CLOCK DISTRIBUTION NETWORK BUILDING ALGORITHM FOR MULTIPLE IPs IN SYSTEM ON A CHIP

By

Tan Tze Liang

A Dissertation submitted for partial fulfillment of the requirement for the degree of Master of Science (Microelectronic Engineering)

August 2017

Acknowledgement

First of all, I would like to express my gratitude to all parties who have provided me with valuable inputs and suggestions while working on this project. I have gained a lot of knowledge and experience while working on this project that will be invaluable for my career. With this project, I was able to implement my ideas and suggestions that are previously not proven. Therefore, I would like to use this opportunity to show my gratitude to everyone who assisted me while working on this project.

Firstly, I would like to thank Universiti Sains Malaysia and School of Electrical and Electronics Engineering for giving me a platform to implement and prove my ideas I have while working on this project. By working on this project, I have been able to prove the feasibility of my idea and also future enhancements that may be needed in order to fine tune the process.

Next, I would like to thank my supervisor, Dr. Bakhtiar Affendi bin Rosdi for giving me the attention and guidance needed to complete this project. Over the course of doing this project, he had given me a lot of advices and guidances that have provided me with the correct directions for the project. Without his valuable guidances and advices, completing this project would be much more difficult for me.

Also, I would like to thank Intel Microelectronics for providing me with the tools and resources needed to implement this project. With the tools and resources provided, I was able to work and implement the ideas I had to work on this project. Without the resource provided, I will be facing issues in securing resources and the necessary tools to work on my project.

Besides that, I would also like to thank Mr. Wong Soon Kin for providing me with ideas and suggestions that could help me in achieving the objectives in this project. His suggestions and guidance have provided me with the ideas on the overall flows that I have needed to work on this project. Without his help, I would struggle in understanding the overall flows for working on this project.

Last but not least, I would like to give my thanks to my family and friends who have given me their support over the course of doing this project. Their constant encouragement gives me the support I have needed to overcome the obstacles I have encountered. Without their support, I would not be able to comple this project successfully.

Table of Contents

List of Tables

List of Figures

List of Abbreviation

CDN	Clock Distribution Network
CTS	Clock Tree Synthesis
EDA	Electronic Design Automation
HDL	Hardware Description Language
IC	Integrated Circuit
IP	Intellectual Property
PLL	Phase Locked Loop
PnR	Place and Route
RTL	Register Transfer Level
SoC	System on a chip
VLSI	Very Large Scale Integration

ALGORITMA PEMBINAAN RANGKAIAN PENGEDARAN JAM KE PELBAGAI IP UNTUK SISTEM DALAM CIP

Abstrak

Projek ini mencadangkan satu algoritma pembinaan rangkaian jam yang mengedarkan jam ke semua blok yang mengandungi pelbagai IP dalam SoC menggunakan saluran di antara blok. Cara konvensional membina global CDN memerlukan campur tangan manual yang mengurangkan kecekapan membina global CDN dalam SoC. Projek ini telah mencadangkan algoritma pengautomasian proses pembinaan global CDN yang memberikan rangkaian CDN seimbang secara keseluruhan yang tidak dicapai cara konvensional. Walaupun terdapat beberapa teknik pembinaan struktur CDN berbeza yang memudahkan proses pembinaan global CDN, teknik-teknik ini mempunyai kekurangan yang agak ketara apabila dilaksanakan. Algoritma yang dicadangkan ini hanya memerlukan kekangan dan masukan seperti masa latensi dalam blok sebelum algoritma dilaksanakan untuk membina global CDN. Apabila global CDN siap dibina, kualiti global CDN seperti ralatan masa akan diperiksa secara automatik oleh algoritma ini. Kajian keberkesanan algoritma ini telah dibuat menggunakan dua pelan SoC yang berbeza. Selain itu, tiga jam yang berbeza telah dibina dalam setiap pelan SoC untuk mengurangkan ralat ketika menguji algoritma ini. Masa diperlukan untuk membina global CDN serta bilangan dan kawasan penampan masa digunakan telah diambil untuk menilai kecekapan algoritma. Keputusan yang didapati menunjukkan pengurangan sebanyak 50% dalam masa pembinaan global CDN serta pengurangan sebanyak 5% dalam bilangan dan kawasan diperlukan oleh penampan masa. Pengurangan ini telah menunjukkan algoritma yang dicadangkan projek ini mempunyai kecekapan tinggi dalam keupayaan mereka bentuk global CDN berbanding dengan cara konvensional.

CLOCK DISTRIBUTION NETWORK BUILDING ALGORITHM FOR MULTIPLE IPs IN SYSTEM ON A CHIP

Abstract

In this project, an algorithm is proposed and developed to build the global CDN that is used to distribute the clocks to all partitions in the SoC using the channels available between partitions. The conventional method of building the global CDN involves manual interventions which decrease the global CDN building efficiency and increase the overall SoC design cycle. To solve this issue, an algorithm is proposed to automate the global CDN building process and at the same time obtain a balanced overall CDN not achieved by the conventional method. Other researches have proposed different CDN structures to simplify the design process but the proposals often sacrifice placement resources to achieve this. The algorithm first collects the partition clock latency numbers and other constraints needed as setup. When the setup is done, the global CDN is build and routed. The algorithm checks for clock skew and scenic routing issues before proceeding to shield the global CDN to prevent cross-talk issues. The algorithm is done when a final checking on the clock skew is done. The algorithm is tested on two different floorplans with varying size and available channels using three different clocks for each floorplan to ensure the accuracy of the algorithm. Finally, the global CDN build using the algorithm is evaluated based on the time needed to build the global CDN and the clock buffer numbers and areas used. The algorithm is shown to be able to reduce 50% of the global CDN design cycle and save 5% of clock buffer numbers and areas. The improvement achieved by the algorithm in this project shows the efficiency in designing the global CDN improved tremendously compared to conventional method.

Chapter 1

Introduction

1.1 Introduction

In today's technology world, more and more increasing complex intellectual properties (IPs) are being integrated into system on a chip (SoC) to enable a single chip which is capable of performing a myriad of different tasks and functions. As the modern day SoCs are required to perform multiple tasks and functions in the smallest possible package, IPs are designed separately and independantly before being integrated together. Although this process potentially increases the complexity of the SoC when integrating the different IPs, the IC design process will still able to save a tremendous amount of time and effort because these IPs can be designed in parallel instead of designing a single huge entity that incorporates every needed function. Designing a huge single entity is extremely time and resource consuming for the IC design process.

In the beginning of a typical IC design flow, the partitions are designed first using a typical hardware description language (HDL) such as VHDL or Verilog which describes the circuit to be created for the partition as the basis of the partition register transfer level (RTL). With the created RTL, the synthesis process will commence to turn the RTL codes into netlists which are the equivalent connections and standard cell representations of RTL in preparation for the backend process sometime through sacrificing accuracy to obtain better area optimized netlists (Soares, et al., 2015). The netlist will then be used in the backend process for the place and route (PnR) flow as the "blueprint" for building the physical design of the chip. In the general PnR flow, the flow will be divided into a few smaller stages with each handling different aspects of building the physical design of the

SoC. These general stages are the floorplanning, placement, clock tree synthesis (CTS) and routing. In the floorplanning stage, the available area and power grid design to be implemented for the physical design is determined. This is sometimes done through using algorithms that needed minimal inputs such as the supply voltage information (Sengupta, et al., 2011). A standard power grid will be built and then proceed to the placement stage to place the standard cells defined in the netlist through timing-driven placement or routability-driven placement (Markov, et al., 2015) depending on the applications and the constraints. Sometimes, before the placement of all cells, important cells may be pre-placed to prevent IR drop issues (Soman, et al., 2011). After the placement of the standard cells, the PnR flow will proceed to the CTS stage. In this stage, clock trees will be build to evenly distribute the clock with minimal clock skew. When the CTS stage is done, the PnR flow will then proceed to the route stage to route all the remaining open connections with minimal routing lengths for each connections through methods such as "rip and re-routing" method (Udgirkar & Indumathi, 2016; Xu, et al., 2009) or "decomposition of multiple-pin net" method (Udgirkar & Indumathi, 2016; Gao, et al., 2008). When all of the stages in the PnR flow is done, the entire SoC design will be ready to be manufactured in the foundry.

The CTS stage is one of the important stages in the entire SoC design. This is because during this stage, the clock for the partitions within the SoC is being build. As the control signal for the synchronous design used in modern SoC, an efficient partition clock tree is important to achieve a balanced clock skew in the overall clock distribution network (CDN). The CDN for the SoC consists of the global CDN which are housed in channel between the partitions and partition clock tree housed within the partitions. Compared to conventional IC design which are not restricted by partition boundaries, modern SoCs tends to restrict the design of the CDN due to the presence of different partitions which varies in shapes and sizes (Ramanathan & Shin, 1989). This leads to the division of the CDN into the global CDN and partition clock tree. During the execution of CTS stage in the PnR flow, the partition clock tree will be designed and build. On the other hand, the global CDN is build entirely through using a different process. The overall CDN will only be considered done when the global CDN connecting all partition clock trees have been built entirely. The design in both global CDN and partition clock tree needs to complement each other to ensure the entire CDN achieves minimal clock skew.

As mentioned in the title of this dissertation, this dissertation will attempt to propose an algorithm that could automatically handle the design and building of a balanced global CDN. While there have been many reserches that focuses on the partition's clock tree in the SoC, there are not a lot of researches that tried develop algorithms that could help in building the global CDN that distributes the clock across partitions in the SoC. As the design for modern SoCs becomes more complex, the quality of the overall CDN becomes more important and will directly affect timing convergence for partitions during PnR. The conventional method of designing the global CDN requires much manual input from the designer and often time consuming. The dissertation will discuss and analyze the requirements and the constraints that is required by the algorithm and the effectiveness of the algorithm in terms of time saving and the quality of the global CDN. Figure 1.1 shows the schematic of the location of the global CDN in the overall CDN and figure 1.2 shows the channel which will house the global CDN.

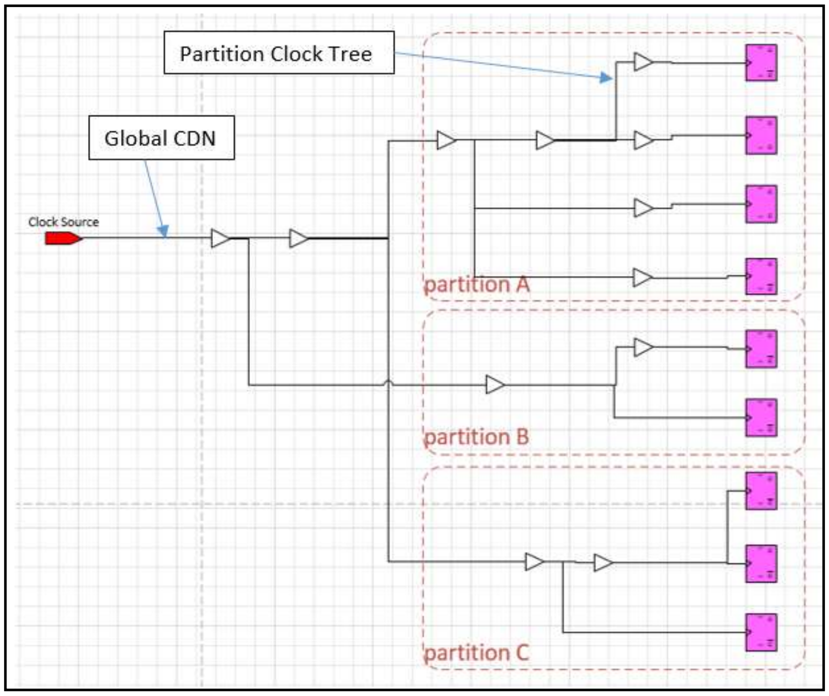

Figure 1.1: Global CDN in the overall CDN for SoC

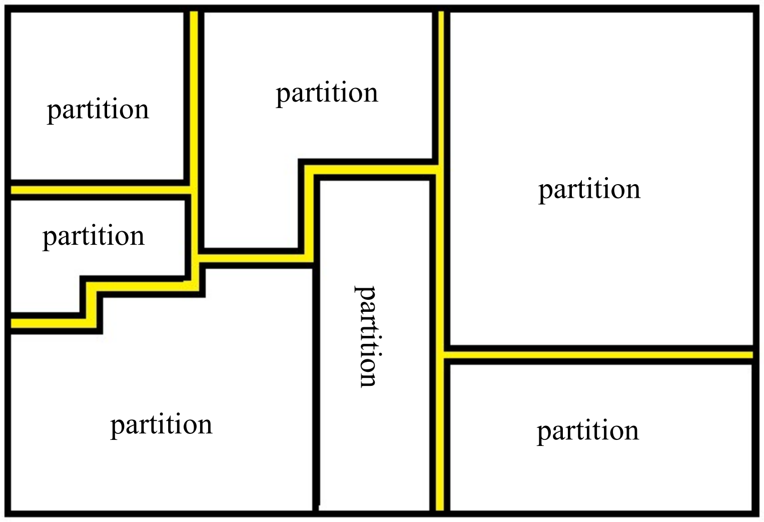

Figure 1.2: Example floorplan with channels (highlighted in yellow) that houses the global CDN

1.2 Problem Statements

With more and more complex partitions being crammed into a SoC, the CDN will have to be properly planned to make sure the timing paths to the different partitions are able to converge properly (Gupta & Kulkarni, 2006). An improperly designed CDN will create clock skew issues between partitions. Therefore, building an efficient global CDN is important to help in achieving lesser clock skew in the overall CDN. Clock skews are undesirable because it would lead to timing convergence issue for the SoC. To solve the issue caused by clock skew, the SoC would have to insert more delay buffers to fix the timing violations (Sitik, et al., 2016) or reduce the clock frequency for the SoC (Remil & Jayasanthi, 2015; Parthibhan, et al., 2012). Both methods have their own consequences. For the buffer insertion method, a higher power consumption will be observed (Sitik, et al., 2016) while the reduced frequency method will reduce the performance of the SoC (Remil & Jayasanthi, 2015).

One of the issues on the current method of designing the global CDN is that it takes significant time and manual effort from the designer to maximize the usage of the allocated channel between partitions (Gupta & Kulkarni, 2006). The reason for this is because the channel are often constrained and limited. The global CDN built using by existing method focuses on distributing the clock to all partitions as fast as possible without considering the partition clock tree. The balancing of the overall CDN would have to be done separately within the partitions. This means the partitions may need to redo the clock tree synthesis (CTS) to obtain balanced overall CDN after obtaining the data from the global CDN so that the clock skew between partitions are minimal, increasing the design cycle of the SoC significantly. Some researches tried to address this issue by proposing methods such as the divide and conquer technique (Gupta & Kulkarni, 2006) which divide and build the global CDN in smaller sections so that the clock can be distributed in a balanced manner before reaching the partitions. However, this method proposed is not suitable for designing the global CDN because it requires large areas from the channel to build the global CDN using the bottoms-up approach.

Another issue that is seen in the current method of building the global CDN is the inefficient use clock buffers across partitions. Due to the fact that clock have the highest power consumption in a SoC where at least one-third to half of the power is consumed by the CDN (Esmaeili & Al-Kahlili, 2013; Paliwal et al., 2013), controlling the power consumption for clocks becomes important. While there are researches that shows that varying the supply voltage of the CDN (Kulkarni & Khandekar, 2012) or implement a differential current-mode clock distribution (Islam, et al., 2015) is effective in controlling the power consumption, efficient clock buffer usage on the overall CDN is also able provide significant power savings. This is because the amount of clock buffers used directly affects the power consumption of the clock network (Chen & Chen, 2006). However, current method of building the global CDN does not take account of the partition clock tree seen in different partitions. There are no sharing of clock buffers when each individual partition clock tree are rerunning CTS to obtain the balanced overall CDN. This causes the SoC to have a high number of clock buffers in the overall design that will affect the power consumption of the SoC.

As a result of the two issues seen in the current method of building the global CDN, a new method of building the global CDN is needed to address the issues mentioned above. This is because as the SoC design becomes more complex, the designer for the global CDN may by overwhelmed by the increasing global CDN complexity. The method of building the global CDN needs to be able to assist the designer by reducing the time needed to achieve a balanced overall CDN and reduce the clock buffer usage and area on the clock. It is impractical for the designer to continue using the current method to build the global CDN that is time consuming and utilises limited resources in a very inefficient way.

1.3 Research objectives

i. To reduce the processing time to build a global CDN by proposing an algorithm that could build the global CDN with minimal manual effort from the designer.
ii. To reduce the clock buffer numbers and area in the overall SoC while achieving a balanced CDN by building a global CDN which takes account the clock latency within the partitions.

1.4 Research scope

In this research, the SoC will employ the hybrid CDN with a buffered global distribution and local H-tree clock distribution. The research will focus on creating an algorithm which will enable the building of the buffered global CDN. The algorithm will then be implemented to automate the building of the global CDN. The global CDN will be routed and evaluated. This research will not focus on the quality of the local H-tree clock distribution created by the CTS (clock tree synthesis) within each partitions of the SoC. The data for the partitions' CTS result will be used as one of the constraints for the implementation of this developed algorithm.

There are a few assumptions made for the duration of this research. First of all, this research begins with a completed floorplan with partition area and size already predetermined. Besides that, this research also assumes the clock connection in between partitions are already defined. There are no more clock connection planning required.

This research only focuses on the building of the CDN that is distributing the clock to the partitions. Finally, this research also begins with the clock tree within each partition already completed CTS from the PnR flow. This means the clock tree within the partition is already built with the clock tree latency numbers within the partitions already known. The clock latency numbers within the partition will be used as one of the design constraints for the developed algorithm.

1.5 Thesis outline

There are five chapters for this thesis organized as the following. Chapter 2 discusses and reviews the research that have been done in detail currently. This chapter discusses the advantages and the disadvantages of the research that have been done and elaborate them in detail. Chapter 3 outlines the methodology used in developing the algorithm for the CDN. The chapter will discuss the constraints and the requirements needed for the algorithm. This chapter will also discuss how the algorithm being tested and being evaluated for its effectiveness. Chapter 4 will evaluate the CDN build using the developed algorithm. This chapter will compile the result of CDN and compare the overall quality of the CDN based on the clock buffer used in the entire SoC, the time needed for building the entire clock tree in the SoC and the area needed. Chapter 5 discusses the conclusion of the research and the algorithm. This chapter will conclude the findings of the developed algorithm. With the findings, this chapter also discusses the possible future enhancements that can be done.

Chapter 2

Literature Review

2.1 Introduction

In the previous chapter, an overview on the typical design flow for a SoC from RTL to the final physical design have been described and discussed. While there have been multiple researches and attempts such as the developed memetic algorithm (Shanavas, et al., 2010), improved adjacency matrix (Desai, et al., 2016) and modified Genetic Algorithm (Sangwan, et al., 2014) to improve the design steps within the PnR flow so that the designed SoC is more efficient and improved with better designed SoC quality, there are not much focus on handling the signal crossing between partitions that is associated to the fact that modern SoCs usually consists of multiple smaller partitions integrated together. This presents an entire different issue that is not specifically encoutered or handled within the general PnR flow that focuses on the physical design on the partitions only.

Compared to clocks within a partition which is not restricted by area, modern SoCs needs global CDN to properly distribute the clocks to every partition in the SoC utilizing the channel available in between partitions. Because the channel is restricted in terms of available space and also because clocks are one of the major factor in determining the performance of the SoC, designing and building the global CDN through this confined channel between the partition usually consumes much time due to the amount of work needed in planning to make efficient use of available channel (Gupta & Kulkarni, 2006). This chapter will discuss about the theory behind the general structure of the CDN in modern SoCs and the problems of designing the CDN. Besides that, a section will be allocated to discuss about the importance of the physical aspects of the global CDN that is the clock buffer insertion and routing. After that, this chapter will list out two researches that attempt to build the CDN and discuss the advantages and disadvantages of the proposed method. With the two researches shown, a summary of the issues and disadvantages of each research are discussed at the end of this chapter together with suggestions to make improvement on the issues or disadvantages.

2.2 Purpose and structure of the global CDN

In this section, the purpose and structure of the global CDN is discussed. The first sub-section will give a brief introduction and the evolution of the global CDN to give an insight for the creation and developement of the global CDN from the conventional IC design. This is to show the importance of the global CDN in affecting the quality of the entire CDN. This section will also show the challenges encountered when designing the global CDN in modern SoC and how these challenges warrants the development of an algorithm to build the global CDN.

Besides that, a sub-section will be used to discuss about the available clock tree structure that can be used for the CDN design. A detailed advantage and disadvantage of each type of clock tree structure is shown. Then, the clock tree structure chosen for the CDN design is discussed to show the reason behind chosing the desire clock tree structure and how the chosen clock tree structure is being implemented in the modern SoC.

2.2.1 Purpose of CDN

As mentioned in the beginning of the chapter, the clock is a control signal that is responsible for controlling the function of the SoC since most modern SoCs are designed to be working synchronously. In a synchronous design, the clock that is propagated to each and every endpoints in the partition have to obtain similar latency. Clock skew will be introduced if there are slight difference in the time needed by the clock to reach the endpoints. Since the clock have to be propagated to the entire SoC with endpoints that could be located a distance away from each other which may also be separated by the barriers imposed due to the existence of different partitions, a CDN that will propagate the clock evenly to all partitions housing these endpoints have to be designed and build.

The CDN is generally sub-divided and built separately in two parts in modern SoC, the global tree and the local tree. The global tree of the CDN consists of the clock networks that is distributing the clock from the sender (phase locked loop or PLL that is usually housed within a partition) to the receiver partition. The global tree will have minimal divergence of the clock, as its main purpose is to propagate the clock to all receiver partitions. On the other hand, the local tree of the CDN starts at the input of the receiver partition. The CTS stage for the PnR flow is purposed to build the partition clock network with proper balancing to all endpoints within the partition. The algorithm for building the global tree will be the main focus for this dissertation.

2.2.2 Evolution of global CDN

In the early days of VLSI design, clock distribution in VLSI designs are relatively simple and without much constraints. During the time, a typical VLSI design would contain the entire design in a single chip. There were no concern of distributing the clock signals as the centralized clock signals are free to be distributed within the chip without any barriers or limitations. However, as times passes by with more complex VLSI designs and systems being introduced, a single chip that houses all systems as one single entity becomes impractical. Therefore, the VLSI designs grows from a system in a single chip to multiple system in a single chip design. With this growth in design complexity, maintaining synchronous relationship between systems becomes a challenge. During this period, a method is proposed to arrange the systems on a chip in an equal and orderly manner and leave some spaces so that the spaces can be used for clock routing (Wann & Franklin, 1983) as shown in figure 2.1. This is the earliest example of global CDN in channels between partitions (known as systems during that time).

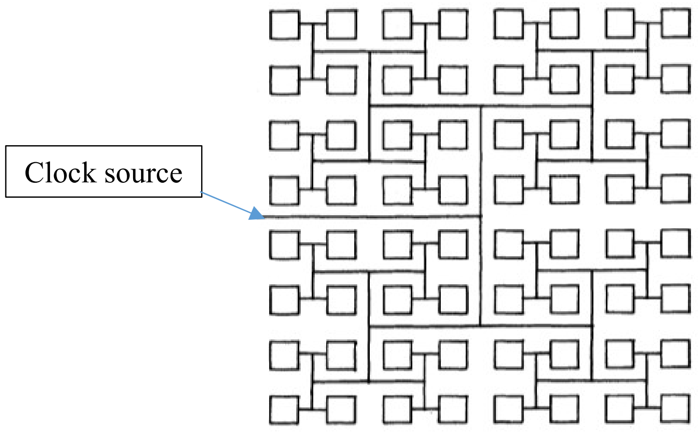

Figure 2.1: Early design of global CDN. The boxes represents each individual partitions with symmetrical and uniform size (Wann & Franklin, 1983)

However, as the technology advances, the rate of these "systems" grow differently between each other. This makes arranging the "systems" in an orderly manner becomes impossible and impractical. This means instead of having an evenly spaced and straight channels between partitions, the channel becomes asysmmetrical due to the fact the partitions have grown to be non-symmetrical in shape and size (Ramanathan & Shin, 1989) as shown in figure 2.2. The CDN no longer can be routed straight to partitions with fixed clock entry point. More time and efforts are needed to identify the clock entry point for each and every partitions. After the clock entry points have been identified, the clock will have to be routed to each assigned clock entry point for the partitions. The routing of the clock have to be properly controlled so that the longest delay to the furthest point will be as minimal as possible. Figure 2.3 shows an example of floorplan with the global CDN already routed. In today's SoC, most of the global CDN continues to utilize this distribution method. This thesis will proposed an algorithm to build the global CDN that will be using an extension of this clock distribution method.

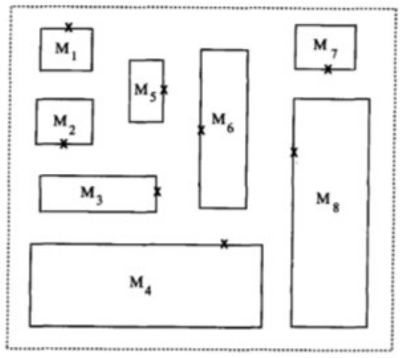

Figure 2.2: Typical floorplan for modern SoC with marked clock entry points (Ramanathan & Shin, 1989)

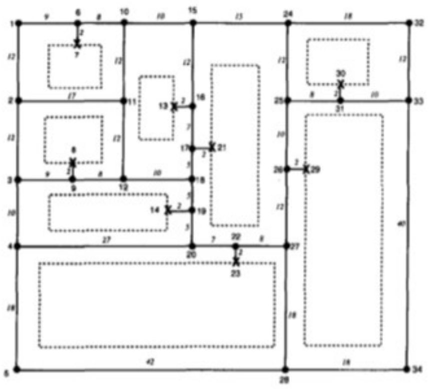

Figure 2.3: Floorplan with routed global CDN (Ramanathan & Shin, 1989), the clock source could be coming from any of the partitions shown

Besides the method of utilizing the area available for the CDN as mentioned, there are new and totally different way of distributing the clock signals which do not need the channel between the partitions also being proposed. One of this method is the grid method of distributing the clock signals such as the modern clock meshes. This method of clock distribution provides an extremely robust CDN that covers the entire chip. This ensures some freedom in obtaining the desired latency due to the fact the clock can have different paths of reaching the endpoint clock sinks. In addition to this, grid style CDN often do not need significant time and effort to implement. However, although the grid style CDN do not need to be routed in the channel between the partitions, the fact that it is routing as a grid means that the available routing resources for other routings will take a signifcant hit due to more spaces consumed by the CDN (Abdelhadi, et al., 2013; Jung, et al., 2015). Because of this, the use of grid style CDN may not be suited for all applications especially applications that are constrained by available die area. Due to its limited use, this method of CDN have been repurposed as the basis and evolved into the modern clock mesh structure which is more useful as a clock tree structure to obtain the perfect skew. Figure 2.4 shows an example of a typical clock mesh structure. The green boxes represents the sinks for the clock.

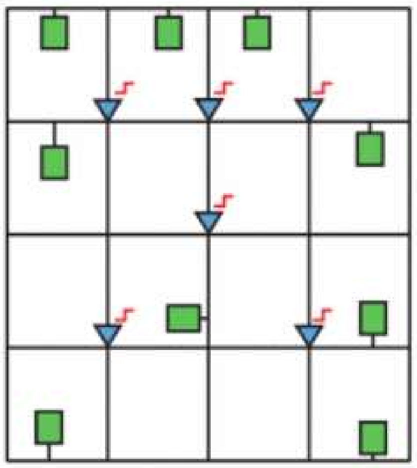

Figure 2.4: A clock mesh structure (Abdelhadi, et al., 2013)

2.2.3 Available Clock tree structure used for CDN

In a general SoC clock tree design, there are a few general types of clock tree that is being used for clock distribution purposes. This includes the buffered clock distribution tree, symmetric-H tree or X tree clock distribution (Remil & Jayasanthi, 2015) or a hybrid of these two types of clock distribution to try to balance the advantages and disadvantages of the different clock trees. While early VLSI designs uses either the buffered clock distribution tree or the symmetric H/X tree clock distribution, modern SoC tends to use a hybrid of these two different clock trees to cater for the need to design modern SoC with increasing complexity.

For the buffered clock distribution tree, the advantages of this clock tree design is the ability to precisely control the latency of each clock branches with the use of buffers which also doubles as the amplifier for the degrading clock waveform and isolating the clock nets from the upstream load impedance. For this type of clock tree as shown in figure 2.5, buffers have to be inserted on the path propagating the clocks because as the clock propagates through the clock nets, the clock waveform will gradually degrade caused by the clock transition. If there are no buffers to amplify the clock signal, the clock waveform would degrade so much that eventually the clock pulse may be degraded to the fact that there are no more clock pulse. This effect will be more pronounced if the physical clock nets are too long. Therefore to solve this issue, intermediate buffers have to be placed in between the clock nets. Because each individual buffers contributes some latency to the clock network, the buffers can be used to carefully fine tune each individual branches of the clock tree to attain a balanced clock tree. However, because the buffers are individually inserted to each branches of the buffered clock tree and compounded by the effect that these branches may be physically located a distance away in the SoC, obtaining an extremely balanced clock tree are harder than the symmetric H/X tree.

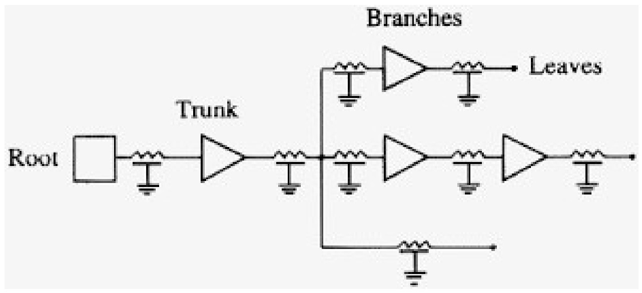

Figure 2.5: The structure of the buffered clock distribution tree (Remil & Jayasanthi, 2015)

On the other hand, one of the main traits of the symmetric H clock distribution tree (as shown in figure 2.6) or the symmetric X clock distribution tree is the clock skew is almost zero due to the structure of the clock tree (Remil & Jayasanthi, 2015; Loo, et al., 2009). This type of clock distribution tree carefully balances the clock skew by distributing the interconnects or the clock nets and the buffers evenly. On the clock net routing side, this clock distribution structure also allows for denser clock routing than the buffered clock distribution tree. This means lesser areas are needed for cell placement and lesser routing resources needed. However, the structure that gives this clock distribution tree is also this clock distribution tree's biggest disadvantage. To build this clock distribution tree, wide and open areas are needed so that the structure can be placed and routed effectively. If the available die area is constrained and confined, this method of clock distribution is not possible to be used. Another problem with this symmetrical H clock distribution tree is the clock is more susceptible to crosstalk issues. This is because as the clock nets are routed closer and denser, the spacing between clock nets become too close. This effect will be more apparent as the process nodes for manufacturing the SoC becomes smaller. Besides that, IR drop issues may also arises when the cells are tightly packed for the symmetric H clock distribution tree. IR drop occurs when there is a voltage drop that happens at a resistive component (in this case the tightly packed cells). If this issue is too severe, this may leads to logic failures within the SoC.

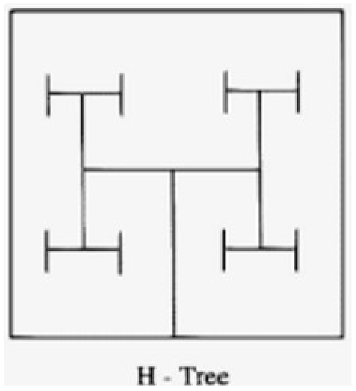

Figure 2.6: The symmetric H clock distribution tree (Remil & Jayasanthi, 2015)

With both buffered clock distribution tree and symmetric H or X clock distribution tree having their advantages and at the same time each have their own crippling disadvantages, modern SoC designs usually combines these two types of clock distribution tree as one hybrid clock distribution tree. By doing this, the disadvantages of each clock distribution tree can be eliminated. As mentioned in previous chapter, a general SoC clock structure consists of trunks and leaves. The trunk of the clock structure are usually located outside of the partition and in the space restricted channels. While on the other hand, the leaves of the clock structure are usually located within the partition. Due to constrained and limited channel available for trunk, the symmetric H clock distribution tree is not suitable to be used. More often than not there are not enough empty areas to house the symmetric H clock distribution structure in the channel between the partitions. However, the structure of the buffered clock distribution tree is perfectly suited to be implemented for the trunk portion of the CDN. The easily tunable nature of the bufferd clock distribution tree is also a desired advantage because this means the individual clock branches can be easily tuned to the required latency before entering the partitions. Once the clock reaches and propagates into the partition, the symmetric H clock distribution tree structure is used to build the clock tree within the partition. This is because the symmetric H clock distribution tree is able to build a clock with almost zero clock skew. Because the clock tree within the partition are handled by the partition PnR flow (CTS stage), this thesis will focus on proposing an alogrithm for building the global CDN.

2.3 Clock buffer insertion and routing

With the die area assigned as the channel for the clock distribution and the clock tree structure for distributing the clocks chosen, the buffer insertion and the routing for the CDN will have to be planned and built carefully. Carefully inserted buffers is needed to conserve the confined available area with no clock transition issues. Most of the time the buffers to be inserted for the channel needs to be guided manually to reduce scenic routing. This is important so that partitions located furthest away from phase locked loop (PLL) can receive the clock in the shortest possible time. The reason for this requirement is because when the clock is balanced inter-partition, not only the registers within the partition (sinks) needs to have minimal clock skew, the sinks in other partitions also need to have minimal clock skew to the partition. If the latency for the clock is too long when propagating to the sinks in furthest partition, there is a high chance that the clock propagating to other partitions will have to be increased deliberately either within the respected partition or on the clock trunk that is sending the clock to the partition. Both of this scenarios means more buffers need to be inserted. To properly insert the buffers in the channel available both the resistances and capacitances for the buffers and the interconnects (clock routing) have to be properly calculated to get the desired clock latency (Wu & Sherwani, 1992). All of this have to be done manually for the channel because current EDA tools available do not work well when trying to build the global CDN in a confined channel. Most EDA tools are more suited to used in the PnR flow for building the clock tree within partitions where wide and open areas are available. This is one of the most time consuming steps in designing the global CDN because of the amount of manual planning that needs to be done.

After the buffer insertion for the clock distrution network is done for the channel, the clock entire clock networks have to routed physically to complete the clock network. The quality of the clock routing is important because it will directly affect the all parameters that are important for the global CDN. For an efficient global CDN, higher metal layers are often used for routing the clock network. This is because higher metal layers are more affected by the wire capacitance and lesser by the resistance of the routing (Fievet, et al., 2015). This is because higher metal layers are wider compared to lower metal layers, the resistance of the metal layer are inversely proportional to the metal layer width. Another reason for using the high metal layer for global CDN is because as the manufacturing process reaches the nanometer domain (65nm and below), the process variation in the manufacturing process becomes increasingly apparent. The process variation of the routing are usually unpredictable until the SoC is manufactured. Hence, it is almost impossible to predict for the variation of the global CDN through simulation and timing checks. This process variation in the global CDN will negatively affect the circuit performance. The variation in the manufactured chips often comes from three different factors: random dopant fluctuations, sub-wavelength lithography, power supply temperature variation and even via resistance due to partial via failures (Narasimhan &

Sridhar, 2017; Herath & Noé, 2010; Farhangi, et al., 2010). Therefore to minimize the possible process variation and more predictability for the global CDN, high metal layers have to be used for routing. However, due to the fact that higher metal layers are often wider, there are lesser available routing resources available for the global CDN in a confined channel. A balance between utilizing the available routing resource and minimizing effect of performance affecting process variation needs to considered when routing the global CDN.

2.4 Previously Proposed global CDN for SoCs

In this section, two different previous papers regarding the research on clock tree design used in SoC will discussed. The first paper to be discussed in the first sub-section is the integrated power and clock distribution network (Esmaeili & Al-Kahlili, 2013). This paper proposes an integrated power and clock distribution network in an attempt to reduce the supporting the clock logics (such as clock buffers) used in the CDN. However, as shown in the sub-section, this method of clock tree design have a few disadvantages that could limit its useability and render it unusable in some SoC designs.

The second paper to be discussed is the timing–driven variation–aware synthesis of hybrid mesh/tree CDNs (Abdelhadi, et al., 2013). This paper proposed to synthesize a hybrid between the clock mesh and a non-uniform tree in order to build a CDN that is extremely balanced and immunes against variation in the process. However, as shown in the sub-section, this method of clock tree design requires extremely high number of clock buffers for supporting it and the complexity asssociated with the design requires significant time and effort to achieve the desired results.

2.4.1 Integrated Power and Clock Distribution Network

In this paper, the global CDN is being being combined and integrated together with the power grid. Unlike the traditional SoC design where the power grid and the CDN is separated, this paper proposes integrating the CDN onto the power grid. In conventional SoC design, the CDN takes up a significant amount of routing and placement resources including the additional clock buffers that are needed to "drive" the clock signal to support the network. By combining and integrating the CDN onto the power grid, a significant amount of routing and placement resource can be saved for the SoC. Figure 2.7 shows the physical structure of the integrated power and CDN. With the omission of a separate dedicated CDN, the components (such as buffers) associated with the dedicated CDN are removed. There are no required algorithm for specially building the integrated power and CDN apart from automation to insert the specially designed clock buffers to extract the clock signal.

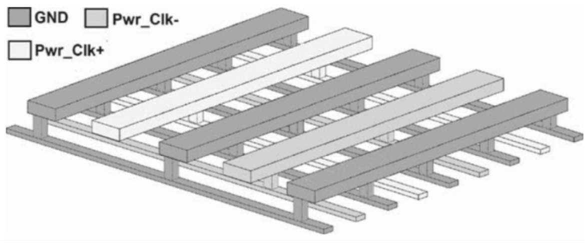

Figure 2.7: The physical structure of the proposed integrated power and CDN (Esmaeili & Al-Kahlili, 2013)

In the paper, an oscillating signal with dc component and sinusoidal swing is generated the using a differential LC $P_{wr}\_C_{lk}$ driver (figure 2.8) through the V_{DD} port. To extract the clock signal from the structure, a specially designed clock buffer (figure 2.9) that feeds into the clock pin of the load is used. Figure 2.10 shows the waveform of the combined power and clock signal and the extracted clock signal after going through the clock buffer. From the research, it is found that this technique of integrating power and CDN can achieve about 20% reduction in power with potential for more savings if network capacitance becomes dominating. At the same time, the global CDN can be eliminated entirely if this technique is used for distributing the clock.

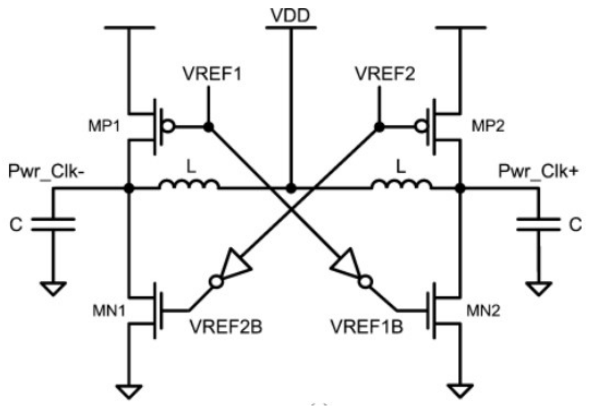

Figure 2.8: Structure of differential LC $P_{wr}\_C_{lk}$ driver (Esmaeili & Al-Kahlili, 2013)

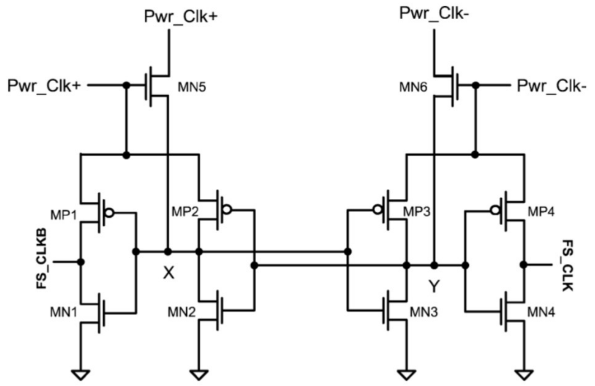

Figure 2.9: Structure of the specially designed clock buffers to extract the clock signal from the integrated power and CDN (Esmaeili & Al-Kahlili, 2013)

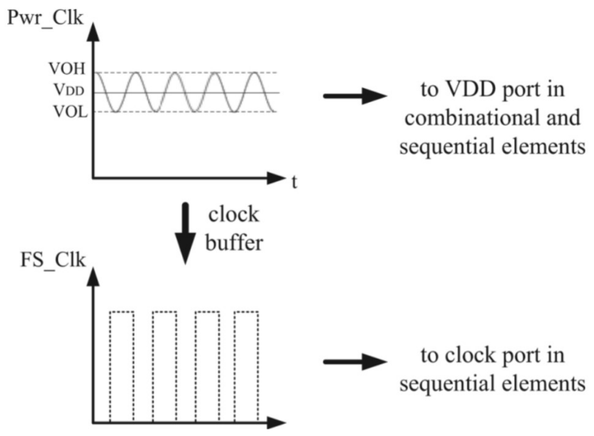
Pwr_Clk
VOH
VDD
VOL
t
to VDD port in combinational and sequential elements
clock buffer
FS_Clk
to clock port in sequential elements

Figure 2.10: The combined power and clock signal and the extracted clock signal after feeding through the specially design clock buffers (Esmaeili & Al-Kahlili, 2013)

While the proposed method of distibuting the clock signal shows significant routing and placement resource savings, there are a few disadvantages associated with implementing this design. Firstly, this network is only suitable to use on digital circuits, analog circuits such as PLLs, oscillators and op-amps cannot be powered by this integrated power and CDN because the function of these circuits are very sensitive towards power supply variation. To implement this design in SoC, a separate power grid which is powered by a more stable supply voltage is needed. This requirement will increase the complexity and time required to build the network especially in the area adjacent to the analog circuits significantly because these areas could be occupied by two different grids that is needed by analog and digital components of the SoC.

Another issue with the integrated power and CDN lies with the clock buffers that is designed specially to extract the clock signal. The clock buffer that is used to extract the clock signal have a significantly larger footprint than a normal clock buffer used for clock tree building during CTS. In the paper, it is mentioned that the specially designed clock buffer can be used to supply the clock for three registers with about a reduction to

750mV peak for the clock signal. Further register sharing is not possible because the peak of the clock signal may reduce too much until the clock signal cannot be recognized. This means a high number of these specially designed clock buffers will be needed to supply the clock signal to SoC designs with high number of registers. The clock buffer areas saved by removing the CDN may be instead consumed by the insertion of these specially designed clock buffers.

Another disadvantage of implementing this integrated power and CDN is the quality of the clock signal could easily be affected by the crosstalk effect in the SoC. As shown in figure 2.10, the extracted clock signal depends on the difference between the VOH and the VOL (amplitude of the waveform). In the paper, it is shown that the signal when both power and clock component are integrated, the voltage swing of the combined signal is small. The buffers extracting the clock signal will need to be very precise in extracting the signal. If the grid is being affected with crosstalk, the waveform propagating in the grid may not be uniform with a specific period. This means the extracted clock signal will be affected and this may corrupt the clock signal rendering the SoC unusable. This effect will become more pronounced as the manufacturing process decreases because the crosstalk effect will become increasingly apparent.

2.4.2 Timing–driven variation–aware synthesis of hybrid mesh/tree CDNs

In this paper, a hybrid of mesh and non-uniform tree CDN is being proposed to distribute the clock signal. The purpose of proposing a hybrid of mesh and non-uniform tree is because mesh CDN (figure 2.11) are able to distribute clock signal with toleration against non-uniform switching and unbalanced clock distribution. The mesh CDN also