Shift Left Trends for Design Convergence in SOC: An EDA Perspective

Vivek Bhardwaj
Intel Corporation
Penang, Malaysia

ABSTRACT
Design convergence of System on Chips has become a major problem to solve in semiconductor industry. The advent of deep sub-micron nanometer scale transistor design, fabrication technologies and accompanying scale in design sizes have posed a major challenge regarding their implementation. For many years Electronic design automation companies have been an active partner of the design and semiconductor Fab ecosystem and have been devising new ways in methodologies and automation to solve multiple challenges the semiconductor design has been facing since its inception. Today's complex designs and their blocks need innovative and out of the box approach in solving the convergence problem in terms of timing, power and area and additionally the runtime as well since the scale of designs is also increasing along with the complexity. Through this paper, we are going to look at the background and motivation of doing many of these shift left strategies that the design tool vendors have deployed over the years to solve the design convergence problem. After going through this paper that covers multiple aspects of the physical implementation and signoff process, the reader would get a better appreciation of why the tools are having converged methodologies and would develop a better sense of appreciation towards the software tools and its various artifacts and flows

General Terms
Chip Design flow, computer aided design, system-on-chip design, performance, power, software tools, turnaround time

Keywords
ASIC, SOC, VLSI CMOS integrated circuits, Moore's law, Electronic design automation, physical design, timing, STA, Quality of results

1. INTRODUCTION
Shift Left is a software industry origin word where traditional software testing is shifted to the early development cycle for better quality and results[1].

In generic context shift left means shifting the processes that happen later in the cycle or flow to the left. EDA (Electronic Design automation) is a specialized branch of system software design tools that help in architecture, design, implementation and testing of very large scale integrated circuits(VLSI) or even complex system on chips(SoC). System on chips is full scale explosion of the VLSI CMOS(complimentary Metal-oxide-semiconductor) technology wherein relatively smaller circuits or sub-systems like memory, processors(both graphic and computational), displays and inputs/outputs are brought together on a single semiconductor die and fabricated and integrated as a single system. With the overall complexity of the systems increasing(Figure1), the entire design convergence process is not possible without the scale and automation of the design tools provided by the EDA companies like synopsys, cadence and mentor graphics, to name a few. In context of design implementation and EDA, shift left does not mean shifting the process cycle or effects of validation cycles which are anyways very important. Shifting left in EDA/design context imply that overall design cycle and effective time to market is shortened by accounting for later flow steps early in the cycle since it maximizes gains and minimizes overall turnaround time(by removing bottlenecks early and reaching convergence faster). For example, if previously a design takes three months to close, now with the left shifted software it could be closed in two months with even higher benchmarks of design industry. This paper outlines many such techniques used in the EDA tools. This paper mainly covers physical design(also called as physical implementation, construction or simply implementation) and signoff stage of a digital design methodology flow. The paper studies the shift left techniques done inter- domain(implementation for example) or intra-domain(synthesis-implementation or implementation-signoff). Each step is described in brief detail to give some background to uninitiated and then the motivation as to why shift left was introduced in that part of the domain or within domains.

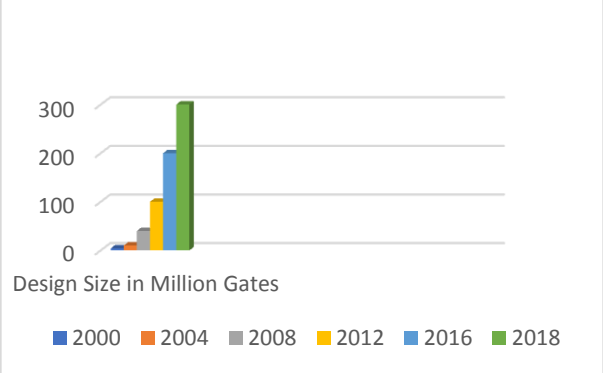

Fig 1: Design scalability trends(Moore's law and new nodes)

2. INTRA DOMAIN TRENDS
In this section we would look at the trends within a single domain.

2.1 Timing awareness pan flow
A physical design process consists of many steps and sub-steps. We are not going to go in details of each step here but suffice to say that each step/sub-step is an algorithmically computationally intensive task with its own set of goals. For example, placement's job is to place the standard cells of each timing path in such a way that distance between each logic element is minimized and satisfies the various physical constraints of a chip(E.g a partition boundary surrounding standard cells, placement and routing blockages, low power domain requirements etc). Similarly, the goal of routing is to connect these physically placed elements in shortest way possible thereby meeting the detailed routing constraints. One key trend that is progressively being done is to make all distinct steps

done for physical design either timing aware or timing driven. The difference in the two methodologies being that the first does not disturb the slack graph through its operation, while the latter actually puts timing slack as one of its edges in its weight driven graph solver. Depending on the criticality and stage of the step, either operation is being called in all individual sub-steps. This shift results in a progressive timing closure . At the time when timing the design through STA(Static Timing Analysis) was decoupled from the key physical steps, designers often found surprises and found it very difficult to close timing progressively , to an extent where designs often remain unclosed. Besides timing, power is also being made aware but to a much lesser extent compared to timing as power recovery remains a post process and an architecture driven operation. Initially when this trend started, placement and then routing were major steps but gradually this awareness is made available in all steps of implementation and even extending to the late sign off stage.

There is no specific data around this as this is a generic enhancement.

2.2 Around Clock Tree

Enhancements in the clock tree handling in the whole flow is a perfect example of a very early shift left happening in the physical design process which was made possible by the combined result of designer's feedback along with fundamental changes in the way clock tree is handled. The clock tree is created right after initial placement of standard cells and accompanying physical optimization process. The clock tree creation is the process where clock is made physically available at all the clock end points(also called sinks) by creating complex structures of buffers and wires. The goal of any clock tree process is to minimize the delay and skew which is an essential when it comes to minimizing the area and power of the chip while meeting the timing requirements of all the paths. The clock tree structure contains various elements like clock buffers/inverters as well as clock gates(architectural and inferred) which serves as design elements to reduce power and pipeline the logic through a select line. After every clock tree creation(also called as clock synthesis) another round of physical optimization is done to fine grain the data paths without touching the clock paths which are already frozen after the clock tree creation. Any perturbation in these paths at this stage could incur a heavy penalty on convergence since these are high fanout logic cones and not isolated structures.

The software tools made two very fundamental changes to this above process that was based on designers inputs as well as recognizing inherently how the EDA tools worked. The degrees of freedom available for physical optimization to a tool before the creation of clock tree is several orders of magnitude more than after the clock tree is fully laid out. Based on this principle, the first important shift left was co-optimizing the clock and data paths during the clock tree creation. What this enabled for a design was optimizing the data paths at the same time the clock tree was constructed(as opposed to leaving this for later which was the regular flow.). Consequently, the tools had more degrees of freedom now to fix the data paths while ensuring clock targets are met at all times. Doing this also enabled them to leverage a concept of 'useful skew'(Figure 2). This concept ensures creation of a clock tree in such a way that it purposely introduces a skew(in other words didn't try to meet the skew target) by foreseeing the slacks of the data paths the clocks aims to serve. This not only saves area and power but also ensured unnecessary work the tool did to meet skew and hence the tools could focus on real target(i.e. data path timing, chip power and area).

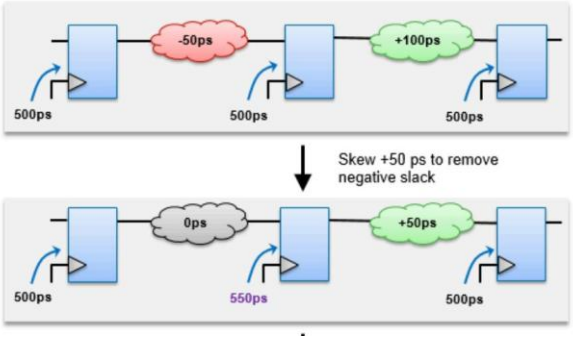

Fig 2: Useful skew concept

The second important shift left the tools did was to make the clock tree architectural gates placement aware . They also did basic clock tree build up stages like clustering [2] to ensure that the placement sees basic clock tree structure before doing the standard cell placement. This was done primarily because it was realized that generally the QoR is better achieved if the architectural clock gates are placed closer to standard cell locations and that could only be done with some kind of tree construction. The other reason was that in congested designs , if the location of clock path elements if approximately known, then later effects like detail routes could be better predicted leading to more accurate placement. Note that this enablement resulted in more runtime as well as some extra steps are to be done here unlike the co-optimization discussed as the first trend. Very recently, tools have been offering dynamic Updation of latencies based on the integrated clock gates(ICG) placement(Figure 3).

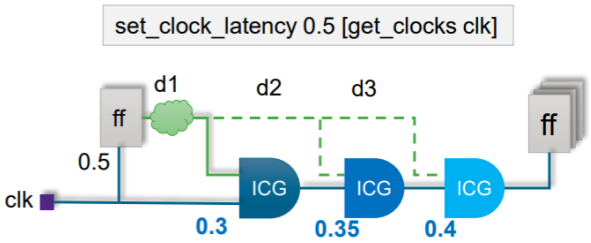

Fig 3: Clock Gates dynamic placement

Several other small sub-steps have been done to refine the flow further, however the above two major shifts resulted in better chip divergence compared to the regular flow.

Changes in clock tree handling and useful skew brings about 10-15% improvement in performance in final timing.

2.3 Modeling post-route in pre-route

A major source of design non-convergence is the mismatch between the pre route and post route RC (resistance and capacitance) scaling factors. The design implementation process is so designed to progress in stages rather than implementing all steps at once. To that effect, the runtime, details and complexity of routing also depends on the stage of the flow. The timing state of the design depends as first order on the resistance and capacitance of the wires during routing(in short also called parasitics or RC factors). RC factors are the resistive-capacitive factors for an interconnecting wire. These factors in turn directly depend on the quality of extraction engine used and the kind of routing done. The extraction engine typically deployed is in tune with the type of routing done. The finer and detailed the quality of routing, more detailed would be the extraction engine used. So the value of parasitics or the RC factors also differ according to stage of the

design. Having understood this correlation, it is widely observed that there is significant gap between the pre-route and pos-route parasitics. The route here refers to the final detailed router and the not the initial coarse router. At times, the difference is so much that huge divergence is observed in timing numbers which can be completely attributed to the RC values of the wire interconnects. To mitigate these issues, designers need to often plug these detailed factors earlier into the flow to model the effect of what is going to happen later in the flow. This is a way of shifting left your post detailed route design state back into pre-detailed route to achieve a better convergence and physical optimization of the design. EDA tools have thus provided ways of generating these values automatically through special artifacts and have provided defined methodologies to achieve the same.

2.4 Layer promotion shift

Advanced nodes have become increasingly difficult to close for a variety of reasons. One of the major reasons for this is ever increasing metal stack that is used for actual chip fabrication. New metal layers are being added with each process node to accommodate the increasing size of the designs. Along with the number, the electrical characteristics of the stack is also increasing in complexity. There is significant difference observed in geometries less than or equal to 10nm(nanometer) when it comes to the metal stack. The stack is so designed in these geometries that the bottom layers are compact and spatially co-located compared to top layers that are thicker and have wider pitch. Combined with the trend of increasing line resistance with shrinking geometries, it is observed that the upper layers have significantly lower resistance and capacitances relative to the lower layers(Figure 4). As a result top layers are more suited to high performance and timing critical paths compared to bottom ones[3]. In geometries greater than 10nm, the tools used to opportunistically route the critical nets on upper layers depending on congestion and timing and that too at only later stages of the flow and typically after clock tree generation . This is so to preserve the upper layers for power nets and clock nets that have higher switching and wider impact on the chip. Now, in even lower geometries(<10nm), as designs have become increasingly difficult to achieve closure, the tools have to make the shift left of continuously tuning all wires in such a way so as promote and demote nets on different layers vertically through the stack and continuously through all stages of the flow to achieve optimal timing for all types of nets(clock, data, power). These new strategies have been resulting in a smoother timing convergence as the design flow is executed, in this new shrinking world.

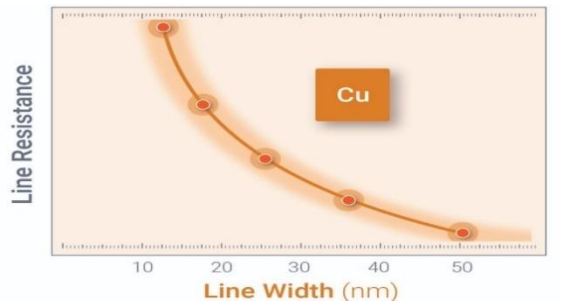

Fig 4: Geometry- Resistance trend

Changes in layer promotion methodology brings about 5-10% improvement in performance in final timing.

2.5 Predictive hierarchical time budgeting

Time Budgeting is a process of budgeting time to the partition boundaries[4]. In a full hierarchical flow, the system is partitioned to many sub-systems in such a way that each sub-system can be independently implemented with its own set of independent inputs. To that effect, time budgeting is a process of carving out a timing constraint file for each partition. The process is heuristic in nature rather than algorithmic. It depends on the current state of the timing graph. The decisions made based in the current state of the graph might not align with the future state of the graph which would include routing and physical optimization. Additionally, the clock needs to be independently budgeted and is based on the pre clock tree latencies[5][6][8][9]. The clock budgeting as such would be mismatched with the future state as well where a full block clock tree is implemented. Hence, in order to make time budgeting process more accurate and predictive, one can do either a full graph virtual optimization and/or a trimmed graph physical optimization[7]. This is way of shifting left the optimization cycle upfront to attain faster closure in a full hierarchical flow. Similarly, tools deploy various abstraction techniques as well for virtual prototyping which is early estimation of the design, in order to make timely and early decisions with respect to either the floorplan or the design inputs.

3. INTER DOMAIN TRENDS

In this section we will look at some of the emerging trends that affect the interactions between various domain of the chip design flow.

3.1 RTL-Design merge

RTL(Register Transfer Level) is nothing but an abstraction of a digital circuit consisting of combinational and sequential elements. RTL is specified in a standard language which is also known as HDL(hardware description language) like Verilog . RTL has always been designed at a functional level where the concept of physical placement , floorplan, connectivity and routing are non-existent and non-characterized. Generally, RTL designers write RTL for their sub-systems by providing optimal logic for functionality and let downstream synthesis and physical design tools manage the performance and power. With the advent of low power RTL based techniques there have been quite a few power improvements or power squeezes that have happened at the RTL level. However, with the increase in nanometer complexity, it is becoming increasing difficult either to converge at existing targets or increase the PPA(standard industry nomenclature that stands for power, performance and area) targets in light of increasing competitiveness in semiconductor design space. It has been increasingly felt by designers that only changes at architectural level introduced at the RTL stage can only drastically improve the PPA. Hence, now there is an extreme shift left being introduced by the EDA companies like synopsis(RTL-Architect®) who are introducing a concept of shifting physical design all the way to RTL design space. This means that an RTL designer would now effectively see the physical impact and timing of its RTL code instantaneously. The potential of this flow is that it does close to accurate physical synthesis, floorplanning, placement and routing that would have only happened if the RTL would have fully been taken through the full physical design flow. The tools should then point the designers to bad timing or congestion and then correlate it back to the RTL. The RTL designer could then make a decision to modify the RTL for additional pipelining or control the number of stages or any other additional architectural improvements including any power related changes like clock gating. Apart from the RTL changes, RTL

hierarchical regrouping and other transforms could also be done after viewing the module hierarchical floorplanning and connectivity analysis. This would, if found to be accurate and correlated with the full physical implementation process, this shift can phenomenally improve the PPA and design convergence and also save substantial amount of turnaround time since in typical process flow, the late RTL changes come in late after the physical implementation and have be fed back to the physical process through late ECO's . Absorption and convergence of these late RTL ECO's also take time that could potentially be saved to improve the overall time to market.

Data needs to be collected for this trend since it is very new and yet to be fully adopted by the designers.

3.2 Synthesis-physical design interlock

Synthesis is the step of using RTL and converting it into a gate level netlist which represents real world boolean digital logic. There are various steps associated with this process. A RTL is mapped and optimized to various gate level library cells in an artifact called as a netlist which contains a gate level logical interconnection to form the circuits as desired by the SOC or IP designer. Traditionally the synthesis process was isolated from the physical design process . However, with recent design scales and convergence becoming a bottleneck, the EDA companies have increasingly integrated the two distinct processes(cadence iSpatial® technology or Synopsys Fusion ompiler®). They have brought back more place and route awareness into the synthesis process thereby resulting in a much better and optimized netlist. When synthesizer optimizes the netlist it anticipates(or in other words shift left) the process of early placement/routing and the associated physical optimization/wire delays along with it. This makes this process more tightly tuned to the reality of the floorplan and the physical design process, thereby resulting in a more optimal netlist that is optimized for power , performance and the area metrices. Hence, by merging these two algorithmically very distinct process that work on different inputs and vectors, a unique shift left and shift right(since information flows both ways in this flow) has been brought about in the design industry. Substantial QoR(Quality of results like timing, power consumption and chip area) gains are claimed by both tools over the traditional way. An additional benefit is the decrease in turnaround time since some redundancies have been eliminated by the merge.

Shifting left of physical awareness in synthesis results in about 3-6% improvement in performance and 2-3% improvement in power. There is about 10% improvement in overall turnaround time.

3.3 ECO implementation-signoff loop

ECO(Engineering Change Order) is the process of implementing the netlist changes that are coming out of signoff environment timing/power optimizations or any other behavioral changes to the design. For this paper, we are going to look at the changes coming from timing/power perspective. The ECO process is a software automation improvement over the manual ECO process that was hitherto done by the designers. The process of cleaning up the last remaining timing and Design rule violations like max_transition and max_capactiance is a very tedious process that requires very careful hand placement and editing of wires in such a way so as not to disturb the existing portion of the chip that has already been closed from timing and other perspective. Not very long ago, much of this whole ECO process has been automated by many companies resulting in much faster convergence since it resolves most of the violations automatically and leave the hand editing to select few paths. Along with timing and design rule violations, dynamic and leakage power is also taken in consideration by these ECO tools. The first attempt at automation by the EDA tools is done in a most classically thought manner of separating out the sign off and the implementation tools. The traditional model of this flow is such that once the designer enters the signoff environment, the existing violations carried over from implementation and the newly introduced violations(due to more accurate signoff settings) are 'logically' fixed. However, the actual physical implementation of these logical changes is done in the physical design tool itself. There are multiple shift-left trends observed in this space over the past few years which are attributed to the gaps designers saw in this flow and long cycles.

3.3.1 Physically-aware ECO

This was the first trend that was executed. In the original model described above, there were lot of inherent issues. The whole process resulted in lots of mismatches when actually implemented compared to what was predicted in the signoff environment and the changes generated based on those predictions. These mismatches were both on optimistic and pessimistic side but mostly they were optimistic- i.e the signoff predicted a better numbers but when actually implemented, due to a variety of factors in a physical world, resulted in pessimistic unconverged timing and power. Hence this resulted in multiple iterations or back and forth between the two environments to result in a converged block or SoC. The left side of figure 5 shows this cyclical behavior and the need to switch between different environments to perform those operations. Hence, the tool vendors provided a new concept of physically aware ECO. The idea behind this concept is bringing over the physical information of the design to the signoff environment and doing the logical ECO with awareness of the physical environment of the cell being resized or the net being buffered. Many physical artifacts like blockages, congestion information, multi-height cells, power domain info etc were brought over to ensure few surprises and hence fewer iterations in the whole flow. Be aware , that for the final implementation one had to go back to the implementation tool. So the cyclical behavior still remained, but the number of cycles were reduced. This supposedly resulted in better convergence but was still not perfect as there were limits to the awareness.

3.3.2 Implementation within signoff

To overcome the above noted constraint, the vendors came up with the idea of creating an implementation environment within the signoff tool itself. The signoff tool would do the logical calculation based on its own algorithms that would result in optimal ECO which are, as expected, physical aware and would then execute new commands that are physical implementation commands remaining in the same environment. This is the latest trend(as shown in the right box of figure 5) and promises to ensure minimum iterations between signoff and implementation as the logical changes and physical changes co-exist in the same environment and hence any refinements or perturbations could be contained easily. The real impact of this technology is still to be evaluated by the designers yet. This trend offers more than just iteration reduction(which means convergence); it also implies ease of use for the designer since switching between tools is automatically eliminated .

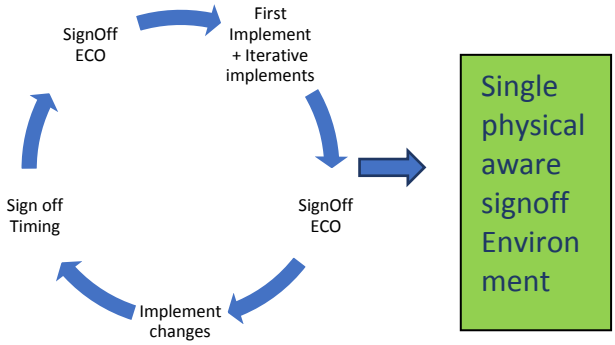

Fig 5: Shifting trends in ECO

3.3.3 Signoff within implementation

Somewhere between the latest model and the oldest model is sandwiched this model which is the shifting of the core signoff ECO algorithms in the physical design process itself. Within the implementation environment, typically after detailed routing and final physical optimization of the design, these more detailed ECO logical optimizations are performed. Since we are already in the implementation environment(as opposed to signoff environment as in last para), the implementation is much easier to perform with more reliable results. We should note that these results have their limitations since they are not done under sign off environment which, as stated earlier, is significantly different and more accurate than the physical design environment. So, we would still need to do a sign off timing and ECO if required. However, by shifting the ECO process to implementation, the design convergence cycle and iterations are significantly shortened resulting in faster tape out schedules.

In this section, we saw the history of ECO process and innovative trends over a period of years. There are many solutions and work models available and for a given project schedule, typically a combination of the trends and solutions are needed to converge and no one single solution is enough. However, the innovations and strive towards a fully automated converged solution with minimum overheads is appreciated by the SoC designers and they are quick top adapt to these solutions.

New ECO trends have resulted in about 15% runtime savings with about 4% fixing rate improvement

3.4 IR-drop aware placement

IR analysis(Current increase and resistance analysis) of a design is a sign off reliability check where voltage drop is measured at specific tap points in the design to ensure that the supply voltage is reliable and does not cause any malfunctioning and/or timing issues in the real silicon. One of the main causes of IR drop is high current demand at a specific location in the chip due to simultaneous switching of aggressor nets thus creating instantaneous voltage drops also called as dynamic power drops. The measurement of this effect is typically done in signoff stage and by that time it could be too late to fix or the fix could be very expensive in terms of late architectural changes that have to be introduced to mitigate the issue. Hence it is now shifted left in the incremental placement steps done at the physical design stage after clock tree or detailed routing. The idea is to reduce the number of hotspots by spreading the cells to various locations based on switching characteristics and congestion in the design. The padding of these cells is thus needed to ensure that hotspots around them is not there. Various engines like power analysis, timing analysis and physical cell spreader engine are invoked to implement this functionality.

Another variant of the same concept(Figure 6) is to introduce aggressive clock based scheduling changes to meet the aggressive power demand by the simultaneous switching of sequential end points creating localized and global peak current demands. By forcing skews based on current demands, one could mitigate the IR drop effects.

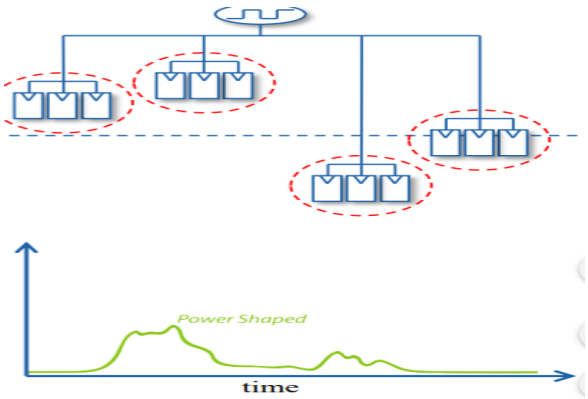

Fig 6. schedule based peak current

4. CONCLUSION

Through this paper we have seen and understood some of the effects of the design scalability and process advancements on convergence and how EDA software tools have made paradigm shifts in their approach to handle design approach right from RTL stage, leading to dramatic improvements in SoC convergence. New engineers can get a good perspective of the trends in EDA industry over a period of many years when these engineer were not around. Experienced engineers can benefit from the work done by the EDA companies and the designers to achieve better performance and/or design closures in their own designs and methodologies. Through this paper, academia would also benefit by observing the trends seen in the design and automation industry and could relate to and apply the learnings back to new research in the EDA domain.

5. ACKNOWLEDGMENTS

The author would like to acknowledge various publicly available user guides and reference manuals from synopsis and cadence design systems, the two leading EDA companies. Some general concept figures are drawn from synopsys guides for reuse and illustration purpose only.

Synopsys® reference manuals- www.synopsys.com

Cadence® reference manuals- www.cadence.com

6. REFERENCES

[1] https://en.wikipedia.org/wiki/Shift-left_testing

[2] A. D. Mehta, Yao-Ping Chen, N. Menezes, D. F. Wong and L. T. Pilegg. 1997. Clustering and load balancing for buffered clock tree synthesis. Proceedings International Conference on Computer Design VLSI in Computers and Processors, Austin, TX, USA, pp. 217-223, doi: 10.1109/ICCD.1997.628871.

[3] https://www.eetimes.com/layer-aware-optimization/

[4] V. Bhardwaj, O. Levitsky, D. Gupta. 2015. Machine readable products for single pass parallel hierarchical timing closure of integrated circuit designs. US patent 9165098.

[5] V. Bhardwaj, O. Levitsky, D. Gupta. 2013. Flow methodology for single pass parallel hierarchical timing closure of integrated circuit designs. US patent 8365113.

[6] V. Bhardwaj, O. Levitsky, D. Gupta. 2013. Systems for single pass parallel hierarchical timing closure of integrated circuit designs. US patent 8539402.

[7] Vivek Bhardwaj. 2020. Hierarchical Methodology Approach to SOC Design- A comprehensive look.

[8] V. Bhardwaj, O. Levitsky, D. Gupta. 2015. Methods for single pass parallel hierarchical timing closure of integrated circuit designs. US patent 8935642.

[9] V. Bhardwaj, D. Seropian, O. Levitsky. 2013. User interface for timing budget analysis of integrated circuit designs. US patent 8504978.