International Journal of Research
Available at https://edupediapublications.org/journals
p-ISSN: 2348-6848
e-ISSN: 2348-795X
Volume 03 Issue 11
July2016

Low Power Asynchronous Domino Logic Pipeline Design Strategy

[1] P.SRAVANTHI, [2] P.MURALIKRISHNA

[1] *M.Tech student, Department of ECE, Sri Krishnadevaraya University College Of Engineering & Technology, Ananthapuramu.*
[2] *Lecturer in Department of ECE, Sri Krishnadevaraya University College Of Engineering & Technology, Ananthapuramu.*

ABSTRACT

In today's world pipelining is a key element of high-performance design. Distributed synchronization is one of the key strengths and one of the major difficulties of synchronous pipelining. It automatically provides elasticity and on-demand power consumption. For that, this project presents a survey on high-throughput and ultra low-power asynchronous pipeline design method targeting to latch-free and extremely fine-grain design. Since they are asynchronous, these pipelines avoid problems related to high-speed clock distribution, such as clock power, clock skew, and rigidity in handling varied environments. The survey is mainly done on the data path logic. The data path may be single-rail, dual-rail or combination of the both logic. Asynchronous pipeline based on constructed critical data-path is combination of both the data path. Critical path compose of dual- rail logic and noncritical enables single-rail logic. Based on this critical data path, the handshake circuits are simplified, which offers the pipeline low power consumption as well as high throughput by reducing the overhead problems. This design is going to be implemented by TANNER EDA simulations model.

INTEX ITEMS: *Asynchronous pipeline, Critical datapath, Domino logic*

I. INTRODUCTION

High performance energy efficient logic style is one of the famous research topics in the field of VLSI circuits because of the continuous demands of power, speed and area constraints. As transistor integration increases, new advances are established in the concerned design strategies and related tools are used to design the VLSI circuits, which results cost effective and low power high performance VLSI circuits.

Low-power VLSI circuit design is a dynamic research area driven by the growing reliance on battery-powered portable computing and wireless communication products. In addition, it has become critical to the continued progress of high-performance and reliable microelectronics system. The earliest and still the most urgent demands for low power electronics originate from the requirements for the small size and weight, long operating life, utility and reliability of battery operated equipment such as wrist watches, pocket calculators and cellular phones, hearing aids. Implantable cardiac pacemakers and a myriad of portable military equipment used by individual foot soldiers. Perhaps no segment of the electronics industry has a growth potential as explosive as that of the personal digital assistant (PDA) which has been characterized as a combined pocket cellular phone, pager, e-mail terminal, fax, computer, address directory, etc.

The VLSI low power design problems can be broadly classified into two: analysis and optimization. Analysis problems are concerned about the accurate estimation of the power or

energy dissipation at different phases of the design process. The purpose is to increase confidence of the design with the assurance that the power consumption specifications are not violated. Evidently, an analysis technique differs in their accuracy and efficiency. The accuracy of analysis depends on the availability of the design information. In early design phases, the emphasis to obtain power dissipation estimates rapidly with very little available information on the design. In these phases, less analysis results are expected and tolerated. As the design proceeds to reveal more lower- level details, a more accurate analysis can be performed. Here, better accuracy is demanded and longer analysis time is allowed. Analysis techniques also serve design, given an optimization goal, without violating design specifications. An automatic design optimization algorithm requires a fast analysis engine to evaluate the merits of the design choices. Manual optimization also demands a reliable analysis tool to provide accurate estimation of power dissipation.

A decision to apply a particular low power technique often involves trade-offs from different sources pulling in various directions. Major criteria to be considered are the impact to the circuit delay, which affects the performance and throughput of the chip, and the chip area, which directly translates to manufacturing costs. Other factors of chip design such as design cycle time, testability, quality, reliability, reusability, risk, etc., may all be affected by a particular design decision to achieve low power requirement. Power efficiency cannot be achieved without yielding to one or more of these factors. The task of a design engineer is to carefully weigh each design choice within the specification constraints and select the best implementation.

a)*Pipelining*

Pipelining is generally used to achieve high performances digital system design. It is classified in to two types as synchronous and asynchronous pipelining. In synchronous system, is a straight forward technique which is used to increase parallelism and hence boost system throughput. Synchronous pipelining design consists of complex functional blocks which are subdivided into smaller blocks; registers are inserted to separate functional blocks. The global clock is applied to all registers. But in asynchronous pipeline design, local clock is applied to all registers. This pipeline design send data from left to right and an acknowledge control signal is send from right to left that is bidirectional communication, which is implemented by handshaking protocol. Asynchronous pipeline is classified based on the data path logic as static and dynamic logic. Each class uses different approaches for control and data storage. Four important features in providing design flexibility and modularity for asynchronous design are as follow. First, in synchronous systems, all stages operate at the same fixed rate, and the worst-case stage delay must be less than the clock period. In Asynchronous system, all stages need not have equal delay. Due to dynamically varying delay asynchronous adder have data dependent problem. This should be avoided to improve average system latency and throughput. Second, asynchronous pipelines provide elasticity. Input data items arrive in irregular manner hence the spacing and throughput rate are determined dynamically. Third, asynchronous pipelines provide automatic flow control whereas synchronous pipeline have no flow control. Finally switching activity occurs only when data items are being processed, so dynamic power consumed only on demand. Thus asynchronous

pipelining is more efficient than synchronous

phase dual rail protocol for handshaking and dual rail or single rail data path logic for valid data passage.

B) Problems With Synchronous Approach

The synchronous approach predominated, largely because it is easier to design chips in which things happen only when the clock ticks. As chips get bigger, faster and more complicated, distributing the clock signal around the chip becomes harder. Another drawback with clocked designs is that they waste a lot of energy, since even inactive parts of the chip have to respond to every clock tick. Clocked chips also produce electromagnetic emissions at their clock frequency, which can cause radio interference. Each tick must be long enough for **signals to traverse even a chip's longest wires in** one cycle. However, the tasks performed on parts of a chip that are close together finish well **before a cycle but can't move on until the next** tick. As chips get bigger and more complex, it becomes more difficult for ticks to reach all elements, particularly as clocks get faster.

In today's chips, the clock remains the key part of the action. As a microprocessor performs a given operation, electronic signals travel along microscopic strips of metal forking, intersecting again, and encountering logic gates-until they finally deposit the results of the computation in a temporary memory bank called a register. Let's say you want to multiply 4 by 6. If you could slow down the chip and peek into the register as this calculation was being completed, you might see the value changing many times, say, from 4 to 12 to 8, before finally settling down into the correct answer. That's because the signals transmitted to perform the operation travel along many different paths before arriving at the register; only after all pipelining. Asynchronous pipelining uses four signals have completed their journey is the correct value assured. The role of the clock is to guarantee that the answer will be ready at a given time. The chip is designed so that even the slowest path through the circuit-the path with the longest wires and the most gates-is guaranteed to reach the register within a single clock-tick. The **chip's clock is an oscillating crystal that vibrates** at a regular frequency, depending on the voltage applied. This frequency is measured in gigahertz or **megahertz. All the chip's work is** synchronized via the clock, which sends its signals out along all circuits and controls the registers, the data flow, and the order in which the processor performs the necessary tasks. An advantage of synchronous chips is that the order **in which signals arrive doesn't** matter.

Signals can arrive at different times, but the register waits until the next clock tick before capturing them. As long as they all arrive before the next tick, the system can process them in the **proper order. Designers thus don't have to worry** about related issues, such as wire lengths, when working on chips. And it is easier to determine the maximum performance of a clocked system. With these systems, calculating performance simply involves counting the number of clock cycles needed to complete an operation

C) Asynchronous Logic Circuits

As its name suggests, it does away with the cardinal rule of chip design: that everything marches to the beat of an oscillating crystal **"clock". For a 1GHz chip, this clock ticks one billion times a second, and all of the chip's** proces**sing unit's co**-ordinate their actions with these ticks to ensure that they remain in step. **Asynchronous, or "clockless", designs, in** contrast, allow different bits of a chip to work at different speeds, sending data to and from each other as and when appropriate.

International Journal of Research
Available at https://edupediapublications.org/journals
p-ISSN: 2348-6848
e-ISSN: 2348-795X
Volume 03 Issue 11
July2016

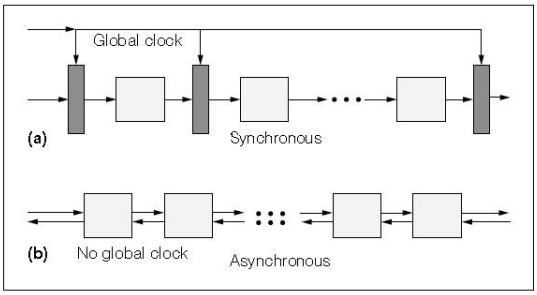

Figure 1. An abstracted view of synchronous (a) versus asynchronous (b) pipelines

D) How Clock-Less Chip Works

There are no purely asynchronous chips **yet. Instead, today's clockless processors are** actually clocked processors with asynchronous elements. Clockless elements use perfect clock gating, in which circuits operate only when they have work to do, not whenever a clock ticks. Instead of clock-based synchronization, local handshaking controls the passing of data between logic modules. The asynchronous processor places the location of the stored data it wants to read onto the address bus and issues a request for the information. The memory reads the address off the bus, finds the information, and places it on the data bus. The memory then acknowledges that it has read the data. Finally, the processor grabs the information from the data bus.

a) Simple, efficient design

Logic modules could be developed without regard to compatibility with a central clock frequency, which makes the design process easier. Also, because asynchronous processors **don't need specially designed modules that all** work at the same clock frequency, they can use standard components. This enables simpler, faster design and assembly. However, the recent use of both domino logic and the delay-insensitive mode in asynchronous processors has created a fast approach known as integrated pipelines mode.

Domino logic improves performance because a system can evaluate several lines of data at a time in one cycle, as opposed to the typical approach of handing one line in each cycle. Domino logic is also efficient because it acts only on data that has changed during processing, rather than acting on all data throughout the process. The delay-insensitive mode allows an **arbitrary time delay for logic blocks. "Registers** communicate at their fastest common speed. If one block is slow, the blocks that it communicates with slow down. This gives a system time to handle and validate data before passing it along, thereby reducing errors.

b) Different styles

There are several styles of asynchronous design. Conventional chips represent the zeroes and ones of binary digits using low and high voltages on a particular wire. One clock-less **approach, called "dual rail", uses two wires for** each bit. Sudden voltage changes on one of the wires represent a zero, and on the other wire a one. "Dual-rail" circuits use two wires giving the chip communications pathways, not only to send bits, but also to send "handshake" signals to indicate when work has been completed. Replacing the conventional system of digital logic with what he calls "null convention logic," a scheme that identifies not only "yes" and "no," but also "no answer yet" a convenient way for clock-less chips to recognize when an operation has not yet been completed. Another approach is **called "bundled data". Low and high voltages on** 32 wires are used to represent 32 bits, and a change in voltage on a 33rd wire indicates when the values on the other 32 wires are to be used.

c) Asynchronous for Low Noise and Low Emission

Sub circuits of a system may interact in unintended and often subtle ways. For example, a digital sub circuit generates voltage noise on the power-supply lines or induces currents in the silicon substrate. This noise may affect the performance of an analog-to-digital converter connected so as to draw power from the same source or that is integrated on the same substrate. Another example is that of a digital sub circuit that emits electromagnetic radiation at its clock frequency , and a radio receiver sub-circuit that mistakes this radiation for a radio Due to the absence of a clock, asynchronous circuits may have better noise and EMC (Electro-Magnetic Compatibility) properties than synchronous circuits. This advantage can be appreciated by analyzing the supply current of a clocked circuit in both the time and frequency domains signal.

Circuit activity of a clocked circuit is usually maximal shortly after the productive clock edge. It gradually fades away and the circuit must become totally quiescent before the next productive clock edge. Viewed differently, the clock signal modulates the supply current as depicted schematically in Figure. Due to parasitic resistance and inductance in the on-chip and off-chip supply wiring this causes noise on the on-chip power and ground lines.

In this paper, our proposed pipeline reduces both the dual-rail encoding overhead in data paths and the detection overhead in hand- shake control logic by designing based on a constructed critical data path. A stable critical data path is constructed using redesigned dual-rail domino gates. By detecting the stable critical data path, a 1-bit completion detector is enough to get the correct handshake signal regardless of the data path width. Such design does not only greatly reduce the detection overhead but also partially maintains the good properties in the four-phase dual-rail protocol design. Moreover, the stable critical data path serves as a matching delay to solve the dual-rail encoding overhead problem in data paths. With the help of the redesigned dual-rail domino gates, single-rail domino logic is successfully applied in noncritical data paths. As a result, the asynchronous domino logic pipeline has a small overhead in both handshake control logic and function block logic, which greatly improves the circuit efficiency.

This paper is organized as follows. Section II introduces the background of asynchronous domino logic pipeline. PS0 is introduced to demonstrate the advantages and problems of asynchronous domino logic pipeline based on dual-rail protocol. Several related designs are also simply introduced. Synchronizing logic gates (SLGs) and synchronizing logic gates with a latch function (SLGLs) are introduced to construct a stable critical data path. The robustness of the pipeline structure and the constructed critical data path is analyzed. Then, more complex pipeline structures are further discussed. Section III focuses on comparison of different parameters of the pipeline structures. Section IV presents the conclusion.

ASYNCHRONOUS PIPELINE BASED ON CRITICAL DATAPATH

The pipeline is designed based on a stable critical data path that is constructed using 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 pre-charge ports of their previous stages. The difference is that a total done signal is generated by detecting only the critical data path instead of the entire data paths. 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

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 pre-charge ports of their previous stages. The difference is that a total done signal is generated by detecting only the critical data path instead of the entire data paths. 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. APCDP is more familiar to bundled-data asynchronous pipeline because the critical data path essentially works as a matching delay, which controls the correct data transfer in noncritical data paths. Compared with conventional bundled -data design using a separate matching delay to match the worst case delay in function blocks, the proposed design method reuses the existing function block logic to provide the matching delay. Such design has many advantages. First, the matching delay is accurate. The matching delay in APCDP is exactly the same worst case delay in function blocks. Second, the matching delay is robust for delay variations. Dual-rail critical data path supplies delay-insensitive property, which can be self-adaptive to delay variations in function blocks. Third, the handshake control logic is efficient in area and power. The handshake control logic is implemented by reusing the existing function block logic.

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. For example, the ripple carry adder .The ripple carry path seems to be the stable critical data path. However, actually, the critical data path varies according to different input data patterns. Because of the gate-delay data-dependence problem, the carry function gate can be triggered early by the input bits (a_n and b_n) 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 method introduces an efficient solution that uses SLGs to construct the critical data path. The SLGs solve the gate-delay data-dependence problem by making sure that SLGs cannot start evaluation until all valid data arrive. 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.

1) Asynchronous data encoding

To encode data, the single control req wire is replaced by a data field. Two common approaches, each for four-phase handshaking. Dual-rail codes and delay insensitive encoding In dual-rail encoding, each bit is implemented by a pair of wires, or rails. In the reset phase, both rails are low, forming a spacer, which indicates no valid data. During the evaluate phase, the sender asserts exactly one of the two rails high, thereby indicating the actual value of the bit (0 or 1) as well as data validity. The receiver typically uses a completion detector to identify that a valid codeword has been received. The remainder of the four-phase protocol proceeds as

International Journal of Research
Available at https://edupediapublications.org/journals
p-ISSN: 2348-6848
e-ISSN: 2348-795X
Volume 03 Issue 11
July2016

expected: after the dual-rail data is transmitted, ack is asserted high by the receiver,
the dual-rail data is reset to zero, and ack is de asserted low by the receiver. Such an encoding is one instance of a delay-insensitive (DI) code in which each codeword uniquely identifies its own validitity.

2) Structure of Asynchronous Pipeline based on Constructed Critical Data Path

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 pre-charge/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. Therefore, single-rail domino gates are used in the noncritical data path to save logic overhead. Encoding converter is used to bridge the connection between single rail domino logic and dual rail domino logic.

3) 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 = $a\_t \cdot b\_t$ and the false side by out_f =$a\_f + b\_f$ .. In traditional dual-rail domino AND gate, there are three transistor paths: 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 both ON, which leads to a much quicker signal transfer. As a result, the gate delay has a large variation depending on different data patterns

4) Synchronizing Logic Gates

SLGs are dual-rail domino gates that have no gate-delay data-dependence problem. 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 out_$f = a_t \cdot b_f + a_f \cdot (b_t + b_f)$.1) [$a\_t$, b_t]; 2) [a_t, b_f]; 3) [a_f, b_t]; and [a_f, b_f]. Every path has two transistors at the sequential position, and there is only one path turns ON corresponding to an input data pattern. As a result, the gate delay becomes independent on different data patterns. 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.

5) Synchronizing Logic Gates With a Latch Function

Based on the characteristics of SLGs, SLGLs are extended synchronizing AND gate with a latch function and the table of latch states. An SLGL has an enable port (en_t, en_f), which controls the opaque and transparent state of the SLGL. The principle is that SLGLs cannot start evaluation without the presence of

6) Power Analysis

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. Each energy consumption is an average value calculated from 100 cycles. The results show that APCDP consumes much less energy than LP2/2-SR. The adoption of single-rail domino

gates in APCDP reduces the toggle rate in data paths since single-rail domino logic does not toggle when transferring low-voltage signal. Besides, the toggling rate relates to the injected data patterns Therefore, the energy consumption

of APCDP varies a lot according to different data patterns. . On the other hand, the energy consumption of LP2/2-SR remains almost constant. LP2/2-**SR's full dual**-rail domino data paths constant toggling rate regardless of the injected data patterns.

7) Robustness Analysis

APCDP has pipeline failure in the situation that a pipeline stage does not finish evaluating before its previous stage start precharge. In such situation, the pipeline stage cannot correctly finish evaluating because the precharge of its previous pipeline stage removes the valid data from the inputs. To avoid this pipeline failure, APCDP needs to satisfy an assumption that, in a pipeline stage,none of the other bits across the entire data paths is slower than the detected bit by more than the delay through a static NOR gate and the drive buffer chain following it. The robustness of APCDP is analyzed based on this assumption.

CONCLUSION

This paper introduced 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. The evaluation result shows that the proposed design has better performance than a bundled-data asynchronous domino logic pipeline.

REFERENCES

[1] B. H. Calhoun, Y. Cao, X. Li, K. Mai, L. T. Pileggi, and R. A. **Rutenbar, "Digital circuit** design challenges and opportunities in the era of **nanoscale CMOS," Proc. IEEE,** vol. 96, no. 2, pp. 343–365,Feb. 2008.

[2] J. Sparsø and S. Furber, Principles of Asynchronous Circuit Design: A Systems Perspective. Boston, MA, USA:
Kluwer, 2001.

[3] M. Krstic, E. Grass, F. K. Gurkaynak, and **P.Vivet, "Globally asynchronous, locally** synchronous circuits: Overview and outlook," IEEE Des.Test Comput., vol. 24, no. 5, pp. 430–441, Sep. /Oct. 2007.

[4] A. J. Martin and M. Nystrom, **"Asynchronous techniques** for system on-chip **design," Proc. IEEE,** vol. 94, no. 6, pp. 1089–1120, Jun. 2006.

[5] J. Teifel and R. Manohar, "An asynchronous **dataflow FPGA architecture," IEEE Trans.** Comput., vol. 53, no. 11, pp. 1376–1392, Nov. 2004.

[6] H. S. Low, D. Shang, F. Xia, and A. **Yakovlev, "Variation tolerant AFPGA architecture," in Proc. ASYNC, 2011, pp. 77–**86.

[7] M. Hariyama, S. Ishihara, and M. **Kameyama, "Evaluation of a field** programmable VLSI based on an asynchronous **bitserial architecture," IEICE Trans. Electron.,** vol. E91-C, no. 9, pp. 1419–1426, Sep. 2008.

[8] T. E. **Williams, "Self**-timed rings and their application to divisio**n," Ph.D. dissertation,** Dept. Comput. Sci., Stanford Univ., Stanford, CA, USA, Jun. 1991.

[9] A. **M. Lines, "Pipelined asynchronous circuits," Dept. Comput. Sci., California Inst.** Technol., Pasadena, CA, USA, Tech. Rep., 1998.