ISSN (Print) : 2320 – 3765
ISSN (Online): 2278 – 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 12, December 2014

Flip-Flop Grouping in Data-Driven Clock Gating for Dynamic Power Management

N.Indhumathi[1,] Dr.S.Nirmala[2]

PG Student [Applied Electronics], Dept. of ECE, Muthayammal Engineering College, Namakkal, Tamilnadu, India[1]

Professor, Dept. of ECE, Muthayammal Engineering College, Namakkal, Tamilnadu, India[2]

ABSTRACT: Data driven clock gating is a popular technique used in many synchronous circuits for reducing dynamic power dissipation. Power optimization plays the important role in the recent years. Clock gating is a popular technique used in many synchronous circuits for reducing dynamic power dissipation.Gating manually inserted into the register transfer level (RTL) design. when a logic unit is clock, its underlying sequential elements receive the clock signal regardless of whether or not they will toggle in the next cycle. In this flip-flops are grouped so that they share a common clock enabling signal to reduce the hardware overhead. It is observed that the commonly used synthesis based gating still leaves a large amount of redundant clock pulses. In these d-flip flops are used to grouping for reducing the power. Here the Xilinx software tool will be used for implementing this proposal system.

KEYWORDS*:* Clock gating, clock networks, dynamic power Reduction

I.INTRODUCTION

One of the major dynamic power consumers in computing and consumer electronics products is the system's clock signal, where it takes 30%–70% of the total dynamic power consumption [1]. There are many techniques used to reduce the dynamic power are developed, in which clock gating is predominant. Ordinarily, when a logic unit is clocked, it is based on the sequential elements receiving the clock signal, sequentially they will toggle in the next cycle whether it is required or not. With clock gating, the clock signals are ANDed with explicitly predefined enabling signals. Clock gating is employed at all levels: system architecture, block design, logic design, and gates [2], [3]. Several methods to take advantage of this technique are described in [4]–[6], with all of them depending on various heuristics in an attempt to increase clock gating opportunities. With the rapid increase in design complexity, computer aided design tools supporting system-level hardware description have become commonly used. Although substantially increasing design productivity, such tools require the employment of a long chain of automatic synthesis algorithms, from register transfer level (RTL) down to gate level and net list. Unfortunately, such automation leads to a large number of unnecessary clock toggling, thus increasing the number of wasted clock pulses at flip-flops (FFs).

In a recent paper, a model for data-driven gating is developed based on the toggling activity of the constituent FFs [9]. The optimal fanout of a clock gater yielding maximal power savings is derived based on the average toggling statistics of the individual FFs, process technology, and cell library in use. In general, the state transitions of FFs in digital systems depend on the data they process. Assessing the effectiveness of data-driven clock gating requires, therefore, extensive simulations and statistical analysis of the FFs' activity. Another grouping of FFs for clock switching power reduction, called multibit FF (MBFF). MBFF attempts to physically merge FFs into a single cell such that the inverters driving the clock pulse into its master and slave latches are shared among all FFs in a group. MBFF grouping is mainly driven by the physical position proximity of individual FFs, while grouping for data driven clock gating should combine toggling similarity with physical position considerations. While answered the question of what is the group size that maximizes power savings, this paper studies the questions of: 1) which FFs should be placed in a group to maximize the power reduction and 2) how to algorithmically derive those groups. Implementation .In the next section, we briefly overview data-driven clock gating, which motivates this paper.. Section III introduces layout considerations into FF grouping and describes a near-optimal grouping algorithm. Section IV presents the problem of optimal FF grouping and its inheren difficulty. Section V discusses the implementation of a practical design flow. Section VI

10.15662/ijareeie.2014.0312062

presents experimental results obtained for digital signal processor (DSP) and 3-D graphic designs. Final conclusions are presented in Section VII.

II. DATA-DRIVEN CLOCK GATING

Clock enabling signals are very well understood at the system level and thus can effectively be defined and capture the periods where functional blocks and modules do not need to be clocked. Those are later being automatically synthesized into clock enabling signals at the gate level. In many cases, clock enabling signals are manually added for every FF as a part of a design methodology. Still, when modules at a high and gate level are clocked, the state transitions of their underlying FFs depend on the data being processed. It is important to note that the entire dynamic power consumed by a system stems from the periods where modules' clock signals are enabled.. Fig. 1 shows the FFs' toggling activity in an arithmetic block comprising 22K FFs, designed in 40-nm technology ,taken from Ceva's X1643 DSP core for multimedia and wireless baseband applications The statistics is obtained from extensive simulations of typical modes of operation, consisting of 240-K clock cycles. when the FFs clock signal is enabled is only 10%, which is still responsible for the entire dynamic power consumed by that block. The clock enabling signals are obtained by RTL synthesis and manual insertions. As Fig. 1 shows, a FF toggled its state only 2.9% of the clock enabled time period, on the average, thus more than 97% of the clock pulses driving FFs are useless.

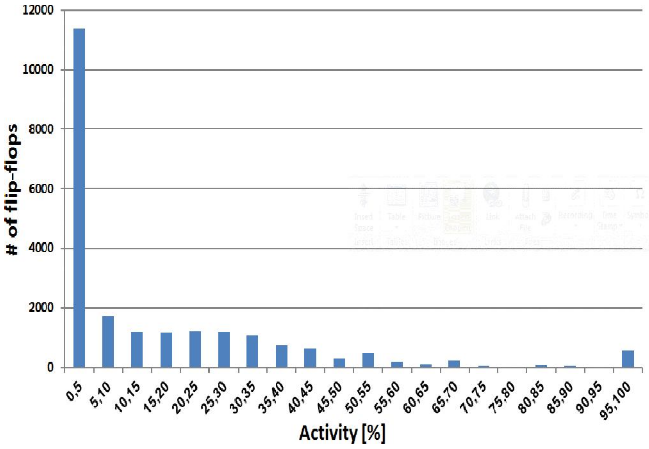


Fig. 1: Toggling statistics of Ceva's X1643 DSP core over 240-K clock cycles.

A FF finds out that its clock can be disabled in the next cycle by XOR ing its output with the present data input that will appear at its output in the next cycle. The outputs of *k* XOR gates are ORed to generate a joint gating signal for *k* FFs, which is the n latched to avoid glitches. The combination of a latch with AND gate is commonly used by commercial tools and is called integrated clock gate (ICG). Such data driven gating is used for a digital filter in an ultralow-power design. A single ICG is amortized over *k* FFs. There is a clear tradeoff between the number of saved (disabled) clock. pulses and the hardware overhead. With an increase in *k*, the hardware overhead decreases but so does the probability of disabling, obtained by OR ing the *k* enable signals. Let the average toggling probability of a FF (also called activity factor) be denoted by *p* ($0 < p < 1$). The latch and gater (AND gate) overheads are amortized over *k* FFs. It is shown in [9] that the number *k* of jointly gated FFs for which the power savings are maximized is the solution of

$$(1-p)k\ \ln(1-p)\ (cFF+cW)+clatch/k2=0 \quad (1)$$

where *c*FF is the FFs clock input capacitance, *c*W is the unit-size wire capacitance ,and *c*latch is the latch capacitance including the wire capacitance of its *clk* input. Such a gating scheme has considerable timing implications, which are discussed in [9]. We will return to those when discussing the implementation of data-driven gating as a part of a complete design flow. For the scheme proposed in Fig. 2 to be beneficial, the clock enabling signals of the grouped FFs should preferably be highly correlated. Data-driven clock gating is shown to achieve savings of more than 10% of the total dynamic power consumed by the clock tree It took advantage of the very low dynamic range of the data in a digital filter. The gating logic is tailored to the structure of the filter, whereas the approach discussed in this paper is more general and applies to large scale and a wide range of designs.

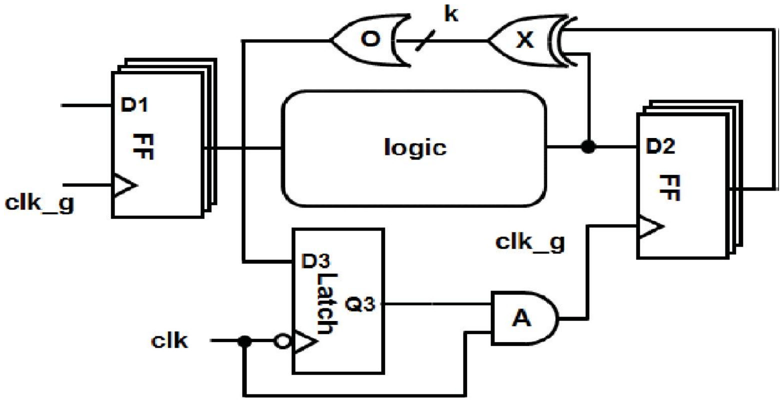


Fig. 2: Practical data-driven clock gating.

III. PHYSICAL LAYOUT CONSIDERATIONS IN FFS GROUPING

Finding sets of FFs that minimize the number of redundant clock pulses is not enough to maximize power savings. Grouping must account for the on-die locations of FFs and gaters, which affect the power consumption due to the capacitive loads resulting from their connections. The physical locations of FFs affect also the delay and clock skew, and it is therefore desirable for FFs driven jointly by the same clock gater, to be placed in proximity of each other. and energy consumption are calculated.

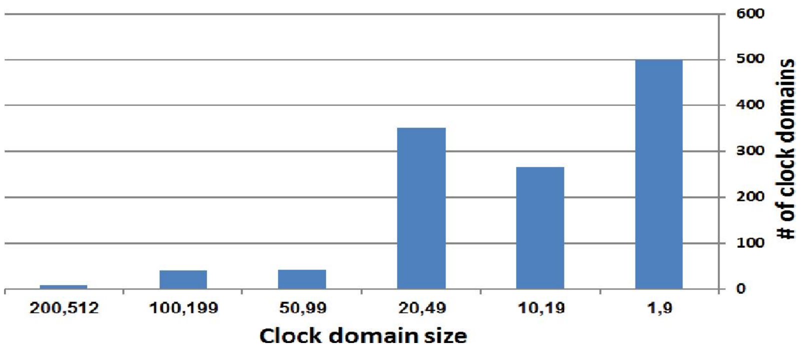


Fig. 3. Distribution of the number of FFs sharing a common pre-enabled clock signal of a dsp core

IV. OPTIMAL FFS GROUPING FOR JOINT CLOCK GATING

Knowing the optimal group size *k*, the next step is to partition the FFs of a system into *k*-size sets such that the power savings will be maximized .Such tools are focusing on skew, power, and area minimization, but they are not aware of the toggling correlations of the underlying FFs, which this paper is focusing on. The optimal value of *k* is obtained from (1) under toggling independence assumption, but in reality the toggling may be correlated, so in practice one can expect higher saving than the theoretical lower bound obtained under independence assumption. A practical design methodology should preserve the integrity of system clock enabling signals. This means that the FFs of a *k*-size set must all belong to the same enabled clock (called hereafter pre-enabled).

10.15662/ijareeie.2014.0312062

A bottom-up process for a coarse, block-level gating is proposed in [5] by repeating the MCPM algorithm. We have adapted this idea to FF-level gating. Starting with n individual FFs and constructing the associated n-vertex FF pair wise activity graph, an MCPM algorithm then finds the best FFs pairing. A new $n/2$-vertex pair wise activity graph is then defined where its vertices correspond to the matching ($n/2$ edges) found in the former step. The process repeats K times until groups of size $k = 2K$ are determined. For $k = 2(K = 1)$, MCPM indeed solves the problem of minimizing the number of redundant clock pulses, but its repetitive application for $k > 2$ ($K > 1$) may not find the minimum, as otherwise this would contradict the NP-hardness. Still, the iterative MCPM algorithm is practical and has acceptable run time.

V. IMPLEMENTATION AND INTEGRATION IN A DESIGN FLOW

In the following, we describe the implementation of data driven clock gating as a part of a standard backend design flow. It consists of the following steps.

1) Estimating the FFs toggling probabilities involves running an extensive test bench representing typical operation modes of the system to determine the size k of a gated FF group by solving (1).

2) Running the placement tool in hand to get preliminary preferred locations of FFs in the layout.

3) Employing a FFs grouping tool to implement the model and algorithms presented in Sections III and IV, using the toggling correlation data obtained in Step 1 and FF locations' data obtained in Step 2. The outcome of this step is k-size FF sets (with manual overrides if required), where the FFs in each set will be jointly clocked by a common gater.

4) Introducing the data-driven clock gating logic into the hardware description (we use Verilog HDL). This is done automatically by a software tool, adding appropriate Verilog code to implement the logic described in Fig. 2. The FFs are connected according to the grouping obtained in Step 3. A delicate practical question is whether to introduce the gating logic into RTL or gate level description. This depends on design methodology in use and its discussion is beyond the scope of this paper. We have introduced the gating logic into the RTL description.

5) Re-running the test bench of Step 1 to verify the full identity of FFs' outputs before and after the introduction of gating logic. Although data-driven gating, by its very definition, should not change the logic of signals, and hence FFs toggling should stay identical, a robust design flow must implement this step.

6) Ordinary backend flow completion. From this point, the backend design flow proceeds by applying ordinary place and route tools. This is followed by running clock tree synthesis.

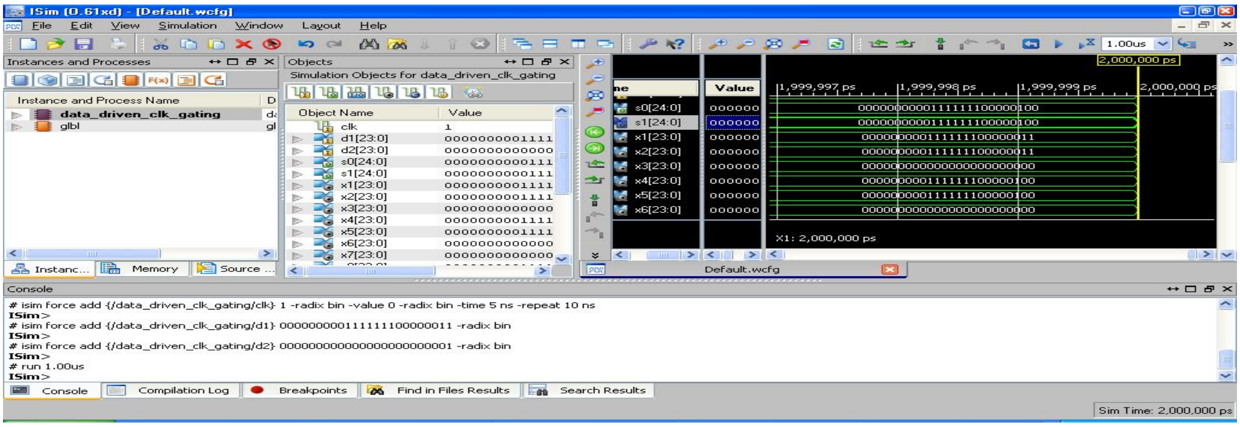
Few timing-related comments are in order. The extra gating delay introduced by the feedback loop in Fig. 2 should not exceed the delay margins of paths from the clock input *clk_g* of FF1 to the data input D2 of FF2. In ordinary designs, notably in automatically synthesized blocks, most of the delay margins are large enough to absorb the introduction of the gating logic. If at a later stage timing violations due to the gating are found, one can simply drop the data-driven gating from the troublesome FFs. We found very few of those in our designs, less than 5% of the FFs. Relaxation of the clock cycle can also overcome this problem, but it must be considered in a wider context of power-delay trade off and product specifications, which is beyond the scope of this paper.

VI. SIMULATION RESULTS

The design flow described in Section IV is experimented on a DSP core comprising 22 k FFs (Fig. 1), another large vectored DSP core comprising 100 k FFs, a 3-D graphics accelerator [9], and a network processor control block. For FF toggling probabilities from $p = 0.01$ to $p = 0.05$, the group size maximizing the net power savings is The latter is due to the very low toggling rate of that processor. As shown in the tables, the net dynamic power savings are 15% for the DSP cores and 20% for the 3-D graphics accelerator. Experiments for the network processor control block also yield nearly 20% power savings where the optimal group size is $k = 8$. between $k = 8$ and $k = 4$. We measured the power savings compared with the nominal designs using the Spy Glass EDA power simulator [8].

Fig 4 shows that the output waveform for data driven clock rating with carry skip adder. Here to give a 24 bit d1 and d2 input bits and get a 24 bit s0 and s1 output bits. The output waveform s0 indicates sum of the d1 and d2 inputs then next clock pulse to obtain the same input s1.

10.15662/ijareeie.2014.0312062


ISim (O.61xd) - [Default.wcfg]
File Edit View Simulation Window Layout Help
Instances and Processes
Instance and Process Name
data_driven_clk_gating
glbl
Objects
Simulation Objects for data_driven_clk_gating
1.00us
2,000,000 ps
1,999,997 ps
1,999,998 ps
1,999,999 ps
2,000,000 ps
s0[24:0] 0000000000111111100000100
s1[24:0] 0000000000111111100000100
x1[23:0] 000000000111111100000011
x2[23:0] 000000000111111100000011
x3[23:0] 000000000000000000000000
x4[23:0] 000000000111111100000100
x5[23:0] 000000000111111100000100
x6[23:0] 000000000000000000000000
X1: 2,000,000 ps
Default.wcfg
Instanc... Memory Source ...
Console Compilation Log Breakpoints Find in Files Results Search Results
Sim Time: 2,000,000 ps

Object Name	Value
clk	1
d1[23:0]	0000000001111
d2[23:0]	0000000000000
s0[24:0]	000000000111
s1[24:0]	000000000111
x1[23:0]	0000000001111
x2[23:0]	0000000001111
x3[23:0]	0000000000000
x4[23:0]	0000000001111
x5[23:0]	0000000001111
x6[23:0]	0000000000000
x7[23:0]	0000000000000

```
# isim force add {/data_driven_clk_gating/clk} 1 -radix bin -value 0 -radix bin -time 5 ns -repeat 10 ns
ISim>
# isim force add {/data_driven_clk_gating/d1} 000000000111111100000011 -radix bin
ISim>
# isim force add {/data_driven_clk_gating/d2} 000000000000000000000001 -radix bin
ISim>
# run 1.00us
ISim>
```

Fig 4: Power Analysis for flip flop grouping

It is interesting to compare the relative power savings achieved by applying synthesis-based gating only or data driven gating only or both. To this end, we ran RTL power simulation of the processors in four gating modes:

1) no gating;

2) synthesis-based available from an EDA vendor ;

3) data-driven; and

4) both combined.

Summarizes the total power consumption of each case and compares it with the native design without any gating [shown in (1)]. The synthesis-based gating alone 2) reduced the total power to 84%, 34%, and 59%, respectively. Applying data-driven gating on top of the synthesis-based one 4) further reduced the power to 68%, 29%, and 50%, respectively. However, for all three designs, the application of data driven gating alone yielded higher power savings than the combination of data-driven and synthesis-based, reduction the power to 65%, 27%, and 42%, respectively. This is due to the fact that data-driven gating stops any unnecessary clock pulse, and the inclusion of the synthesis-based gating only adds logic circuits that becomes redundant once the data-based gating is applied. Although synthesis-based clock gating is a well-established design methodology, the above experiments encourage its replacement by data-driven gating. To further characterize the benefits of the two gating modes, we analyzed several different circuit types, for example, control units, arithmetic units, and register files.

Fig 5 shows

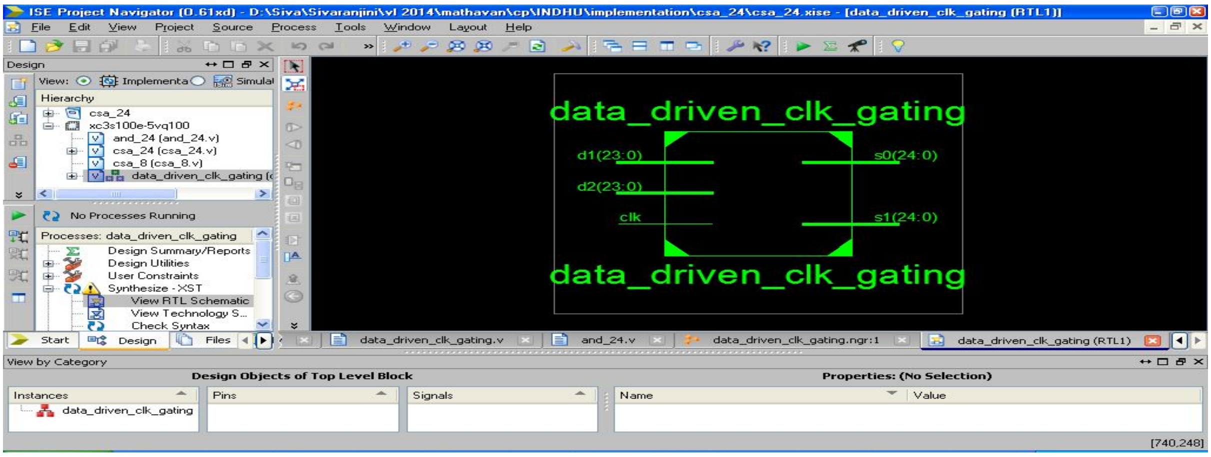
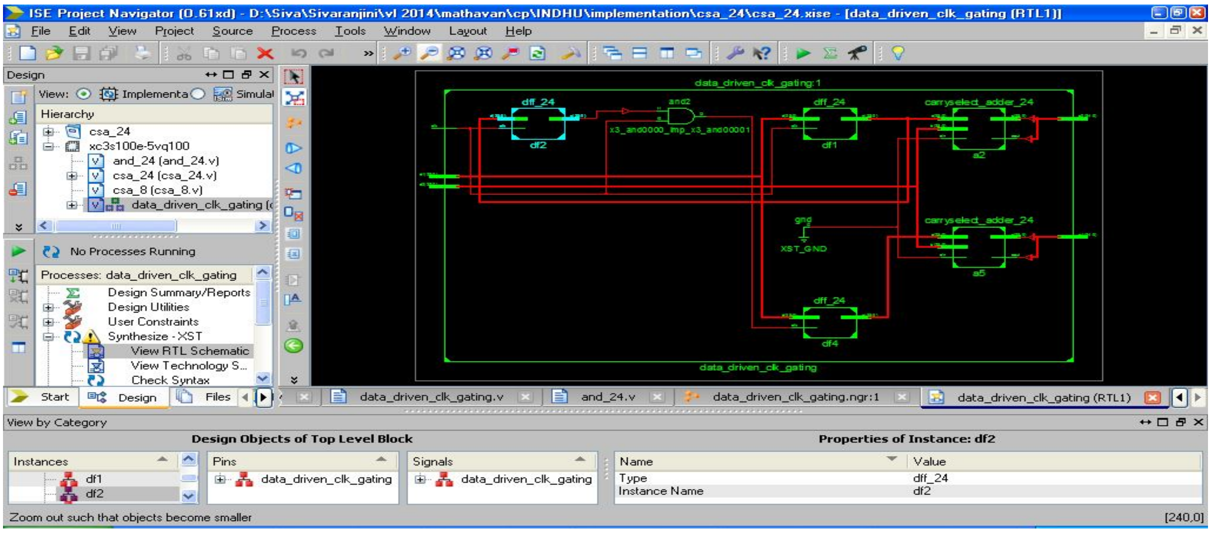


Fig:5 power analysis of register transfer level(RTL)

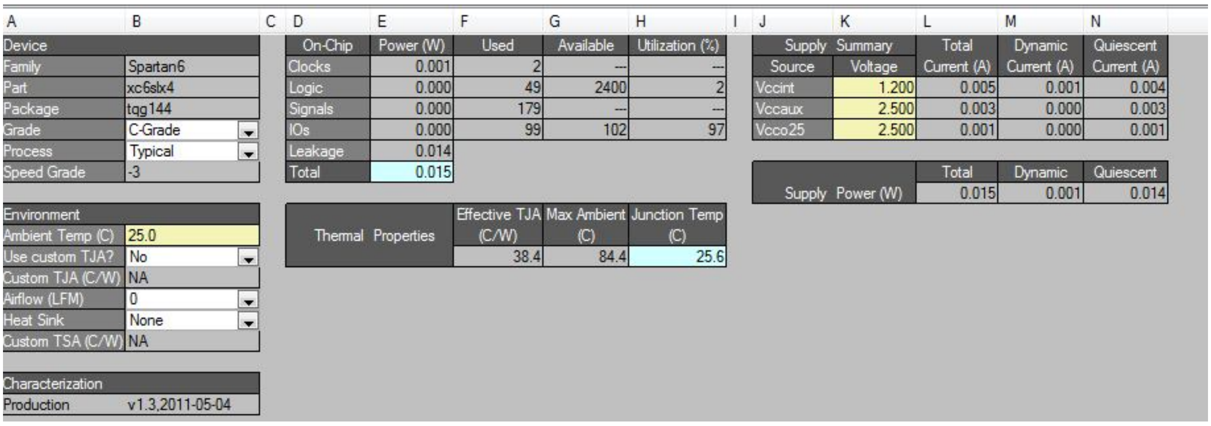
To further characterize the benefits of the two gating modes, we analyzed several different circuit types, for example, control units, arithmetic units, and register files. We selected such units from the large designs that we analyzed above, where the type of the circuit is specified. The resulting power consumptions (in percentages), are relative to the no clock gating case, and the lowest power is indicated in red (circled). It can be seen that for control circuits, data-driven gating 3) is outperforming synthesis-based gating 2). This is explained by their very low toggling rate, where data-driven is most useful. Similar behavior is observed for arithmetic circuits (their IO registers are included). Expectedly, synthesis-based gating is still favored for register files. This follows since only one register is changing its data at a time, a condition that can easily be caught and defined in the RTL code.



Name	Value
Type	dff_24
Instance Name	df2

Fig:6 View of rtl schematic using carry skip adder

Fig 6 shows that the RTL schematic diagram for data driven clock rating with carry skip adder using Xilinx tool from this we obtain the logic and gate block of the following synthesis code


A
B
C
D
E
F
G
H
I
J
K
L
M
N

Device	
Family	Spartan6
Part	xc6slx4
Package	tqg144
Grade	C-Grade
Process	Typical
Speed Grade	-3

Environment	
Ambient Temp (C)	25.0
Use custom TJA?	No
Custom TJA (C/W)	NA
Airflow (LFM)	0
Heat Sink	None
Custom TSA (C/W)	NA

Characterization	
Production	v1.3,2011-05-04

On-Chip	Power (W)	Used	Available	Utilization (%)
Clocks	0.001	2	---	---
Logic	0.000	49	2400	2
Signals	0.000	179	---	---
IOs	0.000	99	102	97
Leakage	0.014			
Total	0.015			

Thermal Properties	Effective TJA (C/W)	Max Ambient (C)	Junction Temp (C)
	38.4	84.4	25.6

Supply Source	Summary Voltage	Total Current (A)	Dynamic Current (A)	Quiescent Current (A)
Vccint	1.200	0.005	0.001	0.004
Vccaux	2.500	0.003	0.000	0.003
Vcco25	2.500	0.001	0.000	0.001

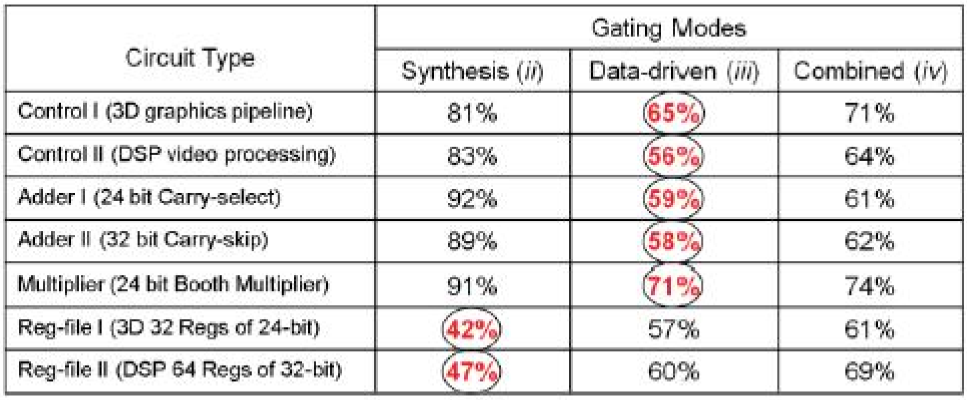
	Total	Dynamic	Quiescent
Supply Power (W)	0.015	0.001	0.014

Fig:7 Power analysis of data driven clock gating

Fig.7 shows the power analysis for the data driven clock rating with carry skip adder. Here mention supply power, leakage power, dynamic current and quiescent current.

10.15662/ijareeie.2014.0312062

It can be seen that for control circuits, data-driven gating 3) is outperforming synthesis-based gating 2). This is explained by their very low toggling rate, where data-driven is most useful. Synthesis -based gating is still favoured for register files. This follows since only one register is changing its data at a time, a condition that can easily be caught and defined in the RTL code. Therefore, applying data driven gating on top of synthesis-based mostly adds circuit overhead. The results of the combined synthesis-based and data-based gating scheme are worse than the data-driven only gating for all seven circuits. Thus, unless register files can undergo only synthesis-based gating and data-based gating will not be applied to them, synthesis-based gating should be completely replaced by data-based gating.



Circuit Type	Gating Modes		
	Synthesis (*ii*)	Data-driven (*iii*)	Combined (*iv*)
Control I (3D graphics pipeline)	81%	65%	71%
Control II (DSP video processing)	83%	56%	64%
Adder I (24 bit Carry-select)	92%	59%	61%
Adder II (32 bit Carry-skip)	89%	58%	62%
Multiplier (24 bit Booth Multiplier)	91%	71%	74%
Reg-file I (3D 32 Regs of 24-bit)	42%	57%	61%
Reg-file II (DSP 64 Regs of 32-bit)	47%	60%	69%

Table:1 power comparison table

Fig. 8 illustrates the margin (slack) distribution for 200-MHz clock cycle. It can be seen that the margin distribution has slightly worsened as more paths now have a negative slack. The violations need to be taken care of for timing closure and a variety of actions and techniques are possible, but their discussion is beyond the scope of this paper.

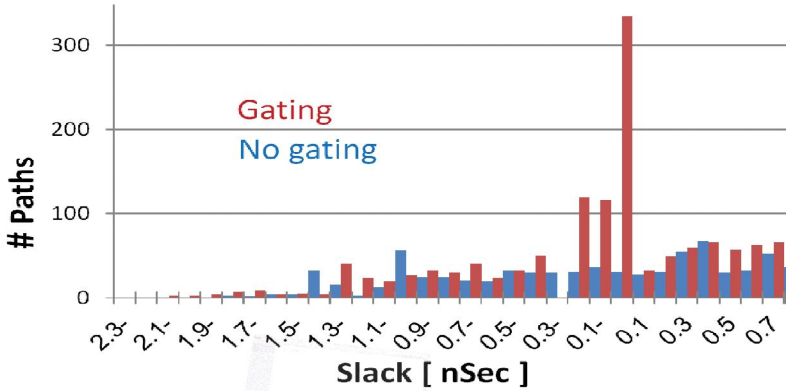


Fig: 8 Negative slack distribution in the 3-D graphics accelerator [9], [25], for 200-MHz clock cycle

VII. CONCLUSION

This project studied the problem of grouping FFs for joint clocking by a common gater to yield maximal dynamic power savings. Although the problem was NP-hard, several practical algorithms to solve it and found several of them to be useful in a real design automation implementation. The solution was integrated in a practical design flow. In Future going to use carry select adder instead of carry skip adder. Experimental results of DSP cores, a network processor control block, and a 3-D graphics accelerator were presented, achieving 15%–20% total power reduction. The FF grouping problem also arised in MBFF, where distinct FFs were combined in one physical cell to share their internal clock drivers. It is interesting to consider the combination of data-driven gating with MBFF in an attempt to yield further power savings.

10.15662/ijareeie.2014.0312062

REFERENCES

[1] V. G. Oklobdzija, Digital System Clocking—High Performance and Low-Power Aspects. New York, NY, USA: Wiley, 2003.

[2] L. Benini, A. Bogliolo, and G. De Micheli, "A survey on design techniques for system-level dynamic power management *Syst.*, vol. 8, no. 3, pp. 299–316,Jun. 2000.

[3] R.Saranya, K.Radhika, Dr.S.nirmala, K.Priyameenkshi, "power optimization using clock gating techniques,"international journel of innovative research in computer and communication enginnering[IJIRCCE], vol 2,issue 11,Nov 2014.

[4] C. Chunhong, K. Changjun, and S. Majid, "Activity-sensitive clock tree construction for low power," in Proc. Int. Symp. Low Power Electron. Design, 2002, pp. 279–282.

[5] A. Farrahi, C. Chen, A. Srivastava, G. Tellez, and M. Sarrafzadeh, "Activity-driven clock design," IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol. 20, no. 6, pp. 705–714, Jun. 2001.

[6] W. Shen, Y. Cai, X. Hong, and J. Hu, "Activity and register placement aware gated clock network design," in Proc. Int. Symp. Phys. Design, 2008, pp. 182–189. 778 IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION (VLSI) SYSTEMS, VOL. 22, NO. 4, APRIL 2014

[7] M. Donno, E. Macii, and L. Mazzoni, "Power-aware clock tree planning," in Proc. Int. Symp. Phys. Design, 2004, pp. 138–147.

[8] Dr. S. Nirmala and R. Praveena, " Modified Partial Product Bypassing Multiplier for DSP Application," International Conference on Innovation in Electronics and Communication Engineering(ICIECE),pp. 128, Aug 2013.

[9] S. Wimer and I. Koren, "The Optimal fan-out of clock network for power minimization by adaptive gating," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 20, no. 10, pp. 1772–1780, Oct. 2012.

[10] Y.-T. Chang, C.-C. Hsu, M. P.-H. Lin, Y.-W. Tsai, and S.-F. Chen, "Post-placement power optimization with multi-bit flip-flops," in Proc. IEEE/ACM Int. Conf. Comput., Aided Design, Nov. 2010, pp. 218–223.

.