ISCAS 2000 - IEEE International Symposium on Circuits and Systems, May 28-31, 2000, Geneva, Switzerland

CONTROLLER REDESIGN BASED CLOCK AND REGISTER POWER MINIMIZATION

Srikanth Rao M. and S. K. Nandy
msrao@rishi.serc.iisc.ernet.in, nandy@serc.iisc.ernet.in
Supercomputer Education and Research Center
Indian Institute of Science, Bangalore, India 560012

ABSTRACT

Clock gating is an effective technique for minimizing dynamic power in sequential circuits. However, clock gating has some practical difficulties viz., possibility of glitches on the gated clock and in use of static timing analysis for verifying timing of the design. In this paper, we describe a robust scheme for power minimization that eliminates these difficulties of clock gating and yet provides nearly the same power savings. This scheme does not rely on propagation delays in the circuit for functioning, and is robust across process technologies. In this scheme, the controllers sequencing operations in a datapath are modified so that the control signals themselves are used as clocks for registers in the datapath. Since these "control clocks" typically operate at lower frequencies, power is saved in the registers and in the clock drivers. This scheme also preserves the cycle boundaries on which registers in the original circuit load data, thereby allowing reuse of test cases developed for the functional verification of the original circuit.

1. INTRODUCTION

Phenomenal advances in semiconductor process technologies have enabled higher integration and higher operational frequencies in present day VLSI designs. High operational frequencies increase power dissipation per unit area of the die, which affects packaging, cooling, and reliability of these designs. Market demand for portable devices powered by batteries has also influenced the need for power minimization in such systems. Power dissipation is therefore a key design parameter in present day VLSI designs [1][2].

Power dissipation in a circuit may be classified as static and dynamic [1][2]. Leakage and quiescent currents cause static power dissipation, whereas charging and discharging of parasitic capacitances and current between supply rails cause dynamic power dissipation. In many high frequency CMOS digital circuits, the dynamic power dominates over static power dissipated. The focus of most minimization techniques is therefore dynamic power. In sequential circuits, techniques for minimizing dynamic power are generally based on the following:

1. Precomputation and Guarded Evaluation [8][9], in which a function computed by a large combinational block is computed using a smaller block for certain inputs.
2. Clock gating [3][4], in which the clocks to flip-flops are disabled when the circuit is idle.
3. Retiming [7], in which the registers are used to reduce the depth of combinational logic.
4. State Assignment [5][6], in which the states of an FSM are encoded suitably to reduce power in the combinational logic used for generation of the next state.

In a sequential circuit, the sources for dynamic power dissipation are:

(a) Clock drivers: Minimization techniques based on (2) above can be applied here.
(b) Registers: Techniques based on (2) and (3) above can be applied here.
(c) Combinational blocks driven by the registers: Techniques (1), (3) and (4) can be applied here for power minimization.

Power dissipation in the clock drivers is significant in many sequential circuits as the clock lines are loaded heavily and operate at the highest frequency. Further, registers in synchronous circuits are clocked every cycle, even when idle, resulting in wasteful power dissipation. Figure 1a shows a single flip-flop that is a part of a register in which the multiplexer recirculates data when the register is idle. A commonly used technique for reducing wasteful power dissipation in registers and clock drivers is clock gating, in which the clock to registers is disabled when idle. Figure 1b illustrates this concept. If there are several flip-flops sharing the same gated clock, power is saved in the drivers of the CLOCK signal. Since the gated clock typically has a lower frequency of operation compared to the CLOCK, power dissipation inside the flip-flop is also reduced.

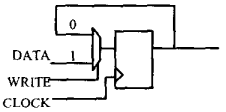


Figure 1a. Flip flop that is part of a register.

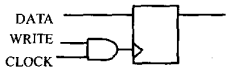


Figure 1b. A flip-flop with clock gating.

However, clock gating has some practical difficulties, viz.

(a) Possibility of glitches on the gated clock signal: Unless the CLOCK signal in the circuit corresponding to Figure 1b is delayed suitably with respect to the WRITE signal, the gated clock will glitch. This dependence on the propagation delays for correct operation complicates the migration of the design to newer process technologies.

(b) Difficulties in using static timing analysis effectively: In many circuits the gating signal is a complex function of inputs and internal signals. This results in non-deterministic delays on the gated clock, which complicates static timing analysis.

In this paper we describe a new technique named "Control Generated Clocking" (CGC) that overcomes these drawbacks of clock gating, and results in nearly the same power savings. This technique also preserves the cycle boundaries on which registers load data, allowing reuse of test cases for functional verification of the original circuit.

The organization of the rest of this paper is follows. In section 2 we provide an overview of the scheme. In section 3, we describe briefly static timing analysis and cost-benefit aspects of CGC. Results from application of this technique in the register file of a processor are discussed in section 4 and we finally conclude in section 5.

0-7803-5482-6/99/$10.00 ©2000 IEEE

2. CONTROL GENERATED CLOCKING (CGC)

Consider a general sequential circuit shown in figure 2, in which multiple state machines are shown, sequencing operations in the datapath. Control signals from the state machines enable the writes to registers in the datapath. The master clock MCLK synchronizes all activities in the datapath registers as well as in the controlling state machines. Power is wasted in the drivers of MCLK and in the registers, when writes to certain registers are infrequent.

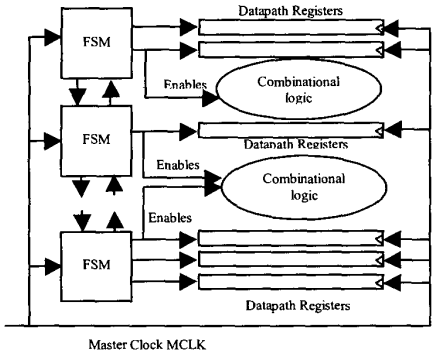


Figure 2. Schematic of a general sequential circuit.

Figure 3 shows the timing diagram for a write to a single flip-flop in a datapath register. The figure shows the control signal WRITE that is synchronous to the clock MCLK. The flip-flop loads the data on the next rising edge of clock MCLK. Observe that for a single write operation, this also happens to be the falling edge of the control signal. Our scheme essentially uses this control signal as the clock for the flip-flop. The flip-flop must now respond to the falling edge of the WRITE signal. Note that this simple arrangement works only when there are no back-to-back writes to the flip-flop. In such a scenario, the WRITE signal changes only at the end of the last write to the flip-flop. We alleviate this problem in the CGC scheme using RZ pulse generator circuits.

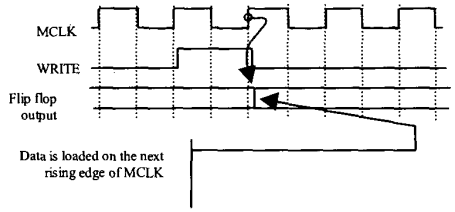


Figure 3. Timing diagram of a single flip-flop in a register loading data.

In CGC, the general system shown in figure 2 is transformed to a system shown in the figure 4. This scheme introduces several changes to the original circuit as listed below:

(a) **Changes to the datapath registers:** Positive edge triggered flip-flops in the datapath registers are replaced by negative edge triggered ones, because writes to registers in the original circuit complete on the falling edge of the control signals.

(b) **Changes to the clocks:** Clocks to datapath registers in the target logic are replaced by control generated clocks, indicated as CCLKs in the figure. The CCLKs are control signals that have been conditioned by the RZ pulse generators. The RZ pulse generator circuit is described in subsection 2.2

(c) **Changes to the controlling state machines:** In this step, the state machines are recoded in order to reduce the skew of each CCLK with respect to MCLK and to make each CCLK glitch free. This step is described in detail in subsection 2.1.

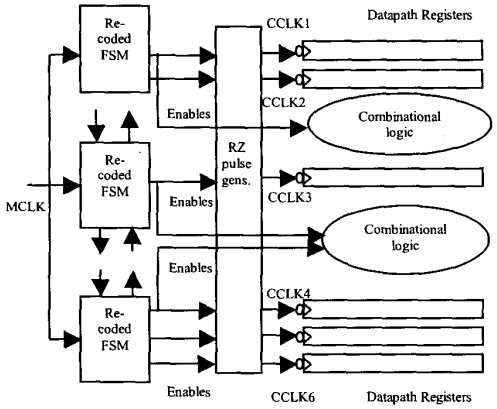


Figure 4. Sequential circuit after implementing CGC.

2.1 FSM recoding

Steps (1)-(3) listed below ensure that the control signals of interest are available as the output of flip-flops, which in turn ensures that the control signals are glitch free and have minimal skew with respect to the master clock.

1. Convert the given state machine into a Moore state machine to eliminate combinational paths from inputs to the outputs.
2. To each state that causes a write to atleast one register, assign a code with a one hot prefix. To each state that does not write to any register, assign a binary code.
3. For any register that can be written to by more than one state, insert shadow flip-flops and retime the control signal.

This procedure is described in the following example. Consider a state-machine with states $S_1, S_2, \ldots, S_{10}$. If S_1, S_2 and S_3 are the only states that cause a write to any register in the datapath, then the states are encoded as shown in Table 1.

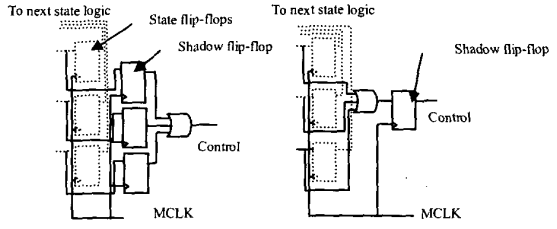
The assigned codes consist of a one hot prefix followed by a binary string. The one hot prefix is zeroed out for all states that do not write to a register. The binary string is zeroed out for states that write to at least one register. In the example, note that

S_1, S_2 and S_3 are assigned one hot prefixes, and the other states are encoded in binary.

Table 1. States and their encoding.

State	Code	State	Code
S_1	001 000	S_6	000 010
S_2	010 000	S_7	000 011
S_3	100 000	S_8	000 100
S_4	000 000	S_9	000 101
S_5	000 001	S_{10}	000 110

Step (3) is required if there is more than one state that can write to the same register. In the example, if states S_1, S_2 and S_3 write to the same register, then the control clock to that register is derived by performing the logical OR of outputs of state flip-flops S_1, S_2 and S_3. This causes the control clocks to glitch when the FSM transitions between two such states. This problem is solved using shadow flip-flops and a simple retiming operation. The shadow flip-flops mirror all states that write to a particular register. In order to do this, the shadow flip-flops receive the same data inputs as the state flip-flops. The retiming operation allows the shadow flip-flop to be shifted ahead of the OR gate. This makes the output glitch free, and suitable for use as a control clock. In the absence of shadow flip-flops, we need to OR the outputs of the state flip-flops. This step however precludes any retiming because the outputs of the state flip-flops are used for the generation of the next state. Figure 5a illustrates a portion of the FSM after inserting the shadow flip-flops. Note how the shadow flip-flops mirror those states that write to a particular register. Figure 5b illustrates the state machine after retiming. After this step, the controls are guaranteed to be glitch free.



Figures 5 (a). After insertion of shadow flip-flops. (b) After retiming the circuit.

2.2 RZ pulse generators

The RZ pulse generator takes the glitch free control signal, and the master clock signal MCLK as inputs. It produces an output signal that toggles once every cycle when the control signal is at logic '1' for more than one cycle at a time, when back-to-back writes occur to the corresponding register. Back to back writes are possible to a register if there is a path of length 1 between two states that write to the same register in the state transition diagram of the controller. Figure 6 shows the circuit for the RZ pulse generator and its timing diagram. The output of the RZ pulse generator is glitch free. The registers in the datapath load data on the falling edge of the control clocks output by this circuit.

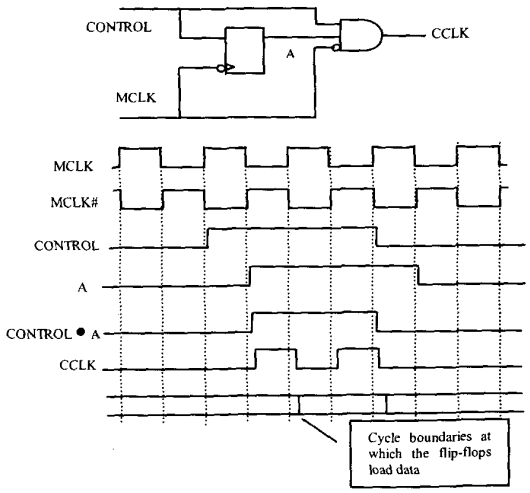


Figure 6. Return to Zero (RZ) pulse generator circuit and corresponding waveforms.

3. TIMING ANALYSIS AND COST-BENEFIT ASPECTS OF CGC

The maximum and minimum delay requirements of every timing path present in a design with CGC can be easily derived [10]. In sum, static timing analysis of a design with CGC is simpler compared to a design with clock gating. This is due to the fact that the registers are now clocked with signals that have a fixed propagation delay with respect to the master clock.

Analytical expressions for power savings in the registers and the clock drivers can be easily derived [10]. The power saved in registers is a function of the activity in the registers and the electrical characteristics of the flip-flops used to implement the registers. The power saved in the clock drivers is a function of the activity in the registers and the change in the clock net capacitance after implementing CGC. The number of additional flip-flops required to implement CGC has a linear relationship with the number of states that write to any datapath register.

4. RESULTS

In this section we discuss results obtained after implementing CGC in an experimental processor. The processor has an instruction set based on the DLX described in [11]. The processor has thirty-two 32-bit registers in its register file. The control unit of the processor has an FSM with six states. When the FSM is in the WRITEBACK state, the processor writes results of the instruction into the register file. The processor is coded in VHDL using RTL constructs. The design is flip-flop based and strictly synchronous, with all flip-flops clocked by the same clock signal. A gate level netlist was obtained using logic synthesis with a

commercial synthesis tool Design Compiler. The gate level netlist has a gate count equivalent to 28,800 two-input NAND gates. The target standard cell library has a minimum feature size of 0.18μ. The cells operate at 1.8 Volts under nominal conditions.

Several testcases were run on the gate level netlist within a VHDL simulation environment using a commercial simulator Modelsim. The switching activity information obtained during this stage was used to analyze the power consumed by different blocks in the processor using a commercial tool DesignPower. We observed that the clock drivers consumed 26% and the register file consumed 36% of the total dynamic power on an average. The register file consumes a large portion of the dynamic power because it is clocked in all states of the control FSM. The register file is written only when the FSM is in the WRITEBACK state. In all other states, the registers re-circulate data, dissipating power unnecessarily. The register file which has 1024 flip-flops also loads the clock line heavily, and causes significant power dissipation in the clock drivers.

We implemented the CGC scheme for the register file in RTL. The entire register file was clocked only during the WRITEBACK state, using a control signal from the FSM controller. Logic synthesis was again used to obtain a gate level netlist of the circuit with CGC. In the new netlist, the timing improved marginally and the gate count was lower. The same test cases were run on the new netlist, and power analysis was performed as before. Implementation of CGC resulted in 65% reduction in the register file power and 62% reduction in clock power. The overall power reduced by 35% with CGC.

To compare CGC against clock gating, we implemented another register file with clock gating. The gating logic enabled the clock to all registers only in the WRITEBACK state. The results from power analysis indicate negligible difference in power consumed between the two circuits. However, the gate count of the circuit with clock gating is higher, and its timing is worse than the circuit with CGC.

Table 2. Summary of results

	Without CGC/clock gating	With CGC	With Clock Gating
Number of gates	28,800	26,794	29,150
Worst case path	8.97ns	8.64ns	9.30ns
Average power consumed	28.49mW	18.29mW (35% reduction)	18.29mW (35% reduction)
Average register file power	10.34mW	3.59mW (65% reduction)	3.59mW (65%reduction)
Average clock driver power	7.36mW	2.75mW (62% reduction)	2.75mW (62% reduction)

For all three implementations of the processor, we used the same timing constraints for synthesis. The operating voltage was set to 1.95 Volts and all three designs were clocked at 100Mhz for power analysis. Table 2 summarizes the important results obtained.

We have also implemented a tool TICTOC, for automatic synthesis of CGC. It accepts a state machine description and outputs synthesizable VHDL code after inserting CGC. The tool also outputs VHDL code for a testbench that instantiates the FSM with CGC and the original FSM. The test bench uses VHDL assertion checks to verify the equivalence of the two state machines. The tool has been implemented in Perl and has been written in approximately 1200 lines of code.

5. CONCLUSIONS

In this paper, we describe a new scheme named "Control generated clocking" (CGC) for minimizing dynamic power dissipation of registers and clock drivers in a synchronous sequential circuit. CGC minimizes power by utilizing control signals generated by FSMs as clocks for the registers. The power savings obtained from CGC are comparable to the savings obtained from clock gating. CGC is a robust method in which power minimization is achieved without the possibility of introducing glitches on the clocks. Timing analysis of a circuit with CGC is simpler than in circuits with gated clocks. CGC preserves the cycle boundaries on which registers load data thereby allowing reuse of test cases developed for the original design without any modifications. CGC is a structured method and can be easily incorporated into a synthesis tool for automation.

6. REFERENCES

1. A survey of Power Estimation Techniques in VLSI circuits, Farid N. Najm, IEEE Transactions on VLSI, December 1994.
2. Power Minimization in IC design: Principles and Applications, Massoud Pedram, ACM TODAES Vol. 1, No. 1, January 1996.
3. Automatic Synthesis of Gated Clocks for Power reduction in Sequential Circuits, L. Benini, P. Siegel, G. De. Micheli, IEEE Design and Test, Winter 1994, pp. 32-41
4. Transformation and Synthesis of Finite State Machines for Low Power Gated Clock Implementation, L. Benini & G. De Micheli , IEEE Transactions on CAD of Integrated Circuits and Systems, Vol. 15. No. 6., 1996.
5. State Assignment for Low Power Dissipation, L. Benini & G. De. Micheli, IEEE Journal of Solid State Circuits 1995.
6. Low Power State Assignment targeting Two- and Multi-level Logic Implementation, C. Y. Tsui, M. Pedram & A. M. Despain, Proceedings of the ICCAD 1994.
7. Retiming Sequential Circuits for Low Power, J. Montiero, S. Devadas & A. Ghosh, Proceedings of the ICCAD 1993
8. Precomputation based Sequential Logic Optimization for Low Power, M. Alidina, J. Montiero, S. Devadas & A. Ghosh, IEEE transactions on VLSI systems, 1994
9. Guarded Evaluation: Pushing Power management to Logic Synthesis design, V. Tiwari, S. Malik & P. Ashar, Proceedings of the International Symposium on Low Power Design, 1995.
10. Control Generated Clocking: A technique for minimizing power in sequential circuits, Srikanth Rao M, Research report RR-CADL-99-08, CAD lab, SERC, Indian Institute of Science.
11. Computer Architecture: A quantitative approach, 2nd ed., John Hennesey and David Patterson, Morgan Kaufman Publishers, 1996.