Grenze International Journal of Engineering and Technology, Jan Issue

Datapath Design and Power Optimization in High Speed Core

V.Anandi[1], M. Ramesh[2] and Suraj Ramesh[3]
[1]M S Ramaiah Institute of Technology/ECE Department, Bangalore, India
Email: anandi.v@msrit.edu
[2]Alliance University/School of Management, Bangalore, India
[3]Consultant/M3 Inc, Bangalore, India
Email: ramandsur@gmail.com, surajrameshm@gmail.com

***Abstract*—In today's high performance advanced VLSI circuit design including most semicustom designs adapt efficient utilization of circuit resources with better productivity offering a good layout density and high performances with optimum resources in submicron designs. The major focus of this work is to optimize the Functional Unit Block which is part of a processor in terms of timing, power and area. Datapath design methodology along with the enhancement of the existing design technology and optimization methods, is used to deliver a quality design and this work also proposes various logical optimization, RTL changes, placement, routing optimization techniques to meet the specifications in terms of power, timing. Datapath designing is a method where the logic synthesis of the design is done manually without any tool based, approach, to give more control over the design. The major focus of the work is to understand the functionality of the functional unit block, and implement the best suitable power optimization methodologies that include sizing sequential cells, Latch conversions, Redundant buffer removal, Clock gating and buffer merging to optimize the FUB in terms of timing and power to meet the specifications and also the quality. A power gain of 0.4% is claimed in this paper.**

***Index Terms*— datapath, synthesis, timing, clock tuning, Clock Gating, Buffer Insertion.**

I. INTRODUCTION

The continuous growth in the VLSI industry is ending up in delivering high performance products and at the same time it involves more design challenges. In the design of advanced microprocessors, different IP blocks are combined with the core and fabricated on a single chip. With this technology evolution single core microprocessors are replaced with multi-core processor, able to fit in the same area and deliver high performance. The entire core is divided into different clusters based on the concept of divide and conquer. The cluster division is based on functionality of the core. These includes memory, execution, instruction fetch and decode, integer processing units, floating point processing units, out of order instruction queuing in different clusters. The clusters are divided into sections and sections are branched out into different units and the leaf cell in hierarchy is called Functional Unit Blocks (FUBs). This paper proposes a semi-custom data path design synthesis implementation of the FUB and performs power optimization techniques to reduce the power consumption [1].

Grenze ID: 01.GIJET.7.1.507_3

II. Synthesis Methodologies

Designs can be synthesized using two different methodologies, firstly RTL to Layout Synthesis (RLS) which is a design automation tool based synthesis methodology. But automation tools synthesize the design without the knowledge of the placement of the cells which causes incorrect estimation of interconnect timing. Control over design is quite less. Secondly, complex designs can be synthesized manually by the designer using Datapath design to have more control over the design. The physical designing step of Place and Route is also done manually. It provides the flexibility for hierarchical design methodology.

A. Datapath Design Synthesis

In a full chip floorplan, the datapath designs are divided into dedicated FUBs and the designer has to implement the design manually and place the standard cells in a bit slice manner. The paper [3] discusses a methodology that modifies the traditional ASIC design flow to integrate both logic synthesis and physical design as a single flow and the main objective by doing so is to reduce cost function that depends on timing, routing and area constraints of the design. The FUB design follows the semi-custom design approach using standard cell libraries with the gate-level implementation of the given RTL code. It is comparatively easy at the RTL level to experiment with structural changes in design to save power and improve timing rather than doing it in backend or gate-level implementation.

III. Design Methodology

A. Design Specifications and RTL Coding

Every FUB design should undergo varied design milestones before each tape-in stepping. Each milestone has its own dedicated targets and during the last milestone, FUB should completely meet its quality and specifications. In this FUB design implementation 5 milestones are identified at circuit, layout (initial and final) and tape-in. FUB should undergo functional, timing, quality, noise and power reviews before passing any milestones. Design Specifications are decided according to the needs of the customer, after market research. The architectural specifications of the design are represented in the form of Register Transfer Level (RTL) abstraction. The RTL coded in Verilog code describes and infers the structural components and their connections. The front end stage deals with optimization of the code to reduce the logic for the block functionality. The input to the back end stage is the HDL code which is developed in the front-end stage. Fig.

1. shows the top-level flow design flow on the back end side.

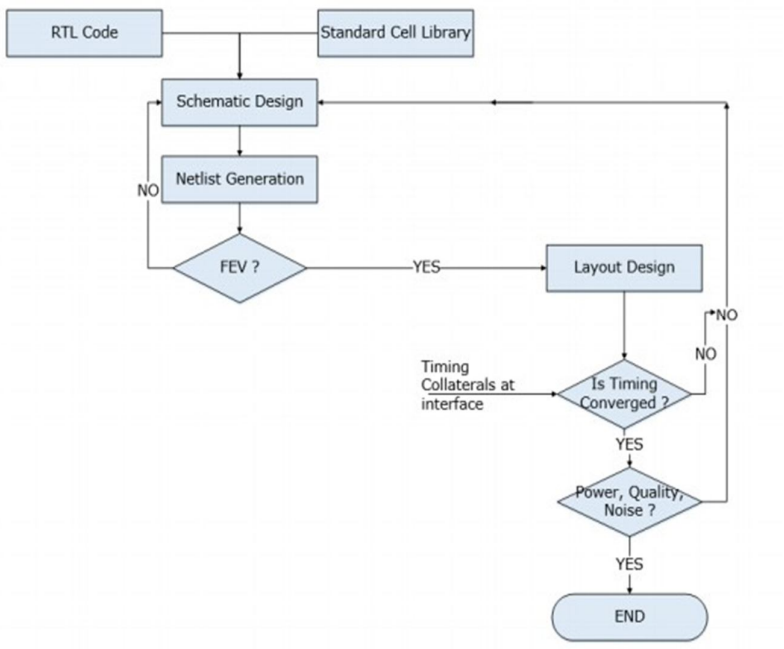

Figure 1. Backend Design Flow for Semicustom Design

IV. Logic Synthesis and Formal Equivalence Verification (FEV)

The output of logic synthesis is a synthesized and optimized schematic, block called a converged FUB. The main advantages with FUBs are reduced complexity in design, reusing of functional blocks, easiness in analyzing the timing, power, layout, noise. FEV checks the netlist generated by the FUB against the one extracted from the RTL code. It serves as a debugging platform and finds the logical bugs in the implemented design at early stages thereby avoiding silicon debugs which can be a drastic problem. Equivalence is faster as it makes use of mathematical proof rather than simulation FEV files from central database compile . sch files to .vs HDL file format. Also HDL files are converted to binary .exe and. exlif files compiling the IMP/SPEC side.

V. Floorplan and Placement

Floor planning is an estimation step where the area for the netlist, pin locations, global power grids, FUB power grids, Clock grids, and route guides are estimated. The floor plan decides the placement of logic blocks and control blocks in the initial stages. All the multiplexers are placed in the middle of the layout since the control logic needs to serve the control lines and all the control lines will be oriented towards left and right. The floor plan also decides the clock tree optimization [2]. Gates/cells in netlist are physically implemented and routed in this stage. For the allocated FUB the placement can be either pin-driven placement or signal-driven placement.

VI. Power Analysis and Estimation

The reduction in technology node enables more logic to fit in the same amount of area requiring higher operating frequency to match the expected performance. Also, fast switching requires faster standard cells which can support this high frequency. This increase in frequency proportionally increases the dynamic power consumption of the whole core. For data path FUBs around 13 tests are carried out to check the amount of power dissipated by the FUB for the given frequency and supply voltage. It gives the amount of power utilized by the netlist and then estimates the clock-switching power, power grids, and gated power. The various factors to be considered for power estimation are i. Activity Factor (AF)of a signal defined as the average number of transitions per clock cycle. Activity factor at any node gives the dynamic power consumed at that node. ii. Signal Probability (SP)of a signal is defined as the probability of its ON state. iii. Power Estimation Quality Metric depends on four factors namely Diff of AF, Driven Inputs, Average Validity, and Capacitance Quality. Driven inputs are defined as the percentage of inputs driven and average validity is defined as the percentage of nodes driven by the power estimation tool.

VII. Implemented Design Flow and Methodology

A. Formal Equivalence Verification

The first step after schematic implementation of the design from the given specified RTL is the FEV. FEV formally verifies Design X == Design Y, where design X from RTL and Y from an implemented schematic. Schematic is the outcome of the logic synthesis. FEV verifies the logical equivalency and detects the logical bugs in the early stage of the design leading to a reduction in silicon bugs. FEV works on a mathematical model based on which it compares the netlist generated from the synthesis and the RTL code giving very accurate results based on the mathematical proofs.

From the FEV flow diagram as shown in Fig.2, FEV verifies two models for the functional equivalence between the SPEC model from the RTL code The netlist containing the design instances, connectivity between inputs and outputs is extracted from the implemented design. In the further step, RTL code and schematic netlists are compiled into HDL file format and serve as input to Verify interface stage. If there exist any errors including mismatched cell names, mismatched block names, dangling pins, and nets without connectivity the extraction stage will fail and all the errors need to be fixed to proceed down further stages of the FEV. In VERIFY interface stage the FUB is considered as a block box and the input, output nets of both implemented and SPEC models are compared. If there is a mismatch of net names between the RTL net name and the implemented, then the verify interface stage of FEV will flag a failure. To clear such interface failures, the designer maintains a map file in the supported file format of the tool, which includes the mappings of the nets having different names in implemented design and RTL code. Debugging the interface failure is easier if all the sequential elements are mapped properly.

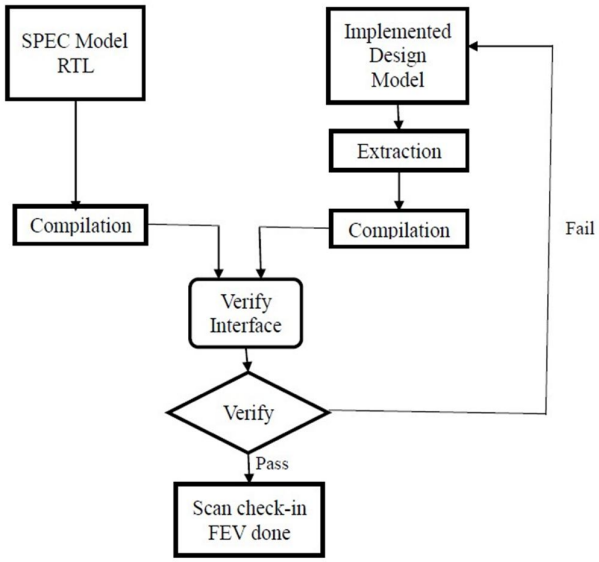

Figure 2. FV Flow diagram

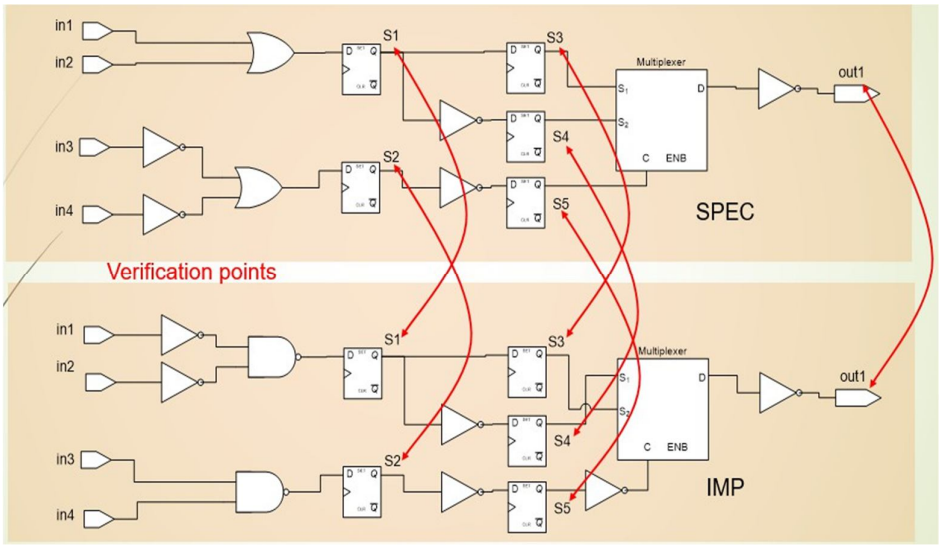

Figure 3. Equivalence verification between SPEC and Implemented Design

Fig.3 shows both the designs are divided into different cones and are based on the output of each sequential element either a latch or flip flop in the design. Vertex of each cone is known as a cut point or the verification point. In some cases wherein the design is complex FEV tool decides the cut points based on its mathematical model and also FEV gives flexibility to the designer to add cut points by adding mapping of those particular nets in the map file and import it to the FEV tool. FEV failures occur on output pins when the implemented design is not the same as the specified design that is the RTL code. From Fig 3 all the nodes s1, s2, s3, s4, s5 are all mapped and they provide an easy way to debug the faults if any.

VIII. Power Optimization Methods

Apart from leakage power, dynamic power is also a dominant power-consuming factor at the core level. Device dimensions, load capacitance, the transition period of the input signal will have an impact on the short circuit power. The dynamic capacitance is a parameter that is in the control of the designer and wherein the designer can play with the effective capacitance and activity factor (AF) to reduce the dynamic power. Optimizations are

based on methods to reduce capacitance or methods to reduce AF and capacitance using EDA tools. Some of the methods to reduce capacitance are discussed below.

A. Redundant Buffer Removal

Buffers might be added to the design to drive large capacitive loads and for long lengthy routing even though it is not logically required. Buffers would have to be added in the path to fix hold time violations where there is no setup time constraint. Sometimes drivers and receivers are placed close to each other, in which case adding a buffer will be redundant which adds up to power.

B. Large sequential cells

Sequential cells, with a clock signal are considered power-hungry cells. In the case where sequential circuits need to drive a very high load, then the size must be high enough as it results in very high power consumption as the clock signal has high activity factor of 1. This case can be resolved by downsizing the sequential cell and adding a buffer of to drive such a high load, with less capacitance at its output node will help in maintaining the desired amount of slope with a significant amount of power reduction. In Fig 4 a large-sized sequential of size 24 diffusion grid is converted to a small-sized sequential with the size 3 diffusion grid followed by an inverter of size 24 diffusion grid, which will be able to drive a huge load.

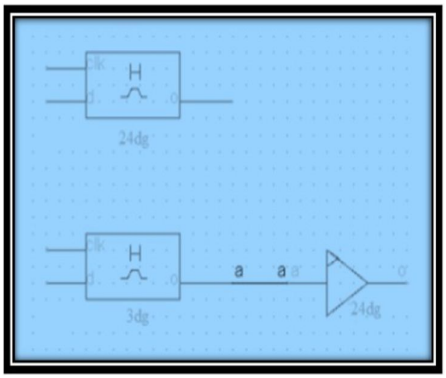

Fig 4. Large sequential

C. Dual/Quad Latch Conversion

This method converts two latches into a single dual latch as shown in Fig. 5 and also based on the availability, placement of the dual latches are further converted into quad latch as shown in Fig 5. By doing so the gate capacitance of the previous stage will be reduced and also in each latch the internal structure has a clocked inverter that drives a pass gate logic, saving an inverter which further reduces the gate capacitance of the previous stage.

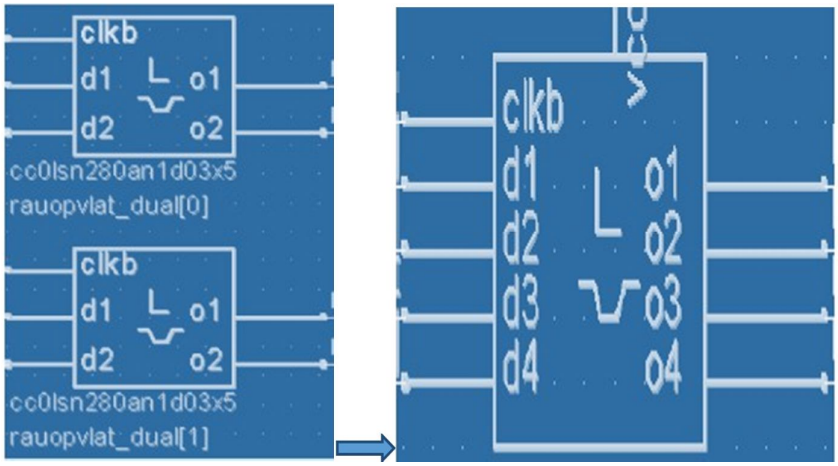

Figure 5. Latch conversion

IX. Methods to Reduce AF

A. Regional Clock Buffer (Rcb)/Local Clock Buffer (Lcb) Merge

At the core level, there exists a master clock and is re-distributed to each section through RCB and is further distributed to each FUB through the LCB and are mainly used for clock gating. Merging RCB/LCB can help in reducing the number of clock signals that do exist in the design. Two RCB's which are driving the corresponding LCBs within the FUB can be merged into a single RCB that will be able to drive all the LCBs. Another factor that needs to be considered is to verify if the enable signal for each RCB that needs merging is the same for both. Hence reducing the number of clock signals and clock elements will eventually reduce the AF and power consumption.

B. Multiple sequential cells connected to static cell

Static cells such as NAND, NOR, Complex Multiplier with a sequential at its input can be optimized to have a single sequential with inputs coming up from the static cell. Consider an example shown in Fig.6, where the data outputs of each sequential are connected to a static NAND gate. Design can be optimized by connecting the data inputs to the NAND gate and further the NAND output is given to a sequential. Hence one sequential count can be reduced, thereby reducing a clock signal. Overall the design AF will be reduced and will impact the power consumption.

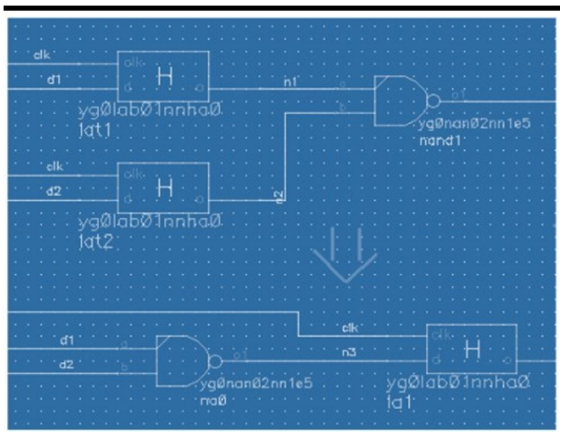

Figure 6. RCB merging

C. Static Cells With Inverted Inputs

Wide static gates with three inputs or more, in which at least two inputs are driven by inverter then such design can be converted to narrow gates. Consider the example shown in Fig 7 wherein for a three-input NAND gate two inputs are driven by inverters and it can be converted to NOR gate which drives the NAND.

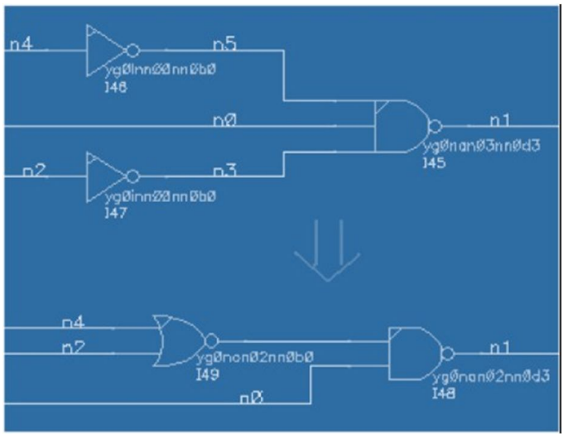

Figure 7. Static cells with inversion

X. RESULTS

The power optimization methods discussed are implemented in one of the FUBs of the execution cluster of the core. All the results are briefed up here.

A. Interface Mapping Verification Results

Interface mapping verification status for the FUB is given below Table I, where each input and output signal in the SPEC model is verified with the IMP model. Such signals missing mapping in the IMP model but present in the SPEC model, are mapped again to get appropriate results in the FEV.

TABLE I. INTERFACE MAPPING

Interface mapping	Specification	Implemented
Mapped inputs	932/932 (100%)	1005/1005 (100%)
Mapped outputs	916/916 (100%)	1372/ 1372 (100%)
Mapped states	956/960 (99.9%)	1460 /1472 (99%)
Mapped power nodes	1/1 (100%)	1/1 (100%)

B. Equivalence Verification Results

This stage of FEV verifies the logical equivalency based on the cones that are formed between SPEC and IMP model. The result is shown in below Table II.

TABLE II. EQUIVALENCE VERIFICATION

Unit	Total pairs	Equivalent pairs
FUB	3253	3253

C. Power Results

Various power optimization techniques are described based on capacitance reduction and AF reduction. EDA tool calculates the power results by running the FUB design on the top power-hungry application and gives the results on comparing with the budgets defined. Table III gives the power results for the FUB, where the FUB design is run on a top power-hungry application like Photoshop using an EDA tool, and dynamic capacitance is listed before timing convergence and also after timing, power convergence. The results are mentioned with a 0.4% power gain.

TABLE III. POWER RESULTS

Test application	FUB Cdyn	FUB Final Cdyn gain	FUB Cdyn before timing convergence	FUB Cdyn after timing convergence
Photoshop	9.3%	0.4%	8.9%	10.6%

XI. CONCLUSIONS

In this paper, the digital circuits were designed following the back-end design flow. All the optimization techniques were discussed in detail and results were tabulated. Some new power optimization experiments were discussed in this study which can be performed in VLSI circuits to save dynamic power at high frequency up to 4.8GHz. The paths that have to be optimized for power must have good margins. The power optimisation shows 0.4% improvement in saving adopting the techniques mentioned in the paper. The power gain numbers shown in this study are achieved only from the paths with positive setup/hold margins. The multi-bit Flip flop technique to merge two flip flops can be used to merge multiple latches, and going further dual latches or dual flip flops can be further be converted to quad latches/FFs to reduce the overall load on the clock.

REFERENCES

[1] Linumon Thomas and Kiran V, "Timing Convergence Techniques in Digital VLSI Designs," IEEE International Conference on Power, Control, Signals and Instrumentation Engineering (ICPCSI-2017).

[2] Mohamed Khalil Hani and Nasir Shaikh-Husin, "Simultaneous Routing and Buffer Insertion Algorithm for Interconnect Delay Optimization in VLSI Layout Design," International Conference on Microelectronics, 2008.

[3] Ali Dasdan and Ivan Hom, "Handling Inverted Temperature Dependence Static Timing Analysis," ACM Transactions on Design Automation of Electronic Systems, vol. 11, no. 2, pp. 306-324, April 2006.