A COMPACT (M,N) PARALLEL COUNTER CIRCUIT BASED ON SELF TIMED THRESHOLD LOGIC

Peter Celinski, Said Al-Sarawi, Derek Abbott

Centre for High Performance Integrated
Technologies & Systems (CHiPTec),
The Department of Electrical and Electronic
Engineering, Adelaide University,
SA 5005, Australia.
celinski@eleceng.adelaide.edu.au

José F. López

Research Institute for Applied
Microelectronics, Universidad
de Las Palmas de G.C.,
35017-Spain.
lopez@iuma.ulpgc.es

ABSTRACT

The main result of this paper is the development of a novel, highly compact implementation of the general (m,n)-parallel counter (ie. population counter) based on Self-Timed Threshold Logic (STTL). The presented method is a modification of the Minnick counter. The novel feature of the design is the sharing among all threshold-gates of a single capacitive network for computing the weighted sum of all input bits. Additionally, the differential structure of STTL allows the efficient implementation of the networks of negative weights for layer 1 to layer 2 interconnections. This results in very significant reduction in the number of capacitors and interconnect routing cost and hence total area reduction over other recently reported counter designs. A (7,3) counter is designed using this method. The counter consists of 5 threshold gates arranged in two layers, that is, the resulting circuit has a logic depth of two. Simulation results for the (7,3) counter designed in an industrial 0.25 μm process indicate less than 880μW power dissipation operating at 300 MHz.

1. INTRODUCTION

As the demand for higher performance very large scale integration processors with increased sophistication grows, continuing research is focused on improving the performance, area efficiency, and functionality of the arithmetic and other units contained therein. Low power dissipation has become a major issue demanded by the high performance processor market in order to meet the high density requirements of advanced VLSI processors. The importance of low power is also evident in portable and aerospace applications, and is related to issues of reliability, packaging, cooling and cost.

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 traditional AND-OR-NOT (AON) logic-gate based design. However, lack of efficient physical realizations has meant that TL has, until recently, had little impact on VLSI. Efficient TL gate realizations have recently become available [1] [2] [3] [4] [5], and a number of applications based on TL gates have demonstrated its ability to achieve high operating speed and significantly reduced area compared to conventional logic [6].

This paper presents a novel, highly compact, low power implementation of the general (m,n)-parallel counter based on Self-Timed Threshold Logic (STTL). Section 2 gives a brief overview of threshold logic, followed by a description of Self-Timed Threshold Logic in Section 3. Section 4 contains the main results of this work and includes an overview of previous work related to parallel counter design, a description of the proposed parallel counter and its simulation results. Finally, the results of this work are summarized in Section 5.

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,$\ldots$,X_n and produces a single binary output Y, as shown in Fig. 1. A linear weighted sum of the binary inputs is computed followed by a thresholding operation.

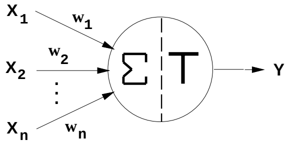

Fig. 1. Threshold Gate Model

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 corresponding to the ith input variable X_i. The output Y is given by

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

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

$$Y = \operatorname{sgn}\left(\sum_{i=1}^{n} w_i X_i - T\right). \tag{2}$$

The sgn function is defined as $\operatorname{sgn}(x) = 1$ if $x \geq 0$ and $\operatorname{sgn}(x) = 0$ if $x < 0$. Alternatively, expressions of the type $\operatorname{sgn}(x - T)$ may be conveniently (and informally) written simply as T^+, where it is understood that the actual sgn function argument is $x - T$. This will allow us to easily describe feed-forward TL networks with composite expressions such as $y = \operatorname{sgn}(x - 2 - 3 \cdot 5^+)$.

Any threshold function can be computed with positive integral weights and a positive real threshold, and all Boolean functions can be realized by a threshold gate network of depth at most two. 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 $T = n$ 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 be obtained, especially in applications requiring a large number of input variables.

3. SELF-TIMED THRESHOLD LOGIC (STTL)

Both static and dynamic synchronous TL gate implementations have been devised. Purely static gates such as neuron-MOS suffer from limited fan-in [6], typically less than 12 inputs. Also, some of the existing dynamic gates have relatively high short circuit and dynamic power dissipation, and some require multiple clock phases [2] [6] [7].

Fig. 2 shows the proposed circuit structure for implementing a self-timed threshold gate with positive weights and threshold. The main element is the cross coupled NMOS transistor pair (M3, M4) which generates the output Q and its complement Q_b after buffering by the two inverters. The gate operates in two phases. Precharge and evaluate are specified by the dual enable signals E and its complement E_b. The inputs X_i are capacitively coupled onto the floating gate ϕ of M10, and the threshold is set by the gate voltage T of M11. 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 realised by setting capacitors C_i to appropriate values. Typically, these capacitors are implemented between the polysilicon 1 and polysilicon 2 layers, although alternatives, such as trench capacitors used in DRAMs, or MIM capacitors, are available in some processes.

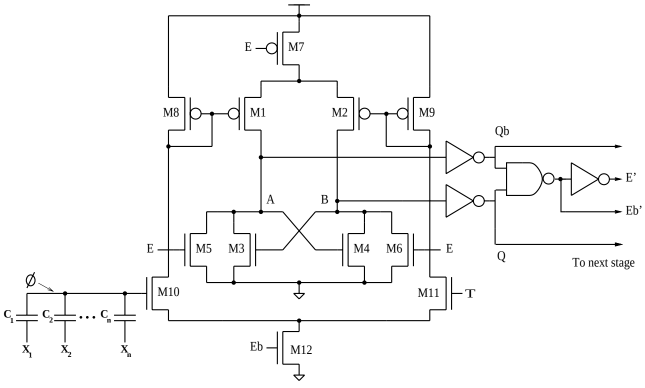

Fig. 2. The proposed Self-Timed Threshold Logic gate structure

The enable signals, E and E_b, control the precharge and activation of the sense circuit. When E is high the voltages at nodes A and B are are discharged to ground . When E is low and E_b is high, the outputs are disconnected from ground and the differential circuit (M10, M11 and M12) draws different currents from the supply via M8 and M9. The currents in M8 and M9 are mirrored by M1 and M2 respectively, and the gates of M3 and M4 (nodes A and B) begin to charge at different rates. As the charge rates are different and the capacitances at those two nodes are the same (ensured by identical sizing of the two buffer inverters), a voltage difference begins to develop between nodes A and B. When this difference is sufficiently large, either M3 or M4 turns on, but not both. The outputs Q and Q_b are evaluated and passed to the next stage. In this way, the circuit structure effectively determines if the weighted sum of the inputs, ϕ, is greater or less than the threshold, T, thus realizing a thresholding operation. The two buffer inverters serve to provide a balanced capacitive load for nodes A and B and also to drive the inputs of the next stage.

The Enable signals for the next stage (STTL gate) are generated by the NAND gate and its inverse, E'_b and E', respectively. During the precharge phase of the first stage, the Enable signals for the next stage are E'_b=0 and E'=1, which means that this stage is also in the precharge phase, and only begins to evaluate after the outputs of the first stage (Q and Q_b) are established. Correct timing is ensured by setting the combined delay of the two buffer inverters and the NAND gate to be larger than the evaluation delay of the first gate. Thus, outputs of each gate propagate through the chain in a self-timed fashion.

4. THE PROPOSED PARALLEL COUNTER

Parallel counters, or simply counters, are multiple-input circuits which count the number of inputs in a given state (normally logic 1). They are important in various applications, the most common of which are the reduction of the partial product tree in parallel multipliers and the realization of multiple-input adders. We begin this section by giving an introduction to counters, an overview of previous designs and a description of the proposed design and the simulation results.

4.1. Background and Related Work

An (m,n) counter is a combinatorial network which generates a binary coded output vector of length n which corresponds to the number, (or *count*), of logic 1's in the m-bit input vector. Usually $n = log_2(m+1)$ and such counters are referred to as *saturated*. The full adder is a particular case of a counter with 3 inputs and 2 outputs, thus it is a (3,2) counter.

In conventional logic, higher order counters, such as (7,3), (15,4) or (31,5), have traditionally been implemented by using trees of (3,2) counters because of the disadvantages of a direct implementation [8]. However, counters consisting of such full adder trees have a relatively high delay and grow rapidly with input vector size in terms of the required number of full adders. Swartzlander reported the number of full adders for an m-input population counter as $m - log_2 m$ [9]. It would be ideal if it were possible to design area efficient higher order counters which could operate at much higher speeds than the same counters built using trees of full adders [10]. Threshold logic allows us to do exactly this.

There are a number of TL based counter designs, and those of note include the "basic" counter [11], the Kautz counter [12] and the Minnick counter [13]. They differ in the number of threshold gates, logic depth, maximum fan-in and maximum weight.

The "basic" counter is expensive in terms of gate number and number of interconnects and is mentioned here only for completeness. The Kautz counter is the most area efficient, considering interconnection, weight size and gate number requirements, but an (m,n) Kautz counter has a logic depth equal to n. This means that circuits based on this counter, while being compact, have a relatively high delay.

A different approach to counter design was shown in [14]. This approach is based on a sorter circuit followed by one layer of threshold gates to obtain the final counter outputs. While it was shown to have improved speed and power dissipation over a conventional logic full adder based design for the case of a (15,4) counter, its logic depth is the same and its gate count is almost double that of the Minnick counter.

The Minnick counter offers the best tradeoff in terms of area and delay. The worst case delay for all outputs is equal to two threshold gates and is independent of the order of the counter. The most significant output always has a delay of one threshold gate.

For these reasons we have chosen the Minnick counter as the basis for our modified implementation, and we will refer to our counter as the Modified Minnick Counter (MMC).

The general Minnick counter is best explained by example using the (7,3) case. In essence, all inputs are connected with weight one to the threshold gates in both the first and second layers. Additionally, outputs of the first layer are connected to the second (output) layer with appropriate negative weights to shift the apparent thresholds of the output layer gates.

The truth table for the (7,3) counter, and the (7,3) Minnick counter design are shown in Fig. 3. The input v consists of the seven input bit lines, each having a weight of 1, and is denoted by a thick black line to differentiate it from the single bit lines. In effect v represents the arithmetic sum of 1's in the 7 inputs. From the truth table, the MSB of the output, y_2, is 1 when $v \geq 4$, hence y_2 is the output of the first layer gate which has threshold 4. The y_1 output is 1 when $2 \leq v < 4$ and $v \geq 6$. Therefore the second layer gate which has threshold 2 computes y_1 and this gate has an input weighted -4 from the first layer gate which has threshold 4. Similar reasoning may be extended to the output y_0. In the general case, it can be seen that the MSB will be computed by a first layer gate, and the lesser significance outputs are computed in the second layer which has as inputs, in addition to v, the negatively weighted outputs from the first layer to isolate the desired ranges of v where those outputs are 1.

v	y_2	y_1	y_0
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1

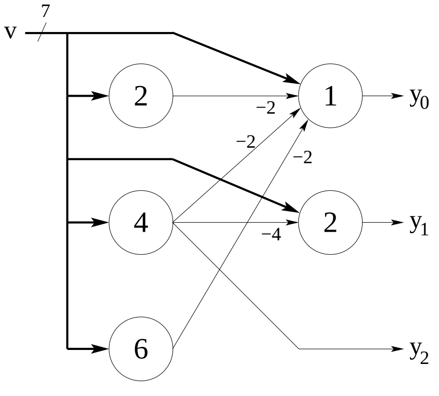

Fig. 3. The (7,3) counter truth table and the Minnick implementation

The operation of the (7,3) Minnick counter can be described by the following expressions:

$$y_2 \;=\; \text{sgn}(v-4) \;=\; 4^+$$

$$\begin{aligned} y_1 &= \mathrm{sgn}(v - 2 - 4 \cdot 4^{+}) \\ y_2 &= \mathrm{sgn}(v - 1 - 2 \cdot 2^{+} - 2 \cdot 4^{+} - 2 \cdot 6^{+}) \qquad (3) \end{aligned}$$

4.2. The Proposed Modified Minnick Counter (MMC) Design

As was described in the previous section, each of the gates in the first and second layers includes among its inputs all of the inputs to the counter. This means that the network which performs the weighted summation of the counter inputs (with all weights being 1) in the VLSI layout of the counter is replicated at each threshold gate. In the recently proposed capacitive threshold gate designs, this contributes to a very significant portion of the total counter area. For example, in the (7,3) counter discussed previously, the total number of capacitors performing the summation of the input bits at each of the 5 gates is 35 (5 gates × 7 input bits). Additionally, there is significant area associated with routing the 7 interconnect lines to each of the 5 gates. These drawbacks have an even greater impact on total area for higher order counters.

The innovation proposed here is to separate, at the circuit level, the two functions performed by the threshold gate, namely weighted addition and thresholding. In other words, the capacitive network which calculates the analog value of the sum of the counter input bits needs only to be implemented once, and this value becomes one input of the sense amplifier in any number of STTL gates. In the second layer gates, the other sense amplifier input is connected to the capacitive networks which implement the negative weights of the layer 1 to layer 2 interconnections. Additional capacitors can also be connected to the other input to set the gate threshold. Such an arrangement is possible only because of the differential nature of the SSTL gate and is not possible with other recent TL gate designs including neuron-MOS [1], LPTL [2], CTL [3] or the approach described in [5]. It reduces the number of capacitors required from 39 to 22 in the Modified Minnick (7,3) counter implementation.

The circuit diagram showing this design is shown in Fig. 4. The numbers next to the capacitors indicate the multiple of the unit capacitor. The enable signals, E and E_b are not shown to improve clarity. The two gates in the second layer are enabled after the outputs from the first layer are evaluated, as discussed in Section 3. The enable signal of one of the first layer gates drives the enable inputs of both second layer gates. The capacitors shown connected to Gnd and V_{dd} adjust the effective threshold of each STTL gate. The outputs of the first layer gates are connected to the capacitors which implement the negative weights. The inputs denoted by I_1 and I_2 in Fig. 4 correspond to the ϕ and T inputs, respectively, shown in Fig. 2

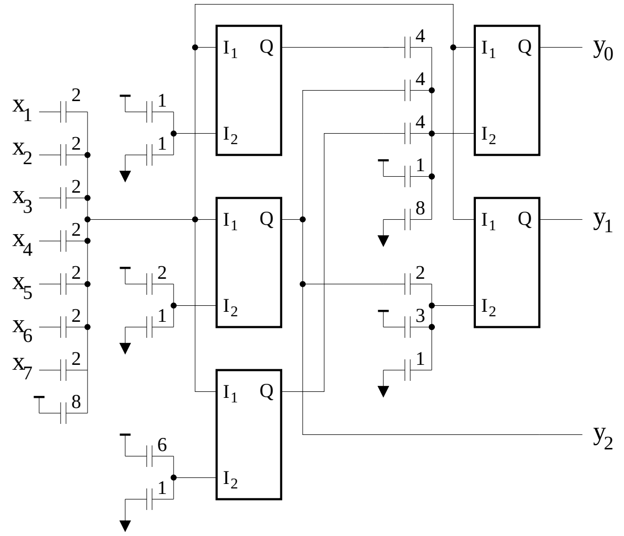

Fig. 4. Circuit diagram of the proposed STTL Modified Minnick (7,3) counter

4.3. Simulation Results

The counter circuit shown in Fig. 4 was simulated with HSPICE using 0.25 μm process parameters at a supply voltage of 2V. The value of the unit capacitor was chosen to be 5fF. The simulation waveforms are shown in Fig. 5 and include the first layer enable signal, E, the weighted input vector signal, I_1, and the three output bits. The waveform which has the form of a staircase is the I_1 input to each STTL gate and increases as the number of 1's in the input vector, $(x_1, \ldots, x_7)$, is increased from 0 to 7. It can be seen that when E goes low ,the outputs y_2, y_1 and y_0 evaluate correctly for all values of the input vector.

It should be noted that the output y_2 is available after one gate delay and the remaining two outputs are available after two gate delays. All outputs can be made to evaluate simultaneously by adding one additional STTL gate which would act as a delay element for y_2. The enable signal frequency for the first layer gates was 300 MHz and the power dissipation was measured to be 870 μW. The counter delay is less than 1.4ns, measured from the falling edge of the enable signal to y_0 or y_1.

5. CONCLUSIONS

A compact implementation of the general (m,n)-parallel counter based on Self-Timed Threshold Logic was presented. The design was shown to have a reduced number of capacitors over previous designs. Simulation results of a (7,3) counter designed using the proposed method in a 0.25 μm process show that it has low power dissipation and is capable of operating at high speeds.

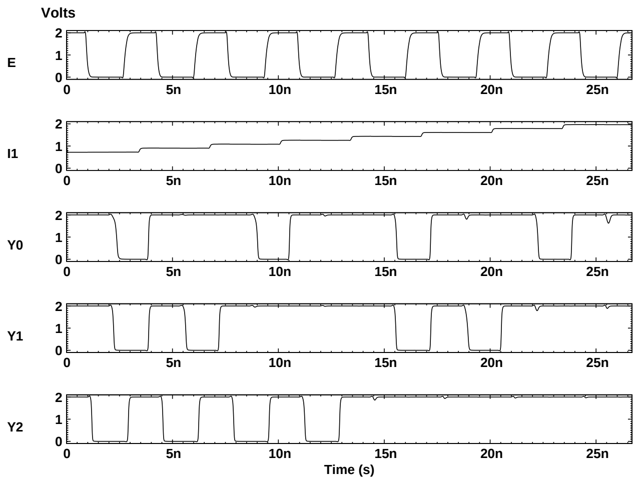

Fig. 5. Simulation results of the STTL Modified Minnick (7,3) counter

Acknowledgments

The support of the Australian Research Council and the Sir Ross and Sir Keith Smith Fund is gratefully acknowledged.

6. REFERENCES

[1] T. Shibata and T. Ohmi, "An intelligent MOS transistor featuring gate-level weighted sum and threshold operations," in *IEDM, Technical Digest*, New York, NY, USA, Dec 1991, IEEE.

[2] M.J. Avedillo, J.M. Quintana, A. Rueda, and E. Jiménez, "Low-power CMOS threshold-logic gate," *IEE Electronics Letters*, vol. 31, no. 25, pp. 2157–2159, Dec. 1995.

[3] H. Özdemir, A. Kepkep, B. Pamir, Y. Leblebici, and U. Çiliniroğlu, "A capacitive threshold-logic gate," *IEEE JSSC*, vol. 31, no. 8, pp. 1141–1149, August 1996.

[4] P. Celinski, J. F. López, S. Al-Sarawi, and D. Abbott, "Low power, high speed, charge recycling CMOS threshold logic gate," *IEE Electronics Letters*, vol. 37, no. 17, pp. 1067–1069, August 2001.

[5] J. Fernandez Ramos, J. A. Hidalgo Lopez, M. J. Martin, J. C. Tejero, and A. Gago, "A threshold logic gate based on clocked coupled inverters," *International Journal of Electronics*, vol. 84, no. 4, pp. 371–382, 2001.

[6] K. Kotani, T. Shibata, M. Imai, and T. Ohmi, "Clocked-neuron-MOS logic circuits employing auto-threshold-adjustment," in *ISSCC Digest of Technical Papers*, 1995, pp. 320–321.

[7] H.Y. Huang and T.N. Wang, "CMOS capacitor coupling logic (C^3L) circuits," in *Proc. of IEEE Asia Pacific Conference on ASIC*, 2000, pp. 33–36.

[8] P. J. Song and G. D. Micheli, "Circuit and architecture tradeoffs for high-speed multiplication," *IEEE Journal of SolidState Circuits*, vol. 26, pp. 1184–1198, September 1991.

[9] E. E. Swartzlander, "Parallel counters," *IEEE Transactions on Computers*, vol. C-22, pp. 1021–1024, 1973.

[10] V. G. Oklobdzija, D. Villeger, and S. S. Liu, "A method for speed optimized partial product reduction and generation of fast parallel multipliers using an algorithmic approach," *IEEE Transactions on Computers*, vol. C-45, no. 3, pp. 294–305, March 1996.

[11] Tijs Huisman, "Counters and multipliers with threshold logic," M.S. thesis, Delft University of Technology, May 1995.

[12] W. H. Kautz, "The realization of symmetric switching functions with linear input logical elements," *IRE Transactions on Electronic Computers*, vol. EC-10, pp. 371–378, March 1961.

[13] R. C. Minnick, "Linear-Input Logic," *IRE Transactions on Electronic Computers*, vol. EC-10, pp. 6–16, March 1961.

[14] E. Rodriguez-Villegas, M.J. Avedillo, J.M. Quintana, G. Huertas, and A. Rueda, "A vMOS based sorter for arithmetic applications," *VLSI Design*, vol. 11, no. 2, pp. 129–136, 2000.