RTL Design Quality Checks for Soft IPs

Smrutisikta Patra

Department of Electronics and Communication Engineering
National Institute of Technology Rourkela

RTL Design Quality Checks for Soft IPs

Thesis submitted to the

National Institute of Technology Rourkela

In partial fulfilment of the requirements for

the degree of

Master of Technology

In

Electronics and Communication Engineering

(Specialization – VLSI Design and Embedded Systems)

By

Smrutisikta Patra

(Roll Number: 214EC2192)

Under the supervision of

Prof. Sougata Kumar Kar

May 2016

Department of Electronics and Communication Engineering
National Institute of Technology Rourkela

Department of Electronics and Communication Engineering
National Institute of Technology Rourkela

May 26, 2016

Certificate of Examination

Roll Number: *214EC2192*

Name: *Smrutisikta Patra*

Title of Thesis: *RTL Design Quality checks for Soft IPs*

We the below signed, after checking the dissertation mentioned above and the official record book (s) of the student, hereby state our approval of the dissertation submitted in partial fulfilment of the requirements for the degree of Master of Technology in Electronics and Communication Engineering at National Institute of Technology Rourkela. We are satisfied with the volume, quality, correctness, and originality of the work.

Sougata Kumar Kar
Supervisor

Department of Electronics and Communication Engineering
National Institute of Technology Rourkela

Prof. Sougata Kumar Kar

Assistant Professor

May 26, 2016

Supervisor's Certificate

This is to certify that the work presented in this dissertation entitled "***RTL design quality checks for Soft IPs***" by "***SMRUTISIKTA PATRA***, Roll Number **214EC2192**, is a record of original research carried out by her under my supervision and guidance in partial fulfilment of the requirements for the degree of ***Master of Technology*** **in** ***Electronics and Communication Engineering***. Neither this dissertation nor any part of it has been submitted for any degree or diploma to any institute or university in India or abroad.

Sougata Kumar Kar

I would like to dedicate my thesis to
my beloved family

Declaration of Originality

I, **Smrutisikta Patra,** bearing Roll Number **214EC2192** hereby declare that this thesis entitled "***RTL design Quality checks for soft IPs* "** represents my original work carried out as a postgraduate student of NIT Rourkela and, to the best of my knowledge, it contains no material previously published or written by another person, nor any material presented for the award of any other degree or diploma of NIT Rourkela or any other institution. Any contribution made to this research by others, with whom I have worked at NIT Rourkela or elsewhere, is explicitly acknowledged in the dissertation. Works of other authors cited in this dissertation have been duly acknowledged under the section "Bibliography". I have also submitted my original research records to the scrutiny committee for evaluation of my dissertation.

26th May 2016

NIT Rourkela

Smrutisikta Patra

Acknowledgement

I would like to thank my project supervisor Prof. Sougata Kar whose support helped me for my project work and writing thesis.

I am very grateful to Prof. (Dr.) Kamala Kanta Mahapatra, Prof. (Dr.) D.P Acharya, Prof. Ayas Kanta Swain, Prof. Shantanu Sarkar, Prof. (Dr.) Nurul Islam and all other faculties and staff of ECE Department, NIT Rourkela for their help and support for carrying out my internship and project work at industry. I am very much thankful to all of my classmates and other friends who had helped making my stay in NIT a pleasant experience.

Lastly I am thankful to my family for their admiration and support. I express my highest gratitude for my parents for their inspiration and encouragement which helped me to finish my work.

26th May 2016

NIT Rourkela

Smrutisikta Patra

Roll No. - 214EC2192

Abstract

Soft IPs are architectural modules which are delivered in the form of synthesizable RTL level codes written in some HDL (hardware descriptive language) like Verilog or VHDL or System Verilog. They are technology independent and offer high degree of modification flexibility.

RTL is the complete abstraction of our design. Since SOC complexity is growing day by day with new technologies and requirement, it will be very much difficult to debug and fix issues after physical level. So to reduce effort and increase efficiency and accuracy it is necessary to fix most of the bugs in RTL level. Also if we are using soft IP, then our bug free IP can be used by third party. So early detection of bugs helps us not to go back to entire design and do all the process again and again.

One of the important issue at RTL level of a design is the Clock Domain Crossing (CDC) problem. This is the issue which affects the performance at each and every stage of the design flow. Failure in fixing these issues at the earlier stage makes the design unreliable and design performance collapses. The main issue in real time clock designs are the metastability issue. Although we cannot check or see these issues using our simulator but we have to make preventions at RTL level. This is done by restructuring the design and adding required synchronizers.

One more important area of consideration in VLSI design is power consumption. In modern low power designs low power is a key factor. So design consuming less power is preferred over design consuming more power. This decision should be made as early as possible. RTL quality check helps us on this aspect. Using different tools power estimation can be performed at RTL stage which saves lots of efforts in checking and redesigning.

This project aims at checking clock domain crossing faults at RTL stage and doing redesign of circuit to eliminate those faults. Also an effort is made to compare quality of two designs in terms of delay, power consumption and area.

Contents

Certificate of Examination ii

Supervisor's Certificate iii

Declaration of Originality v

Acknowledgment vi

Abstract vii

List of Figures x

List of Tables xii

1. Introduction **1**

1.1 Introduction to IP 1

1.1.1 Soft IP 1

1.1.2 Hard IP 1

1.2 ASIC Design Methodology 2

1.3 RTL Description in VLSI Flow 3

1.4 Requirement of RTL Quality Check 4

2. Clock Domain Crossing Checks On RTL **11**

2.1 Clock Domain 11

2.2 Multiple Clock Domains in SOC 12

2.3 Clock Domain Crossing (CDC) 13

2.4 Basic Issues In Clock Domain Crossing 14

2.4.1. metastability 15

2.4.2. Data Loss 16

2.4.3. Data Incoherency 17

2.5 Tools Used To Check Clock Domain Crossings At RTL Stage 18

3. Analysis of CDC Violations **21**

3.1 Absence of Synchronizer 21

3.2 Some Scenarios with Potential CDC Bugs 21

3.2.1 Two Phase Shifted Sequencing Control Signal 21

3.2.2 Encoded Control Signal Passing 23

3.3 Why 2 DFF Synchronizer can Not Used For Multiple Data Crossing Between Clock Domains 25

3.3.1. Handshaking between two clock domains 26

3.3.2. Passing data through a FIFO with clocsed loop acknowledgement setup 26

3.4 Multiple Cycle Path (MCP) Formulation With Feedback Acknowledged 26

3.5 Multi-Bit CDC Signal Passing Using FIFO Synchronizer 29

4. Power Estimation on RTL **31**

4.1 Requirement of Power Estimation of RTL 31

4.2 Calculation of Switching Activity 31
4.3 Average Power Calculation 32
4.3.1. Static Power Consumption 33
4.3.2. Dynamic Power Consumption 33
4.4 Introduction To Power Artist Tool 34
4.5 Power Analysis Flow 35
4.6 Schematic Representation Of MCP Implementation In Power Artist 36
4.6 Schematic Representation Of FIFO Synchronization In Power Artist 37
5. Results of RTL Quality Checks 40
5.1 Simulation Output Of Data Transfer By MCP Technique And FIFO Mechanism 40
5.1.1 Delay Comparision Between MCP And FIFO 41
5.2 Activity Output Waveform Of MCP Technique And FIFO Technique 42
5.3 Textual Reports Of Power Analysis 43
5.4 Textual Reports Of Area Analysis 45
6. Conclusion and Future scope **47**
6.1 Conclusion 47
6.2 Future Scope of the project 48
7. References **49**

List of Figures

Figure 1.1 Digital VLSI design Flow ……3

Figure 1.2 Example of RTL coding of D Flip-Flop in system verilog ……..…..4

Figure 1.3 RTL representation of the D Flip-Flop in VCS tool…….5

Figure 2.1 Examples of synchnonous clock domains…….....11

Figure 2.2 Two clock signals synchronous to each other…….....12

Figure 2.3 Two clock signals asynchronous to each other…….…..12

Figure 2.4 SOC containing multiple IP with multiple clocks……..13

Figure 2.5 Clock Domain Crossing in between CLKA and CLKB domain……....14

Figure 2.6 Set up time and Hold time for a synchronous signal……..15

Figure 2.7 Example of metastability case……15

Figure 2.8 wave form of the above case showing metastability case……..16

Figure 2.9 wave form showing a scenario of Data loss……...16

Figure 2.10 A scenario where no data loss is present ……..…..17

Figure 2.11 A scenario with data loss……..17

Figure 2.12 An Example of data incoherancy…….....18

Figure 2.13 Questa CDC tool Flow…….....19

Figure 2.14 GUI of Questa CDC…….....20

Figure 2.15 two D flip-flop synchronizer…….....20

Figure 3.1 Example of two control signals which are phase shifted……...22

Figure 3.2 Simulation output of phase shifted control signal……..…23

Figure 3.3 solution for sending two control signals…….....23

Figure 3.4 output waveform of the modified circuit…….…..24

Figure 3.5 scenario of encoded signal crossing between clock boundries…….….24

Figure 3.6 waveform showing a problem with the used circuit…….….25

Figure 3.7 revised circuit for the solution to the previous problem…….....25

Figure 3.8 Pulse Generator circuit diagram……26

Figure 3.9 Output of closed loop solution of above problem……26

Figure 3.10 Circuit diagram of MCP procedure……28

Figure 3.11 State diagram of the FSM used in the circuit……..29

Figure 3.12 RTL block diagram of MCP technique……...29

Figure 3.13 Block diagram of FIFO technique……...30

Figure 3.14 RTL block diagram of FIFO technique……...…31
Figure 4.1 Flow chart of power analysis……..37
Figure 4.2 Colorized Shematic of MCP in PowerArtist……..38
Figure 4.3 Colorized Shematic of alogic block in PowerArtist……..39
Figure 4.4 Colorized Shematic of blogic block in PowerArtist…….39
Figure 4.5 Colorized Shematic of FIFO synchronizer in PowerArtist……40
Figure 4.6 Colorized Shematic of dpram block in PowerArtist……..40
Figure 4.7 Colorized Shematic of wct1 block in PowerArtist…….40
Figure 4.8 Colorized Shematic of rct1 block in PowerArtist……41
Figure 5.1 VCS simulation output waveform of MCP formulation……43
Figure 5.2 VCS simulation output waveform of FIFO synchronizer……..44
Figure 5.3 Activity waveform of MCP formulation……45
Figure 5.4 Activity waveform of FIFO synchronization…….45
Figure 5.5 Total power consumption in MCP formulation technique…….46
Figure 5.6 Total power consumption in FIFO synchronization technique……..46
Figure 5.7 power consumption classified interms of gate/model type for MCP…….47
Figure 5.8 power consumption classified interms of gate/model type for FIFO…….47
Figure 5.9 power consumed by each component for MCP formulation……..48
Figure 5.10 power consumed by each component for FIFO synchronization……48
Figure 5.11 Area occupied by each component for MCP technique……..49
Figure 5.12 Area occupied by each component for FIFO synchronization……....49

List of Tables

Table 5.1 Delay comparison of MCP and FIFO design…….……....45

Table 6.1 Characteristic comparison of MCP and FIFO design……51

Chapter 1

Introduction

1.1 Introduction to IP

Intellectual Property (IP) is nothing but a reusable unit of logic, cell or chip layout design. The IP can be used by one party or can be given to another party as a third party IP for use in other design. The IP cores are used as building blocks in ASIC chip or FPGA based designs. IPs have much more pronounced effect in System ON chip (SOC) design field because SOC is the collection and connection of several IPs. If many of the IPs are taken from third parties and are used then the design cost as well as design time get reduced because we don't need to design everything from scratch. And this plays a very important contributing factor in chip design industries.

IPs can be categorized as the following categories:

1.1.1. Soft IP

These IPs are generally offered as synthesizable codes written in Hardware Descriptive Language (HDL) like Verilog or VHDL which are analogous to high level languages like C in the field of computer programming. IPs delivered to chip makers as RTL permit chip designers to modify designs at the functional level, though many IP vendors offer no guarantee for support for modified designs.

IPs are also sometimes offered in the form of generic gate-level netlists. The netlists are Boolean-algebra representation of the IP's logical function implemented in terms of generic gates or any process specific specified standard cells. An IP implemented as generic gates is more portable to any process technology. A gate level netlist is similar to an assembly code in the field of computer programming. Netlist gives IP vendor reasonable protection against reverse engineering. Both synthesizable code as well as netlists are called Soft IPs.

1.1.2. Hard IP

These IPs are defined in lower level, physical description. They offer better predictability for chip performance in terms of timing performance and area. Generally analog and mixed signal logics are delivered as Hard IPs. For example: DAC, ADC, PLL

etc. Analogs IPs are provided to the chip makers in transistor-layout format (such as GDSII). Digital IPs are sometimes delivered in layout format also.

1.2 ASIC Design Methodology

Any Integrated circuit (IC) other than a general purpose IC which contains the functionality of thousands of gates is usually called an ASIC (Application Specific Integrated Circuit). ASICs are designed to fit a certain application. An ASIC is a digital or mixed-signal circuit designed to meet specifications set by a specific project.

Typical ASIC design flow requires several general steps that are perhaps best illustrated using a process flow chart:

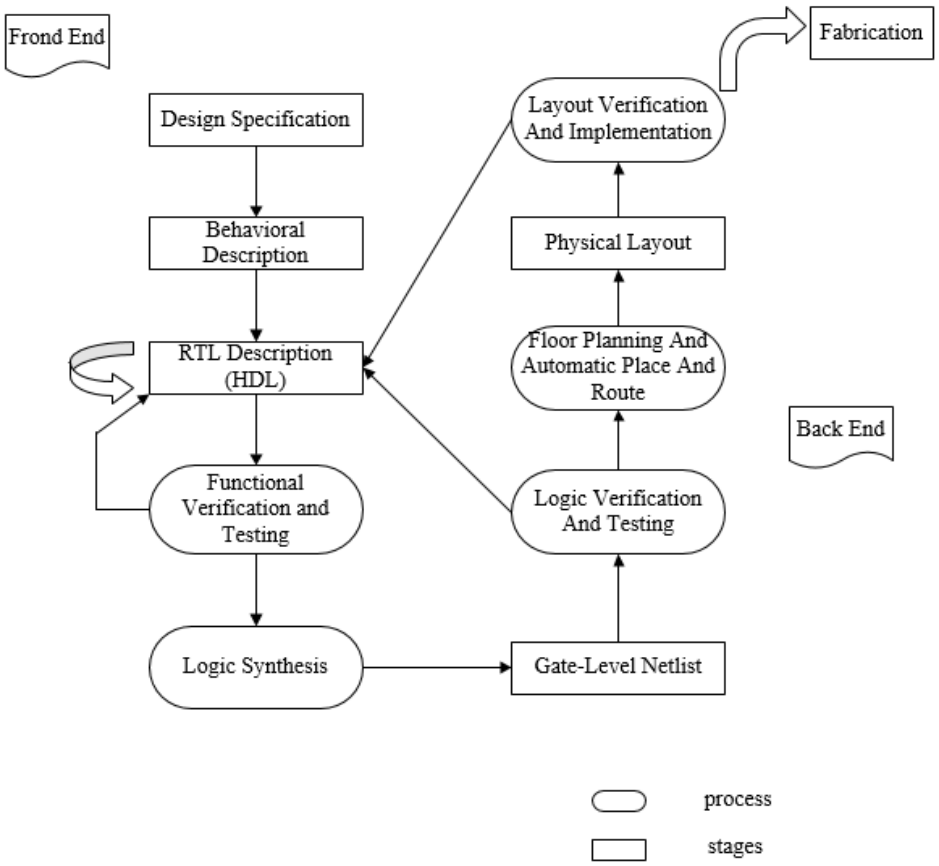


Figure 1.1 Digital VLSI design Flow

Specifications comes first, they describe abstractly the functionality, interface, and the architecture of the digital IC circuit to be designed.

- Behavioral description is then created to analyze the design in terms of functionality, performance, compliance to given standards, and other specifications.
- RTL description is done using HDLs. This RTL description is simulated to test functionality. From here onwards we need the help of EDA tools.
- RTL description is then converted to a gate-level netlist using logic synthesis tools. A gate-level netlist is a description of the circuit in terms of gates and connections between them, which are made in such a way that they meet the timing, power and area specifications.
- Finally a physical layout is made, which will be verified and then sent to fabrication.

1.3 RTL Description in VLSI Flow

The Register Transfer Logic (RTL) is a design abstraction in digital logic design. It models a synchronous digital circuit in terms of data flow in between different hardware registers and logical operations performed on other signals.

After the behavioral description of your design is done, the next step is to create high-level representations of a circuit, which is nothing but the RTL, from which lower-level representations and ultimately actual wiring are derived. These RTL are represented using a Hardware Description Language (HDL) like VHDL and Verilog.

RTL is used in the design stage of the IC design flow. This abstract level description is then generally converted to a gate level description of the circuit by a logic synthesis tool. Then this synthesis results are used by placement and routing tools to create a physical level layout. Logic simulation tools also use a design's RTL description for verifying its correctness.

A simple Example of RTL description of a D flip-flop is stated below:

```
module dff (clk, d, q);
input clk, d;
output q;
reg q;
always @ (posedge clk)
begin
q<= d;
end
endmodule
```

Figure 1.2 Example of RTL coding of D Flip-Flop in system verilog

The above RTL description is done using Verilog language. It contains the input, output ports declaration along with the internal logic, in this case the sequential logic which

describes the function of the D flip-flop. So the RTL describes our design in terms of a box which has several ports and internally all are interconnected as per the logic.

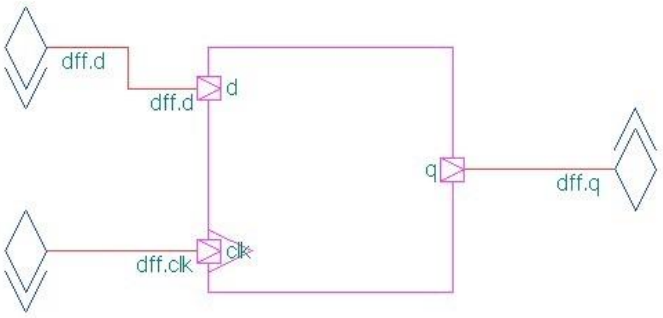


Figure 1.3 RTL representation of the D Flip-Flop in VCS tool

1.4 Requirement of RTL Quality Check

Since RTL is the design abstraction it is the earliest stage where the design can be modified as much as possible as per satisfying our requirements. As we go for further design processes in the flow the design flexibility reduces gradually. The scope of redesign of certain module or the entire design decreases as we enter in post silicon level. So RTL plays a very important role in the design phase.

RTL quality check is otherwise known as RTL sign off. There are two System on Chip (SOC) challenges that mandate RTL signoff.

1. The increasing complexity of SOC which complicates the chip assembly and validation.
2. The quality of the third IP uses must be verified and assured.

Running checks at RTL detects a lot of issues which are potential bugs in design. Run time at RTL level is much more less in magnitude than that of run time at post silicon level. If our RTL quality check gives us a good quality of results then we can find and fix a lot of issues within less time. Errors caught at the post silicon phase, have to go back through the entire design flow and do all the changes and redesign if required. This whole process is a time consuming process. So RTL checks are emphasized more in design early phase.

RTL quality checks consists of checking comprehensive set of some “must pass” requirements. These requirements include some of the following:

a. Functional coverage

b. Lint checks of the RTL code
c. Clock domain crossing checks
d. Timing constraints
e. Power intent
f. Power consumption and power reduction
g. Testability

The more the RTL will be cleaned the less will be the probability of functional errors in post silicon level and the less will be the time required for the SOC to do the final tape out. Some of the quality checks can be briefly describes as following:

a. **Functional coverage**: A good functional coverage is a measure that our Design under Test (DUT) behaves correctly in our designed verification environment. Since the complexity of circuit is growing more and more complex day by day there are more and more number of transistors and thus more number of states of the variables, to analyze. So a comprehensive verification environment must be created for that reason. Functional coverage is used as a guide for directing the verification resources by identifying tested and untested portions of the design and reports those in the form of different metrics.

   Many formal verification tools are available which checks the functional coverage. Some of them are, Conformal LEC by Cadence, FormalPro by Mentor graphics etc.

a. **Lint checks**: Linting checks means performing a static analysis using a RTL verification tools which comprises of a set of predefined rules and guidelines which reflect a good coding practice. This static analysis catches common errors which may tend to lead to a buggy code. When any of the rule is breached the tool reports the errors in terms of a report which contains all detailed violation list categorized in terms of the rule ID. Generally this linting step is performed on hardware descriptive language i.e. HDL which is nothing but the RTL of our DUT, prior to simulation. It helps to catch a very large number of bugs with a very few iterations, which saves time and cost both.

   Some of the linting tools available in market are Leda RTL checker by Synopsys, Spyglass by Atrenta etc. The basic flow of all the tools are same. They

perform lint checks on RTL and report any errors and warnings present in term of rule IDs. So it is easier to analyze because once we know the fix for a particular rule, we can resolve all errors under that particular rule group which saves a lot of time.

b. Clock Domain Crossing checks: This is one of the most important checks which is performed on RTL. When we deal with multiple clock domains in SOCs, there the concept on clock domain crossing or CDC comes into picture. Because as the SOC complexity is increasing day by day, the number of clock frequencies used are also getting increased. So large number of signals cross different clock boundaries and interact with modules present in other clock frequency domain. It should be handled with so much accuracy that we must ensure by proper checks that the signals are crossing safely and there is no functional errors expected due to these crossings. These checks are done on RTL using many tools. CDC check addresses problems like metastability, data loss and data incoherency which are potential issue in RTL.

Many tools are available in market, from several vendors which automate the total CDC check flow. The tool has its in build CDC schemes which are nothing but group of possible clock domain crossing scenarios. It flow is such that it will take the RTL as input and the constraint file which contains the clock groups and details of which port in in which clock domain. Then the tool analyzes all the clock domain crossing signals and reports the violations signals in term or textual report or a GUI. One such tool is Questa CDC tool provide by mentor graphics.

c. Timing constraints: Correct timing constraints are key to a smooth chip design flow. Incorrect constraint causes wasted timing closure iterations because it increases the run time of implementation tool. Also it can cause silicon bug. So using correct and efficient constraints are very much necessary. In today's VLSI era automation of constraint generation, their verification and management task in very much important.

Many tools are there to automate this whole process, supplied by many vendors. For example, FishTail, which is a leading provider of several design automation tools which automates the whole constraint related flow. The products allow designers to generate clock constraints, I/O delays, clock senses and clock groups for ensuring that the clocking of our DUT is accurately reflected in the clock

constraints. Also the RTL designers can generate multi-cycle path (MCP) and timing critical false cycle path (FCP), which are very much required to be analyzed at early level of design flow. Also this product allows RTL designers to check that they have correctly specified the clocks and case analysis on the DUT from verification prospective to formally verify the MCP and FCP are correct.

d. Power Intent: With the advancement of technology day by day device size is shrinking. So optimizing our design for leakage and dynamic power helps us reduce the power and energy consumption and also reduces packaging cost. But these low power methods also tend to complicate our verification task and they introduce risk during synthesis and physical implementation. Full chip simulations are not effective practical method for verifying today's large, complex designs now a days.

Now a days designers are configuring the RTL by power partitioning i.e. separating RTL from power intent of the design. The power intent is described in a separate format. Unified Power Format (UPF) is one of the popular format of the standard for specifying power intent of designs. The power intent consists of definition of power rails supplies, power domains, retention strategies, isolation strategies, level shifting strategies etc. There are many EDA tools available in market to check these power intent compatibility with RTL and if there is any mismatch then those are reported so that required change can be done to the UPF and again we will check. Some of the examples of these tools are Spyglass-LP by Atrenta and VC-LP by Synopsys.

e. Power consumption and power reduction: The advantage of using combination of low-power components along with low-power design techniques is more important now a days. For saving battery life in both portable and non-portable devices. Since static power dominates over dynamic power as technology nodes advances raising the need for new power management technique. Some of the power reduction techniques are

- Clock gating
- Power gating
- Multiple supply voltages
- Dynamic voltage and frequency scaling

- Multi-threshold CMOS
- Active body bias

We can estimate the performance of the power deduction method used in our design, at the RTL stage using any of the EDA tool available for power estimation. One of such tool is Power Artist which is provided by Apache. We can calculate average power, time based power, activity of each module etc. using this tool. We can modify our DUT if calculated power becomes very high, which gives us flexibility of redesigning and in turn saves money and time.

f. Testability: Earlier, testability of a design (DFT) used to be checked at gate level since designs used to be small and simple, back then. But day by day since device size is shrinking and performance is getting enhanced, the transistor count is increasing. This large device density, tight timing, power loss issues, yield loss etc. begin to pose serious challenges. When these effects are combined with core usability the issues become more complex. To reduce some efforts, testability issues are now a days fixed at RTL level because it saves many iterations as compared to testability checks at gate level.

Fixing DFT issues at RTL stage allows designer to generate a testable RTL core which can be reused without any repetition of DFT checking and any repair process. One of the tools available for DFT check is Spyglass-DFT which is provided by Atrenta. It checks for DFT issues and report them. This also allows the logic/scan tool and physical synthesis tool, which takes the layout information to optimize area, power and timing after DFT fixes are made.

1.5 Literature Review

Saurabh Verma, et all [7] discussed basics of clock domain crossing (CDC) problem. The main issues of clock domain crossing are metastability, data loss, data incoherency etc. Ginosar [3] has shown how metastability is degrades the performance of circuit and how synchronizers are used to resolve the problem of metastability in digital designs.

Also Clifford [1] has discussed many practical scenarios of CDC in digital designs. Generally at RTL stage CDC is analyzed and desired modifications are done to the code of the design written in system verilog/verilog. Naghmeh, et all [2] has proposed methods to

perform detection and diagnosis of CDC failures in SOCs. Generally SOCs contain multiple clock domains for which CDC check is a must step to follow. D. J. Kinniment, et all [5] gave overview of synchronization circuit performance. H.-K. Kim, et all [8] discussed regarding Testing of synchronizers in asynchronous FIFO.

Hatture, et all [10] gave details regarding Open loop and closed loop solution for clock domain crossing faults. These are basic methods to pass multiple bit data signal across clock domains. N. sharif, et all [12] discussed about Quantitative Analysis of state-of-Art synchronizers from clock domain crossing prospective. Ankush Patharkar, et all [14] focused on another issue in CDC called data loss. Some scenarios of this problem is also discussed in this project in subsequent sections.

C. Leong, et all [17] has proposed methodologies for Built-In CDC test and Diagnosis in GALS systems. J. Horstmann, et all [18] has discussed metastability from CMOS prospective. Metastability can be caught in CMOS master/slave circuit. Also S. Beer, et all [22] have proposed An on-chip metastability measurement circuit to characterize synchronization behavior in 65nm.

A robust synchronizer can be implemented as proposed by J.Zhou, et all [24]. Method and circuit for improving metastable resolving time in low-power multi-state devices can be implemented as suggested by R. L. Cline [25].

Also power estimation can be done at RTL stage. D. L. Liu, et all [4] has discussed regarding Power consumption estimation in CMOS VLSI chips. Also M. Nemani et all [6] have proposed High-level area and power estimation for VLSI circuits.

1.6 Thesis Outline

Following this chapter introduction, the remaining part of the thesis can be summarized as follows:

Chapter-2: Clock domain crossing checks on RTL

This chapter describes the problem in Clock Domain Crossing (CDC) basics and the main issues in CDC are discussed.

Chapter-3: Analysis of CDC Violations

This chapter contains analysis of basic CDC scenarios and have discussed the solutions for those CDC issues..

Chapter-4: Power estimation on RTL

This chapter describes importance of power estimation at RTL stage and also has discussed RTL power estimation procedure.

Chapter-5: Results of RTL Quality Checks

This chapter contains results of RTL quality checks in terms of CDC and power estimation.

Chapetr-6: Conclusion and Future scope

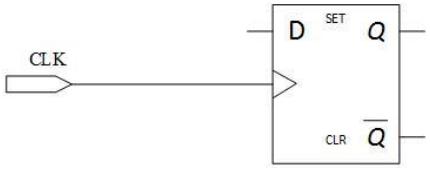
This chapter is final section of the thesis which contains the conclusions of the complete project work. Also it has discussed regarding the future scope of the project i.e. how this project can be carried forward for more enhancements.

Chapter 2

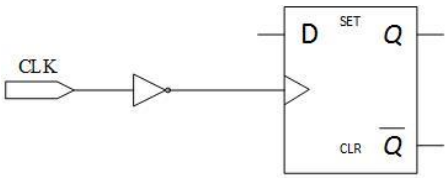
Clock Domain Crossing Checks On RTL

2.1 Clock Domain

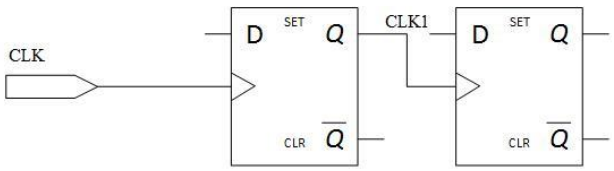
A clock domain is a part of a design, which has a clock. In that domain all components with in that domain will use that particular clock as a reference i.e. all synchronous component will be sensitive to that particular clock. So all transitions will happen with respect to that clock transition. The clock belonging to a clock domain operates either asynchronous to, or has a variable phase relationship with another clock signal in the design. For example, a clock and its derived clock (derived using a clock divider) are in the same clock domain since they have a constant phase relationship. But, for example 50MHz and 37MHz clocks (phase relationship between them changes over time) define two separate clock domains. Figure 2.1 illustrates three different clocks used in a design, but all are synchronous to each other. (a) Represents CLK, (b) represents its inversion and (c) represents CLK1 (derived from CLK). All three are synchronous to each other.



(a)



(b)



(c)

Figure 2.1 Examples of synchnonous clock domains

Figure 2.2 shows two clock signals, one with frequency 10Hz and other one is 5Hz. Despite of two different frequencies these two clocks are synchronous to each other because first clock frequency is integral multiple of frequency of second clcok frequency. So they toggle at the same instant ,if there is no phase delay and if there is any phase delay then also that phase difference is constant always.

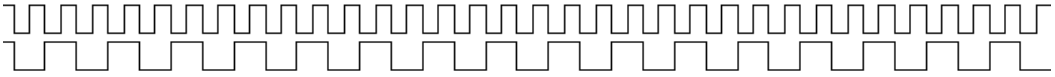


Figure 2.2 Two clock signals synchronous to each other

Figure 2.3 shows two clock signals, the first one is of frequency 14Hz and second one is of 10Hz. Even if they start with zero phase difference the phase difference between the clock edges are not constant always. And if there is any initial phase shift then the phase difference between clock edges become more random and more unpredictable. So these two clock signals are considered as asynchronous to each other.

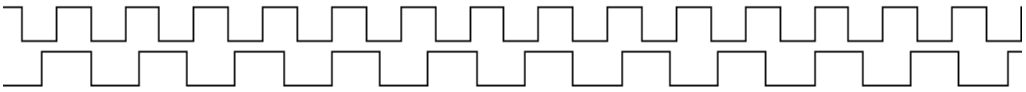


Figure 2.3 Two clock signals asynchronous to each other

2.2 Multiple Clock Domains in SOC

In modern VLSI design, the System on Chip (SOC) is emerging as an evolutionary methodology for solving the performance issues which arises due to long interconnects. SOC is nothing but connection of several IPs. So SOC design is completely dependent on the communication between different IP sub blocks and as well as consideration on less energy dissipation.

The IP blocks present in the SOCs may operate with different clock frequencies with respect to others according to their internal individual functionality. So when a SOC is ready with after integrating all IP blocks, the connections are also made for communication in between IPs. So there are situations which involves signals crossing in between one or more than one different clock boundaries.

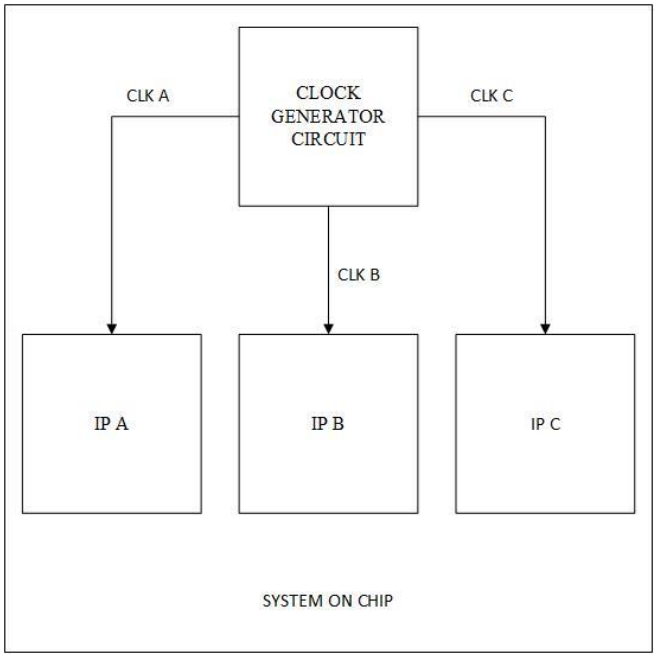


Figure 2.4 SOC containing multiple IP with multiple clocks

As shown in figure it is schematic of a simple SOC, we see that the SOC consists of the IP clocks named as IP A,IP B and IP C. These three IPs can be designed by a single vendor or can come from three different vendors . If the IP is taken from some vendor other than the SOC vendor then it can be considered as a third party IP. The third party IPs can be directly integrated into our design with out any analyzys of its internal RTL because it is the role of the third party IP vendor to deliver reliable IP block. So these IPs are generally made blackboxed for during our RTL checks.

As depicted in the above SOC diagram we see the three IPs work at three different clock domains i.e. IP A works at frequency of clock CLK A, IP B works at frequency of CLK B and IP c works at frequency of CLK C. considering these three clocks are different we have three different clock domains. So when IPs are integrated to form the SOC all the signals and ports get connected at required place. So we have many signals going through more than one clock domain, which is other wise known as clock domain crossings.

2.3 Clock Domain Crossing (CDC)

SOCs are becoming more complex day by day. A lot of functionalities are getting added to chips and data is transferred from one clock domain to another. So, clock domain crossing (CDC) verification has transformed as one of the major verification challenges in today's deep submicron designs. If a design fails at this stage then it is must that we should fix this issue in RTL. It does not make any sense to use the RTL with CDC violations

because the device performance will never be correct. A single CDC bug ceases the performance of whole IP and some time the whole SOC too.

A clock domain crossing occurs when data is transferred from a flip-flop driven by one clock to a flip-flop driven by another clock. Fig 2.5 shows the scenario of a clock domain crossing. There are two clock domains one is CLK A which is the sending clock domain and other one is CLK B which is the receiving clock domain. In each domain separate D flip-flops are involved to sample data. In CLK A domain the D flip-flop is sampling the data signal at a rate as of CLK A transitions. The sampled data is then going to another flip-flop which is in clock domain CLK B. So this flip-flop is sampling the received signal at a rate that of CLK B transitions.

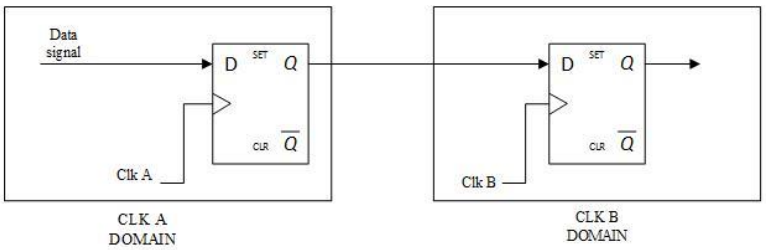


Figure 2.5 Clock Domain Crossing in between CLKA and CLKB domain

So the output of the second flip-flop is now a sampled signal in new clock domain and ready to be used by further circuitry in clock domain CLK B. So signal must be correctly sampled to be used in the receiver domains. But in physical level this sampling does not happen always correctly. There are different physical issues which comes into picture at post silicon level. Generally at pre silicon level no issue comes when we check clock domain crossing. But to reduce time of analysis and redesigning, we check the potential clock domain crossing issues at RTL level only which is the basic requirement for CDC checks on RTL.

2.4 Basic Issues In Clock Domain Crossing

The main issues when a signal crosses from one clock domain to another clock domain are as given below:

- Metastability
- Data loss
- Data incoherency

2.4.1. metastability

Metastability refers to signals that do not assume stable 0 or 1 states for some duration of time. A signal goes metastable when there is set up or hold time violation while data is getting sampled by a particular clock.

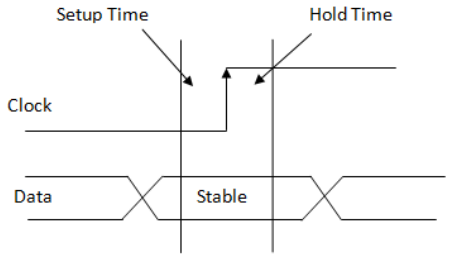


Figure 2.6 Set up time and Hold time for a synchronous signal

- **Setup time:** It is the time span by which data must be stable before the clock edge transition. If data toggles within this setup time window then data will not be sampled correctly.

- **Hold time:** It is the time span by which data must be stable after the clock edge transition. If data makes transition within hold time window then data may not get properly sampled.

If the signal crossing different clock domains with different clock frequencies, then there is a chance of the signal becoming metastable means it will go to unpredictable state and if at this stage the output is used at several other places in the design then this will lead to functional error. Figure 2.7 shows the scenario of metastability.

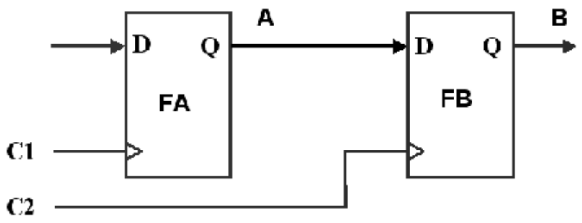


Figure 2.7 Example of metastability case

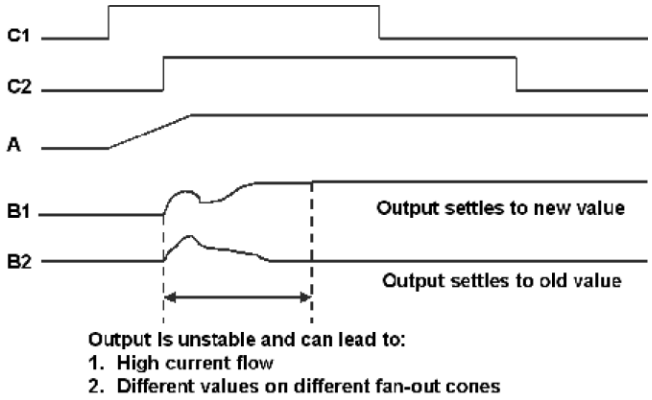


Figure 2.8 wave form of the above case showing metastability case

The metastable signal if used in further circuitry then it affets the functionality of the whole design. So to fight this issue we have to take measure at RTL stage. We have to make our RTL metastable resistant by adding extra required logic. This involves addition of synchronizers.

2.4.2. Data Loss

Due to metastability sometimes even if new data is generated by the source the source the destination domain may not capture the data correctly. Figure 2.9 shows how due to metastability the data is lost during the first transition of data A and then getting captured in the next clock cycle.

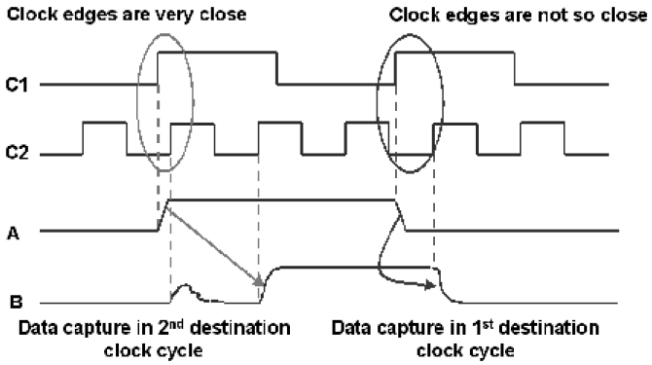


Figure 2.9 wave form showing a scenario of Data loss

As long as each transition of source domain data is captured in the destination domain there is no data loss. So it is must obeyed that data signal coming from sending domain must be kept stable for atleast one clock duration of the receiver domain.

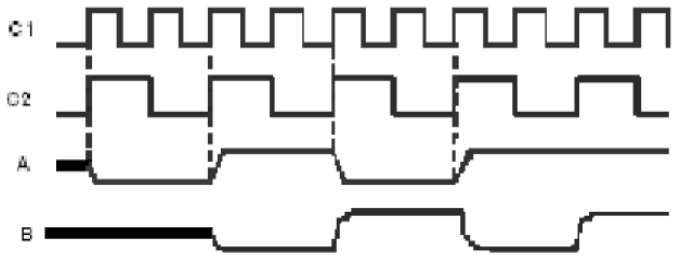


Figure 2.10 A scenario where no data loss is present

Figure 2.10 shows that there is no data loss because the data signal is kept stable for atleast duration of one clock period one clock C2. But the following figure 2.11 shows that there is a data miss at output. It is because data A is kept at stable value 1 for a less time so data is not getting captured due to metastability and also in the next clock cycle the data is chaning to a new vaue that is 0. So the pulse of value 1 is lost completely.

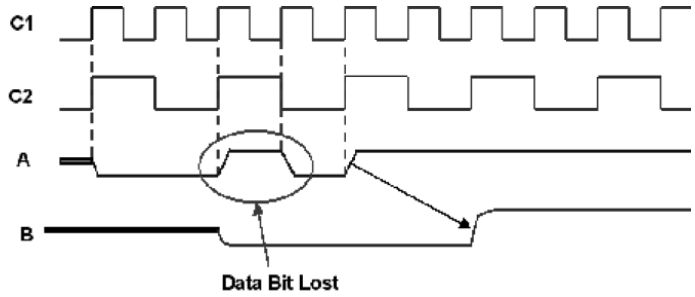


Figure 2.11 A scenario with data loss

Some of the solution to this data loss problem are:

- Keep the sending data stable for some particular duration so that data can be sampled atleast once. To automate this process we can implement a FSM technique which can generate source data at a rate such that one thata will be in stable state atleast for one cycle of destination clock. This method is generally used with synchronous clock domains.
- For asynchronous clock domain crossing use of FIFO helps. The basics of FIFO design and its working principle is explaianed in section 3.5.

2.4.3. Data Incoherency

When multiple data or a bus is transferred in between two clock domains, then if there will be transition in some bits of that bus at a time ans if the time duration falls within the setup and hold time window, then all those bits goes metastable at a time and attain unpredictable state for some period until in the next clock transition new value will be sampled. So these intermediate value which is unpredictable causes functional issue in our

design. This is called data incoherency issue. One instance of this problem is depicted in figure 2.12. In the figure we can see that when the both bit X[0] and X[1] are becoming metastable for a period giving an invalid data i.e. valie 2'b01. Our transmitted values of X were 2'b00 and 2'b11. But our received values of Y are 2'b00,2'b01 and 2'b11. So it creates a potential problem in terms of functionality.

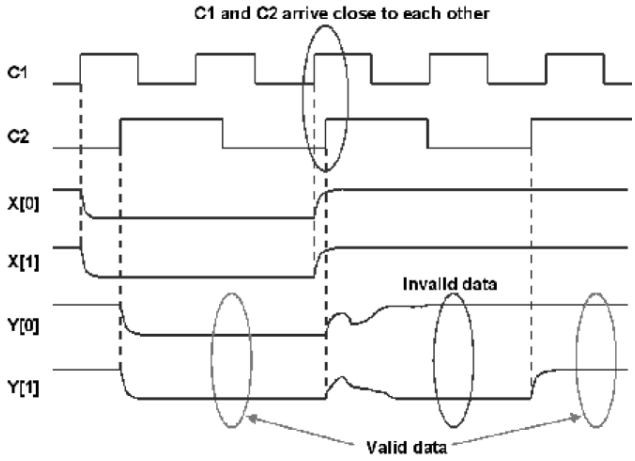


Figure 2.12 An Example of data incoherancy

Some of the solutions to this problem are

- Use of gray coded control signal if they cross across different clock domains. Gray coding will not affect the performance since it is the control signal and only occurance of all pattern matters.
- For data signals we can not use gray coding since it will create wrong sequence of data which is undesired. So for for data signals FIFO technique as described in section 3.5 and multi cycle path (MCP) as decribed in section 3.4, are used mostly.

These issues as explained, are potential threat to a design. These issues are more pronounced at post silicon level. After getting a bug at this level we have to go back to RTL and redesign to it takes so many iteration. So we have to make our RTL such that the possible post silicon bugs can be caught at RTL level itself and we can fix those which will save time and cost both. There are tools which helps to identify these bugs at RTL stage. The basic of how these tools work is explained in below section.

2.5 Tools Used To Check Clock Domain Crossings At RTL Stage

There are different vendors providing tools for checking the clock domain crossing issues. The flow of the tools are almost same for all. Some of these tools are as below

- Questa CDC® mentor graphics

- spyglass CDC® Atrenta
- VC CDC® Synopsys

Out of these in this project I have used Questa CDC tool to analyze my RTL. The tool flow is explained in figure 2.13.

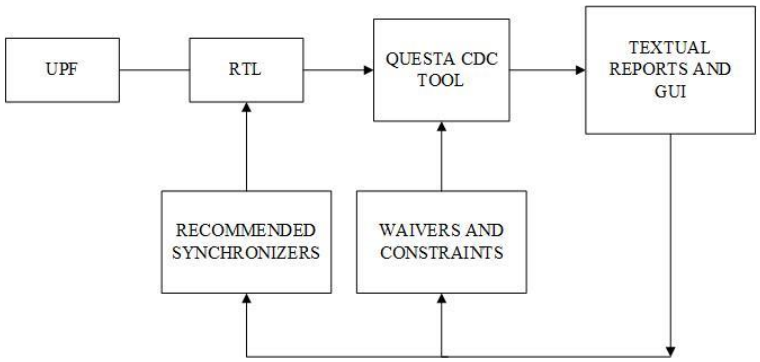


Figure 2.13 Questa CDC tool Flow

The Questa CDC tool takes the UPF file and RTL as the inputs. The tool has internal set up to check all signals which cross in between the clock domains. So when we run the tool after loading our RTL, it checks for all possible clock domain crossing signals and according to the category all CDC paths are classified in the report. The report can be seen in the GUI like shown in figure 2.14.

The GUI contain the results of CDC run. We can see all the CDC paths grouped accordance to the scenario which is called typically “schemes”. Out of all schenarios I have worked on “missing synchronizer” , “multiple bits”, “FIFO” and “Handshake” schemes which will be discussed in the next chapter.One of such scheme is “missing synchronizer” which is focused on this project. The solution to this problem is to add a 2 Dff synchronizer as shown in figure. The working principle is if data does not follow set up and hold time then the output of the first flip flop becomes metastable. But the second flop gives this metastable signal one full clock period time to attain stable. So the final output is stable 0 or 1 value with no metastability.

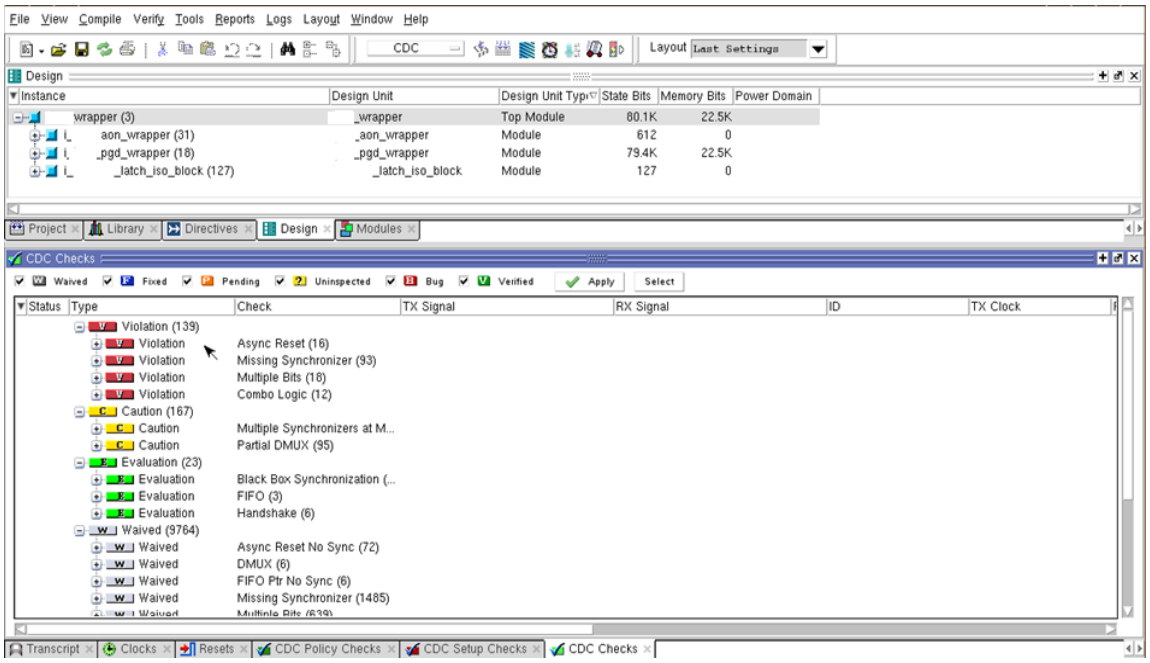


Figure 2.14 GUI of Questa CDC

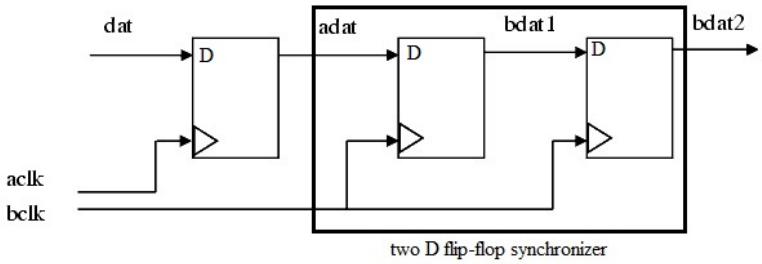


Figure 2.15 two D flip-flop synchronizer

The above diagram is the circuit of a simple synchronizer. But simply we can not add synchronizers to each and every CDC signals blindly. We have to analyze properly before adding any synchronizer otherwise it may lead to additional functional error. Some of such scenarios are discussed in next chapter.

Chapter 3

Analysis of CDC Violations

3.1 Absence of Synchronizer

As we studied in section 2.4.1 that a clock domain crossing signal can cause metastability while crossing from one clock domain to another. We cannot see the metastability in simulation since it comes at post silicon level. The only thing we can do is to add synchronizer in RTL CDC paths so that it will avoid metastability.

3.2 Some Scenarios with Potential CDC Bugs

I found some of the issue during my CDC analysis on the RTL which are described below. I have fixed the issues using different methodologies.

3.2.1 Two Phase Shifted Sequencing Control Signal

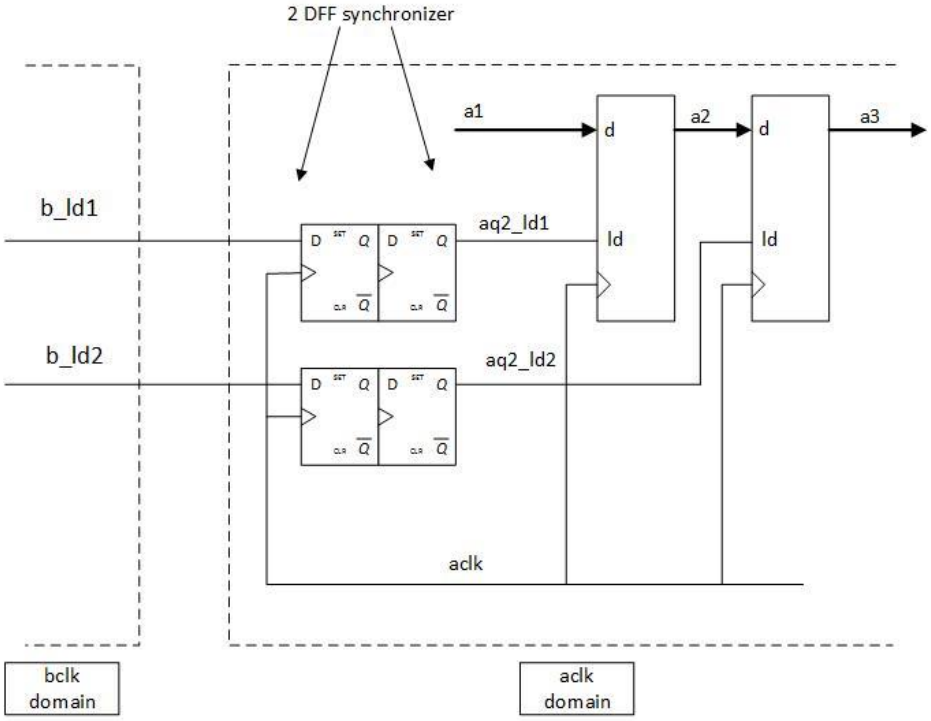


Figure 3.1 Example of two control signals which are phase shifted

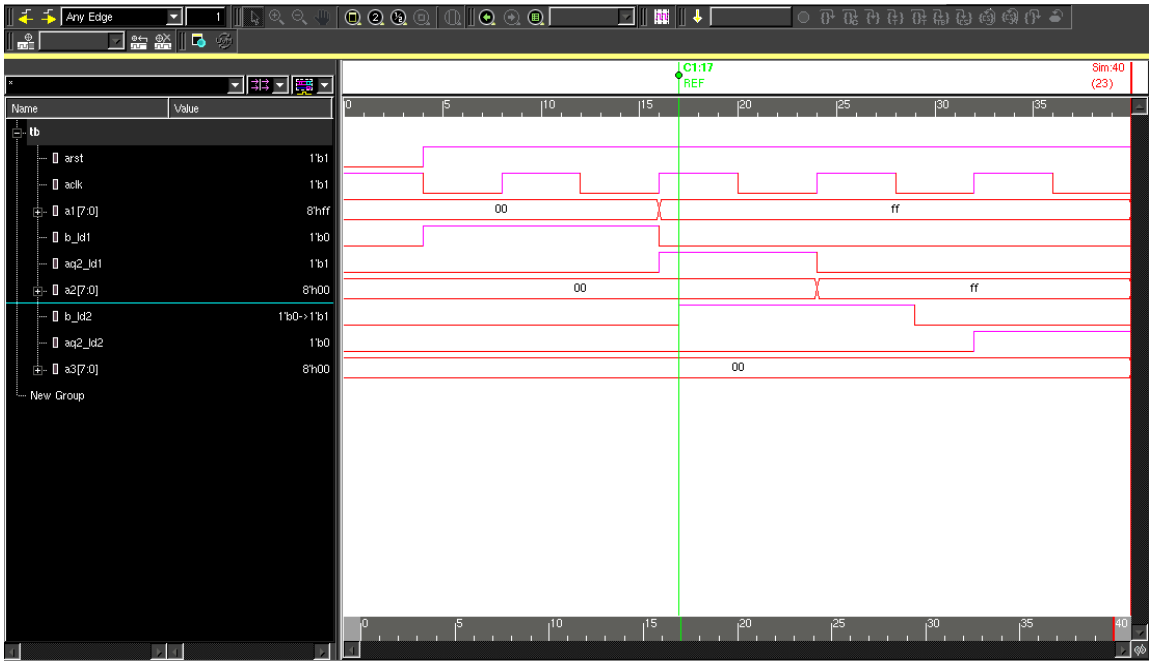


Figure 3.2 Simulation output of phase shifted control signal

As shown in above figure due to little skew between aq2_ld1 and aq2_ld2 signals data is missed where it is required. To resolve this issue we will cross one signal in between two domains and we will generate the other load signal is the receiver domain using a D flip-flop as shown in Figure.

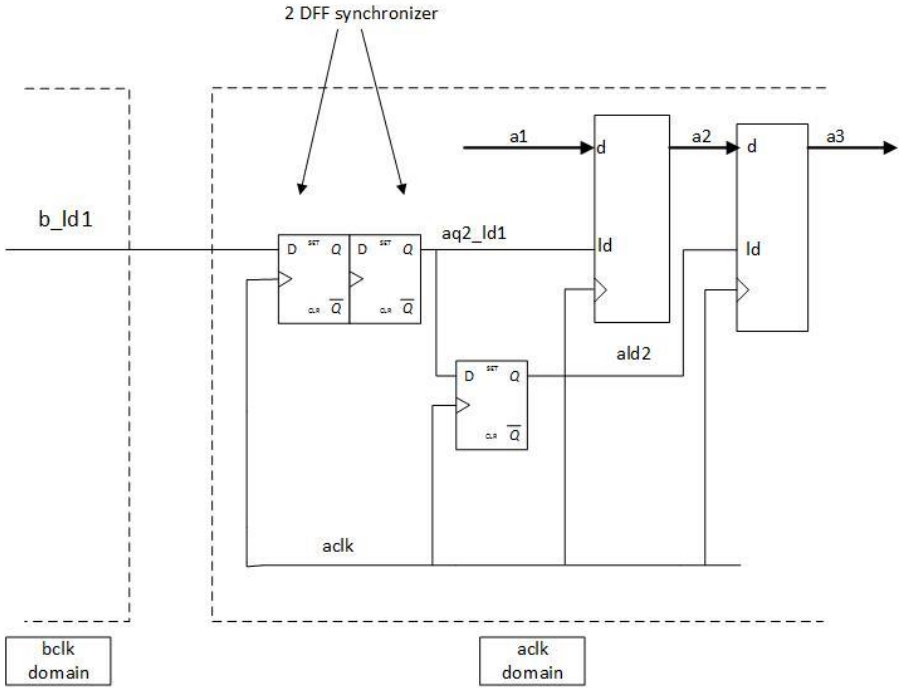


Figure 3.3 solution for sending two control signals

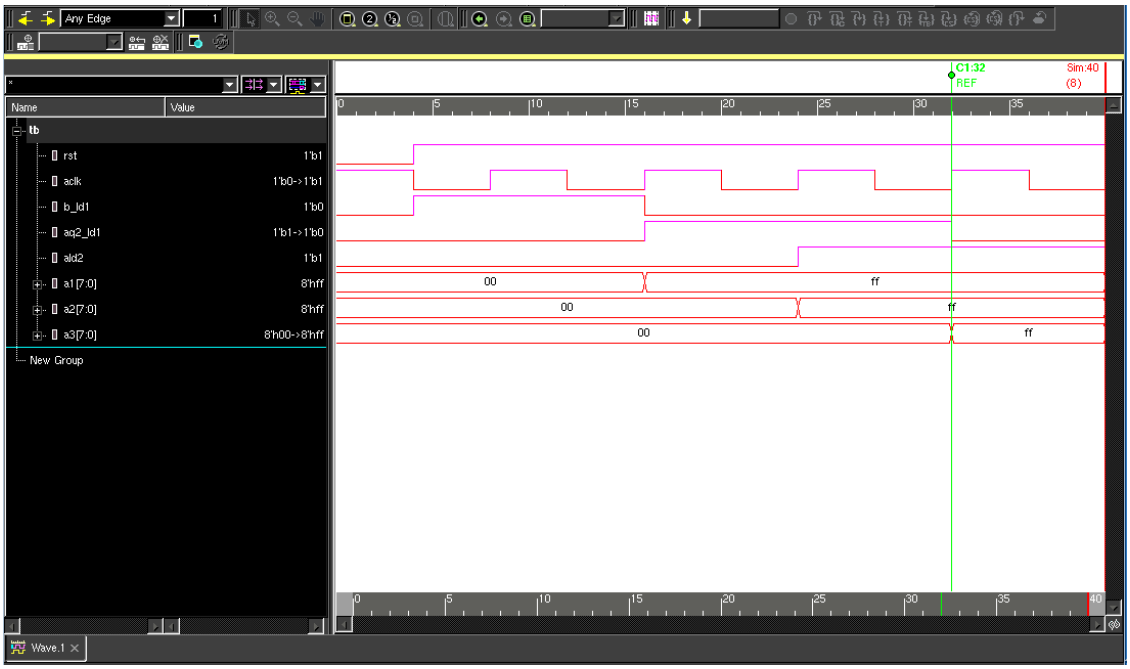


Figure 3.4 output waveform of the modified circuit

3.2.2 Encoded Control Signal Passing

Since the two encoded signals are slightly skewed when sampled errorneous decoded output is getting generated . The solution is to pass signal as it is and send an enable signal and synchronize it .

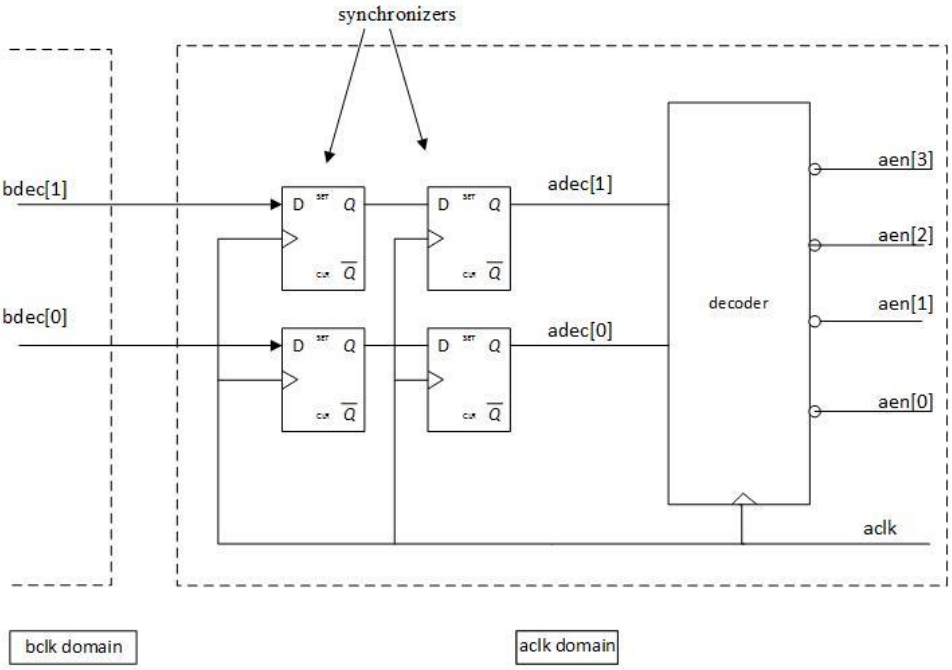


Figure 3.5 scenario of encoded signal crossing between clock boundries

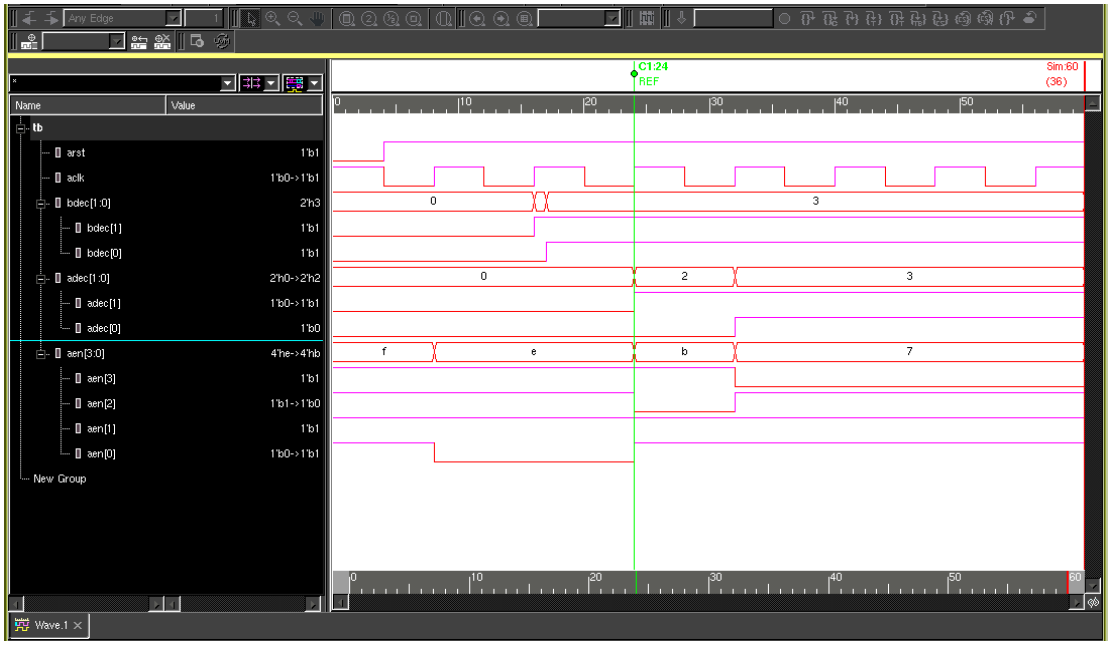


Figure 3.6 waveform showing a problem with the used circuit

The adec bus values 0'b10 is undesired which is occuring due to the slight skew between the bdec signals. Due to this output is wrongly decoded. But using an synchronized enable signal to load the bdec inputs is eliminating this issue and signlas is properly decoded at the output. The block diagram of the revised circuit is depicted in figure 3.7.

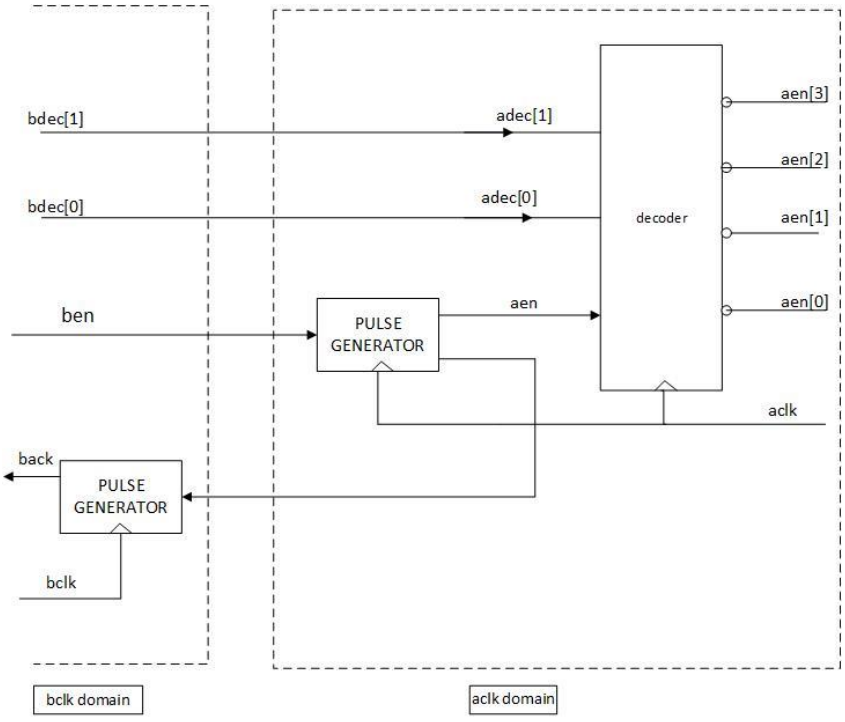


Figure 3.7 revised circuit for the solution to the previous problem

In the above implementation we are not synchronizing data rather data is directly sent along with a synchronized enable signal “ben”. So data will be properly sampled in this case by the receiving domain clock. Also a simple feedback mecahnism is there to send an acknowledgement to the sending domain to indicate that data is received. The following is the circuit for pulse generator.

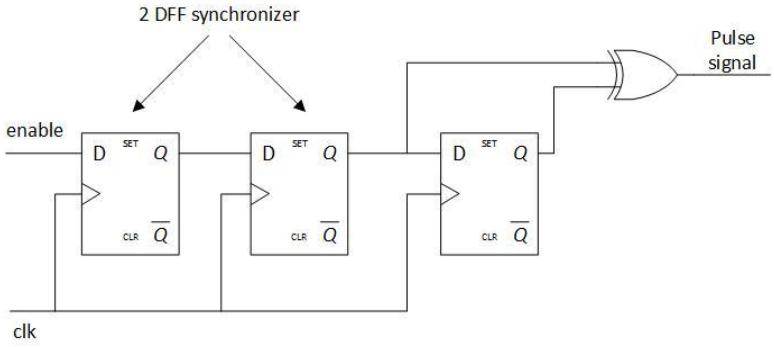


Figure 3.8 Pulse Generator circuit diagram

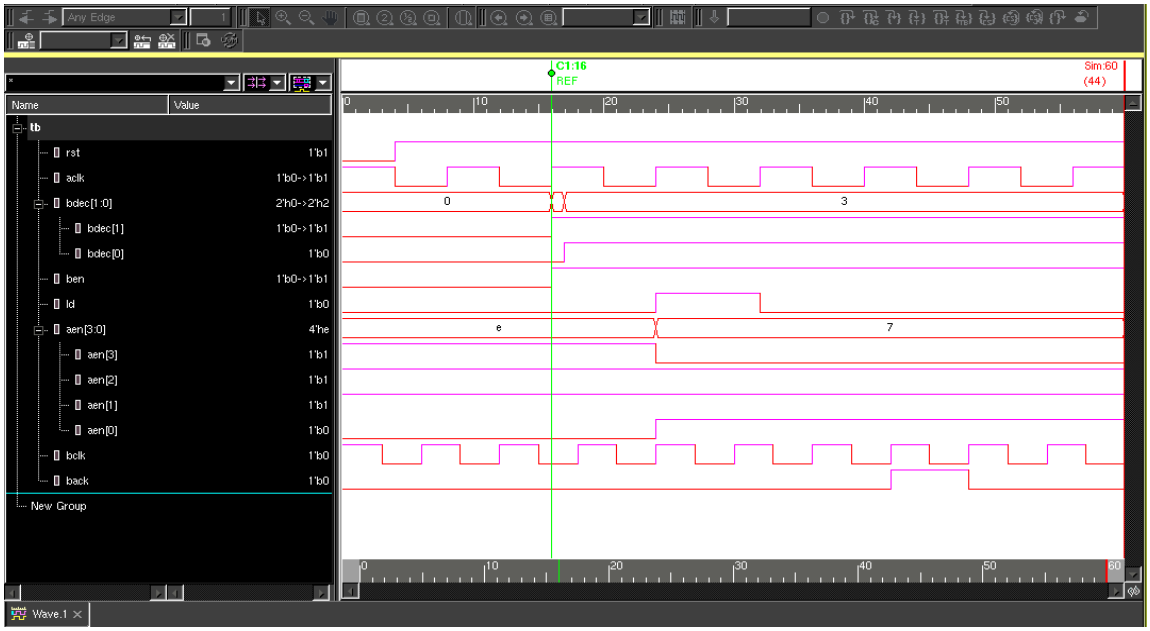


Figure 3.9 Output of closed loop solution of above problem

3.3 Why 2 DFF Synchronizer can Not Used For Multiple Data Crossing Between Clock Domains

As shown in figure 3.5 it is proved that using two DFF synchronizer for each bit of a bus is not a proper method to resolve CDC issues because it will lead to skew between

individual bits of the bus and it will lead to data incoherency. Unnecessary data value gets captured due to this and it leads to functional errors.

The solution to this issue is to use special kind of scenario for data buses. Some of the techniques are

3.3.1. Handshaking between two clock domains:

It involves data transaction in between two clock domains. Data is sent from the sender domain along with some control signal and that control signal can be synchronizd via a 2 DFF. Then that control signal will be used to load data in the receiver domain. There will be similar setup in the receiving domain to create a load control pulse to load the data in the output. Then there can be an acknowledgement signal sent back to sending domain to indiacte that next data can be loaded. Since data crossing ishappening in coordination with both domains the process is called Handshaking. Also specifically this technique is best known as Multiple Cycle Formultion(MCP) technique. Detailed implementation is shown in further section.

3.3.2. Passing data through a FIFO with clocsed loop acknowledgement setup：

Passing multiple bits, whether data bits or control bits, can be done through an asynchronous FIFO. An asynchronous FIFO is a shared memory or register buffer where data is inserted from the write clock domain and data is removed from the read clock domain. Since both sender and receiver operate within their own respective clock domains, using a dual-port buffer, such as a FIFO, is a safe way to pass multi-bit values between clock domains. A standard asynchronous FIFO device allows multiple data or control words to be inserted as long as the FIFO is not full, and the receiver and then extract multiple data or control words when convenient as long as the FIFO is not empty.

3.4 Multiple Cycle Path (MCP) Formulation With Feedback Acknowledged

MCP formulation is a common technique for safely passing multiple CDC signals.The main advantage of this technique is

a) Sending clock domain is not required to calculate the appropriate pulse width to send between clock domains

b) Sending clock domain is required to toggle an enable signal into the receiving clock domain to indicate data has been passed and ready to be loaded.

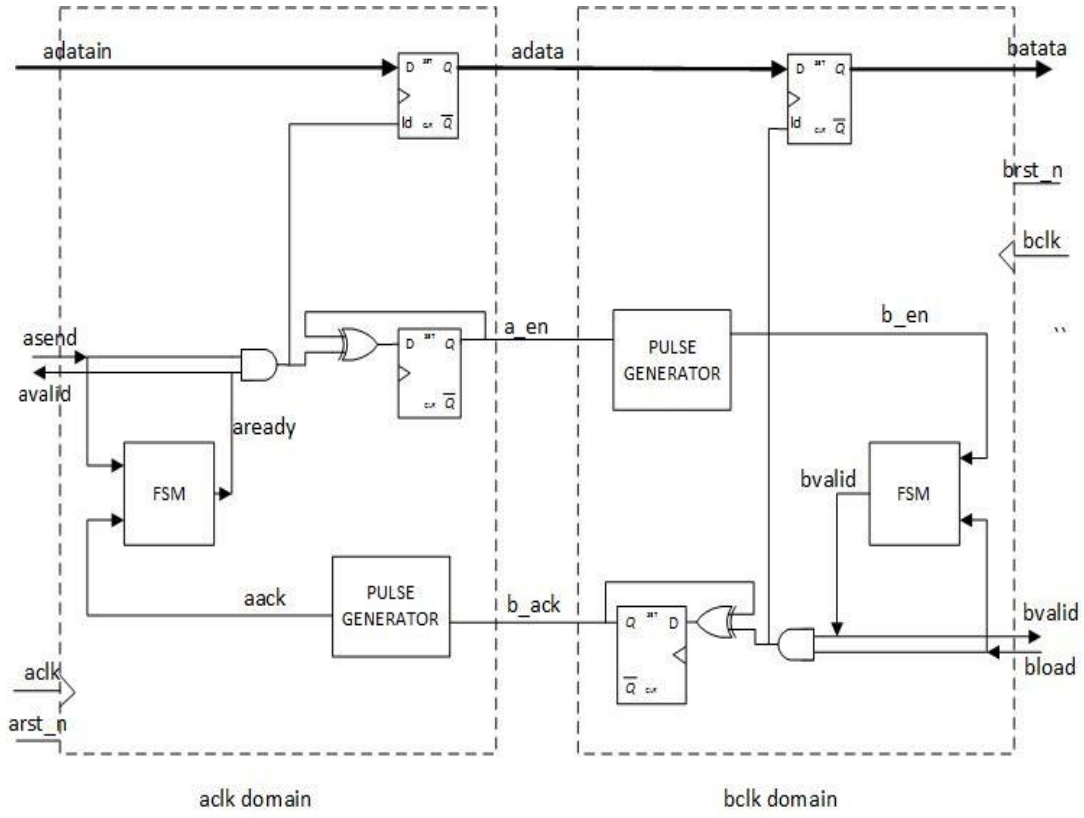


Figure 3.10 Circuit diagram of MCP procedure

The above block diagram is the simple block level representation of a handshaking mechanism in between two clock domans i.e. “aclk” and “bclk”. Signal “asend” is made high when data is sent. But data does not get loaded until we get high value on acknowledgement signal “avalid”. Signal “aready” is generated by an FSM mechanism. The FSM state diagram is given in figure 3.11 . When “aready” is high data gets loaded to the flop present in aclk domain. Also enable signal “a_en” toggles for each 0 to 1 transition of aready and this signal is synchronized as well as a pulse is generated by a pulse generator circuitry as shown earlier in figure 3.8. The synchronized output i.e. “b_en” is used along with signal bload are used in an FSM meachanism to generate “bvalid” signal which is used to load data finally to the output. Also a backword mechanism is used to send an acknowledgement “aack” to aclk domain as a mark to load the next data.

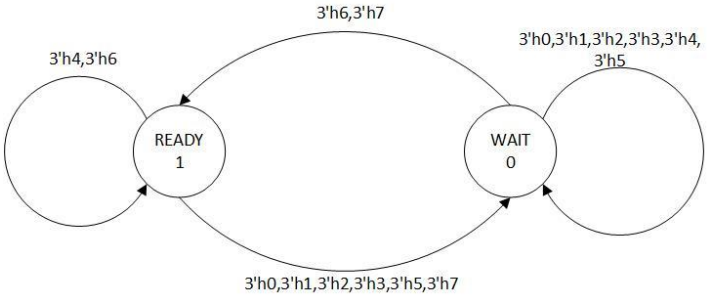


Figure 3.11 State diagram of the FSM used in the circuit

The FSM used is an example of Moore's machine. It is having two states named READY and WAIT which gives output 1 and 0 respectively. The RTL level block diagram representation of MCP formulation technique is given if figure:

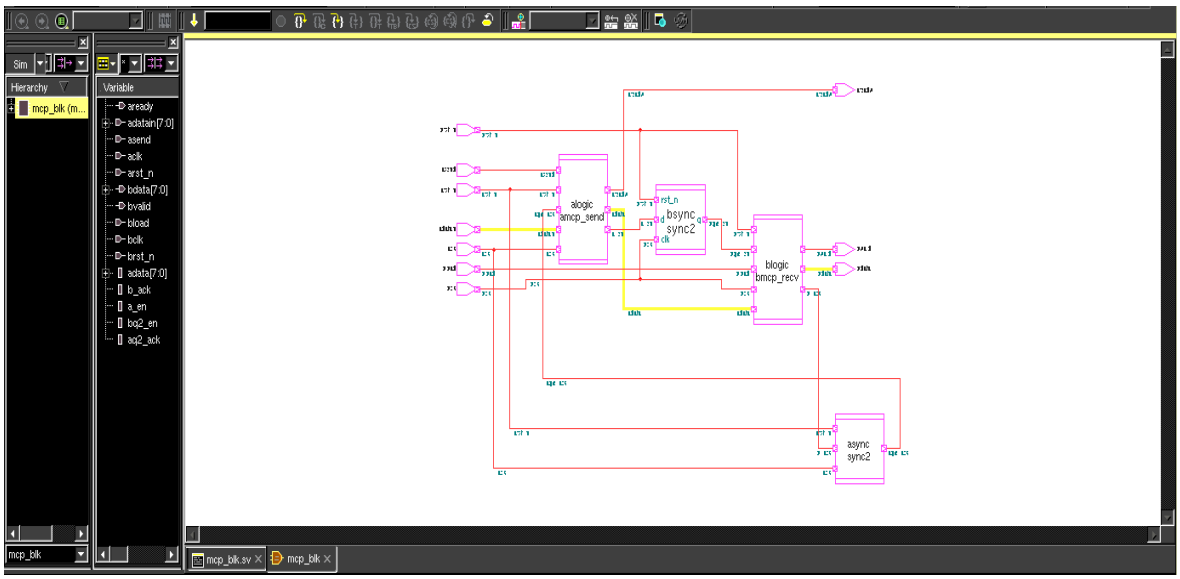


Figure 3.12 RTL block diagram of MCP technique

Advantage:

a. Total design occupies less area than the counter part i.e. FIFO synchronizer.

Disadvantage:

a. Dynamic power consumption is more comparatively.
b. Time delay between data sent and data getting received is more. So MCP method is slow comparatively.

3.5 Multi-Bit CDC Signal Passing Using FIFO Synchronizer

Another technique to pass multiple control and data bits across CDC boundaries involve use of a 1-deep two register FIFO. At one end data will be sent and at one end data will be fetched. Am FIFO implementation is shown is figure 3.13. Data is written into the FIFO in wclk domain. On the other side data is read from the FIFO in the rclk domain.

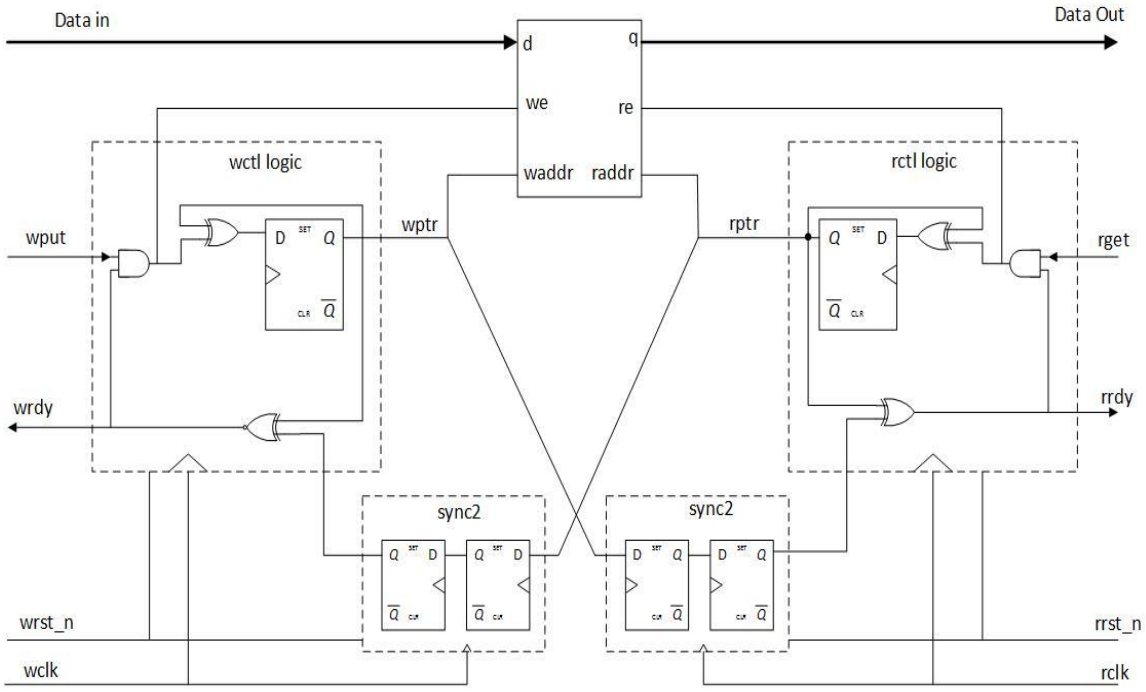


Figure 3.13 Block diagram of FIFO technique

A 2-deep FIFO memory is used in this case i.e. FIFO can at a time store 2 data values each with 8 bits. “wptr” and “rptr” are two pointers in wclk and rclk respectively. Signal “wput” is made high when a data is sent. But data is written at “waddr” location when “we” signal becomes high i.e when we get high value on acknowledgement signal “wrdy”. Also “wptr” is synchronized using a two flip-flop synchronizer and the synchronized output “rrdy” is used as an indication that data can be read now. “rget” signal is made high to read data. The “rptr” signal toggles for each transtion on “re” from 0 to 1. “rptr” signal is synchronized and the output is used to get an acknowledgement output “wrdy” which indicated that next data set can be sent now.

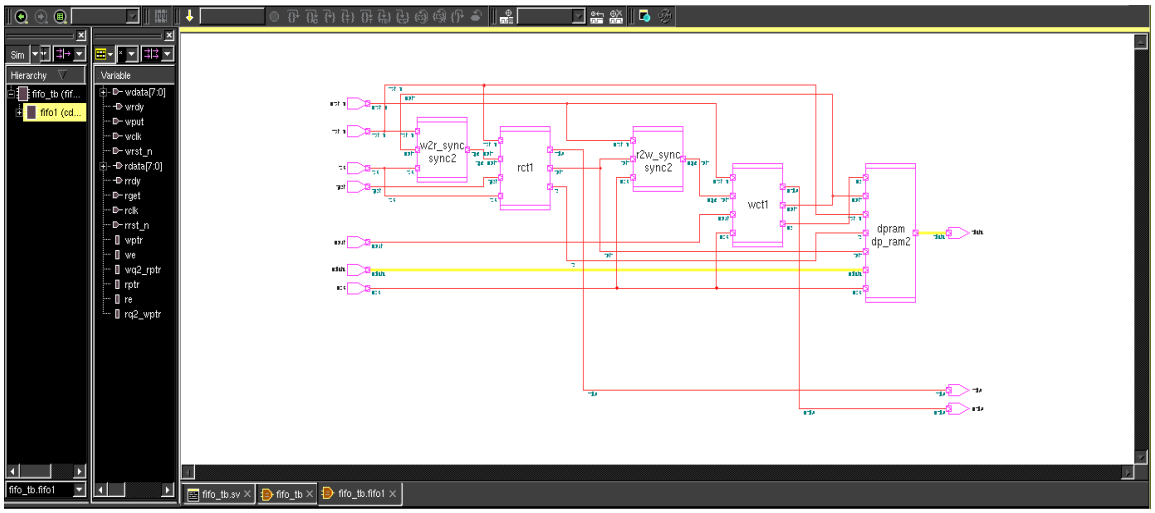


Figure 3.14 RTL block diagram of FIFO technique

Figure 3.14 shows the RTL level block diagram shown by the VCS tool. It is not full schematic. It just represent different ports and different blocks.

Advantage:

a. Less clock cycle required for data transmission as compared to MCP so fast.

Disadvantage:

c. FIFO occupies more area than that of MCP technique.

d. We have to take into consideration of the FIFO full and empty conditions which was not required in MCP.

Chapter 4

Power Estimation on RTL

4.1 Requirement of Power Estimation of RTL

Low power consumptions is the most important factor of consideration now a days. Designs are becoming more complex and the component count, in turn transistor count is increasing. With the increase in device density the power consumption is also increasing. But there is always a limitation to the power supply. The less is the power consumed the more is the battery life and the more reliable the device is. Now a days there are many low power methodologies implemented in VLSI designs to reduce static and dynamic power consumption like clock gating, power gating etc. For all the implementations separate circuitry are added to RTL making the total DUT more complex and more time taking to analyze.

Like the early detection of CDC bugs are done by doing a CDC checks on RTL using different tools, it is also necessary to do a power estimation also in early phase of design cycle. Because if the required power consumption range is not satisfied then we cannot go ahead with that design. If at any post silicon stage we catch that power criteria is not met and we need to redesign then we have to go to the RTL again do required coding which is a very large amount of re-do task. It takes lot of iteration to finally get the design ready with desired power intent.

Power analysis at RTL reduces lot of redesigning efforts, time, and cost too. So RTL is the best stage to do all analysis. The power check involves estimation of different power consumed in a design like static power, dynamic power, clock power, latch power, leakage power etc. If any power reduction methodology is used in our design then it power analysis also gives us the efficiency of that implementation. So we can compare between different methods and decide which technique is more reliable for our design. So RTL power estimation helps designer a lot.

4.2 Calculation of Switching Activity

The term “activity” as it refers to the net’s switching activity, can be defined as the ratio of the frequency at the net to the input clock frequency. Activity is calculated by taking the total toggles on that net, dividing it by the total number of toggles on the clock net for

the same period of time and multiplying that number by 2. The activity factor of a group of nets can be calculated by averaging the activities of all the named nets in the group.

For example, suppose the system clock frequency = f. Let $f_{sw} = \alpha f$, where α = activity factor.

- If the signal is a clock, $\alpha = 1$
- If the signal switches once per cycle, $\alpha = ½$

In the execution of the vector analysis, each group will have average activity values as a function of time i.e. $f(t)$. The value of the average activity is calculated by the activity value for all nets in that group from the simulation data and computing their average value. The nets include ports of modules as well as local nets. A point to remember here is, a group is not only a module but all the subsequent children instantiations inside the module.

Switching activity factor has contribution in dynamic power. The formulae for calculating dynamic power is given as

$$P_{switching} = \alpha C V_{DD}^2 f$$

where, α = activity factor

C = total capacitance of the Net

V_{DD} = supply voltage

f = frequency of the clock

So from the above formula we can conclude that switching activity is a contributing factor for dynamic power comsumption. The more is the switching activity of the net the more is the dynamic power consumed by the net.

4.3 Average Power Calculation

Power dissipation in CMOS circuits comes from two components

1. Static dissipation due to
    - sub threshold conduction through OFF transistors
    - tunneling current through gate oxide
    - leakage through reverse-biased diodes
    - contention current in ratioed circuits
2. Dynamic dissipation due to
    - charging and discharging of load capacitances
    - short circuit current while both PMOS and NMOS networks are partially ON

So the total power is the static power and dynamic power.

$$P_{Total} = P_{Static} + P_{Dynamic}$$

We have discussed most of the static dissipation factors before. Below 130nm static power is rapidly becoming a primary design issue eventually, static power dissipation may become comparable to dynamic power Ratioed circuits (e.g. pseudo circuits) have more static dissipation.

4.3.1. Static Power Consumption

Static power dissipation is the power loss during static state which means the net value is constant at either V_{cc} or at ground. Static power can be calculated by the formulae

$$P_{static} = V_{cc} * I_{cc}$$

4.3.2. Dynamic Power Consumption

Primary source of dynamic dissipation is charging of the load capacitance. Suppose load C is switched between V_{DD} and GND at average frequency f_{sw}. Over time T, load is charged and discharged Tf_{sw} times. In one complete charge/discharge cycle, a total charge of Q = CV_{DD} is transferred between V_{DD} and GND.

The average dynamic power dissipation is

$$P_{dynamic} = \frac{1}{T}\int_0^T i_{DD}(t)V_{DD}\,dt = \frac{V_{DD}}{T}\int_0^T i_{DD}(t)dt$$

Taking the integral of the current over interval T as the total charge delivered during time T

$$P_{dynamic} = \frac{V_{DD}}{T}[Tf_{sw}CV_{DD}] = CV_{DD}{f_{sw}}^2$$

Not all gates switch every clock cycle, so the above quantity is multiplied by α. For clock α=1, for data maximum is α=0.5, empirically static CMOS has α=0.1 Also due to non-zero input rise and fall times (slew), both NMOS and PMOS will be ON. Causes short circuit current that depends on input slew and output capacitance.

4.4 Introduction To Power Artist Tool

Power estimation at RTL stage is done with the help of a tool. There are many tools available in market. Some of them are

- Design Power (Synopsys)
- PowerArtist (ANSYS Apache)
- InCyte Chip Estimator (Cadence)

The basic mechanism is almost same for all the tools just the difference is that they are provided by different vendors. For my project I have used ANSYS PowerArtist tool which runs with Apache license. It is a very popular tool for power estimation. RTL designers working on different variety of applications, from mobile to computer's CPU to networking to automotive ICs, use this PowerArtist tool to optimize the designs throughout the development cycle. It enables RTL-to-GDS design for different power methodology by providing early RTL power estimation and analysis-driven power reduction capabilities.

The Embedded PowerArtist Calibrator and Estimator (PACE) technology ensures that early RTL power estimation tracks the final gate-level power numbers. PACE supports wide range of process technology nodes and also design styles. It helps the R&D teams to confidently make design trade-off decisions using early RTL power estimates. This drifts the methodology away from time-consuming and tedious gate-level power analysis, a difficult-to-do process that is more often done late in the late phase of design cycle.

The area of power estimation involves many fields like:

- Generating activity wave form
- Calculating average power
- Calculating time based power
- Calculating clock power
- Getting efficiency of power reduction methodologies
- Estimation of area occupied

These estimation helps designer to have over all porformance idea of the the design so that proper descision can be made about the design,which is going to be implemented.

4.5 Power Analysis Flow

The basic flow regarding how the PowerArtist tool works and how it do analysis to generate results in terms of different power reports is depicted in the following figure.

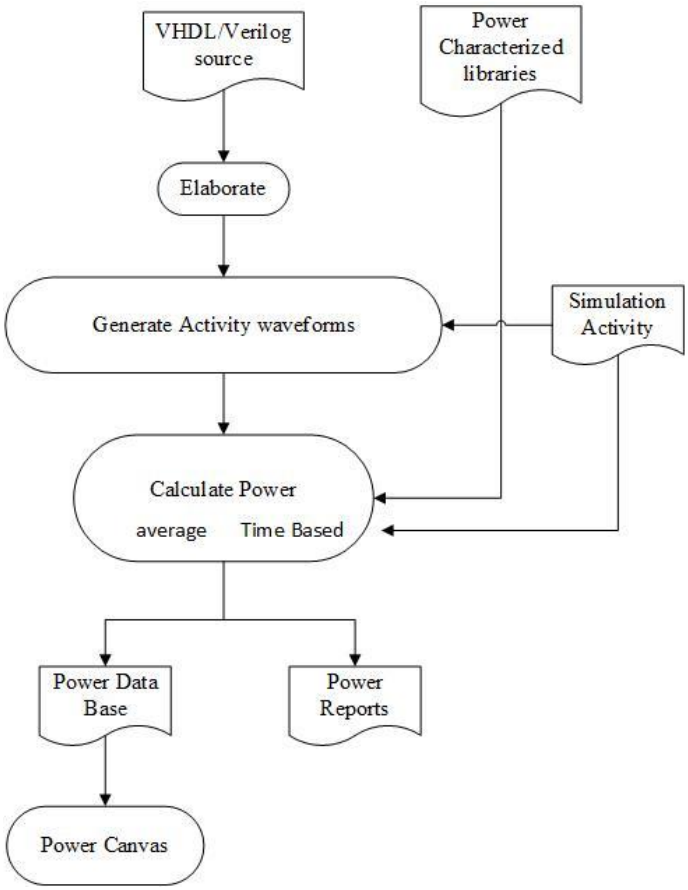


Figure 4.1 Flow chart explaining process of power analysis

The major steps involved in the power analysis flow are explained below

1. Elaborate:

Compiles the HDL design description into an internal binary format called the scenario file. This scenario file can be used recursively if there is no RTL change.

2. Generate Activity Waveforms:

Analyzes the activity file and generates the waveform files representing the activity in the design. This is where the toggling component comes into the picture.

3. Calculate Flop Clock Activity:

Monitors the activity at the input clock pins of registers.

4. Calculate Power:

a. analysis_type_average — performs an average based power analysis.

b. analysis_type_time_based — performs a time based power analysis.

4.6 Schematic Representation Of MCP Implementation In Power Artist

PowerArtist tool performs dummy synthesis in its environment. There is built in library provided with the tool which helps to perform this synthesis. All components of the RTL will be replaced with real gates and accordingly power and area will be estimated by the tool.

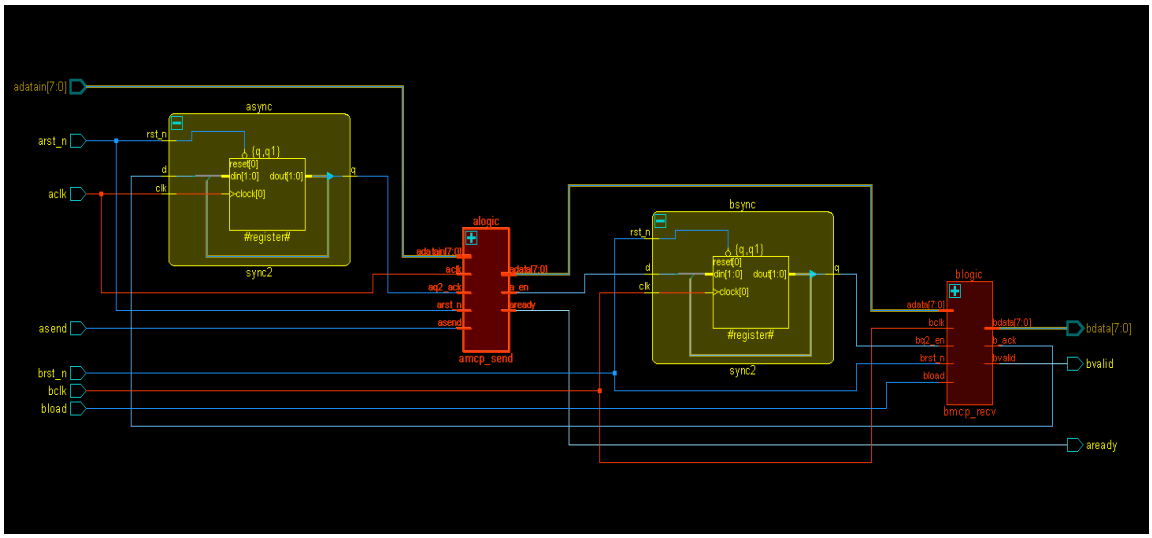


Figure 4.2 Colorized Shematic of MCP in PowerArtist

The figure shown above is the full schematic of the MCP technique with all component synthesized. Also the schematic is colorized by different colors. The color coding is according to power consumption. Module consuming highest power is denoted by Red and moderate power consuming modules are denoted by Green color and components having least power consumption are denoted by Blue color. This color coding helps us to understand the overall power distribution at a glance, without going through the textual reports. The expanded schematic of “alogic” and “blogic” blocks are shown in figure 4.3 and figure 4.4 respectively.

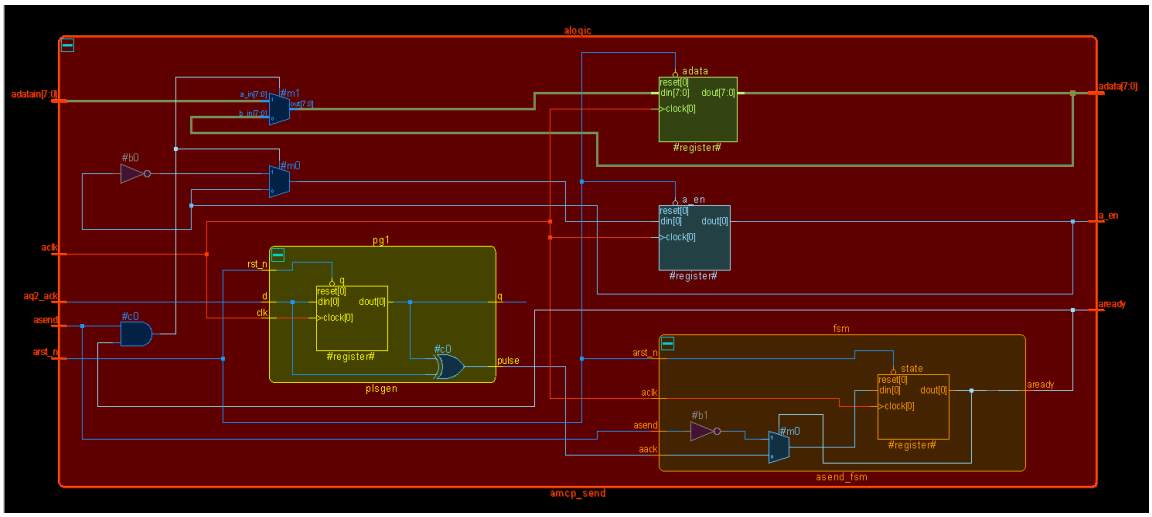


Figure 4.3 Colorized Shematic of alogic block in PowerArtist

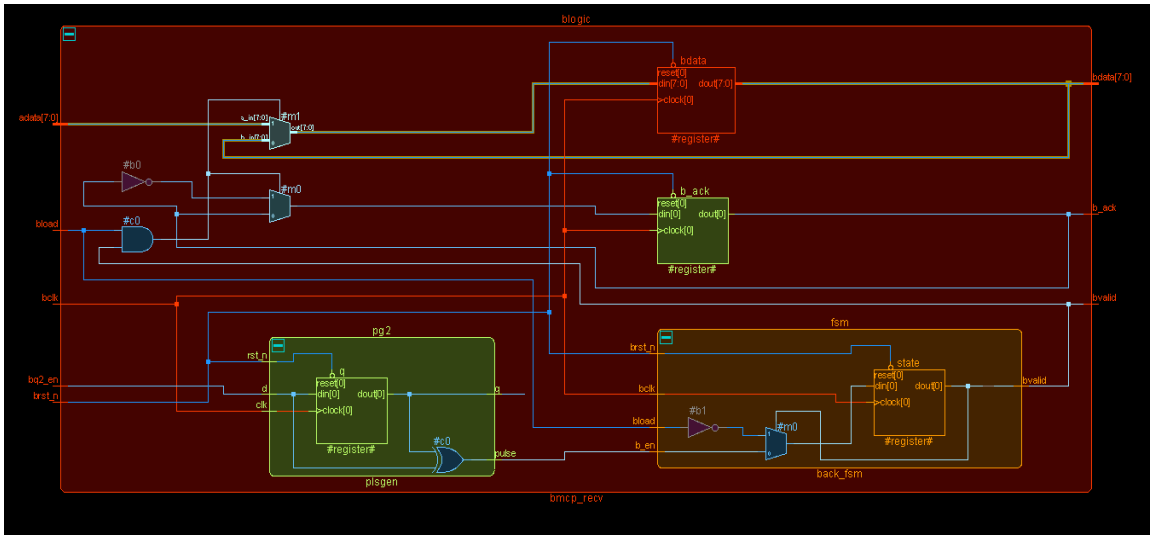


Figure 4.4 Colorized Shematic of blogic block in PowerArtist

4.6 Schematic Representation Of FIFO Synchronization In Power Artist

As explained in the above section PowerArtist tool helps us to estimate power consumption of the RTL by performing dummy synthesis. Figure 4.5 shows full schematic of FIFO synchronizer which consists of three sub blocks “dpram”, “wct1” and “rct1”. The schematic of each sub block is given in subsequent three figures.

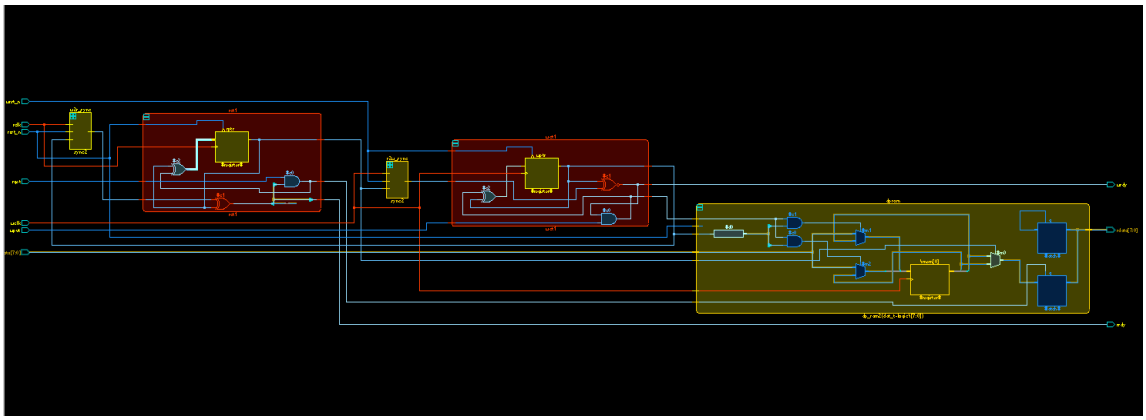


Figure 4.5 Colorized Shematic of FIFO synchronizer in PowerArtist

As shown in the above figure the “dpram” block i.e the FIFO memory element consumes less power than that of “wct1” and “rct1” blocks. Figure 4.6 shows expanded schematic of “dpram” circuit. The most power consuming blocks are “wct1” and “rct1” block whci are shown in figure 4.7 and figure 4.8.

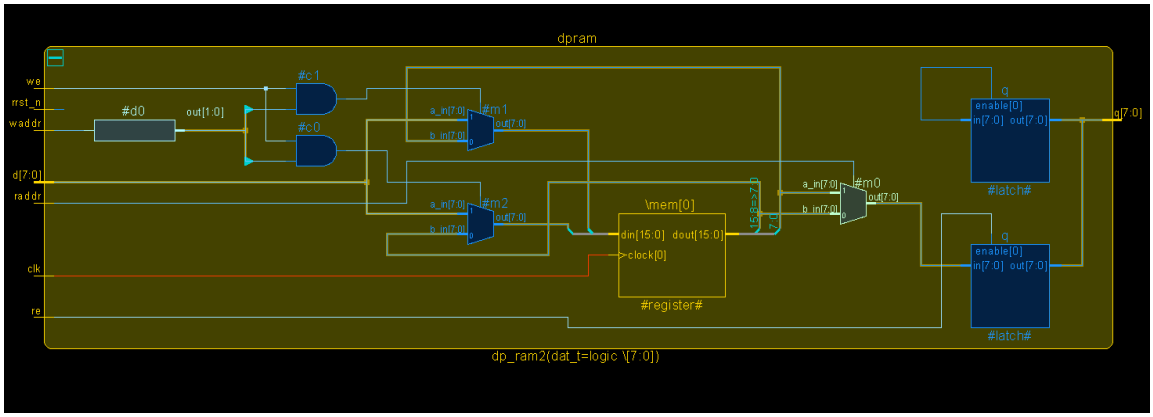


Figure 4.6 Colorized Shematic of dpram block in PowerArtist

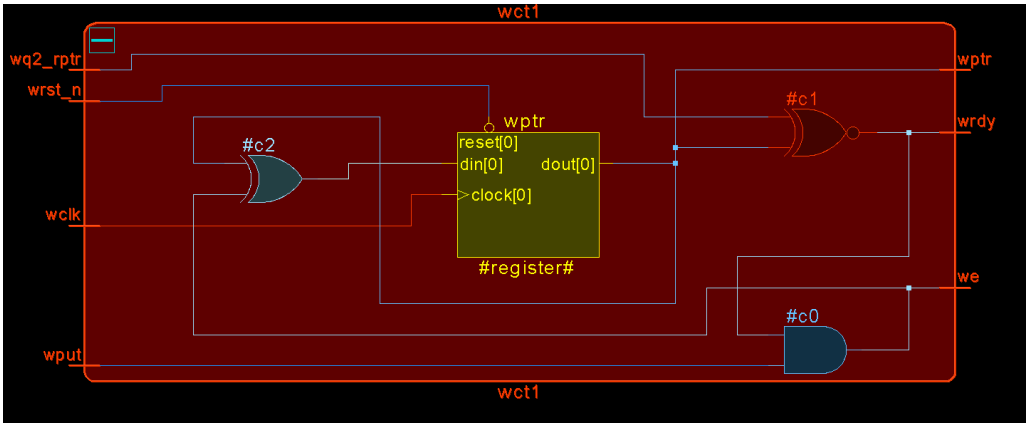


Figure 4.7 Colorized Shematic of wct1 block in PowerArtist

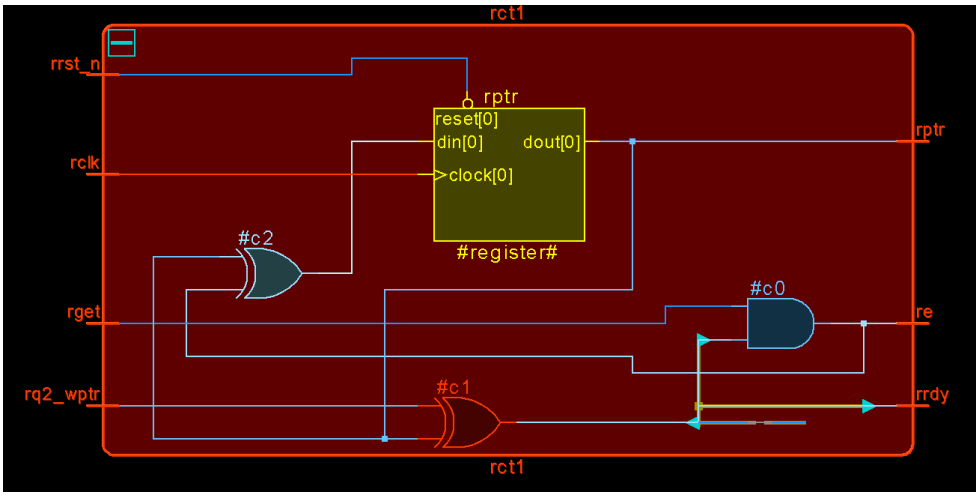


Figure 4.8 Colorized Shematic of rct1 block in PowerArtist

hese netlist produced are used for the calculation of power and area according to the library used. The results of power estimation and area estimation are explained in next chapter.

Chapter 5

Results of RTL Quality Checks

5.1 Simulation Output Of Data Transfer By MCP Technique And FIFO Mechanism

Using VCS simulator simulations were performed on both MCP and FIFI design and simulation outputs were observe as below.

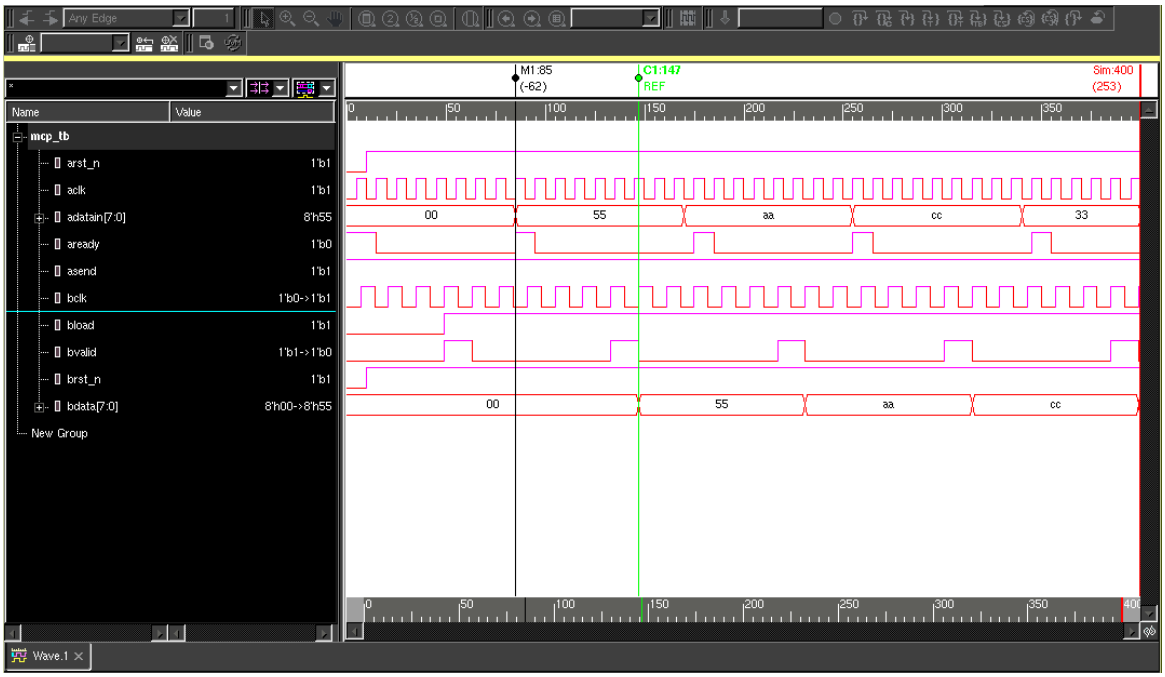


Figure 5.1 VCS simulation output waveform of MCP formulation

Simulation output of MCP formulation technique is shown in figure 5.1. The time delay between data being sent and data getting received is calculated to be 147-85=62 time unit. Time scale taken is 1ns/1ns. So absolute delay is 62ns.

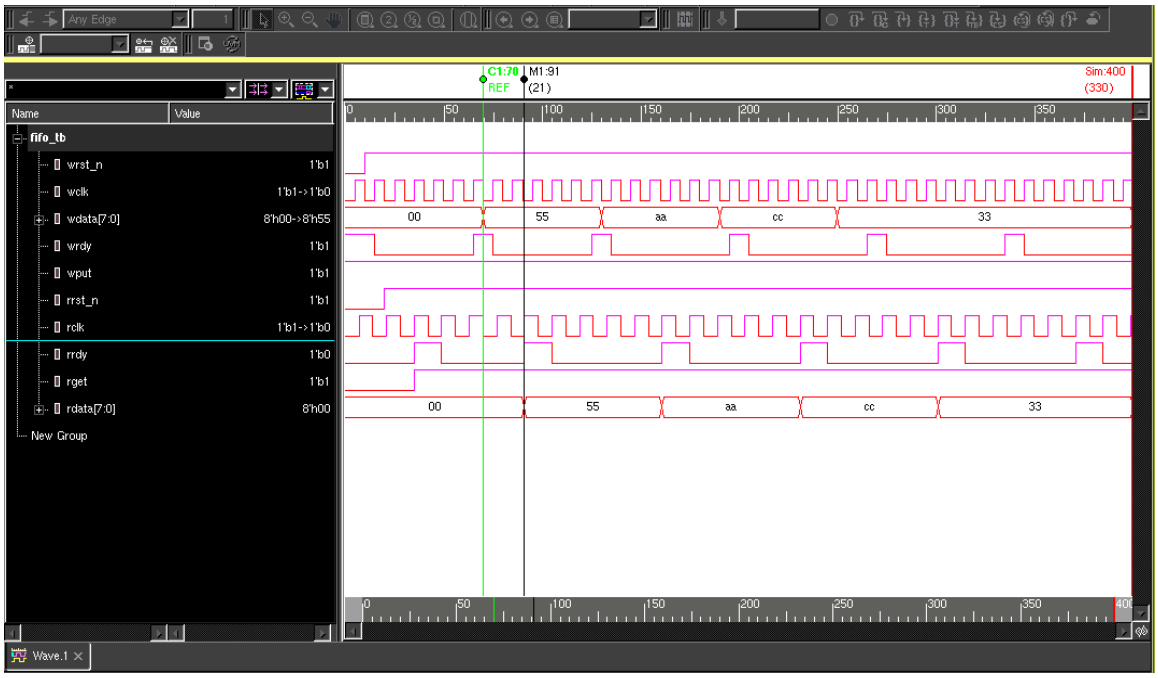


Figure 5.2 VCS simulation output waveform of FIFO synchronizer

Figure 5.2 shows simulation output of FIFO synchronization technique used. The delay is calculated to be 91-70=21 time unit. Since time scale used is 1ns/1ns the delay is 21ns.

5.1.1 Delay Comparision Between MCP And FIFO

From the output waveforms that we got for MCP design and FIFO design we can calculate the delay in both the technique. Delay in this case is nothing but the time gap between the data getting uploaded in the sending clock domain and the data gettting sampled as output in the receiving clock domain. The delay is tabulated as below

Table 5.1 Delay comparison of MCP and FIFO design

Design	MCP	FIFO
DELAY(ns)	62	21

From the above table it is concluded that data transmission by MCP takes more time than FIFO technique. So using FIFO is better in terms of delay,when taken into consideration. But one thing we have to ensure that FIFO full and empty conditions are taken care of. So the conclusion here is that FIFO is faster than MCP technique.

5.2 Activity Output Waveform Of MCP Technique And FIFO Technique

Using Power Artist tool the activity waveform is generated for both MCP and FIFO synchronization methodology. The x axis is time in ns unit and the y axis is frequency in Hz.

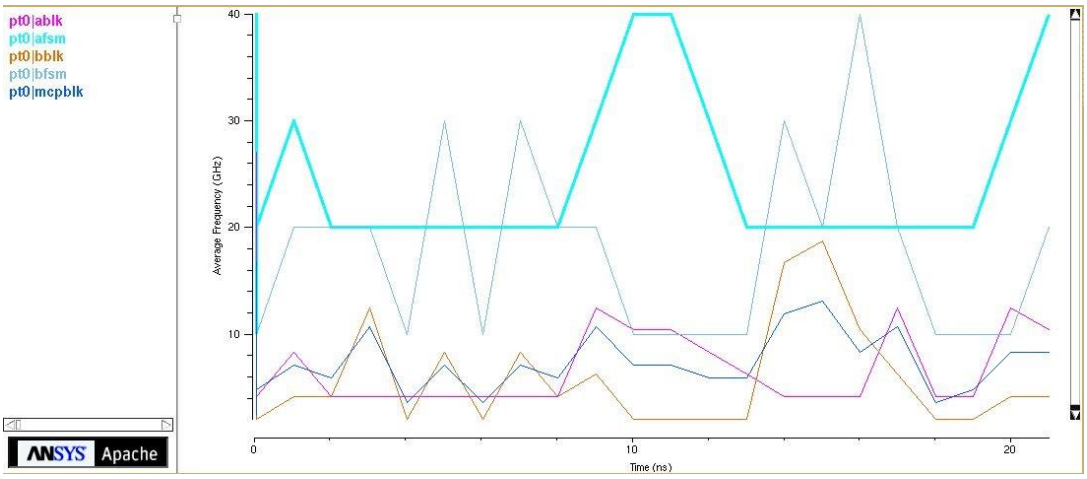


Figure 5.3 Activity waveform of MCP formulation

Figure 5.3 shows activity waveform of MCP technique implemented eairlier. The activity is quite random and the maximum activity is found to be 40 Hz.

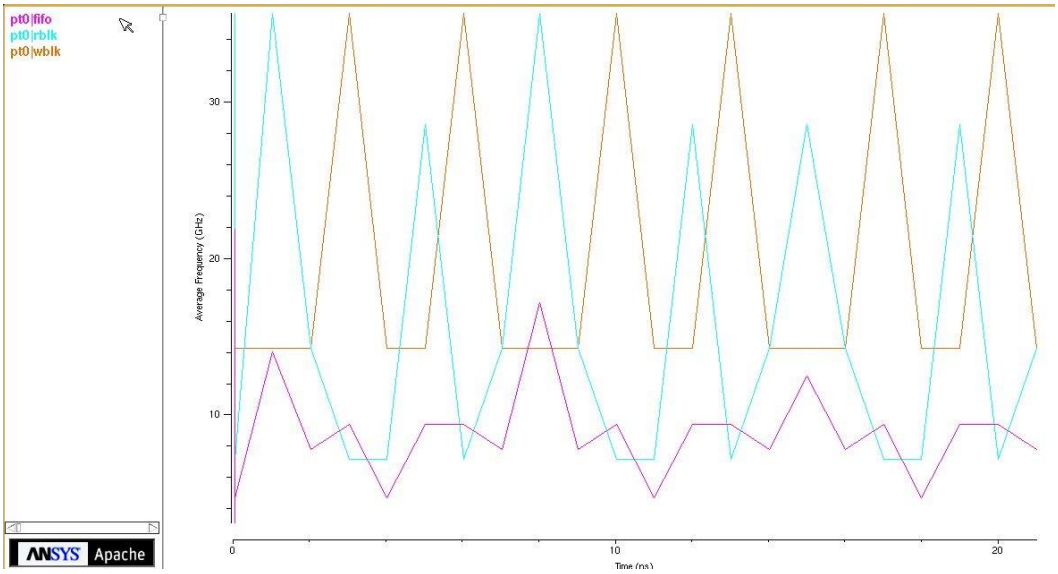


Figure 5.4 Activity waveform of FIFO synchronization

The above figure shows activity of FIFO synchronizer. The activity is quite periodic and the maximum activity is found to be 35 Hz.

5.3 Textual Reports Of Power Analysis

Power estimation at RTL helps us to calculate power of our design. This power may not be the actual power which willl be consumed once we will fabricate the chip. So the whole point of estimating power at RTL is that, we can compare in between multiple possible designs and choose the optimum one according to our requirement. Since all estimations will be done using same library, this method is very much efficient in performing quality checks from power prospective. Also for a single design, we can know which module is more active and which module is consuming more power and vice versa.

```
1. Total power consumption
==========================

                                                    Power(Watts)
Power contribution                                  Static  Dynamic    Total
----- ------------                                  ------  -------    -----
Internal power
   Internal register power                           180nW   1.69mW   1.69mW
   Internal latch power                                 0W       0W       0W
   Internal memory power                                0W       0W       0W
   Other internal power                              68.6nW   5.29uW   5.36uW
   Total internal power                               248nW    1.7mW    1.7mW
Pad power                                               0W       0W       0W
Clock power                                         8.41nW     32uW     32uW
Inferred Buffer power                               8.81nW   8.69nW   17.5nW

Total power                                          266nW   1.73mW   1.73mW
```

Figure 5.5 Total power consumption in MCP formulation technique

```
1. Total power consumption
==========================

                                                    Power(Watts)
Power contribution                                  Static  Dynamic    Total
----- ------------                                  ------  -------    -----
Internal power
   Internal register power                           133nW   17.4uW   17.5uW
   Internal latch power                                 0W       0W       0W
   Internal memory power                                0W       0W       0W
   Other internal power                              89.4nW   1.37mW   1.37mW
   Total internal power                               223nW   1.39mW   1.39mW
Pad power                                               0W       0W       0W
Clock power                                         8.41nW   30.5uW   30.5uW

Total power                                          231nW   1.42mW   1.42mW
```

Figure 5.6 Total power consumption in FIFO synchronization technique

Figure 5.5 and figure 5.6 show the power contribution in MCP and FIFO method respectively. Powers are classified in terms of static, dynamic and total power consumed by internal components as well as clock signal. More detailed reports regarding power estimation, classified as register, latch, buffer and clock power are depicted in figures below.

```
9. Power consumption by model/gate type
=======================================

                                    Power(Watts)
Component                  Model    Cell     Static  Dynamic   Total
                                    Count
---------                  -----    ------   ------- -------   -----
mcp_blk                             30     266nW   1.73mW   1.73mW
  Register Power                    10     180nW   1.69mW   1.69mW

  Latch Power                        0       0W       0W       0W

  Memory Power                       0       0W       0W       0W

  Other Power                       16   68.6nW   5.29uW   5.36uW

  Pad Power                          0       0W       0W       0W

  Clock Power                        2   8.41nW     32uW     32uW

  Inferred Buffer Power              2   8.81nW   8.69nW   17.5nW

                            and      2    1.3nW   3.01uW   3.01uW
                        connect      2       0W       0W       0W
                    connect_inv      4       0W       0W       0W
                         mux21       6   61.3nW   1.85uW   1.91uW
                       register     10    180nW   1.69mW   1.69mW
                            xor      2      6nW    431nW    437nW
Leaf Driver (Clock Buffer Tree)      2   8.41nW     32uW     32uW
  Leaf Driver (Net Buffer Tree)      2   8.81nW   8.69nW   17.5nW
-------------------------------------------------
```

Figure 5.7 power consumption classified interms of gate/model type for MCP

```
9. Power consumption by model/gate type
=======================================

                                    Power(Watts)
Component                  Model    Cell     Static  Dynamic   Total
                                    Count
---------                  -----    ------   ------- -------   -----
cdc_syncfifo                        22     231nW   1.42mW   1.42mW
  Register Power                     5     133nW   17.4uW   17.5uW

  Latch Power                        2       0W       0W       0W

  Memory Power                       0       0W       0W       0W

  Other Power                       13   89.4nW   1.37mW   1.37mW

  Pad Power                          0       0W       0W       0W

  Clock Power                        2   8.41nW   30.5uW   30.5uW

  Inferred Buffer Power              0       0W       0W       0W

                            and      4   2.18nW   2.69uW    2.7uW
                        connect      1       0W       0W       0W
                        decoder      1   1.83nW    348nW    350nW
                          latch      2       0W       0W       0W
                         mux21       3   73.2nW    862nW    935nW
                       register      5    133nW   17.4uW   17.5uW
                           xnor      1      3nW    683uW    683uW
                            xor      3   9.17nW    683uW    683uW
Leaf Driver (Clock Buffer Tree)      2   8.41nW   30.5uW   30.5uW
-------------------------------------------------
```

Figure 5.8 power consumption classified interms of gate/model type for FIFO

Also for more clarification we can look into power report shown in figure 5.9 and figure 5.10. These reports shows power consumed by each and every primitive component used along with the component type i.e. whether gate or a MUX or a register etc.

mcp_blk	user	248nW	1.7mW	1.7mW
alogic	user	110nW	577uW	577uW
#b0	connect_inv	0W	0W	0W
#c0	and	642pW	1.5uW	1.5uW
#m0	mux21	24.7nW	546nW	571nW
#m1	mux21	3.09nW	142nW	145nW
a_en	register	6.93nW	916nW	923nW
adata	register	55.5nW	5.09uW	5.15uW
fsm	user	9.58nW	568uW	568uW
#b0	connect	0W	0W	0W
#b1	connect_inv	0W	0W	0W
#m0	mux21	2.91nW	238nW	241nW
state	register	6.67nW	567uW	567uW
pg1	user	9.93nW	957nW	967nW
#c0	xor	3nW	215nW	218nW
#n12	register	6.93nW	742nW	749nW
async	user	13.9nW	1.57uW	1.58uW
{q1,aq2_ack}	register	13.9nW	1.57uW	1.58uW
blogic	user	110nW	1.12mW	1.12mW
#b0	connect_inv	0W	0W	0W
#c0	and	660pW	1.51uW	1.51uW
#m0	mux21	24.6nW	545nW	570nW
#m1	mux21	3.07nW	142nW	145nW
b_ack	register	6.93nW	752nW	759nW
bdata	register	55.5nW	546uW	546uW
fsm	user	9.67nW	569uW	569uW
#b0	connect	0W	0W	0W
#b1	connect_inv	0W	0W	0W
#m0	mux21	2.94nW	235nW	238nW
state	register	6.74nW	568uW	568uW
pg2	user	9.93nW	795nW	805nW
#c0	xor	3nW	216nW	219nW
#n12	register	6.93nW	579nW	586nW
bsync	user	13.9nW	1.24uW	1.25uW
{q1,bq2_en}	register	13.9nW	1.24uW	1.25uW
Total internal power		248nW	1.7mW	1.7mW

Figure 5.9 power consumed by each component for MCP formulation

		Power(Watts)		
Component	Model	Static	Dynamic	Total
cdc_syncfifo	user	223nW	1.39mW	1.39mW
dpram	user	168nW	14.8uW	15uW
#c0	and	426pW	921nW	921nW
#c1	and	426pW	921nW	922nW
#d0	decoder	1.83nW	348nW	350nW
#l0	latch(0)	0W	0W	0W
#l1	latch(0)	0W	0W	0W
#m0	mux21	23.5nW	196nW	220nW
#m1	mux21	24.9nW	332nW	357nW
#m2	mux21	24.9nW	334nW	359nW
mem	register	91.8nW	11.8uW	11.9uW
r2w_sync	user	13.9nW	1.55uW	1.57uW
{q1,wq2_rptr}	register	13.9nW	1.55uW	1.57uW
rctl	user	13.7nW	684uW	684uW
#b0	connect	0W	0W	0W
#c0	and	673pW	228nW	229nW
#c1	xor	3nW	682uW	682uW
#c2	xor	3.08nW	350nW	353nW
rptr	register	6.95nW	1.7uW	1.71uW
w2r_sync	user	13.9nW	1.23uW	1.24uW
{q1,rq2_wptr}	register	13.9nW	1.23uW	1.24uW
wctl	user	13.7nW	685uW	685uW
#c0	and	651pW	624nW	625nW
#c1	xnor	3nW	683uW	683uW
#c2	xor	3.09nW	350nW	354nW
wptr	register	6.96nW	1.17uW	1.18uW
Total internal power		223nW	1.39mW	1.39mW

Figure 5.10 power consumed by each component for FIFO synchronization

5.4 Textual Reports Of Area Analysis

Apart from power analysis as shown if previous section, we can make an estimation of area occupied by the total design also. Since our tool does a dummy synthesis in its environment, the netlist occupies finite area. This area estimation is also done by

PowerArtist tool. Figure 5.11 and figure 5.12 shows the area occupied by the RTL of our both designs. It shows total area along with area of each and every component.

Component	Width	Height	Regs	Gates
mcp_blk	19	19	26	268
alogic	12.6	12.6	11	118
#c0	1.13	1.13	0	1
#m0	5.54	5.54	0	24
#m1	1.96	1.96	0	3
a_en	3.25	3.25	1	8
adata	9.19	9.19	8	62
fsm	3.79	3.79	1	11
#m0	1.96	1.96	0	3
state	3.25	3.25	1	8
pg1	3.67	3.67	1	10
#c0	1.7	1.7	0	2
#n12	3.25	3.25	1	8
async	4.6	4.6	2	16
{q1,aq2_ack}	4.6	4.6	2	16
blogic	12.6	12.6	11	118
#c0	1.13	1.13	0	1
#m0	5.54	5.54	0	24
#m1	1.96	1.96	0	3
b_ack	3.25	3.25	1	8
bdata	9.19	9.19	8	62
fsm	3.79	3.79	1	11
#m0	1.96	1.96	0	3
state	3.25	3.25	1	8
pg2	3.67	3.67	1	10
#c0	1.7	1.7	0	2
#n12	3.25	3.25	1	8
bsync	4.6	4.6	2	16
{q1,bq2_en}	4.6	4.6	2	16
Total Counts			26	268
Total Internal Area	360			
Total Clock Tree Area	6.4			
Total Inferred Buffer Tree Area	6.4			
Total Net Routing Area	3.27K			
Total Area (Gates+Routing)	3.64K			

Figure 5.11 Area occupied by each component for MCP technique

Component	Width	Height	Regs	Gates
cdc_syncfifo	18.4	18.4	22	256
dpram	16.1	16.1	16	199
#c0	1.13	1.13	0	1
#c1	1.13	1.13	0	1
#d0	2.71	2.71	0	1
#m0	5.54	5.54	0	24
#m1	5.54	5.54	0	24
#m2	5.54	5.54	0	24
mem	12.6	12.6	16	124
r2w_sync	4.6	4.6	2	16
{q1,wq2_rptr}	4.6	4.6	2	16
rctl	4.2	4.2	1	13
#c0	1.13	1.13	0	1
#c1	1.7	1.7	0	2
#c2	1.7	1.7	0	2
rptr	3.25	3.25	1	8
w2r_sync	4.6	4.6	2	16
{q1,rq2_wptr}	4.6	4.6	2	16
wctl	4.2	4.2	1	13
#c0	1.13	1.13	0	1
#c1	1.7	1.7	0	2
#c2	1.7	1.7	0	2
wptr	3.25	3.25	1	8
Total Counts			22	256
Total Internal Area	338			
Total Clock Tree Area	6.4			
Total Inferred Buffer Tree Area	0			
Total Net Routing Area	4.76K			
Total Area (Gates+Routing)	5.1K			

Figure 5.12 Area occupied by each component for FIFO synchronization

Chapter 6

Conclusion and Future scope

6.1 Conclusion

Clock domain crossing must be checked and fixed at the RTL stage. Simple two D flip-flop is used to pass single bit data. But for data bus two D flip-flop synchronizer should never be used for individual bits. It causes unrequired functional problems like data loss and incoherency, even if it resolves metastability issue. In this project some of such cases are also discussed. Multiple data crossing has to handle by special methodologies. Either handshaking mechanism like MCP formulation technique or FIFO synchronization method is used for multiple bit signal to cross between clock domains. This project has shown a generic implementation of MCP and FIFO using system verilog coding. Also an RTL quality comparison has been performed on the basis of delay, power consumption and area occupied.

Table 6.1 Characteristic comparison of MCP and FIFO design

Characteristics	MCP	FIFO	Preferred
Delay (ns)	62	21	FIFO
Switching Activity MHz)	40	35	FIFO
Total Power (mW)	1.73	1.42	FIFO
Area (nm sq.)	3.64K	5.1K	MCP

The table shows the result of RTL quality checks. The area of analysis are delay, switching activity, total power consumption and area occupied by both circuit. In terms of delay FIFO is preferred over MCP because it has high speed. Also FIFO has less switching activity and thus consumes less dynamic power than that of MCP. But with FIFO main disadvantage is that it occupies more area than that of MCP technique. The major area is consumed by the DRAM. Also with increase in RAM size the area again increase. So when area is a constraint to consider, then FIFO is less preferred. But most of the design prefers FIFO now a days. So according to requirement a tradeoff has to be made to choose which circuit has to be used.

6.2 Future Scope of the project

This project can meet the requirement of customers effectively because of it flexibility and reusability. But it can have further scope of enhancements like

1. Power reduction technique can be implemented in the modules consuming more power.
2. The delay in between data being sent and getting received can be minimized by adding additional methodologies like synchronization latency hiding mechanism. In this method synchronization, which impose the major delay, is overlapped by computation that require received data only.

References

[1] Clifford E. Cummings “Clock Domain Crossing (CDC) Design & Verification Techniques using SystemVerilog”, *SNUG2008-www.sunburstdesign.com/papers/CummingsSNUG2008 Boston_CDC.pdf*

[2] Naghmeh Karimi, and Krishnendu Chakrabarty “Detection, Diagnosis, and Recovery from Clock-Domain Crossing Failures in Multiclock SoCs”, *IEEE transactions on computer-aided design of integrated circuits and systems,* vol. 32, no. 9, September 2013.

[3] R. Ginosar, “Metastability and synchronizer: A tutorial,” *IEEE Des.Test Comp.*, vol. 28, no. 5, pp. 23-35, Sep. 2011.

[4] D. L. Liu and C. Svensson, "Power consumption estimation in CMOS VLSI chips", *IEEE Journal SSC,* vol. 29, no. 6, pp. 663-670, 1994

[5] D. J. Kinniment, A. Bystrov and A. V. Yakovlev, “Synchronization circuit performance”, *IEEE J. Solid-State Circuits,* vol. 37, no. 2, pp. 202-209, 2002

[6] M. Nemani and F. Najm, “High-level area and power estimation for VLSI circuits”, *IEEE Trans. Computer-Aided Design*, vol. 18, pp. 697–713, June 1999

[7] Saurabh Verma and Ashima S. Dabare, “Understanding clock domain crossing issues”, *white paper by Atrenta.*

[8] H.-K. Kim, L.-T. Wang, Y.-L. Wu, ·and W.-B. Jone, “Testing of synchronizers in asynchronous FIFO,” *J. Electron. Testing Theory Appl.*, vol. 29, no. 1, pp. 49–72, 2013

[9] Y. Li, B. Nelson, M. Wirthlin, "Synchronization Techniques for Crossing Multiple Clock Domains in FPGA-Based TMR Circuits," *IEEE Transactions on Nuclear Science*, Vol. 57, Issue 6, pp. 3506-3514, 2010.

[10] Hatture, S.; Dhage, S. "Open loop and closed loop solution for clock domain crossing faults", *Communication Technologies (GCCT), 2015 Global Conference on,* on page(s): 645 – 649

[11] N. Karimi, K. Chakrabarty, P. Gupta and S. Patil, "Testing of clock-domain crossing faults in multi-core system-on-chip", *proc. Asian Test Symp.*, pp. 7-14

[12] N. sharif, N. ramzan, F.K. Lodhi and O. hasan, "Quantitative Analysis of state-of-Art synchronizers: Clock domain crossing perspective", *IEEE conf.*, pp. 1-6

[13] Ankush Patharkar, Vivek E. Khetade, Dr.S.S. limaye, Ashish Bhopale and Akshay Patharkar, "Analysis of synchronizer, Data loss and Occurrence of Metastability", *IEEE conf.*, pp. 392-397

[14] R. Dobkin, R. Ginosar and C. Sotiriou, High Rate Data Synchronization in GALS SoCs, *IEEE Transactions on Very large Scale Integration (VLSI) Systems*, 14(10):1063-1074, Oct. 2006.

[15] C. Dike and E. Burton, "Miller and Noise Effects in a Synchronizing Flip-Flop," *IEEE Journal of Solid-State Circuits*.

[16] Al-bayati Zaid, O. At Mohamed, S. Rafay Hasan and Y. Savaria, "Design of a C-Element Based Clock Domain Crossing Interface", *IEEE, International conference on microelectronics (ICM)*

[17] C. Leong, P. Machado, V. Bexiga and J.P. Texeira, "Built-In Clock Domain crossing (CDC) test and Diagnosis in GALS systems", *IEEE conf.*, pp. 72-77

[18] J. Horstmann , E. H. and R. Coates, "Metastability behavior of CMOS master/slave flip-flops", *IEEE Trans. Circuits Syst.*, vol. 24, no. 1, pp. 146-157, 1992

[19] M. Berg, "Embedding asynchronous FIFO memory blocks in Xilinx Virtex series FPGAs targeted for critical space systems applications", *NASA Conf. Military Applications of Programmable Logic Devices (MAPLD)*, 2009, [online] Available: online

[20] L. Kleeman and A. Cantoni, "Metastable behavior in Digital Systems," IEEE D&T, 4(6):4-19, 1987.

[21] T.J. Chaney and C.E. Molnar, "Anomalous behavior of synchronizer and arbiter circuits," TComp, 22:421-422, 1973.

[22] S. Beer, et al. "An on-chip metastability measurement circuit to characterize synchronization behavior in 65nm", (ISCAS) - May 2011.

[23] D. Chen, D. Singh et al., "A comprehensive approach to modelling, characterizing and optimizing for metastability in FPGAs," FPGA 2010.

[24] J.Zhou, D.J.Kinniment, G. Russell, and A. Yakovlev, "A Robust Synchronizer Circuit", *in Proc. IEEE Comp. Soc. Ann. Symp. On VLSI (ISVLSI'06)*, pp. 442-443, 2006.

[25] R. L. Cline. "Method and circuit for improving metastable resolving time in low-power multi-state devices" US patent 5,789,945, February 27, 1996.