

CMOS CIRCUIT SPEED AND POWER OPTIMIZATION
USING SIMPLIFIED RC DELAY MODEL

By

SUNIL KUMAR LAKKAKULA

Bachelor of Technology
Electrical & Electronics Engineering
Acharya Nagarjuna University
Guntur, Andhra Pradesh, India
2007

Master of Science
Electrical Engineering
Oklahoma State University
Stillwater, Oklahoma
2009

Submitted to the Faculty of the
Graduate College of the
Oklahoma State University
in partial fulfillment of
the requirements for
the Degree of
DOCTOR OF PHILOSOPHY
May, 2015

CMOS CIRCUIT SPEED AND POWER OPTIMIZATION
USING SIMPLIFIED RC DELAY MODEL

Dissertation Approved:

Dr. Louis G. Johnson

Dissertation Adviser

Dr. Gary Yen

Dr. Rama Ramakumar

Dr. Blayne Mayfield

ACKNOWLEDGEMENTS

First, I would like to thank my adviser Dr. Louis G. Johnson for his wonderful support, guidance and encouragement throughout my PhD program. I am honored to be his PhD student. I also thank him for providing me with financial assistance through Graduate Research Assistantship during most of my PhD program.

I would like to thank my committee members Dr. Gary Yen, Dr. Rama Ramakumar and Dr. Blayne Mayfield for serving on my committee and providing their valuable comments and feedback on my work.

I am also thankful to the Department of Campus Life and the Service-Learning Volunteer Center at Oklahoma State University for allowing me to work as a Graduate Teaching Assistant during which I have learning great community skills and volunteer skills.

I am also very grateful and thankful to the following people who played a major role in my life during my stay in the United States. Firstly my friends Rajashekhar Yaramasu, Siddarth Kota, Brian Joseph, Dr. Satyanarayana Achanta, Satish Bhattiprolu, Satyashil Shinde, Dr. Anand Govindarajan, Dr. Upasana Manimegalai Sridhar, Dr. Kumar Singarapu, Jeet Turakhia, Suresh Kumar Jayaraman, Vijayalakshmi Sethuraman, Samyukta Koteeswaran, Ram Kumar Isakki for their support and cooperation. Also, Tim Huff, Regina Henry, Ruthie Loffi, Kent Sampson, Joyce Montgomery, Marie Basler from the Department of Campus Life for their valuable support and guidance for me in

learning leadership and service skills. My labmates Dr. Julius Marpaung, Wira Mulia for their assistance in learning new technical skills. My American friends Alice Sharrock, Russ Sharrock, Janina Graves, Mason Williams, Grice's Family for their valuable friendship. My relatives in the United States for welcoming me and my wife into their homes with great love and affection.

Finally, I thank my parents Nageswara Rao Lakkakula and Lakshmi Lakkakula for their constant support and trust in me without which I have not come this far and be able to finish my PhD. I thank my brother Anil Kumar Lakkakula for his blessings from heaven. I also thank my friend and sister Surekha Bathula for her support. Last but not the least I thank my wife Mounika Tiruamalsetty for being with me and supporting me in finishing my PhD successfully.

.Name: SUNIL KUMAR LAKKAKULA

Date of Degree: May, 2015

Title of Study: CMOS CIRCUIT SPEED AND POWER OPTIMIZATION USING
SIMPLIFIED RC DELAY MODEL

Major Field: ELECTRICAL ENGINEERING

Abstract: A simplified RC delay model which is expressed explicitly in terms of transistor widths is presented to easily perform circuit analysis and quickly optimize the transistor widths for delay and power without having to do tedious layout or schematic simulation. A novel heuristic gradient descent method is used to effectively solve the optimization problem. Popular parallel prefix adders such as Brent-Kung, Skylansky and Kogge-Stone adders are modeled using the proposed simplified RC delay model and a power-delay performance comparison is done after optimization to show the ability of the model. The optimization results suggest that the Brent-Kung adder is more efficient in terms of both delay and power by having the lowest power-delay product followed by the Skylansky adder and then the Kogge-Stone adder.

TABLE OF CONTENTS

Chapter	Page
I. INTRODUCTION.....	1
1.1 Background.....	2
1.1.1 MOSFET Operation and Characteristics.....	2
1.1.2 Circuit Delay and Power.....	4
II. REVIEW OF LITERATURE.....	6
2.1 Review of device models.....	6
2.2 Review of the Method of Logical Effort.....	9
2.3 Review of the gradient descent based optimization.....	13
2.4 Review of optimization techniques based on transistor sizing.....	15
2.5 Review of parallel prefix adders and their performance.....	18
III. METHODOLOGY.....	21
3.1 Simplified RC delay Model.....	21
3.2 Power Model.....	31
3.3 Optimization Methodology.....	33
3.4 Model Validation.....	31
3.4.1 Lightly loaded case.....	36
3.4.2 Heavily loaded case.....	37
3.5 Convexity of the objective function.....	38
IV. PERFORMANCE EVALUATION & OPTIMIZATION RESULTS.....	40
4.1 Parallel prefix adders.....	40
4.2 Optimization results.....	43
V. CONCLUSION.....	46
REFERENCES.....	47
APPENDICES.....	51
Appendix A.....	51
Appendix B.....	56
Appendix C.....	61

LIST OF TABLES

Table	Page
Table 2.1 Logical effort for inputs of static CMOS gates, assuming $\gamma=2$	10
Table 2.2 Parasitic delay estimates of different logic gates.....	11
Table 3.1 Parasitic delay expressions for 2-input NAND gate.....	22
Table 3.2 Capacitance values for various processes.....	25
Table 3.3 CPU time comparison.....	38

LIST OF FIGURES

Figure	Page
Figure 1.1: Cross section of nMOS transistor.....	2
Figure 1.2: I-V characteristics of nMOS transistor.....	3
Figure 1.3: Plot defining circuit delay	4
Figure 1.4: Example RC Circuit for Elmore Delay	5
Figure 2.1: Switched-Resistor Transistor Model	8
Figure 2.2: Piecewise Linear Model	8
Figure 2.3: FO4 – Inverter driving four identical inverters	11
Figure 2.4: Logic network consisting of three two-input NAND gates.....	12
Figure 3.1: 2-input NAND gate	22
Figure 3.2: Channel Propagation Capacitance.....	24
Figure 3.3: Diffusion Capacitance with diffusion contact	26
Figure 3.4: Diffusion Capacitance with no diffusion contact	26
Figure 3.5: Diffusion Capacitance with shared diffusion contact.....	27
Figure 3.6: 2x4 Decoder circuit with enable signal	29
Figure 3.7: 3-input NAND gate and inverter	29
Figure 3.8: OF Surface when minimizing delay.....	34
Figure 3.9: Four inverter chain with a light load and no C_{wire}	36
Figure 3.10: Model Vs SPICE power-delay plot for lightly loaded case	36
Figure 3.11: Four inverter chain with a heavy load and large C_{wire}	37
Figure 3.12: Model Vs SPICE power-delay plot for heavily loaded case	37
Figure 4.1: 16-bit Brent-Kung adder	42
Figure 4.2: 16-bit Skylansky adder.....	42
Figure 4.3: 16-bit Kogge-Stone adder	42
Figure 4.4: Power-delay comparison plot for 32-bit adder case	44
Figure 4.5: Power-delay comparison plot for 64-bit adder case	44
Figure 4.6: Power-delay comparison plot for 128-bit adder case	45

CHAPTER I

INTRODUCTION

A major concern in VLSI circuit design is the delay and power dissipation of the circuit. Circuit optimization based on transistor sizing is one very useful method to optimize the circuit for speed and power dissipation to achieve better needed circuit performance. Hence a model which can be expressed explicitly in terms of transistor widths is required to effectively analyze the effects of changing the transistor widths on circuit speed and power to allow flexible trade-off. Using RC transistor models to model and simulate a CMOS circuit is popular because of its simplicity and small computational burden compared to complex and computationally expensive BSIM models that are used in SPICE [1] simulators. This dissertation primarily presents the implementation of a simplified RC delay model to quickly estimate the CMOS circuit delay and its application in the circuit speed and power optimization by correctly sizing the transistor widths. In many cases there are several possible critical paths for the circuit to be optimized. There is no simple solution for optimum transistor sizes when multiple paths are optimized simultaneously. For better accuracy numerical techniques are needed to solve the optimization problem. It is much easier to use the numerical techniques with our simplified RC delay equations rather than SPICE for the following reasons: the delays are explicit functions of transistor widths making it obvious what happens when any width is changed; the width dependence of the delays is smooth so that simple numerical methods can be used to quickly find the optimum transistor widths; if the optimum widths found do not meet the specification, then there is probably no feasible solution and the specification must be changed.

Since addition forms the basis for many processing operations and adder circuits are of great interest to digital system designers, popular prefix adders such as Brent-Kung[2], Skylansky[3] and Kogge-Stone[4] adders are modeled using the proposed simplified RC delay model in this dissertation and a power-delay performance comparison was done to show the ability of the method in quickly optimizing the complex circuits like adders.

1.1 Background

Now, let's look at some back ground to understand how the MOS transistors work and how the circuit delay and power are calculated.

1.1.1 MOSFET Operation and Characteristics

A Complementary Metal Oxide Semiconductor (CMOS) circuit consists of two types of transistors, nMOS transistor and a pMOS transistor. nMOS is called n-channel MOSFET and the majority charge carriers are electrons. pMOS is called p-channel MOSFET and the majority charge carriers are holes. The design and operation of an nMOS transistor is discussed below.

In nMOS, the source and drain are n-type regions and the body is p-type region. With sufficient positive voltage at the gate, holes from the p-type body are driven away from the gate, forming an n-type channel between the p-type body and the oxide. This channel extends between the source and the drain, when a voltage is applied between the drain and the source current is conducted through it. Figure 1.1 shows the cross section of an nMOS transistor without and with a channel formation.

The operation of a MOSFET is categorized into three different regions or modes. They are Cut-off, Ohmic or linear and Saturation. Let us discuss the operation regions of an nMOS transistor.

Cut-off region ($V_{GS} < V_{th}$): In this region the gate to source bias V_{GS} is less than the threshold voltage V_{th} and there is no significant conduction between drain and source. Hence the transistor can be considered as turned off in this region and the drain current is equal to zero.

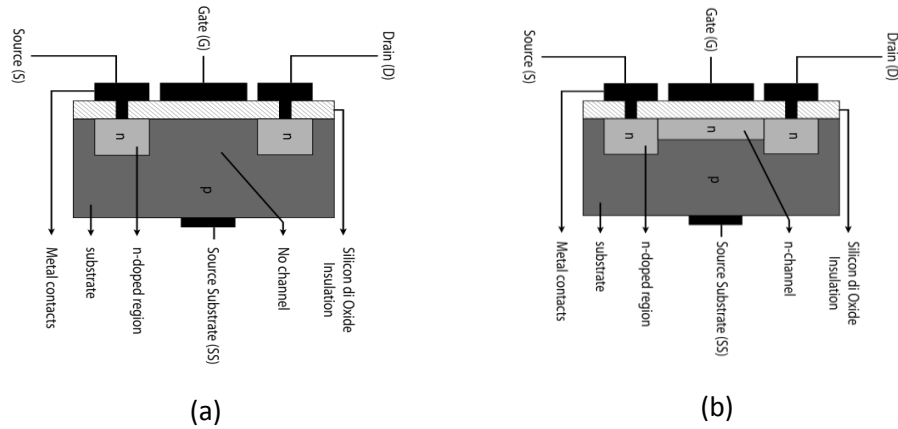


Figure 1.1: Cross section of nMOS transistor (a) without channel (b) with channel

Ohmic region ($V_{GS} > V_{th}$ and $V_{DS} < V_{GS} - V_{th}$): In this region the gate to source bias is greater than the threshold voltage and the drain to source bias is less than the gate voltage $V_{GS} - V_{th}$. The transistor is turned on and the current conduction takes place between drain and source. In this region as the drain to source voltage increases the drain current (I_D) also increases.

Saturation region ($V_{GS} > V_{th}$ and $V_{DS} > V_{GS} - V_{th}$): In this region the gate to source bias is greater than the threshold voltage and the drain to source bias is greater than the gate voltage $V_{GS} - V_{th}$. The transistor is turned on and the current conduction takes place between drain and source. In this region as the drain to source bias voltage increase there is almost no significant increase in the value of the drain current.

The current vs voltage (I-V) characteristics of an nMOS transistor are shown in Figure. 1.2.

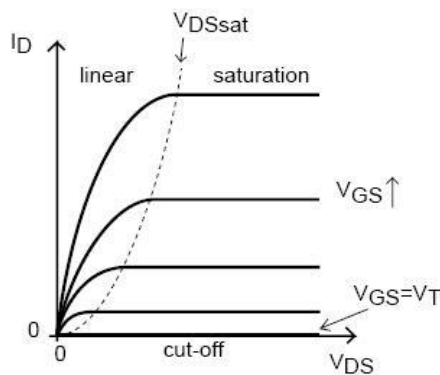


Figure 1.2: I -V characteristics of nMOS transistor

In pMOS, the source and drain are p-type regions and body is n-type. The working of a pMOS transistor is similar to nMOS but with a negative voltage applied at its gate terminal.

1.1.2 Circuit Delay and Power

Delay: The usual definition of the circuit delay is the time difference between the half- V_{dd} point of the circuit's input voltage waveform and the half- V_{dd} point of the circuit's output voltage waveform.

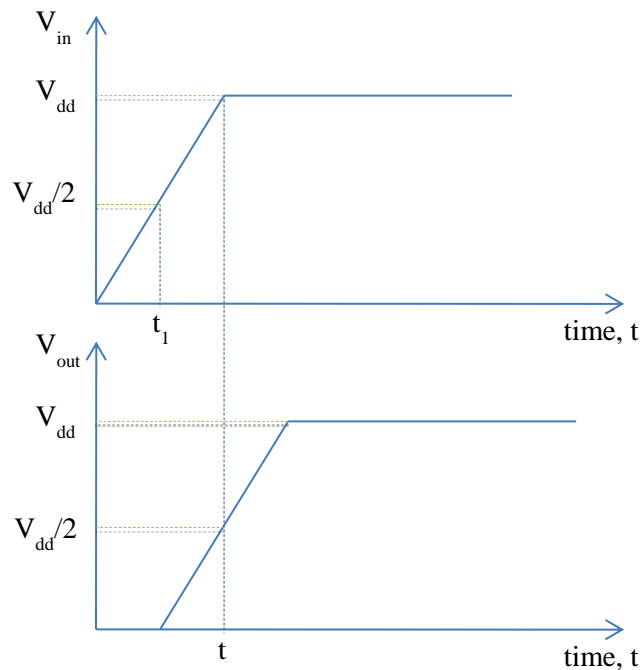


Figure 1.3: Plot defining circuit delay

From the above figure, delay = $t - t_1$. To determine the delay of a circuit accurately the exact waveforms of the input and output are required. The process of determining the exact waveforms of input and output is computationally expensive. Hence Elmore[5] defined the delay at node e , t_{de} , as the centroid of the output impulse response $e'(t)$ curve which can be found independently of the exact waveforms. It is given by the expression shown in Eq. (1.1). It is also called the first moment of the impulse response. Elmore delay is a good approximation for delay in tree RC networks.

$$Delay(t_{de}) = \int_0^{\infty} t e'(t) dt \approx \sum_k R_{ek} C_k \frac{\Delta V_k}{\Delta V_e} \quad (1.1)$$

where R_{ek} is defined as the resistance of the path to the V_{dd}/Gnd node shared by node e and node k . C_k is the k^{th} node capacitance. ΔV_k is the voltage difference between V_{dd}/Gnd node and the initial voltage at the node k . $\Delta V_k = V_k(0)$ for falling voltages and $\Delta V_k = V_{dd} - V_k(0)$ for rising voltages. Similarly ΔV_e is the voltage difference between V_{dd}/Gnd node and the initial voltage at the node e . $\Delta V_e = V_e(0)$ for falling voltages and $\Delta V_e = V_{dd} - V_e(0)$ for rising voltages. When all of the node voltages start at the same voltage then the last voltage fraction drops out. For example consider the following RC network shown in Figure 1.3 and assume that all of the node voltages start at the same voltage.

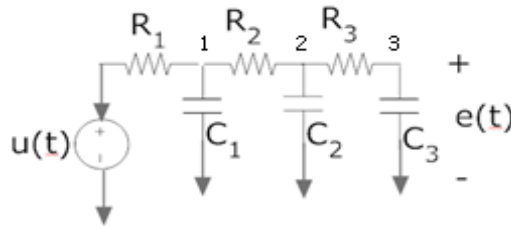


Figure 1.3: Example RC Circuit for Elmore Delay

The Elmore delay at node 3 is given as $t_{d3} = R_1 C_1 + (R_1 + R_2) C_2 + (R_1 + R_2 + R_3) C_3$

Power: The usual definition of power dissipation is the amount of heat energy dissipated in unit time.

The average power dissipation P_{avg} is defined as shown in Eq. (1.2)

$$P_{avg} = \frac{1}{T} \int_0^T I_{V_{dd}}(t) V_{dd} dt \quad (1.2)$$

where $I_{V_{dd}}(t)$ is the power supply current and V_{dd} is the supply voltage.

The power dissipative components in CMOS circuits consist of off-state leakage power, dynamic power due to charging and discharging of node capacitances, short-circuit power, switching power due to parasitic capacitances and glitch power due to unequal arrival of signals. Estimating the power dissipation with accurate models is difficult and computationally expensive.

CHAPTER II

REVIEW OF LITERATURE

2.1 Review of device models

Prior to late 1960s, performance estimation techniques were computationally simple as the number of transistors on a single chip were small and used physical models based upon the approximate modeling of the physical phenomena within a transistor. As the number of transistors increases on a single chip, the complexity of the circuit also increases making the physical models inadequate for quantitative analysis as they are computationally expensive. A circuit simulator, like SPICE [1], handles the complex nonlinear physical models of the transistor and solves the whole circuit as a big matrix to get the node outputs. Usually no more than a few thousand transistors may be simulated in a reasonable amount of computation time. Initial attempts to create transistor models for the large circuit simulation used empirical models to estimate the delay and power. These models are entirely based upon curve fitting, using whatever functions and parameters that best fit the measured data. The use of empirical models limits the type of circuits that can be modeled. The disadvantages of using empirical models are lack of error control in the resulting models and difficulty in relating the performance values back to the circuit elements. To solve this problem, a combination of physical and empirical models is used to analyze the circuit's performance. In the Shockley square law model [6] the drain current I_D is expressed as shown in Eq. 2.1.

$$I_D = \begin{cases} 0, & V_{GS} \leq V_{TH} & \text{Cutoff - region} \\ K\{(V_{GS} - V_{TH})V_{DS} - 0.5V_{DS}^2\}, & V_{GS} > V_{TH}, V_{DS} < V_{DSsat} & \text{Ohmic - region} \\ 0.5K(V_{GS} - V_{TH})^2, & V_{GS} > V_{TH}, V_{DS} \geq V_{DSsat} & \text{Saturation - region} \end{cases} \quad (2.1)$$

Where $V_{DSsat} = V_{GS} - V_{TH}$ is drain saturation voltage and V_{TH} is threshold voltage. K is a drivability factor and is given by $\mu \left(\frac{\epsilon_{ox}}{t_{ox}} \right) \left(\frac{W}{L} \right)$, where μ denotes an effective mobility, ϵ_{ox} a dielectric constant of a gate oxide, t_{ox} a gate oxide thickness, W a channel width and L channel length. In the alpha power law model [7], which is an extension of Shockley's square law model in the saturation region the drain current in the saturation region is modeled as shown in Eq. 2.2.

$$I_D \propto (V_{GS} - V_{TH})^\alpha, \quad V_{GS} > V_{TH}, V_{DS} \geq V_{DSsat} \quad \text{Saturation - region} \quad (2.2)$$

It includes the velocity saturation effects to predict the circuit behavior in the sub-micrometer regime which the original Shockley square law model does not take into account in case of deep submicron processes. The value of α is calculated directly from the measured data and usually lies between 2 and 1. For deep submicron processes the value of α is close to 1. By using this nonlinear device model the circuit simulation is computationally expensive and there is no closed form expression for determining the circuit delay which requires the waveforms for the input and output.

To further simplify the analysis problem and to improve the simulation speed, all non-linear elements were approximated by appropriate linear elements by performing small signal analysis. In the Switched-Resistor model used in the IRSIM circuit simulator [8], the transistor is modeled as a voltage controlled switch connected to a series resistance as shown in Figure 2.1.

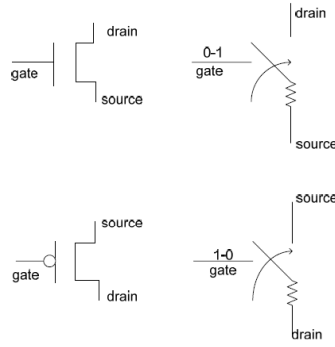


Figure 2.1: Switched-Resistor Transistor Model

Hence a MOS circuit can be considered as an RC network and the delay through a circuit path can be computed by using the simple expression given by Elmore in Eq. (1.1). By modeling the transistor as a switched-resistor the delay and power analysis of a circuit can be done with much less computational burden and circuits with several hundreds of thousands of transistors can be simulated in a reasonable amount of time. Though this model is less accurate, as it approximates a non-linear transistor as a linear resistor but is very helpful in doing the circuit analysis with less computational burden and to quickly estimate the delay without needing to determine the input and output waveforms.

Reference [9] discusses a piecewise linear device model which is a modified switched resistor model with better accuracy. In this model the transistor is approximated as an open switch when the transistor is in the cutoff region, as a linear resistor when operating in the ohmic region and as a current source that is a function of input voltage when operating in the saturation region.

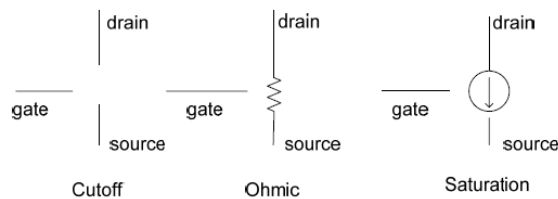


Figure 2.2: Piecewise Linear Model

The drain current I_D using this model is given by Eq. 2.3.

$$I_D = \begin{cases} 0, & V_{GS} < V_{TH} & \text{Cutoff - region} \\ a \cdot G \cdot V_{DS}, & V_{GS} > V_{TH}, V_{DS} < V_{DSsat} & \text{Ohmic - region} \\ G_m \cdot (V_{GS} - V_{TH}), & V_{GS} > V_{TH}, V_{DS} > V_{DSsat} & \text{Saturation - region} \end{cases} \quad (2.3)$$

This model also includes the input slope, effects due to short circuit current and velocity saturation. It also uses a linearized BSIM3 capacitance model. As the above includes all the major physical phenomena that are important in submicron and deep submicron regimes it is more accurate than the above discussed models but using this model is still computationally expensive and it also cannot produce a closed form expression to determine the circuit delay and hence requires input and output waveforms.

Hence using RC device models will be more useful for fast circuit analysis and efficient circuit optimization as the circuit delay can be defined as a closed form expression without needing the exact input and output waveforms.

2.2 Review of the Method of Logical Effort

Estimating the circuit delay early in the design process is very useful and can save a lot of time in designing a circuit meeting the required design specifications. As discussed earlier in section 2.1, using complex device models would make the delay estimation very difficult as there is no closed form solution for estimating the delay easily and quickly hence requiring the circuit layout simulation to be done to achieve the input and output waveforms to be able to estimate the delay. With this approach the time to successfully design a specific circuit would take very long to meet the required design specifications. The method of logical effort [10] is an easy way to estimate delay in CMOS circuits without requiring to do a tedious layout simulation and also allows early modifications to the circuit design to achieve the greatest speed or to meet any delay constraints by comparing delay estimates of different logic structures.

The delay incurred by a logic gate is comprised of two components, a fixed part called the parasitic delay p and a part that is proportional to the load on the gate's output, called the effort delay or stage effort f . The total delay d measured in units of τ , is the sum of the effort and parasitic delays and is given by Eq. 2.4

$$d = f + p \tag{2.4}$$

where τ is the delay of an inverter driving an identical inverter with no parasitics. The effort delay is given as $f=gh$ where g is the logical effort and h is the electrical effort given by C_{out}/C_{in} . Hence the delay through a single logic gate is

$$d=gh+p \tag{2.5}$$

Logical effort is defined so that an inverter has a logical effort of 1. The logical effort of any other logic gate tells how much worse it is at producing output current than is an inverter, given that each of its inputs may present only the same input capacitance as the inverter. The table 2.1 below shows the logical effort values for different logic gates with different inputs.

Gate Type	Number of inputs					
	1	2	3	4	5	n
Inverter	1					
NAND		4/3	5/3	6/3	7/3	(n+2)/3
NOR		5/3	7/3	9/3	11/3	(2n+1)/3
Multiplexer		2	2	2	2	2

Table 2.1: Logical effort for inputs of static CMOS gates, assuming $\gamma = 2$ [6]

where γ is the ratio of an inverter's pull-up transistor width to pull-down transistor width.

The parasitic delay of a logic gate is fixed and is given as multiples of the parasitic delay of an inverter denoted as p_{inv} and is typically 1.0 delay units. The table 2.2 shows crude estimates of

parasitic delay for a few logic gates. Though these not very accurate but is convenient for hand analysis.

Gate Type	Parasitic Delay
Inverter	p_{inv}
n-input NAND	np_{inv}
n-input NOR	np_{inv}
n-way multiplexer	$2np_{inv}$

Table 2.2: Parasitic delay estimates of different logic gates

Let's look at how the delay of the logic gate can be estimated using the method of logical effort.

For example consider a fanout-of-4 (FO4) inverter as shown in figure 2.3

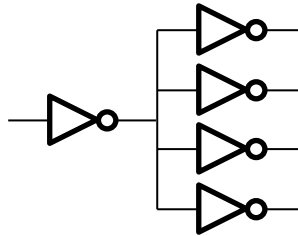


Figure 2.3: FO4 – An inverter driving four identical inverters

Because each inverter is identical, $C_{out}=4C_{in}$, so $h=4$. Since the logical effort g of an inverter is 1, the effort delay $f=gh=1 \times 4=4$. The parasitic delay for an inverter is $p=p_{inv}=1$. Hence the delay of an inverter is given by $d=f+p=4+1=5.0$ delay units.

The method of logical effort can be extended to multistage logical networks and it also can help reveal the best number of stages in a multistage network to obtain the least overall delay. The path logical effort G is given by $G=\prod g_i$ and the path electrical effort H is given by $H=C_{out}/C_{in}$ where C_{in} and C_{out} refer to the input and output capacitances of the path. In the case of estimating the path delay, the branching effort b at the output of a logic gate needs to be taken into consideration to account for the fanout within a network. Hence the path branching effort B is

given as $B = \prod b_i$. Now the path effort delay F can be defined as $F = GBH$ and the path parasitic delay P is defined as $P = \sum p_i$. Having known all the terms, the path delay is given as

$$D = F + P \quad (2.6)$$

In case of an N-stage logic network, the path delay is found to be the least when each stage in the path bears the same stage effort. i.e $f_{opt} = g_i h_i = F^{1/N}$. Hence the minimum delay achievable along an N-stage logic network path is

$$D_{opt} = NF^{1/N} + P \quad (2.7)$$

Let's look at how the path delay in a multistage network can be estimated using the method of logical effort. For example consider the circuit as shown in figure 2.4

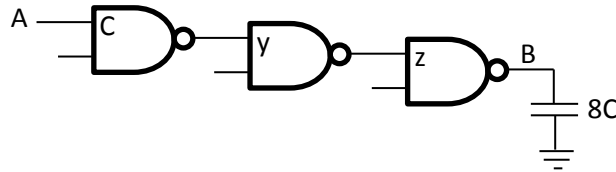


Figure 2.4: Logic network consisting of three two-input NAND gates

From the above discussion on multistage logic networks, to computer the path delay from A to B for the above circuit shown in figure 2.4

$$G = \prod g_i = (4/3)^3; H = 8C/C = 8; B = \prod b_i = 1; F = GBH = 18.96; P = \sum p_i = 3(2p_{inv}) = 6$$

Hence least path delay is $D_{opt} = NF^{1/N} + P = 3(18.96)^{1/3} + 6 = 14.0$ delay units. Now let's look at how the transistors to be sized along the path to achieve this least delay. Since $f_{opt} = 18.96^{1/3} = 8/3$, now $g_3 h_3 = 8/3$; $h_3 = 8/3 g_3$; $8C/z = 8/3 g_3$; $z = 8C(g_3)(8/3)$; $z = 8C(4/3)(8/3) = 4C$. Similarly, $y = z(4/3)(8/3) = 2C$.

Though the method of logical effort is simple and effective in quickly estimating the circuit delay, it allows only fixed p transistor width to the n transistor width ratios γ when determining the

transistor sizes while a more efficient design is possible when the p and n transistor widths can be chosen irrespective of the other. Also, when using the method of logical effort, there is no easy way to include the interconnect wire capacitance in estimating the circuit delay and hence makes it a less accurate method.

In this dissertation we present a simplified RC delay model that can easily take into account the interconnect wire capacitance in estimating the path delays and also allows the choice of p and n transistor widths irrespective of the other to be able to design a more efficient circuit.

2.3 Review of the gradient descent based optimization

Optimization is useful in achieving the minimum or maximum of an objective function (OF) by determining optimum values for the decision variables. In most cases it is required to minimize the function. When the OF is continuous w.r.t its decision variables, gradient based methods are very useful as the method tells clearly which direction to go in order to find a better OF value faster. In case of gradient descent method, one should take steps proportional to the negative of the gradient of the function at the current point to obtain a minimum OF value. Gradient descent method is useful in finding the local minimum of the function but does not guarantee that the solution found is a global optimum except when the OF is a convex function. When the OF is not continuous, the gradient of the function cannot be determined at all the values of the decision variables and hence direct search methods like particle swarm, leapfrogging can be used to optimize the function without requiring to evaluate the gradient of the function. Since the objective function in our analysis is continuous, gradient descent method can be used for optimization.

Let's look into the details of how the gradient descent method is implemented in general. Let $F(x)$ be a multivariable objective function that is differentiable in the neighborhood of any given point

x_n in the decision variable space then the new point x_{n+1} satisfying $F(x_n) \geq F(x_{n+1})$ in the direction of negative gradient of $F(x)$ for a small value of δ_n is defined as

$$x_{n+1} = x_n - \delta_n (F'(x_n)) \quad (2.8)$$

It takes several iterations to obtain the best possible minimum value for $F(x)$, and the value of δ_n can change in each iteration. The best value of δ_n can be chosen via a line search which is again an iterative process using the gradient descent method to obtain the optimum value for $\delta_n = \delta_n^*$ that minimizes $F(x_n)$ in the negative direction of gradient of $F(x_n)$. Hence the Eq. 2.8 is now defined as

$$x_{n+1} = x_n - \delta_n^* (F'(x_n)) \quad (2.9)$$

The Eq. 2.9 is the Cauchy method [11] popularly known as the steepest descent method. The convergence of the algorithm is decided using a small value ε usually in the order of 10^{-5} such that $F'(x_n) \leq \varepsilon$ and/or $\Delta x = x_{n+1} - x_n \leq \varepsilon$. Most often a fixed small value of $\delta_n = \delta$ is assumed in each iteration to reduce the convergence time. Though this method is simple and has the optimal property of finding the best minimum OF value it performs poorly in terms of convergence and may not converge in a reasonable amount of time when the OF is bumpy and has v-shaped minima, which is the case when minimizing the circuit delay in our analysis.

Reference [12] presented formulae to calculate the step-size δ_n in each iteration instead of doing a line search that helped in much faster convergence of the conventional steepest descent algorithm.

$$\delta_n = \frac{s_{n-1}^T y_{n-1}}{\|y_{n-1}\|_2^2} \quad (2.10)$$

or

$$\delta_n = \frac{\|s_{n-1}\|_2^2}{s_{n-1}^T y_{n-1}} \quad (2.11)$$

where $s_{n-1} = x_n - x_{n-1}$ and $y_{n-1} = F'(x_n) - F'(x_{n-1})$. Though this method is better than Cauchy method, it still takes significant amount of time to converge when the OF surface is bumpy and has v-shaped minima. Hence there is a need for a better and faster way of implementing the gradient descent algorithm to achieve faster convergence when the OF surface is bumpy and has v-shaped minima with satisfactory results compared to the above discussed methods.

2.4 Review of optimization techniques based on transistor sizing

Literature suggests that there are several techniques that are used in improving circuit performance. One such technique is circuit optimization. Again circuit optimization can be done using various methods such as transistor sizing, transistor reordering, transistor tapering etc. However, transistor sizing is considered to be the simplest and most effective method in CMOS circuit optimization. Previous research shows many such attempts were made to optimize the circuit for speed and power using transistor sizing. Some have used RC transistor models to approximate the circuit delay/power, while some have used non-RC models to approximate the circuit delay/power. Let's review some of them below.

One of the early tools used for circuit speed optimization based on transistor sizing is discussed in [13] and is called SLOP (Switch Level Optimization) which uses an RC tree approximation for the circuit to estimate the circuit path delay. The total delay of path i including the transistor is given by the sum of all the delay contributions from each transistor in that path.

$$d_i = t_1 + t_2 + \dots + t_j + \dots + t_n \quad (2.8)$$

To minimize d_i w.r.t the transistor width W_j , evaluate $\frac{\partial d_i}{\partial W_j} = 0$ and solve for W_j^* which is the minimum delay contribution width of transistor j in path i. The critical path delay $d_c = (d_i)_{\max}$ determines the speed of the circuit. The optimization is done in three main stages:

- a) Initialization – to record the initial delays of d_i and find the critical delay by the first simulation
- b) Global test – to establish delay, gain and device matrices. Delay matrix is established by each time changing one transistor width with one step, simulating and recording delays of each path. The gain matrix is established by calculating the partial derivatives of d_i with respect to the transistor width W_j . The device matrix is established by recording device number with non-zero gain.
- c) Critical path test – to only test devices in the critical delay column.

Optimization done by this method is complicated and time consuming as it requires to change only one transistor width at a time and determine the delays of each path. Proper path balancing may not be achieved with this method hence there will be a problem of glitching.

[14] also uses an RC model to estimate the delay but the transistor sizing is done to minimize the area subject to a delay constraint. In this the optimization problem is considered as a convex programming problem. Convex optimization is considered to be efficient because any local minimum solution will also be the global minimum solution. In this case the minimization is done on one path at a time but there will be a need to minimize multiple path delays at a time in most of the cases which is when the optimization problem becomes non-convex and there will be no simple solution for the optimization problem.

[15] uses a different approach in using the RC model for the circuit area/power optimization. It tries to place large size transistors and route to meet the delay specification of the circuit and then the interconnect wire length is extracted. The circuit is optimized for area/power subject to the delay constraint and as a result the transistor sizes decrease creating spaces in the layout and the interconnect wires can be re-routed utilizing these spaces in such a way to reduce the overall wire

length which in turn reduces the wire capacitance. In this the optimization is achieved majorly by minimizing the interconnect wire length.

[16] also uses an RC model and the transistor sizing is done for optimizing the power-delay product rather than only delay or power of a CMOS circuit. The delay calculations are done using Elmore's RC delay model. A loading coefficient $\alpha=C_L/C_d$ is considered as the decision variable for the power-delay product (PDP) objective function optimization. CL is the load capacitance at a node and Cd is the circuit internal capacitance at that node. The PDP is expressed in terms of α and the optimum α_{opt} can be determined from solving the expression $\frac{\partial PDP}{\partial \alpha} = 0$. As the model is not explicitly expressed in terms of transistor widths it is difficult to analyze the effects of changing the transistor widths on the power delay product.

[17] also uses RC Elmore delay model. In this both the transistor widths and lengths are used as decision variables for the optimization problem. The optimization problem was solved as a multi-objective optimization. The transistor widths and lengths were modified to match all the circuit path delays with the critical path delay to achieve path balancing to avoid glitching and at the same time minimize circuit power consumption. In achieving optimum results the transistor lengths have to be increased, which results in both increased gate capacitances and area. To reduce this negative influence of the increased transistor lengths, two alternate ways were proposed: twin transistors and merged transistors. In order to equalize all the path delays w.r.t the critical path, every path requires individual optimization. In this the delay is not essentially minimized but is made equal to the critical path delay for all the circuit paths to achieve path balancing.

[18],[19],[20],[21] uses complex and non-linear models to minimize delay and power. Though these models are accurate compared to simple RC models, circuit optimization for speed and

power using these models is difficult and is not very efficient because of the complexity of the models.

[22],[23] discusses different techniques like transistor reordering and transistor tapering to optimize the circuit for delay and power, however we prefer to stick with transistor sizing techniques as it is the simple and effective to optimize the circuit for speed and power.

2.5 Review of parallel prefix adders and their performance

Binary addition is one of the most often used arithmetic operations on microprocessors. A large variety of algorithms and implementations have been proposed for binary addition. When high operation speed is of great importance, parallel-prefix adders such as Brent-Kung[2], Skylansky[3] and Kogge-Stone[4] adders are commonly used. Any decrease in the delay will directly relate to an increase in the adder throughput. The primary requirements for any adder are that it should be fast and efficient in terms of power consumption and chip area. When the adder size is small, the above mentioned parallel prefix adders show similar performance in terms of delay and power but as the adder size gets large ($N > 16$ bit), the difference in their performance becomes significant due to the difference in their implementation.

The Skylansky adder presents a least depth prefix network at the cost of increased fan-out at certain nodes. The fan-out increases exponentially as the adder size increases. The Kogge-Stone adder has optimal depth and low fan-out but produces massively complex circuit and also accounts for a large number of interconnects when the adder size is big. The Brent-Kung adder has the advantage of minimal number of circuit nodes, which yields in reduced area but this implementation requires maximum depth that causes an increase in the latency when compared to the other structures. The above analysis is mostly true when fixed transistor sizes or minimum sized transistors are used. Hence it is useful to learn the performance of these adders for different adder sizes with optimum transistor widths to allow a better choice between these adders in the

design of high speed microprocessors. Since the process of designing the adders and their optimization is a time taking process, it would be very useful to use a simple device model like RC transistor model and estimate the delay and power without requiring to draw the layout.

[24] presents the design and performance comparison of various high-speed adders using CMOS and Transmission gate technology. This work showed that the adders designed with transmission gate has much less power and delay than those with CMOS gates. But the focus of this dissertation is the design with CMOS gates. This work requires the adder layout/schematic to be done to do the analysis. Moreover, there is no information on optimum transistor widths being used in their study of adder comparison.

[25] presents the design and performance of any parallel prefix adders by using alternating odd and even cells by eliminating unwanted buffers between the cells from two stages in the conventional design. Similar approach will be used in this dissertation to eliminate the unwanted buffers in the adder design. Even this approach requires the adder layout/schematic to be done to do the analysis and does not discuss the optimization of the adders for optimum delay or power.

[26] also presents a performance comparison of various high-speed adders using Xylinx ISE for simulation and synthesis without discussing the optimization of the adders for improved performance.

[27] presents a comparison of adder performance with radix-4 and radix-2 in case of a 32 bit parallel prefix adder. This work shows that the adder implementation with radix-4, Sparse-4 has reduced delay with a minor increase in power than with radix-2 implementation. Also, this method requires the adder analysis using the layout/schematic and does not discuss the use optimum transistor widths in the design.

This dissertation presents the adder analysis without having to do the tedious layout/schematic simulation by using a simplified RC delay model and also presents a performance comparison of

different high-speed adders with optimum transistor widths for different delay and power constraints.

CHAPTER III

METHODOLOGY

3.1 Simplified RC Delay Model

The simplified RC delay model [28] consists of parasitic delay t_{dP} that arises due the circuit's internal parasitic resistances and capacitances and is estimated using Elmore delay technique, and also consists of effort delay t_{dF} that arises due to the capacitive load on the circuit's output node.

Hence the delay (t_d) of any circuit path is given by

$$t_d = t_{dP} + t_{dF} \quad (3.1)$$

$$t_{dF} = R_{out} C_{load} \quad (3.2)$$

When there is no load on the output node i.e $C_{load}=0$ then $t_d = t_{dP}$ and when there is a load on the output node then $t_d = t_{dP} + t_{dF}$. The parasitic delay and output resistance R_{out} are determined by the topology of the circuit.

To discuss the implementation of the simplified RC delay model in quickly estimating the circuit or gate delay, consider a 2-input NAND gate as shown in Figure 3.1 as an example circuit. The transistor level diagram is shown on the left and the corresponding stick diagram is shown on the right in Figure 3.1. The circuit has four transistors and two parasitic node capacitances C_Y and C_I . C_Y is same as the C_{out} which is the parasitic capacitance on the output node Y.

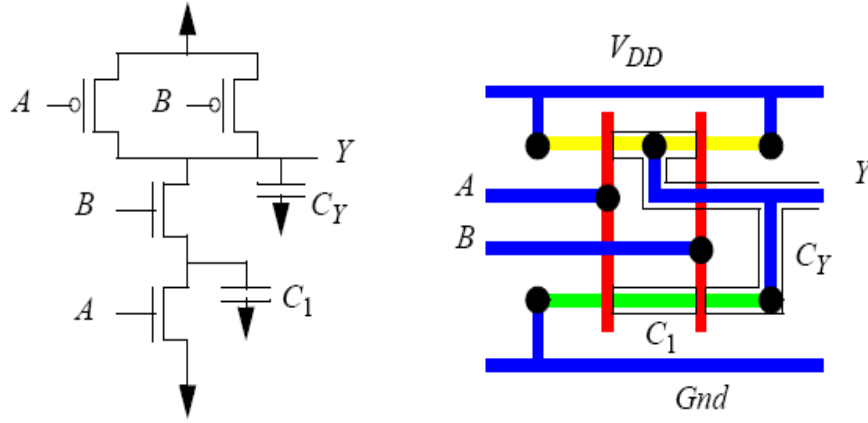


Figure 3.1: 2-input NAND gate a) Transistor diagram b) Stick diagram

The parasitic delay and the output resistance of this circuit are determined for different delay paths based on Elmore delay calculations as shown in Table 3.1.

A	B	Y	t_{dP}	R_{out}
\uparrow	1	\downarrow	$R_{nA}C_1 + (R_{nA} + R_{nB})C_{out}$	$R_{nA} + R_{nB}$
\downarrow	1	\uparrow	$R_{pA}C_1 + R_{pA}C_{out}$	R_{pA}
1	\uparrow	\downarrow	$(R_{nA} + R_{nB})C_{out}$	$R_{nA} + R_{nB}$
1	\downarrow	\uparrow	$R_{pB}C_{out}$	R_{pB}

Table 3.1: Parasitic delay expressions for 2-input NAND gate

Where R_{pA} , R_{nA} , R_{pB} , R_{nB} are the channel resistances of the transistors. From the above table the delay paths are determined assuming only one input is changing at a time and other inputs are not switching. In the case of the 2-input NAND gate when one input is switching the other input has to be '1' to be able to determine the delay of the path from the changing input to the output because if the non-switching input is '0' there will be no change in the output even when the switching input is falling or rising, hence there is no question of delay in that case.

Estimation of channel resistance in terms of Unit R:

The major goal of the simplified RC delay model is to see the effect of changing transistor widths without having to do detailed layout and simulation. The channel resistances of individual transistors are inversely proportional to the transistor channel widths and are proportional to the transistor channel lengths and is given by

$$R_X = R_s \frac{L_X}{W_X} \quad (3.3)$$

Where R_X is the channel resistance of transistor X, R_s is the transistor sheet resistance which is a process constant, L_X is the transistor channel length and W_X is the transistor channel width. In our design the transistor channel length is chosen to be the minimum feature size of the process which is L_{min} . Hence the channel resistances for the nFET and pFET transistors are given as shown below. The subscripts 'n' and 'p' refers to the n-channel and p-channel respectively.

$$R_{nX} = R_{sn} \frac{L_{min}}{W_{nX}} \quad (3.4)$$

$$R_{pX} = R_{sp} \frac{L_{min}}{W_{pX}} \quad (3.5)$$

To simplify calculations of channel resistance and make them process independent, lets define channel resistance as a multiple of R , the channel resistance of a minimum size nFET.

$$R \equiv R_{sn} \frac{L_{min}}{W_{min}} \quad (3.6)$$

$$R_{nX} = R_{sn} \frac{L_{min}}{W_{nX}} = \frac{W_{min}}{W_{nX}} R \quad (3.7)$$

$$R_{pX} = R_{sp} \frac{L_{min}}{W_{pX}} = \frac{W_{min}}{W_{pX}} 2R \quad (3.8)$$

Note that there is an extra factor of 2 for pFET channel resistance to account for the significant difference between the n-channel and p-channel sheet resistance. For 0.18um technology $R = 5K\Omega$ approximately.

Estimation of parasitic capacitances in terms of Unit C:

The transistor gate capacitance C_{in} per unit width W has stayed relatively constant if all the three dimensions of the gate scale by the same factor and is equal to $2fF/\mu m$ approximately. To simplify calculations of parasitic capacitance and make them process independent, lets define Unit Capacitance C given as

$$C \equiv \left(\frac{C_{in}}{W}\right)W_{min} \tag{3.9}$$

For TSMC 0.18um technology $C = 0.89fF$ approximately.

Hence capacitance at node i in terms of Unit C is given as

$$C_i \equiv \frac{C_i}{\left(\frac{C_{in}}{W}\right)W_{min}} C \tag{3.10}$$

The internal parasitic capacitance at a node is separated into propagation channel capacitance (C_{prop}) and diffusion capacitance (C_{diff}).

The propagation channel capacitance C_{prop} for the n-channel transistor and in terms of transistor channel width is given as

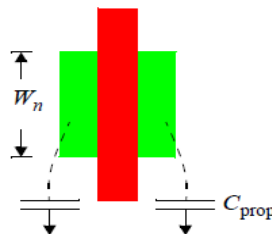


Figure 3.2: Channel Propagation Capacitance

$$C_{prop} = \begin{cases} \left(\frac{C_{ngdo}}{P}\right) W_{nX}, & \text{off} \\ \left[\frac{1}{2}\left(\frac{C_g}{A}\right) L_{min} + \left(\frac{C_{ngdo}}{P}\right)\right] W_{nX}, & \text{on} \end{cases} \quad (3.11)$$

Normalizing to the unit capacitance

$$C_{prop} = \begin{cases} \frac{\left(\frac{C_{ngdo}}{P}\right) W_{nX}}{\left(\frac{C_{in}}{W}\right) W_{min}} C, & \text{off} \\ \frac{\left[\frac{1}{2}\left(\frac{C_g}{A}\right) L_{min} + \left(\frac{C_{ngdo}}{P}\right)\right] W_{nX}}{\left(\frac{C_{in}}{W}\right) W_{min}} C, & \text{on} \end{cases} \quad (3.12)$$

The numbers for various processes are shown in Table 3.2.

Co.	line width	C_{ngdo}/C_{in}	C_{pgdo}/C_{in}
AMIS	0.6 μ m (SUBM)	0.10	0.16
IBM	0.5 μ m (SUBM)	0.16	0.22
AMIS	0.35 μ m (SUBM)	0.14	0.14
IBM	0.35 μ m (SUBM)	0.25	0.20
TSMC	0.35 μ m (SUBM)	0.15	0.23
IBM	0.25 (DEEP)	0.27	0.29
TSMC	0.25 (DEEP)	0.24	0.25
IBM	0.18 (DEEP)	0.20	0.27
TSMC	0.18 (DEEP)	0.26	0.30
IBM	0.13 (DEEP)	0.20	0.19

Table 3.2: Capacitance values for various processes

From the above table the average value of C_{ngdo}/C_{in} for various processes is about 0.25. Hence

$$C_{prop} \approx \begin{cases} \frac{1}{4} \frac{W_{nX}}{W_{min}} C, & \text{off} \\ \frac{1}{2} \frac{W_{nX}}{W_{min}} C, & \text{on} \end{cases} \quad (3.13)$$

The above expression makes the node capacitance different depending on whether the transistors are ON or OFF which greatly complicates analysis without significantly improving accuracy. To simplify the model C_{prop} is approximated as $C_{prop} = \frac{1}{2} \frac{W_{nX}}{W_{min}} C$. Similarly for the p-channel

$$C_{prop} = \frac{1}{2} \frac{W_{pX}}{W_{min}} C.$$

The diffusion capacitance (C_{diff}) varies depending on a) if there is a diffusion contact b) if there is no diffusion contact between two transistors c) if there is a diffusion contact but is shared between two transistors.

- a) The diffusion capacitance of transistor A in terms of unit capacitance C when there is a diffusion contact for n-diffusion and p-diffusion is modeled as

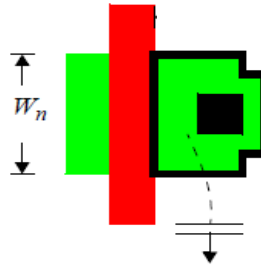


Figure 3.3: Diffusion Capacitance with diffusion contact

$$C_{ndcX} = \frac{1}{2} \left(\frac{KW_{nA}}{W_{min}} + K_1 \right) C \quad (3.14)$$

$$C_{pdcX} = \frac{1}{2} \left(\frac{KW_{pA}}{W_{min}} + K_1 \right) C \quad (3.15)$$

Where K and K_1 are process constants.

- b) The diffusion capacitance when there is no diffusion contact between two transistor A and B for n-diffusion and p-diffusion is modeled as

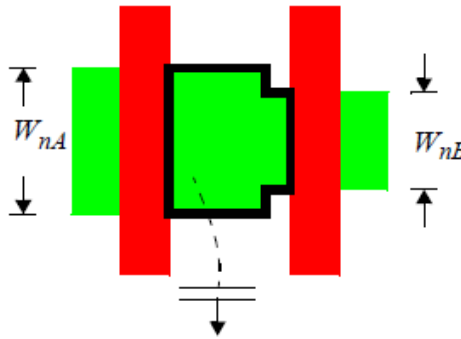


Figure 3.4: Diffusion Capacitance with no diffusion contact

$$C_{ndncXY} = \frac{1}{2} \left(\frac{K(W_{nA} + W_{nB})}{W_{min}} + K_1 \right) C \quad (3.16)$$

$$C_{pdncXY} = \frac{1}{2} \left(\frac{K(W_{pA} + W_{pB})}{W_{min}} + K_1 \right) C \quad (3.17)$$

- c) The diffusion capacitance when there is a diffusion contact but shared between two transistors A and B for n-diffusion and p-diffusion is modeled as

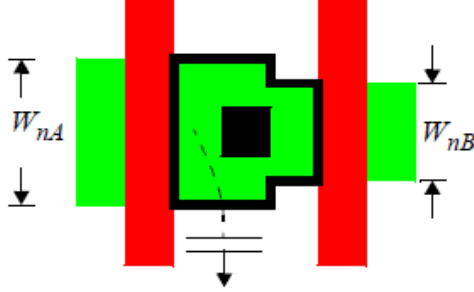


Figure 3.5: Diffusion Capacitance with shared diffusion contact

$$C_{ndscXY} = \frac{1}{2} \left(\frac{K(W_{nA} + W_{nB})}{W_{min}} + K_1 \right) C \quad (3.18)$$

$$C_{pdscXY} = \frac{1}{2} \left(\frac{K(W_{pA} + W_{pB})}{W_{min}} + K_1 \right) C \quad (3.19)$$

The diffusion capacitance when there is no diffusion contact and when there is a sharing contact between two transistors is modeled similarly as there is no significant difference within the accuracy of the model.

In case of 2-input NAND gate, the parasitic node capacitance C_{out} and C_1 is calculated as shown below by using the above derived expressions for the C_{prop} and C_{diff}

$$\begin{aligned} C_{out} &= \frac{1}{2} \frac{W_{nB}}{W_{min}} C + \frac{1}{2} \frac{W_{pA}}{W_{min}} C + \frac{1}{2} \frac{W_{pB}}{W_{min}} C + \frac{1}{2} \left(\frac{KW_{nB}}{W_{min}} + K_1 \right) C + \frac{1}{2} \left(\frac{K(W_{pA} + W_{pB})}{W_{min}} + K_1 \right) C \\ &= \frac{1}{2} \left(\frac{(1+K)(W_{nB} + W_{pA} + W_{pB})}{W_{min}} + 2K_1 \right) C \end{aligned} \quad (3.20)$$

$$\begin{aligned}
C_1 &= \frac{1}{2} \frac{W_{nA}}{W_{min}} C + \frac{1}{2} \frac{W_{nB}}{W_{min}} C + \frac{1}{2} \left(\frac{K(W_{nA}+W_{nB})}{W_{min}} + K_1 \right) C \\
&= \frac{1}{2} \left(\frac{(1+K)(W_{nA}+W_{nB})}{W_{min}} + K_1 \right) C
\end{aligned} \tag{3.21}$$

Similarly the channel resistances in case of 2-input NAND gate are given below

$$R_{nA} = \frac{W_{min}}{W_{nA}} R \qquad R_{nB} = \frac{W_{min}}{W_{nB}} R \tag{3.22}$$

$$R_{pA} = \frac{W_{min}}{W_{pA}} R \qquad R_{pB} = \frac{W_{min}}{W_{pB}} 2R \tag{3.23}$$

By substituting the channel resistances and the parasitic node capacitances in the parasitic delay equations for the 2-input NAND gate in table 3.1 the path delays can be expressed explicitly in terms of transistor widths. Hence the model is greatly useful for faster and easy estimation of circuit delay and can be effectively used for circuit speed and power optimization using transistor sizing.

Similarly the input gate capacitance (C_{in}) and the interconnect wire capacitance (C_{wire}) can be modeled as shown below.

$$C_{in} = \left(\frac{W_{nX} + W_{pX}}{W_{min}} \right) C \tag{3.24}$$

Modern deep submicron processes have a capacitance of about 0.2fF per micrometer of wire length.

$$C_{wire} = 0.2fF/\mu m \cdot L_{wire} \tag{3.25}$$

Normalizing to the unit capacitance

$$C_{wire} = \frac{0.2fF/\mu m \cdot L_{wire}}{\left(\frac{C_{in}}{W}\right) \cdot W_{min}} C = \frac{0.2fF/\mu m \cdot L_{wire}}{2fF/\mu m \cdot W_{min}} C = \frac{1}{10} \frac{L_{wire}}{W_{min}} C \tag{3.26}$$

In our analysis the wire resistance is not considered. Designers should add inverter repeaters to make the wire resistance negligible compared to transistor channel resistance. For more detailed discussion on how the expressions for the channel resistances and parasitic capacitances are derived please refer to [28].

This delay model can be easily extended to larger circuits where multiple gates are connected together by including interconnect wire capacitance and the input gate capacitance. For example consider a 2x4 Decoder circuit (DEC2) with enable signal using 3-input NAND gates and inverters as shown in Figure 3.6.

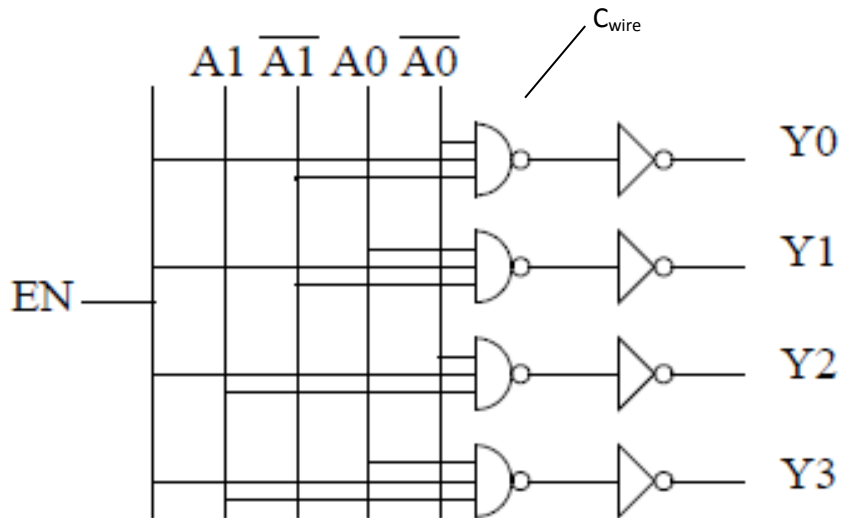


Figure 3.6: 2x4 Decoder circuit with enable signal

It is assumed that each of the NAND gates is identical and each of the inverters is identical. Hence we only have to design one NAND gate and one inverter and use the same design for the others.

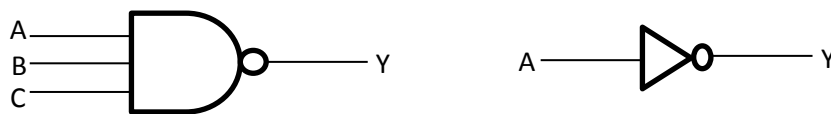


Figure 3.7: a) 3-input NAND b) Inverter

Similar to the 2-input NAND gate we discussed earlier, determine the parasitic path delays to the output from each input for the 3-input NAND gate and the inverter circuit shown in Figure 3.7.

Now the possible path parasitic delays in case of 2x4 Decoder circuit shown in Figure 3.6 are

$$t_{dP(A0Y0r)DEC2} = t_{dP(AYf)NAND3} + R_{out(AYf)NAND3} [C_{wire} + C_{in(AYr)INV}] + t_{dP(AYr)INV} \quad (3.27)$$

$$t_{dP(A0Y0f)DEC2} = t_{dP(AYr)NAND3} + R_{out(AYr)NAND3} [C_{wire} + C_{in(AYf)INV}] + t_{dP(AYf)INV} \quad (3.28)$$

$$t_{dP(A1Y0r)DEC2} = t_{dP(BYf)NAND3} + R_{out(BYf)NAND3} [C_{wire} + C_{in(AYr)INV}] + t_{dP(AYr)INV} \quad (3.29)$$

$$t_{dP(A1Y0f)DEC2} = t_{dP(BYr)NAND3} + R_{out(BYr)NAND3} [C_{wire} + C_{in(AYf)INV}] + t_{dP(AYf)INV} \quad (3.30)$$

$$t_{dP(ENY0r)DEC2} = t_{dP(CYf)NAND3} + R_{out(CYf)NAND3} [C_{wire} + C_{in(AYr)INV}] + t_{dP(AYr)INV} \quad (3.31)$$

$$t_{dP(ENY0f)DEC2} = t_{dP(CYr)NAND3} + R_{out(CYr)NAND3} [C_{wire} + C_{in(AYf)INV}] + t_{dP(AYf)INV} \quad (3.32)$$

To get the total path delay, add the corresponding effort delay $R_{out} \cdot C_{load}$ to each of the above expressions in Eq. 3.27 to Eq. 3.32.

Since the simplified RC delay model is explicitly expressed in terms of transistor channel widths, it helps in faster estimation of the circuit delay of any CMOS circuit and also helps analyze the effects of changing transistor channel widths on circuit delay easily. Unlike the logical effort model discussed in Chapter 2, this model allows p and n channel widths selection to be independent of each other in designing an effective circuit in terms of delay and power and is also more accurate than the logical effort model as it takes into consideration the interconnect wire capacitance which plays a significant role in determining the circuit delay when the wire lengths are long.

3.2 Power Model

Minimizing just the delay using the above discussed delay model would result in large transistors which will consume more power and may not use the silicon area efficiently. Hence, there should be a constraint on power when minimizing the delay. Before using the power as a constraint to find the optimum transistor sizes, we need a model for the power changes when changing transistor widths [28]. Fortunately almost all power consumption is proportional to the width of the transistor channels. The power dissipative components in CMOS circuits consist of off-state leakage power, dynamic power due to charging and discharging of node capacitances, short-circuit power, switching power due to parasitic capacitances and glitch power due to unequal arrival of signals. But the major components are the leakage power, dynamic power and the short-circuit power. Let us look at how these components are modeled as proportional to transistor channel widths.

Off-state leakage power

When transistors are turned off, there can be a small drain current (I_{Doff}). Though this current is too small to have a considerable impact on the delay, it can have a significant impact on the power consumption.

$$P_{leak} = V_{dd}I_{Doff} = V_{dd} \left(\frac{I_{Doff}}{I_{Don}} \right) I_{Don} \quad (3.33)$$

Where I_{Don} is the drain current when the transistor is on. If we approximate I_{Don} with channel impedance

$$I_{Don} = \frac{V_{dd}}{R_{on}} = \frac{V_{dd}}{R} \frac{W}{W_{min}} \quad (3.34)$$

Where R is the unit resistance, W is the channel width, and W_{min} is the minimum transistor width.

Hence the leakage power is expressed in terms of transistor width as

$$P_{leak} = V_{dd} \left(\frac{I_{Doff}}{I_{Don}} \right) I_{Don} = \frac{V_{dd}^2}{R} \left(\frac{I_{Doff}}{I_{Don}} \right) \frac{W}{W_{min}} \quad (3.35)$$

Dynamic power

The dynamic power consumption for a single logic gate (neglecting C_{wire}) is given as

$$P_{dyn_gate} = (C_{out} + C_{load}) V_{dd}^2 f \quad (3.36)$$

Where C_{out} comes from the parasitic delay and C_{load} from the effort delay. Since each transistor appears as part of the parasitic delay in one logic gate and the effort delay in another logic gate, it follows that each transistor contributes capacitance both through C_{out} and C_{load} . Therefore, the dynamic power per transistor is approximately modeled as

$$P_{dyn_tran} = \left[\left[(1 + K) \frac{W}{W_{min}} + K_1 \right] C + C \right] V_{dd}^2 f = [(2 + K)C] V_{dd}^2 f \frac{W}{W_{min}} + K_1 C V_{dd}^2 f \quad (3.37)$$

Where C is the unit capacitance, f is the frequency of charging and discharging the node capacitance.

Short circuit power

Short circuit power is the only power dissipation component that cannot be modeled as proportional to transistor width. Fortunately, it is almost always negligible compared with other power dissipative components. Therefore, we do not need to include that in our optimization calculations.

Since most of the power consumption per transistor is proportional to the transistor channel width, the total power consumption of a CMOS circuit can be approximated as proportional to the sum of all the transistor widths in the circuit.

3.3 Optimization Methodology

In order to effectively optimize the circuit for speed and power by determining the optimum transistor channel width sizes, a heuristic approach based on gradient descent method is used. As discussed in Chapter 2, the gradient descent, also known as steepest descent, is a first order optimization algorithm used to find a local minimum by taking steps proportional to the negative of the gradient of the objective function (OF) at the current point in the decision variable (DV) space.

$$X_{i+1} = X_i - \delta[\nabla OF / |\nabla OF|] \quad (3.38)$$

Where δ is the step size.

In our optimization analysis, each transistor width (W_i) is bounded between a minimum (W_{\min}) and a maximum (W_{\max}) to avoid very large transistor widths and the decision variable space is normalized so that each transistor width after normalization varies from 0 to 1.

$$W_{norm} = \frac{W_i - W_{\min}}{W_{\max} - W_{\min}} \quad (3.39)$$

When minimizing power, the OF is considered as the sum of the transistor widths in the circuit as discussed in the power model.

$$OF = \sum W_i \quad (3.40)$$

When minimizing delay, the OF is the maximum of all possible critical path delays.

$$OF = \max\{ path_delays \} \quad (3.41)$$

In this case, the minimum of the OF lies at the intersection of two delay paths and looks like a surface as shown in the figure below.

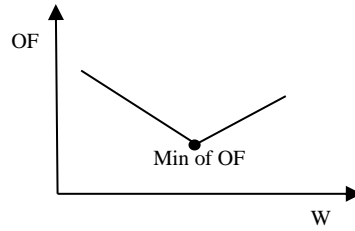


Figure 3.8: OF surface when minimizing delay

Since the minimum of the OF is V-shaped, it is very difficult to find a point at which the gradient is zero or very small to determine the convergence. Hence the convergence of the optimization algorithm is achieved when the step size falls below a small value ϵ usually in the order of 10^{-5} or when the gradient of the OF w.r.t the decision variables is less than or equal to ϵ . The heuristic approach used in order to achieve faster and effective convergence is discussed below in steps.

1. Choose an initial step size $\delta = \delta_{int}$ and start the optimization using gradient descent method
2. While tracking the best OF point in the DV space found in each iteration, if the algorithm did not find a better OF value in n iterations from the current iteration start the optimization from the previously found best point with a reduced value of δ i.e. $\delta = \delta/2$
3. To speed up the convergence of the optimization problem, when the value of δ is less than δ_1 but greater than δ_2 such that $\delta_{int} > \delta_1 > \delta > \delta_2$ reduce the number of iterations n to n_1 required to achieve the better OF value. When the δ value is less than δ_2 such that $\delta_1 > \delta_2 > \delta > \epsilon$ further reduce the number of iterations n_1 to n_2 required to achieve a better OF value.

To achieve satisfactory results without effecting the optimization results much, but to significantly reduce the convergence time choose $n_1 = n/2$, $n_2 = n/10$. δ_1 , δ_2 can be chosen depending on the value of δ_{int} . For example, when $n=50$ then $n_1=25$, $n_2=5$ and when $\delta_{int}=0.5$ then $\delta_1=0.1$, $\delta_2=0.01$.

The circuit optimization is done to minimize the circuit delay with power constraint and to minimize the circuit power with delay constraint. The optimization algorithm is defined in such a way, when minimizing the delay with power constraint, at the current point in DV space if the power constraint is met then the OF is circuit delay otherwise the OF will be the circuit power. Similarly when minimizing the power with delay constraint, at the current point in DV space if the delay constraint is met then the OF is the circuit power otherwise the OF will be the circuit delay.

This heuristic approach helps to optimize the circuit much faster than the conventional methods discussed in [11] and [12] while converging to a satisfactory solution to obtain the optimum transistor widths for the circuit optimization involving the multiple delay paths and hundreds of transistor widths.

3.4 Model Validation

To validate the simplified RC delay model a simple four inverter chain circuit is considered. The model is tested against SPICE and Logical Effort(LE) in the case of the circuit with lightly loaded condition, i.e., with no interconnect wire capacitance, C_{wire} and small load capacitance, C_{load} and also in case of the circuit with heavily loaded condition, i.e., with large C_{wire} and C_{load} .

The analysis is done for 0.18 micrometer process technology. In case of the model for 0.18 micrometer process the values for K and K_1 in equations (3.14) - (3.19) is equal to one i.e. $K=K_1=1$. The model is equivalent to the logical effort method when $K_1=0$ and C_{wire} is ignored.

3.4.1 Lightly loaded case

Consider the circuit of four inverter chain as shown in the Figure 3.9 with no interconnect wire capacitance and a small load capacitance on the output node.

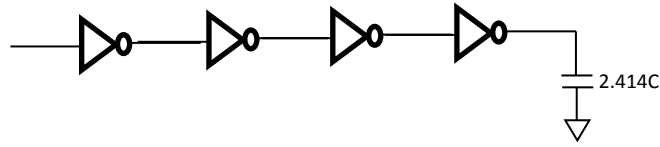


Figure 3.9: Four inverter chain with a light load and no C_{wire} . $C=0.89fF$ for $0.18\mu m$ process

It is assumed that the first inverter gate transistor widths are kept fixed. The p-transistor width is fixed at $1.414W_{min}$ and the n-transistor width is fixed at W_{min} . For the remaining inverter gates all the transistor widths are set at W_{min} initially, where $W_{min} = 0.36$ micrometers for a 0.18 micrometer process. The circuit is then optimized with different power constraints using the simplified RC delay model and the results are shown in the plot below.

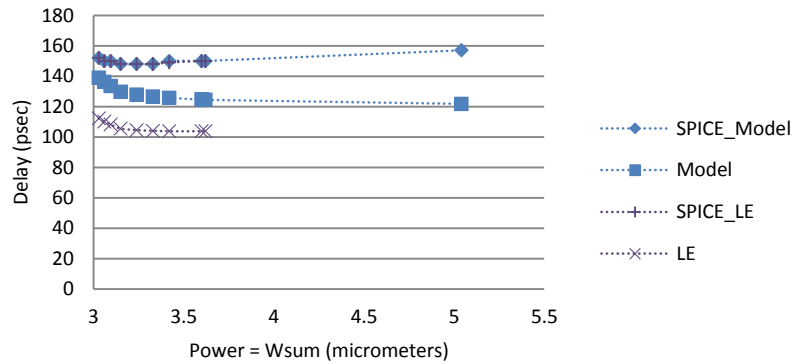


Figure 3.10: Power-delay plot for lightly loaded case

From the above plot, in case of lightly loaded case the optimum transistor widths found by the model agrees well with SPICE results in the region with tight power constraints but did not agree well when the power constraint is kept loose. Also, it is obvious that the simplified RC delay model predicts delays much more accurately than the logical effort method.

The above four inverter chain can be solved in closed form for minimum delay to determine the optimum transistor sizes as discussed in [28] and the optimum value of transistor size ratio is found to be $Z_i = \sqrt{2}$; where $Z_i = W_{pi}/W_{ni}$ and i represents the number of the inverter in the

inverter chain. The optimum widths found by the simplified RC delay model for minimum circuit delay in the lightly loaded case also agrees with the closed form solution i.e $Z_i = \sqrt{2}$

3.4.2 Heavily loaded case

Consider the circuit of four inverter chain as shown in Figure 3.11 with large interconnect wire capacitance and a large load capacitance on the output node.

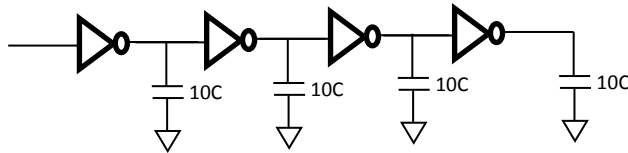


Figure 3.11 Four inverter chain with a heavy load and large C_{wire} . $C=0.89\text{fF}$ for $0.18\mu\text{m}$ process

Similar to the lightly loaded case it is assumed that the first inverter gate transistor widths are kept fixed, p-transistor width is fixed at $1.414W_{min}$ and the n-transistor width is fixed at W_{min} . For the remaining inverter gates all the transistor widths are set at W_{min} initially, where $W_{min} = 0.36$ micrometers for 0.18 micrometer process. The circuit is then optimized with different power constraints using the simplified RC delay model and the results are shown in the plot below.

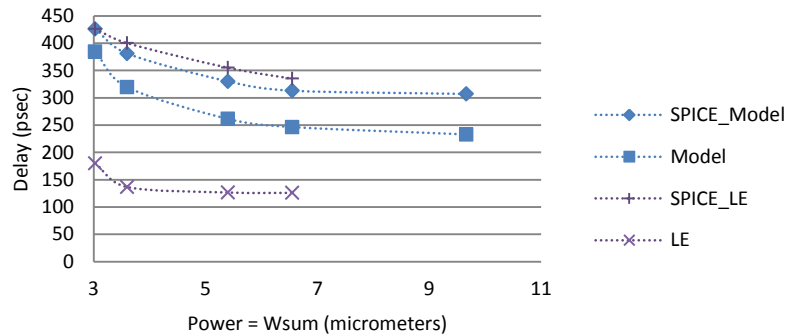


Figure 3.12 Power-delay plot for heavily loaded case

From the above plot, in case of heavily loaded case the optimum transistor widths found by the model agrees well with SPICE results. Again, it is obvious that the simplified RC delay model predicts delay much accurately than the logical effort method.

The above SPICE results are the values when the optimum transistor widths found from optimizing the inverter chain using the model and the logical effort method are used in SPICE simulation to justify the widths found are actually reducing the SPICE delay.

Also, a comparison of CPU computation time taken to estimate the delay in case of cascaded inverter gates is done and is shown in Table 3.3.

	CPU Time (seconds)		
	SPICE	IRSIM	Model
INV	0.022	0.001	$< 10^{-3}$
INV ₂	0.022	0.001	$< 10^{-3}$
INV ₄	0.024	0.001	$< 10^{-3}$
INV ₈	0.029	0.001	$< 10^{-3}$
INV ₁₆	0.102	0.001	$< 10^{-3}$
INV ₃₂	0.188	0.002	$< 10^{-3}$
INV ₆₄	0.411	0.003	$< 10^{-3}$
INV ₁₂₈	0.953	0.005	$< 10^{-3}$
INV ₂₅₆	1.582	0.008	$< 10^{-3}$

Table 3.3: CPU time comparison

In the above table the subscript beside the term INV refers to the number of cascaded inverter gates. As you can see from the table 3.3, SPICE takes significant amount of computation time for circuits with just couple of hundreds of transistors, while the IRSIM and the Model did not take much computation time compared to SPICE. Overall the model performed well in quickly estimating the delay of the circuit.

3.5 Convexity of the objective function

Reference [29] says a real function f is convex on an interval $[a, b]$ if for any two points x_1 and x_2 in $[a, b]$ and any λ where $0 < \lambda < 1$,

$$f[\lambda x_1 + (1 - \lambda)x_2] \leq \lambda f(x_1) + (1 - \lambda)f(x_2) \quad (3.42)$$

In our optimization using transistor sizing, each transistor width is bounded between a minimum and a maximum i.e. $W_i \in [W_{min}, W_{max}]$. The parasitic channel resistance, parasitic channel and diffusion capacitances, wire capacitance terms discussed in Section II are all convex on the interval $[W_{min}, W_{max}]$ as they all obey the above condition for the convex function. It is also observed that the product of the channel resistance and the parasitic capacitance is also convex on the interval $[W_{min}, W_{max}]$. From the properties of the convex functions, if two functions are convex then the sum of two convex functions is also convex. Hence the path delay expressions which are just the sum of the convex functions is also convex over the interval $[W_{min}, W_{max}]$.

When minimizing the delay, the objective function is the maximum of the possible critical path delays. Again, from the properties of convex functions, if two functions are convex then the maximum of the two convex functions is also convex. Since each path delay is convex on the interval $[W_{min}, W_{max}]$, the maximum of the path delays is also convex on the interval $[W_{min}, W_{max}]$.

When minimizing the power, the objective function is the sum of the transistor widths which is also convex on the interval $[W_{min}, W_{max}]$ as per the definition of the convex function.

CHAPTER IV

PERFORMANCE EVALUATION & OPTIMIZATION RESULTS

4.1 Parallel prefix adders

Addition forms the basis for many processing operations. As a result, adder circuits are of great interest to digital system designers. Many adder architectures serve different speed and area requirements. The focus of this dissertation is to model the popular high speed parallel prefix adders such as Brent-Kung, Skylanksy and Kogge-Stone using the simplified RC delay model and compare their performance w.r.t speed and power by optimizing with different performance constraints.

These adders perform the addition operation based on carry generation and propagation logic.

The expressions to describe whether a group spanning bits $i \dots j$, inclusive, generate or propagate a carry are given as shown









$$G_{i:j} = G_{i:k} + P_{i:k} \cdot G_{k-1:j} \quad (4.1)$$

$$P_{i:j} = P_{i:k} \cdot P_{k-1:j} \quad (4.2)$$

With the base case

$$G_{i:i} = A_i \cdot B_i \quad (4.3)$$

$$P_{i:i} = P_i = A_i \oplus B_i \quad (4.4)$$

For an N bit adder size, the Brent-Kung adder computes the carry generate and propagate prefixes for 2-bit groups. These are used to find prefixes for 4-bit groups, which in turn are used to find prefixes for 8-bit groups, and so forth. The prefixes then fan back down to compute the carries-in to each bit. The adder requires $2(\log_2 N) - 1$ stages. The fanout is limited to 2 at each stage. The Sklansky adder reduces the delay $\log_2 N$ stages by computing intermediate prefixes along with the large group prefixes. This comes at the expense of fanouts that double at each level. These high fanouts cause poor performance on wide adders unless the gates are appropriately sized. The Kogge-Stone adder achieves both $\log_2 N$ stages and fanout of 2 at each stage. This comes at the cost of many long wires that must be routed between stages. The adder also contains more PG cells; while this may not impact the area if the adder layout is on a regular grid, it will increase the power consumption. Despite these costs, the Kogge-Stone adder is widely used in high-performance 32-bit and 64-bit adders. This dissertation presents power-delay performance comparison for these adders in case of 32bit, 64bit and 128bit adder sizes. For simplicity 16-bit adder structures are presented as shown in the figures Figure 4.1, Figure 4.2 and Figure 4.3 to discuss the design methodology used for 32-bit, 64-bit and 128-bit adders.  - refers to bitwise PG cell,  - refers to gray cell designed as AOI gate,  - refers to black cell designed as combination of AOI gate and NAND gate,  - refers to gray cell designed as OAI gate,  - refers to black cell designed combination of OAI gate and NOR gate,  - refers to an inverter gate,  - refers to a pair of inverter gates and  - refers to sum bit generating gate. [30] discusses the design of bitwise PG cell, gray cell, black cell and sum bit cells in detail. It is important to note that AOI gate takes in un-inverted inputs and generates an inverted output. Similarly the OAI gate takes in inverted inputs and generates an un-inverted output. Hence inverter gates are used as needed to provide the right input signal to the corresponding gates. Also, in order to simplify the adder design and analysis problem it is assumed that gates are identical in size to other gates if the gates are driving a similar load. Since many gates drive similar loads, in each stage of the adder structure there will be groups of identical gates.

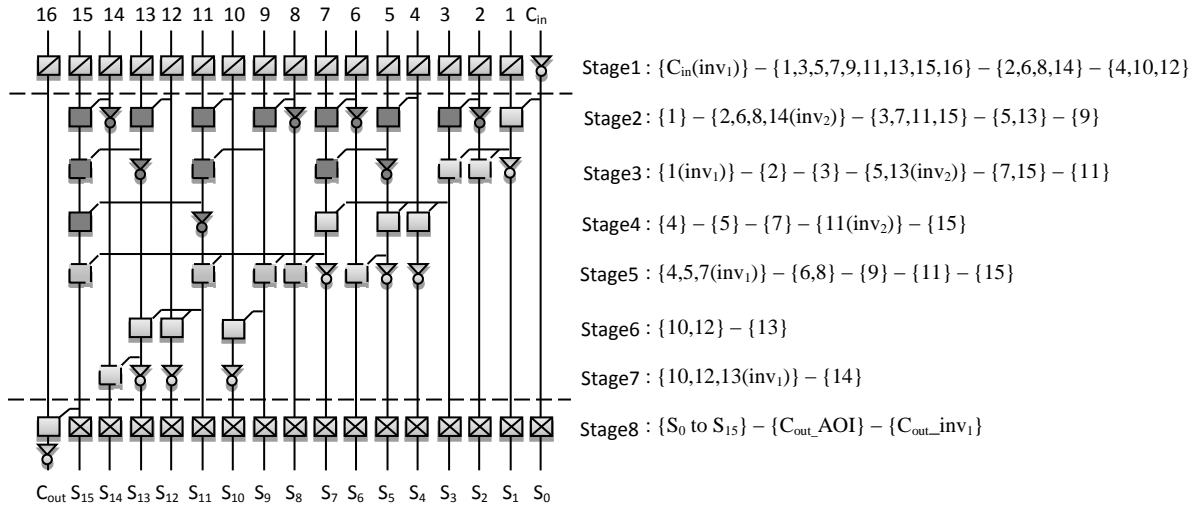


Figure 4.1: 16-bit Brent-Kung adder

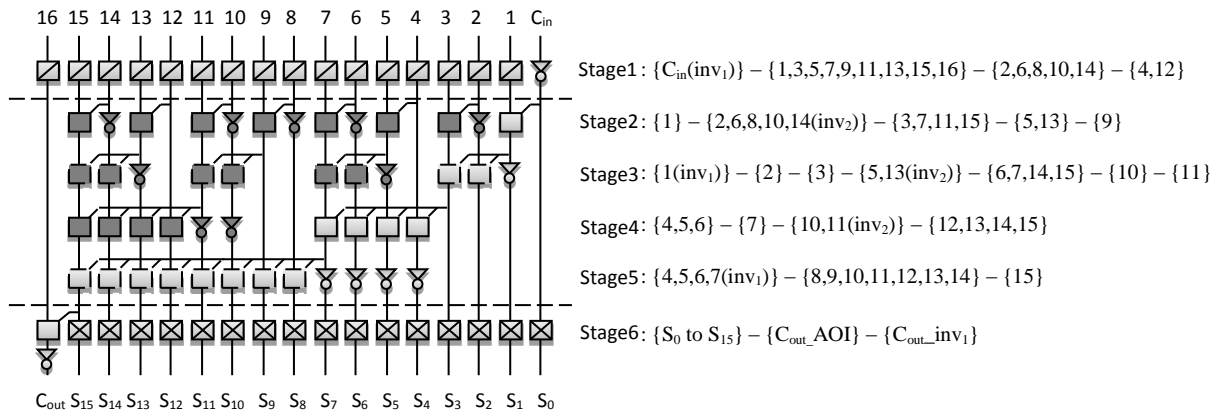


Figure 4.2: 16-bit Skylansky adder

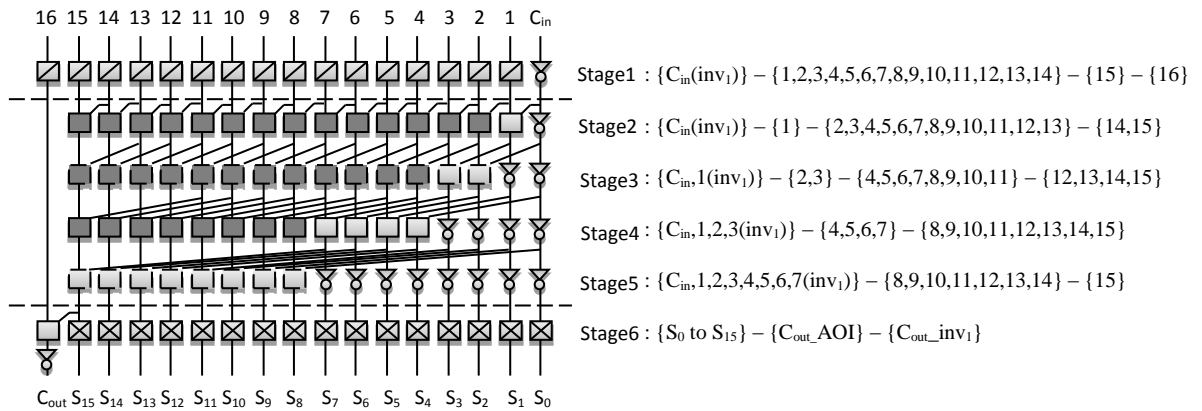


Figure 4.3: 16-bit Kogge-Stone adder

In the above figures of the adder structures, each identical group is enclosed in curly braces ‘{ }’ and each logic gate in a stage is represented by the input bit position for quick reference to the identical groups.

The path delay expressions are determined using the simplified RC delay model as discussed earlier for those paths in order to define all the possible critical path delays and also to include all the circuit’s non-identical transistor widths in the decision variable space. The wire capacitance as modeled in (3.26) is determined by the length of the interconnect wire. It is assumed that the wire capacitance for small wires is zero and only the wire capacitances of those wires that are significantly long were included as non-zero capacitance in calculating the circuit delay.

4.2 Optimization Results

The circuit analysis is done in the case of TSMC 0.18 micrometer process technology i.e. $K=1$, $R=5K\Omega$, $C=0.89fF$ and $C_{load}=2C$. Using the optimization methodology discussed in Section III, the three adder structures are optimized to determine the optimum transistor widths while meeting the performance constraints and the results are shown in Figure 4.4, Figure 4.5 and Figure 4.6 for 32-bit, 64-bit and 128-bit adder sizes respectively. During the optimization, it is assumed that all the transistors widths in the first stage of the adders are kept fixed at the minimum transistor width. The first data point on the left-side of each power-delay curve refers to the minimum delay point found with no power constraint, the last data point on the right-side of each power-delay curve refers to the data point when minimum transistor widths are used for all the circuit transistors and the rest of the data points were determined by minimizing power with different delay constraints. The small solid circle on each curve refers to the minimum power-delay product data point.

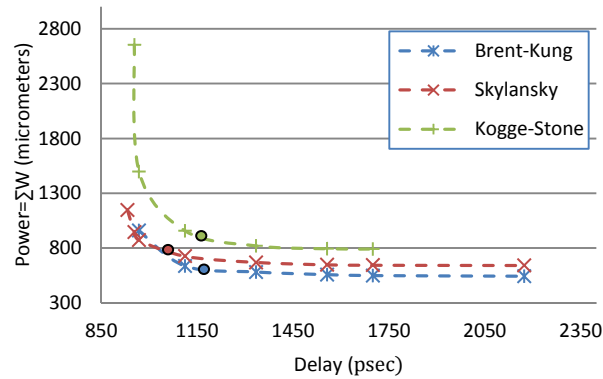


Figure 4.4: Power-delay comparison plot for 32-bit adder case

In the case of a 32-bit adder after circuit optimization the Skylansky adder is the fastest of all three adders, followed by Kogge-Stone and then Brent-Kung. However, the Brent-Kung has the minimum power-delay product followed by Skylansky and then Kogge-Stone.

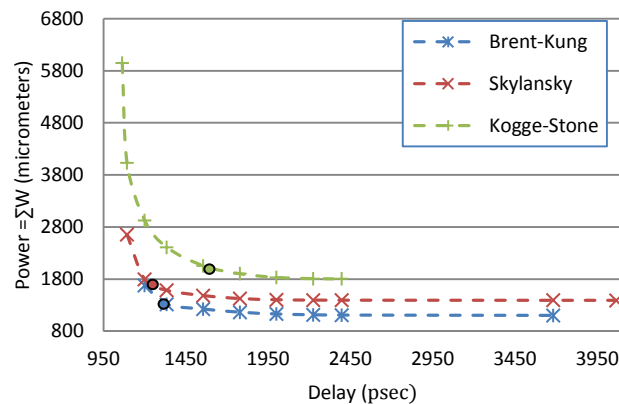


Figure 4.5: Power-delay comparison plot for 64-bit adder case

In the case of a 64-bit adder after circuit optimization the Kogge-Stone adder is the fastest of all three adders, but its power consumption is excessive. It is followed by Skylansky and then Brent-Kung in terms of speed, whereas Brent-Kung still has the minimum power-delay product followed by Skylansky and then the Kogge-Stone.

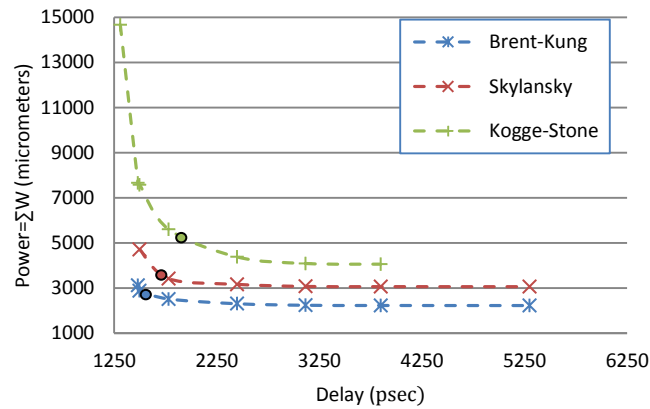


Figure 4.6: Power-delay comparison plot for 128-bit adder case

In the case of a 128-bit adder after circuit optimization the Kogge-Stone adder is still the fastest of all three adders, but again with excessive power consumption. Kogge-Stone is now followed by Brent-Kung and then the Skylansky in terms of speed. Whereas Brent-Kung still has the minimum power-delay product followed by Skylansky and then the Kogge-Stone.

From the above three plots it can be observed that Kogge-Stone is the fastest when power consumption is not a problem. The Brent-Kung adder has the minimum power-delay product when optimum transistor widths are used in all the three adder sizes. Appendix A shows more details on the optimum transistor widths found and the histograms of the transistor widths for all the three adders at various constraints.

CHAPTER V

CONCLUSION

The proposed simplified RC delay model can quickly estimate the circuit delay for any complex CMOS circuit. It is more accurate and allows more efficient circuit design than the existing logical effort method in that it includes the effects of interconnect wire capacitance in determining the circuit delay and determines p and n transistor widths independently. It is much easier to use the numerical optimization techniques with our simplified RC delay equations rather than SPICE for the following reasons:

1. The delays are explicit functions of transistor widths making it obvious what happened when any width is changed.
2. The width dependence of the delays is smooth so that simple numerical methods can be used to quickly find the optimum transistor widths.

Using the model and a heuristic gradient based optimization technique discussed in Chapter III the three popular parallel prefix adders are optimized and the power-delay performance comparison was done without having to do tedious layout and simulation. The results suggest that the Brent-Kung adder has the smallest power-delay product when optimum transistor widths are used, followed by the Skylansky adder and then the Kogge-Stone adder. While Kogge-Stone is the fastest, it comes at the expense of excessive power consumption.

REFERENCES

- [1] L. W. Nagel, and D. O. Pederson, "SPICE (Simulation Program with Integrated Circuit Emphasis)", Memorandum No. ERL-M382, University of California, Berkeley, Apr. 1973.
- [2] R. P. Brent and H. T. Kung, "A Regular Layout for Parallel Adders", IEEE Transactions on Computers, vol-31, no.3, pp. 260-264, Mar 1982.
- [3] J. Skylansky, "Conditional-Sum Addition Logic", IRE Transactions, EC-9, pp. 226-231, Jun2 1960.
- [4] P. M. Kogge and H. S. Stone, "A Parallel Algorithm for the Efficient Solution of a General Class of Recurrence Equations", IEEE Transactions on Computers, vol. 22, no. 8, pp. 786-792, Aug 1973.
- [5] W.C. Elmore, "The transient response of damped linear networks with particular regard to wideband amplifiers", Journal on Applied Physics, vol. 19, pp. 55-63, Jan 1948.
- [6] W. Shockley, "A unipolar field effect transistor", Proc. IRE, vol. 40, pp. 1365-1376, Nov. 1952.
- [7] T. Sakurai and A. R. Newton, "Alpha-Power Law MOSFET Model and its Applications to CMOS Inverter Delay and Other Formulas", IEEE Journal of Solid-State Circuits, vol. 25, no. 2, April 1990.
- [8] A. Slaz and M. Horowitz, "IRSIM: An Incremental MOS Switch-Level Simulator", Proc. 26th Design Automation Conference, pp. 173-178, June 1989.

- [9] J. Chang, "A Piecewise Linear Delay Modeling of CMOS circuits", Ph.D. dissertation, ECEN, OSU, Stillwater, OK, 2006.
- [10] I. Sutherland, B. Sproull and D. Harris, "Logical Effort: Designing Fast CMOS Circuits", CA: Morgan Kaufmann Publishers, 1999, pp. 1-83
- [11] A. Cauchy, "Méthode générale pour la resolution des systéms d'equations simultanées", Comp. Rend. Sci. Paris, 25, pp. 46-89, 1847.
- [12] J. Barzilai and J. M. Bowrwein, "Two point step size gradient methods", IMA Journal of Numerical Analysis, 8, pp. 141-148, 1988.
- [13] Jiren Yuan, Christer Svensson, "CMOS Circuit Speed Optimization Based on Switch Level Simulation", Circuits and Systems, IEEE International Symposium, vol.3, pp. 2109 – 2112, 1988.
- [14] Sachin S. Sapatnekar, Vasant B. Rao, Pravin M. Vaidya, Sung-Mo Kang, "An Exact Solution to the Transistor Sizing Problem for CMOS Circuits Using Convex Optimization", IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol.12, no.11, pp. 1621-1634, November 1993.
- [15] Masaaki Yamada, Sachiko Kurosawa, Reiko Nojima, Naohito Kojima, "Synergistic Power/Area Optimization with Transistor Sizing and Wire Length Minimization", IEEE Symposium on Low Power Electronics, pp. 50-51, 1994.
- [16] Jiren Yuan, Christer Svensson, "Principle of CMOS Circuit Power-Delay Optimization with Transistor Sizing", IEEE International Symposium on Circuits and Systems, vol.1, pp.637-640, 1996.
- [17] A. Wroblewski, O. Schumacher, C. V. Schimpfle, J. A. Nossek, "Minimizing Gate Capacitance with Transistor Sizing", IEEE International Symposium on Circuits and Systems, vol. 4, pp.186-189, 2001.

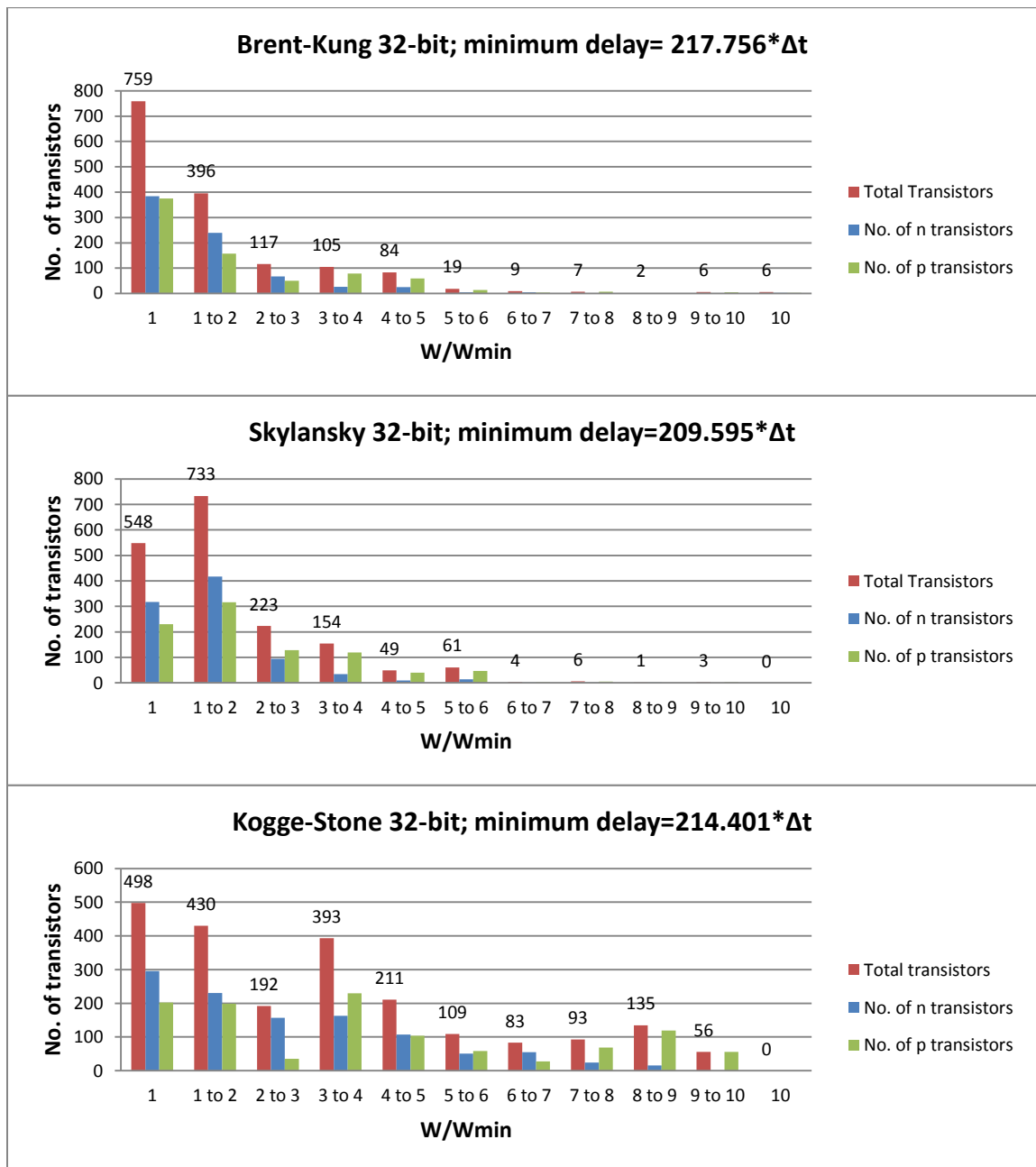
- [18] H.Y. Chen, S.M Kang, "A New Circuit Optimization Technique for High Performance CMOS Circuits", IEEE Transactions on Computer-Aided Design, vol.10, no.5, May 1991.
- [19] Manjit Borah, Robert Michael Owens, Mary Jane Irwin, "Transistor Sizing for Low Power CMOS Circuits", IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol.15, no.6, June 1996.
- [20] Robert Rogenmoser, Hubert Kaeslin, "The Impact of Transistor Sizing on Power Efficiency in Submicron CMOS Circuits", IEEE Journal of Solid-State Circuits, vol.32, no.7, July 1997.
- [21] Maitham Shams, Mohamed I. Elmasry, "Delay Optimization of CMOS Logic Circuits using Closed-Form Expressions", International Conference on Computer Design, pp. 563-568, 1999.
- [22] Bradley S. Carlson, Suh-Juch Lee, "Delay Optimization of Digital CMOS VLSI Circuits by Transistor Reordering", IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol.14, no.10, October 1995.
- [23] Li Ding, Pinaki Mazumder, "Optimal Transistor Tapering for High-Speed CMOS Circuits", Proceedings of the Design, Automation and Test in Europe Conference and Exhibition, pp. 708-713, 2002.
- [24] A. Baliga, D. Yagain, "Design of High speed adders using CMOS and Transmission gates in Submicron Technology: A Comparative Study", IEEE Fourth International Conference on Emerging Trends in Engineering and Technology, pp.284-289, 2011.
- [25] K. Nehru, A. Shanmugam, S. Vadivel, "Design of 64-Bit Low Power Parallel Prefix VLSI Adder for High Speed Arithmetic Circuits", IEEE International Conference on Computing, Communication and Applications, pp.1-4, 2012.

- [26] A. N. Jayanthi, C. S. Ravichandran, "Comparison of Performance of High Speed VLSI Adders", IEEE International Conference on Current Trends in Engineering and Technology, pp.99-104, 2013.
- [27] N. Poornima, V. S. Kanchana Bhaaskaran, "Power-Delay Optimized 32 Bit Radix-4, Sparse-4 Prefix Adder", IEEE Fifth International Conference on Signal and Image Processing, pp.201-205, 2014.
- [28] L. G. Johnson, "Advanced Digital VLSI Design", Lecture notes for ECEN 6263, OSU, Stillwater, OK, 2010. Available <http://lgjohn.okstate.edu/6263/index.html>
- [29] W. Rudin, "Principles of Mathematical Analysis", p. 101, 1976
- [30] N. H. E. Weste and D. M. Harris, "CMOS VLSI DESIGN: A Circuits and Systems Perspective", 4th ed., ch. 11, pp. 429-461, 2011.

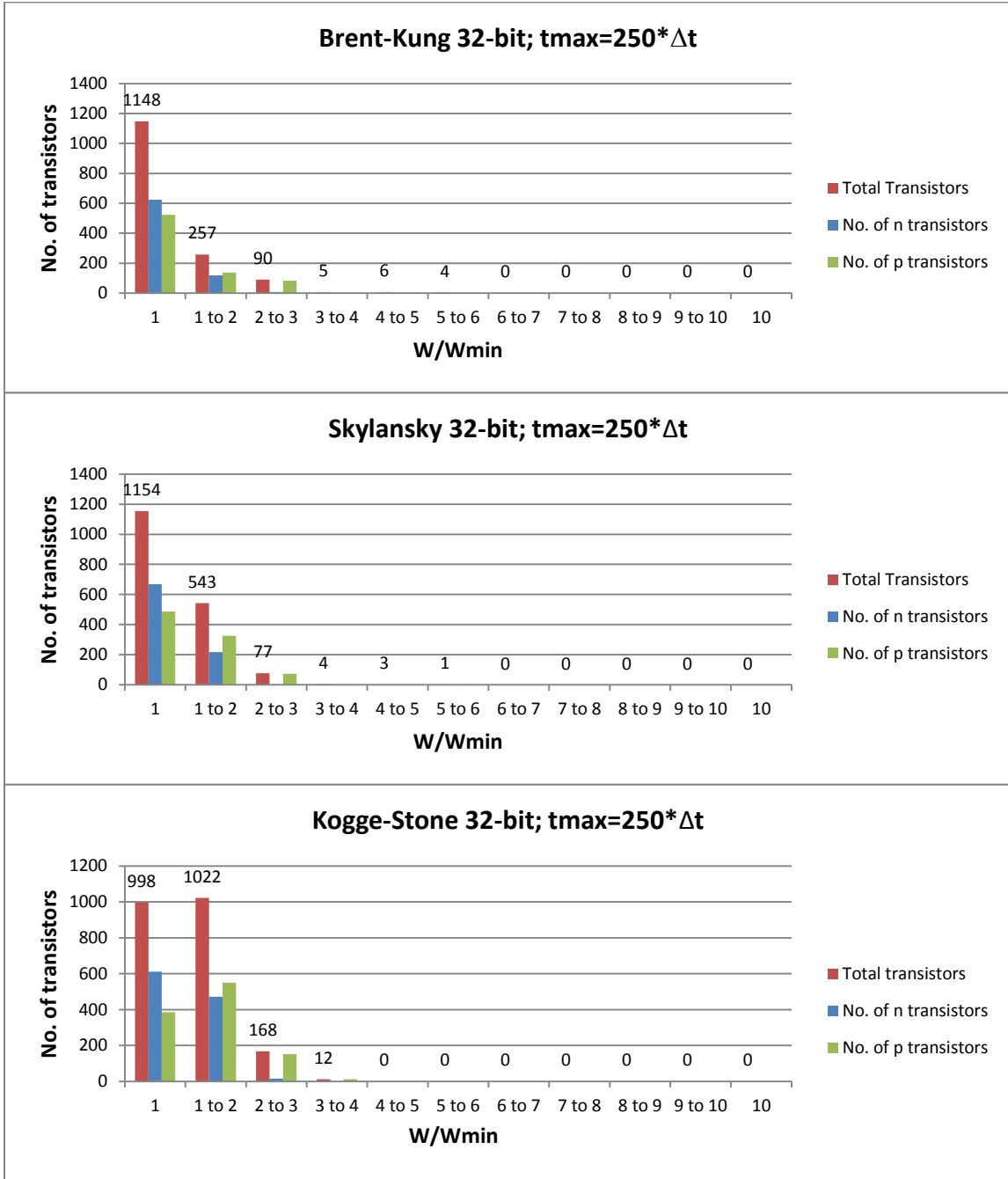
APPENDICES

APPENDIX A

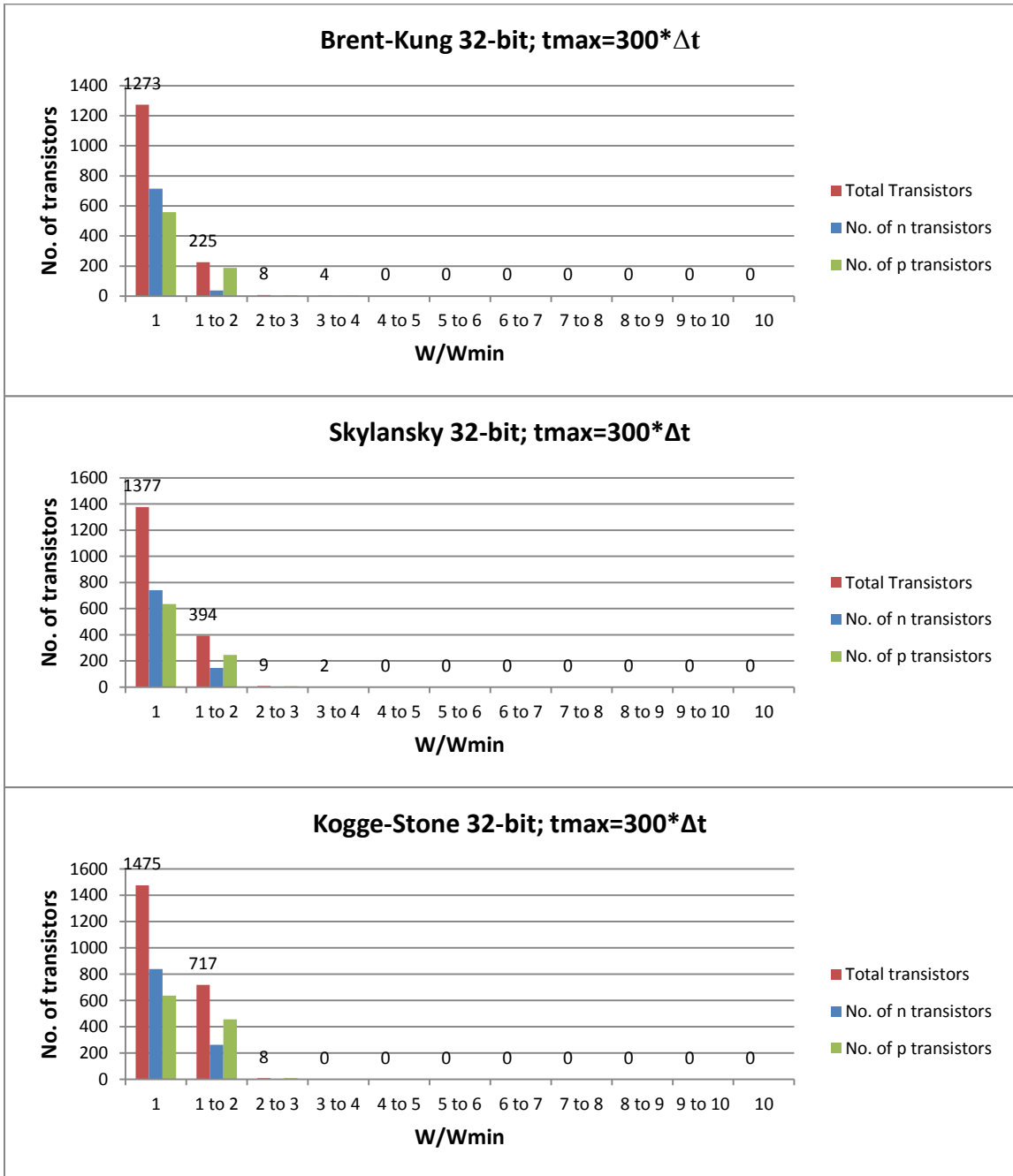
Histograms for 32-bit adder size for minimum delay case; where $\Delta t = R \cdot C$



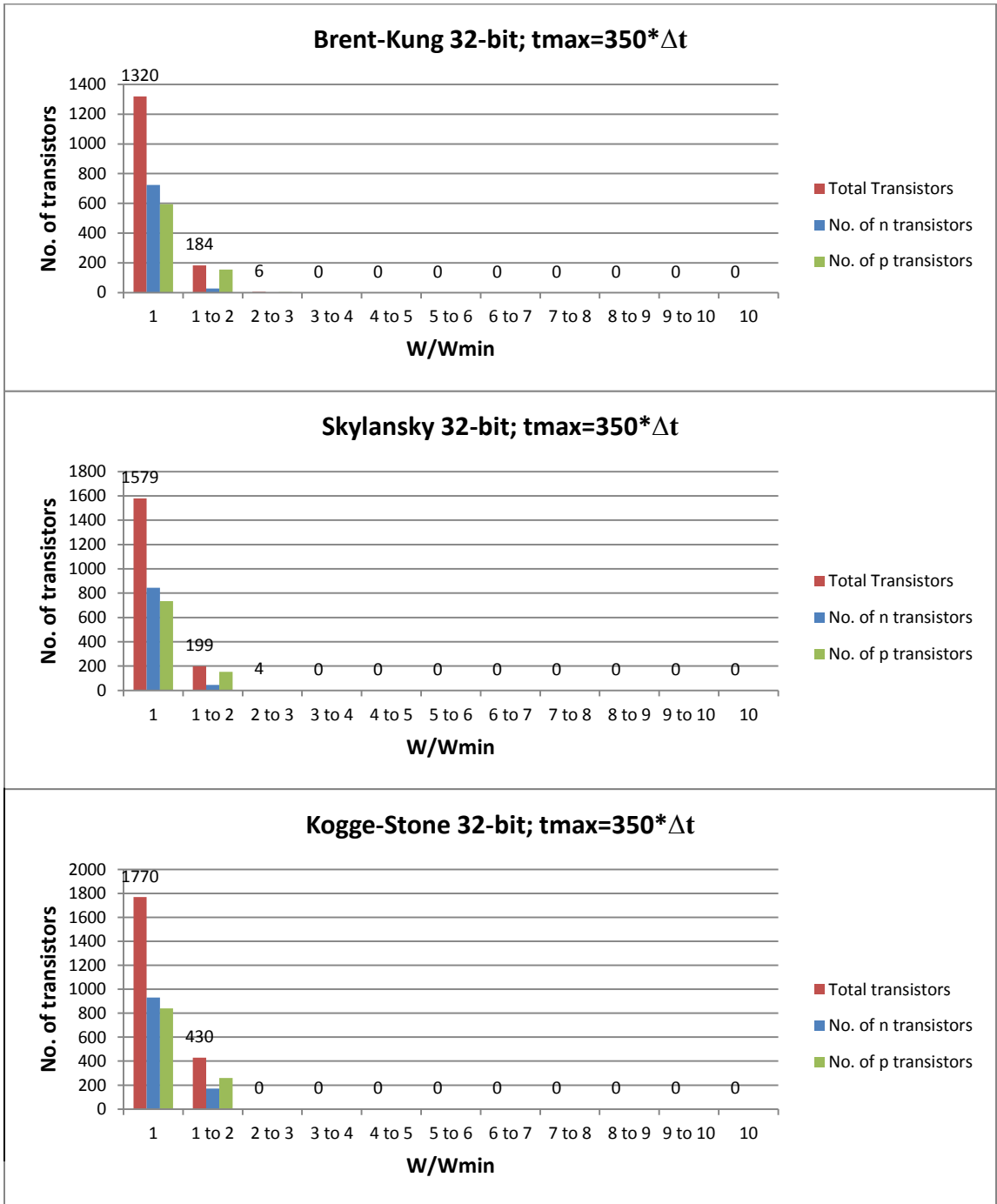
Histograms for 32-bit adder size with delay constraint of $t_{max}=250*\Delta t$, where $\Delta t=R*C$



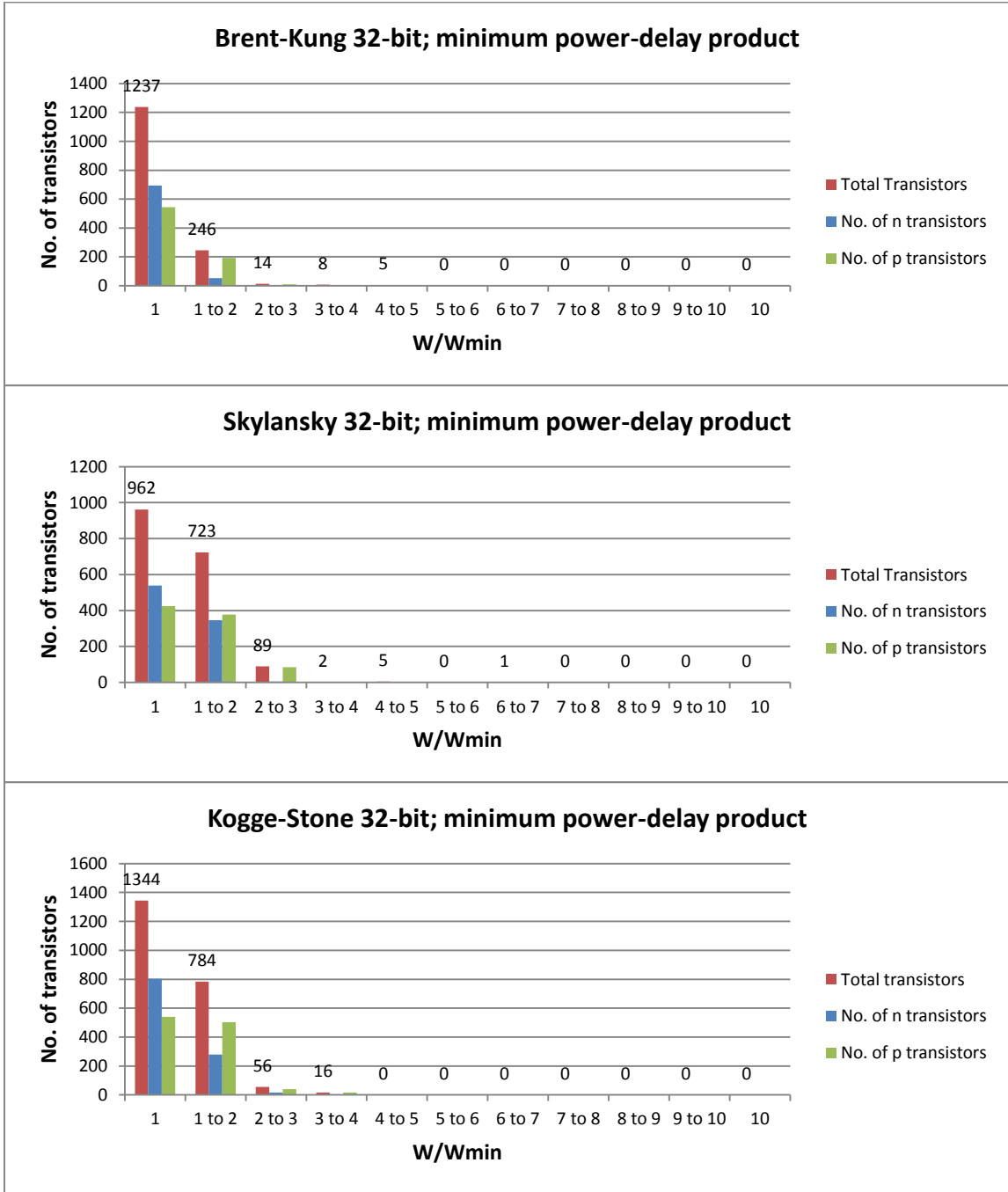
Histograms for 32-bit adder size with delay constraint of $t_{max}=300*\Delta t$, where $\Delta t=R*C$



Histograms for 32-bit adder size with delay constraint of $t_{max}=350*\Delta t$, where $\Delta t=R*C$

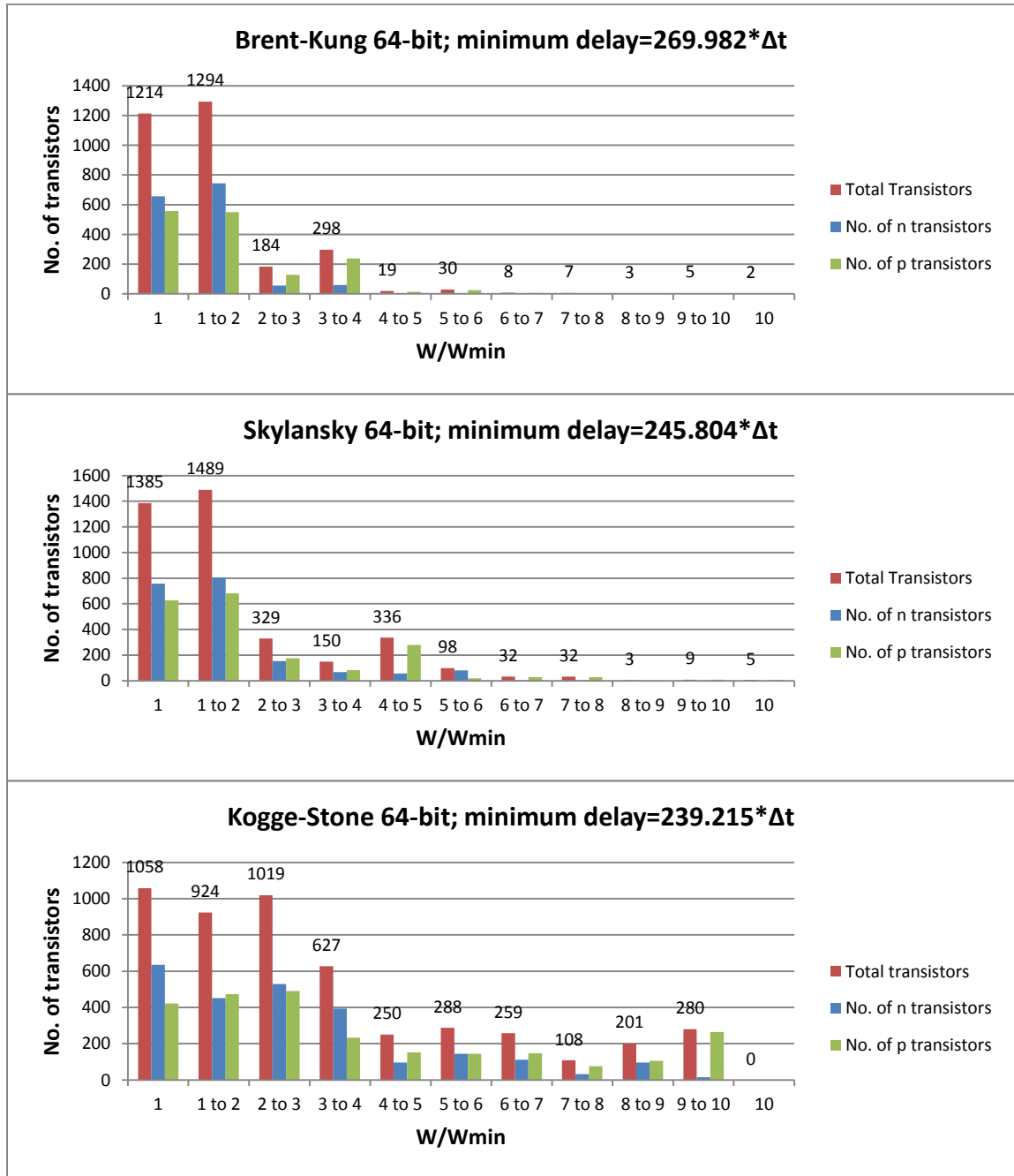


Histograms for 32-bit adder size for minimum power-delay product

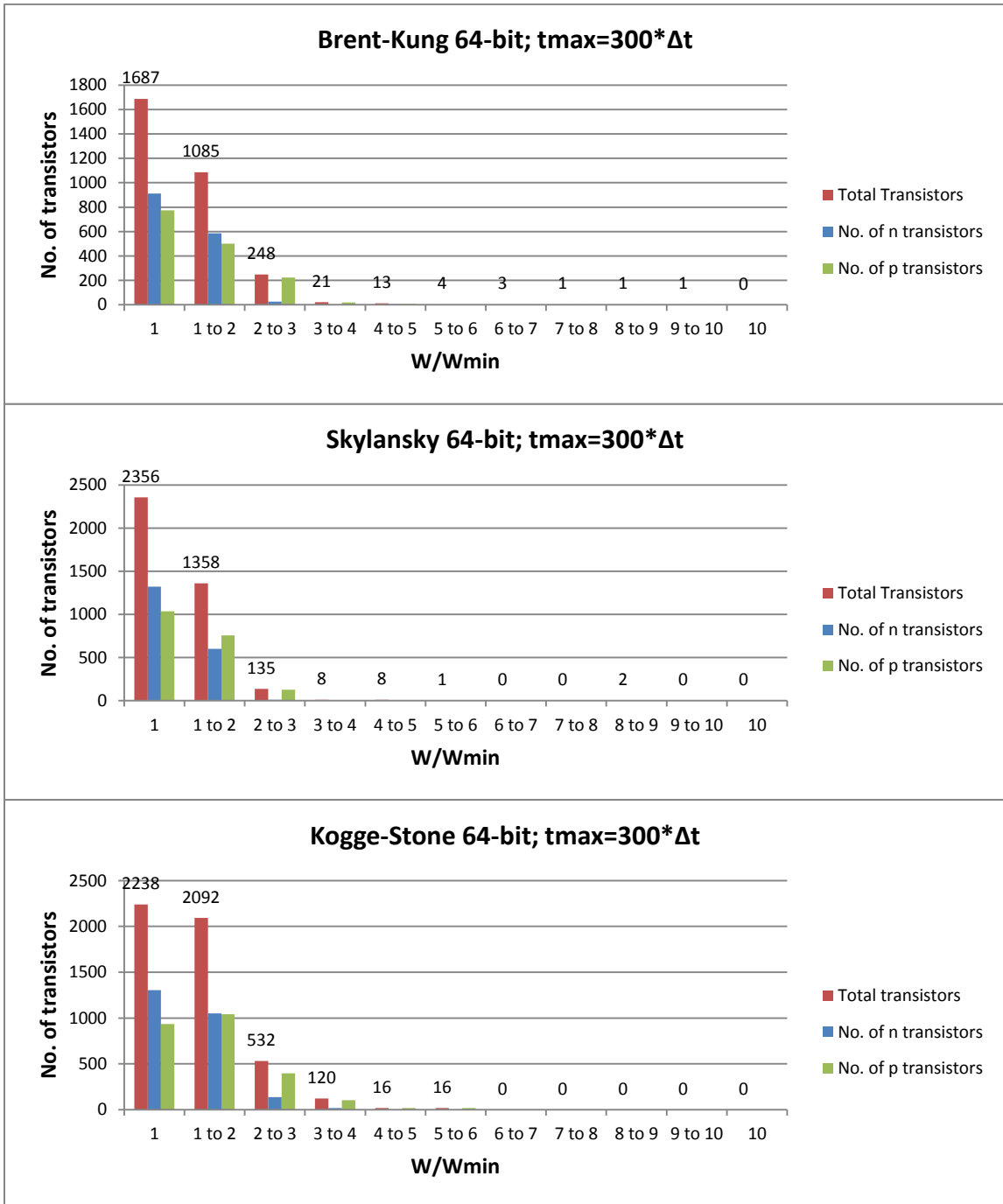


APPENDIX B

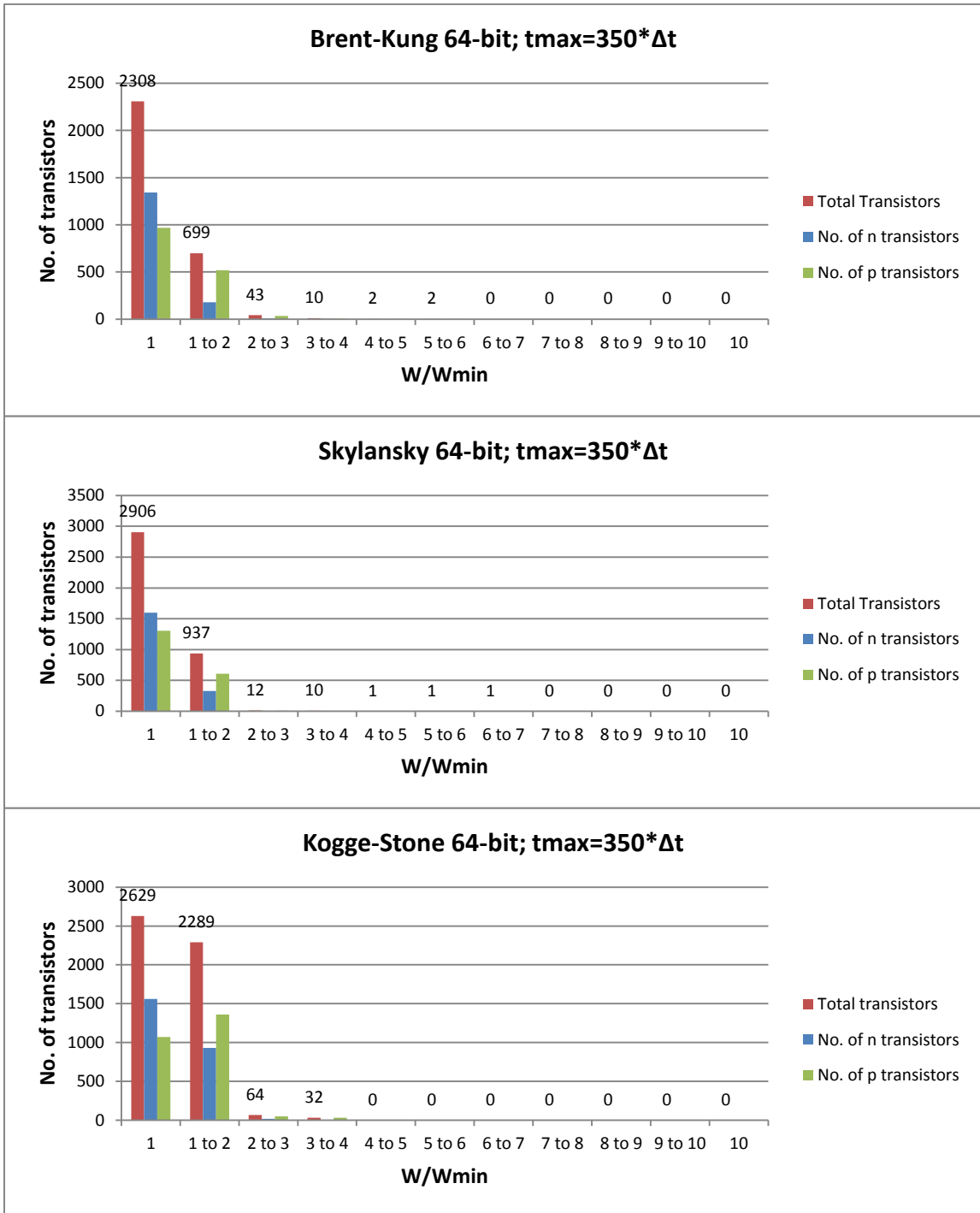
Histograms for 64-bit adder size for minimum delay case; $\Delta t=R*C$



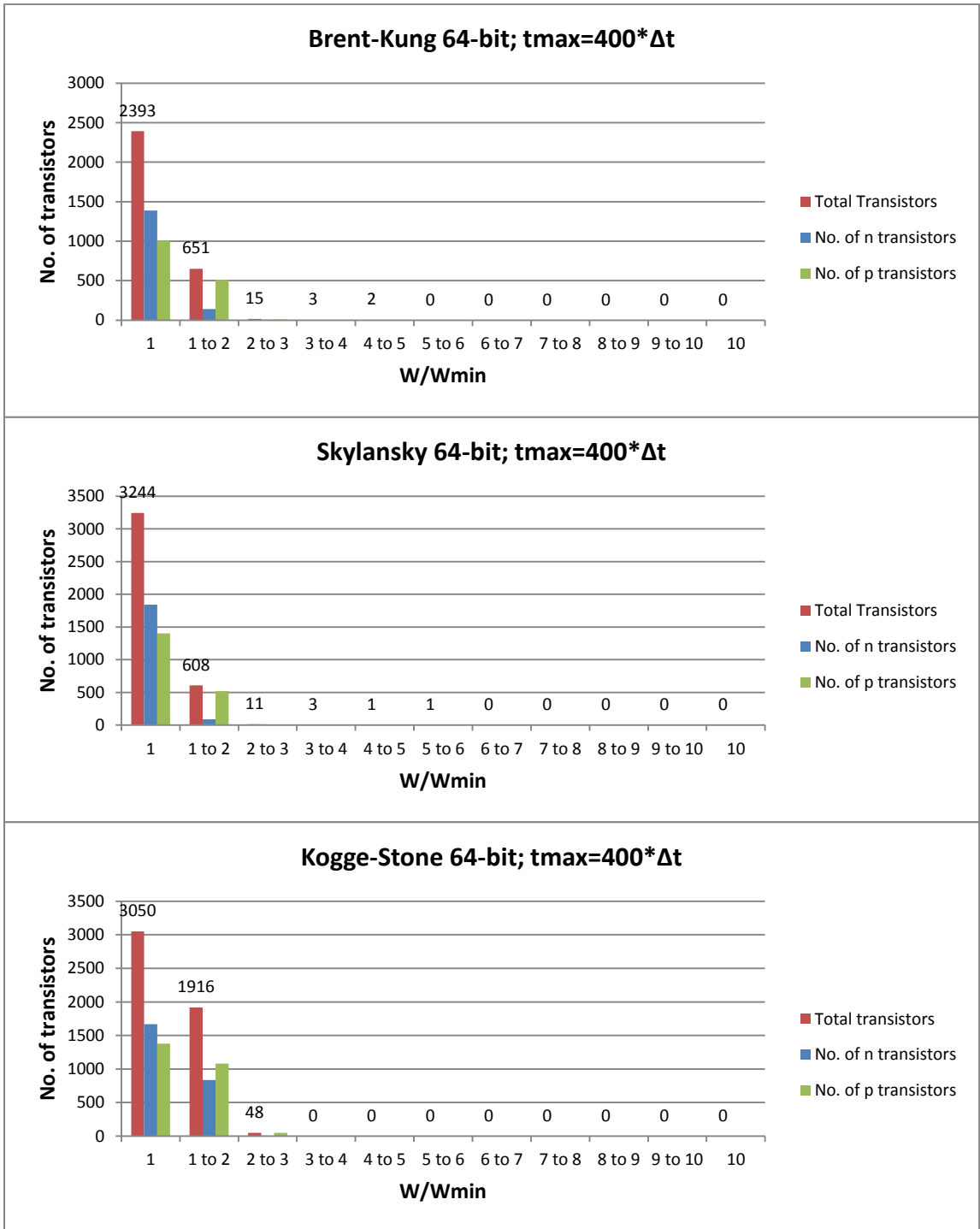
Histograms for 64-bit adder size with delay constraint of $t_{max}=300*\Delta t$, where $\Delta t=R*C$



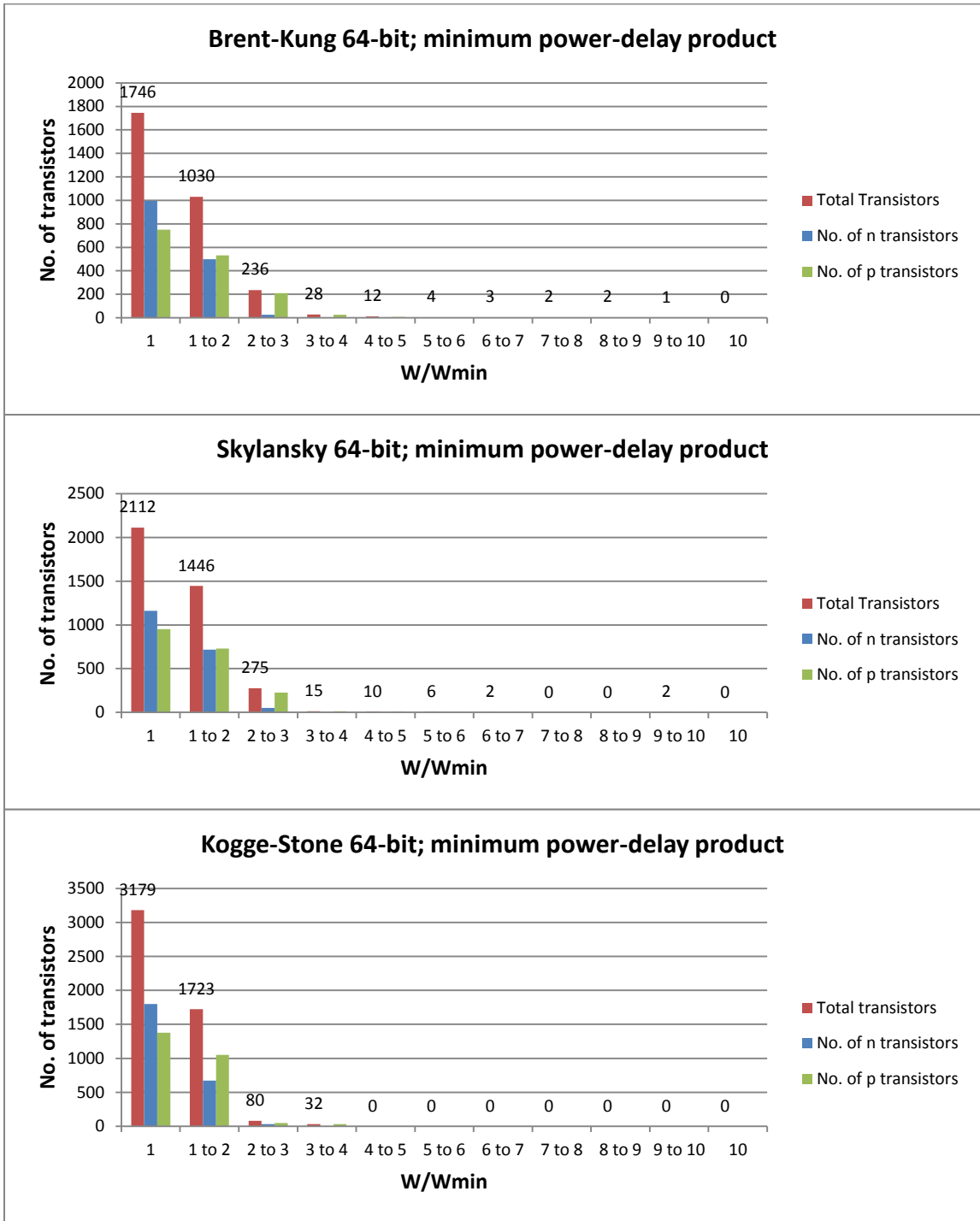
Histograms for 64-bit adder size with delay constraint of $t_{max}=350*\Delta t$, where $\Delta t=R*C$



Histograms for 64-bit adder size with delay constraint of $t_{max}=400*\Delta t$, where $\Delta t=R*C$

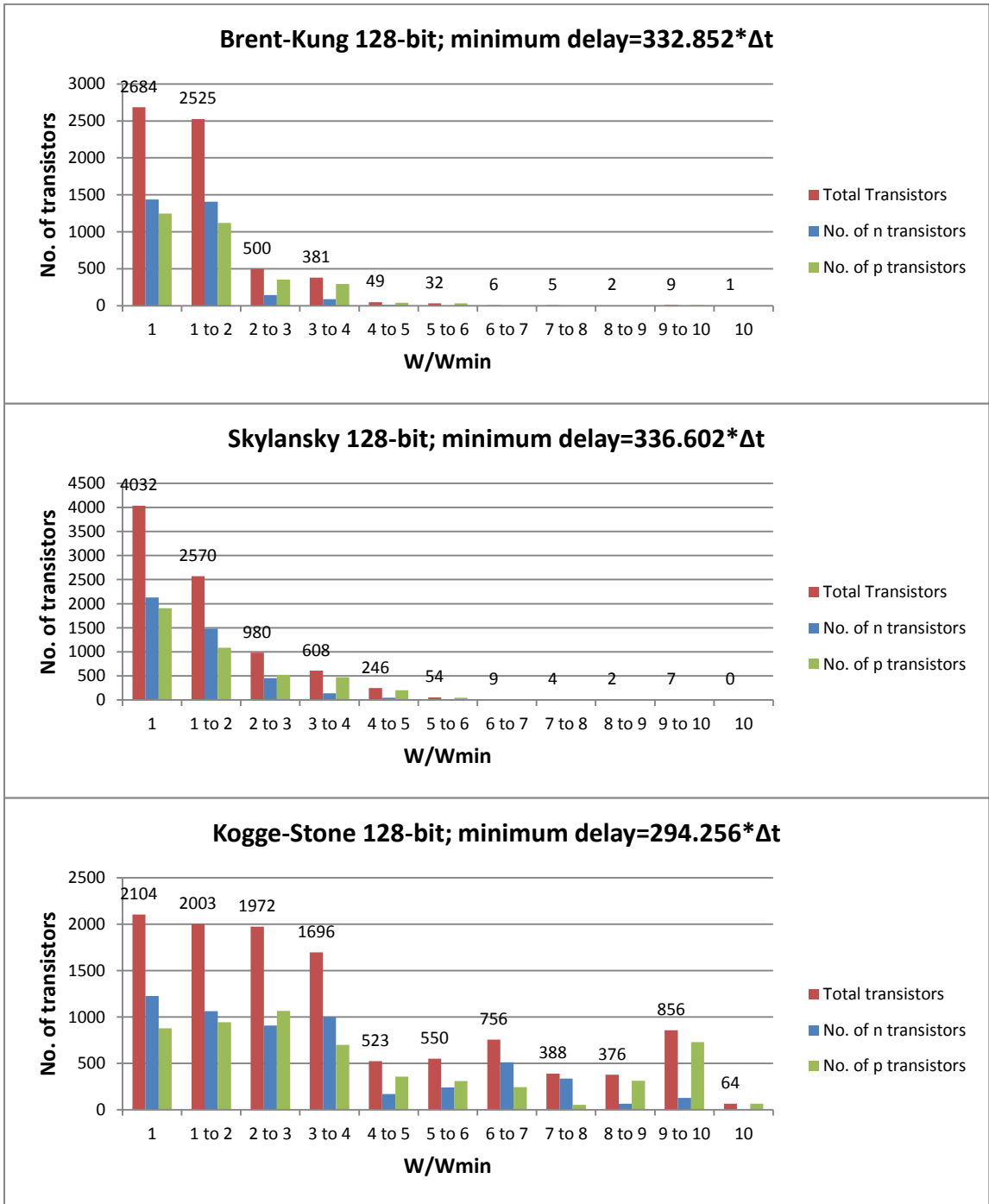


Histograms for 64-bit adder size for minimum power-delay product

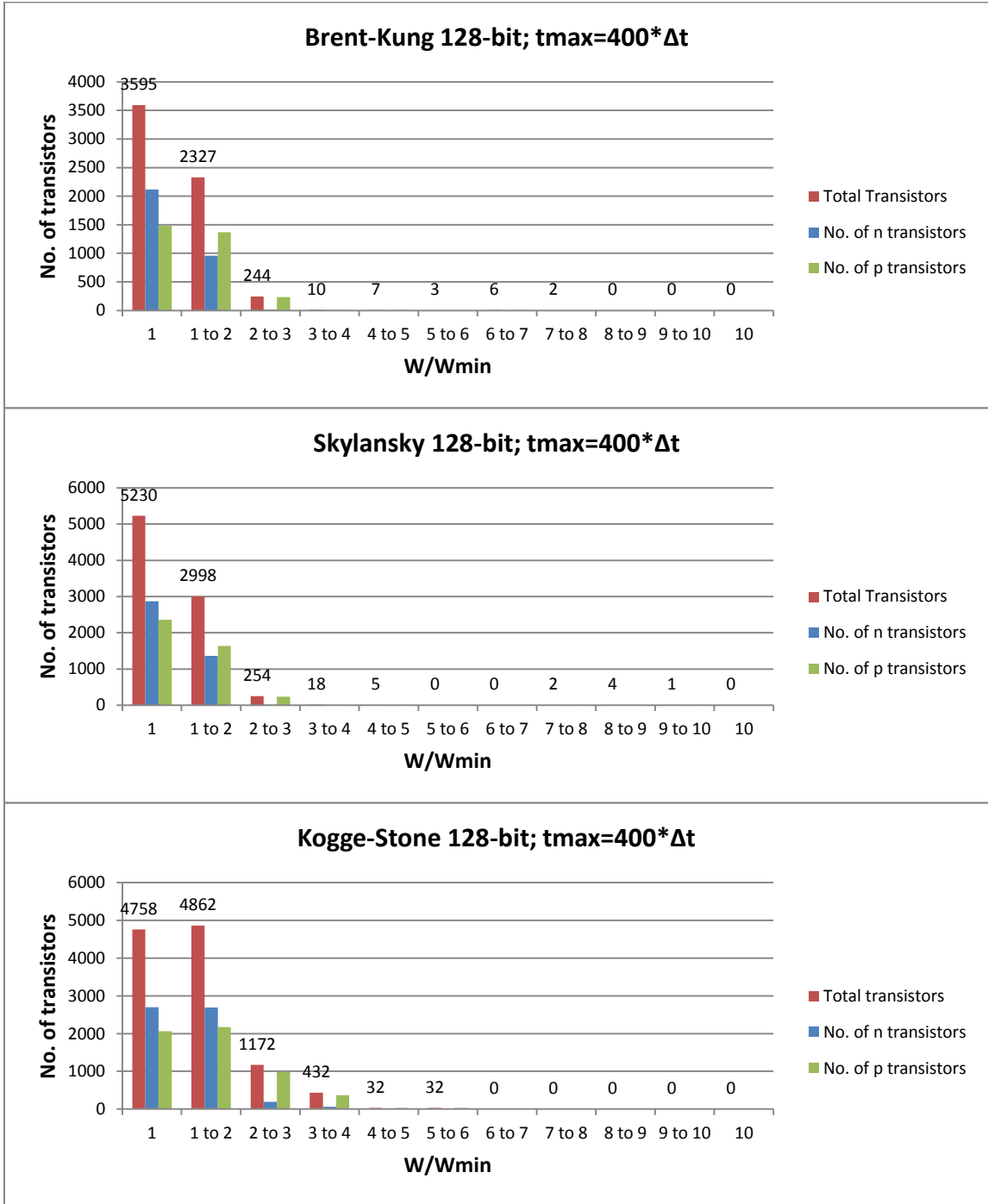


APPENDIX C

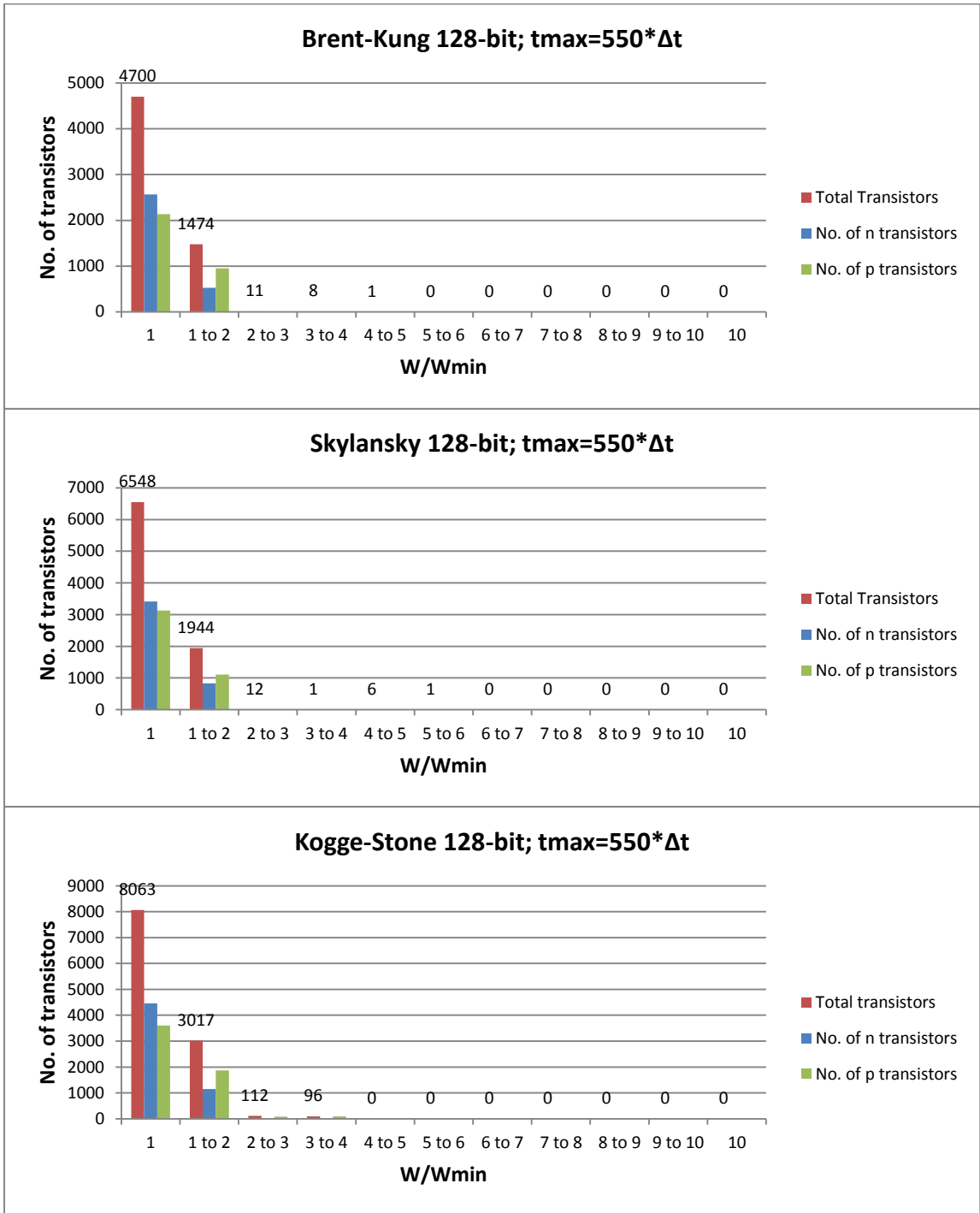
Histograms for 128-bit adder size for minimum delay case; $\Delta t=R*C$



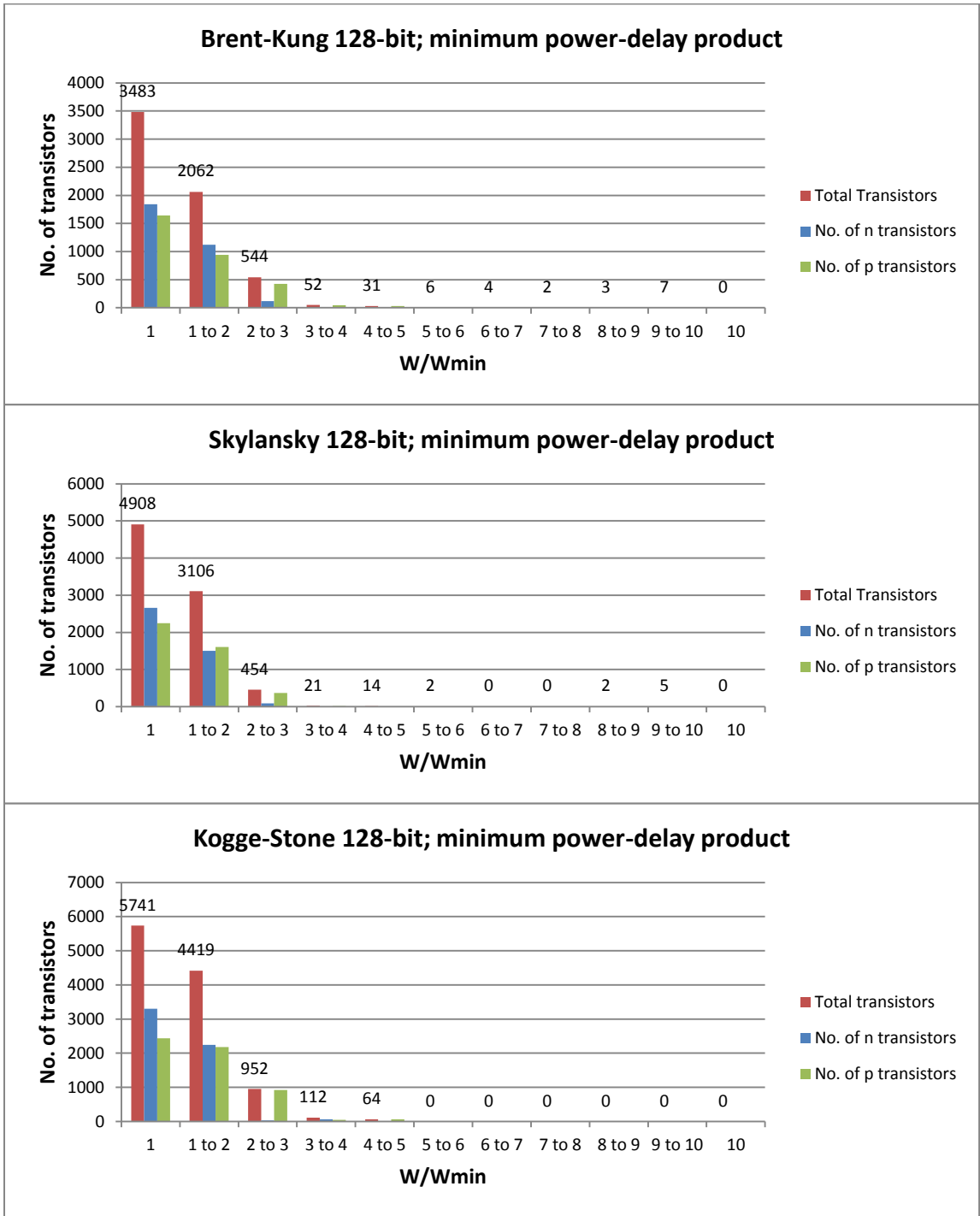
Histograms for 128-bit adder size with delay constraint of $t_{max}=400*\Delta t$, where $\Delta t=R*C$



Histograms for 128-bit adder size with delay constraint of $t_{max}=550*\Delta t$, where $\Delta t=R*C$



Histograms for 128-bit adder size for minimum power-delay product



VITA

Sunil Kumar Lakkakula

Candidate for the Degree of

Doctor of Philosophy

Thesis: CMOS CIRCUIT SPEED AND POWER OPTIMIZATION USING
SIMPLIFIED RC DELAY MODEL

Major Field: Electrical Engineering

Biographical:

Education:

Completed the requirements for the Doctor of Philosophy in Electrical Engineering at Oklahoma State University, Stillwater, Oklahoma in May, 2015.

Completed the requirements for the Master of Science in Electrical Engineering at Oklahoma State University, Stillwater, Oklahoma in December, 2009.

Completed the requirements for the Bachelor of Technology in Electrical and Electronics Engineering at Acharya Nagarjuna University, Guntur, Andhra Pradesh, India in April, 2007.

Experience:

Graduate Research Associate, Oklahoma State University.

Professional Memberships:

Member, International Society of Automation

Member, Golden Key