



## POWER OPTIMIZED MULTI BIT FLIPFLOP USING HIGH THROUGHPUT MODIFIED CLOCK GATING TECHNIQUE

1. Y SUGANDHI NAIDU, 2.I RANI CHINNARI

1. Assistant Professor, Dept. Of ECE, Aditya College of Engineering, Surampalem, AP

2. M. Tech, Dept. Of ECE, Aditya College of Engineering, Surampalem, AP

### ABSTRACT:

This project introduces a set of techniques that, considering the dynamic streaming behavior of algorithms, can achieve power savings by selectively switching off parts of the circuits when they are temporarily inactive. The techniques being independent from the semantic of the application can be applied to any application and can be integrated into the synthesis stage of a high-level dataflow design flow. This concept describes an approach for developing energy optimized run-time reconfigurable designs which benefit from clock gating. Field Programmable Gate Arrays FPGAs are highly desirable for implementation of digital systems due to their flexibility, programmability and low end product life cycle. The goal is to reduce the power consumption without sacrificing much performance or incurring a large chip area so that the territories of FPGAs applications can expand more effectively. Reducing the power of FPGAs is the key to lowering packaging and cooling costs, improving device reliability, and opening the door to new markets such as mobile electronics. Further, this project is enhanced by using modified clock gating technique with c elements and double edge triggered flipflops to further reduce power consumption.

### INTRODUCTION:

The sharp rise in integrated circuit fabrication costs and the need for fast time-to-market and for in-field upgrades have accounted for the increasing popularity of reconfigurable devices such as Field Programmable Gate Arrays (FPGAs). The flexibility of reconfigurable devices, however, comes with overheads in latency, area and power consumption—for instance it has been shown [9] that the dynamic power consumption for FPGAs can be up to 14 times worse than that for Application-Specific Integrated Circuits (ASICs). Two methods for reducing power consumption for reconfigurable devices have been proposed. The first method involves clock gating; disabling the clock for the inactive regions in a design to minimise signal transitions and hence dynamic power [8, 17]. The second method involves reconfiguring an FPGA with multiple bit-streams, one for each configuration, so that a small device with low power consumption can be used [1]. However, there are delay and energy overheads for reconfiguration with bit-streams. The

aim of this paper is to present a novel approach for reconfiguring designs. The approach involves two run-time reconfiguration techniques: multiplexer-based reconfiguration, and reconfigurable word-length optimization. Clock-gated reconfiguration with physical multiplexers is faster than reconfiguration by adopting a new FPGA bit-stream, but it does not provide the area advantage offered by bit-stream reconfiguration. We also derive the conditions under which the proposed approach involving clock-gated reconfiguration would require less energy than bit-stream reconfiguration. While much work [14, 21] in this field reports that clock gating reduces power consumption, there is at least one paper [5] which reports the contrary. Moreover, most results from previous work are obtained by vendor's simulators such as XPower. In contrast, all the power consumption results in this paper are obtained by measuring the current and voltage of various applications Field-Programmable Gate Arrays (FPGAs) are integrated circuits (ICs) that can be

Copyright @ 2019 ijearst. All rights reserved.

INTERNATIONAL JOURNAL OF ENGINEERING IN ADVANCED RESEARCH  
SCIENCE AND TECHNOLOGY

Volume.01, IssueNo.03, July -2019, Pages: 254-261

programmed to implement any digital circuit. The main difference between FPGAs and conventional fixed logic implementations, such as Application Specific Integrated Circuits (ASICs), is that the designer can program the FPGA on-site. Moreover, using an FPGA instead of a fixed logic implementation eliminates the non-recurring engineering (NRE) costs and significantly reduces time-to-market. Hence, FPGAs are highly desirable for implementation of digital systems due to their flexibility, programmability and low end product life cycle and all this makes them ideal for prototyping, debugging, and for small to medium volume applications. FPGAs are slower and less efficient than fixed implementation, due to the added circuitry that is needed to make them very flexible. In FPGA, programmable switches controlled by configuration memory occupy a large area and add a significant amount of parasitic capacitance and resistance to the logic and routing resources. As a result, FPGAs are approximately 3 times slower, 20 times larger, and 12 times less power efficient compared to ASICs [1].

#### **LITERATURE SURVEY:**

Zhang et al. [21] analyse the effect of clock gating on power efficiency showing that FPGAs, although not as efficient as ASICs, can achieve significant power reductions. Clock gating may not always be the most energy-efficient solution even if it is the most power-efficient solution in some cases. Cadenas et al. [5] implement a clock gating technique in a pipelined Cordic core with the goal of reducing bit-switching. They do not obtain power improvements using a Cordic pipelined design. We explore optimizations both with clock gating and bit-stream reconfiguration, and use word-length analysis techniques to improve results. In some cases we set part of the input to zero to reduce dynamic power consumption if clock gating has little effect, to reduce the area overhead. An alternative to run-time bit-stream reconfiguration is based on using multiplexers and demultiplexers for time multiplexing designs [6, 11, 20]. This method supports fast reconfiguration but requires a large area. However, both this approach and the clock gating approach require configurations known at design time because they need to be installed on the chip at the start, while bit-stream reconfiguration can be used when downloading new configurations. In control-flow analysis, Styles and Luk use information about branch

frequencies to reduce the hardware used for implementing branches that are infrequently taken [18]. Since program executions often change their behaviour based on input data, the circuit needs to evolve at run time in order to keep the error to a minimum. Bondalapati and Prasanna reconfigure the circuit at run time and show that it can be used to reduce execution time by up to 37% [4]. Our approach in Globally asynchronous locally synchronous (GALS)-based systems consist of several locally synchronous components which communicate with each other asynchronously. Works on GALS can be separated into three categories: 1) partitioning; 2) communication devices; and 3) dedicated architectures. Dataflow design modeling, exploration, and optimization for GALS-based designs has been studied previously by several authors. Suhaib *et al.* [8] proposed a design and evaluation framework for modeling application-specific GALS-based dataflow architectures for cyclostatic applications, where system performance, e.g., throughput, is taken into account during optimization. Similarly, Wu and Vrudhula [9] and Ghavami and Pedram [10] proposed a method for automatic synthesis of asynchronous digital systems. These two approaches were developed for fine-grained dataflow graphs, where actors are primitives or combinational functions. Related to this paper, Brunet *et al.* [11] proposed a multiple clock, domain-design methodology for reducing the power consumption of dataflow programs. Their design objective was to optimize the mapping of an application while still meeting design performance requirements. This optimization was achieved by assigning each clock domain an optimized clock frequency to reduce power consumption. Power gating techniques on FPGA-based platforms have been investigated by academic and industrial researchers [7-10]. A lookup table-level, gate-level fine-grain and unused logic blocks power gating techniques are proposed in [7], [8] and [9], respectively. After all, the internal structure of the an FPGA could be changed by the manufacturer based on these approaches. A system level power gating technique for Xilinx Zynq SoC is presented by authors [10], investigating the overhead of the technique. Utilising this work, our approach in this paper explains when and how we can apply the FPGA power gating on streaming applications. An unused block RAM power gating technique is

presented by Xilinx in 28nm 7-series devices [11] in which only block RAMs are utilised by a design consume power. Independently controllable power domains are supported in Xilinx Zynq-7000 [5] and Zynq UltraScale+ MPSoC [12] which makes them suitable for system level power gating techniques. In this paper, we utilise this feature in the Zynq-7000 SoC to reduce the energy consumption. The problem of designing the processor and minimizing the power dissipation by various power minimization techniques is addressed by many authors and a brief overview of their work is mentioned below. Brunelli Claudio et al.[9] presented a VHDL model and implementation of a coarsegrain reconfigurable coprocessor for a RISC core. Julien Lamoureux et al

**CLOCK GATING:** Clock gating is a popular technique used in many synchronous circuits for reducing dynamic power dissipation. Clock gating saves power by adding more logic to a circuit to prune the clock tree. Pruning the clock disables portions of the circuitry so that the flip-flops in them do not have to switch states. Switching states consumes power. When not being switched, the switching power consumption goes to zero, and only leakage currents are incurred. Clock gating works by taking the enable conditions attached to registers, and uses them to gate the clocks. A design must contain these enable conditions in order to use and benefit from clock gating. This clock gating process can also save significant die area as well as power, since it removes large numbers of muxes and replaces them with clock gating logic. This clock gating logic is generally in the form of "Integrated clock gating" (ICG) cells. However, note that the clock gating logic will change the clock tree structure, since the clock gating logic will sit in the clock tree.

Clock gating logic can be added into a design in a variety of ways:

1. Coded into the RTL code as enable conditions that can be automatically translated into clock gating logic by synthesis tools (fine grain clock gating).
2. Inserted into the design manually by the RTL designers (typically as module level clock gating) by instantiating library specific ICG (Integrated Clock Gating) cells to gate the clocks of specific modules or registers.

3. Semi-automatically inserted into the RTL by automated clock gating tools. These tools either insert ICG cells into the RTL, or add enable conditions into the RTL code. These typically also offer sequential clock gating optimisations.

Note: Any RTL modifications to improve clock gating will result in functional changes to the design (since the registers will now hold different values) which need to be verified.

Sequential clock gating is the process of extracting/propagating the enable conditions to the upstream/downstream sequential elements, so that additional registers can be clock gated.

Although asynchronous circuits by definition do not have a "clock", the term **perfect clock gating** is used to illustrate how various clock gating techniques are simply approximations of the data-dependent behavior exhibited by asynchronous circuitry. As the granularity on which you gate the clock of a synchronous circuit approaches zero, the power consumption of that circuit approaches that of an asynchronous circuit: the circuit only generates logic transitions when it is actively computing.<sup>[2]</sup> Chip families such as OMAP3, with a cell phone heritage, support several forms of clock gating. At one end is the manual gating of clocks by software, where a driver enables or disables the various clocks used by a given idle controller. On the other end is automatic clock gating, where the hardware can be told to detect whether there's any work to do, and turn off a given clock if it is not needed. These forms interact with each other and may be part of the same enable tree. For example, an internal bridge or bus might use automatic gating so that it is gated off until the CPU or a DMA engine needs to use it, while several of the peripherals on that bus might be permanently gated off if they are unused on that board. With the decrease of feature sizes and increase of clock frequencies in integrated digital circuits, power consumption has become a major concern for modern integrated circuit designs. Power dissipation has a dynamic component, due to the switching of active devices, and a static component, due to the leakage of inactive devices. Since our work targets dynamic power only, further references to "power" in this we will imply the dynamic power. Clock gating is one of the most effective and widely

used techniques for saving clock power. The clock net is one of the nets with the highest switching density, resulting in high power dissipation in the adders. A promising technique to reduce the power dissipation of the clock net is selectively stopping the clock in parts of the circuit, called "clock gating". It is very well integrated into semi-custom design flows nowadays. By gating the clock, the switching activity of the adders clock signal is reduced. However, clock gating circuitry itself occupies chip area and consumes additional power; therefore a judicious selection of circuit

**MEMORY ORGANIZATION:**

This section describes PJMEDIA's implementation of delay buffer. Delay buffer works quite similarly like a fixed jitter buffer, that is it will delay the frame retrieval by some interval so that caller will get continuous frame from the buffer. This can be useful when the operations are not evenly interleaved, for example when caller performs burst of put() operations and then followed by burst of operations. With using this delay buffer, the buffer will put the burst frames into a buffer so that get() operations will always get a frame from the buffer (assuming that the number of get() and put() are matched).

The buffer is adaptive, that is it continuously learns the optimal delay to be applied to the audio flow at run-time. Once the optimal delay has been learned, the delay buffer will apply this delay to the audio flow, expanding or shrinking the audio samples as necessary when the actual audio samples in the buffer are too low or too high.

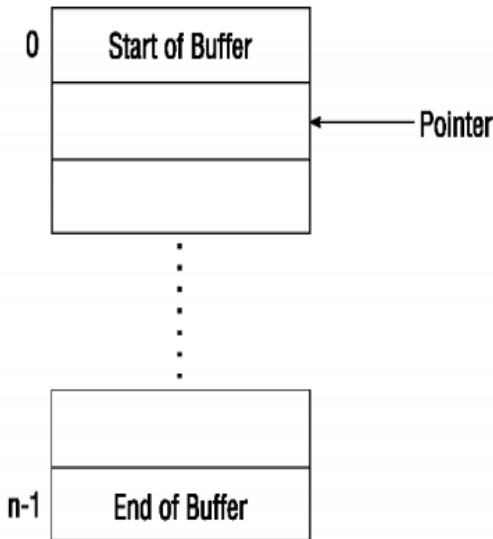


Fig : Buffer

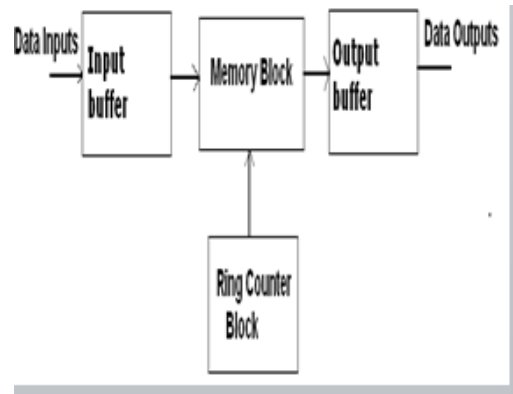


Fig : Existing Block Of Memory Organisation

**INPUT BUFFER:**

The Input buffer is also commonly known as the input area or input block. When referring to computer memory, the input buffer is a location that holds all incoming information before it continues to the CPU for processing. Input buffer can be also used to describe various other hardware or software buffers used to store information before it is processed. Some scanners (such as those which support "include" files) require reading from several input streams. As flex scanners do a large amount of buffering, one cannot control where the next input will be read from by simply writing a YY\_INPUT() which is sensitive to the scanning context. YY\_ () is only called when the scanner reaches the end of its buffer, which may be a long time after scanning a statement such as an include statement which requires switching the input source.

**MEMORY BLOCK:**

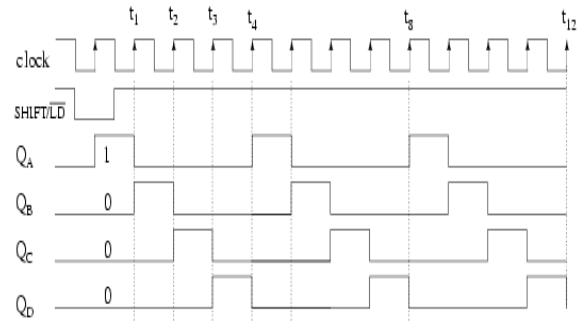
(RAM) Random-access memory (RAM) is a form of computer data storage. Today, it takes the form of integrated circuits that allow stored data to be accessed in any order (that is, at random). "Random" refers to the idea that any piece of data can be returned in a constant time, regardless of its physical location and whether it is related to the previous piece of data. The word "RAM" is often associated with volatile types of memory (such as DRAM memory modules), where the information is lost after the power is switched off. Many other types of memory are RAM as well, including most types of ROM and a type of flash memory called NOR-Flash.

Scan design has been the backbone of design for testability (DFT) in industry for about three decades because scan-based design can successfully obtain

controllability and observability for flip-flops. Serial Scan design has dominated the test architecture because it is convenient to build. However, the serial scan design causes unnecessary switching activity during testing which induce unnecessarily enormous power dissipation. The test time also increases dramatically with the continuously increasing number of flip-flops in large sequential circuits even using multiple scan chain architecture. An alternate to serial scan architecture is Random Access Scan (RAS). In RAS, flip-flops work as addressable memory elements in the test mode which is a similar fashion as random access memory (RAM). This approach reduces the time of setting and observing the flip-flop states but requires a large overhead both in gates and test pins. Despite of these drawbacks, the RAS was paid attention by many researchers in these years. This paper takes a view of recently published papers on RAS and rejuvenates the random access scan as a DFT method that simultaneously address three limitations of the traditional serial scan namely, test data volume, test application time, and test power.

**RING COUNTER:** A ring counter is a type of counter composed of a circular shift register. The output of the last shift register is fed to the input of the first register.

There are two types of ring counters: A *straight ring counter* or *Overbeck counter* connects the output of the last shift register to the input of the first shift register and circulates a single one (or zero) bit around the ring. For example, in a 4-register one-hot counter, with initial register values of 1000, the repeating pattern is: 1000, 0100, 0010, 0001, 1000... . Note that one of the registers must be pre-loaded with a 1 (or 0) in order to operate properly. Loading binary 1000 into the ringcounter, above, prior to shifting yields a viewable pattern. The data pattern for a single stage repeats every four clock pulses in our 4-stage example. The waveforms for all four stages look the same, except for the one clock time delay from one stage to the next. See figure below.



Load 1000 into 4-stage ring counter and shift

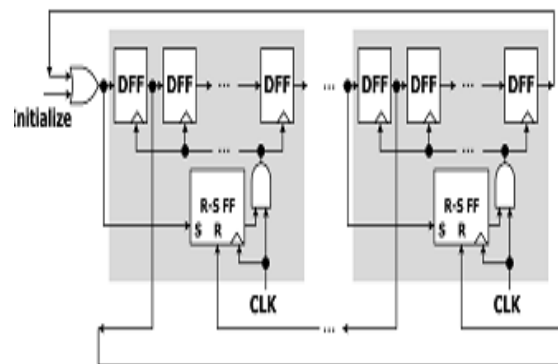


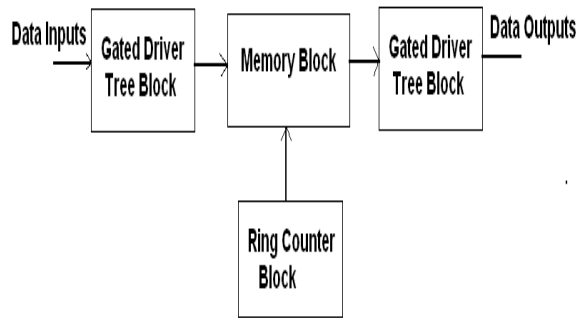
Fig : Ring Counter With SR Flip-Flops

The above block diagram shows the power controlled Ring counter. First, total block is divided into two blocks. Each block is having one SR FLIPFLOP controller

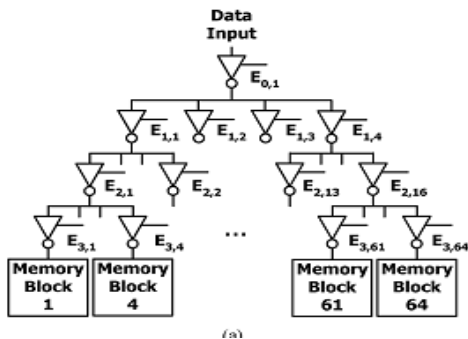
S	R	Q(t-1)	Q
0	0	0	0
0	0	1	1
0	1	1	0
0	1	1	0
1	0	1	1
1	0	1	1
1	1	0	X
1	1	1	X

Table : SR Flip Flop Truth Table

**PROPOSED TECHNIQUE:**



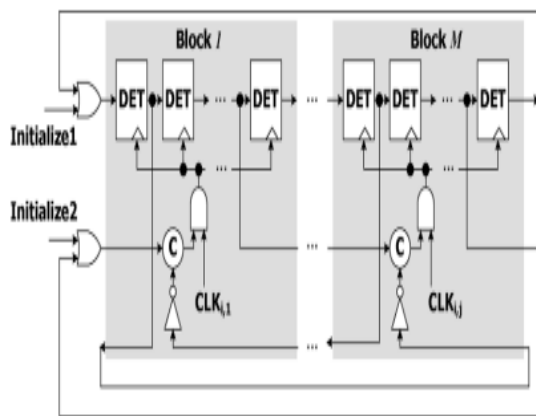
**Fig: Block Diagram For Proposed Delay Buffer**  
**GATED DRIVER TREE:**



**Fig : Gated Driver Tree**

Gated driver tree derived from the same clock gating signals of the blocks that they drive. Thus, in a quad-tree clock distribution network, the “gate” signal of the gate driver at the level (CKE) should be asserted when the active DET flip-flop

**MODIFIED RING COUNTER:**

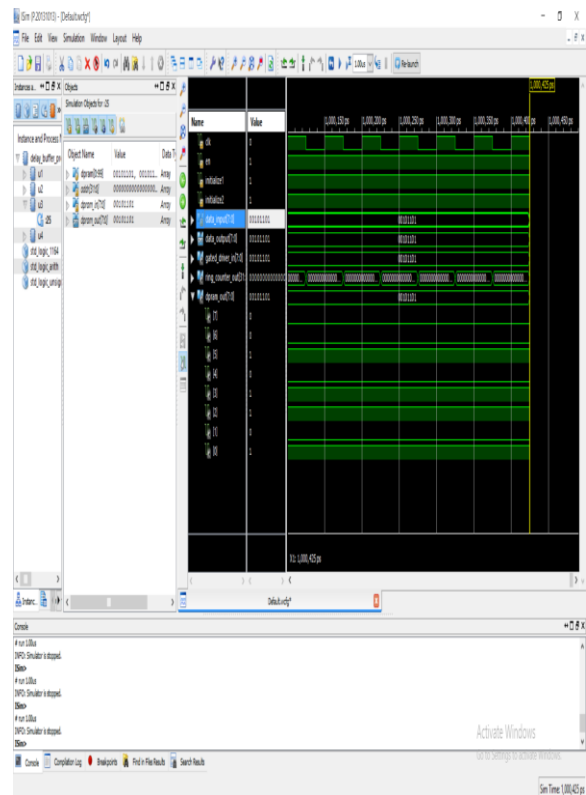


**Fig: Modified Ring Counter**

**DET (Double edge triggered flip-flops:**

Double-edge-triggered (DET) flip-flops are utilized to reduce the operating frequency by half. The logic construction of a double-edge-triggered (DET) flip-flop, which can receive input signal at two levels the clock, is analyzed and a new circuit design of CMOS DET. In this paper, we propose to use double-edge-triggered (DET) flip-flops instead of traditional DFFs in the ring counter to halve the operating clock frequency. Double edge-triggered flipflops are becoming a popular technique for low-power designs since they effectively enable a halving of the clock frequency. The paper by Hossain et al [1] showed that while a single-edge triggered flipflop can be implemented by two transparent latches in series, a double edge-triggered flipflop can be implemented by two transparent latches in parallel; the circuit in Fig. 1 was given for the static flipflop implementation. The clock signal is assumed to be inverted locally. In high noise or low-voltage environments, Hossain et al noted that the p-type pass-transistors may be replaced by n-types or that all pass-transistors may be replaced by transmission gates

**RESULT:**



## CONCLUSION:

Leakage power and a steady increase in dynamic power with each successive process generation. Field programmable gate arrays (FPGAs) require considerable hardware overhead to offer programmability, making them less power-efficient than custom ASICs for implementing a given logic circuit. The huge numbers of transistors on the largest FPGA chips suggest that the power trends associated with scaling may impact FPGAs more severely than custom ASICs. Finally, this concept designed a CG methodology applied to dataflow designs that can be automatically included in the synthesis stage of an HLS design flow. The application of the power saving technique is independent from the semantic of application and does not need any additional step or effort during the “design” of the application at the dataflow program level. The CG logic is generated during the synthesis stage together with the synthesis of the computational kernels connected via FIFO queues constituting the dataflow network. Conceivably, these techniques could be extended to other dataflow methods of computation.

## REFERENCES:

- [1] Shih-Jung Hsu Rung-Bin Lin —Clock Gating Optimization with Delay-Matching Design, Automation & Test in Europe Conference & Exhibition (DATE), 2011, PP-1-6, 14-18, March 2011
- [2] Hao Xu, Ranga Vemuri, and Wen-Ben Jone —Dynamic Characteristics of Power Gating During Mode Transition Very Large Scale Integration (VLSI) Systems, IEEE Transactions on , VOL. 19, NO. 2, pp-237 - 249, February 2011
- [3] Ting-Hao Lin, Chung-Yang (Ric) Huang Using SAT-Based Craig Interpolation to Enlarge Clock Gating Functions DAC'11 San Diego, California, USA ACM 2011
- [4] Baosheng Wang, Jayalakshmi Rajaraman, Kanwaldeep Sobti, Derrick Losli, and Leff Rearick —Structural Tests of Slave Clock Gating in Low-power Flip-flop VLSI Test Symposium (VTS), 2011 IEEE 29th, pp- 254 - 259, 1-5 May 2011
- [5] Li Li, Ken Choi, Haiqing Nan —Effective Algorithm for Integrating Clock Gating and Power Gating to Reduce Dynamic and Active Leakage Power Simultaneously Quality Electronic Design (ISQED),

2011 12th International Symposium on , pp- PP1 - 6, 2011.

[6] Shmuel Wimer and Israel Koren, —The Optimal Fan-Out of Clock Network for Power Minimization by Adaptive Gating Very Large Scale Integration (VLSI) Systems, IEEE Transactions on , No. 99 , pp-1 - 9, 2011

[7] Frederic Rivoallon — Reducing Switching Power with Intelligent Clock Gating WP370 (v1.3) March 1, 2011

[8] Jatin N. Mistry, Bashir M. Al-Hashimi, David Flynn and Stephen Hill —Sub-Clock Power-Gating Technique for Minimising Leakage Power during Active Model Design, Automation & Test in Europe Conference & Exhibition, pp- 1 - 6, 2011

[9] Hiroki Matsutani, Michihiro Koibuchi, Daisuke Ikebuchi, Kimiyoshi Usami, Hiroshi Nakamura, and Hideharu Amano, —Performance, Area, and Power valuations of Ultrafine-Grained Run- Time PowerGating Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on, VOL. 30, NO. 4, pp-520 - 533, APRIL 2011 [10] Weixiang Shen, Yici Cai, Xianlong Hong, and Jiang Hu, —An Effective Gated Clock Tree Design Based on Activity and Register Aware Placement Very Large Scale Integration (VLSI) Systems, IEEE Transactions on , VOL. 18, NO. 12, pp-1639 - 1648 , DECEMBER 2010

[11] Shih-Hsu Huang, Chia-Ming Chang, Wen-Pin Tu, Song-Bin Pan —Critical-PMOS- Aware Clock Tree Design Methodology For AntiAging Zero Skew Clock Gating Design Automation Conference (ASP-DAC), 2010 15th Asia and South Pacific , Page(s): 480 - 485, 2010