International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 07 Issue: 01 | Jan 2020 www.irjet.net p-ISSN: 2395-0072

CLOCK GATING OF STREAMING APPLICATIONS FOR POWER MINIMIZATION ON FPGA'S

M. Venkata Raghudeep[1], K. Anji Babu[2]

[1]Student, Dept of ECE, A.K.R.G College of Engineering and Technology, JNTU Kakinada, AP, India
[2]Assistant Professor, Dept of ECE, A.K.R.G College of Engineering and Technology, JNTU Kakinada, AP, India

---

Abstract - This project investigates 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. 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. Experimental results of at-size applications synthesized on field-programmable gate arrays platforms demonstrate power reductions achievable with no loss in data throughput.

Key Words: Clock gating, Streaming applications, Power saving, Field programmable gate arrays, Data flow.

1. INTRODUCTION

In the past, the major concerns of the VLSI designer were area, performance, cost and reliability; power considerations were mostly of only secondary importance. In recent years, however, this has begun to change and, increasingly, power is being given comparable weight to area and speed. Several factors have contributed to this trend. Perhaps the primary driving factor has been the remarkable success and growth of the class of personal computing devices (portable desktops, audio- and video-based multimedia products) and wireless communications systems which demand high-speed computation and complex functionality with low power consumption.

1.1 Streaming Applications

Many applications in embedded systems perform the same or similar operations on regular sequences of data, which can be categorized as streaming applications. For example, on a smart-phone, one can easily find many streaming applications that are essential to the functionality of the system, like wireless communication, high-definition video/audio codes and 3D graphics rendering. A streaming application usually contains a set of independent processing stages. The structure of a H.264 video codec, which is a typical streaming application. The different stages or the codec form a pipeline that is used to encode or decode videos frames. A frame in a H.264 stream can be an I-frame, a P-frame or a B-frame.

The streaming processing structure in these applications creates lots of optimization opportunities, such as pipelined parallel execution of different parts, and exploiting data level parallelism in certain stages. More importantly, streaming applications are among the most performance demanding applications in embedded systems. The rapid introduction of these applications becomes an important driving force for the development of new embedded technologies. So this thesis focuses on developing energy efficient architectures and compilation techniques for streaming applications.

1.2 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 is a methodology of turning off the clock for a particular block when it is not needed and is used by most SoC designs today as an effective technique to save dynamic power.

The simplest and most common form of clock gating is when a logical "AND" function is used to selectively disable the clock to individual blocks by a control signal, as illustrated in Figure.1 During synthesis, the tools identify groups of FFs which share a common enable control signal and use them to selectively switch off the clocks to those groups of flops. Both of these clock-gating methods will eventually introduce physical gates in the clock paths which control their downstream clocks. These gates could introduce clock skew and lead to setup and hold-time violations even when mapped into the SoC. However, this is compensated for by the clock-tree synthesis and layout tools at various stages of the SoC back-end flow.

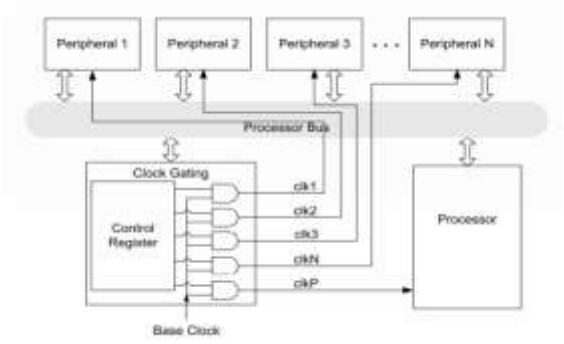

Fig -1: Block Diagram of Clock Gating

2. PROPOSED SYSTEM

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. However, this methodology can be modified to support other FPGA vendors primitives.

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.

3. IMPLEMENTATION

Algorithm:

1) Step1: Initialize en=1.when F=0,AF=0;
2) Step2: Now queue has empty space. Wait with en=1 until AF=1.
   If AF=1 go to next state.
3) Step3: Now CG-ckt ready to give en=0. If AF=0 again go to step2.
   Else if AF=1, F=0 be in the same state.
   Else F=1, AF=1 go to next state.
4) Step4: Now queue is in full state so, disable clk input to actor by asserting en=0;
5) Step5: Wait until F=0;

   If AF=1, F=1 be in the same state.

   Else F=0, AF=1 go to next state;

6) Step6: Now queue is in almost full state en=1;
7) If F=0,AF=0 queue has empty space then goto step2;
8) Elseif
   F=1,
   AF=1
   again goto step5.

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.

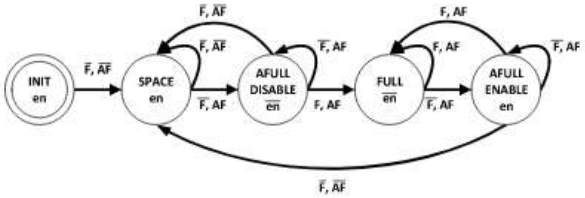

Fig -2: State machine of the clock enabling controller.

The power reduction gain of the aforse mentioned methodology is evaluated by applying it to a video decoder design. In the reader can find a variety of RVC-CAL applications for dataflow programs. One of these 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 re factored to result in 32 actors.

The intra MPEG-4 SP description contains 32 actors and it is 4:2:0 decoder which is separated into eight processing blocks.

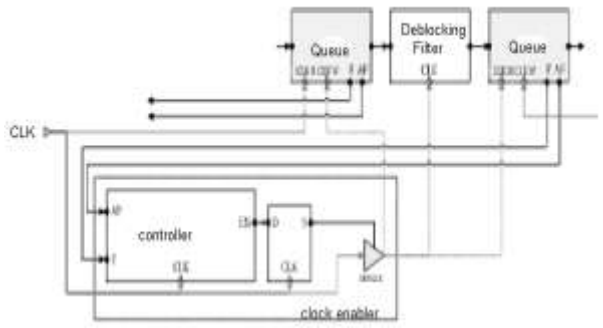

Fig -3: streaming application element (De-blocking filter) with clock gating.

4. RESULTS:

Table -1: comparison table of proposed and extension method

PROJECT	SLICE LUTS	SLICE REGISTERS	LUT-FFS	DELAY
PROPOSED	368	171	97	8.775ns
EXTENSION	268	155	81	8.775ns

4. CONCLUSION

This project presents 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 schematic 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. Experimental results are very encouraging: 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. In these circumstances CG is a simple, automatic, and effective way to recover power otherwise lost in "idle" cycles. 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. Furthermore, it will be necessary to develop tools that partition complex applications onto the limited number of clock domains for more efficient implementations. Lastly, additional considerations could be given to controlling clock speed and, possibly, voltage transitions.

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 Theory. 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.