An Optimized Fine Grain Domino Asynchronous Pipeline Design for Low Power

J. Sathya\textsuperscript{1}, T. Boobalan\textsuperscript{2} & V. S. Ramesh\textsuperscript{3}
PG Student\textsuperscript{1}, Assistant Professor\textsuperscript{2}, Director\textsuperscript{3}

Abstract-A novel design method of asynchronous domino logic pipeline, which focuses on improving the circuit efficiency and making asynchronous domino logic pipeline design more practical for a wide range of applications. The data paths are composed of a mixture of dual-rail and single-rail domino gates. Dual-rail domino gates are limited to construct a stable critical data path. Based on this critical data path, the handshake circuits are greatly simplified, which offers the pipeline high throughput as well as low power consumption. Moreover, the stable critical data path enables the adoption of single-rail domino gates in the noncritical data paths. A high-throughput and ultralow-power asynchronous domino logic pipeline design method, targeting to latch-free and extremely fine-grain or gate-level design. The data paths are composed of a mixture of dual-rail and single-rail domino gates. Dual-rail domino gates are limited to construct a stable critical data path. Based on this critical data path, the handshake circuits are greatly simplified, which offers the pipeline high throughput as well as low power consumption. Moreover, the stable critical data path enables the adoption of single-rail domino gates in the noncritical data paths. This further saves a lot of power by reducing the overhead of logic circuits. An $4 \times 4$ array style multiplier is used for evaluating the pipeline method. As a result, asynchronous domino logic pipeline has a small overhead in both handshake control logic and function block logic.

Keywords-Asynchronous pipeline, critical data path, dual-rail domino gate, single-rail domino gate, partial charge reuse.

I. Introduction

Asynchronous domino logic pipeline is an interesting pipeline style that can entirely avoid explicit storage elements between stages by exploiting the implicit latching functionality of domino logic gates. The latch less feature provides the benefits of reduced critical delay, smaller silicon area and low power consumption. However, asynchronous domino logic pipeline has a common problem that dual-rail domino logic has to be used to compose the domino data path. Single-rail domino logic cannot be used because it would break the domino data path since only non inverting logic gate can be implemented. In asynchronous design, the choice of handshake circuit protocols affects the circuit implementation (area, speed, power, robustness, etc.). The four-phase bundled-data protocol and the four-phase dual-rail protocol are two popular protocols that are used in most practical asynchronous circuits.

The four phase bundled-data protocol design most closely resembles the design of synchronous circuits. Handshake circuits generate local clock pulses and use delay matching to indicate valid signal. It normally leads to the most efficient circuits due to the extensive use of timing assumptions. On the other hand, the four-phase dual-rail protocol design is implemented in an elaborate way that the handshake signal is combined with the dual-rail encoding of data. Handshake circuits are aware of the arrival of valid data by detecting the encoded handshake signal, which allows correct operation in the presence of arbitrary data path delays. This feature is very useful for dealing with data path delay variations in advanced VLSI systems, such as asynchronous field-programmable gate arrays (FPGAs).

Asynchronous Circuits:

An asynchronous circuit, or self-timed circuit in a sequential digital logic circuit which is not governed by a clock circuit or global clock signal. Instead they often use signals that indicate completion of instructions and operations, specified by simple data transfer protocols. This type is contrasted with a synchronous circuit in which changes to the signal values in the circuit are triggered by repetitive pulses called a clock signal.
Most digital devices today use synchronous circuits. However asynchronous circuits have the potential to be faster, and may also have advantages in low power consumption, lower electromagnetic interference and better modularity in large systems.

Asynchronous circuits are an active area of research in digital logic design. In asynchronous circuits, there is no clock, and the state of the circuit changes as soon as the input changes. Since they don't have to wait for a clock pulse to begin processing inputs, asynchronous circuits can be faster than synchronous circuits, and their speed is theoretically limited only by the propagation delays of the logic gates. However, asynchronous circuits are more difficult to design and subject to problems not found in synchronous circuits. This is the result state of an asynchronous circuit can be sensitive to the relative arrival times of inputs at gates. If transitions on two input arrive at almost same time, the circuit can go into the wrong state depending on slight differences in the propagation delays of the gates. This is called a race condition. In synchronous circuits this problem is less severe because race conditions can only occur due to inputs from outside the synchronous system, asynchronous input. Although some fully asynchronous digital systems have been built, today asynchronous circuits are typically used in a few critical parts of otherwise synchronous systems where speed is at a premium, such as signal processing circuits. Asynchronous functions without a clock signal and so individual logic elements cannot be relied upon to have a discrete true/false state at any given time. Boolean logic is inadequate for this and so extensions are required.

An asynchronous pipelines are used in the asynchronous circuits and have their pipeline registers clocked asynchronously. Generally speaking, they use a request acknowledge system, wherein each stage can detect when it's "finished". When a stage is finished and the next stage has sent it a "request" signal, the stage sends an "acknowledge" signal to the next stage, and a "request" signal to the previous stage. When a stage receives an "acknowledge" signal, It clocks its input registers, thus reading in the data from the previous stage.

Critical Data Path:

The critical path is defined as the path between an input and an output with the maximum delay. Once the circuit timing has been computed by one of the techniques below, the critical path can easily be found by using a trace back method. The arrival time of a signal is the time elapsed for a signal to arrive at a certain point. To calculate the arrival time, delay calculation of all the components in the path will be required. Arrival times, and indeed almost all times in timing analysis, are normally kept as a pair of values - the earliest possible time at which a signal can change, and the latest. Another useful concept is required time. This is the latest time at which a signal can arrive without making the clock cycle longer than desired. The computation of the required time proceeds as follows: at each primary output, the required times for rise/fall are set according to the specifications provided to the circuit. Next, a backward topological traversal is carried out, processing each gate when the required times at all of its fan outs are known. The slack associated with each connection is the difference between the required time and the arrival time. A positive slack s at a node implies that the arrival time at that node may be increased by s without affecting the overall delay of the circuit. Conversely, negative slack implies that a path is too slow, and the path must be sped up (or the reference signal delayed) if the whole circuit is to work at the desired speed.

In static timing analysis, the word static alludes to the fact that this timing analysis is carried out in an input-independent manner, and purports to find the worst-case delay of the circuit over all possible input combinations. The computational efficiency (linear in the number of edges in the graph) of such an approach has resulted in its widespread use, even though it has some limitations. A method that is commonly referred to as PERT is popularly used in STA. In fact, PERT is a misnomer, and the so-called PERT method discussed in most of the literature on timing analysis refers to the critical path method (CPM) that is widely used in project management. While the CPM-based methods are the dominant ones in use today, other methods for traversing circuit graphs, such as depth-first search, have been used by various timing analyzers.

System Analysis

PSO is a well-known as implementation style of asynchronous domino logic pipeline based on dual-rail protocol. It is an important foundation for most later proposed styles. Since our proposed pipeline is also based on PSO, we will begin by reviewing PSO pipeline style, and then simply introducing two other advanced styles: 1) a timing-robust style called precharge half-buffer and 2) a high-throughput style called look ahead pipeline.
TABLE I

Code Table Of The Four-Phase Dual-Rail Encoding

<table>
<thead>
<tr>
<th>Codeword (w_t, w_f)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data 0: (0, 1)</td>
</tr>
<tr>
<td>Data 1: (1, 0)</td>
</tr>
<tr>
<td>Spacer: (0, 0)</td>
</tr>
<tr>
<td>Not used: (1, 1)</td>
</tr>
</tbody>
</table>

Fig. 1. Block diagram of PS0.

PS0:

Four-Phase Dual-Rail Protocol:

PS0 is designed based on the four-phase dual-rail protocol. Fig. 1 shows an example of data transfer based on the four-phase dual-rail protocol, and Table 1 shows the code table of the four-phase dual-rail encoding. The four-phase dual-rail encoding encodes a request signal into the data signal using two wires, (w_t, w_f). The data value 0 is encoded as (0, 1), and value 1 is encoded as (1, 0); the spacer is encoded as (0, 0); (1, 1) is not used.

Fig. 2.a) Dual-rail domino AND gate. Fig. 2.b) Two-bit completion detector

Structure of PS0:

Fig. 1 shows a block diagram of PS0. In PS0, each pipeline stage is composed of a function block and a completion detector. Each function block is implemented using dual-rail domino logic.

Fig. 2.a) shows an example of the dual-rail domino AND gate and Fig. 2.b) shows a 2-bit completion detector. A two-input NOR gate serves as the 1-bit completion detector to generate a bit done signal by monitoring the outputs of dual-rail domino gate.

Protocol of PS0:

The protocol of PS0 is quite simple. F(N) is precharged when F(N+1) finishes evaluation. F(N) evaluates when F(N+1) finishes its reset, or precharge.

II. Asynchronous pipeline based On Constructed critical data path:

Fig. 3 shows the block diagram of the proposed asynchronous pipeline (APCDP). The pipeline is designed based on a stable critical data path that is constructed using a special dual rail logic. The critical data path transfers a data signal and an encoded handshake signal. Noncritical data paths, composed of single-rail logic, only transfer data signal. A static NOR gate detects the dual-rail critical data path and generates a total done signal for each pipeline stage. The outputs of NOR gates are connected to the precharge ports of their previous stages. APCDP has the same protocol as PS0.

Finding a stable critical data path in function blocks is very important in the proposed design. The problem is that it is difficult to get a stable critical data path using traditional logic gates. Traditional logic gates have the gate delay data dependence problem; the gate delay is dependent on input data patterns.

However, actually, the critical data path varies according to different input data patterns. Because of the gate-delay data dependence problem, the gate function can be triggered early by the input bits (an and bn) regardless of the carry bit. Since the input bit travels faster in the buffer path than the carry bit in the ripple carry path, it cannot guarantee that the critical transition signal always presents on the ripple carry path. Adding delay elements is an intuitive way to construct a stable critical data path. However, this method needs complex timing analysis and would cause huge overhead of delay elements. This paper introduces an efficient solution that uses SLGs to construct the critical data path. The SLGs solve the gate-delay problem.
This feature does not only help to construct a stable critical data path but also enable the adoption of single-rail domino logic in the noncritical data paths. As a result, the proposed design is significantly area and power efficient.

### Synchronizing Logic Gate:

SLGs are dual-rail domino gates that have no gate-delay data-dependence problem. Fig. 4 shows the synchronizing AND gate and Table 2 truth table of dual-rail AND logic. The principle is that, in the pull-down network, there is exactly one path activated according to one data pattern, and the stack of all possible paths is kept constant at the sequential position. Compared with the traditional design, the false side logic expression is changed to \( \text{out}_f = a_t \cdot b_f + a_f \cdot (b_t + b_f) \).

This kind of gates is named as SLGs because they can synchronize their inputs. The SLGs verify that all data signal transitions have arrived on their inputs before changing their outputs. The characteristics of SLGs are listed as follows.

1) An SLG has a certain number, inputs’ number, of transistors in pull-down transistor paths at the sequential position

2) An SLG has no gate-delay data-dependence problem. Its gate delay mainly relates to the inputs number.

Fig. 4 Synchronizing AND gate, Table 2 Truth table of dual-rail AND logic.

The critical data path in dual-rail asynchronous pipeline can be easily constructed using SLGs and SLGLs.

### III. SYSTEM IMPLEMENTATION

#### Structure of APCDP:

Fig. 5 shows the structure of APCDP. The solid arrow represents a constructed critical data path (dual-rail data path), the dotted arrow represents the noncritical data paths (single-rail data paths), and the dashed arrow represents the output of single-rail to dual-rail encoding converter. In each pipeline stage, a static NOR gate is used as 1-bit completion detector to generate a total done signal for the entire data paths by detecting the constructed critical data path. Driving buffers deliver each total done signal to the precharge/evaluation control port of the previous stage. Since the completion detector only detects the constructed critical data path, the noncritical data paths do not have to transfer encoded handshake signal anymore.

#### Construction of the Critical Data Path:

It is difficult to construct a stable critical data path using traditional logic gates for their gate-delay data-dependence problem. The critical signal transition varies from one data path to others according to different input data patterns. Since SLGs have solved the gate-delay data-dependence problem, a stable critical data path can be easily constructed by the following steps:

1) finding a gate (named as Lin gate) that has the largest number of inputs in each pipeline stage;

2) changing these Lin gates to SLGs;

The first pipeline stage, the critical signal transition is on the output of the SLG because all gates evaluate at the same time for the input control of latches or registers. After linking each pipeline stage’s SLG together, the SLG in the following pipeline stage would be the last gate to start evaluation since it always waits for the critical signal transition from the previous SLG. As a result, the linked SLG data path becomes a stable critical data path. Linking each pipeline stage’s SLG together is partially done in the process of selecting Lin gate in each pipeline stage. When searching Lin gate, there might be more than one
option. It is best to select the Lin gate that is originally linked to the Lin gate in the following pipeline stage. After changing these Lin gates to SLGs, SLGs are naturally linked.

**Encoding Conversion:**

Since the completion detector detects only the constructed critical data path, the noncritical data paths do not have to transfer encoded handshake signal anymore. The logic overhead in the noncritical data paths can be reduced using single-rail domino gates instead of dual-rail domino gates. However, single-rail domino gate and dual-rail domino gate use different encoding schemes. It has encoding compatibility problem when a single-rail domino gate connects to a dual-rail domino gate. Encoding converter needs to be designed to solve the problem.

**Fig.6 Encoding converters.** (a) Intuitive design. (b) Proposed design.

<table>
<thead>
<tr>
<th>pc</th>
<th>in</th>
<th>out</th>
<th>out</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

**Table 3**

Truth table of Encoding Converter

**Fig.6** shows two implementations of encoding converter. Table 3 shows the truth table. In precharge phase pc=0, encoding converter outputs a dual-rail data0 (out, out)=(0, 1). In evaluation phase pc=1, if the input is a single-rail data0 in =0, the converter keeps the dual-rail data0. If the input is a single-rail data1 in=1, the converter outputs a dual-rail data1 (out, out)=(1, 0).

This is focus on improving the conversion from the single-rail data1 in =1 to the dual-rail data1 (out, out)=(1, 0). Fig.6,(b) shows the proposed design of the encoding converter. When the converter enters the evaluation phase, the input in =1 can immediately pull down out. No matter the output (out, out) is a instant spacer (0, 0) or the valid data1 (1, 0), it effectively protects the data transfer error. On the other hand, the intuitive design in Fig.6,(a) has a longer signal transition delay.

**IV. Asynchronous Pipeline Based On constructed Critical Data Path**

**A. Overview**

The pipeline is designed based on a stable critical data path that is constructed using a special dual-rail logic. The critical data path transfers a data signal and an encoded handshake signal. Noncritical data paths, composed of single-rail logic, only transfer data signal. A static NOR gate detects the dual-rail critical data path and generates a total done signal for each pipeline stage. The outputs of NOR gates are connected to the precharge ports of their previous stages.

Such design method has two merits. First, the completion detector is simplified to a single NOR gate, and the detection overhead is not growing with the data path width. Second, the overhead of function block logic is reduced by applying single-rail logic in noncritical data paths. As a result, APDCP has a small overhead in both handshake control logic and function block logic, which greatly improves the throughput and power consumption.

**B. Logic Gates**

In VLSI circuits, it is difficult to get a stable critical data path using traditional logic gates due to the gate-delay data-dependence problem. The true side of logic is implemented by out\_t=a\_t·b\_t and the false side by out\_f=a\_f+b\_f. Table II shows the states of pull-down transistorpaths on different data patterns. In traditional dual-rail domino AND gate, there are three transistorpaths: 1) [a\_t,b\_t]; 2) [a\_f]; and 3) [b\_f]. First of all, these paths have different number of transistors at the sequential position. When they turn on, respectively, [a\_f] and [b\_f] cause less delays than [a\_t,b\_t]. Moreover, when the data pattern is\{0,1,0,1\}, [a\_f] and [b\_f] will be bothON, which leads to a much quicker signal transfer. As a result, the gate delay has a large variation depending on different data patterns. To solve the gate-delay data-dependence problem, SLG and SLGL are introduced.

**C. Results**

The results show that APDCP has high throughput, the small-est transistor count, and the lowest forward latency in all designs. The results show that APDCP is the most power efficient design.

1) Transistor Count: Table IV shows that
APCDP, respectively, reduces the transistor count by 25.3% and 18.1% compared with LP2/2-SR and Sync. APCDP uses a mixture of dual-rail domino logic and single-rail domino logic. The single-rail domino logic gates in the noncritical data paths save a lot of the transistor count. Although the SLGs and SLGLs in APCDP consume more transistors than traditional dual-rail domino gates, they are in a small quantity (only 56, used in the critical data path), which have small impact on the transistor count.

The results also show that the transistor count of LP2/2-SR is larger than that of synchronous pipeline, which indicates that conventional latchless pipelines are difficult to realize the potential advantage of small silicon area. The main reason is the dual-rail encoding overhead in domino data path. Because of the latchless feature, a lot of implicit storage elements (dual-rail domino buffers) have to be added at each pipeline stage to store data. These added dual-rail domino buffers cause a large overhead. Although LP2/2-SR is significantly more area efficient than LP2/2, it still consumes more transistors than Sync and Sync-CG.

In addition to the transistor count, total FET width is a better metric to figure the relative difference of capacitance between designs. Table IV shows that APCDP, respectively, reduces the total FET width by 34.8% and 39.3% compared with LP2/2-SR and Sync. An interesting result is that LP2/2-SR has a smaller total FET width even it has a larger transistor count compared with Sync. This is because domino gates use keepers to protect charge leakage problem [20]. These keepers have a very small transistor size.

2) Latency: APCDP and LP22-SR have about one-third lower forward latency than Sync and Sync-CG. This is because latchless design has no sequential overhead (no registers or latches) on its forward path. Compared with LP22-SR, APCDP has a little larger latency. The latency is sacrificed for constructing the stable critical data path. Fortunately, this degradation is not serious.

3) Throughput: The performances of throughput are evaluated without considering design margins, which are all ideal results from the schematic simulations.

The results show that LP2/2-SR has the best throughput performance. This benefits from the bundled-data asynchronous design of LP2/2-SR. Traditional dual-rail domino data paths in LP2/2-SR actually have better signal transition speed than the data paths composed of SLGs/SLGLs. Bundled-data design can exploit this benefit to increase the pipeline throughput. However, delay margins need to be added in practical bundled-data design, which would decrease the performance of throughput.

Because of the dual-rail critical data path, does not have to add design margins in practical

In Section III-D1, it shows that the pipeline structure of APCDP originally supplies some time margins. Although APCDP has a slower pipeline speed and a higher forward latency than LP2/2-SR in the ideal evaluation, it is possible that APCDP may have a faster pipeline speed and a lower latency than LP2/2-SR in practical design if the timing margins required in LP2/2-SR exceed the detection overhead in APCDP. This can make the estimation of matching delay in LP2/2-SR overly conservative with a negative impact on performance.

Although the throughput performance can be improved using fine-grain design, the power consumption increases simultaneously. Therefore, Sync and Sync-CG are carefully designed considering the tradeoff between through-put and power. Although Sync and Sync-CG have the same throughput performance with APCDP, they hardly can win APCDP in practical design because synchronous design has to add design margins.

4) Power: The energy consumption of VLSI circuits relates to the toggling rate in data paths. In APCDP, the adoption of single-rail domino gates in the noncritical data paths saves not only silicon area by reducing transistor count but also energy consumption by reducing the toggling rate.

V. Conclusion

A novel design method of asynchronous domino logic pipeline. The pipeline is realized based on a constructed critical data path. The design method greatly reduces the overhead of handshake control logic as well as function block logic, which not only increases the pipeline throughput but also decreases the power consumption. Moreover, the stable critical data path enables the adoption of single-rail domino gates in the noncritical data paths. This further saves a lot of power by reducing the overhead of logic circuits It is even comparable with a synchronous pipeline with sequential clock gating. It is even comparable with a synchronous pipeline with sequential clock gating. By using synchronous domino logic pipeline design method, it is extremely fine grain method greatly reduces the power consumption and delay. Hybrid logic design represents the both dual-rail and single rail. Hybrid logic design provides the low delay. PCRs connected to the hybrid logic design. PCR [partial charge reuse] mechanism instead of NOR gate to
reduce area as well as power consumption. The evaluation results show that the proposed design has better performance than a bundled-data asynchronous domino logic pipeline (LP2/2-SR).

ACKNOWLEDGEMENT

The authors acknowledge the contributions of the students, faculty of Sri Eshwar College of Engineering, Kinathukadavu for helping in the design and tool support. The authors also thank the anonymous reviewers for their thoughtful comments that helped to improve this paper.

REFERENCES


