Reducing the Power Consumption of FPGAs through Retiming

Robert Fischer
Institute for
Computer Engineering
Universität der Bundeswehr
Munich, Germany
robert.fischer@
informatik.unibw-muenchen.de

Klaus Buchenrieder
Institute for
Computer Engineering
Universität der Bundeswehr
Munich, Germany
klaus.buchenrieder@
informatik.unibw-muenchen.de

Ulrich Nageldinger
Infineon Technologies AG
Munich, Germany
ulrich.nageldinger@
infineon.com

Abstract

High power dissipation is one of the major disadvantages of FPGAs. A main part of the power consumed is caused by glitches. This paper analyzes the effect of retiming to reduce the power dissipation of a Xilinx Virtex-II FPGA. The authors introduce a method to insert staging registers into large designs, that are constructed from a high abstraction level language algorithmic description. Results obtained by measurements suggest a high potential for power savings through retiming.

1. Introduction

Field Programmable Gate Arrays (FPGAs) are important devices in modern signal processing applications. Using FPGAs, large amounts of data can be processed in parallel, yielding high performance. With hardware description languages on the system level, like C-based languages, reconfigurable devices are easy and fast to program.

Compared to ASIC implementations, FPGA realizations consume a great amount of power. For this reason, ASIC solutions are often preferred over FPGA realizations despite the advantage in cost.

Reconfigurable devices are increasingly used for battery powered applications like mobile phones or mobile sensors, therefore, the reduction of power dissipation is becoming a major issue.

The second reason to focus on low power consumption is heat production by FPGAs, especially when the devices are driven at frequencies of 500MHz and above. At this speed individual chips can consume more than 40W of power, what requires active cooling. To counteract the production of this heat it is necessary to reduce the power consumption of the device.

The method described in this contribution is an attempt to reduce the dynamic power consumption of FPGAs. Section 2 of this contribution provides technical background information. Section 3 explains the method and effects of retiming, describes the experimental setup and provides results obtained by measurements. Section 4 shows a method of easily introducing new staging registers into a design. In the last section conclusions and a proposal for future work is given.

2. Technical background

In this Section, a short introduction into the power consumption of FPGAs is given and glitches are briefly explained.

2.1. Power consumption of FPGAs

Total power consumption of a FPGA can be split into two parts: Static power consumption and dynamic power consumption. Static power consumption is caused by the leakage current of elementary devices and is not affected by the custom logic circuit implemented with the FPGA. It sorely depends on the voltage the device is operating on and the temperature of the device and not on the custom design implemented on it. Normally manufactures provide a typical value in the data sheet of the device.

Even though static power consumption is becoming increasingly significant, the dynamic power still dominates, even in a $0.13\mu m$ technology [11]. In FPGAs, like in any other CMOS circuit, it is caused by two phenomena:

1.) While switching a signal line through two transistors from the high-state to low-state or the other way around, brief short-circuits can occur because of slight delays in the switching of the transistors (See [10]).

2.) Charging and discharging of load capacitors, due

Proceedings of the 12th IEEE International Conference and Workshops on the Engineering of Computer-Based Systems (ECBS'05)
0-7695-2308-0/05 $20.00 © 2005 IEEE

to switching action. For n load capacitors, the average power consumption [1], P_{avg} is:

$$P_{avg} = \frac{1}{2} \sum_{n=1}^{m} Cn \cdot f_n \cdot V^2 \quad (1)$$

C_n is the capacitance of net n with the average toggle rate (switching activity) f_n, and V is the supply voltage.

In FPGA realizations the second effect is most significant. High capacitive loads result from long wires interconnecting configurable logic blocks (CLBs). For the Xilinx Virtex-II FPGA family the interconnect attributes roughly 60% of the total power consumed [8].

2.2. Glitches

In ASIC realizations glitches cause about 20% of the total power dissipation, but can even be as high as 70% of the total power consumed [9].

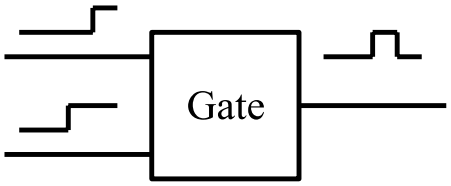

Figure 1. Unequal delays in the input signals are causing glitches

Glitches are caused, when two signals are run through paths of different length, leading to unmatched delays. This effect results in erroneous pulses or transitions at gated outputs, as shown in Figure 1. When gates are directly chained in series, these transitions can build up to multiple and dynamic hazards. Figure 2 shows such a build up of glitches. These transitions increase the switching activity f_n and therefore increase power consumption, as Equation (1) suggests.

Compared to ASICs, in FPGAs power consumption is dominated by capacitive loads on interconnects, so that the impact of glitches on the total power dissipation is dominant.

Glitches are normally prevented by balancing the delays in custom designs. But developing FPGA designs on a high level of abstraction with C-based hardware description languages (HDL), becomes more and more popular. With languages like Handel-C it is not possible to access certain functions like multiplications or divisions directly on the gate level. Therefore, it is not directly possible to ensure, that the delays of all the gates are balanced and glitches are prevented.

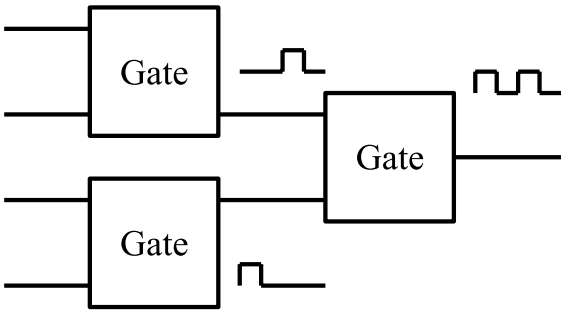

Figure 2. More gates in series can cause a build-up of glitches

3. The effect of retiming on power

Retiming is an optimization method in VLSI design for sequential logic circuits, based on the "Retiming Lemma", which was first formulated 1981 by Leiserson and Saxe[5]. This method was created to improve performance of synchronous circuits without changing its behavior. The lemma stated, that under "appropriate initializations" retiming does not change the functionality of the circuit.

During retiming, flip-flops are redistributed along a signal path, so that the circuit can be driven at a higher clock rate. It balances flip-flop stages in the circuit to minimize the logic delay without introducing new flip-flop stages.

As an example, consider the circuit in Figure 3. Prior to retiming the logic delay is twice the delay compared to the circuit after retiming, yielding a circuit, that can be driven with the doubled frequency.

Today many modern synthesis programs, like Simplicity's Simplify Pro or Celoxica's DK Design Suite offer a retiming option, to improve the performance of a design without much interaction from the designer.

3.1. Retiming for low power

As described earlier, glitches are the main cause of unnecessary switching activity resulting in a high power drain. Glitches can be prevented by synchronizing the input signals of a gate by insertion of flip-flops at gate inputs. By this measure, even existing glitches in the input signals are removed what prevents hazard build-up and decreases the power consumption of the circuit realized with the FPGA.

The availability of flip-flops in modern FPGAs is not a problem, since every LUT is followed by a flip-flop. In the Xilinx series LUTs can be used as additional 16-Bit shift registers, which are basically identical to a series of 16 flip-flops.

The positive effect of retiming on glitches is state-of-the-art in the ASIC arena [6] [3] [4], however, there are

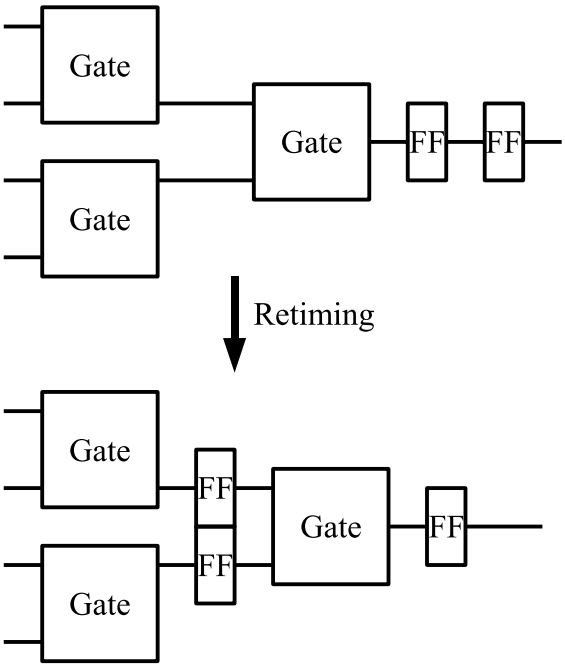

Figure 3. The redistribution of flip-flops through retiming

nearly no experimental results, to achieve a lower power consumption, for FPGAs. The approach described in [11] uses *Pipelining* to reduce the occurrence of glitches and optimize the power consumption of FPGAs. The Authors illustrate the baseline and provide results using Altera FPGAs. The measures provided in their work have been obtained by manual introduction of new staging flip-flops into a netlist file. This, of course, does not conserve the original behavior of the circuit. As shown new flip-flop stages introduced into the signal path require to resynchronize parallel signal flows.

3.2. Experimental setup

For the experiments conducted in our research, a Celoxica RC200 Board [7] with a Xilinx Virtex II (XC2V1000-4FG456C) FPGA was used.

Comparative figures were obtained by measuring the power directly at the board level. We ensured, that no other parts of the board draw dynamic power and therefore could interfere with the measurements by putting all of the other parts into sleep mode. Permanent power drains, like the touch-screen display were disconnected from the power source.

The test bench was written in Handel-C and compiled with the Design Kit 3 of Celoxica to an EDIF-file. This software allows retiming on this high level of abstraction, which is preferred, because there optimization through retiming can be most effective. Subsequent placement and routing was done with the Xilinx ISE 6.3 software.

The following test suite was deployed:

add: The program adds a counter value to the 16-bit values of a vector with K elements parallel

mult: The program multiplies a counter value with the 16-bit values of a vector with K elements parallel (the fixed multiplier units of the Xilinx Virtex II were not used, instead the multipliers were laid out normally on FPGA)

shift: Performs parallel shift operations on 32-bit words of a 64*64 Matrix

dist: Calculates the distance between two points (also no fixed multipliers were used)

mand: Performs Mandelbrot calculations for a 10000 x 10000 matrix (also the fixed multipliers were not used)

Each program was compiled once with retiming disabled and then with retiming enabled. Although the retimed programs can be driven at a higher frequency, the clock rate before and after retiming was not altered.

3.3. Results

The results in Table 3.3 show, that there is a clear potential to save up to 10% of a boards total power dissipation. In our experimental setup the power measured included the unused on-board ram, a CPLD, several other hardware components and the losses of the on-board voltage converters. That means, that the savings in respect to the FPGA core power consumption alone are even greater.

Results also show, that for some test programs, power savings are minimal or close to zero. This is partly due to the fact, that Celoxica's retiming feature is targeted to improve the speed of circuits and is not optimized for saving power. Also some programs can not be optimized by this method, if the circuit is already fully pipelined or the flip-flop stages are already optimized. Celoxica's retiming algorithm has also some limitations [2], for example it can not move flip-flops through block RAM stages or the fixed multiplier units of the Virtex II. Also all flip-flops can only be moved in a circuit with the same clock source.

Especially for the *shift* program the benefits of retiming are not measurable. This is not surprising, since a shift circuit is nothing more than a series of flip-flops, with no logic or path delays in between them. This way it is already optimized and no glitches can occur.

PUT	K	R	Current in mA	Power in W	Power saving
add	64		204	2,45	
	64	X	195	2,34	4,41%
	128		215	2,58	
	128	X	194	2,33	9,77%
mult	64		202	2,42	
	64	X	194	2,33	3,96%
	256		217	2,60	
	256	X	195	2,34	10,14%
shift			200	2,40	
		X	200	2,40	0%
dist			221	2,65	
		X	211	2,53	4,5%
mand			206	2,47	
		X	206	2,47	0%

Table 1. Experimental results of power measurements on the RC200 board for the different programs under test (PUT) without and with retiming(R)

4. Introduction of new staging registers on system level

For some programs shown in Section 3, the power optimizations did not yield optimal results, because there were not enough flip-flops in the circuit, that can be used for reallocation by the retiming algorithm.

For better understanding consider the example in Figure 4, that contains only one flip-flop and therefore only one register stage. Clearly, optimization would require at least one more register stage, which could then be reallocated by the retiming algorithm to minimize the logic delay of the circuit.

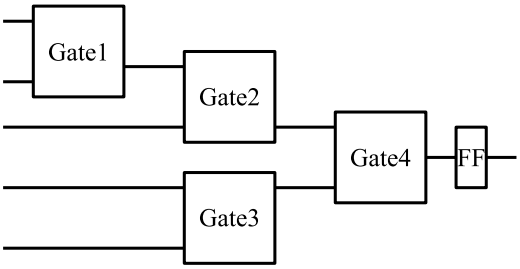

Figure 4. Circuit, which cannot be optimized well by retiming

More glitches can be prevented, if more register stages are introduced into a sequential circuit, thereby pipelining the design. This, of course, does not preserve the original behavior of the circuit. If one register stage is added, the results of this circuit will be delayed by one clock cycle.

Adding flip-flops into the design is not a trivial task. Especially in highly parallelized designs, adding flip-flops at the wrong places can lead to dysfunctional circuits.

If for example in Figure 4 a flip-flop is added between $Gate2$ and $Gate4$, the inputs of $Gate4$ would be out of synch. The result of $Gate2$ would always be one clock cycle behind the one of $Gate3$, and thereby the result of the whole circuit would come out wrong. To correct this, the result from $Gate3$ must also be delayed by one clock cycle, by adding another flip-flop between $Gate3$ and $Gate4$. This way a complete register stage is added and the result of the whole circuit is correct again (simply delayed by one clock cycle).

Another problem is, that on system level it is not always possible to insert register stages between all gates on the gate level, since some functions like multiplications or divisions are complex gate structures, which are not fully accessible on this abstraction level.

4.1. Pipelining by retiming

By shifting the inputs or the results of a circuit through some flip-flops, new register stages can easily be added to the design. For our example we add two more register stages to the design, by shifting the the result at the end as shown in Figure 5.

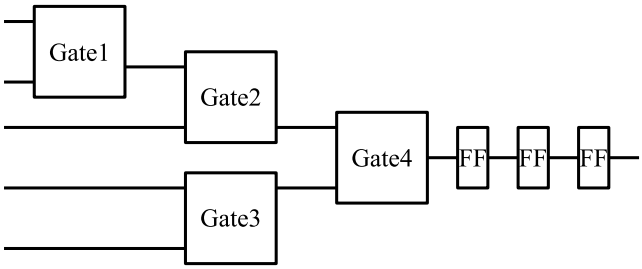

Figure 5. Circuit with added register stages

As a result, there are more register stages in the circuit, that can be moved by the retiming algorithm, resulting in a more pipelined circuit, where more glitches can be prevented. Therefore the power consumption of the design is lower, the timing behavior, however, different from the original design. In the ideal case the whole circuit becomes fully pipelined with flip-flops after every gate, as illustrated in Figure 6.

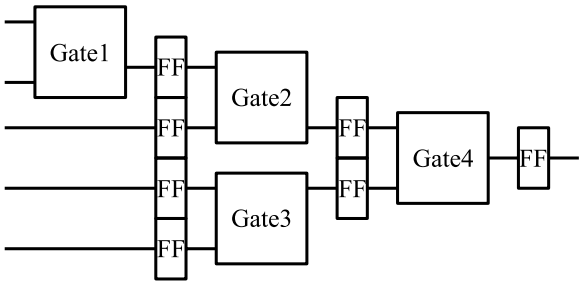

Figure 6. Retimed circuit

The downside of this pipelining is, that the design now takes two clock cycles longer to compute. The overall throughput, however, remains the same.

4.2. Experimental Setup

The setup for the measurement remained the same as for the retiming measurements.

For these experiments the dist program was modified, so that the inputs could be shifted through a given number RS of register stages. The program was always compiled with the retiming option set.

4.3. Results

As comparison for the "Power saving" percentages in table 4.3 the power consumption of the design without retiming was taken.

PUT	RS	Current in mA	Power in W	Power saving
dist	0	211	2,53	4,5%
	1	208	2,50	5,9%
	4	190	2,28	14,0%
	7	198	2,38	10,4%

Table 2. Experimental results of power measurements on the RC200 board for the different programs under test (PUT) with different numbers of added register stages (RS)

Results in Table 4.3 show, that for increasing number of inserted register stages the power consumption of the circuit decreases up to a point where the design is saturated and fully pipelined. After that point the power consumption begins to increase again.

5. Conclusions and future work

Even though Celoxica's retiming algorithm is not optimized for saving power, measurements in Section 3 show, that retiming can be used to lower the power consumption of FPGA realizations. If more power reduction is needed, the design can easily be pipelined with the method described in Section 4.

For power critical designs retiming has provided good results despite a high compilation time of up to one hour on a Pentium 4 with 3.2GHz for large designs.

Current work focusses on exact power measurements of FPGA circuits. For this an improved test board is currently under design. This board will allow to directly probe the FPGA power pins.

In our work we found, that FPGA designers need better tools to estimate the power consumption of their circuit on a high abstraction level. As a consequence, we develop a new suite of test programs, different algorithms and constructs in Handel-C, to obtain new models for power estimation at an early stage of a system design cycle.

References

[1] J. H. Anderson and F. N. Najm. Power estimation techniques for FPGAs. In *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, volume 12, pages 1015–1027, Oct. 2004.

[2] Celoxica. *The application of retiming to the synthesis of C based languages using the Celoxica DK Design Suite - white paper*. Mar. 2004. http://www.celoxica.com/.

[3] N. Chabini, I. Chabini, E. M. Aboulhamid, and Y. Savaria. Unification of basic retiming and supply voltage scaling to minimize dynamic power consumption for synchronous digital designs. In *GLSVLSI '03: Proceedings of the 13th ACM Great Lakes Symposium on VLSI*, pages 221–224. ACM Press, 2003.

[4] Y.-L. Hsu and S.-J. Wang. Retiming-based logic synthesis for low-power. In *ISLPED '02: Proceedings of the 2002 International Symposium on Low Power Electronics and Design*, pages 275–278. ACM Press, 2002.

[5] C. E. Leiserson and J. B. Saxe. Optimizing synchronous systems. In *22nd Annual Symposium on Foundations of Computer Science*, pages 23–36. IEEE, New York, NY, USA, 1981.

[6] J. Monteiro, S. Devadas, and A. Ghosh. Retiming sequential circuits for low power. In *Proceedings of the 1993 IEEE/ACM International Conference on Computer-Aided Design*, pages 398–402. IEEE Computer Society Press, 1993.

[7] C. M. Pepler). *RC200/203 Hardware and PSL Reference Manual*. Oct. 2004. http://www.celoxica.com/.

[8] L. Shang, A. S. Kaviani, and K. Bathala. Dynamic power consumption in Virtex-II FPGA family. In *Proceedings of the 2002 ACM/SIGDA tenth International Symposium on Field*

Programmable Gate Arrays, pages 157–164. ACM Press, 2002.

[9] A. Shen, A. Ghosh, S. Devadas, and K. Keutzer. On average power dissipation and random pattern testability of CMOS combinational logic networks. In *Proceedings of the 1992 IEEE/ACM International Conference on Computer-Aided Design*, pages 402–407. IEEE Computer Society Press, 1992.

[10] N. H. E. Weste and K. Eshraghian. *Principles of CMOS VLSI design: A systems perspective*. Addison-Wesley Longman Publishing Co., Inc., 1985.

[11] S. J. Wilton, S.-S. Ang, and W. Luk. The impact of pipelining on energy per operation in field programmable gate arrays. In *Field-Progammable Logic and Applications, LNCS 3203*, pages 719–728. Springer-Verlag, 2004.