International Journal of Research
Available at https://edupediapublications.org/journals
e-ISSN: 2348-6848
p-ISSN: 2348-795X
Volume 04 Issue14
November 2017

Power Efficient Implementation of Streaming Applications using low power Clock-Gating method on FPGAs

Kalvala Srikanth
Assistant professor, Dept of ECE, Sree Dattha Group of Institutions, Sheriguda, Rangareddy,Telangana, India.
srikalvala@gmail.com

Abstract: *This paper presents the reduction of dynamic power for streaming applications yielded by asynchronous dataflow designs by using clock gating techniques. Streaming applications constitute a very broad class of computing algorithms in areas such as signal processing, digital media coding, cryptography, video analytics, network routing, packet processing, etc. Clock gating is a power-saving feature in semiconductor microelectronics that enables switching off circuits. This paper introduces clock gating 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. Experimental results show that power reduction is achieved with no loss in data throughput.*

Index Terms—Clock-gating, dataflow, high-level synthesis.

I.INTRODUCTION

Power dissipation is currently the major limitation of silicon computing devices. Reducing power has also other beneficial effects, it implies less stringent needs for cooling, improved longevity, longer autonomy in the case of battery operated devices and obviously, lower power costs. For all these reasons power also frequently affects the choice of the computing platform right at the outset. For example, field-programmable gate arrays (FPGAs) imply higher power dissipation per logic unit when compared to equivalent application-specific integrated circuit (ASIC), but often compare favorably to conventional processors used for the same functional tasks.

For any silicon device, power dissipation can be partitioned into two components:

1) A static and
2) A dynamic component.

Static power dissipation also referred to as quiescent or standby power consumption, is the result of the leakage current of the transistors, also affected by the ambient temperature. By contrast, dynamic power dissipation is caused by transistors being switched and by losses of charges being moved along wires. Power dissipation increases linearly with frequency, largely due to the influence of parasitic capacitances. To counteract this effect, ASIC designers have employed clock gating (CG) techniques in the last 20 years [1]–[3].

Different strategies for optimizing power consumption on ASICs and FPGAs. Clock gating is a power-saving feature in semiconductor microelectronics that enables switching off circuits. Many electronic devices use clock gating to turn off buses, controllers, bridges and parts of processors, to reduce dynamic power consumption. These papers describe the impact of a chosen technology for a given architecture, but do not describe how to reduce power at the design abstraction level. As a consequence, adding power controllers at the behavioral description design stage constitutes an additional task that has to be carried-out with care to avoid introducing undesired application behaviors and might reduce the portability of the code (i.e., platform is changed during the development process). 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. Dynamic dataflow [5]–[7] designs such as for instance the ones expressible using the formal RVC-CAL language possess interesting properties that can be exploited for reducing the power consumption without affecting, by construction, the behavioral characteristics of the application. In RVC-CAL, every actor can concurrently execute processing tasks, executions might be disabled by input blocking reads, and every communications among actors can occur only by means of order preserving lossless queues. As a consequence, an actor may be stopped for a certain period if its processing tasks are idle or its outputs queues (buffers) are full without impacting the overall throughput and semantical behavior of the design.

II.LITERATURE REVIEW

High-level synthesis of dynamic dataflow programs on heterogeneous MPSoC platforms

The growing complexity of digital signal processing applications make a compelling case the use of high-level design and synthesis methodologies for the implementation on programmable logic devices and embedded processors. Past research has shown that, for complex systems, raising the level of abstraction of design stages does not necessarily come at a penalty in terms of performance or resource requirements. Dataflow programs provide behavioral descriptions capable of expressing both sequential and parallel components of application algorithms and enable natural design abstractions, modularity, and portability. In this paper, an open source tool, implementing dataflow programs onto embedded heterogeneous platforms by means of high-level synthesis, software synthesis and interface synthesis is presented Experimental design results demonstrate the capability and the effectiveness of the tool for implementing a wide range of applications when combined with Vivado HLS.

Comparing models of computation

We give a denotation framework (a "meta model") within which certain properties of models of computation can be compared. It describes concurrent processes in general terms as sets of possible behaviors. A process is determinate if, given the constraints imposed by the inputs, there are exactly one or exactly zero behaviors. Compositions of processes are processes with behaviors in the intersection of the behaviors of the component processes. The interaction between processes is through signals, which are collections of events. Each event is a value-tag pair, where the tags can come from a partially ordered or totally ordered set. Timed models are where the set of tags is totally ordered. Synchronous events share the same tag, and synchronous signals contain events with the same set of tags. Synchronous processes have only synchronous signals as behaviors. Strict causality (in timed tag systems) and continuity (in untimed tag systems) ensure determinacy under certain technical conditions. The framework is used to compare certain essential features of various models of computation, including Kahn process networks, dataflow, sequential processes, concurrent sequential processes with rendezvous, Petri nets, and discrete-event systems.

Clock-gating and its application to low power design of sequential circuits

This paper models the clock behavior in a sequential circuit by a quaternary variable and uses this representation to propose and analyze two clock-gating techniques. It then uses the covering relationship between the triggering transition of the clock and the active cycles of various flip flops to generate a derived clock for each flip flop in the circuit. A technique for clock gating is also presented, which generates a derived clock synchronous with the master clock. Design examples using gated clocks are provided next. Experimental results show that these designs have ideal logic functionality with lower power dissipation compared to traditional designs.

III.PROPOSED ARCHITECTURES

Current FPGA families support different CG strategies and each manufacturer provides its own IP for managing these different approaches. The methodology described here is based on using primitives specific to Xilinx FPGA architectures. It is briefly described how CG techniques are implemented on Xilinx FPGAs and how an automatic CG strategy within Xronos HLS is realized.

A. Profile Guided Buffer Size

The execution of a dataflow program consists of a sequence of action firings. These firings can be correlated to each other in a graph-based representation using an approach called execution trace graphing (ETG). The graph is an acyclic directed graph where each node represents an action firing, and a directed arc represents either a data or a control dependency between two different action firings. The effectiveness of analyzing a dataflow program using an ETG is demonstrated in [12]. Xronos provides profiling for each firing execution in clock cycles. This is achieved by retrieving the difference of DONE and GO signals for each action firing during register-transfer level simulation [13]. Timing information is added to the ETG for each firing and each dependency arises according to a corresponding time value, thus transforming the ETG into a weighted graph. A close-to-optimal buffer size configuration, in terms of execution throughput and buffer memory utilization, can be obtained through an iterative analysis of the algorithmic critical path evaluated using the weighted ETG. For a detailed description, the interested reader can refer to [14].

B. Coarse-Grained Clock Gating Strategy

When the output buffer of any actor is full, the clock of this actor should be turned off as the actor is idle. This is because switching off its clock will not have an impact on design throughput. Even though RVC-CAL dataflow designs are used for the behavioral description, such CG strategy is more general and can be applied to systems that represent the execution of a process that communicates with asynchronous FIFO buffers. The queues should be asynchronous for lossless communication when an actor is clock gated and a design has differing input clock domains.

This strategy consists of adding a clock enabler circuit for activating the actors' clock. This circuit contains: a controller for each output port queue of each actor, a combinatorial logic for the configuration of the output ports, and a clock buffer (which enables the clock). A representation of an actor with a single output port being clock gated is illustrated in Fig. 1. As depicted, queues are asynchronous. Queues have two input clocks: one for consuming tokens and one for producing them. Additionally, queues have two output ports:

1) AF for almost full and
2) F for full.

The actors input clock is connected to the output of the clock enabler circuit. Finally, the clock buffer BUFGCE input clock should be connected with a flip-flop for glitch-free CG [15]. The flip-flop will introduce a one-clock latency when the clock is switched off, but this additional clock cycle will not have an impact on actors that are on the critical path. Those actors are not being clock gated because the TURNUS dimensioning of the FIFO queues is based on critical path analysis. Hence, this approach does not impact overall performance.

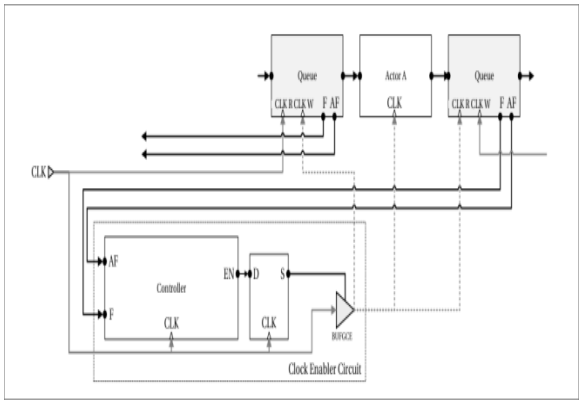

Fig. 1: CG methodology applied for actor A.

The actor A has two outputs one of those have a fanout of two. The clock enabling circuit takes the Almost Full and Full signal of each queue and a

clock from a clock domain and as a result it is going to activate or deactivate the clock of actor A.

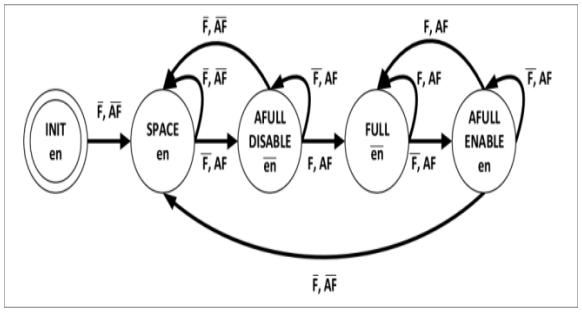

Fig. 2. State machine of the clock enabling controller.

The controller has two inputs, F for full, AF for almost full and one output en as the enable signal.

1) Clock Enabling Controller:

The clock enabling controller is represented in Fig. 2. The controller is implemented as a finite state machine (FSM) having a clock; a reset; input F, for full; input AF, for almost full; and output EN, for enable. The AF input becomes active high when there is only one space left on its FIFO Queue. Its FSM has five states S = {INIT, SPACE, AFULL_DISABLE, FULL, AFULL_ENABLE}. The controller starts with the INIT state and maintains the EN output port at active high until F and AF become active low. The active high EN is maintained during the SPACE state. As a queue becomes full, the state changes to AFULL_DISABLE. In this state, the EN output passes to an active low. A conservative approach is taken in this state as the BUFGCE disables the output clock on the high-to-low edge. The clock enables entering the BUFGCE should be synchronized to the input clock. Once the queue becomes full, the controller maintains the EN at active low. When a token is consumed from the queue, the controller passes to the AFULL_ENABLE state, and it activates the clock. Then, depending on whether the buffer becomes full or almost full, the state changes to either the FULL or the SPACE state.

2) Strategy:

The user can choose a mapping configuration that indicates which actor should be clock gated. To do so, an attribute is given to each actor. If an actor has been selected for CG, all of its output FIFO queues, A and AF, are connected to a clock enabler controller. Output queues can be connected through a fanout or directly to a queue. In the first case, the controller results are connected to an AND logic port. This is a safe approach in the case that one of the queues in the fanout is full. In this case, the fanout should command the actor not to produce a token. For the latter case, if an actor's output is connected directly to a queue without a fanout, the result should be connected to an OR logic port as the next actor may need to consume a certain number of tokens to output a token. This may lead the system to lock due to the unavailability of data. In the third case, if there is a combination of outputs with or without a fanout, then an n-input OR logic port is inserted. Fig. 3 depicts these configurations. A pseudo-template of the clock Enabler circuit is given in Template 1. This template generates a Verilog file that takes into account the different cases described previously. These situations are detected and generated automatically as described in the "always clause."

International Journal of Research
Available at https://edupediapublications.org/journals
e-ISSN: 2348-6848
p-ISSN: 2348-795X
Volume 04 Issue14
November 2017

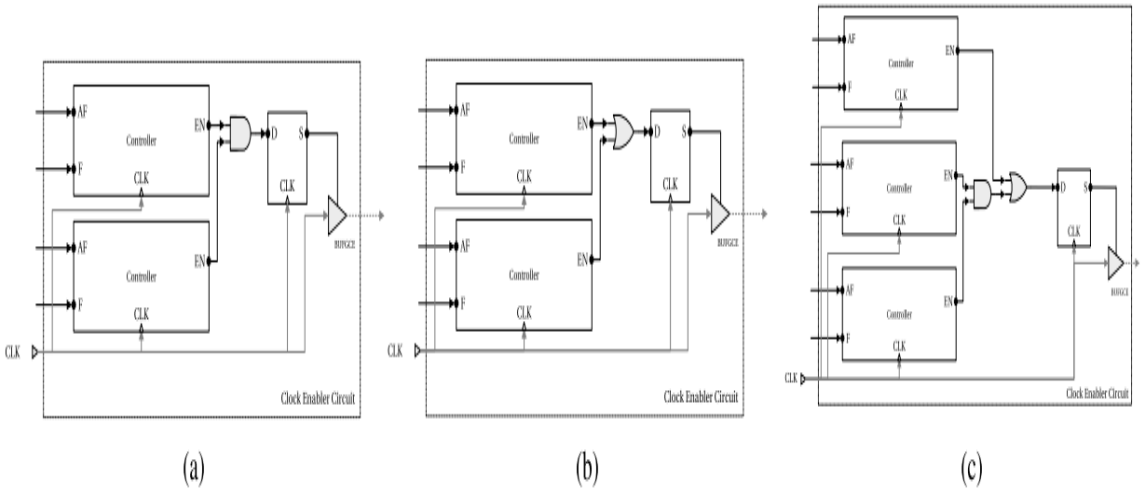

Fig. 3. Clock enabler circuit in three different configurations. (a) Single output port with a fanout. (b) Two different output ports. (c) Single output port with a fanout and another ouput port.

Template 1: Clock Enabler Circuit Module Creation

```
module clock_enabler
Input  : actor
Input  : enable
Input  : clk_in
Input  : reset
Input  : ∀P^out_almost_full
Input  : ∀P^out_full
Output : clk_out
always p in ∀P^out being
  wire ["sizeof(p.fanout)":0] "nameof(p)"_enable;
reg clock_enable;
wire buf_enable;
always p in ∀P^out being
  always idx in sizeof(p.fanout) being
    controller c_"nameof(p)"_"idx"(
    .almost_full("nameof(p)"_almost_full["idx"])
    .full("port.name"_full["idx"]),
    .enable("port.name"_enable["idx"]),
    .clk(clk),
    .reset(reset));
always @(posedge clk) being
  clock_enable <= always p ∀P^out SEPARATOR "−" being
    if sizeof(p.fanout) > 1 then
      always idx in sizeof(p.fanout) SEPARATOR "&" being
        nameof(p)_enable["idx"]
    else
      nameof(p)_enable
assign buf_enable = en ? clock_enable : 1;
BUFGCE clock_enabling (.I(clk), .CE(buf_enable), .O(clk_out));
endmodule
```

A flip-flop (created by the always clause) is connected between the BUFGCE and the final OR or AND port. Thus, clock glitches are eliminated and the clock enabling is runt free. The last output of the CG is a new clock that is connected to the actors, its fanouts, and its queues' write and read clocks (CLK W and CLK R, respectively) as visualized in Fig. 1.

IV.RESULTS

Experimental Results are shown for one of the applications is the intra MPEG-4 simple profile decoder. Due to restrictions on the number of clock buffers in Xilinx FPGAs, the design selected was refactored to result in 32 actors.

Design Summary:

Logic Utilization	Non Clock Gated	Clock Gated	Available
Slices	9214	12776	607200
LUTs	21499	25126	303600
BUFGCTRLs	1	32	32
BRAMs	7	7	1030
DSPs	18	18	2850
Max Freq.	109	109	-

Power Saving Efficiency Over Decoder Throttling

As described in Table I the maximum decoder throughput rate is 350 frames/s for a QCIF image (176×144 pixels). For the experiment, the decoder is throttled such that it decodes only 30 images per second for two resolutions QCIF and CIF (384×288 pixels). Fig. 4(b) reports the power consumption and the activation rate for each actor's clock (found on the intra MPEG-4 simple profile decoder). The activation rate of the actor's clock demonstrates that some of them have an activation rate of less than 10%.

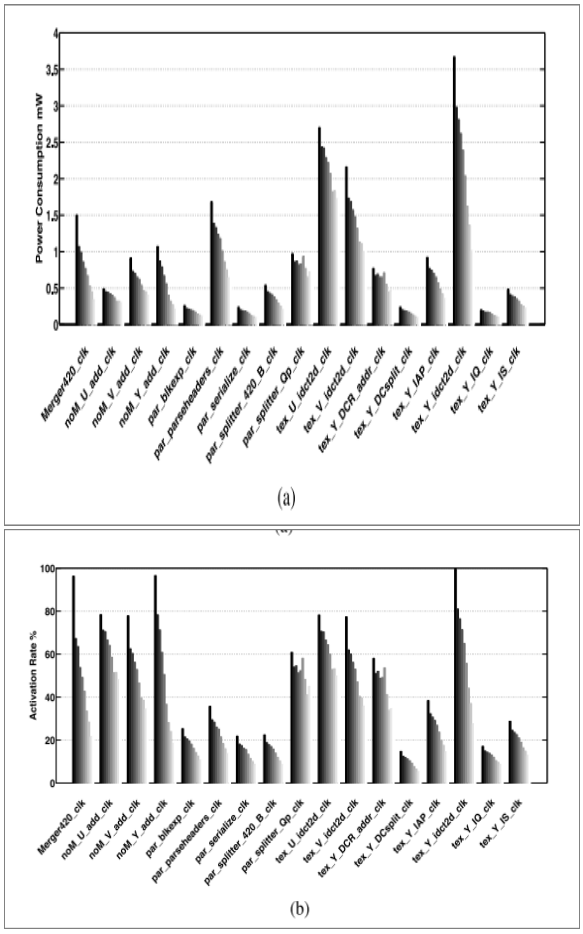

Fig. 4: Power consumption and activation rate of each clock gated actor clock of the MPEG-4 SP decoder. Median values were retrieved from an MPEG-4 reference QCIF input stimuli (video sequence). (a) Actors clock power consumption. (b) Actors clock activation rate.

V.CONCLUSION

This paper presents a CG methodology applied to dataflow designs that can be automatically included in the synthesis stage of an HLS design flow. 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. Experimental results show that savings in power dissipation achieved with a slight increase in control logic without any reduction in throughput have been achieved. Unsurprisingly, CG is attractive in situations where the design is not used to its full capacity. As a result, this technique is particularly interesting in applications with dynamically varying performance requirements, when designing to a particular performance point is impossible, and when power consumption is deemed costly. Further investigations into CG should consider more aggressive control logic, whereby control is given to each individual actor, allowing greater flexibility to actor inactivity.

REFERENCES

[1] M. Pedram, "Power minimization in IC design: Principles and applications," ACM Trans. Design Autom. Electron. Syst., vol. 1, no. 1, pp. 3–56, Jan. 1996. [Online]. Available: http://doi.acm.org/10.1145/225871.225877

[2] Q. Wu, M. Pedram, and X. Wu, "Clock-gating and its application to low power design of sequential circuits," IEEE Trans. Circuits Syst. I, Fundam. Theory Appl., vol. 47, no. 3, pp. 415–420, Mar. 2000.

[3] G. E. Tellez, A. Farrahi, and M. Sarrafzadeh, "Activity-driven clock design for low power circuits," in IEEE/ACM Int. Conf. Comput.-Aided Design Dig. Tech. Papers (ICCAD), San Jose, CA, USA, Nov. 1995, pp. 62–65.

[4] E. A. Lee and A. Sangiovanni-Vincentelli, "Comparing models of computation," in Proc. IEEE/ACM Int. Conf. Comput.-Aided Design, Austin, TX, USA, 1997, pp. 234–241.

[5] G. Kahn, "The semantics of simple language for parallel programming," in Proc. IFIP Congr., Stockholm, Sweden, 1974, pp. 471–475.

[6] E. A. Lee and D. G. Messerschmitt, "Static scheduling of synchronous data flow programs for digital signal processing," IEEE Trans. Comput., vol. 36, no. 1, pp. 24–35, Jan. 1987.

[7] E. A. Lee and T. M. Parks, "Dataflow process networks," Proc. IEEE, vol. 83, no. 5, pp. 773–801, May 1995.

[8] S. Suhaib, D. Mathaikutty, and S. Shukla, "Dataflow architectures for GALS," Electron. Notes Theor. Comput. Sci., vol. 200, no. 1, pp. 33–50, 2008.

[9] T.-Y. Wuu and S. B. K. Vrudhula, "Synthesis of asynchronous systems from data flow specification," Inf. Sci. Inst., Univ. Southern California, Los Angeles, CA, USA, Tech. Rep. ISI/RR-93-366, Dec. 1993.

[10] B. Ghavami and H. Pedram, "High performance asynchronous design flow using a novel static performance analysis method," Comput. Elect. Eng., vol. 35, no. 6, pp. 920–941, Nov. 2009.

[11] S. C. Brunet et al., "Partitioning and optimization of high level stream applications for multi clock domain architectures," in Proc. IEEE Workshop Signal Process. Syst. (SiPS), Taipei, Taiwan, Oct. 2013, pp. 177–182.

[12] S. Casale-Brunet, "Analysis and optimization of dynamic dataflow programs," Ph.D. dissertation, STI Elect. Eng., STI, Lausanne, Switzerland, 2015.

[13] E. Bezati, "High-level synthesis of dataflow programs for heterogeneous platforms," Ph.D. dissertation, STI, Lausanne, Switzerland, 2015.

[14] S. Casale-Brunet, M. Mattavelli, and J. W. Janneck, "Buffer optimization based on critical path analysis of a dataflow program design," in Proc. IEEE Int. Symp. Circuits Syst. (ISCAS), Beijing, China, May 2013, pp. 1384–1387.

[15] Analysis of Power Savings from Intelligent Clock Gating, Xilinx, San Jose, CA, USA, Aug. 2012.

[16] (2014). Open RVC-CAL Applications. Accessed on Feb. 25, 2014. [Online]. Available: http://github.com/orcc/orc-apps

[17] M. Canale, S. Casale-Brunet, E. Bezati, M. Mattavelli, and J. Janneck, "Dataflow programs analysis and optimization using model predictive control techniques," J. Signal Process. Syst., vol. 84, no. 3, pp. 371–381, 2015. [Online]. Available: http://dx.doi.org/10.1007/s11265-015-1083-4.

Author's Profile:

KALVALA SRIKANTH is currently working as an assistant professor in ECE department in Sree Dattha Group of Institutions, Sheriguda. He received his Master's degrees in Embedded system and VLSI system design from Jawaharlal Nehru Technological University, Hyderabad.He received his Bachelor's degree in Electronics engineering from Nagpur University. His current research interests include very large scale integration (VLSI) low power design, test automation and fault-tolerant computing.
Email Id: srikalvala@gmail.com