A-DELTA: A 64-bit High Speed, Compact, Hybrid Dynamic-CMOS/Threshold-Logic Adder

Peter Celinski[1,3], Sorin D. Cotofana[3] and Derek Abbott[2]

[1] Centre for High Performance Integrated Technologies & Systems (CHiPTec),
[2] Centre for Biomedical Engineering,
The Department of Electrical and Electronic Engineering,
The University of Adelaide, SA 5005, Australia,
celinski@eleceng.adelaide.edu.au
[3] Computer Engineering Group, Electrical Engineering Department,
Delft University of Technology, Mekelweg 4, 2628 CD Delft, The Netherlands,
sorin@ce.et.tudelft.nl

Abstract. A high speed 64-bit dynamic adder, the Adelaide-Delft Threshold Logic Adder (A-DELTA), is presented. The adder is based on a hybrid carry-lookahead/carry-select scheme using threshold logic and conventional CMOS logic. A-DELTA was designed and simulated in a 0.35 μm process. The worst case critical path latency is 670 ps, which is shown to be on average 30% faster than previously proposed high speed Boolean dynamic logic adders while at the same time reducing the transistor count on average by over 30% compared to the same adders.

1 Introduction

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 physical realizations has meant that TL has, over the years, had little impact on VLSI. Efficient TL gate realizations have recently become available, and a small number of applications based on TL gates have demonstrated its ability to achieve high operating speed and significantly reduced area [1–3]. Additionally, it was suggested in [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, no large scale arithmetic building blocks for processors have been designed using TL. We address this issue by proposing a 64-bit TL adder.

Addition is one of the most critical operations performed by VLSI processors. This paper proposes a 64-bit adder using a two fold hybrid scheme, carry-lookahead/carry-select and mixed Charge Recycling Threshold Logic (CRTL) and dynamic-CMOS logic. Our aim is to design the fastest possible dynamic 64-bit adder, taking into account the advantages and limitations of CRTL. The proposed 64-bit adder has a simulated critical path delay of 670 ps in a 0.35 μm process.

J. Mira (Ed.): IWANN 2003, LNCS 2687, pp. 73-80, 2003.

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 develops the proposed mixed carry-lookahead/carry-select scheme using CRTL/dynamic-CMOS for 64-bit addition and in Section 5 results of the proposed 64-bit addition scheme are presented. Finally a brief conclusion is given in Section 6.

2 Threshold Logic

A threshold logic gate is functionally similar to a hard limiting neuron. The gate takes n binary inputs x_1, x_2,..., x_n and produces a single binary output y. A linear weighted sum of the binary inputs is computed followed by a thresholding operation.

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 corresponding to the i^{th} input variable x_i. The binary output y is given by

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

A TL gate can be programmed to realize many distinct Boolean functions by adjusting the threshold T. For example, an n-input TL gate with unit weights and $T = n$ will realize an n-input AND gate and by setting $T = n/2$, the gate computes a majority function. This versatility means that TL offers 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 indicated 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 in [2] and used in the proposed adder is now described. Fig. 1 shows the circuit structure. The sense amplifier (cross coupled transistors M1-M4) generates output y and its complement $\bar{y}$. 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 ϕ of M5, and the threshold is set by the gate voltage T of M6. The potential ϕ is given by $\phi = \sum_{i=1}^{n} C_i x_i / C_{tot}$, where C_{tot} is the sum of all capacitances, including parasitics, at the floating node. 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.

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

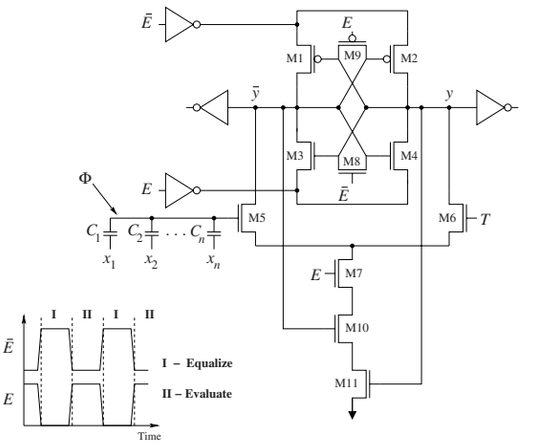

Fig. 1. The CRTL gate circuit and Enable signals.

outputs are disconnected and the differential circuit (M5-M7, M10, M11) draws different currents from the formerly equalized nodes y and $\bar{y}$. The sense amplifier is activated after the delay of the enable inverters and amplifies the difference in potential now present between y and $\bar{y}$, 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. Transistors M10 and M11 turn off the differential circuit after evaluation is completed to reduce the power dissipation. The gate was shown to reliably operate at high speed.

4 64-bit Addition Using CRTL/dynamic-CMOS

The underlying approach for the proposed 64-bit adder is to utilize at the circuit level, a hybrid scheme of combined conventional CMOS logic and TL. The goal is to propose an optimized design of a 64-bit adder using a hybrid CRTL/dynamic-CMOS approach.

At the architecture level, the design is structured to optimize speed based on the advantages and limitations of the logic gates used (static and dynamic CMOS and CRTL). A combination of anticipated-computation, carry-lookahead, and carry-select is used. At the implementation level within the blocks comprising the 64-bit adder, CRTL is used to achieve the minimum critical path delay, and where possible, static and dynamic CMOS are used to reduce the area.

The block diagram of the proposed 64-bit adder is depicted in Fig. 2. In this scheme, the 64-bit input addends are divided into four 16-bit adder blocks. Each 16-bit block generates a carry signal. These carry signals form the inputs of the global carry-lookahead unit which generates the select signals, c_{31}, c_{47} and c_{64}. These select signals are used to select the appropriate 16-bit final sums using three 32:16 multiplexers. The critical path of the 64-bit adder consists of the

generation of the carry-outputs of one 16-bit block, the global carry-lookahead unit which selects the most significant 16-bit anticipated sum and one 32:16 MUX delay.

The three anticipated sums are computed in the 16-bit blocks using pairs of 16-bit adders with c_{in}=0 and c_{in}=1. The 16-bit anticipated sums must be ready before the global carries arrive to perform the selection. It is this balance of quickly computing the carries out of the 16-bit adders and the global carries considering the maximum fan-in and delay of CRTL which dictates a partition of the 64-bit inputs into four 16-bit wide blocks.

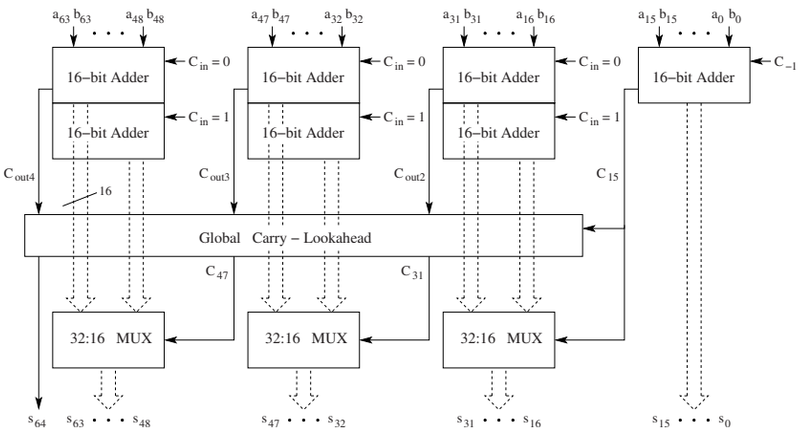

Fig. 2. A-DELTA block diagram.

To achieve the lowest critical path delay, the computation of the carry-output of each 16-bit adder and the global carries must be as fast as possible. In theory, it is possible to compute the carry signal for such a 16-bit block in one TL gate using the scheme first presented in [7]. However, this leads to a requirement for the sum of weights in the TL gate of $2^{17} - 2$ which is prohibitively large for the state-of-the-art CRTL. For this reason, the 16-bit carry computation is performed in two levels of TL gates, as discussed in the following section.

4.1 Design of the 16-bit Adders

The design of the 16-bit adders uses carry lookahead. This is a well-known technique for designing logarithmic depth addition networks. To design the high speed 16-bit adder, the high fan-in capability of CRTL is utilized to implement carry-lookahead prefix-cells which compute group carry-generate and group carry-propagate signals from a large number of input bits.

The design of the CRTL prefix-cells is influenced by the maximum sum of weights of CRTL. This is limited by the available precision with which the capacitive weights are implemented and, more significantly, the resolution of the

sense amplifier under the influence of process mismatch and noise. The maximum sum of weights sets the minimum voltage which is required to be resolved by the sense amplifier in the CRTL gate. For this reason we limit the sum of weights to approximately 30, and use this as the basis on which to optimize the critical path delay of the 16-bit adder. This, as will be shown, leads to the CRTL prefix-cells having 4-bit word inputs.

The computation of carries, c_j, can be expressed in prefix notation in terms of group-generate, G_j^i and group-propagate, P_j^i, signals at each bit position j as follows [8]:

$$G_j^i = \begin{cases} a_j \cdot b_j, & \text{for} \quad i = j \\ G_j^k + P_j^k \cdot G_{k-1}^i, & \text{for} \quad 15 \geq j \geq k > i \geq 0 \end{cases} \tag{2}$$

$$P_j^i = \begin{cases} a_j + b_j, & \text{for} \quad i = j \\ P_j^k \cdot P_k^i, & \text{for} \quad 15 \geq j \geq k > i \geq 0, \end{cases} \tag{3}$$

where $j = 0, \ldots, 15$, a_j and b_j are the input addend bits, c_j denotes the carry generated at position j and c_{-1} denotes the carry into the LSB position. Assuming that $c_{-1} = 0$, then the carry signal at position j, c_j, is given by $c_j = G_j^0$.

The direct 2-level AND-OR approach to implementing Equations (2) and (3) in, for example, static CMOS is not practical for any useful word length ≥ 16, since the amount of circuitry required to assimilate the MSB carry becomes prohibitive. To circumvent this, the input operands $(a_i,\ b_i)$, $i = 0, \ldots, 15$, are grouped into 4-bit blocks. The most significant group-generate and group-propagate signals of each 4-bit group, G_j^{j-3} and P_j^{j-3}, are computed directly from the input operands utilizing a set of CLA equations in a form suitable for implementation in threshold logic as introduced in [7]. A carry is generated in a group (G_j^{j-3}=1) if the sum of the two 4-bit words in the group exceeds (is strictly greater than) the maximum number representable by 4 sum bits ($2^4 - 1 = 15$). Similarly, the group propagates a carry originating in the neighbouring group of lower significance if the sum of the two 4-bit words in the group is equal to or greater than 15. This may be written in TL equation form as:

$$G_j^{j-3} = \operatorname{sgn}\left\{ \sum_{i=j-3}^{j} 2^{i-(j-3)}(a_i + b_i) - 16 \right\} \tag{4}$$

$$P_j^{j-3} = \operatorname{sgn}\left\{ \sum_{i=j-3}^{j} 2^{i-(j-3)}(a_i + b_i) - 15 \right\}. \tag{5}$$

The input weights for the gates which calculate G_j^{j-3} and P_j^{j-3} are the same, and the gate thresholds differ by 1. This means that the input weights can be shared between two CRTL gates to reduce area. Equations (4) and (5) give the sum of weights to be 30, which meets the previously stated requirement.

For example, the group generate, G_3^0, and group propagate, P_3^0, signals are computed from the 4-bit grouping of the input addend bits as follows:

$$G_3^0 = \operatorname{sgn}\{8a_3 + 8b_3 + 4a_2 + 4b_2 + 2a_1 + 2b_1 + a_0 + b_0 - 16\} \tag{6}$$

$$P_3^0 \;=\; \mathrm{sgn}\{8a_3 + 8b_3 + 4a_2 + 4b_2 + 2a_1 + 2b_1 + a_0 + b_0 - 15\}. \tag{7}$$

The three least significant group-generate and group-propagate signals in each 4-bit group are computed using the dynamic-CMOS implementation of Equations (2)-(3) using p_j and g_j signals computed in static-CMOS.

In the second level, a single CRTL gate assimilates the 4-bit group-generate and group-propagate signals to compute the most significant carry, c_{15}, as follows:

$$\begin{aligned} G_{15}^0 &= G_{15}^{12} + P_{15}^{12}G_{11}^8 + P_{15}^{12}P_{11}^8G_7^4 + P_{15}^{12}P_{11}^8P_7^4G_3^0 \\ &= \mathrm{sgn}\left\{8G_{15}^{12} + 4P_{15}^{12} + 4G_{11}^8 + 2P_{11}^8 + 2G_7^4 + P_7^4 + G_3^0 - 8\right\}. \end{aligned} \tag{8}$$

The sum bits are computed using static-CMOS XOR gates,

$$h_j = a_j \oplus b_j \tag{9}$$

$$s_j = h_j \oplus c_{j-1}. \tag{10}$$

The computation of the half-sum, h_j, is performed in parallel with the calculation of the carries in each 16-bit block. The sum bits, s_j, are calculated in parallel with the 64-bit adder global carry-lookahead.

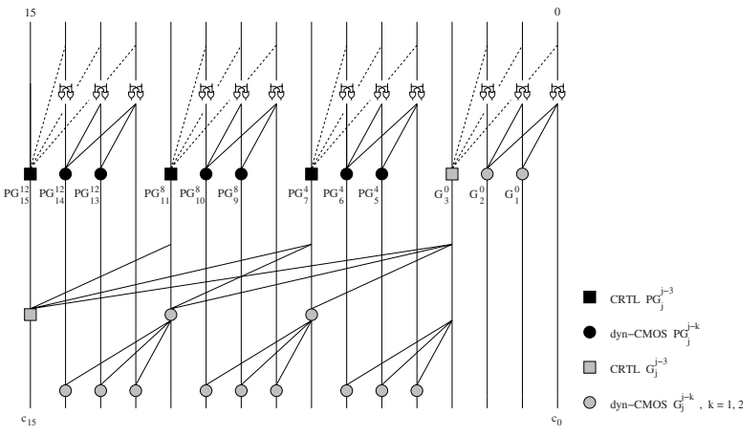

Fig. 3. 16-bit adder carry prefix-tree schematic.

The schematic diagram of the proposed 16-bit adder carry prefix-tree is shown in Fig. 3. The critical path uses CRTL prefix-cells to compute the carry-propagate and carry-generate signals, PG_j^i, for groups of 4-bits using Equations (4) and (5) as indicated by the black squares in Fig 3. Each black square cell consists of two CRTL gates with thresholds differing by 1 and sharing the same capacitive network for computing the weighted sum of inputs. Grey squares represent group carry generate cells G_j^i. The MSB carry output is computed using a single CRTL group-generate cell. The remaining cells of the prefix-tree

which are not on the critical path consist of conventional dynamic-CMOS carry generate and propagate cells to reduce overall area. These are indicated by grey circles in Fig. 3 and the circuits of these cells are given in Fig. 4.

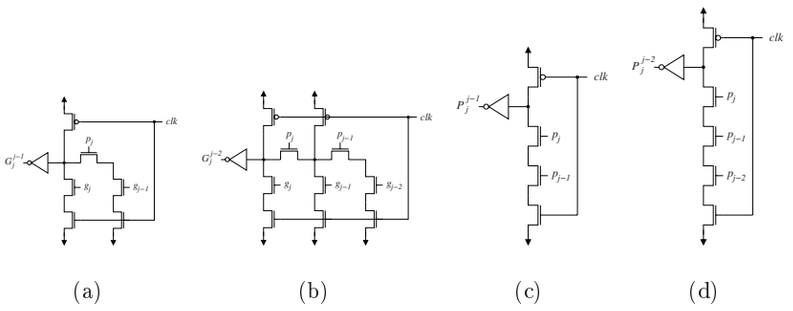

Fig. 4. Dynamic-CMOS circuits for (a) G_j^{j-1}, (b) G_j^{j-2}, (c) P_j^{j-1} and (d) P_j^{j-2}.

5 Results of the Proposed 64-bit Adder Design

The critical path of the 64-bit adder consists of two CRTL gates to compute c_{15}, one dynamic-CMOS gate to compute c_{47} in the global carry-lookahead block and the precharged-MUX used to select the sum bits $s_{63}, \ldots, s_{48}$. The critical path latency was simulated using 2P/4M, 0.35 μm process parameters in HSPICE and was found to be approximately 670 ps (from schematic only, layout and fabrication of the proposed adder is the subject of ongoing work). To compare with the delay of other recently reported adders, the delay was normalized with respect to the "fanout-of-four inverter delay", FO4. The FO4 delay is the delay of a minimum sized inverter driving four minimum sized inverters and the FO4 normalized delay of combinational logic is relatively constant over a wide range of processes [9]. Under typical environmental conditions, one FO4 delay in the 0.35 μm process used in this work is approximately 160 ps. To estimate the area cost, the total number of transistors for the 64-bit adder was also calculated.

Table 1. Comparison of A-DELTA with other 64-bit high speed adders

	Naffziger, HP [10]	Woo *et. al.* [11]	Sun *et. al.* [12]	A-DELTA
Transistors	6924	5460	6590	4325
Normalized FO4 Inv. Delays	7	5.6	5.5	4.3

The transistor count and FO4 normalized delay comparison with three other dynamic high speed adders is shown in Table 1. It should be noted that the delay results presented in Sun *et. al.* are based on simulation and the results in both Naffziger [10] and Woo *et. al.* [11] are from chip measurements. The results in Table 1 suggest an average speedup of almost 30% and an average reduction in the transistor count by over 30% compared to other high speed adder designs.

6 Conclusions

A high speed 64-bit adder based on a hybrid carry-lookahead/carry-select scheme using Charge Recycling Threshold Logic and dynamic-CMOS has been proposed. The worst case critical path delay and transistor count were shown to be both improved, on average, by approximately 30% compared to previously proposed conventional dynamic-logic high-speed adders. The results show that by combining TL and conventional CMOS logic with the appropriate architectural strategy, relatively fast and compact arithmetic circuits may be achieved.

References

1. Özdemir, H., Kepkep, A., Pamir, B., Leblebici, Y., Çiliniroğlu, U.: A capacitive threshold-logic gate. IEEE JSSC **31** (1996) 1141–1149
2. Celinski, P., López, J.F., Al-Sarawi, S., Abbott, D.: Low power, high speed, charge recycling CMOS threshold logic gate. IEE Electronics Letters **37** (2001) 1067–1069
3. Padure, M., Cotofana, S., Vassiliadis, S.: A low-power threshold logic family. In: Proc. IEEE International Conference on Electronics, Circuits and Systems. (2002) 657–660
4. Padure, M., Cotofana, S., Vassiliadis, S.: High-speed hybrid Threshold-Boolean logic counters and compressors. In: Proceedings of the 45th IEEE International Midwest Symposium on Circuits and Systems. (2002) 457–460
5. Leblebici, Y., Özdemir, H., Kepkep, A., Çiliniroğlu, U.: A compact high-speed (31-5) parallel counter circuit based on capacitive threshold-logic gates. IEEE JSSC **31** (1996) 1177–1183
6. Celinski, P., López, J.F., Al-Sarawi, S., Abbott, D.: Low depth carry lookahead addition using charge recycling threshold logic. In: Proc. IEEE International Symposium on Circuits and Systems, Phoenix (2002) 469–472
7. Vassiliadis, S., Cotofana, S., Bertels, K.: 2-1 addition and related arithmetic operations with threshold logic. IEEE Trans. Computers **45** (1996) 1062–1067
8. Rabaey, J.M.: Digital Integrated Circuits, A Design Perspective. Prentice Hall, New Jersey, USA (1996)
9. Ho, R., Mai, K.W., Horowitz, M.: The future of wires. Proceedings of the IEEE **89** (2001) 490–504
10. Naffziger, S.: A sun-nanosecond 0.5μm 64b adder design. In: International Solid State Circuits Conference Digest of Technical Papers, IEEE (1996) 362–363
11. Woo, R., Lee, S.J., Yoo, H.J.: A 670 ps, 64bit dynamic low-power adder design. In: Proceedings of the International Symposium on Circuits and Systems, Switzerland, IEEE (2000) I28–I31
12. Sun, X., Mao, Z., Lai, F., Ye, Y.: A high speed 0.25μm 64-bit CMOS adder design. In: Proceedings of ASIC, IEEE (2001) 581–583