On the Static CMOS Implementation of Magnitude Comparators

Costas Efstathiou#1 and Yiorgos Tsiatouhas*2

#Department of Informatics and Computer Engineering, University of West Attica, Greece

* Department of Computer Science and Engineering, University of Ioannina, Greece

1cefsta@uniwa.gr, 2tsiatouhas@cse.uoi.gr

Abstract—Digital magnitude comparators are used in computer systems to compare two binary numbers and determine if these are equal, or if one number is greater or less than the other. In this work, a new magnitude comparator's architecture is presented. The proposed comparator architecture is designed in static CMOS logic and compared against the state of the art magnitude comparators in the literature, shows less area overhead, and for small input operands (which are commonly used in practice) presents lower delay and power-delay product.

Keywords—Magnitude comparators; digital arithmetic; static CMOS; full custom design.

I. Introduction

Digital magnitude comparators [1], [2] are important elements for any computer architecture, image and signal processing systems [3], [4] modern wireless communication systems [5], [6] and other applications. In most cases, magnitude comparators that can recognize the three possible conditions between two operands $A>B$, $A=B$ and $A<B$ are highly desirable. In the last years, the design of efficient magnitude comparators has received significant attention. Some of the existing comparator designs use dynamic logic to improve performance. A high performance tree structured comparator design using all-n-transistor dynamic logic is proposed in [7]. In [8] a single-cycle two-phase magnitude comparator based on a priority-encoding algorithm, with higher performance than that of [7] is described. Comparators based on parallel-most-significant bit-checking algorithms instead of priority encoders are proposed in [9], [10]. A more efficient multiplexer-based magnitude comparator is described in [11]. These comparators operate faster than the previous designs, but have higher transistor count. In [12] a magnitude comparator, which uses bitwise competition logic after pre-encoding the inputs, is proposed. The design in [12] compared to the previous ones achieves lower transistor count and a delay improvement. The magnitude comparators in [13]-[15] are designed using dynamic logic Manchester like carry ahead modules.

While dynamic logic has superior performance, as compared to static one, its power consumption is higher than that of static logic. Static CMOS full custom designs of magnitude comparators are proposed in [16]-[18]. The design in [16] has the drawback of very high transistor count. The design in [17] is a full custom static CMOS design of the conventional comparator tree architecture using at the first level two input XOR and AND gates with low transistor count and improved power-delay product. A power-time efficient static CMOS design of magnitude comparator is presented in [18].

In this work, a new magnitude comparator architecture is presented, which compared to previous designs requires less area, and for small input operands have lower power-delay product, when implemented in full custom static CMOS logic.

The rest of the paper is organized as follows. In Section II the proposed efficient magnitude comparator architecture and its full custom static CMOS design are presented. Experimental results and comparisons are given in Section III. Our conclusions are drawn in Section IV.

II. Design Methodology

Let $A=a_{n-1}a_{n-2}\ldots a_1a_0$, $B=b_{n-1}b_{n-2}\ldots b_1b_0$ be two n-bit unsigned binary numbers. Number A is greater than B ($A>B$) if $a_{n-1}>b_{n-1}$, or if $a_{n-1}=b_{n-1}$ and $a_{n-2}>b_{n-2}$, or if $a_{n-1}=b_{n-1}$ and $a_{n-2}=b_{n-2}$ and $a_{n-3}>b_{n-3}$, …, or if $a_{n-1}=b_{n-1}$ and $a_{n-2}=b_{n-2}$ and … and $a_1=b_1$ and $a_0>b_0$. Then, $A>B$ if GT=1, where:

$$GT = a_{n-1}\overline{b}_{n-1} \vee (a_{n-1} \oplus \overline{b}_{n-1})a_{n-2}\overline{b}_{n-2} \vee \ldots \vee (a_{n-1} \oplus \overline{b}_{n-1})\ldots (a_1 \oplus \overline{b}_1)a_0\overline{b}_0 \qquad (1)$$

Also $A=B$ if EQ=1, where:

$$EQ = (a_{n-1} \oplus \overline{b}_{n-1})(a_{n-2} \oplus \overline{b}_{n-2})\ldots(a_1 \oplus \overline{b}_1)(a_0 \oplus \overline{b}_0) \qquad (2)$$

Obviously $A<B$ if signals GT=0 and EQ=0.

Let $g_i = a_i\overline{b}_i$, $e_i = a_i \oplus \overline{b}_i$, then:

$$GT = g_{n-1} \vee e_{n-1}g_{n-2} \vee \ldots \vee e_{n-1}\ldots e_1 g_0 \qquad (3)$$

$$EQ = e_{n-1}e_{n-2}\ldots e_1 e_0 \qquad (4)$$

The computation of signals GT and EQ is transformed into a prefix form using an associative operator o, which is defined as follows:

$$(g_k, e_k)\circ(g_l, e_l) = (g_k \vee e_k g_l, e_k e_l) \qquad (5)$$

We have that $GT=G_{n-1:0}$ and $EQ=E_{n-1:0}$, where:

$$(G_{n-1:0}, E_{n-1:0}) = (g_{n-1}, e_{n-1})\circ(g_{n-2}, e_{n-2})\circ\ldots\circ(g_0, e_0) \qquad (6)$$

Since operator o is associative, relation (6) is efficiently implemented in tree form. The comparator design in [17] is a full custom static CMOS tree implementation of relation (6). It uses low complexity and optimized for power pass transistor logic AND and XOR gates at the first level. These gates implement signals $g_i = a_i\overline{b}_i$ and $e_i = a_i \oplus \overline{b}_i$.

978-1-7281-2103-1/19/$31.00 ©2019 IEEE

Relation (1) can be derived by applying recursively relation $GT_i = a_i\overline{b}_i \vee (a_i \oplus \overline{b}_i)GT_{i-1}$, which is simplified as follows:

$$GT_i = a_i\overline{b}_i \vee (a_i \oplus \overline{b}_i)GT_{i-1} = [a_i\overline{b}_i \vee (a_i \oplus \overline{b}_i)][a_i\overline{b}_i \vee GT_{i-1}]$$
$$= (a_i \vee \overline{b}_i)(a_i\overline{b}_i \vee GT_{i-1}) = (a_i \vee \overline{b}_i)a_i\overline{b}_i \vee (a_i \vee \overline{b}_i)GT_{i-1}\,.$$

That is,

$$GT_i = a_i\overline{b}_i \vee (a_i \vee \overline{b}_i)GT_{i-1} \qquad (7)$$

Applying recursively relation (7) we get that:

$$\begin{aligned} GT &= a_{n-1}\overline{b}_{n-1} \vee (a_{n-1} \vee \overline{b}_{n-1})a_{n-2}\overline{b}_{n-2} \vee ... \\ &\vee (a_{n-1} \vee \overline{b}_{n-1})...(a_1 \vee \overline{b}_1)a_0\overline{b}_0 \end{aligned} \qquad (8)$$

That is,

$$GT = g_{n-1} \vee p_{n-1}g_{n-2} \vee ... \vee p_{n-1}...p_1g_0 \qquad (9)$$

where $g_i = a_i\overline{b}_i$ and $p_i = a_i \vee \overline{b}_i$.

Let signal *GP* is defined as follows:

$$GP = p_{n-1}p_{n-2}...p_1p_0 \qquad (10)$$

If *A*>*B*, then signal *GT*=1, while *GP* can be either 0 or 1. If *A*=*B*, signals $g_i = a_i\overline{b}_i = 0$ and $p_i = a_i \vee \overline{b}_i$=1. That is, if *A*=*B* then *GT*=0 and *GP*=1. Obviously if *A*<*B* then *GT*=0 and *GP*=0. Signals *GT* and *GP* are computed in [18] according to a methodology similar to that followed for the Ling carry computation [19].

The computation of signals *GT* and *GP* can also be transformed into a prefix form. We have that signals $GT=G_{n-1:0}$ and $GP=P_{n-1:0}$, where

$$\begin{aligned} (G_{n-1:0}, P_{n-1:0}) &= (g_{n-1}, p_{n-1}) \circ (g_{n-2}, p_{n-2}) \circ ... \\ &\circ (g_1, p_1) \circ (g_0, p_0) \end{aligned} \qquad (11)$$

The proposed OR-based tree magnitude comparator design is a full custom static CMOS implementation of relation (11) in tree form. It is composed of two stages as shown in Fig. 1. The first stage consists of low complexity pass transistor logic AND, OR gates, which implement signals $g_i = a_i\overline{b}_i$ and $p_i = a_i \vee \overline{b}_i$. The full custom design of these gates is shown in Fig. 2. The second stage of the proposed design is a binary tree consisting of black and gray nodes [17] as shown in Fig. 1. The full custom static CMOS design of the black nodes is shown in Fig. 3, while the design of the grey nodes is shown in Fig. 4. Obviously for $\log_2 n$ odd (*n*=8, 32, …), $GG^*=\overline{GG}$ and $GP^*=\overline{GP}$, while for $\log_2 n$ even (*n*=16, 64, …), *GG**=*GG* and *GP**=*GP*.

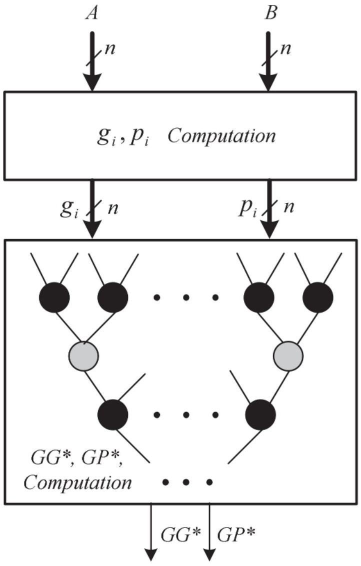


Fig. 1. Block diagram of the proposed magnitude comparators

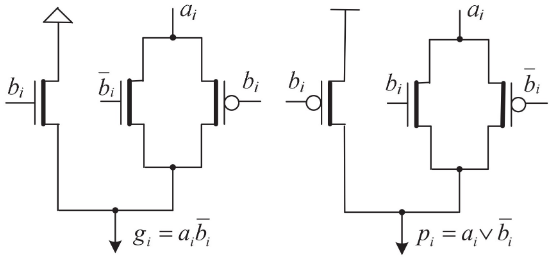


Fig. 2. Implementation of g_i, p_i

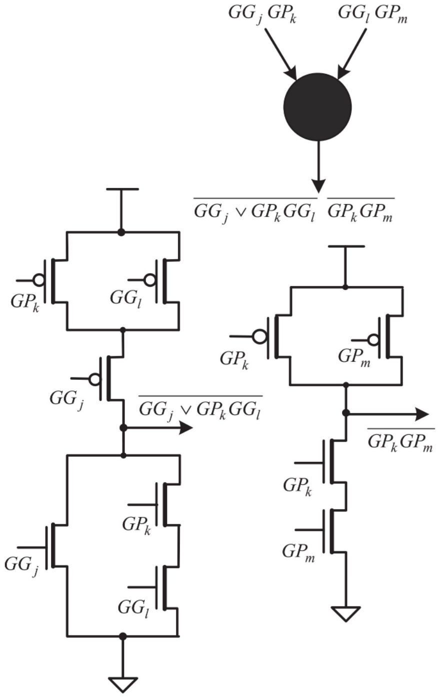


Fig. 3. Implementation of the black nodes

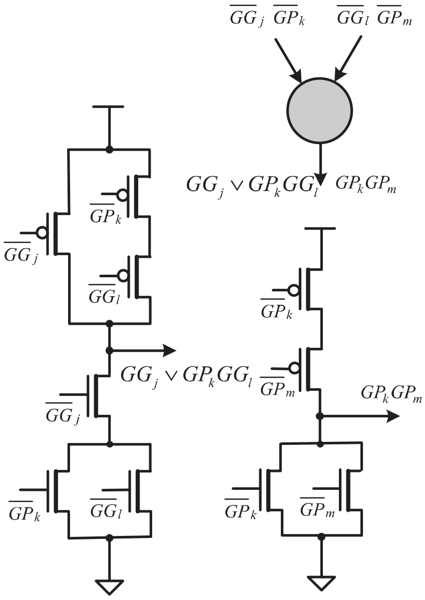


Fig. 4. Implementation of the gray nodes

Example. An 8-bit magnitude comparator designed according to the proposed methodology is illustrated in Fig. 5. The white squares implement signals $g_i = a_i\overline{b_i}$ and $p_i = a_i \vee \overline{b_i}$.

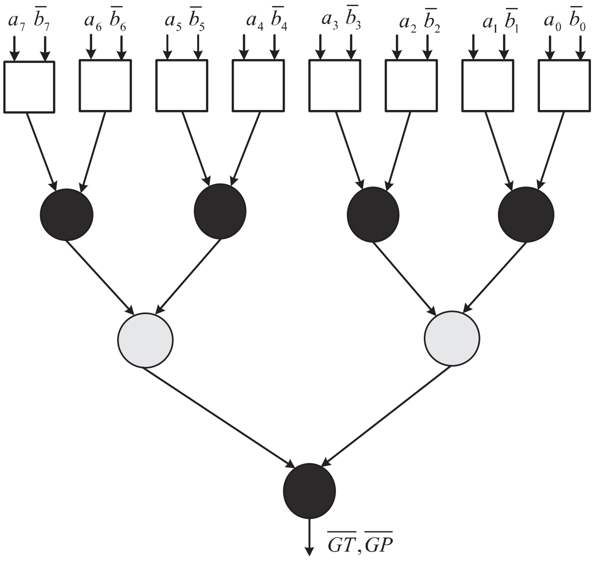


Fig. 5. Proposed design of the 8-bit magnitude comparator

III. Comparisons

The design in [17] is implemented using 19n-10 transistors, while the design in [18] is implemented using 15n transistors. The proposed design is implemented using 2n transistors for complementing one operand, 6n transistors for the implementation of signals g_i, p_i and 10(n−1) transistors for the implementation of the binary tree. Then, the total number of transistors in our design is $NT_{PR}=2n+6n+10(n-1)=18n-10$.

To evaluate the area, speed and power performance of the proposed comparator design over the designs in [17], [18], 8, 16, 32 and 64 bit magnitude comparators have been designed and simulated using the Cadence platform (Virtuoso, Spectre) and the standard 90-nm CMOS technology of UMC (V_{DD}=1V).

Table I presents the number of transistors (NT) and the total transistor gate area (GA =W×L) of the proposed magnitude comparators and that of the magnitude comparators presented in [17], [18]. All the used transistors have the same gate length L=80nm. The simulation results for the delay (T), power (P) and power-delay product (PT) are presented in Table II.

TABLE I. NUMBER OF TRANSISTORS AND TOTAL TRANSISTOR GATE AREA

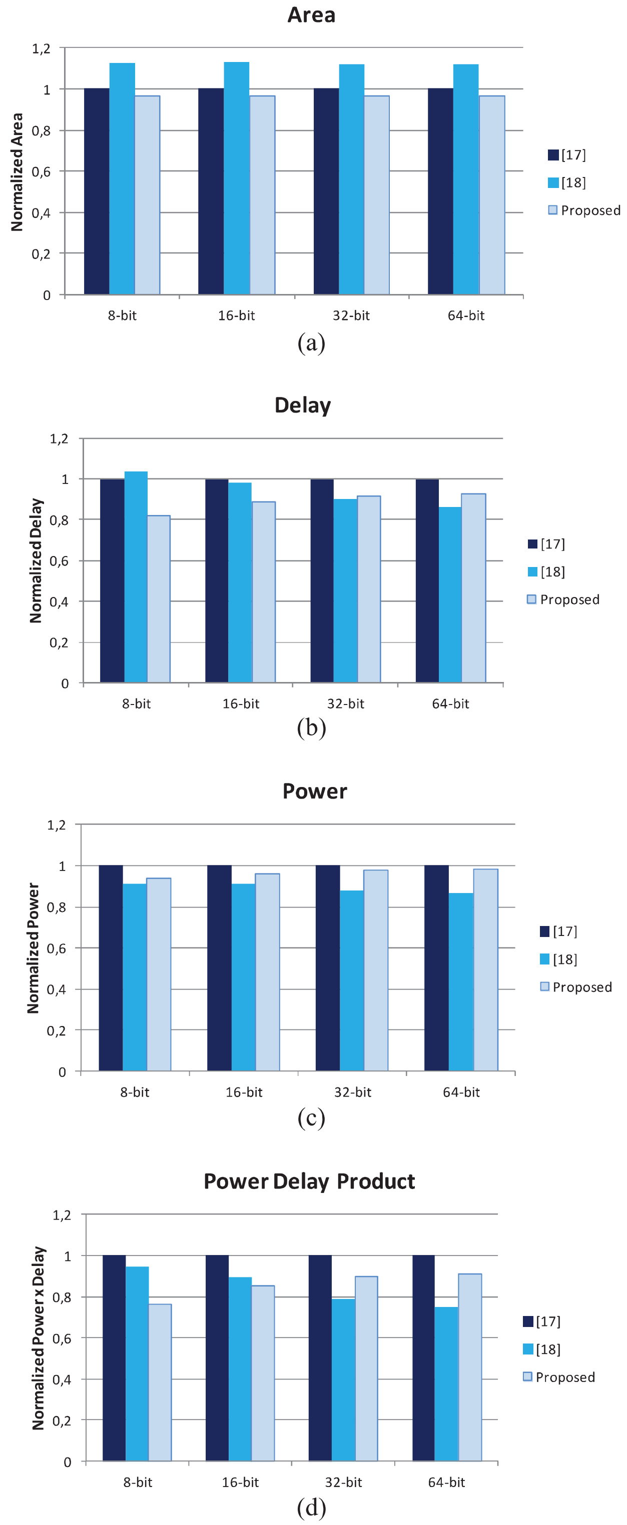
n	$NT_{[17]}$	$NT_{[18]}$	NT_{PR}	$GA_{[17]}$ (nm^2)	$GA_{[18]}$ (nm^2)	GA_{PR} (nm^2)
8	142	120	134	4704000	5299200	4534400
16	294	240	278	9830400	11116800	9491200
32	598	480	566	20041600	22435200	19382400
64	1206	960	1142	40505600	45388800	39187200

TABLE II. COMPARISONS IN DELAY, POWER AND POWER-DELAY PRODUCT

n	$T_{[17]}$ (ps)	$T_{[18]}$ (ps)	T_{PR} (ps)	$P_{[17]}$ (μW)	$P_{[18]}$ (μW)	P_{PR} (μW)	$PT_{[17]}$ (ps×μW)	$PT_{[18]}$ (ps×μW)	PT_{PR} (ps×μW)
8	171.0	177,0	139,7	38,2	35,0	35,9	6539,4	6195,0	5010,3
16	211,0	206,3	187,7	72,9	66,7	70,0	15388,2	13764,1	13137,6
32	240,0	215,7	219,6	145,4	128,1	142,4	34884,4	27628,1	31273,0
64	296,8	256,0	274,7	289,9	252,0	285,7	86037,2	64520,0	78469,1

We observe that the proposed comparator's architecture is implemented with lower number of transistors and gate area in the 90-nm CMOS technology, than the comparators in [17]. Although the total number of transistors of the proposed design is greater than that of the design in [18], the proposed comparator's total transistor gate area is lower. This is primarily because the proposed design requires a lower number of cascade connected pMOS transistors, which need to be sized larger compared with nMOS transistors due to the lower holes mobility, for the critical path logic evaluation. Also our design compared against the design in [17], always operates faster and consumes less power, while compared against the design in [18] for n=8, 16 operates faster and have lower power-delay product.

Next in Fig. 6 normalized comparison graphs (with respect to [17]) are presented for the silicon area, the delay, the dynamic power, the static power and the dynamic power-delay product.



(a)

(b)

(c)

(d)

Fig. 6. Normalized comparison graphs, a) silicon area, b) delay, c) power, and d) power-delay product

IV. Conclusions

In this work a new magnitude comparator's tree architecture has been proposed. This architecture is implemented in full custom static CMOS logic and demonstrated always low-circuit complexity, while for small input operands operates faster and have low power-delay product as compared with the state-of-the-art full custom static CMOS designs of magnitude comparators.

References

[1] M. Ercegovac, T. Lang, J. Moreno, "Introduction to Digital Systems," John Wiley and Sons, 1999.

[2] N. Weste, D. Harris, "CMOS VLSI design: A Circuits and Systems Perspective," 4th Edition, Addison Wesley, 2011.

[3] H. J. R. Liu and H. Yao, "High-Performance VLSI Signal Processing Innovative Architectures and Algorithms," vol. 2, Piscataway, NJ: IEEE Press, 1998.

[4] Y. Sheng and W. Wang, "Design and implementation of compression algorithm comparator for digital image processing on component," in *Proc. 9th Int. Conf. Young Comput. Science*, pp. 1337–1341, 2008.

[5] M. Shabany and P. Gulak, "A 0.13 μm CMOS 655 Mb/s 4 X 4 64-QAM K-Best MIMO detector," in *Proc. IEEE ISSCC*, 2009, pp. 256–257.

[6] T.-H. Kim and I.-C. Park, "Small-area and low-energy k-best MIMO detector using relaxed tree expansion and early forwarding," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 57, no. 10, pp. 2753–2761, Oct. 2010.

[7] C.-C. Wang, C.-F. Wu, and K.-C. Tsai, "1 GHz 64-bit high-speed comparator using ANT dynamic logic with two-phase clocking," in *IEE Proc.-Comput. Digit. Tech.*, vol. 145, no. 6, pp. 433–436, Nov. 1998.

[8] C-. H. Huang, J-. S. Wang, "High-performance and power-efficient CMOS comparators," *IEEE Journal of Solid-state Circuits,* vol. 38, no. 2, pp.254–262, Feb. 2003.

[9] H.-M. Lam, C.-Y. Tsui, "High-performance single clock cycle CMOS comparator," *IEE Electronics Letters*, vol. 42, no. 2, pp. 75–77, Jan. 2006.

[10] H.-M. Lam and C.-Y. Tsui, "High-performance single clock cycle CMOS comparator," in *Proc. Int. Symp. Circuit and System, ISCAS*, pp. 779–782, May 2006.

[11] H.-M. Lam and C.-Y. Tsui, "A MUX based high performance single cycle CMOS comparator," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 54, no. 7, pp. 591-595, Jul. 2007.

[12] J.-Y. Kim and H.-J. Yoo, "Bitwise competition logic for compact digital comparator," in *Proc. IEEE Asian ASSCC*, pp. 59–62, 2007.

[13] S. Perri, P. Corsonello, "Fast low-cost implementation of single-clock-cycle binary comparator," *IEEE Trans. Circuits and Systems II: Express Briefs,* vol. 55, no. 12, pp. 1239–1243, Dec. 2008.

[14] F. Frustaci, S. Perri, M. Lanuzza, and P. Corsonello, "A new low-power high-speed single-clock-cycle binary comparator," in *Proc. IEEE Int. Symp. Circuits Syst.*, pp. 317–320, 2010.

[15] F. Frustaci, S. Perri, M. Lanuzza and P. Corsonello, "Energy-efficient single-clock-cycle binary comparator," *Int. Journal of Circuit Theory and Applications,* vol. 40, no. 3, pp. 237-246, March 2012.

[16] S. Abdel-Hafeez, A. Gordon-Ross and B. Parhami, "Scalable Digital CMOS Comparator Using a Parallel Prefix Tree," *IEEE Trans. on VLSI Systems*, vol. 21, no. 11, pp. 1989-1998, Nov. 2013.

[17] P. Chuang, D. Li, and M. Sachdev, "A Low-Power High-Performance Single-Cycle Tree-Based 64-Bit Binary Comparator," *IEEE Trans. Circuits Syst.-II:Exp. Briefs*, vol. 59, no. 2, pp. 108–112, Feb. 2012.

[18] C. Efstathiou, C. Dimolikas, C. Papaioannou, Y. Tsiatouhas, "Low Power and High Speed Static CMOS Digital Magnitude Comparators, *in Proc. of Int. Conf. on Electronics Circuits and Systems, ICECS*, pp. 249-252, 2018.

[19] G. Dimitrakopoulos, D. Nikolos, "High-speed parallel-prefix VLSI Ling adders", *IEEE Trans. Computers*, vol. 54, no. 2, pp. 225-231, Feb 2005.