2020 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION EUROPE

Clock Controller Unit Design Metrics

Area, Power, Software Flexibility and Congestion Impacts at System Level

Michele Chilla, Senior Engineer, QT Technologies Ireland Limited, (mchilla@qti.qualcomm.com)

Leonardo Gobbi, Engineer, QT Technologies Ireland Limited, (lgobbi@qti.qualcomm.com)

***Abstract* — The aim of this paper is to highlight how same complex Multi-Clock Digital IP System is impacted by different RTL design strategies of its dedicated Clock Control Unit (CCU). Area, Power, Software Flexibility and Routing Congestion have been analyzed and compared. Digital Front-End stage has entirely covered from design to logic synthesis by using DC (Design Compiler), DV (Design Vision) tools and PC (Power Compiler). Outcome results of this paper provide metrics and guidelines to CCU digital designers. The quality of being novel is the fact that current literature shows how clocks impact the design, but all analysis has been made with assumptions and mathematical models have been used to shape design features. This work, instead, picks a real example of edge-cutting Digital Sub-System with +300k flops and all design features such as bus infrastructure linking complex internal cores. RTL also includes DFT logic used to test all the faults of the chip on silicon.**

Keywords:* *RTL Design, Clock Control Unit, Clock Domain, Logic Synthesis, Static Timing Analysis, Internal and Switching Power Analysis, Design Compiler Graphical, Place and Route.

Before Reading…

The work presented here is based on a real analysis made on an edge technology, Qualcomm® Snapdragon™ 855+ Mobile Platform. As exact design features and absolute numbers cannot be shared in the public domain. The analysis results are presented in percentage scale.

Table 1: Terms and Acronyms

CPU	Central-Processing-Units
RTL	Register Transfer Logic
IP	Intellectual Property
SS	Digital Sub-System
SoC	System-On-Chips
CCU	Clock Control Unit – digital block dedicated for generating all clocks of a SS
SW	Software: Firmware used to control logic of digital cores.
DFS	Clock Dynamic-Frequency-Scaling: low power technique
STA	Static Timing Analysis
DFT	Design for Testing
PLL	Phase-Lock Loop

I. INTRODUCTION

A. Overview

Most modern CPUs are sequential circuits where a clock signal is used to synchronize all parts of the chip and trigger the circuit elements to generate outputs. Generally, clock-tree distribution is spread across the entire SoC die and it significantly contributes to power consumption and heat dissipation.

The exponential increase in signal-processing capability leads to SoC which can support multiple functions at the same time while still enabling long-lasting battery for mobile devices. This need has forced digital architectures to migrate from a Single-Clocked System into a Multi-Clocked Sub-Systems (SS) structure, where each SS has its own functional purpose such us Audio, Sensor, Navigation, Video, etc (*Figure 1*).

The Multi-Clocked SS approach requires the development of dedicated CCUs used to generate all internal clocks. Clock Driver Firmware controls the unit managing external clocks sources (e.g. external ring oscillators) while also programming internal clock-mux's, digital PLLs or digital dividers to achieve specific target frequencies depending on the power and functional use-case.

This new approach allows designers to mitigate power consumption by using low-power techniques such as Power-Gating (dynamically shouting down an entire SS when not needed), Clock-Gating and Clock Dynamic Frequency Scaling (DFS) based on specific functional use-cases.

The number of the internal generated clocks can easily go up to +80 for a single high complex SS. Clocks can be also grouped in *Clock Domains* based on the SS design. All clocks belonging to the same clock domain are synchronous with each-other and the logic synthesis tool will make sure that the synchronicity is kept while building the chip on silicon (*Figure 2*).

Also, the multiple clock-domain approach benefits the Back-End stage as timing constraints will be more relaxed giving engineers more flexibility in placing cells and routing the clock tree.

Figure 1: Example of a SoC Die split in Multi-Clocked complex digital Sub-System. Sub-Systems have dedicated CCU

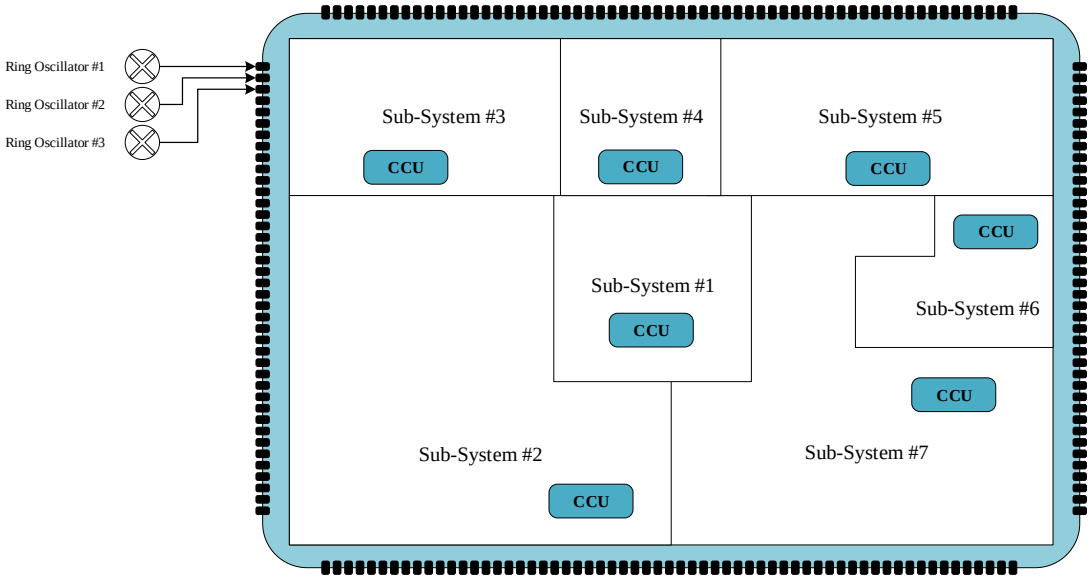
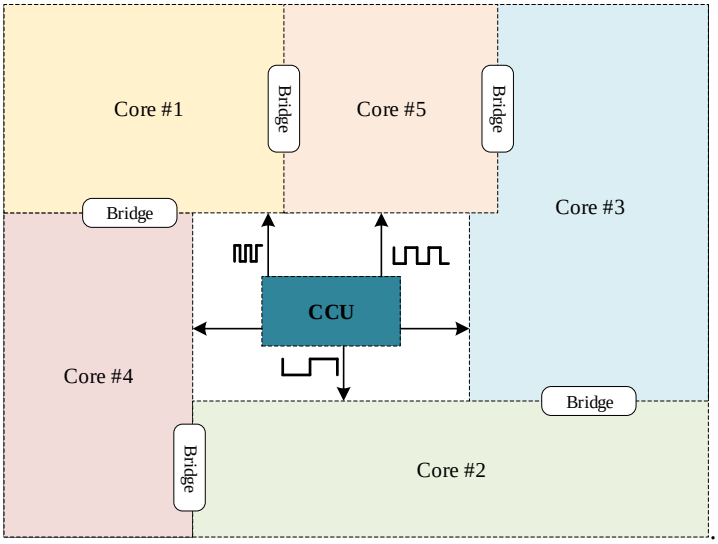
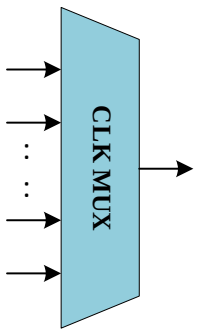
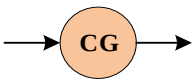


Figure 2: Example of clock distribution within SS cores. Bridges are needed for synchronization.



B. CCU Basic Blocks

General basic blocks of a CCU unit are:

	**Clock Multiplexer** The clock multiplexer, shortened to "CLK-MUX", is a combinational logic circuit designed to switch one of several clock inputs source with different frequency values (e.g. External Ring-Oscillators, Digital PLLs) through to a single common output clock line by the application of a control signal. It differentiates from a well-known data digital multiplexer as its logic prevents glitch propagation when switching between clock inputs. This element is generally controlled by SW and it is widely used to perform DFS and give high SW flexibility.
	**Phase-Locked Loop** Control system that generates an output signal whose phase is related to the phase of an input signal.
	**Digital Clock Divider Cell** Clock divider cell converts frequency of input clock into a lower frequency. Complex divider cells have the ability of dynamically set divider value based on SW configuration.
	**Clock Gating Cell** Clock gating is a common technique for reducing clock power by shutting off the clock to modules by a clock enable signal. Enable signal is controlled either from software or hardware.

C. Sub-System Design Under Test

The Digital Sub-System under test is a real Multi-Clock IP Core used on latest Qualcomm chip with its own CCU clock

II. CCU DESIGN STRATEGIES

The main target for the CCU design changes is to reduce both dynamic and leakage power consumption. This can be achieved by:

- increasing DFS though SW
- grouping clocks in asynchronous branches.
- reducing Buf/Inv Insertion to balance clock tree

CCU functionality must not change (i.e. number of clocks generated and frequency target for each clock).

Three RTL versions with different levels of optimization have been developed for the same CCU.

- **Low-Opt:** All clocks synchronous and generated from same C-MUX. This implementation provides poor DFS as all clock branches are derived from a single C-MUX. Furthermore, having all clocks synchronous to each other is a tight timing requirement at STA and Back-End side.
- **Mid-Opt:** All clocks are generated from a single source but are grouped in different Clock Domains. Even though SW flexibility is still poor (same as Low-Opt design), the design allows more relaxed timing constraints at the Back-End stage which may optimize routing congestion and buffer insertion.
- **High-Opt:** All clocks are generated from dedicated sources (C-MUXs) and grouped in different clock domains. This solution guarantees high DFS and relaxed timing constraints. In following design all C-MUXs share same input.

Table 2: Low/Mid/High-Opt CCU Designs

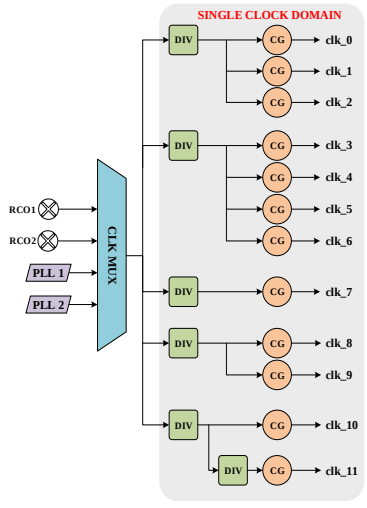
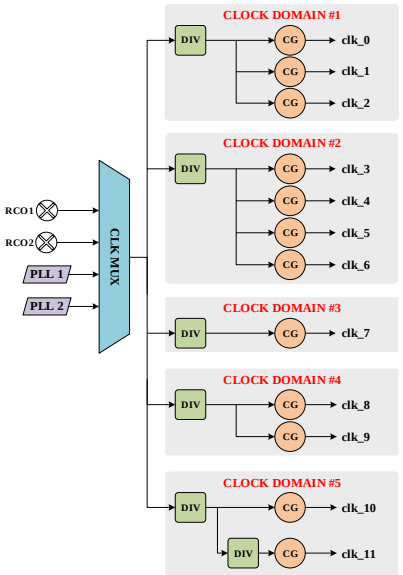
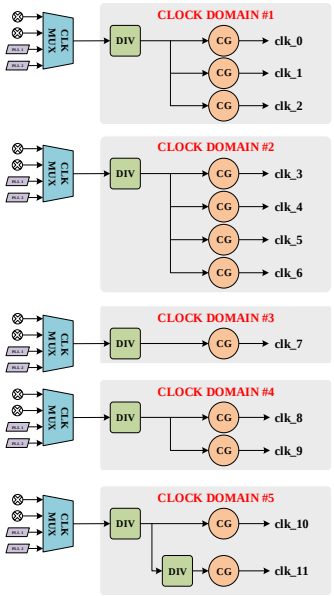
Low-Opt	Mid-Opt	High-Opt
		

Table 3: Cons and Prons of different CCU strategies

Low-Opt CCU	Mid-Opt CCU	High-Opt CCU
STATIC POWER		
✗ Tight timing constraints: clocks forced to be all synchronous each other.	✓ Relaxed timing constraints: clock branches asynchronous.	✓ Relaxed timing constraints: clock branches asynchronous.
✗ High Fan-Out congestions: same CMUX drives all clock branches.	✗ High Fan-Out congestions: same CMUX drives all clock branches.	✓ Fan-Out congestions reduced: dedicated C-MUX per clock branch.
✗ High Buf/Inv insertion.	▬ Moderate Buf/Inv insertion.	✓ Reduce Buf/Inv insertion.
DYNAMIC POWER		
✗ Poor DFS	✗ Poor DFS	✓ High DFS

III. RESULTS

A. *Outcome Results Methodology*

All data has been acquired performing the same analysis steps on the same Sub-System RTL while implementing the three version of the CCU outlined in *Table 2*.

Data comparison will be presented in percentage scale as numbers strictly depend on technology node.

Area: Total cell area has been gathered by the Design Compiler (DC) tool. A sub-section of the area data such as Combo and Noncombination and Buf/Inv insertion has been also investigated.

Leakage and Dynamic Power: Leakage and Dynamic Power has been estimated by using existing *Power Compiler (PC)*, an internal tool of DC.

- Leakage Power Calculation**:** Power Compiler analysis computes the total leakage power of a design by summing the leakage power of design's library used:

$$\boldsymbol{P_{TotalLeakage}} = \sum_{\forall \boldsymbol{cells(i)}} \boldsymbol{P_{CellLeakage_i}}$$

$\boldsymbol{P_{TotalLeakage}}$ = Total leakage power dissipation of the design.
$\boldsymbol{P_{CellLeakage_i}}$ = Leakage power dissipation of each cell *i*.

- Dynamic Power Calculation: Dynamic power is the power when the circuit is acting. The equation is:

$$\boldsymbol{P_{Dynamic}} = \boldsymbol{P_{Switching}} + \boldsymbol{P_{Internal}}$$

PC calculates both factors as follow.

$$\boldsymbol{P_{Switching}} = \frac{\boldsymbol{Vdd^2}}{\boldsymbol{2}} * \sum_{\forall \boldsymbol{nets(i)}} (\boldsymbol{C_{Load_i}} * \boldsymbol{T\,R_i})$$

$\boldsymbol{Vdd}$ = Supply voltage
$\boldsymbol{TR_i}$ = Toggle rate of net *i*, transition per second
$\boldsymbol{C_{Load}}$ = Total capacitive load of net *i*. It includes parasitic capacitance, gate capacitance and drain capacitance of all the pins connected to the net *i*. Power compiler obtains C_{Load} from the wire load model of the net and from the logic library information of the gates connected to the same net.

$$P_{Internal} = \sum_{i=A,B} E_{i\rightarrow Z} * PW_i * T\,R_z$$

$$E_{i\rightarrow Z} = [C_{Load}, Trans_i]$$

$$\sum_{i=A\rightarrow B} PW_i = 1$$

PW = Input Path Weight. Power Compiler calculate latter based on the input toogle rates, transition times and functionality of the cell.
E_Z = Internal Energy for the output Z as a function of input transitions, output load and voltage
$T\,R_z$ = Toggle rate of output pin *z*, transitions per second.
$Trans_i$ = Transition time of input *i*.

Routing Congestion: The Design Vision (DV) tool is to extract a topographical view of all clock branches across the entire SS down to 6 levels of connections. The topographical mode allows a user to accurately predict post-layout timing, area and power at the physical level.

Software Flexibility (DFS): Deducted from design.

B. Result Data Comparison

Mid-Opt VS Low-Opt:

The synthesis outcomes show that having multiple clock domains reduces the *Total Power* by 16.38%. Main contribution to this is due to a reduction of -20.92% of *Switching Power* in Mid-Opt respect to Low-Opt. *Internal Power* has instead increased by +5.92% as extra RTL logic has been added in order to build a multi-domain structure. Impact on *Leakage Power* is negligible (*Figure 4*)

Area analysis has seen a reduction of *Buf/Inv Insertion* by -1.31% (*Figure 3*).

Routing Congestion has significantly improved as the layout tool has more flexibility in routing clock-tree and shuffling flops cells around the floorplan (*Figure 5*).

No relevant differences are observed in terms of *Software Flexibility* (*Table 2*).

High-Opt VS Mid-Opt:

The synthesis outcome shows that higher SW flexibility implies an increase of +21.61% of *Total Power* consumption. This is due mainly to an increase of +23.9% in *Switching Power*. The impact on *Leakage Power* is negligible (*Figure 4*).

From an area perspective, both *Combo* and *Buf/Inv Insertion* have decreased by -0.61% and -0.12% respectively (*Figure 3*).

Routing Congestion has significantly improved as the Layout tool now has more flexibility in routing clock-tree, shuffling flops cells around the floorplan and moving clock cells to near-by sub-cores which they clock (*Figure 5*).

Software Flexibility is maximum (*Table 2*).

High-Opt VS Low-Opt:

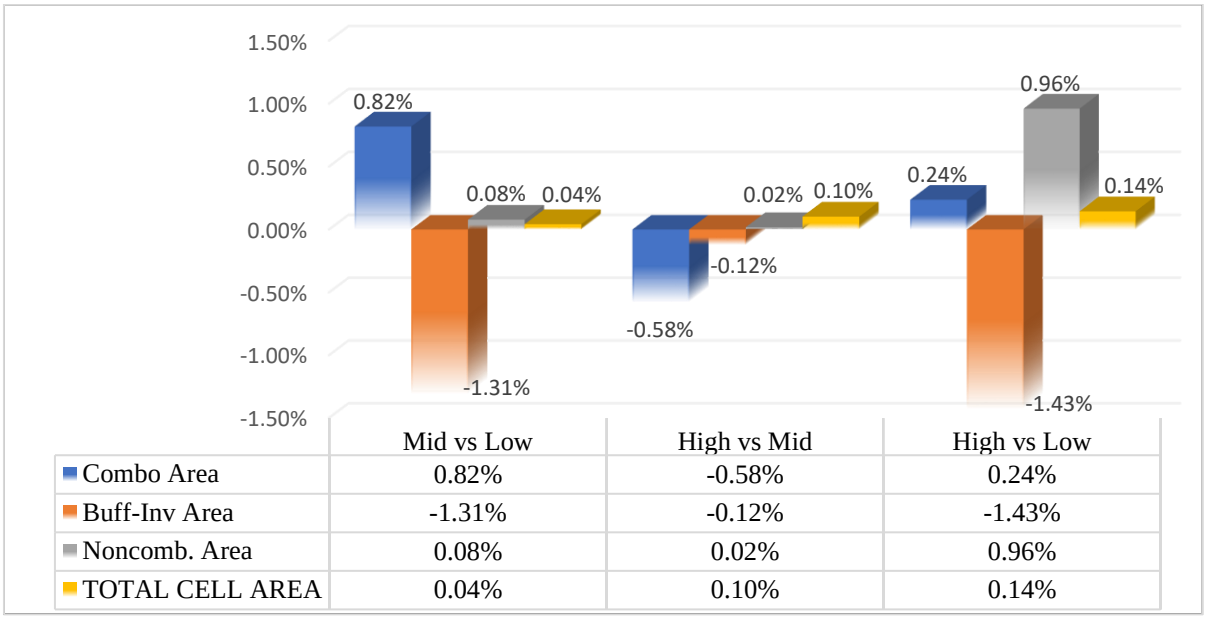
At Sub-System Level, the advantages in terms of splitting the big clock domain prevails on the disadvantages of having changed RTL to increase Software flexibility. High-Opt shows a *Total Power* increase of +1.68% relative to Low-Opt solution. *Switching Power* decreases by -2.02%. Even in this case *Leakage Power* is minimal (*Figure 4*).

Buv/Inv Insertion has decreased by -1.43% due to the multi-domain approach used (*Figure 3*).

Routing Congestion has significantly improved (*Figure 5*).

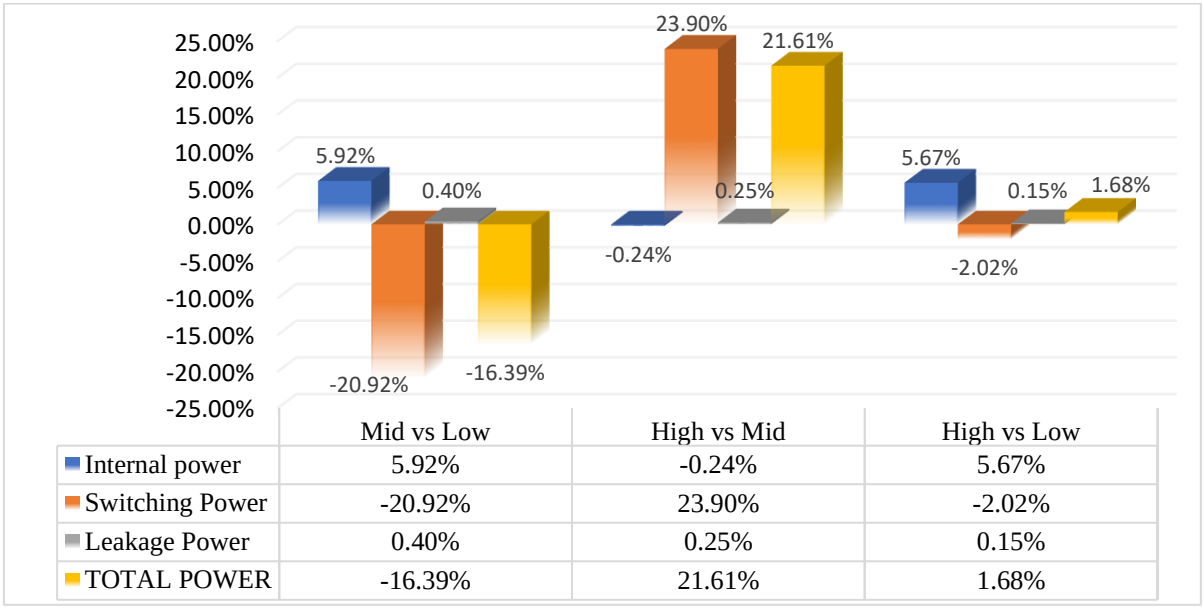
Maximum *SW flexibility* guaranteed in High-Opt solution (*Table 2*).

Figure 3 : Area Data Comparison



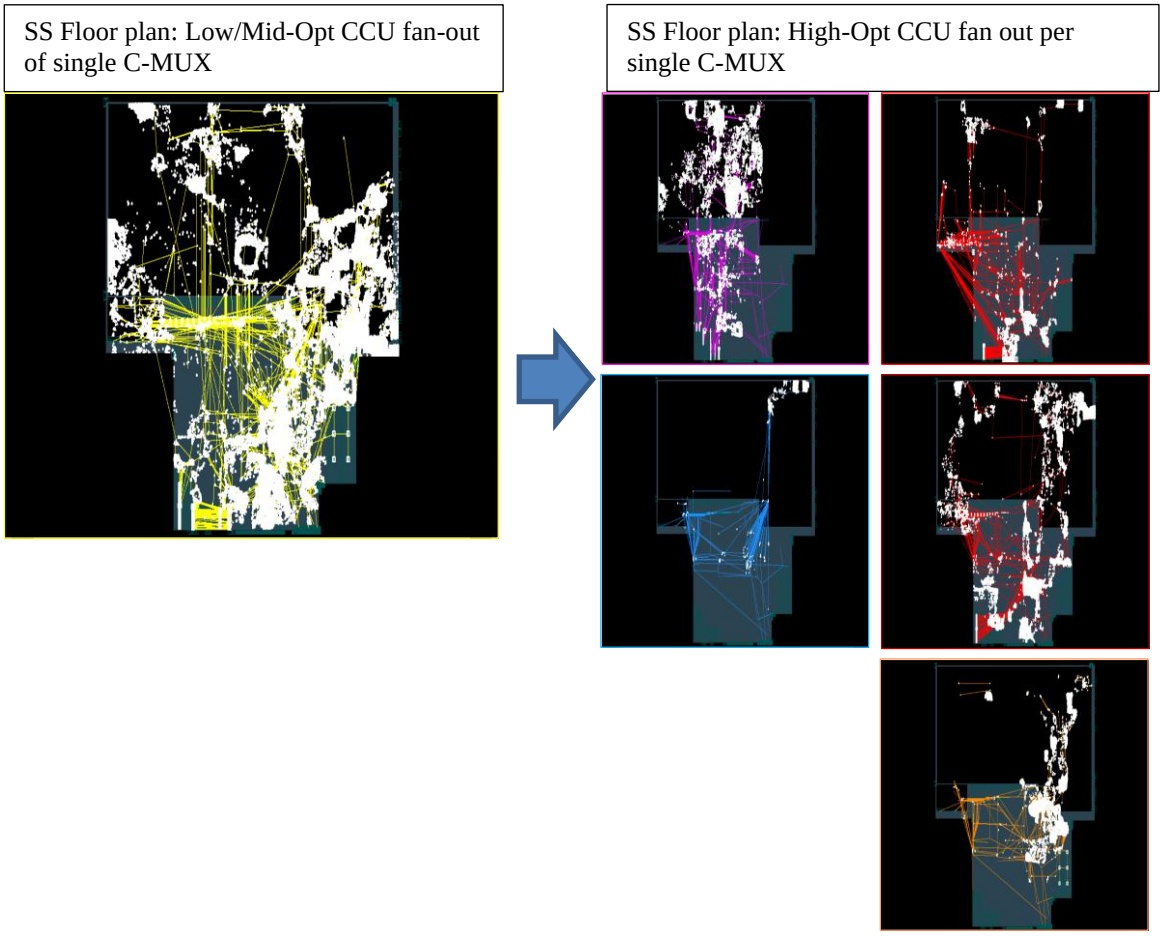
	Mid vs Low	High vs Mid	High vs Low
Combo Area	0.82%	-0.58%	0.24%
Buff-Inv Area	-1.31%	-0.12%	-1.43%
Noncomb. Area	0.08%	0.02%	0.96%
TOTAL CELL AREA	0.04%	0.10%	0.14%

Figure 4: Power Data Comparison



	Mid vs Low	High vs Mid	High vs Low
Internal power	5.92%	-0.24%	5.67%
Switching Power	-20.92%	23.90%	-2.02%
Leakage Power	0.40%	0.25%	0.15%
TOTAL POWER	-16.39%	21.61%	1.68%

Figure 5: C-MUX flylines differences between High/Mid-Opt and Low-Opt CCU at SS Level – 6 Level of Fan-Out



IV. CONCLUSION

The main results observed are that the splitting of one single clock domain (*Low-Opt*) into many of them (*Mid-Opt* & *High-Opt*) leads to a *Switching Power* saving.

High *Software flexibility* (High-Opt) is responsible for an increase in Switching Power while the *Internal Power* effect is negligible.

From the area perspective, interesting results have been experienced in the reduction of *Buf-Inv Insertion* thanks to a multi-clock domain approach (*Mid-Opt* & *High-Opt*). *Total Area* variation is minimal as CCU design changes are negligible respect to the entire design.

Routing Congestions analysis shows that the multi-clock domain approach gives Back-end stage higher flexibility in building the clock-tree.

As mentioned in the introduction, complex Digital Systems are now made up of multiple SS with a dedicated CCU (*Figure 1*). Hence, if we apply the same design strategy to all CCUs, the benefit at SoC level will be more significant.

A. Future Work

All power results presented are based on a static analysis where pre-defined models in DC have been used to model wires and toggling behaviour of all cells.

As known, dynamic power depends directly on the *Switching Activity* (α) of the wires and this information can be accurate obtained from real digital simulation(s) of the design.

$$\boldsymbol{P_{Dynamic} = \propto * Vdd^2 * f * C_{Eff}}$$

Simulators can write out files with waveforms (generally FSDB format) and the *Switching Activity* can be readily extracted from them. *Switching Activity* is a critical element of power analysis and power optimization. The quality of the switching activity information has a direct effect on the quality of results and for power analysis, it provides a key piece of the power equation. For optimization, it helps to identify which optimizations are most useful.

Also, to have a better understanding on leakage and dynamic power consumption due to clock tree distribution and timing constraints, power analysis can be also completed on the final physical-design netlist.

REFERENCES

[1] Chadi Al Khatib, Claire Aupetit, Cyril Chevalier, Chouki Aktouf, A generic clock controller for low power systems: Experimentation on an AXI bus, Univ. Grenobles Alpes, Grenoble, France

[2] Wang Bing, Peng Rui-hua, Wang Qin, A Design Flow for Clock Controller Hard-macro Generation, Department of computer science, Shanghai Jiao Tong University, Shanghai, China.

[3] Design Compiler User Guide, Version X-2005.09, 2005.

[4] Saurabh Verma, Ashima S.Dabare, Understanding clock domain crossing issues, EETimes-India, 2007.

[5] J. Lanmoureux and S. Wilton, FPGA clock network architecture: flexibility vs. area and power, Proceedings of the 2006 ACM/SIGDA 14th international symposium on Field- Programmable Logic and Applications, 2006, pp. 101-108