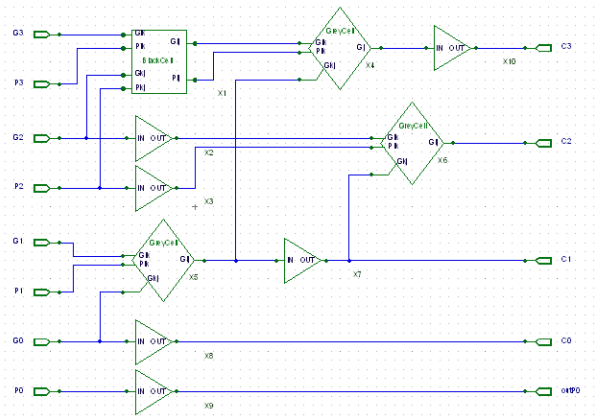
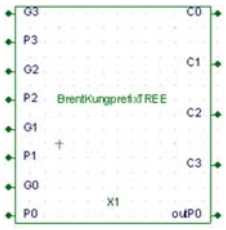


Fig. 8: Schematic and block for the three main cells (Black, Gray and buffer)



(a)



(b)

Fig. 9: (a) 4-bit Brent Kung Parallel Prefix Adder Tree schematic design, (b) Block diagram of the tree adder stage

Final stage of the design namely as Post-processing is primarily doing the exclusive-OR operation between the propagate signal, p_i and a bit lower carry signal output from the tree adder, c_{i-1} refer to equation (4).

As seen in Fig. 10, the entire structure of this stage combined in one block aimed for prevent confusion during work on regulating the full schematic.

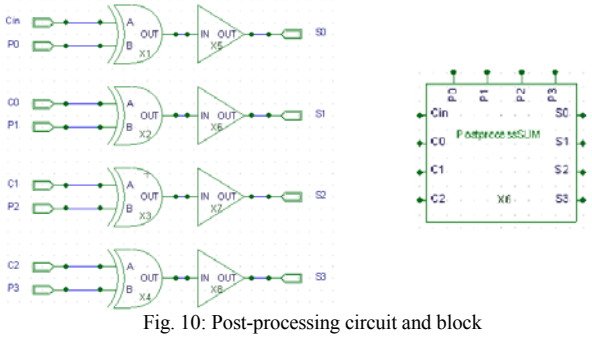


Fig. 10: Post-processing circuit and block

At this point, 4-bit Brent Kung Parallel Prefix Adder in full schematic can be a form and present in Fig.11. All three main stages and match the entire ports connection by related equations are compiling together.

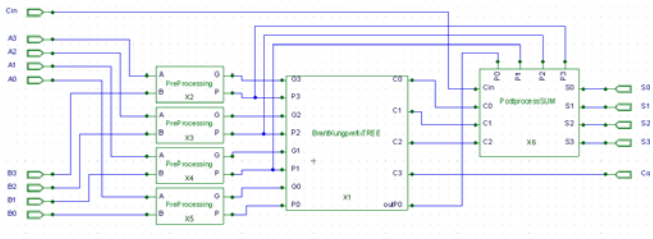


Fig. 11: Complete diagram of 4-bit Brent Kung Parallel Prefix Adder

B. Compound Gate

Other than that, this report also performed schematic arrangement in Compound Gate, which, comprises of combination of the series and parallel switch structures [7]. This technique has evidenced to decrease the number of transistors used and directly subtract the total area consumption in layout design, compared to Basic Logic Gate circuit implementation.

This method can be conducted by converting the circuit inside the block with satisfied as Compound Gate format. The rest of 4-bit Brent Kung Parallel Prefix Adder schematic connections has to be the same arrangement.

Fig. 12 show the XOR schematic in Compound Gate system while the Pre-processing stage circuit for Compound Gate resource established at Fig. 13.

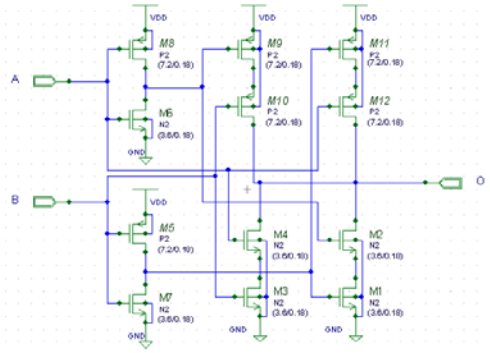


Fig. 12: XOR schematic in Compound Gate

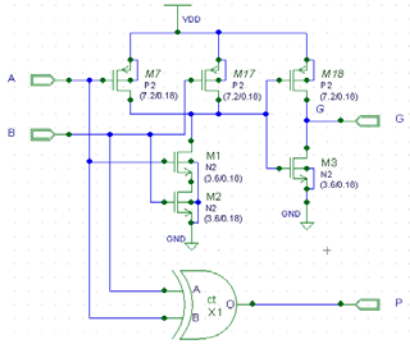


Fig. 13: Pre-processing schematic in Compound Gate

The tree stages which normally generate carry bits signals and comprises Black Cell and Gray Cell as for replace to Compound Gate circuit e.g. Fig. 14.

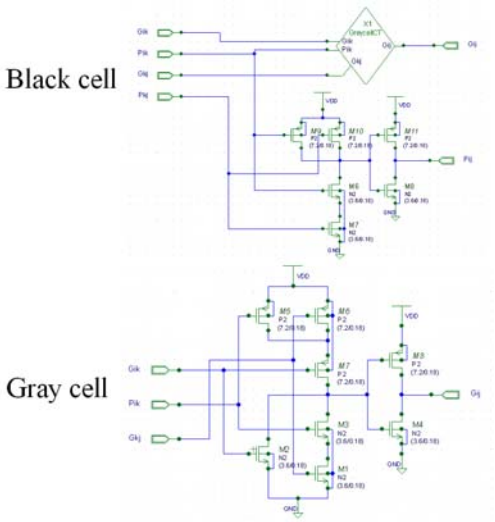


Fig. 14: Black cell and Gray cell in Compound Gate

Eventual operation as Post-processing stage regularly used EXOR and buffer to compute desired result. Thus, Compound Gate system required substitution of logic gate such in Fig. 12 instead of former schematic design.

IV. Result and Discussion

Before the project is forward into further inference, the process of summation have obligatory to test with certain input signals in order to assure the schematic design implementation was exactly and accurately. Illustrated in Fig. 15, act to injecting input signal for the adder and supplied the voltage drain, *Vdd*, the circuit able to simulate subsequently check the equivalent of mathematical answer and practical output result. Provided in Fig. 16, it indicated the signals injection set for input a_i and b_i whereas the output test of this adder signals, c_{out} and s_i are show in Fig. 17.

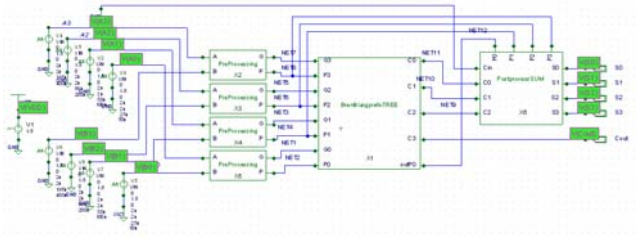
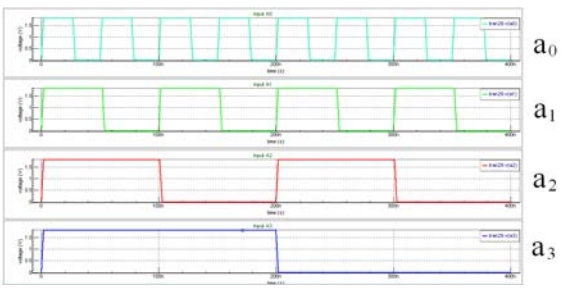


Fig. 15: input signal injected and test port marked



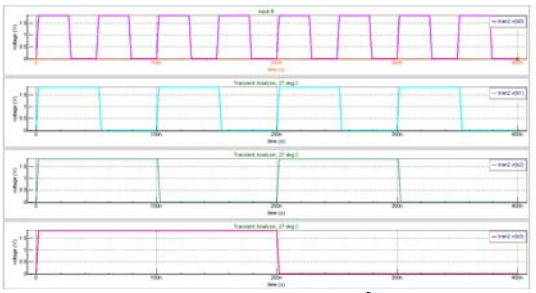


Fig. 16: input a_i and b_i

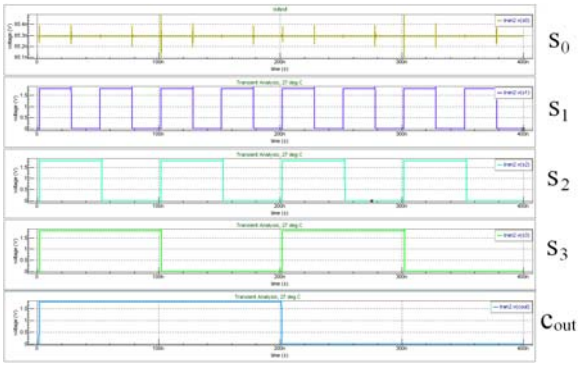


Fig. 17: output c_{out} and s_i

A. *Basic Logic Gate implementation results*

Applied *0.18um* for length and *1.8V* supplied to the drain voltage, $V_{dd;}$ the project begin for Basic Logic Gate implementation considered permanent transistor sizes inside the buffers which mention in Table 1 where include PMOS width, W_p and NMOS width, W_n. Buffer cells normally been used to reduce the glitch and achieve high speed performance at the same time [9]. Guided information in Fig. 18 indicates the separation of two sections inside the buffers to ease for specific the transistors.

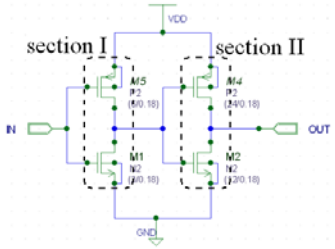


Fig. 18: Divide buffer into two sections

Table 1: Transistor size in the buffers

Buffer transistors: Width size (*um*)	
Section I (W_p/W_n)	**Section II (W_p/W_n)**
6/3	24/12

Next, this project performs in propagation delay measurement using Silvaco EDA Tools (Smart Spice) application. By definition, propagation delay is the amount of time that it takes to a change in the input signal to produce a change in the output signal [8].

Two types of trigger for output signals known as rise (low to high) and fall (high to low) been considered, the measured implementation done as Fig. 19 (a) rising propagation delay and Fig. 19 (b) falling propagation delay for one of the input and output signals simulation.

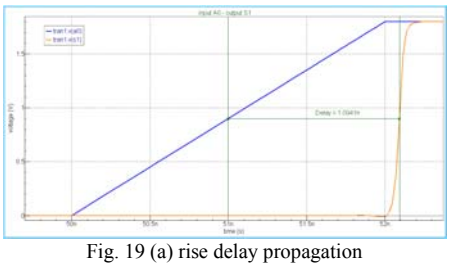


Fig. 19 (a) rise delay propagation

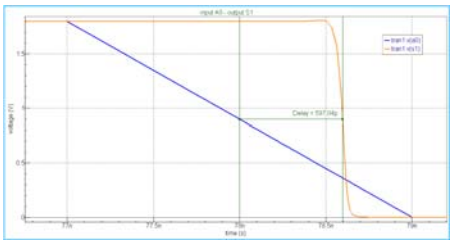


Fig. 19 (b) fall delay propagation

The propagation delay for five different sizes of transistor obtained using the same procedure. All distance delay computed from input a_0 because of the way for the most transistors pass through toward any output pins. As tabulated in Table 2, the scale of every output signal provided that used to refer at the figure of Propagation Delay for the rising at Fig. 20 (a) and the falling as in Fig. 20 (b) on every single output signals in order to study the initial cause that affects in transistor sizes manipulation excluding buffer transistors.

Table 2: Scale of Width Transistors size

Scale Size	**PMOS width, W_p (*um*)**	**NMOS Width, W_n (*um*)**
3	0.54	0.27
4	0.72	0.36
10	1.8	0.9
20	3.6	1.8
40	7.2	3.6

As assumption, the value of scale size determined by the multiplication of 0.18um length of the overall transistors. Fig. 20 recorded and displayed all the measured delay data effected by five different sizes on transistor.

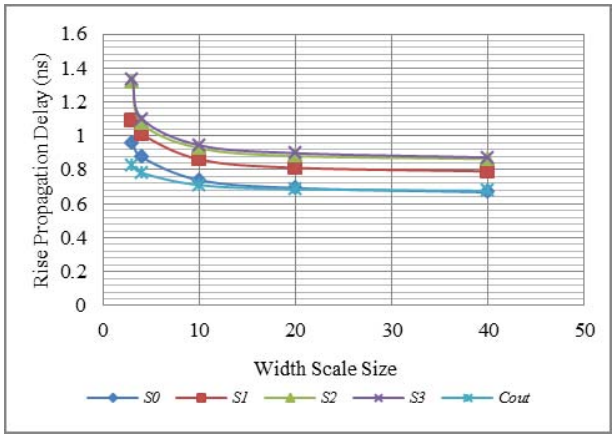


Fig. 20 (a): Comparison of Rising Propagation Delay for every Output signal in Basic Logic Gate

In Fig 20 (a), output S_3 seems to be the highest after S_2 consume the propagation delay for the rising output signals. It followed by S_1, S_0 and C_{out}. Increasing the width of transistors will exponentially minimizing the delays for the initial but showings like linearly decreasing from width scale of *12* and above.

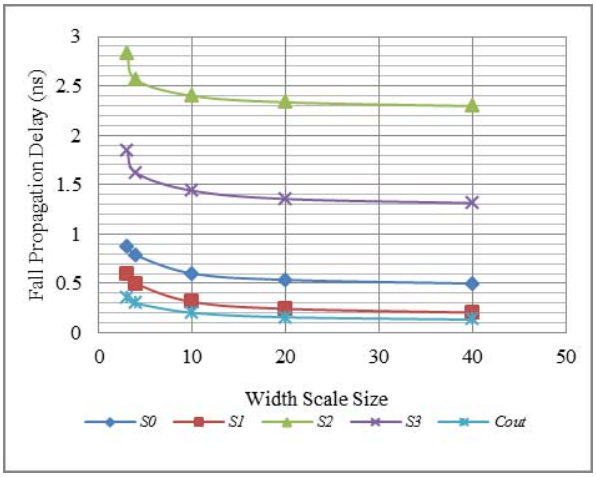


Fig. 20 (b): Comparison of Falling Propagation Delay for every Output Signal in Basic Logic Gate

Falling delay at Fig. 20 (b) shows the same behavior as in the rising part. However, output S_2 generates high propagation delay as well as S_3 which in a state of concern rather than the others signals. These indicate that the adder design totally run through the delay propagation by depending on S_2 signal.

The current value at the drain supply voltage, I_{dd} checked and the power consumption by each of the sizes, P calculated as tabulated in table 3.

Table 3: supplied current and power consumption

Width size, (*um*) *Wp/Wn*	***Supply drain current, I_{dd} (nA)***	**Power consumption, *P=IV* (*nW*)**
0.54/0.27	41.616	74.909
0.72/0.36	41.992	75.586
1.8/0.9	46.580	83.844
3.6/1.8	59.768	107.582
7.2/3.6	101.14	182.052

Expanding the size of transistors will lead to consume more power as prove in table 3. Nonetheless, in order to fabricate a speed characteristic IC, bigger size designed on the transistors as to ensure the minimum delay implemented must be approved.

B. Compound Gate implementation results

Considering in Compound Gate of schematic, the signals from input injects with marked as in Basic Logic Gate implementation and the output signals is going to be compute by the simulation such in Fig. 15. However, the glitch noises occurred often at the output signals. This makes it necessary to improve the signal with reduce unwanted condition. In this situation, Transistors sizes inside the buffer were keeping a prominent role to overcome the problem such in Fig. 21. Six transistor sizes in the buffer are fixing followed in table 4, and the sections noticed at Fig. 18.

Table 4: Six Buffer transistor sizes implementation

Buffer	**Buffer transistors: Width size (*um*)**	
	Section I (W_p/W_n)	**Section II (W_p/W_n)**
1	6/3	24/12
2	6/3	36/18
3	6/3	44/22
4	6/3	54/27
5	12/6	54/27
6	6/3	72/36

Fig. 21: reducing glitch using different sizes of transistor in the buffer

Fig. 21 shows S_0 result that one of the output signals and it seems to be good reduction of the glitch, due to wider size of buffer transistors setting. Buffer 6 as noticed in table 4 and Fig. 21 represent as the biggest size of the transistors and reduce optimum glitch.

The same way is use to produce delay propagation of every output signals in Basic Logic Gate implementation. Compound Gate implementation results of rising and falling propagation delay plot in Fig. 22 (a) and (b).

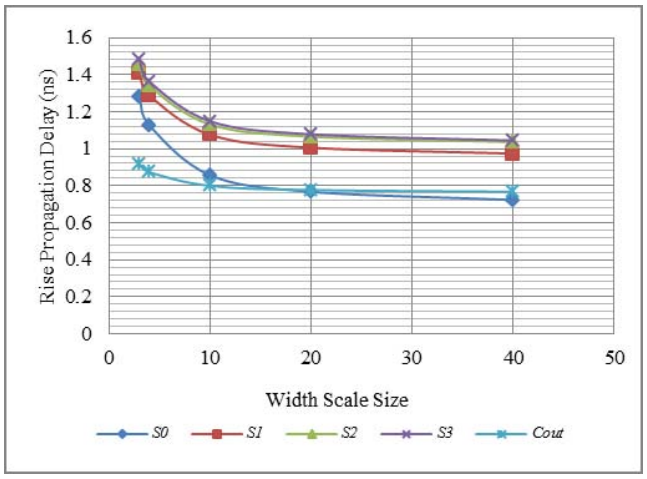


Fig. 22 (a): Comparison of Rising Propagation Delay for every Output signal in Compound Gate

High decrement formed at the earlier of the small width sizes and approximately linear decrease resulting after *15* width scale size. S_3 show as the most rising propagation delay taken and a slightly less develop by S_2 and subsequently on S_1 signals.

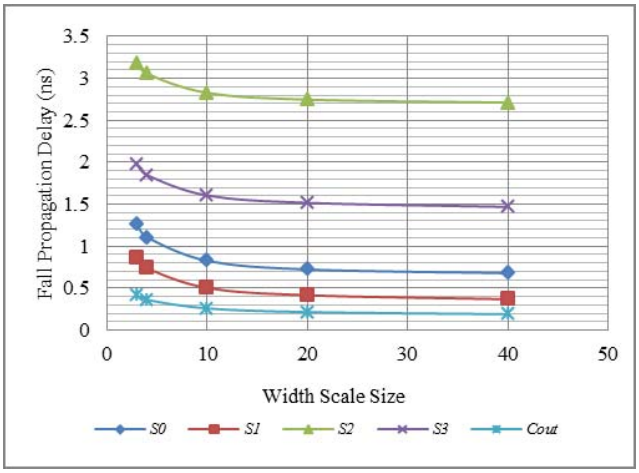


Fig. 22 (b): Comparison of Falling Propagation Delay for every Output Signal in Compound Gate

There is not much different from the Basic Logic Gate implementation results. Signal S_2 takes a lot of different to produce an output. S_3 located at the middle of the highest and the lowest consumption delay signals. Thus, overall speed of this schematic type was virtually depending on the spending delay by S_2 output generated.

Investigation on current and power consumption with specific the data within the various transistor sizes simulation been done and shown in Table 5.

Width size, (*um*) *Wp/Wn*	***Supply drain current, I_{dd} (nA)***	**Power consumption, *P=IV* (*nW*)**
0.54/0.27	130.970	235.746
0.72/0.36	131.210	236.178
1.8/0.9	133.470	240.246
3.6/1.8	139.990	251.982
7.2/3.6	160.100	288.180

Table 5: supplied current and power consumption

Similar from previous specifications results, improving sizes of the transistor will effects to used and required more current utilized and proportionally high power consumption. On the other hand, the comparison between the largest and the smallest of each transistor sizes does not indicate a lot of differences in term of the power consumption which is (*288.180ns – 235.746ns = 52.434ns*).

C. Comparison of Basic Logic Gate and Compound Gate

Both schematics design and results implementation successfully done in part A and part B. Hence, the comparison between these two types observed, for finding the differences abilities on delay propagation, power consumption and the total transistors be required on the adder.

In order to test the differences, three different characteristics for output S_3 signal result combined and all the simulation as provided in Fig. 23, but still entering on *7.2um* constant for PMOS width and *3.6um* NMOS width. Basic Logic Gate simulated using Buffer 1 and 2 Compound Gate. The delay spent, and power required been observed over Buffer 1 and buffer 6 respectively.

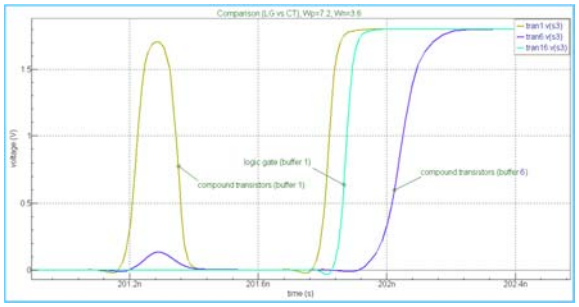


Fig. 23 (a): rise delay comparison

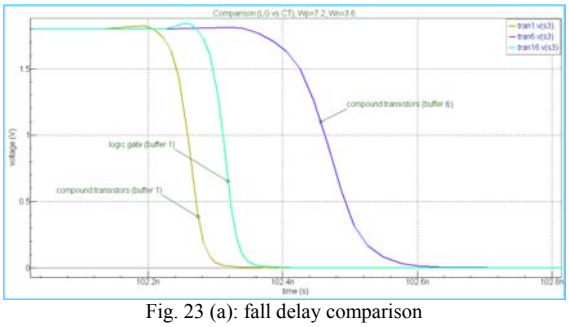


Fig. 23 (a): fall delay comparison

As seen in Fig. 23 (a), Compound Gate was a few lead in term of the speed triggered rather than Basic Logic Gate implementation even it carried a large glitch before triggered. When the design fixing to Buffer 6 which reduced a glitch, the delay extension consume more than Buffer 1 implementation.

Falling delay prove that the Compound Gate changing the state first after Basic Logic Gate with the same buffer. Then, Buffer 6 implementation on Compound Gate schematic yield the last triggered which consume the much bigger delay propagation compared to the Buffer 1 performance.

Next, the current flow, power consumption and the total number of transistors applied to the design of those three implementations arranged in table 6.

	***Supply drain current, I_{dd}* (*nA*)**	**Power consumption, *P=IV* (*nW*)**	**Number of Transistors**
Basic Logic Gate (Buffer 1)	101.14	182.052	294
Compound Gate (Buffer 1)	64.913	116.843	198
Compound Gate (Buffer 6)	160.100	288.180	

Table 6: Current, Power and the number of transistors comparison

According to the table 6, Basic Logic Gate used more transistors than Compound Gate schematic and consumed more power on the design. Compound Gate system has a certain problem regarding to the glitch occurred which disturbed the stabilization of the signals. Recently, applied larger sizes of transistor in the buffer express that high power needs surpassed Basic Logic Gate design.

V. LAYOUT DESIGN

Complete implementation of full custom layout design for Basic Logic Gate of Brent Kung Parallel Prefix Adder using Silvaco EDA Tools (Expert) done in Fig. 24. The goal of this part is to learn how to develop the entire layout without any error detected by Design Rule Check, DRC that practically look at the satisfaction of material overlapping and length between two different materials etc.

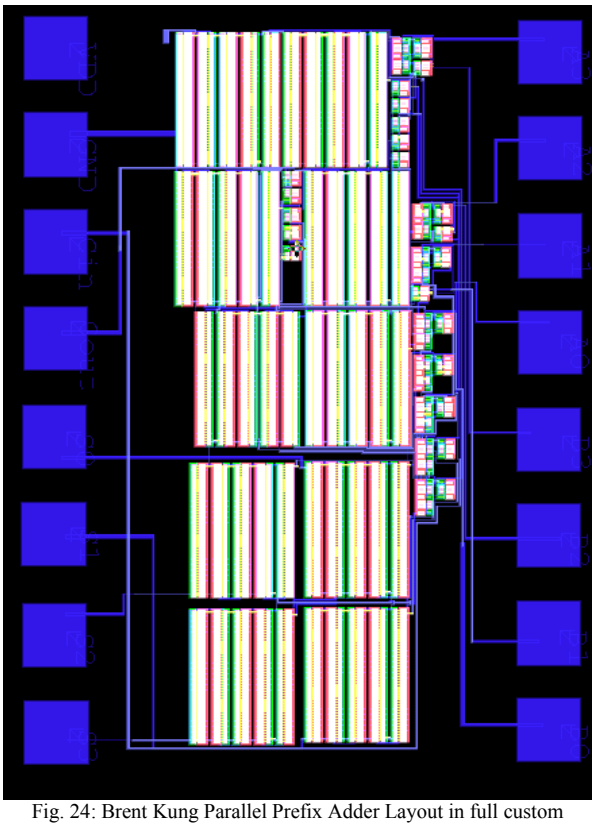


Fig. 24: Brent Kung Parallel Prefix Adder Layout in full custom

VI. CONCLUSION

As the consequences, IC design has always demanded the optimum performance on IC development in term of the speed, power consumption and the area of a single IC. This study paper found that the improvement of the gate sizing will decrease the propagation delay but, need more power

consumption and take more space for the layout area design. Furthermore, the Compound Gate design are able to reduce the complexity in the circuit with subtracting a lot number of transistors used over than Basic Logic Gate schematic done. While, directly decrease the power consumption of the adder and spend less for the delay. Nevertheless, Compound Gate performance has a concern in glitch noises produced at the output signals which force an engineer to sizing wider on the transistors at the buffer cell in order to get over the thing.

ACKNOWLEDGEMENT

This project sponsor by Universiti Teknologi MARA under Research Intensive Faculty Excellent Fund (RIF) reference no 600-RMI/DANA 5/3/RIF (371/2012).

REFERENCES

[1] Cory Merkel, David Brenner, "8-bit Parallel Prefix Adders Using Brent Kung Tree BIST", EECC730, November 2008
[2] Amos R. Omondi, Benjamin Premkumar, "Residue number systems: theory and implementation", Imperial College Press, Technology & Engineering, 2007
[3] Andrew Beaumont-Smith and Cheng-Chew Lim, "Parallel Prefix Adder Design", Department of Electrical and Electronic Engineering, The University of Adelaide,5005, Australia, 2001
[4] Kostas Vitoroulis, "Parallel Prefix Adders", Concordia University, 2006
[5] Vibhuti Dave, Erda Oruklu and Jafar Saniie, "Performance Evaluation of Flagged Prefix Adders for Constant Addition", Department of Electrical and Computer Engineering, Illilois Institute of technology, Chicago, 2006
[6] Haikun Zhu, Chung-Kuan Cheng, Ronald Graham, "Constructing Zero-deficiency Parallel Prefix Adder of Minimum Depth", Department of Computer Science and Engineering, La Jolla, California 92093, 2005
[7] VLSI Design, Technician Publications, ISBN=8184314817, 9788184314816
[8] A. P. Godse and D. A. Godse, "Digital Systems Design" - I: Technical Publications, 2008.
[9] Tien-Yu Wu, Ching-Tsing Jih, Jueh-Chi Chen and Chung-Yu Wu, "A Low Glitch 10-bit 75-MHz CMOS Vidoe D/A Converter", January 1995