Sub-5.5 FO4 Delay CMOS 64-bit Domino/Threshold Logic Adder Design

Peter Celinski[a,c] Sorin D. Cotofana[c] S. Al-Sarawi[a] and Derek Abbott[a,b]

[a] Centre for High Performance Integrated Technologies & Systems (CHiPTec)
Department of Electrical & Electronic Engineering
The University of Adelaide, SA 5005,
Australia.

[b] Centre for Biomedical Engineering (CBME)
The University of Adelaide, SA 5005,
Australia.

[c] Computer Engineering Group
Electrical Engineering Department
Delft University of Technology
Mekelweg 4, 2628 CD Delft
The Netherlands.

*

ABSTRACT

This paper presents the design of a CMOS 64-bit adder using threshold logic gates based on Logical E ort (LE) transistor level delay estimation. The adder is a hybrid design, consisting of domino logic and the recently proposed Charge Recycling Threshold Logic (CRTL). The delay evaluation is based LE modeling of the delay of the domino and CRTL gates. From the initial estimations, the 8-bit sparse carry look-ahead/carry-select scheme has a delay of less than 5.5 FO4 (fan-out-of-four inverter delay), which is more than 1 FO4 delay faster than any previously published domino design.

1. INTRODUCTION

Addition is one of the most critical operations performed by VLSI processors. Adders are used in floating-point arithmetic units, ALUs, memory addressing and program counter-update. The main requirements are speed, power dissipation and low area.

Threshold logic (TL) was introduced over four decades ago, and over the years has promised much in terms of reduced logic depth and gate count compared to conventional logic-gate based design. Lack of efficient CMOS realizations has meant that TL has thus far had little impact on VLSI. Efficient TL gate realizations have recently become available, and a small number of applications based on TL gates[1–3] have demonstrated its ability to achieve high operating speed and significantly reduced area.

Additionally, it was suggested by Padure[4] that combined TL and conventional static-CMOS logic can reduce the delay compared to a pure TL or pure static-CMOS approach for the case of parallel (population) counters. However, to date no large scale arithmetic building blocks for processors have been designed using TL. We address this issue by proposing a high speed 64-bit TL/domino based adder.

P. Celinski, corresponding author: celinski@eleceng.adelaide.edu.au, S. D. Cotofana: sorin@ce.et.tudelft.nl, Said Al-Sarawi: alsarawi@eleceng.adelaide.edu.au, D. Abbott: dabbott@eleceng.adelaide.edu.au

Microelectronics: Design, Technology, and Packaging, edited by Derek Abbott, Kamran Eshraghian, Charles A. Musca, Dimitris Pavlidis, Neil Weste, Proceedings of SPIE Vol. 5274 (SPIE, Bellingham, WA, 2004) · 0277-786X/04/$15 · doi: 10.1117/12.524776

The delay analysis method used in this work enables the comparison of various adder topologies based on logical e ort. Another motivator for this approach is the desire to avoid the common and largely unsatisfactory presentation of circuit performance results commonly found in the literature in the form of delay numbers with insufficient information to allow comparison across di erent process technologies and loading conditions.

This paper presents the design of and critical path delay evaluation of a high speed hybrid CRTL-domino adder. The aim is to use a relatively quick method to determine the fastest possible 64-bit adder without venturing into CAD tool complexity, taking into account the advantages and limitations of CRTL. Simulations are used to verify the accuracy of the delay model estimate. The proposed 64-bit adder has a simulated critical path delay of 5.3 FO4 in a 0.35 μm process.

We begin in Section 2 by giving a brief overview of threshold logic. This is followed by a description of CRTL in Section 3. Section 4 briefly reviews Logical E ort and the delay model for CRTL gates is developed in Section 5. The circuit design examples are presented and evaluated in Section 6. Finally a conclusion and suggestions for future work are given in Section 7.

2. OVERVIEW OF THRESHOLD LOGIC

Threshold logic emerged in the early 1960's as a generalized theory of switching logic and includes conventional Boolean logic as its subset. The output of a threshold logic gate is the thresholded linear weighted sum of the inputs, which is one of the key operations performed by neurons.

The gate takes n binary inputs x_1,x_2,...,x_n and produces a single binary output y, as shown in Fig. 1.

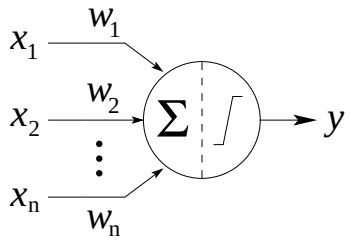

Figure 1. Threshold Gate Model

The Boolean function computed by such a gate is called a threshold function and it is specified by the gate threshold T and the weights w_1,w_2,...,w_n, where w_i is the weight associated with the i^{th} input variable x_i. The output y is given by (all operators algebraic):

$$y = \begin{cases} 1, & \text{if} \quad \sum_{i=1}^{n} w_i x_i \geq T \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

This function can be written in a more compact form using the sgn notation:

$$y = \text{sgn}\left\{ \sum_{i=1}^{n} w_i x_i - T \right\}, \tag{2}$$

where the sgn function is defined as follows, $\text{sgn}(x) = 1$ if $x \geq 0$ and $\text{sgn}(x) = 0$ if $x < 0$.

A TL gate can be programmed to realize many distinct Boolean functions by adjusting the threshold T and/or the weights w_i. For example, an n-input TL gate with $T = n$ will realize an n-input AND gate and by setting $T = n/2$, the gate computes a majority function. This versatility means that TL o ers a significantly increased computational capability over conventional AND-OR-NOT logic. Significantly reduced area and increased circuit speed can therefore potentially be obtained, especially in applications requiring a large number of input variables, such as computer arithmetic. This is illustrated by a number of practical results,[5,4,6] which suggest advantages of TL over conventional Boolean logic.

3. CHARGE RECYCLING THRESHOLD LOGIC

The realization for CMOS threshold gates presented by Celinski[1] and used in the design of TL circuits in this work is now described. Fig. 2 shows the circuit structure. The sense amplifier (cross coupled transistors M1-M4) generates output Q and its complement Q_b. Precharge and evaluate is specified by the enable clock signal E and its complement $\bar{E}$. The inputs x_i are capacitively coupled onto the floating gate of M5, which has potential ϕ, and the threshold is set by the gate voltage t of M6. The potential ϕ is given by

$$\phi = \frac{\sum_{i=1}^{n} C_i x_i}{C_{\mathrm{tot}}}, \tag{3}$$

where C_{tot} is the sum of all capacitances, including parasitics, at the floating gate of M5. Weight values are thus realized by setting capacitors C_i to appropriate values. Typically, in CMOS technology these capacitors are implemented between the polysilicon 1 and polysilicon 2 layers where available, or other dedicated linear layers for linear capacitors.

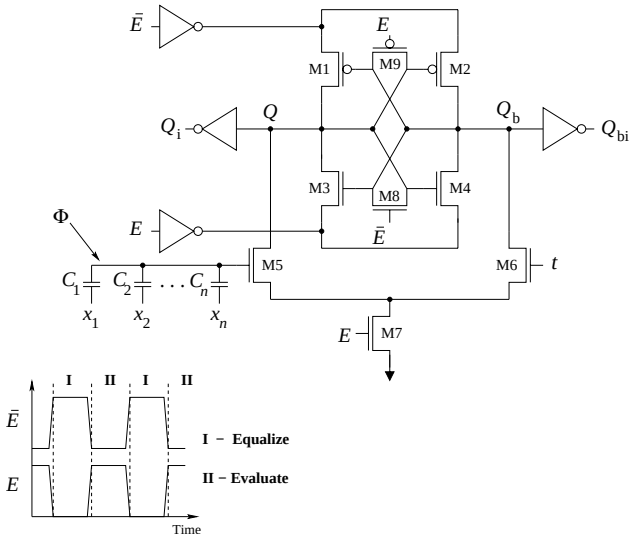

Figure 2. The CRTL gate circuit and Enable signals.

The enable signal E controls the precharge and activation of the sense circuit. The gate has two phases of operation, the equalize-phase and the evaluate-phase. When $\bar{E}$ is high the output voltages are equalized. When E is high, the outputs are disconnected and the di erential circuit (M5-M7) draws di erent currents from the formerly equalized nodes Q and Q_b. The sense amplifier is activated after the delay of the enable inverters and amplifies the di erence in potential now present between Q and Q_b, accelerating the transition. In this way the circuit structure determines whether the weighted sum of the inputs, ϕ, is greater or less than the threshold, t, and a TL gate is realized. The gates may be pipelined in a self-timed manner as described by Kong for the Asynchronous Sense Di erential Logic family.[7] Extensive Monte Carlo, varied voltage and temperature operating point and process corner variation simulations have shown the gate operates reliably at high speed.[1]

4. LOGICAL EFFORT

The technique of Logical E ort (LE) is a design methodology for estimating the delay of CMOS logic circuits, implementing a given logic function.[8,9] It provides a means to determine the best number of logic stages, including bu ers, required to implement a given logic function, and to size the transistors to minimize the delay.

Logical e ort is based on a reformulation of the conventional RC model of CMOS gate delay which separates the e ects on delay of gate size, topology, parasitics and load. The relative simplicity of the method compared

to other delay modeling techniques and sufficient accuracy allow it to be used early in the design process to evaluate alternative circuits.

The total delay of a gate, d, is comprised of two parts, an intrinsic parasitic delay p, and an e ort delay, f, driving the capacitive load. The parasitic delay is largely independent of the transistor sizes in the gate, since wider transistors which provide increased current have correspondingly larger di usion capacitances. The e ort delay in turn depends on two factors, the ratio of the sizes of the transistors in the gate to the load capacitance and the complexity of the gate. The former term is called *electrical e ort*, h, and the latter is called *logical e ort*, g. Electrical e ort is defined as

$$h = \frac{C_{\text{out}}}{C_{\text{in}}}, \tag{4}$$

where C_{out} and C_{in} are the gate load capacitance and input capacitance, respectively. The logical e ort, g, characterizes the gate complexity, and is defined as the ratio of the input capacitance of the gate to the input capacitance of an inverter that can produce equal output current. Alternatively, the logical e ort describes how much larger than an inverter the transistors in the gate must be to be able to drive loads equally well as the inverter. By definition an inverter has a logical e ort of 1.

The delay of a single logic gate can be expressed as

$$d = gh + p. \tag{5}$$

This delay is in units of τ, which is, the delay of an inverter driving an identical copy of itself, without parasitics. This normalization enables the comparison of delay across di erent technologies. The product gh is called the gate or stage e ort.

The considerations so far apply to single gates, but may be extended to the treatment of delay through a path. Using uppercase to denote path parameters, the path electrical e ort, H, is similarly defined as the ratio of the path load capacitance to the path input capacitance. The path logical e ort, G, is given by

$$G = \prod g_i, \tag{6}$$

where the subscript i indexes the logic gates along the path. The e ect of fanout, which causes some of the available drive current to be directed o the path being analyzed, is accounted for by considering the branching e ort, b, which is defined as

$$b = \frac{C_{\text{on-path}} + C_{\text{o -path}}}{C_{\text{on-path}}}, \tag{7}$$

and the path branching e ort is given by

$$B = \prod b_i. \tag{8}$$

Finally, the path e ort, F, is given by the product of the path logical e ort, the path branching e ort and the path electrical e ort

$$F = GBH. \tag{9}$$

The path delay, D, is the sum of the delays of each of the gate stages in the path, d_i, and consists of the *path e ort delay*, D_F, and the *path parasitic delay*, P,

$$\begin{aligned} D &= \sum d_i \\ &= D_F + P \\ &= \sum g_i h_i + \sum p_i. \end{aligned} \tag{10}$$

It can be shown that the path delay is minimized when each stage in the path bears the same stage e ort and the minimum delay is achieved when the stage e ort is

$$f_{\min} = g_i h_i = F^{1/N}. \tag{11}$$

This leads to the main result of logical e ort, which is the expression for minimum path delay

$$D_{\min} = NF^{1/N} + P. \tag{12}$$

To equalize the e ort borne by each stage in the path, the transistor sizes in each logic gate must be chosen according to the electrical e ort given by Eqn. (11)

$$h_{i,\min} = \frac{F^{1/N}}{g_i}. \tag{13}$$

This allows us to calculate the input capacitance and hence transistor size (width, assuming minimum length transistors) by applying the transformation

$$C_{\text{in},i} = \frac{g_i C_{\text{out},i}}{f_{\min}}. \tag{14}$$

This input capacitance is distributed among the transistors within the gate connected to the input.

The preceding steps dictate how to size the gates along a path for minimum delay, taking into account the di ering complexity of the gates as given by the logical e ort. Equally important is the selection of the correct number of stages. It has been shown[9] that for static CMOS logic, the near optimal stage e ort is approximately 4, and stage e orts from 2.4 to 6 give delays within 15% of the minimum. Hence the best number of stages is approximately

$$N \approx \log_4 \text{F}. \tag{15}$$

For domino logic the optimal stage e ort is 2 to 2.75.[9]

To minimize delay, the design should use the correct number of stages of logic and gates with low logical e ort and parasitic delay. Path design may involve iteration, because the path's logical e ort depends on the topology of individual gates, but the best number of stages is not known without knowing the path e ort.

It has been shown that the delay value predicted by Eqn. (5) di ered from simulation results on average by over 20% for a range of gates,[10] mainly as a result of the impact on delay time of the input transition times. However, the accuracy of the delay predicted by Eqn. (5) can be improved by calibrating the model by simulating the delay as a function of load (electrical e ort) and fitting a straight line to extract τ, the inverter parasitic delay, p_{inv}, and the logical e ort, g. We will use this technique to develop a calibrated logical e ort based model for the delay of the CRTL gates.

5. MODELING CRTL DELAY

We begin by providing a set of assumptions that will simplify the analysis, a proposed expression for the worst case delay of the CRTL gate and a derivation of the model's parameters. The model is then applied to two practical circuit examples. The method described below may similarly be applied to other sense amplifier based linear threshold gates.

5.1. Notation and Assumptions

The TL gate is assumed to have n logic inputs (fanin), the total number of gate inputs connected to logic one is denoted by N, and T is the threshold of the gate. The potential of the gate of transistor M6, t, in Fig. 2 is given by

$$t = \frac{T}{n} \times V_{dd}. \tag{16}$$

In the worst case, the voltage ϕ in Eqn. (3) takes the values

$$\phi = t \pm \frac{\delta}{2} \tag{17}$$

Table 1. Delay parameters of the 0.35 μm, 3.3 V, 4M/2P process at 75°C.

τ	p_{inv}	FO4 delay
40 ps	1.18	207 ps

where δ is given by

$$\delta = \frac{V_{dd}}{n}. \tag{18}$$

Eqn. (17) expresses the worst case (greatest delay) condition where the di erence between ϕ and t is minimal, ie. the step voltage generated by the sum of inputs with respect to the threshold voltage is smallest. The value of $\phi = t - \delta/2$ corresponds to the rising and falling edges of the nodes Q and Q_b, respectively, in Fig. 2, and conversely for $\phi = t + \delta/2$.

The gate inputs are assumed to have unit weights, ie. $w_i = 1$, since the delay depends only on the value of N and T. Also, without loss of generality, we will assume positive weights and threshold, since negative weights may easily be accommodated in the di erential structure of the gate by using a network of input capacitors connected to the gate of M6.

Since the gate is clocked, we will measure delay from the clock E to Q_i-Q_{bi}. Specifically, delay will be measured as the average of the 50% point on two falling transitions of E to the 50% points on the corresponding falling and rising edges of Qi and Q_{bi}. Generally, the delay will depend on the threshold voltage, t, the step size, δ, and the capacitive output load on Q_i and Q_{bi}. To simplify the analysis, we will fix the value of t at 1.5 V. This value is close to the required gate threshold voltage in typical circuit applications. Therefore the worst case delay depends only on the fan-in and gate loading, and allows us to propose a model based on expressions similar to those for conventional logic based on the theory of logical e ort.

5.2. Formulation of the Model and Parameter Extraction

The delay of the CRTL gate may be expressed as Eqn. (19). This equation gives the total delay of the sense amplifier and the bu er inverters connected to Q and Q_b, and depends on the load, h, and the fanin, n, as follows

$$d_{E\to Qi} = \{g(n)h + p(n)\}\tau. \tag{19}$$

The load, h, is defined as the ratio of load capacitance on Q_i (we assume the loads on Q_i and Q_{bi} are equal) and the CRTL gate unit weight capacitance. Both logical e ort and parasitic delay in Eqn. (19) are a function of the fan-in.

To determine the values of the parameters in Eqn. (19), we first determine the values of the parasitic delay of an inverter, p_{inv}. From Eqn. (5), the inverter delay is $d = \tau(gh + p_{\text{inv}})$, where by definition $g = 1$ for an inverter. To obtain the values of τ and p_{inv}, we may measure from HSPICE simulations the inverter delay for various values of electrical e ort h, and plot the delay versus h. The slope of this straight line gives the value of τ and the $h = 0$ axis intercept gives the product of τ and p_{inv}.

The delay parameters for the industrial 0.35 μm process used to obtain the simulation results presented here and the simulated FO4 (fan-out of four) inverter delay are given in Table 1. The value of τ is found to be 40 ps.

The values of g and p in Eqn. 19 were extracted by linear regression from simulation results for a range of fanin from $n = 2$ to 60 while the electrical e ort was swept from $h = 0$ to 20 as shown in Table 2. The Table also gives the absolute gate delay for three values of electrical e ort, $h = 1$, 5 and 10, where h is the ratio of the load capacitance to the unit input capacitance of 3.37 fF.

By fitting a curve to the parameters g and p, CRTL gate delay may be approximated in closed form by

$$d_{E\to Qi} = \{(0.002n + 0.34)h + \ln(\text{n}) + 1.6\}\tau. \tag{20}$$

Table 2. Extracted CRTL gate logical e ort, g, parasitic delay, p, parameters for fanin $n = 2$ to 60 and $h = 0$ to 20 for the 0.35 μm, 3.3 V, 4M/2P process at 75°C and the gate delay normalized to FO4 for $h = 1, 5$ and 10.

n	g	p	$d_{E\to Qi}$, h=1	$d_{E\to Qi}$, h=5	$d_{E\to Qi}$, h=10
2	0.346	2.5	0.55	0.82	1.15
5	0.357	3.3	0.71	0.98	1.33
10	0.365	4.0	0.82	1.13	1.48
15	0.376	4.3	0.90	1.19	1.56
20	0.375	4.7	0.98	1.27	1.63
30	0.400	5.0	1.04	1.35	1.74
40	0.424	5.1	1.07	1.40	1.80
50	0.439	5.2	1.09	1.43	1.85
60	0.460	5.2	1.09	1.45	1.90

In order to use the parameters in Table 2 and Eqn. (20), it is necessary to compensate for the parasitic capacitance at the floating gate of M5. From Eqn. (3), the parasitic capacitance, C_p, contributes to a reduced voltage step, δ, on the gate of M5 in Fig. 2 with respect to the threshold voltage, t, as given by Eqn. (21),

$$\delta_{\rm e} \;= \left\{ \frac{\sum_{i=1}^{n} C_i}{\sum_{i=1}^{n} C_i + C_p} \right\} \delta_0, \tag{21}$$

where δ_0 is the nominal step given by Eqn. (18). This reduction in δ is equivalent to an increased value for the fanin. This e ective fanin, $n_{\rm e}$, is given by

$$n_{\rm e} \;= \left\{ \frac{\sum_{i=1}^{n} C_i + C_p}{\sum_{i=1}^{n} C_i} \right\} n_0 \ , \tag{22}$$

where n_0 is the number of inputs to the gate and n_{eff} is the value used to calculate the delay. Typically, for a large fanin CRTL gate, by far the major contribution to the parasitic capacitance will be from the bottom plate of the floating capacitors used to implement the weights. In the targeted process, this corresponds to the poly1 plate capacitance to the underlying n-well used to reduce substrate noise coupling to the floating node.

For example, for a 32 input CRTL gate with 3.37 fF poly1-poly2 unit capacitors ($4\mu{\rm m}^2$), the parasitic capacitance of poly1 to substrate is 29 fF, and the $\sum_{i=1}^{n} C_i = 32 \times 3.37 = 108$ fF. From Eqn. (22) the e ective fanin to be used in the delay calculation is $((108{+}29)/108) \times 32 \approx 41$.

6. 64-BIT ADDER DESIGN AND CRITICAL PATH DELAY ESTIMATION

The delay estimation based on logical e ort has been carried out for a number of high speed adders,[11,12] including the dynamic Kogge-Stone (D-KSA), the dynamic carry look-ahead (D-CLA), the dynamic Ling/conditional-sum (D-LCNSA) and Intel's Quarternary (D-QTA).[13] We extend this work to include CRTL based adders. For completeness, we also include comparison with the HP Ling adder,[14] Harris'[15] adder and the Output Prediction Logic adder developed by Sechen.[16]

6.1. 64-bit Adder Architecture

Our aim was to design the fastest adder possible using CRTL and domino logic. The selection of the adder architecture is heavily influenced by the availability of fast high fan-in CRTL gates. This leads us to use CLA (carry look-ahead) and CSA (carry-select) blocks.

The adders described by Na ziger[14] and Horowitz[15] are based on 4-bit CLA blocks, which is usually the optimal trade o between the depth of the CLA tree and the number of series transistors in a CMOS gate.[17] The carries in these adders are generated at 16-bit boundaries, requiring 16-bit sub-adders for carry-select blocks.

Increasing the number of bits handled by a CLA block to 8-bits results in fewer logic levels and a more regular design and layout.[16] This is impractical in conventional CMOS logic, since it requires 8 series transistors. We can, however, take advantage of the wide AND gates in CRTL. We obtain the regular structure shown in Fig. 3. In this scheme, the 64-bit input addends are divided into eight 8-bit blocks, and it has $\log_8 64$, or two levels of carry look-ahead. The Kogge-Stone scheme generates carries for each bit position, so no carry select is needed. The 4- and 8-bit block versions have depth $\log_4 64$=3 and $\log_8 64$=2, respectively. However, they consist of many more CLA blocks with significantly increased wiring and fanouts.

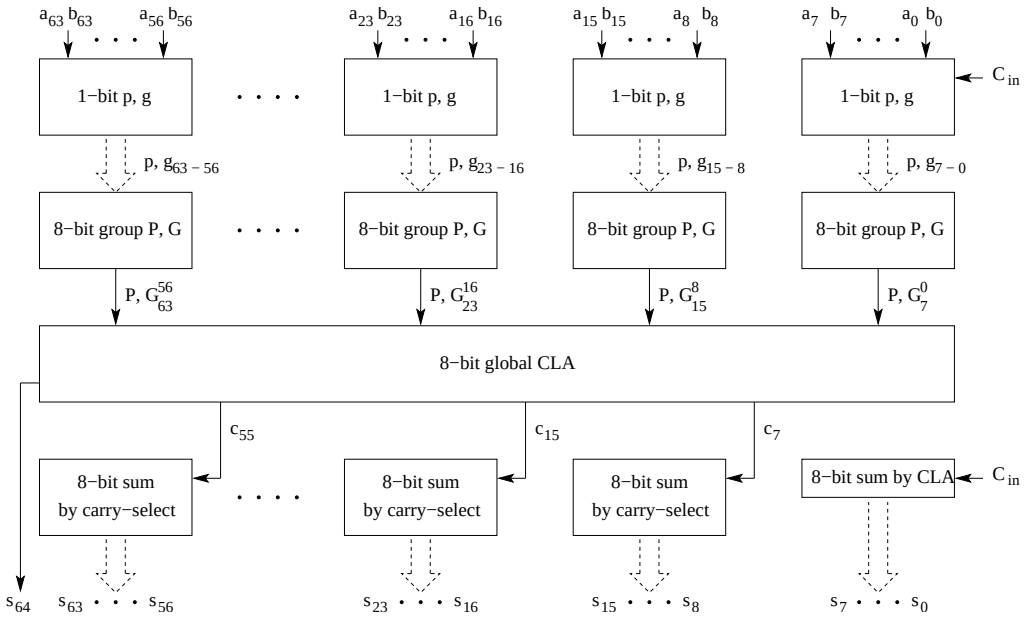

Figure 3. 64-bit adder block diagram.

The structure of the proposed adder is a sparse carry prefix tree. In the first layer, the bitwise propagate and generate signals, p_i, g_i, are formed, followed by the computation of eight pairs of 8-bit group generate and propagate signals P_j^{j-7}, G_j^{j-7} in the second layer. These are then assimilated in the global carry look-ahead block to generate the sum selection carries, c_7, c_{47}, ..., c_{55}, which select pre-computed 8-bit sums. These 8-bit adders are also based on carry look-ahead.

The CLA equations may be written as given as Eqns. (23)-(26). Each CLA level consists of an AND and an OR gate, which requires significantly lower sum of weights in the CRTL implementation that a single gate AND-OR implementation.[6] The six stage critical path of the 64-bit adder consists the domino-OR2 to generate p_7 (despite the lower logical e ort and parasitic delay, this gate has a higher fanout than g_0), CRTL AND8 and OR8 to generate G_7^0, CRTL AND8 and OR8 to generate c_{55} in the global CLA block and a 2:1 MUX to select the sum.

The bitwise propagate and generate signals are computed as follows

$$p_i = a_i + b_i \tag{23}$$

$$g_i = a_i \cdot b_i \ , \tag{24}$$

and from these the 8-bit block group propagate and generate signals are given by

$$P_7^0 = p_7 \cdot p_6 \cdot p_5 \cdot p_4 \cdot p_3 \cdot p_2 \cdot p_1 \tag{25}$$

Table 3. Normalized LE parameters of various gates in 0.35 μm technology, in units of $\tau = 40$ ps.

Gate Type	LE, (g)	Parasitic delay, p
Inverter	1	1.18
Hi-skew Inverter	0.7	1
dyn-NAND2	0.4	1.8
dyn-NOR2	0.3	1.4
2:1 static MUX	1.13	2.6

$$\begin{aligned} G_7^0 &= g_7 + p_7 \cdot g_6 + p_7 \cdot p_6 \cdot g_5 + \ldots \\ &\quad + p_7 \cdot p_6 \cdot p_5 \cdot p_4 \cdot p_3 \cdot p_2 \cdot p_1 \cdot g_0 \ . \end{aligned} \tag{26}$$

Finally the 8-bit block carry outputs are given by

$$c_7 = G_7^0 \ . \tag{27}$$

A similar expression to Eqn. 26 may be written for generating the global look-ahead carries.

6.2. Delay Estimation and Comparison

In addition to the delay model for CRTL discussed earlier, in order to evaluate the adder delay it is necessary to characterize the domino gates using HSPICE simulation for various output loads, according to the LE rules. Characterization of domino gates considers only the one transition of interest, which is the falling transition for the dynamic pull down and rising transition for the hi-skew static inverter. This is repeated for each of the gates, and the results are shown in Table 3. Note that the dynamic gates listed consist of the pull down path only, excluding the static inverter.

The delay of the critical path,s_{63}, dyn-OR2 $\rightarrow$ CRTL-AND8 $\rightarrow$ CRTL-OR8 $\rightarrow$ CRTL-AND8 $\rightarrow$ CRTL-OR8 $\rightarrow$ MUX2 is calculated using Eqns. (10) and (20) and Table 3. For the 8-input CRTL gates we use an n_e value of 10. In addition, we must consider the fan-out of 7 of the dyn-OR2 gate (which drives 7 unit weight CRTL inputs). The other gates have a unity electrical e ort. From Eqn. 12 the optimized delay of the two stage dyn-OR2 gate is therefore given by

$$\begin{aligned} d_{\mathrm{OR2,min}} &= NF^{1/N} + P \\ &= 2\{g_{\mathrm{NOR2}} \times g_{\mathrm{HS-Inv}} \times h_{\mathrm{HS-Inv}}\}^{0.5} + p_{\mathrm{NOR2}} + p_{\mathrm{HS-Inv}} \\ &= 2\{0.3 \times 0.7 \times 7\}^{0.5} + 1.4 + 1 \\ &= 4.8\tau \ . \end{aligned} \tag{28}$$

From this the critical path delay is calculated as follows

$$\begin{aligned} d_{s_{63}} &= d_{\mathrm{OR2,min}} + 4 \times d_{\mathrm{CRTL10}} + (gh + p)_{\mathrm{MUX2}} \\ &= 4.8 + 4 \times 4.26 + 1.13 + 2.6 \\ &= 25.6\tau \\ &= 4.9 \ \mathrm{FO4} \ . \end{aligned} \tag{29}$$

The proposed adder consists of 3653 transistors and 342 unit capacitors. The critical path was also simulated, including wiring capacitance estimations based on traversed CRTL and domino cell pitch, and the extracted gate layouts and the critical path delay thus obtained was 5.3 FO4. Note that the 207 ps FO4 delay is a very slow process corner for a drawn channel length of 0.4 μm, and is the fastest we had available, (Sechen[16] similarly reports 162 ps for the 0.25 μm process used in that work). It is therefore not surprising that the 930 ps delay

Table 4. Comparison of high speed 64-bit adders.

64-bit Adder	# Stages	Tech. μm	LE FO4	Sim. FO4
D-CLA[11]	14	0.18	11.1	13.6
D-LCNSA[11]	9	0.18	9.0	9.5
Intel D-QTA[13]	10	0.10	8.3	-
D-HCA[12]	10	0.10	8.26	-
D-KSA[11]	6	0.18	6.2	7.4
HP mod. Ling[14]	4	0.5	-	7
Harris[15]	-	0.6	-	6.4
OPL[16]	8	0.25	-	2.9
$\rightarrow$ This Work	6	0.35	4.9	5.3

for the 0.5 μm process reported by Na ziger[14] has an FO4 delay less than ours, especially if a faster process corner was used.

The FO4 delay comparison with eight other dynamic high speed adders is shown in Table 4, with the logical e ort estimate and simulated or measured delay values listed where available. The comparison suggests a significant delay speed improvement of almost 1.1 FO4 or 17% compared to Harris' agressive domino design. The OPL adder is included for completeness to acknowledge other novel circuit techniques, it has the significant drawback of requiring 8 clock phases which has significant power dissipation issues, in addition to the reduced noise margin of OPL gates. Table 4 also shows that delay is related to but not proportional to the number of gate levels on the critical path, so comparing delay estimates based on this simple metric is inconclusive.

7. CONCLUSIONS

A high speed 64-bit adder based on a hybrid carry look-ahead/carry-select scheme using Charge Recycling Threshold Logic and conventional domino logic has been proposed. The worst case critical path delay was shown to be significantly improved compared to previously proposed domino high-speed adders. The results show that by combining TL and conventional CMOS logic with the appropriate architectural strategy, relatively fast arithmetic circuits can be achieved. The results presented here suggest that the substantial delay improvement over domino justifies the added complexity of CRTL. The important issue of power dissipation has not been addressed, and is the subject of ongoing work.

ACKNOWLEDGMENTS

The support of the Australian Research Council and the Sir Ross and Sir Keith Smith Fund and the Delft University of Technology is gratefully acknowledged. P. Celinski would like to thank I. Tatomir for the many useful discussions.

REFERENCES

1. P. Celinski, J. F. López, S. Al-Sarawi, and D. Abbott, "Low power, high speed, charge recycling CMOS threshold logic gate," *IEE Electronics Letters* **37**, pp. 1067–1069, August 2001.
2. P. Celinski, S. D. Cotofana, and D. Abbott, "Area efficient, high speed parallel counter circuits using charge recycling threshold logic," in *Proc. IEEE International Symposium on Circuits and Systems*, vol. 5, pp. 233–236, May 2003.
3. M. Padure, S. Cotofana, and S. Vassiliadis, "A low-power threshold logic family," in *Proc. IEEE International Conference on Electronics, Circuits and Systems*, pp. 657–660, 2002.

4. M. Padure, S. Cotofana, and S. Vassiliadis, "High-speed hybrid Threshold-Boolean logic counters and compressors," in *Proceedings of the 45th IEEE International Midwest Symposium on Circuits and Systems*, pp. 457–460, 2002.
5. Y. Leblebici, H. Özdemir, A. Kepkep, and U. Çiliniroğlu, "A compact high-speed (31-5) parallel counter circuit based on capacitive threshold-logic gates," *IEEE JSSC* **31**, pp. 1177–1183, August 1996.
6. P. Celinski, J. F. López, S. Al-Sarawi, and D. Abbott, "Low depth carry lookahead addition using charge recycling threshold logic," in *Proc. IEEE International Symposium on Circuits and Systems*, pp. 469–472, (Phoenix), May 2002.
7. B.-S. Kong, J.-D. Im, Y.-C. Kim, S.J-Jang, and Y.-H. Jun, "CMOS di erential logic family with self-timing and charge-recycling for high-speed and low-power VLSI," *IEE Proceedings Circuits, Devices and Systems* **150**, pp. 45–50, February 2003.
8. I. Sutherland and B. Sproull, "Logical e ort: Designing for speed on the back of an envelope," in *Proceedings of the 1991 University of California Advanced Research in VLSI Conference*, C. H. Sequin, ed., pp. 1–16, MIT Press, 1991.
9. I. E. Sutherland, R. F. Sproull, and D. L. Harris, *Logical E ort, Designing Fast CMOS Circuits*, Morgan Kaufmann, 1999.
10. X. Y. Yu, V. G. Oklobdzija, and W. W. Walker, "Application of logical e ort on design of arithmetic blocks," in *Proceedings of 35th Annual Asilomar Conference on Signals, Systems and Computers*, pp. 872–874, November 2001.
11. H. Dao and V. G. Oklobdzija, "Application of logical e ort techniques for speed optimization and analysis of representative adders," in *Proc. Thirty-Fifth Asilomar Conferenceon Signals, Systems and Computers, 2001*, vol. 2, pp. 1666–1669, November 2001.
12. V. Oklobdzija, B. Zeydel, D. Hoang, S. Mathew, and R. Krishnamurthy, "Energy-delay estimation technique for high-performance microprocessor VLSI adders," in *Proceedings. 16th IEEE Symposium on Computer Arithmetic*, pp. 272–279, June 2003.
13. S. Mathew, M. Anders, R. Krishnamurthy, and S. Borkar, "A 4 GHz 130nm address generation unit with 32-bit sparse-tree adder core," in *Symposium on VLSI Digest of Technical Papers*, pp. 126–127, IEEE, 2002.
14. S. Na ziger, "A sub-nanosecond 0.5μm 64b adder design," in *International Solid State Circuits Conference Digest of Technical Papers*, pp. 362–363, IEEE, 1996.
15. M. Horowitz, "EE271 class notes (adders)." Stanford University, http://eeclass.stanford.edu/ee371/.
16. S. Sun, L. McMurchie, and C. Sechen, "A high-performance 64-bit adder implemented in output prediction logic," in *Proceedings Conference on Advanced Research in VLSI*, pp. 213–222, 2001.
17. A. Beaumont-Smith and C. C. Lim, "Parallel prefix adder design," in *Proceedings of the 15th IEEE Symposium on Computer Arithmetic*, pp. 218–225, (Vail, USA), June 2001.