A Digital CMOS Parallel Counter Architecture Based on State Look-Ahead Logic

Saleh Abdel-Hafeez, *Member, IEEE*, and Ann Gordon-Ross, *Member, IEEE*

***Abstract*—We present a high-speed wide-range parallel counter that achieves high operating frequencies through a novel pipeline partitioning methodology (a counting path and state look-ahead path), using only three simple repeated CMOS-logic module types: an initial module generates anticipated counting states for higher significant bit modules through the state look-ahead path, simple D-type flip-flops, and 2-bit counters. The state look-ahead path prepares the counting path's next counter state prior to the clock edge such that the clock edge triggers all modules simultaneously, thus concurrently updating the count state with a uniform delay at all counting path modules/stages with respect to the clock edge. The structure is scalable to arbitrary N-bit counter widths (2-to-2N range) using only the three module types and no fan-in or fan-out increase. The counter's delay is comprised of the initial module access time (a simple 2-bit counting stage), one three-input AND-gate delay, and a D-type flip-flop setup-hold time. We implemented our proposed counter using a 0.15-μm TSMC digital cell library and verified maximum operating speeds of 2 and 1.8 GHz for 8- and 17-bit counters, respectively. Finally, the area of a sample 8-bit counter was 78 125 μm^2 (510 transistors) and consumed 13.89 mW at 2 GHz.**

***Index Terms*—Architecture design, high-performance counter design, parallel counter design, pipeline counter design.**

I. Introduction and Related Work

Counters are widely considered as essential building blocks for a variety of circuit operations [1], [8], [24], [25], [27], [42] such as programmable frequency dividers, shifters, code generators, memory select management, and various arithmetic operations. Since many applications are comprised of these fundamental operations, much research focuses on efficient counter architecture design. Counter architecture design methodologies explore tradeoffs between operating frequency, power consumption, area requirements, and target application specialization.

Early design methodologies [7] improved counter operating frequency by partitioning large counters into multiple smaller counting modules, such that modules of higher significance (containing higher significant bits) were enabled when all bits in all modules of lower significance (containing lower significant bits) saturate. Initializations and propagation delays such as register load time, AND logic chain decoding, and the half incrementer component delays in half adders dictated operating frequency. Subsequent methodologies [22], [35] improved counter operating frequency using half adders in the parallel counting modules that enabled carry signals generated at counting modules of lower significance to serve as the count enable for counting modules of higher significance, essentially implementing a carry chain from modules of lower significance to modules of higher significance. The carry chain cascaded synchronously through intermediate D-type flip-flops (DFFs). The maximum operating frequency was limited by the half adder module delay, DFF access time, and the detector logic delay. Since the module outputs did not directly represent count state, the detector logic further decoded the module outputs to the outputted count state value. Further enhancements [41] improved operating frequency using multiple parallel counting modules separated by DFFs in a pipelined structure. The counting modules were composed of an incrementer that was based on a carry-ripple adder with one input hardcoded to "1" [35]. In this design, counting modules of higher significance contained more cascaded carry-ripple adders than counting modules of lower significance. Each counting module's count enable signal was the logical AND of the carry signals from all the previous counting modules (all counting modules of lower significance), thus prescaling clocked modules of higher significance using a low frequency signal derived from modules of lower significance. Due to this prescaling architecture, the maximum operating frequency was limited by the incrementer, DFF access time, and the AND gate delay. The AND gate delay could potentially be large for large sized counters due to large fan-in and fan-out parasitic components. Design modifications enhanced AND gate delay, and subsequently operating frequency, by redistributing the AND gates to a smaller fan-in and fan-out layout separated by latches. However, the drawback of this redistribution was increased count latency (number of clock cycles required before the output of the first count value). In addition, due to the design structure, this counter architecture inherited an irregular VLSI layout structure and resulted in a large area overhead.

Hoppe *et al.* [13] improved counter operating frequency by incorporating a 2-bit Johnson counter [18] into the initial counting module (least significant) in a partitioned counter architecture. However, the increase in operating frequency was offset by reduced counting capability. In Hoppe's design, counting modules of higher significance were constructed

Manuscript received May 27, 2009; revised November 12, 2009; accepted February 10, 2010. First published April 05, 2010; current version published May 25, 2011. This work was supported in part by Jordan University of Science and Technology and by the University of Florida.

S. Adbel-Hafeez is with the Jordan University of Science and Technology, IRBID, 22110 Jordan (e-mail: sabdel_99@yahoo.com).

A. Gordon-Ross is with the Electrical and Computer Engineering Department, University of Florida, Gainesville, FL 32611 USA and also with the NSF Center for High-Performance Reconfigurable Computing (CHREC), University of Florida, Gainesville, FL 32611 USA (e-mail: ann@chrec.org).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TVLSI.2010.2044818

of standard synchronous counters triggered by the Johnson counter and additional synchronization logic. However, the synchronization circuit and initial module still limited the operating frequency and resulted in reduced applicability.

Kakarountas *et al.* [17] used a carry look-ahead circuit [11] to replace the carry chain. The carry look-ahead circuit used a prescaler technique with systolic 4-bit counter modules [which used T-type flip-flops (TFFs)], with the cost of an extra detector circuit. The detector circuit detected the assertion of lower order bits to enable counting in the higher order bits. To further improve operating frequency, Kakarountas's design used DFFs between systolic counter modules. The clock period was bounded by the delay of two input gates in addition to the TFF access and setup-hold time. Large counter widths incurred an additional three input logic gate delay. However, since the counter design was limited by control signal broadcasting, Kakarountas's design was not practical for large counter widths even though the Xilinx Data Book [37] shows that several counter designs with the highest operating frequencies use prescaler techniques.

In order to create a more efficient architecture for large counter widths and more amenability to a wider application range, counter architectures, such as up/down counters [30], [32], added extra (redundant) registers (while still using partitioned counter modules [7], [35]) to store the previous counter state during a counter state transition (counter increment). Thus, when the counting direction changed (from up to down or down to up), the contents of these count state registers determine the next counter state.

Jones *et al.* [16] designed a counter specialized for applications with fast arithmetic operations [12], [28] using a half/full adder prefix structure. This prefix structure partially alleviated the cascading adder carry chain delay at the expense of a large area overhead. However, prefix structures are not practical for large counter widths due to an increase in the number of inputs, resulting in a large number of wide adders with large delays.

Several modern counter designs are well suited to applications with various arithmetic operations, such as systolic counters and population counters. Systolic counters [26], [31] have high operating frequencies at the expense of representing the count value using two redundant binary numbers, which results in a large area overhead for state decoding. Population counters [19], [21], [36] and counting responders [10] provide high operating frequencies using the relationship between counter inputs and outputs based on listing all input bits (input vector length). Literature reports population counters as capacitive thresholds-logic gates [19], cascading trees of full/half adders [36], or a shift switch logic structure using an output decoding methodology [21]. Other modern counter designs target particular applications (such as combinatorial optimizations and image processing) using the "n choose k" counter ((n, k)-counter) [14]. However, a logarithmic shift operation delay limits this counter design's applicability to only small n and k values. (A thorough literature review of large parallel counter designs can be found in [33].) Finally, alternative counter designs increase counter operating frequency using ratioed logic dynamic DFFs [6], [38], [39], but however these designs tended to have large area overheads making them not ideal for continued CMOS technology scaling.

In order to reduce high counter power consumption, Alioto *et al.* [2] presented a low power counter design with a relatively high operating frequency. Alioto's design was based on cascading an analog block (these analog blocks were structured using MOS current-mode logic to represent an analog divider stage) such that each counting stage's (module's) input frequency was halved compared to the previous counting stage (module). However, Alioto's counter design's carry chain rippled through all counting stages, resulting in a total critical path delay equal to the sum of all counting stage delays. Subsequently, Alioto's design was not well suited for large counter widths because the carry chain limited operating frequency even though the carry chain voltage was not rail-to-rail. In addition, the counter circuit's continuous standby current required a device shutdown mechanism in order to regulate power consumption. Furthermore, the counter circuit's active margin was bounded by 1/3 of the supply voltage, which resulted in high design costs with current CMOS technologies that usually inherit low supply voltages.

In this paper, we improve counter operating frequency using a novel parallel counting architecture in conjunction with a state look-ahead path and pipelining to eliminate the carry chain delay and reduce AND gate fan-in and fan-out. The state look-ahead path bridges the anticipated overflow states to the counting modules, which are exploited in the counting path. The counting modules are partitioned into smaller 2-bit counting modules separated by pipelined DFF latches. The state look-ahead path is partitioned using the same pipelined alignment paradigm as the counting path and thereby provides the correct anticipated overflow states for all counting stages. Subsequently, all counting states and all pipelined DFFs (in both paths) are triggered concurrently on the clock edge, enabling the count state in modules of higher significance to be anticipated by the count state in modules of lower significance. This cooperation between the counting path and state look-ahead paths enables every counting module (both low and high significance) to be triggered concurrently on the clock edge without any rippling effect.

The main contributions of our proposed parallel counter are as follows.

1) A single clock input triggers all counting modules simultaneously, resulting in an operating frequency independent of counter width (assuming ideal parasitic capacitance on the clock wire path, without loss of generality). The total critical path delay (regardless of counter width) is uniform at all counting stages and is equal to the combination of the access time of a 2-bit counting module, a single three-input AND gate delay, and the DFF setup-hold time.
2) Our parallel counter architecture leverages modularity, which enables high flexibility and reusability, and thus enables short design time for wide counter applications. The architecture is composed of three basic module types separated by DFFs in a pipelined organization. These three module types are placed in a highly repetitious structure in both the counting path and the state look-ahead paths, which limits localized connections to only three signals (thus, fan-in ≤ 3 and fan-out $= 1$).

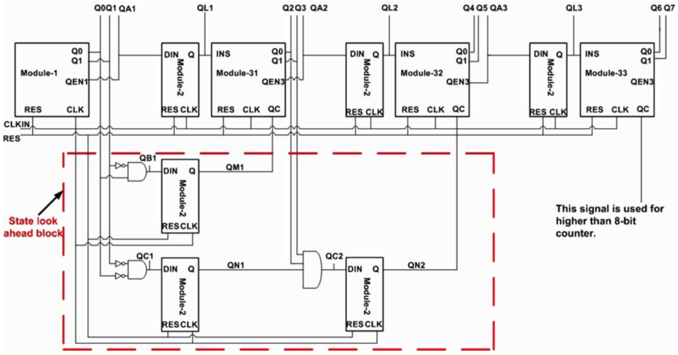
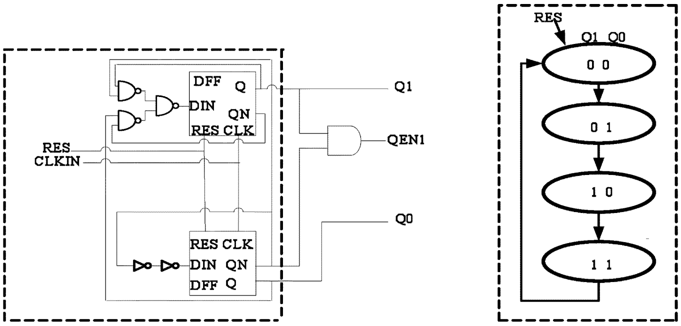


Fig. 1. Functional block diagram of our proposed 8-bit parallel counter with state look-ahead logic and counting logic. The state look-ahead logic consists of all logic encompassed by the dashed box and the counting logic consists of all other logic (not encompassed by the dashed box).



(a)

(b)

Fig. 2. Module-1 (a) hardware schematic and (b) state diagram. Note that the 1 in $QEN1$ denotes that this is the QEN for module-1.

3) The counter output is in radix-2 representation so the count value can be read on-the-fly with no additional logic decoding.
4) Unlike previous parallel counter designs that have count latencies of two or three cycles, depending on the counter width, our parallel counter has no count latency, which enables the count value to be read on-the-fly.

The remainder of this paper is organized as follows. Section II discusses our proposed parallel counter circuit based on a CMOS implementation scheme. Section III provides timing analysis and related formulas. Section IV presents simulation results for our counter synthesized to an Altera field-programmable gate-array (FPGA) device as well as a post layout net-list using the HSPICE simulator. Finally, we conclude and provide future directions in Section V.

II. Parrallel Counter Architecture

Fig. 1 depicts our proposed parallel counter architecture for a sample 8-bit counter. The main structure consists of the state look-ahead path (all logic encompassed by the dashed box) and the counting path (all logic not encompassed by the dashed box). We construct our counter as a single mode counter, which sequences through a fixed set of preassigned count states, of which each next count state represents the next counter value in sequence. The counter is partitioned into uniform 2-bit synchronous up counting modules. Next state transitions in counting modules of higher significance are enabled on the clock cycle preceding the state transition using stimulus from the state look-ahead path. Therefore, all counting modules concurrently transition to their next states at the rising clock edge (CLKIN).

In this section, we describe the architecture and functionality of our parallel counter, the derivation of each counter state equation, and area analysis.

A. Architectural Functionality

The counting path's counting logic controls counting operations and the state look-ahead path's state look-ahead logic anticipates future states and thus prepares the counting path for these future states. Fig. 1 shows the three module types (module-1, module-2, and module-3 S, where $S = 1, 2, 3$, etc. (increasing from left to right) and represents the position of module-3) used to construct both paths. Module-1 and module-3 are exclusive to the counting path and each module represents two counter bits. Module-2 is a conventional positive edge triggered DFF and is present in both paths. In the counting path, each module-3 S is preceded by an associated module-2. Module-3 S's serve two main purposes. Their first purpose is to generate all counter bits associated with their ordered position and the second purpose is to enable (in conjunction with stimulus from the state look-ahead path) future states in subsequent module-3 S's (higher S values) in conjunction with stimulus from the state look-ahead path.

1) Counting Path: Module-1 is a standard parallel synchronous binary 2-bit counter, which is responsible for low-order bit counting and generating future states for all module-3 S's in the counting path by pipelining the enable for these future states through the state look-ahead path. Fig. 2 depicts the (a) hardware schematic and (b) state diagram for module-1. Module-1 outputs $Q1Q0$ (the counter's two low-order bits) and $QEN1 = Q1$ AND $\overline{Q0}$ (the 1 in $QEN1$ denotes that this is the $QEN1$ for module-1). $QEN1$ connects to the module-2's DIN input.

The placement of module-2s in the counting path is critical to the novelty of our counter structure. Module-2s in the counting path act as a pipeline between the module-1 and module-3 1 and between subsequent module-3S s (see Fig. 1). Module-2 placement (coupled with state look-ahead logic described in Section II-A2) increases counter operating frequency by eliminating the lengthy AND-gate rippling and large AND gate fan-in and fan-out typically present in large width parallel counters. Thus, instead of the modules of higher significance requiring the ANDing of all enable signals from modules of lower significance, modules of higher significance (module-3 S s in our design) are simply enabled by the module-3 S's preceding module-2 and state look-ahead logic. Since the coupling of module-2 with module-3 1 introduces an extra cycle delay before module-3 1 is enabled, module-2's DIN is triggered when the module-1's count $Q1Q0 = 10$ (note that this is only the case for the left most module-2 in the counting path in Fig. 1, as subsequent module-2s require state look-ahead logic as well). Thus, the module-2s in the counting path provide a *1-cycle look-ahead mechanism* for triggering the module-3

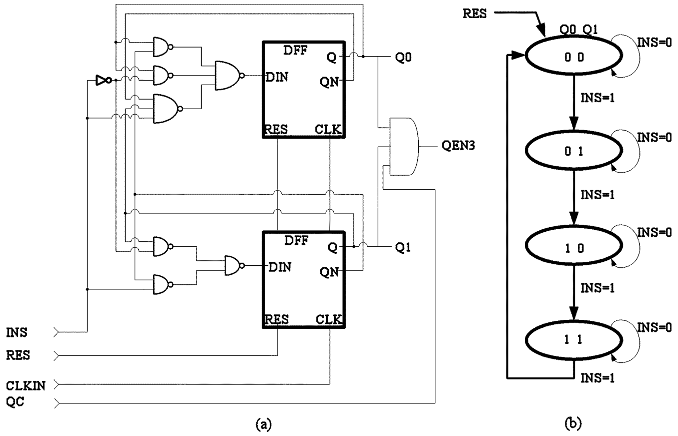


Fig. 3. Module-3S (a) hardware schematic and (b) state diagram. Note that the 3 in $QEN3$ denotes that this is the QEN for module-3.

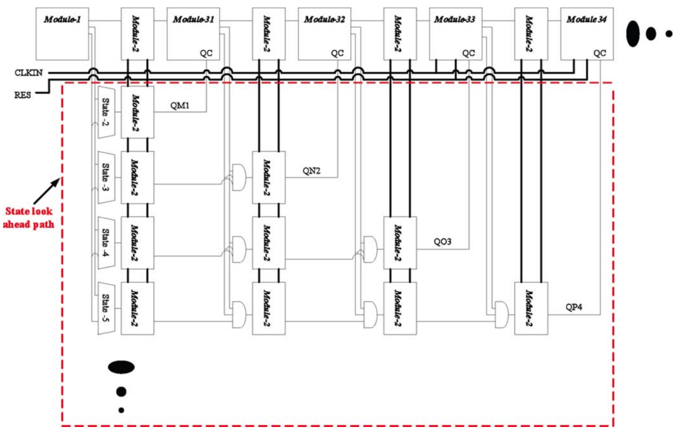


Fig. 4. Generalized counter topology for an N-bit counter showing state look-ahead path details.

S's, enabling the module-2s to maintain a constant delay for all stages and all module-3 S's to count in parallel at the rising clock edge instead of waiting for the overflow rippling in a standard ripple counter.

Fig. 3 depicts the (a) hardware schematic and (b) state diagram for module-3 S. Module-3 S is a parallel synchronous binary 2-bit counter whose count is enabled by INS. INS connects to the Q output of the preceding module-2. Module-3 S outputs $Q1Q0$ (which connect to the appropriate count output bits QX and $Q(X-1)$ as shown in Fig. 1) and $QEN3 = Q1$ AND $Q0$ AND QC (the 3 in $QEN3$ denotes that this is the QEN for module-3 S). The state look-ahead logic provides the QC input (details discussed in Section II-A2). $QEN3$ connects to the subsequent module-2's DIN input and provides the one-cycle look ahead mechanism.

2) State Look-Ahead Path: The state look-ahead path operates similarly to a carry look-ahead adder in that it decodes the low-order count states and carries this decoding over several clock cycles in order to trigger high-order count states. The state look-ahead logic is principally equivalent to the one-cycle look-ahead mechanism in the counting path. For example, in a 4-bit counter constructed of two 2-bit counting modules, the counting path's module-2 decodes the low-order state $Q1Q0 = 10$ and carries this decoding across one clock cycle and enables $Q3Q2 = 01$ at module-3 1 (see Fig. 1) on the next rising clock edge. This operation is equivalent to decoding $Q1Q0 = 11$ and enabling $Q3Q2 = 01$ on the next immediate rising clock edge. The state look-ahead logic expands this principle to an X-cycle look-ahead mechanism. For example, in a traditional 6-bit ripple counter constructed of three 2-bit counting modules, the enabling of bits $Q5Q4$ happens only after decoding the overflow at $Q1Q0$ to enable $Q3Q2$ and decoding the overflow at $Q3Q2$ to enable $Q5Q4$. However, combining the one cycle look-ahead mechanism in the counting path for $Q3Q2 = 10$ and a two-cycle look-ahead mechanism for $Q1Q0 = 01$ from can enable $Q5Q4$ ($Q1Q0 = 01$ is pipelined across two cycles and $Q3Q2 = 10$ is pipelined across one cycle, thus enabling $Q5Q4$ at the next rising clock edge (further details will be discussed in Section II-C). Thus, enabling the next state's high-order bits depends on *early overflow pipelining* across clock cycles through the module-2s in the state look-ahead path. This state look-ahead logic organization and operation avoids the use of an overhead delay detector circuit that decodes the low order modules to generate the enable signals for higher order modules, and enables all modules to be triggered concurrently on the clock edge, thus avoiding rippling and long frequency delay.

Fig. 4 depicts a generalized N-bit counter topology, revealing state look-ahead path details. Module-2s in the state look-ahead logic are responsible for propagating (pipelining) the early overflow detection to the appropriate module-3 S. Early overflow is initiated by the module-1 through the left-most column of decoders (state-2, state-3, etc.). The Q output of the right-most module-2 ($QM1, QN2$, etc.) in each early overflow pipelining chain is connected to the QC input of the appropriate module-3 S. The module-3 S's output $QEN3 = Q1$ AND $Q0$ AND QC (note $Q1$ and $Q0$ refer to the two counting bits stored internally at each module-3 S and do not refer to $Q1$ and $Q0$ of the outputted count value), signaling that not only has that module-3 S overflowed, but all modules preceding that module-3 S have also overflowed, thus enabling the count in the subsequent module-3 $(S+1)$.

Each module-2s early overflow pipelining chain is preceded by a small logic block (*State-X*), which decodes the appropriate $Q1Q0$ value for early overflow pipelining. X denotes the number of clock cycles that the early overflow pipelining must carry through. For example, *State-3* means that the early overflow signal must carry through two clock cycles, and thus enable the appropriate module-3 S on the third clock cycle. Each State-X block consists of simple two-input AND logic that decodes the module-1's $Q1Q0$ output. Fig. 1 shows the internal logic for *State-2* and *State-3* as $\overline{Q1}Q0$ and $\overline{Q1Q0}$, respectively, and whose outputs $QB1$ and $QC1$ (see Fig. 1), respectively, are connected to the appropriate module-2s DIN input, thus starting the early overflow pipelining exactly X clock cycles before the overflow must be detected to enable counting in a module-3 S. Note that module-1 and module-3 S may be of arbitrary bit width, and thus the same look-ahead principle would equally apply.

B. Derivation of Counter State Equations

In this subsection, we present the derivation of each counter state equation based on the early overflow pipelining equation (which preceding bit values are necessary to enable a subsequent counter bit to change). As an example, we derive the counter state equations for the sample 8-bit counter presented in Fig. 1. We denote a *past* counter state using lower case $q7q6q5q4q3q2q1q0$ and the next counter state using upper case $Q7Q6Q5Q4Q3Q2Q1Q0$. Thus, the counter state equation necessary to enable $Q7Q6$ will contain $q5q4q3q2q1q0$ Consequently, $Q3Q2$ at module-3 1 is enabled by the past state $q1\overline{q0}$ from module-1, which carries through one clock cycle in the counting path's module-2, and enables module-3 1 on the next rising clock edge (the one-cycle look-ahead mechanism defined in Section II-A1).

The 4-bit counter state equation (counter state outputs of module-1 and module-3 1 in Fig. 1) can be expressed as

$$Q3Q2Q1Q0 = Q3Q2 \text{ Pipelined } (q1\overline{q0}) \tag{1}$$

where Pipelined (X) denotes that the past bit values represented by X must be pipelined across one clock cycle. This notation may be recursively applied such that Pipelined (Pipelined (X)) would pipeline X across two clock cycles, and so forth. Consequently, the 6-bit counter state equation (counter state outputs of module-1, module-3 1, and module-3 2 in Fig. 1) can be expressed as

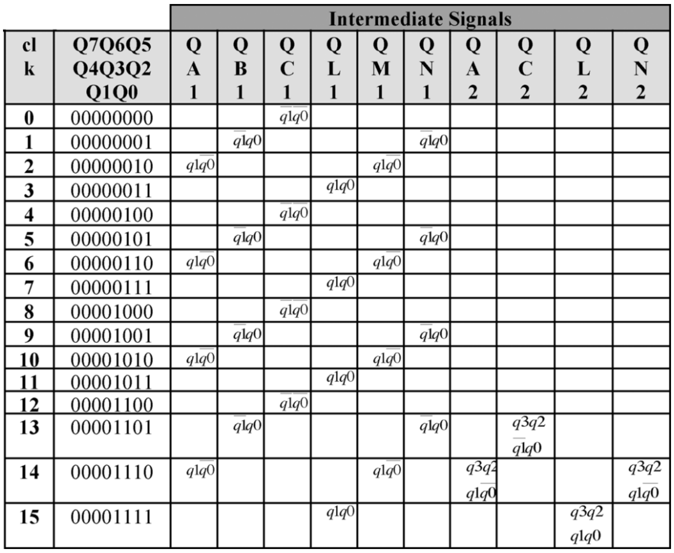
$$\begin{aligned} Q5Q4Q3Q2Q1Q0 = Q5Q4 &\text{ Pipelined} \\ &\times [(q3\overline{q2}) \text{ Pipelined } (\overline{q1}q0)]. \end{aligned} \tag{2}$$

The complete 8-bit counter state equation (counter state outputs of module-1, module-3 1, module-3 2, and module-3 3 in Fig. 1) can be expressed as:

$$\begin{aligned} Q7Q6Q5Q4Q3Q2Q1Q0 = Q7Q6 &\text{ Pipelined}[(q5\overline{q4}) \\ &\text{Pipelined}[(q3q2) \text{ Pipelined}(\overline{q1q0})]]. \end{aligned} \tag{3}$$

For (1), the past state $q1\overline{q0}$ is pipelined using the one-cycle look-ahead mechanism provided by the left-most module-2 in the counting path. For (2) and (3), the state look-ahead path provides the past states $\overline{q1}q0$ and $\overline{q1q0}$ through early overflow pipelining. Table I depicts all relevant intermediate signal values for the four least significant bits of the 8-bit counter in Fig. 1.

TABLE I
RELEVANT INTERMEDIATE SIGNAL (SEE FIG. 1) VALUES FOR COUNTER STATES 0–15



		Intermediate Signals									
clk	Q7Q6Q5 Q4Q3Q2 Q1Q0	QA1	QB1	QC1	QL1	QM1	QN1	QA2	QC2	QL2	QN2
0	00000000			$\overline{q1}\overline{q0}$							
1	00000001		$\overline{q1}q0$				$\overline{q1}q0$				
2	00000010	$q1\overline{q0}$				$q1\overline{q0}$					
3	00000011				$q1q0$						
4	00000100			$\overline{q1}\overline{q0}$							
5	00000101		$\overline{q1}q0$				$\overline{q1}q0$				
6	00000110	$q1\overline{q0}$				$q1\overline{q0}$					
7	00000111				$q1q0$						
8	00001000			$\overline{q1}\overline{q0}$							
9	00001001		$\overline{q1}q0$				$\overline{q1}q0$				
10	00001010	$q1\overline{q0}$				$q1\overline{q0}$					
11	00001011				$q1q0$						
12	00001100			$\overline{q1}\overline{q0}$							
13	00001101		$\overline{q1}q0$				$\overline{q1}q0$		$q3q2$ $\overline{q1}q0$		
14	00001110	$q1\overline{q0}$				$q1\overline{q0}$		$q3q2$ $q1\overline{q0}$			$q3q2$ $q1\overline{q0}$
15	00001111				$q1q0$					$q3q2$ $q1q0$	

C. Counter Area Analysis

In this subsection, we analyze the area overhead of our parallel counter architecture based on the number of internal components (which can easily be translated to gate counts). Module-1 is a 2-bit counter that provides a set of early overflow states to enable future states. In general, if module-1 is an m-bit counter, one early overflow state inputs into the counting path and the remainder of the early overflow states input into the state look-ahead path. Therefore, the total number of early overflow states (EO) generated by module-1 is

$$EO = 2^m - 1 \tag{4}$$

and the number of early overflow components (denoted as state-S in Fig. 4) (EO_Comp) required to propagate early overflow states in the state look-ahead path is

$$EO\_Comp = EO - 1. \tag{5}$$

In order to generalize a skeleton structure to apply to variable counter widths, we consider the number of module-2s associated with each module-3 S (one vertical column of module-2s for every module-3 S in Fig. 4). One module-2 is in the counting path while the remainder of the module-2s are in the state look-ahead path. The number of module-2s in the vertical column state look-ahead path $(M2\_VSLP)$ is

$$\text{M2\_}VSLP = 2m - (S + 1) \tag{6}$$

where S denotes module-3S's horizontal position.

Thus, the total number of module-2s in each vertical column $(M2\_V)$ (including both the counting and state look-ahead paths) is

$$M2\_V = \text{M2\_}VSLP + 1. \tag{7}$$

Using (4), the maximum allowable counter size (CS) is

$$CS = m + (2 * (2^m - 1)). \tag{8}$$

Using (8), the required number of module-3 S's in the counting path $(M3\_CP)$, which is equal to the number of module-2s in the counting path $(M2\_CP)$ is

$$M3\_CP = M2\_CP = \frac{(CS - m)}{2}. \tag{9}$$

TABLE II
NUMBER OF COMPONENTS AND TOTAL COUNTER RANGE FOR SAMPLE COUNTERS WITH RESPECT TO THE MODULE-1 SIZE IN BITS

	Number of Components		
Module-1 size **Component**	**m = 2-bits**	**m = 3-bits**	**m = 4-bits**
CS (8) (*Counter Range*)	8 (2^1 *to* 2^8)	17 (2^1 *to* 2^{17})	34 (2^1 *to* 2^{34})
M3_CP (9)	3	7	15
M2 (11)	6	28	120
AND_SLP (12)	1	15	91
EO_Comp (5)	2	6	14

Furthermore, the total number of module-2s in the state look-ahead path $(M2\_SLP)$ is

$$M2\_SLP = \frac{\left(\frac{CS-m}{2} - 1\right)\left(\left(\frac{CS-m}{2} - 1\right) + 1\right)}{2} \tag{10}$$

and, the total number of module-2s in the entire counter architecture $(M2)$ is

$$M2 = M2\_CP(9) + M2\_SLP(10). \tag{11}$$

The total number of AND logic in the state look-ahead path (AND_SLP) is

$$\text{AND}\_SLP = \frac{\left(\frac{CS-m}{2} - 2\right)\left(\left(\frac{CS-m}{2} - 2\right) + 1\right)}{2}. \tag{12}$$

Table II depicts component counts for various counter sizes based on the module-1 size in number of bits m.

III. TIMING ANALYSIS

In this section, we provide timing analysis for our parallel counter. We present detailed operational timing for a sample counting sequence, followed by clock period analysis.

A. Timing Diagram

Fig. 5 depicts the timing diagram for the sample 8-bit counter in Fig. 1, showing all related events, which occur for a start count state of 101000 (i.e., $Q0 = Q1 = Q2 = Q4 = 0$ and $Q3 = Q5 = 1$) and seven subsequent counts ending in a count state of 101111 (i.e., $Q0 = Q1 = Q2 = Q3 = Q5 = 1$ and $Q4 = 0$). In this example, we begin the counter at a state such that subsequent count increments affect most of the modules in Fig. 1, but yet the required explanation is manageable (note that since $Q7Q6$ do not change in this example and modules associated with those bits will not change, these signals are not shown in the timing diagram). Table III summarizes symbol notation definitions. Without loss of generality we assume all modules have an equal access time T_M, logic AND gates have a unit delay of T_A, and all DFFs are positive-edge triggered. In each cycle, we do not mention every signal event, only those relevant to the progression of the example.

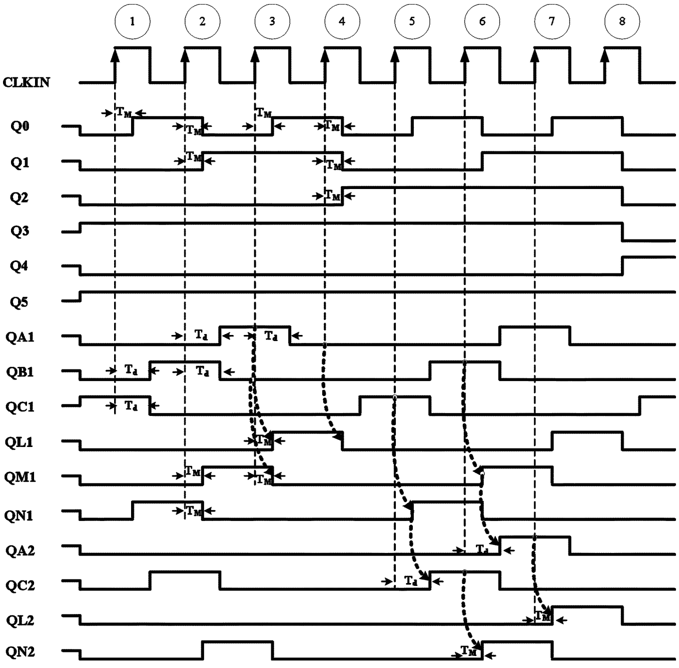


Fig. 5. Timing diagram for the 8-bit counter in Fig. 1 starting with an initial count value of 101000 and operating for seven subsequent count operations. Clock cycle counts are denoted along the top of the timing diagram.

TABLE III
SYMBOL NOTATION DEFINITIONS FOR THE TIMING DIAGRAM IN FIG. 5

SYMBOL	**DEFINITION**
T_M	Module access time (all modules)
T_A	Delay time of an AND gate
T_{NAND3}	Delay time of a three-input NAND
T_{NANDm}	Delay time of an m-input NAND gate
T_{AND3}	Delay time of a three-input AND gate
$T_{setup\text{-}hold}$	DFF setup time + DFF hold time
T_{CLKIN}	System Counter Clock Period
CLKIN	System Counter Clock signal
RES	Counter Reset signal
QA1	QEN1 of Module-1
QA2	QEN3 of Module-31
QA3	QEN3 of Module-32
$Q0$, $Q1$, $Q2$, $Q3$, $Q4$, $Q5$, $Q6$, $Q7$	Counter state (bit values)
$QA1$, $QA2$, $QA3$, $QL1$, $QL2$, $QL3$, $QB1$, $QC1$, $QM1$, $QN1$, $QC2$, $QN2$	Intermediate overflow pipelining signals (Fig. 1)

After CLKIN's rising edge in clock cycle 1, the count state updates to 101001 after a delay of T_M (module-1's $Q0 = 1$). The change to $Q0$ triggers the two-input AND gates in the state look-ahead path, and sets the early overflow detection signals $QB1 = 1$ and $QC1 = 0$ after a delay of $T_d = T_M + T_A$.

After CLKIN's rising edge in clock cycle 2, the count state updates to 101010 after a delay of T_M (module-1's $Q1Q0 = 10$). The change $Q1Q0$ state triggers module-1's internal two-input AND gate (see Fig. 2) with a delay of $T_d = T_M + T_A$, resulting in $QEN1 = QA1 = DIN = 1$ (where DIN refers to the input of the module-2 to left of module-1). Concurrently, in the state look-ahead path, $QM1 = 1$ after a delay of T_M, thus pipelining the early overflow detection signal from $QB1 = 1$ in the previous clock cycle, while $QB1 = 0$ after a delay of T_d due to module-1's state change. All subsequent module-3 S's $(S \geq 1)$ outputs remain constant.

After CLKIN's rising edge in clock cycle 3, the count state updates to 101011 (module-1's $Q1Q0 = 11$) and $QL1 = 1$ (due to pipelining the early overflow signal $QEN1 = QA1 = DIN = 1$) after a delay of T_M. Concurrently, $Q1Q0 = 11$ triggers $QA1 = 0$ after a delay of T_d and $QM1 = 0$ (due to pipelining the early overflow signal $QB1 = 0$) after a delay of T_M. All subsequent module-3 S's $(S \geq 1)$ outputs still remain constant.

Clock cycle four reveals the counter's novel parallel updating mechanism. After the rising clock edge in cycle four, the count state updates to $101100(Q2Q1Q0 = 100)$ through simultaneous updates of $Q2Q1Q0$ at module-1 and module-3 1 after a delay of T_M. These updates occur simultaneously (instead of waiting for the overflow pipelining inherent in ripple-style counters) because $QL1 = 1$ in the previous clock cycle (due to the one-cycle look-ahead mechanism). Concurrently, $QL1 = 0$ resets due to pipelining $QA1 = 0$ in clock cycle three and $QA2 = 0$ holds because $QM1 = 0$ although $Q3 = 1$. Meanwhile, the state look-ahead signal $QC1 = 1$ triggers after a delay of T_d, while all other signals $QB1 = QM1 = QN1 = QC2 = QN2 = 0$ hold. All subsequent module-3 S's $(S \geq 2)$ outputs still remain constant.

After CLKIN's rising edge in clock cycle five, the count state updates to $101101(Q0 = 1)$ after a delay of T_M, thus triggering $QB1 = 1$ and $QC1 = 0$ after a delay of T_d. $QN1 = 1$ pipelines the early overflow detection signal $QC1 = 1$ after a delay of T_M, thus triggering $QC2 = 1$ after a delay of T_d since $Q3Q2 = 0$. $QM1 = QN2 = 0$ still holds, but will trigger on the next clock cycle (clock cycle six) because their input module-2s store 1 during this clock cycle. Again, all subsequent module-3 S's $(S \geq 2)$ outputs still remain constant as well as module-3 1.

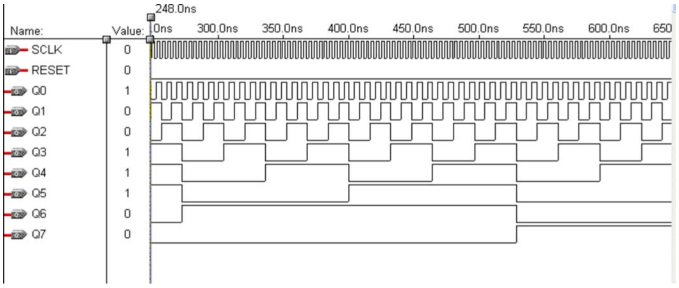
After CLKIN's rising edge in clock cycle six, the count state updates to $101110(Q1Q0 = 10)$ after a delay of T_M and $QA1 = 1$ after a delay of T_d. Concurrently, the state look-ahead path pipelines $QM1 = 1$ and $QN2 = 1$ after a delay of T_M, thus triggering $QA2 = 1$ after a delay of T_d (consisting of the $QM1$ access time and the module-3's internal AND gate delay since $Q3Q2 = 11$ were already high in this cycle as well as the previous cycle).

Consequently, a similar sequence repeats through the remaining clock cycles, with the key feature being counter state anticipation (parallel count operation) by pipelining early overflow detection through state look-ahead logic and the counting path, thus triggering all modules concurrently and avoiding rippling and high fan-in logic gates for detecting the next counter state. These mechanisms enable all modules to maintain the same gate delay (unit delay) regardless of counter width.

B. Clock Period Analysis

Using the timing diagram depicted in Fig. 5, we first derive the clock period T_{CLKIN} for an 8-bit counter based on the signal propagation through modules and AND gate logic, and subsequently generalize the derivation for an n-bit counter. T_{CLKIN} must be greater than both the counting and the state look-ahead path's delay (see Table III for symbol notation definitions)

$$T_{\text{CLKIN}} > T_M + T_{\text{AND3}} + T_{\text{setup-hold}} \tag{13}$$



Name:	Value:
SCLK	0
RESET	0
Q0	1
Q1	0
Q2	0
Q3	1
Q4	1
Q5	1
Q6	0
Q7	0

Fig. 6. Simulation waveforms for a synthesized HDL representation of our 8-bit parallel counter operating on an Altera MAX300A device. The clock (SCLK) frequency is 250 MHz and $V_{\text{dd}} = 3.3$ V.

where (13) is the delay between a module-3 S and a subsequent module-2. In addition, (13) can be represented as logic gate delays in order to avoid technology dependent factors, such that

$$\begin{aligned} T_{\text{setup-hold}} &= 1\text{INVERTER\_DELAY} \\ T_M &= 2\ \text{INVERTER\_DELAY} \\ T_{\text{AND3}} &= \frac{3}{2}\text{INVERTER\_DELAY} \end{aligned} \tag{14}$$

resulting in

$$T_{\text{CLKIN}} = 4.5\text{INVERTER\_DELAYS} \tag{15}$$

for the 8-bit counter in Fig. 1. Even though further speed enhancements are possible using various advanced design techniques for the module-1 and module-3S [9], [20], [27], [40], the purpose of this paper is to emphasize the architecture and parallel operation rather than improving individual components using high cost technology and special circuit design techniques. These improvements are orthogonal to our work.

In order to derive the clock period for larger counter widths, the access time in module-1 becomes the worst-case delay since larger module-1s generate more early overflow states as depicted in (4). Hence, for a 3-bit module-1 (17-bit maximum counter width), the module-1 delay becomes approximately equal to the module-3 S delay and, thus, implying no further delay than (13). For a 4-bit module-1 (34-bit maximum counter width), module-1 becomes the critical path such that

$$T_{\text{CLKIN}} > T_{\text{module-1}} + T_{\text{AND3}} + T_{\text{setup-hold}} \tag{16}$$

since the remainder of the modules (module-2s and module-3 S s), as well as the three-input AND gates, do not change (demonstrated in Fig. 4). The key emphasis of (16) is that the delay is bounded by module-1 for counter widths greater than 17-bits. In general, for very large counter widths (module-1s greater than 4-bits) further enhancements can be made using our topology as a sub-topology of a complete structure. We can further generalize the clock period of our proposed counter based on an m-bit module-1 and counter width derived in (8)

$$\begin{aligned} T_{\text{CLKIN}} > T_{\text{Module-1}}(T_M + T_{\text{AND3}} + T_{\text{setup/hold}}) + T_{\text{AND3}} \\ + T_{\text{setup/hold}}. \end{aligned} \tag{17}$$

On the other hand, T_{CLKIN} also has a secondary constraint due to the clock path's wire parasitic load since the clock drives all modules in the structure simultaneously, which may be alleviated by using clock distributing and buffering approaches [4], [5] with an efficient layout implementation.

IV. Experimental Results

We present synthesis and simulation results for the 8-bit parallel counter in Fig. 1. We provide both functional verification using a synthesized HDL representation and performance verification using an HSPICE simulation of our parallel counter.

A. Functional Verification

In order to illustrate our counter's parallel counting ability, Fig. 6 depicts the simulation waveforms for the top-level design outputs (counter value $Q7Q6Q5Q4Q3Q2Q1Q0$) for several count iterations. We synthesized the HDL using Quartus-II for an Altera MAX3000A FPGA device [3] running at 250 MHz (as we are presenting functional verification only, no optimizations were performed to increase operating frequency, and thus 250 MHz does not reflect the maximum operating frequency). The vertical axis shows traced logic values, while the horizontal time scale is represented in nanoseconds. All signal traces reflect the block diagram in Fig. 1 and follow precise counter timing, for example $Q0 = \mathrm{CLKIN}/2, Q1 = \mathrm{CLKIN}/4$, and $Q7 = \mathrm{CLKIN}/256$. In addition, it should be noted that all Q changes show similar timing delay with respect to CLKIN, exemplifying the key novel parallel counting feature of our design.

As reported by the Altera Quartus power analysis tools [3], the parallel counter has an average power dissipation of 5.41 mW. When synthesized using two-input NAND gates, the 8-bit parallel counter required a total of 235 gates, which consisted of 23 gates for the module-1, 9 gates for each module-2, and 30 gates for each module-3 S, as well as 8 output buffers and 2 input buffers for the clock and reset signals.

B. Performance Verification

To layout our parallel counter, we used Berkley's Magic circuit layout tool [5]. We generated the HSPICE net-list from the Magic layout using resistance-capacitance parasitic extraction and performed performance verification using HSPICE 0.15-μ m TSMC n-well CMOS technology [34] operating at 1.35 V and 125 °C, which provides worst case corner delays [15], [29]). Fig. 7 depicts the HSPICE simulation waveform captured using the Mentor Graphics Powertrain waveform viewer [23]. The HSPICE simulation achieved a maximum operating frequency of 2 GHz for our 8-bit parallel counter with a safe slew rate rise/fall margin of 0.2 ns/v.

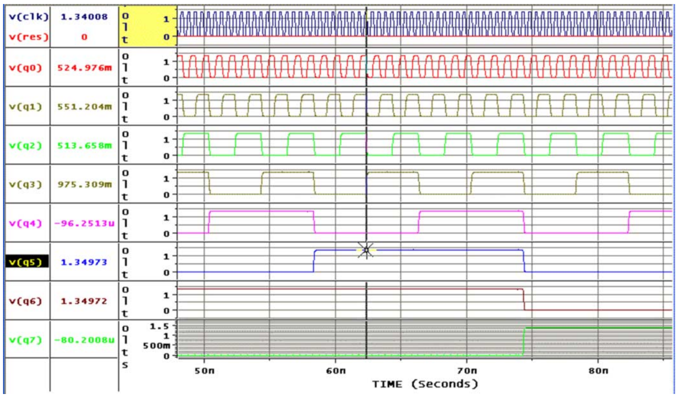


Fig. 7. HSPICE waveforms for our 8-bit parallel counter ($\mathrm{CLK} = 2$ GHz, $V_{\mathrm{DD}} = 1.65$ V) using TCMC 0.15-μm technology. The horizontal scale is in nanoseconds and the vertical scale is in volts.

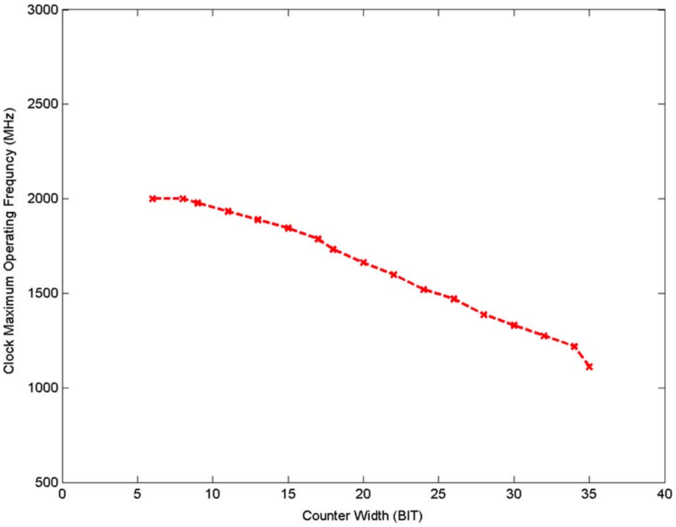


Fig. 8. Worst-case maximum clock frequency verses counter width (size) in bits.

Fig. 8 depicts worst-case maximum clock frequency verses counter width in bits. Overall, the clock frequency decreases at a steady rate of $\log_{6.5}(N)$ for increased counter widths ranging from 8 to 16 bits and 17 to 34 bits. This decrease reflects the increasing cost of parasitic components on the system clock (CLKIN) and not additional gate delays [(4) and (5) show that as the counter width increases between these two ranges, the number of components does not increase]. Subsequently, operating frequency is secondarily affected by clock path parasitic load, and minimizing this load using a clock tree and precise layout could increase operating frequency. On the other hand, a larger decrease in clock frequency occurs at counter widths of 8, 17, and 35 bits, which correspond to the increase in module-1 size to accommodate larger counter widths [(4) analyzes the increase in early overflow states generated and (17) analyzes the increased gate delay, which amounts to one early overflow state and one extra gate delay per increase in module-1 size, respectively].

Fig. 9 depicts worst-case total power consumption (static and dynamic) verses varying clock frequencies for our 8-bit parallel counter. We measure power consumption by setting the power supply voltage to 1.65 V at 0 °C for worst-case process corner [3], [15], [29], [34]. At clock frequencies of 2 GHz, 1 GHz, and 200 MHz, the parallel counter consumes 13.89, 5.78, and 1.872 mW, respectively. Overall, power consumption increases at a moderate linear rate of 7.3 μW/MHz with respect to increasing clock frequencies. The power consumption is primarily dominated by the module's dynamic switching activity and local interconnects on all modules (which is at most a three input logic fan-in and one output logic fan-out with the exception of the system clock). The transistor size is optimized for high performance operation and low power by limiting the transistor width

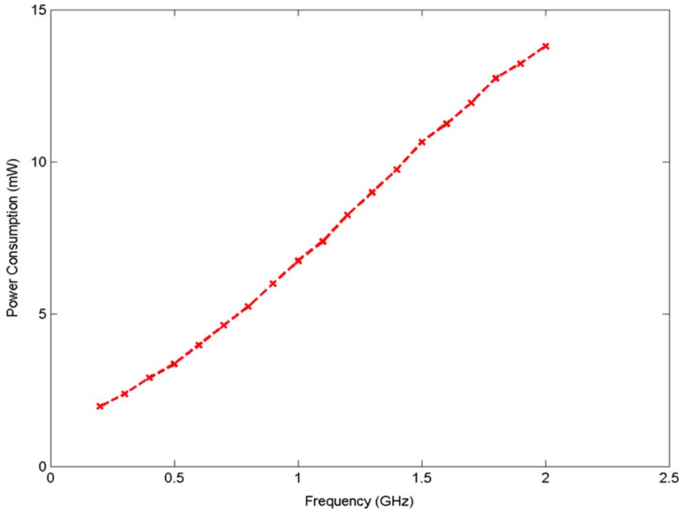


Fig. 9. Counter power consumption verses varying clock frequencies for our 8-bit parallel counter.

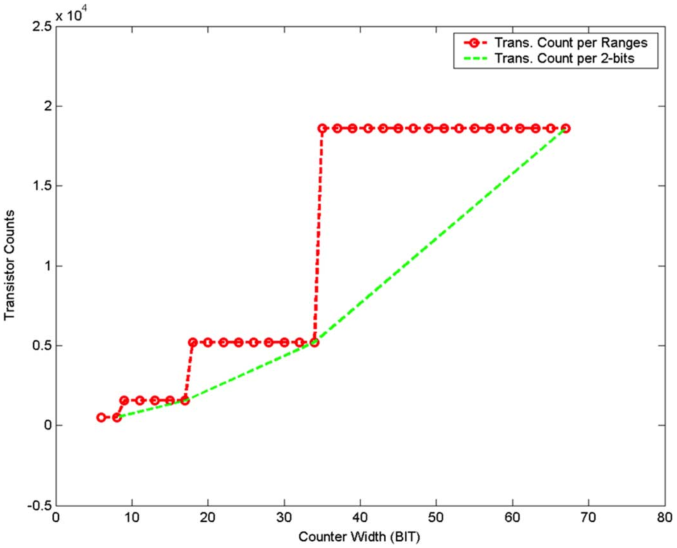


Fig. 10. Total CMOS transistor count verses counter bit size.

TABLE IV
TOTAL AND COMPONENT TRANSISTOR REQUIREMENTS FOR THE 8-BIT COUNTER

Component	Number of Components	Transistors Per Components	Total Transistor Count
Module-1	1	70	70
Module-2	6	90	144
Module-3	3	24	270
States Decoder	2	9	18
3-input AND	1	8	8
			Total = 510

to 5 μm and the channel length to 0.15 μm, with the exception of clock buffer transistors, which are approximately 15 μm in width and 0.15 μm in length. Thus, minimizing the dynamic switching activity and preserving the static leakage power to on the order of nano-Watts.

Table IV summarizes the total number of components and transistors required per component for our 8-bit counter based on the component design modules depicted in Figs. 2 and 3 using CMOS transistor design structures. Module-1, module-2, and module-3 components require 70, 90, and 24 transistors, respectively. The complete 8-bit counter requires only 510 transistors, which equals approximately 78 125 μm^2 of silicon die area.

Fig. 10 depicts total transistor count verses counter width in bits for particular counter ranges (trans count per ranges) and per 2-bit increments in counter width (trans count per 2-bits). Transistor requirements increase by approximately 3× for each counter range increase (a subsequent increase in module-1 width). For counters widths ranging from 1 to 8 bits, 9 to 17 bits, 18 to 34 bits, and 35 to 67 bits, the parallel counter requires 510, 1 555, 5 206, and 18 584 transistors, respectively. Analyzed as transistor increase per 2-bit increments in counter width reveals a modest linear increase of approximately 1.2× per 2-bits.

C. *Comparison Results*

Table V compares our parallel counter to counter designs by Alioto *et al.* [2] (results reported as published), Kakarountas *et al.* [17] (results reported as published), and Yeh *et al.* [41] (we designed and simulated Yeh's design using 0.15 μm to obtain these results) in terms of power consumption in milliwatts, maximum clock frequency in gigahertz, and area requirements in number of transistors for an 8-bit counter. In Alioto's design, the number of cascading stages limits the maximum clock frequency and the biasing current is a factor in the maximum power consumption. These stages [2] were designed using a differential pair of MOS current-mode logic with level shifters, where the propagated voltage swings between stages ranged from (*Vdd-Vth*) to *Vth*. Alioto's design is not amenable to VLSI implementation and continued technology scaling since the operating margin is 1/3 V_{dd}. Additionally, even though Alioto's design has low power consumption, the design requires a shutdown power source due to the continuous standby current drawn at every stage's biasing circuitry and level shifters, thus consuming large leakage power. Table V shows that even though Alioto's design consumes less power and requires fewer transistors than our parallel counter, the maximum clock frequency is only 500 MHz.

Kakarountas *et al.* [17] proposed a parallel counter structure divided into segments of 4-bit systolic counters consisting of a cascaded series of T-type flip-flops (TFFs). The estimated clock frequency was reported as $\text{T} = 2 \times \text{T}_{\text{AND3}} + \text{T}_{\text{AND2}} + \text{T}_{\text{TFF}}$, where $\text{T}_{\text{AND3}}, \text{T}_{\text{AND2}}$, and T_{TFF} are the delays of a three-input AND gate, a two input AND gate, and TFF access time, respectively. However, the setup-hold times and the XOR gates for the TFFs should also be considered in the attainable clock frequency. Consequently, Kakarountas's design reduced the counter delay by using more DFFs inserted between systolic counter segments at the cost of more cascaded logic to detect the write sequence and reload values to the most-significant bits. These additions make Karakountas's design unattractive for wide-range counters due to increases in the chain of

TABLE V
COMPARISON OF OUR PROPOSED PARALLEL COUNTER WITH PREVIOUS WORK BY ALIOTO *ET AL.* [2], KAKAROUNTAS *ET AL.* [17], AND YEH *ET AL.* [41]

	Technology (μm)/VDD	*Power consumption at maximum frequency (mW)*	*Maximum frequency (GHz)*	*Area in number of transistors for an 8-bit counter*	*Design Drawbacks*
Alioto et al. [2]	0.18/1.8V	2.21	0.5	160	* Less attractive for continued technology scaling * Requires biasing circuit
Kakarountas et al. [17]	0.6/5V	21.3	0.467	286	* Irregular structure * Output not in radix-2 binary
Yeh et al. [41]	0.15/1.5V	7.64	1.1	373	* Irregular structure * Output latency depends on counter size
Our parallel counter	0.15/1.5V	13.89	2	510	* Large area

cascaded detected logic gates. Kakarountas *et al.* [17] reported controversial results when for counter widths of 32-bits and larger. Furthermore, the counter output was not in radix-2 binary format, thus requiring further processing of the counter output and increasing the counter delay.

Yeh *et al.* [41] presented a parallel counter with a carry-select structure and a half-adder component associated with every DFF in order to generate the appropriate anticipated states by a factor of approximately 2. Yeh's design structure was divided into irregular sections that required different partitions for different counter sizes and more bits were associated with partitions of higher significance. The maximum clock frequency was limited by the half-adder delay, the chain of large fan-in AND gates, and the time required to load the DFFs. The design used a long chain of AND gates in order to detect the anticipated states for the partitions of lower significance, which made the design irregular and less attractive for wide-range counters. Consequently, the structure was enhanced by inserting DFFs in the AND chain in order to alleviate the long delay required for wide-range counters, with the side effect of the count being read after a two or three cycle delay (depending on the counter length), instead of a one cycle delay as in classical counter designs.

V. CONCLUSION

In this paper, we presented a scalable high-speed parallel counter using digital CMOS gate logic components. Our counter design logic is comprised of only 2-bit counting modules and three-input AND gates. The counter structure's main features are a pipelined paradigm and state look-ahead path logic whose interoperation activates all modules concurrently at the system's clock edge, thus providing all counter state values at the exact same time without rippling affects. In addition, this structure avoids using a long chain detector circuit typically required for large counter widths. An initial m-bit counting module pre-scales the counter size and this initial module is responsible for generating all early overflow states for modules of higher significance. In addition, this structure uses a regular VLSI topology, which is attractive for continued technology scaling due to two repeated module types (module-2s and module-3s) forming a pattern paradigm and no increase in fan-in or fan-out as the counter width increases, resulting in a uniform frequency delay that is attractive for parallel designs. Consequently, the counter frequency is greatly improved by reducing the gate count on all timing paths to two gates using advanced circuit design techniques. However, extra precautions must be considered during synthesis or layout implementations in order to aligned all modules in vertical columns with the system clock. This layout avoids setup and hold time violations, which might ultimately be limited by race conditions.

Results reveal that our counter frequency drops at a rate of $\log_{6.5}(N)$ (where N is the counter width in bits) due to parasitic components that inhibit large fan-out of the system clock path (since all modules operate simultaneously) and the delay of an initial module that increases in size as the counter width increases. This logarithmic frequency drop places our design amongst the fastest counter designs reported in literature, to the best of our knowledge. Furthermore, area requirements increase at a modest rate of 1.2× transistors per 2-bit increase in counter width. Similarly, power consumption increases at a moderate linear rate of 7.3 μW/MHz with respect to each doubling of the clock frequency. Finally, the counter output is determined directly on-the-fly with no additional decoding latency necessary to decode the final output pattern as with most counter designs.

REFERENCES

[1] S. Abdel-Hafeez, S. Harb, and W. Eisenstadt, "High speed digital CMOS divide-by-N frequency divider," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, 2008, pp. 592–595.

[2] M. Alioto, R. Mita, and G. Palumbo, "Design of high-speed power-efficient MOS current-mode logic frequency dividers," *IEEE Trans. Circuits Syst. II, Expr. Briefs*, vol. 53, no. 11, pp. 1165–1169, Nov. 2006.

[3] Altera Corp., Santa Clara, CA, "FLEX8000, field programmable gate array logic device," 2008.

[4] A. Bellaour and M. I. Elmasry, *Low-Power Digital VLSI Design Circuits and Systems*. Norwell, MA: Kluwer, 1995.

[5] J. Ousterhout, Berkeley, 1980, "Berkley magic layout tools," 1980. [Online]. Available: http://opencircuitdesign.com/magic/

[6] B. Chang, J. Park, and W. Kim, "A 1.2 GHz CMOS dual-modulus prescalar using new dynamic D-type flip-flops," *IEEE J. Solid-State Circuits*, vol. 31, no. 5, pp. 749–752, May 1996.

[7] M. Ercegovac and T. Lang, "Binary counters with counting period of one half adder independent of Counter size," *IEEE Trans. Circuits Syst.*, vol. 36, no. 6, pp. 924–926, Jun. 1989.

[8] M. D. Ercegovac and T. Lang, *Digital Arithmetic*. San Mateo, CA: Mogan Kaufmann, 2004.

[9] A. Fard and D. Aberg, "A novel 18 GHZ 1.3 mW CMOS frequency divider with high input sensitivity," in *Proc. Int. Symp. Signals, Circuits, Syst.*, Jul. 2005, pp. 409–412.

[10] C. Foster and F. Stockton, "Counting responders in an associative memory," *IEEE Trans. Comput.*, vol. 20, no. 12, pp. 1580–1583, Dec. 1971.

[11] D. C. Hendry, "Sequential lookahead method for digital counters," *IEEE Electron. Lett.*, vol. 32, no. 3, pp. 160–161, Feb. 1996.

[12] N. Homma, J. Sakiyama, T. Wakamatsu, T. Aoki, and T. Higuchi, "A systematic approach for analyzing fast addition algorithms using counter tree diagrams," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, 2004, pp. V-197–V-200.

[13] B. Hoppe, C. Kroh, H. Meuth, and M. Stohr, "A 440 MHz 16 bit counter in CMOS standard cells," in *Proc. IEEE Int. ASIC Conf.*, Sep. 1998, vol. 244, pp. 241–244.

[14] Y. Ito, K. Nakano, and Y. Yamagishi, "Efficient hardware algorithms for n choose K counters," in *Proc. 20th Int. Parallel Distrib. Process. Symp.*, Apr. 2006, pp. 1–8.

[15] Y. I. Ismail and E. G. Friedman*, On-Chip Inductance in High Speed Integrated Circuits*. Norwell, MA: Kluwer, 2001.

[16] R. F. Jones, Jr. and E. E. Swartzlander, Jr., "Parallel counter implementation," in *Proc. 10th Symp. Comput. Arith.*, 1992, pp. 382–385.

[17] A. P. Kakarountas, G. Theodoridis, K. S. Papadomanolakis, and C. E. Goutis, "A novel high-speed counter with counting rate independent of the counter's length," in *Proc. IEEE Int. Conf. Electron., Circuits Syst. (ICECS)*, UAE, Dec. 2003, pp. 1164–1167.

[18] R. H. Katz*, Contemporary Logic Design*. Redwood City, CA: Benjamin/Cummings, 1994.

[19] Y. Leblebici, H. Ozdemir, A. Kepkep, and U. Cilingiroglu, "A compact high-speed (31, 5) parallel counter circuit based on capacitive threshold-logic gates," *IEEE J. Solid-State Circuits*, vol. 31, no. 8, pp. 1177–1183, Aug. 1996.

[20] B. Liang, D. Chen, B. Wang, D. Cheng, and T. Kwasniewski, "A 43-Ghz static frequency divider in 0.13 μM standard CMOS," in *Proc. IEEE Canadian Conf. Elect. Comput. Eng. (CCECE)*, May 2008, pp. 111–114.

[21] P. Lin, K. E. Kerr, and A. S. Botha, "A novel approach for CMOS parallel counter design," in *Proc. 25th EUROMICRO Conf.*, 1999, pp. 112–119.

[22] D. R. Lutz and D. N. Jayasimha, "Programmable modulo-K counters," *IEEE Trans. Circuits Syst. I, Fundam. Theory Appl.*, vol. 43, no. 11, pp. 939–941, Nov. 1996.

[23] Mentor Graphics, Wilsonville, OR, "Mentor graphics powertrain," 1981. [Online]. Available: http://www.mentor.com/

[24] N. Nikityuk, "Use of parallel counters for triggering nuclear instruments and methods in physics research," , vol. A321, no. 3, pp. 571–582, Oct. 1992.

[25] B. Parhami*, Computer Arithmetic: Algorithms and Hardware Designs*. London, U.K.: Oxford Univ. Press, 2000.

[26] K. Z. Pekmestzi and N. Thanasouras, "Systolic frequency-dividers counters," *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.*, vol. 41, no. 11, pp. 775–776, Nov. 1994.

[27] B. Razavi*, RF Microelectronics*. Upper Saddle River, NJ: Prentice-Hall, 1998.

[28] J. Sakiyama, T. Aoki, and T. Higuchi, "Counter tree diagrams for design and analysis of fast addition algorithms," in *Proc. 33rd Int. Symp. Multiple_Valued Logic (ISMVL)*, May 2003, pp. 1–8.

[29] E. Sanchez-Sinencio and A. G. Andreou*, Low-Voltage/Low Power Integrated Circuits and Systems*. New York: Wiley-IEEE press, 1998.

[30] M. R. Stan, "Synchronous up/down counter with period independent of counter size," in *Proc. IEEE Symp. Comput. Arith.*, Asilomar, CA, Jul. 1997, pp. 274–281.

[31] M. R. Stan, "Systolic counters with unique zero state," in *Proc. IEEE Proc. Int. Symp. Circuits Syst. (ISCAS)*, 2004, pp. II-909–II-912.

[32] M. R. Stan, A. F. Tenca, and M. D. Ercegovac, "Long and fast up/down counters," *IEEE Trans. Comput.*, vol. 47, no. 7, pp. 722–735, Jul. 1998.

[33] J. E. Swartzlander, "A review of large parallel counter designs," in *Proc. IEEE Comput. Soc. Ann. Symp. VLSI Emerg. Trends VLSI Syst. Des. (ISVLSI)*, 2004, pp. 89–98.

[34] Taiwan Semiconductor Manufacturer Corp., Taiwan, "0.15 μm CMOS ASIC process digests," 2002.

[35] J. E. Vuillemin, "Constant time arbitrary length synchronous binary counters," in *Proc. IEEE 10th Symp. Comput. Arith.*, 1991, pp. 180–183.

[36] D. Wong, G. DeMicheli, M. Flynn, and R. Huston, "A bipolar population counter using wave piplelining to achieve 2.5x normal clock frequency," *IEEE J. Solid-State Circuits*, vol. 27, no. 5, pp. 745–753, May 1992.

[37] Xilinx, Inc., San Jose, CA, "Xilinx; The programmable logic data book," Aug. 1997.

[38] H. Yan, M. Biyani, and Kenneth, "A high-speed CMOS dual-phase dynamic-pseudo NMOS ((DP)^2) latch and its application in a dual-modulus prescaler," *IEEE J. Solid-State Circuits*, vol. 34, no. 10, pp. 1400–1404, Oct. 1999.

[39] C.-Y. Yang, G.-K. Dehng, J.-M. Hsu, and S.-I. Liu, "New dynamic flip-flops for high-speed dual-modulus prescaler," *IEEE J. Solid-State Circuits*, vol. 33, no. 10, pp. 1568–1571, Oct. 1998.

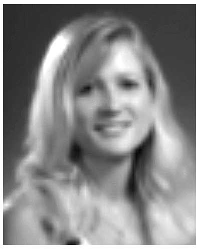
[40] K. Yamamoto and M. Fujishima, "4.3 GHz 44 uW CMOS frequency divider," in *Proc. IEEE Int. Solid-State Circuits Conf.*, 2004, pp. 104–105.

[41] C. Yeh, B. Parhami, and Y. Wang, "Designs of counters with near minimal counting/sampling period and hardware complexity," in *Proc. Asilomar Conf. Signals, Syst., Comput.*, 2000, pp. 894–898.

[42] D. Zhang, G. Jullien, W. Miller, and E. Swartzlander, "Arithmetic for digital neural networks," in *Proc. IEEE 10th Symp. Comput. Arith.*, Jun. 1991, pp. 58–63.

Saleh Abdel-Hafeez (M'02) received the Ph.D. degree in computer engineering from the University of Texas at El Paso, El Paso, and the M.S. degree from New Mexico State University, University Park.

He was a former Senior Member of technical staff at S3.inc and Viatechnologies.com in the area of mixed signal IC's design. He also was adjunct Professor with the Computer Engineering Department, Santa Clara University, Santa Clara, CA, from 1998 to 2002. He is currently the Chairman of the Computer Engineering Department, Jordan University of Science and Technology, Irbid, Jordan. He has two distinguished USA patents 6 265 509 and 6 356 509 with S3.inc. His current research interest includes the area of high speed IC's computer arithmetic algorithms and mixed signal design.



Ann Gordon-Ross (M'00) received the B.S. and Ph.D. degrees in computer science and engineering from the University of California, Riverside, in 2000 and 2007, respectively.

She is currently an Assistant Professor with the Department of Electrical and Computer Engineering, University of Florida, Gainesville. She is also a member of the NSF Center for High Performance Reconfigurable Computing (CHREC) at the University of Florida. Her research interests include embedded systems, computer architecture, low-power design, reconfigurable computing, dynamic optimizations, hardware design, real-time systems, and multi-core platforms.