DIGITAL INTEGRATED CIRCUIT DESIGN:

"ASIC IMPLEMENTATION OF AN ADAPTIVE DIGITAL PREDISTORTER FOR LASERS ON ROF"

By

Charles Lim
Bachelor of Electrical Engineering with Honors
Ryerson University
Toronto, Canada, 2003

A project
presented to Ryerson University
in partial fulfillment of the
requirements for the degree of
Master of Engineering
in the Program of
Electrical and Computer Engineering

Toronto, Canada, 2006

Author's Declaration

I hereby declare that I am the sole author of this project.

I authorize Ryerson University to lend this project to other institutions or individuals for the purpose of scholarly research.

Signature

I further authorize Ryerson University to reproduce this project by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

Signature

Abstract

Title of Project: **"ASIC IMPLEMENTATION OF AN ADAPTIVE DIGITAL PREDISTORTER FOR LASERS ON ROF"**

Charles Lim, Master of Engineering, 2006

Project Directed By: Dr. Reza Sedaghat,
Electrical and Computer Engineering Department

Radio over fiber has become one of the most useful technologies for providing extended coverage of wireless communications services. ROF uses analog fiber optic links to distribute wireless radio signals from a central location to multiple remote locations where the added desired antennas are placed for stronger signal coverage. The adaptive predistortion technique of a LASER ROF chip is implemented using the digital IC design flow. The design flow can be separated into two main parts, namely, the RTL design / synthesis and the generation of the actual chip.

The first part in the design flow consists of generating the proper logical functionality of the IC using a hardware description language (HDL), namely VHDL or Verilog, and synthesizing the code to ensure proper operation. The second part in the design flow consists of floorplanning and physical layout of the ASIC.

Acknowledgements

The list of those to whom I would like to express my appreciation for their support and suggestions that allowed me to complete this project continues to grow. First of all I would like to thank my supervisor, Dr. Reza Sedaghat, who guided me along the way. It would not have been possible to complete this project without Dr. Sedaghat's help and guidance.

Then of course there were the combined efforts of our chip design team. I would like to thank Mr. Moon for his assistance and suggestions to problems that were encountered in my project.

And last but not least, I would like to thank my parents, siblings, and friends for their continued support during my study at Ryerson University.

Table of Contents

List of Tables

List of Figures

Summary

The design project is the implementation of an adaptive digital predistorter for LASERs on ROF links. This digital integrated circuit chip is part of a larger overall design that uses multiple digital and analog sections to implement the full adaptive predistortion technique. To create a better understanding of the IC and ASIC implementation itself, this report will cover the theories of digital ASIC design as well as adaptive digital predistortion technique theory. The implementation section will discuss the CAD tools used to implement the technique. Furthermore, theoretical design will be applied to the practical design of the IC.

Radio over fiber (ROF) refers to base stations being directly attached to fiber optics technology for cheaper yet faster data rates. Adaptive digital predistortion technique on LASERs for ROF technology usage is a new way of interfacing high speed long distance data to a wireless base station at lower costs. The issues with directly connecting the LASER output to the wireless base stations being used by ROF links include the LASER characteristics on the output being non-linear, which causes any analog type of input to be distorted, attenuated and noisy at the wireless base station end of the ROF link. This ASIC is created to solve the issues and limitations of such non-linear ROF link characteristics by using digital signal processing techniques along with a predetermined ROF link characteristic to predistort the signal being sent. This predistortion technique would ensure that the predistorted signal is distorted back after going through the ROF link, technically eliminating the non-linear characteristics of ROF link.

In order to appreciate the implementation of such an integrated circuit, it is necessary to create a certain level of understanding about the adaptive digital predistortion technique and how it functions. As such, the report gives an overview of adaptive digital predistortion theory and radio over fiber links as well as the practical and theoretical algorithms behind the CAD tools and ASIC design.

Objectives

The objective of the project is not to design a digital circuit, but rather to take a designed digital circuit and use the newest available CAD tools to implement it as an ASIC chip. Further optimization of the ASIC is possible, as some design errors still exist. This issue will be discussed later in Sections 6 and 9 of this report.

Chapter 1 Introduction

1.1 Application Specific Integrated Circuits (ASICs)

In today's world, electronic gadgets become smaller and smaller, yet their abilities continue to improve. To keep up with this trend, the need to compress components and functionality into smaller chips are essential. ASICs and system on chip technologies allow for more functionality while achieving smaller packages at lower costs for high volume productions. ASIC design is one of the most difficult designs in engineering since the design work required to implement the design is enormous. There are two main types of ASICs: standard cells and cell based array ASICs. Standard cells contain pre-designed circuit functions that can be joined by interconnecting cells. The designer designs all the components from top to bottom and specifies all the circuits and interconnections to build the design. Cell based arrays on the other hand, consist of pre-designed cells of transistors with varying sizes depending on drive requirements. Cell based ASICs allow the designer to create circuits faster since the only thing required is to interconnect the transistors properly on production and define the transistor sizes.

Table 1 Quick comparison of cell based arrays and standard cells:

	Non-recoverable cost	Turnaround time	Transistor efficiency	Flexibility
Standard Cells	High	Long	High	High
Cell Based Arrays	Low	Short	Medium	Low

The ASIC design used for this project is a normal standard cell ASIC. The design uses predefined functional and physical blocks defined by the foundry used by CMC. The design is created with CMOS 0.18um technology using the newest CAD tools from both Synopsys and Cadence with the design running at 50 Mhz.

1.2 Digital IC design process

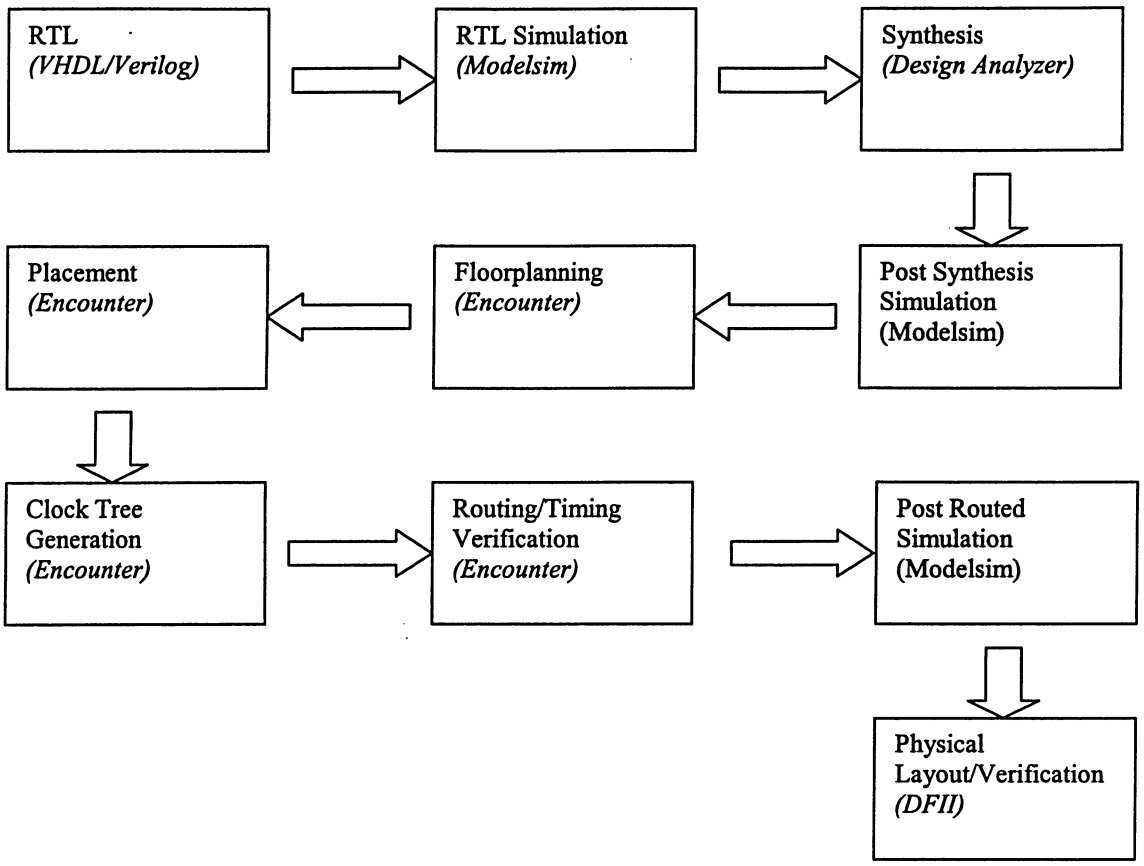


Figure 1 Digital IC design process

The Digital IC design process starting from the RTL level model of the design all the way to physical verification is shown in Figure 1. An overview of both theory and practical design for each stage of the development of digital integrated circuits as shown in Figure 1 will be discussed in the sub sections of 5 and 6.

Chapter 2 Technical functions/materials

2.1 Circuit generation and verification

2.1.1 Synthesis/Simulation tools:

- ✓ VERILOG/VHDL
  - Hardware Description Language used to write circuit code
- ✓ MENTOR GRAPHICS MODELSIM
  - Graphical environment to do VHDL syntax checks
  - Simulates the behavior of the circuit.
- ✓ SYNOPSYS DESIGN ANALYZER
  - Used for synthesis and optimization of the circuit
  - Analyze "timing", "area", "clock skew", etc...

2.2 Layout generation, synthesis and verification

2.2.1 Simulation/Physical design tools:

- ✓ MENTOR GRAPHICS MODELSIM
  - Graphical environment to do VHDL syntax checks
  - Simulates the behavior of the post synthesized and routed circuits.
- ✓ CADENCE FIRST ENCOUNTER
  - Designs the floor plan of the circuit design
  - Implements the proper placement of the core using timing constraints
  - Adds designs to the circuit to meet proper clock timing requirements
  - Routes the nets and verifies timing information
- ✓ CADENCE DFII
  - Final physical check and verification of the chip

2.3 Hardware

2.3.1 Sun Microsystems solaris ultra 10 machine

2.3.2 PC with windows installed along with modelsim SE plus

Chapter 3 Adaptive digital predistortion / ROF Theory

3.1 Radio Over Fiber (ROF)

Radio over fiber has become one of the most useful technologies for providing extended coverage of wireless communications services. It was first proposed by Cooper in 1990 and currently holds a significant market share of the wireless industry. This relatively 'niche' market is expected to grow significantly as new ROF technologies emerge at lower cost. Radio over fiber is becoming the dominant access transmission technology for low-power wireless networks [2].

Indoor fiber optics is more common now with buildings having fiber optics communications and cables installed inside. Figure 2 shows the most basic of ROF link technologies.

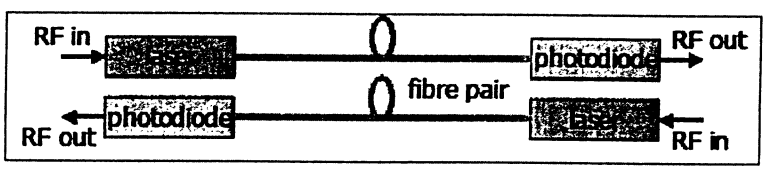


Figure 2 Basic ROF system

ROF uses analog fiber optic links to distribute wireless radio signals from a central location to multiple remote locations where the added desired antennas are placed for stronger signal coverage. The basic radio over fiber system is a bidirectional analog fiber optic link and consists of a laser / photodiode pair at the central site, a laser / photodiode pair at the remote site and optical fibers linking the two sites as shown in Figure 2. Usually for the Figure 2, the LASER light intensity is directly modulated by the RF in signal and the photodiode does direct detection of the light intensity levels. This type of link is called Intensity Modulated Direct Detection (IMDD). IMDD links are the simplest form of modulation and have good wideband performance but are very dependent on individual component specifications. IMDD can be both useful and

useless depending on the characteristics of the components used for the whole ROF link. Characteristics of the components define how linear the intensity modulation will become and at which range the best modulation is possible. Using a suitable LASER with a larger linear output range would mean less signal distortion and no need for adaptive predistortion techniques depending on the modulated signal intensity levels. RF input power on the laser input creates a bias current that modulates the intensity of the output light. Unfortunately, the relationship between the output light intensity to the RF input power is not a linear one. This is where the requirement for adaptive predistortion techniques come in. The modulated signal is transmitted across the optical fiber to the receiving photodiode. The modulated light received at the photodiode creates a modulated photocurrent that produces an RF output power [2].

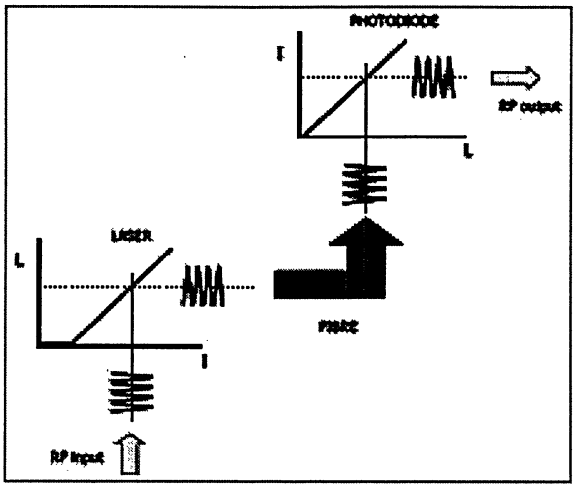


Figure 3 Ideal basic IMDD modulation

Figure 3 illustrates an ideal LASER and photodiode characteristic where both light to current and current to light outputs are linear functions. In reality, both curves are non-linear which is where the need for predistortion and equalizing techniques is required.

Current ROF technologies use directly modulated LASERs and PIN photodiodes due to simpler design and lower cost. The three main ROF link systems are: RF (radio frequency) over singlemode fiber, IF (intermediate frequency) over multimode fiber, and digital over singlemode fiber. Although RF over singlemode fiber is simple, it requires expensive optical components with good analog performance for linearity. IF over multimode fiber solves the issues with expensive analog optical components but requires a more complex antenna unit to implement the up conversion to the intermediate frequency. Digital over singlemode fiber requires complex antenna units and converts the signal to digital before being transmitted and back to analog again after it has been received. Although it can operate with less stringent optical component specifications, it has limited bandwidth due to its electronic processing capability. The digital over singlemode

fiber can traverse longer distances than the previous two technologies which might be advantageous with faster electronic processing in the future [2].

Following this discussion on the basics of radio over fiber, including the current technologies that utilize it and the emerging technologies that support it, it is important to consider the individual components that make up a fiber optic system. Figure 4 shows the main components of a basic radio over fiber system including fiber optic cables, lasers, photodiodes, RF antennas, and the central base station.

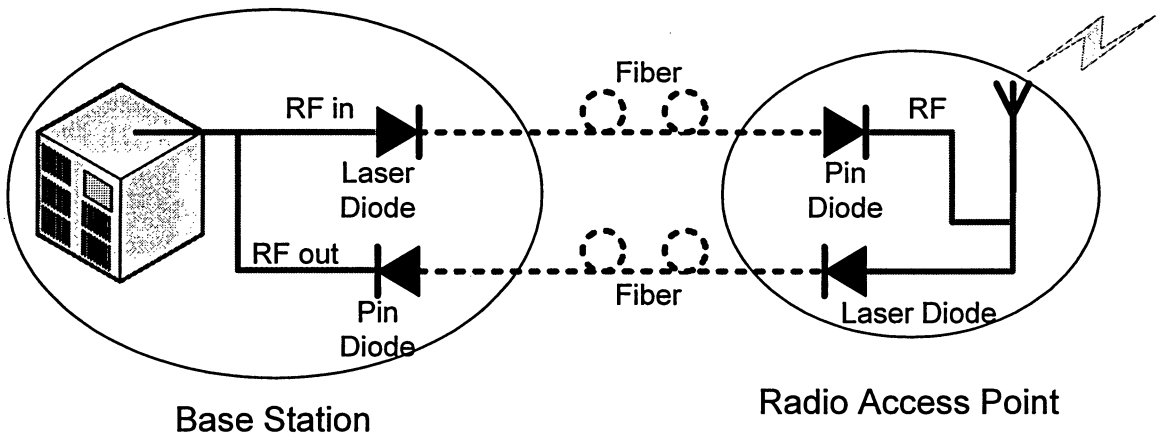


Figure 4 ROF system

3.1.1 Fiber optic cables

Fiber optic cable is the new trend in transmission medium due to its numerous advantages over copper with the only drawback being the cost. Although copper as opposed to ROF is still preferred in some places for radio over copper (ROC) links for cost reasons, future bandwidth demands for will certainly surpass the limitations of copper. This defeats the purpose of designing with copper due to its limited upgradeability. When high bandwidth, distance, and flexibility are required by the application being created, fiber optic cables are definitely the preferred transmission media.

Some of the advantages of fiber optic cables over copper include [6]:

- Speed: Fiber optic networks operate at high speeds of Gbps unlike copper
- Bandwidth: Large carrying capacity, virtually almost unlimited bandwidth
- Distance: Less attenuation meaning longer distances traveled
- Resistance: Immune to EMI since light is used
- Maintenance: Fiber optic cables costs much less to maintain compared to copper

Major disadvantages of fiber optic cables over copper are [6]:

• Cost: Fiber optics initial deployment is very expensive

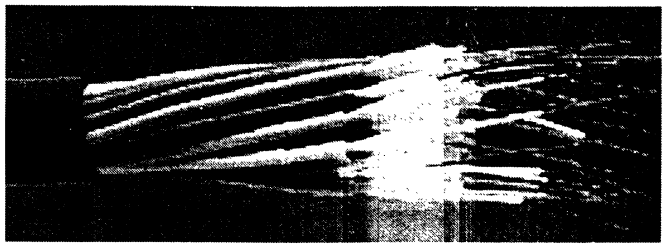


Figure 5 Basic fiber optic cable

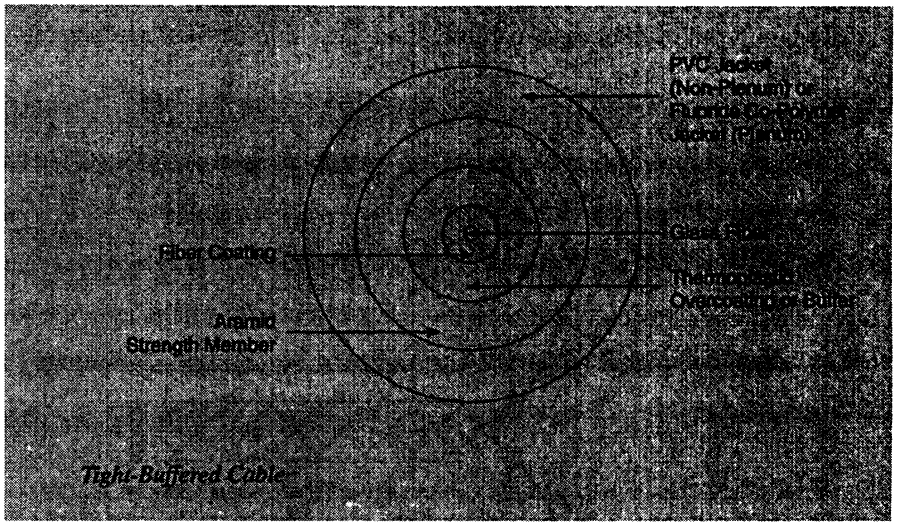


Figure 6 Internal structure of fiber optic cable

Figures 5 and 6 depict a basic fiber optic cable and its internal materials. All fiber optic cables use the theory of total internal reflection to transmit signals. As shown in Figure 7, for total internal reflection to occur, the core/clad interface obeys:

$$\sin(\beta Min) = n2 / n1 \quad (1)$$

where αMax = largest angle the fiber can accept.

n2 = refractive index of fiber cladding

n1 = refractive index of fiber core

The Numerical Aperture (NA), of the light guide, describes αMax:

$$NA = n \sin(\alpha Max) = (n1^2 - n2^2)^{1/2}$$

where $\alpha \leq \alpha Max$ or $\beta \geq \beta Min$ is the limitation for total internal reflection to occur.

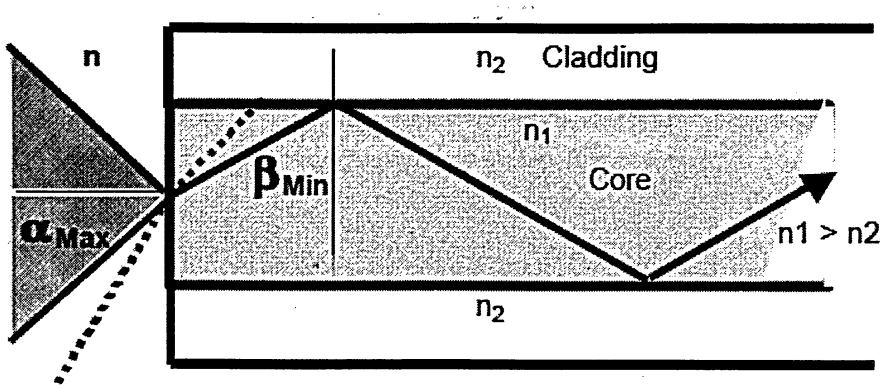


Figure 7 Total internal reflection inside a fiber optic cable

There are two main types of fiber optic cables namely, multimode fiber and singlemode fiber [7].

3.1.1.1 Multimode fiber

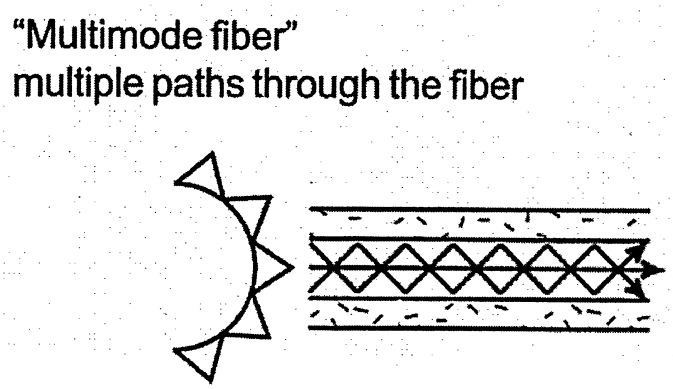


Figure 8 Multimode fiber cable light wave path

Multimode fiber allows different modes (angles) of light wave to enter the fiber optic because its diameter is larger than that of the singlemode fiber as shown in Figure 8. Multimode fiber is made of glass fibers, with the most common core diameter size being 62.5 micron. This allows high bandwidth at high speeds over medium distances. The main disadvantage of multimode is that in cable runs greater than 3000 feet, multiple paths or modes of light can cause signal distortion at the receiving end. This could result in an unclear and incomplete data transmission since different modes of light arrive at different time intervals [6].

3.1.1.1.1 Step index multimode fiber

The refractive index of the different materials being used in the fiber core of the step index multimode fiber changes in steps. This creates an immediate difference in the index of the material from which the light travels on and its outer coating. As such, the light tends to bounce back in angles as depicted in Figure 9. Depending on the angle of light bounce, different modes of light will arrive at different times at the receiver. Distortion is a problem at long distances or higher data rates.

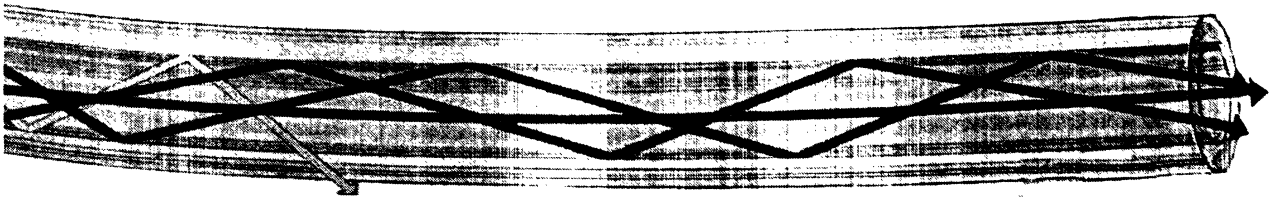


Figure 9 Step index multimode fiber cable light wave path

3.1.1.1.2 Graded index multimode fiber

Graded index fiber is also a multimode fiber. The difference between step index multimode fiber and graded index multimode fiber is that the refractive index diminishes gradually from the core center to the cladding. The angles or modes of light are thus smoother and do not bounce at sharper angles resulting in the different modes of light arriving at the receiver more or less at the same time [6].

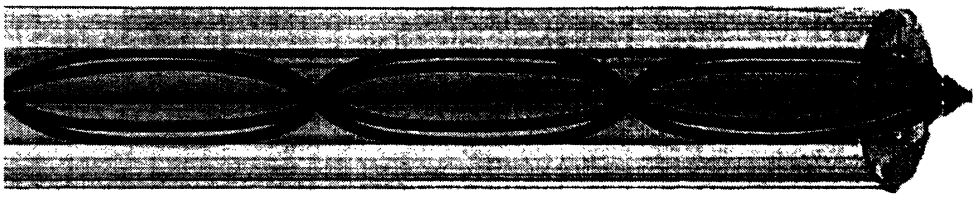


Figure 10 Graded index multimode fiber cable light wave path

3.1.1.2 Singlemode fiber

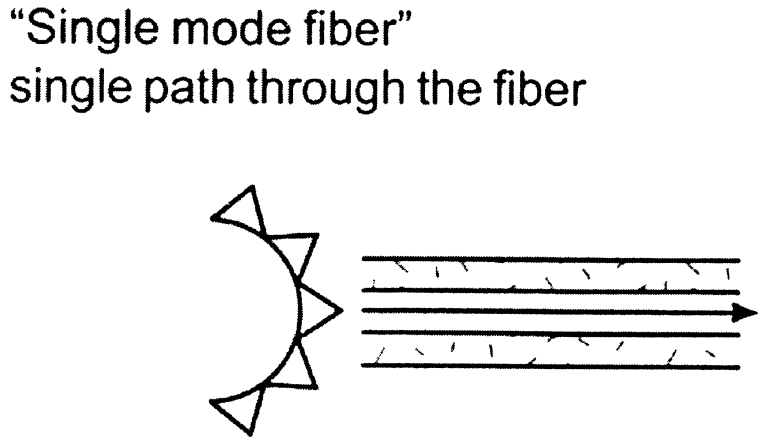


Figure 11 Singlemode fiber cable light wave path

Singlemode fiber allows only one mode (angle) of light wave to enter the fiber optic as its diameter is smaller than that of the multimode fiber. Singlemode fiber is made of a single strand of glass fiber, with a diameter of 8.3 to 10 micron range for the light component. This allows for high bandwidth at high speeds over long distances. The main disadvantage of singlemode is that it is more expensive than multimode due to its expensive light sources and detectors in addition to a more complex manufacturing process. Singlemode fiber surpasses multimode fiber in every aspect of transmission. i.e. speed, distance, distortion and attenuation with the only main drawback being cost.

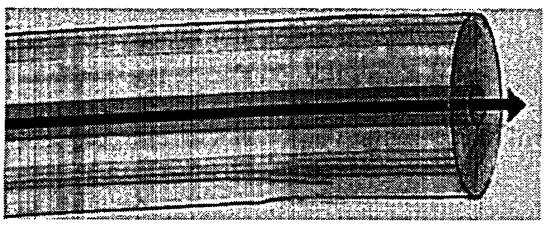


Figure 12 Singlemode fiber cable light mode

3.1.2 Optical light sources

Light sources used in any design depend on the characteristics of the fiber optic cable used as well as the data rate of the application and the distances to be linked. There are two main types of optical light sources namely: Light Emitting Diodes (LEDs) and Light Amplification by the Stimulated Emission of Radiation (LASERs). LEDs can only be used on multimode fibers since the amount of light coupled onto a singlemode fiber from a LED is so minimal that it is essentially useless. The more costly LASERs on the other hand, are generally used in singlemode fibers as there is no point in spending more for a LASER in a multimode fiber if an LED will do the job. LEDs are simple and generate incoherent, lower power light. In contrast, LASERs are more complex and generate coherent, higher power light. When compared with LEDs, the advantages of LASERs include the capability of being modulated at very high speeds, greater optical power and higher efficiency in coupling to the fiber optic cable. The advantages of LED's over LASERs are higher reliability, better linearity, and lower cost.

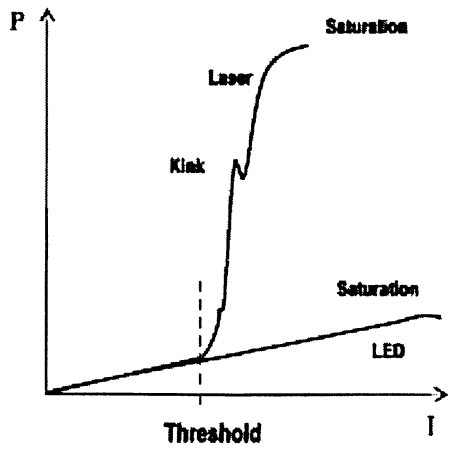


Figure 13 LASER vs LED power to input current

Figure 13 illustrates the optical power output, P, from each of these devices as a function of the electrical current input, I, from the modulation circuitry. As indicated in Figure 13, LEDs have relatively linear P-I characteristics while LASERs have non-linear P-I characteristics. Adaptive predistortion techniques address the non-linearity issues of the P-I characteristics for LASERs as analog signals cannot be modulated properly with LASERs [6].

A major contributor to the ROF link non-linear characteristics is the non-linearity of the LASER diode itself as shown in Figure 13. Laser diode characteristics consist of dynamic and static nonlinearity which both contribute to the nonlinearity issue with static nonlinearity having the most interest.

Dynamic nonlinearity is defined by the laser rate equations as shown below [1]:

Rate of charge of photons: $$\frac{d\Phi}{dt} = Cn\Phi + R_{sp} - \frac{\Phi}{\tau_{ph}}$$

Rate of change of electrons: $$\frac{dn}{dt} = \frac{J}{qd} - \frac{n}{\tau_{sp}} - Cn\Phi$$

where n = number of electrons,

Φ = number of photons,

C = Einstein's Coefficient,

τ_{ph} = photon lifetime,

R_{sp} = rate of spontaneous emission,

τ_{sp} = spontaneous recombination lifetime,

J = injection current density,

q = electron charge

Static nonlinearity is defined by the third order polynomial equation given below [1]:

$$P_{optical}(t) = P_o[1 + A_1 ms(t) + A_2 m^2 s^2(t) + A_3 m^3 s^3(t)]$$

where $s(t)$ = input electrical signal,

P_o = average optical power,

m = optical modulation index,

A_1, A_2, A_3 = device dependent nonlinear coefficients.

3.1.3 Receiver light detectors

The receiver component senses or detects light coupled out of the fiber optic cable and converts light into an electrical signal. It determines if the signal is digital data or analog data then interprets it accordingly. The receivers are comprised of two types of photodiodes, namely Positive Intrinsic Negative (PIN) photodiodes and Avalanche Photodiodes (APD). Receiver circuits are built using these two types of photodiodes and, depending on the application, the circuit can be simple or very complicated. The complete receiver circuit usually needs to have high sensitivity, high bandwidth, and low noise.

For most applications, the PIN photodiode is sufficient as it can be operated from standard voltage power supplies resulting in cheaper costs. APD devices on the other hand, have much greater sensitivity and twice the bandwidth. Unfortunately, APDs requires a stable power supply and cannot be implemented in a standard PCB, resulting in higher costs. A good rule of thumb is PIN photodiodes for low end applications and APD devices for high end applications. Figures 14 and 15 show the basic structure of photodiodes.

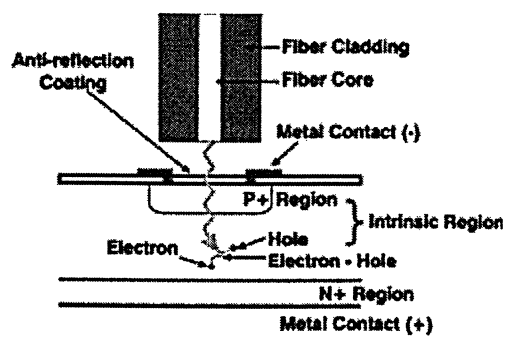


Figure 14 PIN photodiode

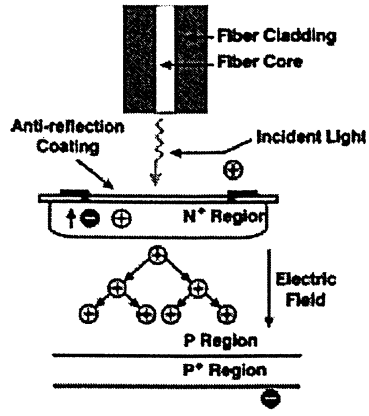


Figure 15 APD photodiode

Photodiodes generate internal noise as well when converting light intensity levels to current output signals. The major noise sources are thermal noise, and shot noise. Thermal noise is noise due to the surrounding temperature around the diode defined by the equation below [1] [10]:

Thermal noise equation: $$i_t^2 = \frac{4k_B TB}{R_L}$$

where k_B = Boltzman's constant,

T = temperature,

R_L = load resistance,

B = signal bandwidth

Shot noise is the random generation of electrons in the diode and is defined by the equation below [1] [10]:

Shot noise equation: $$i_d^2 = 2qI_p B$$

where q = charge of an electron

I_p = output dc current of the diode

3.1.4 Radio frequency antennas

Wireless transmitters, i.e. antennas, are an important aspect of the ROF system since their characteristics define the extent of radio coverage and the frequencies supported. The most basic antenna is the "quarter wave vertical", with a physical antenna length equal to a quarter wavelength (depending on what frequency it receives) and is a vertical radiator. This most basic antenna radiates in all directions similar to the light from a light bulb. Typically, an antenna with cross polarization and low antenna impedance is desired to ensure maximum signal integrity over a longer distance. There are various types of antennas including directional antennas and single polarized antennas. An antenna is chosen with the gain determined by the intended area of coverage. The gain at a given wavelength is achieved by appropriately choosing the size of the antenna [8].

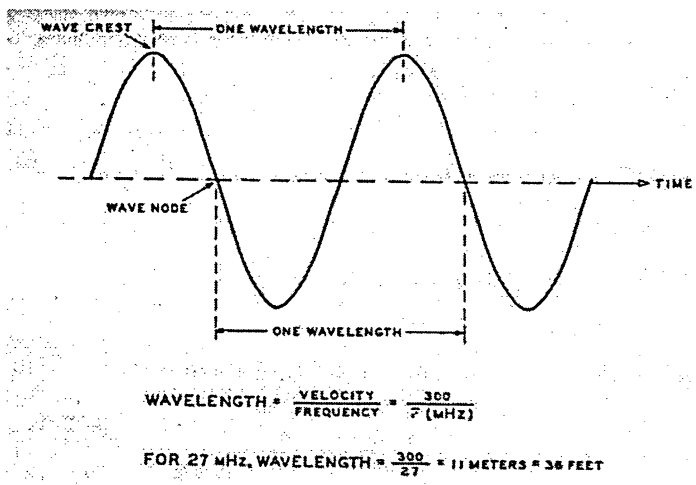


Figure 16 Wavelength to frequency relationship

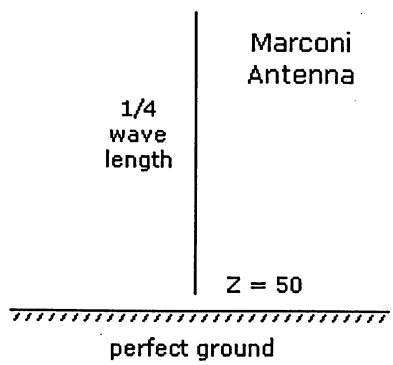


Figure 17 Simple Marconi antenna w/ quarter wavelength and impedance of 50

3.1.5 Central base station

The central base station is the transmitter or processing element within the whole ROF system. The base station handles most of the cellular area coverage, the amount of users allowed per base station and the transmission between base stations when users cross different base station territories. It keeps track of where the user is, how long he uses the phone, as well as all of the radio services offered to the users. Multiple base stations in dense areas are organized in cells which are hexagonal in shape. Each base station uses only a couple of frequency ranges to prevent interfering frequencies between adjacent cells. As shown in Figure 18, although all cell 1 base stations use the same frequencies no interference between frequencies occurs because they are so far apart [11].

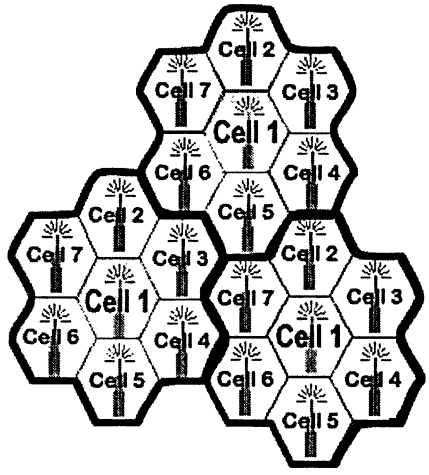


Figure 18 Hexagonal groups of cell base stations

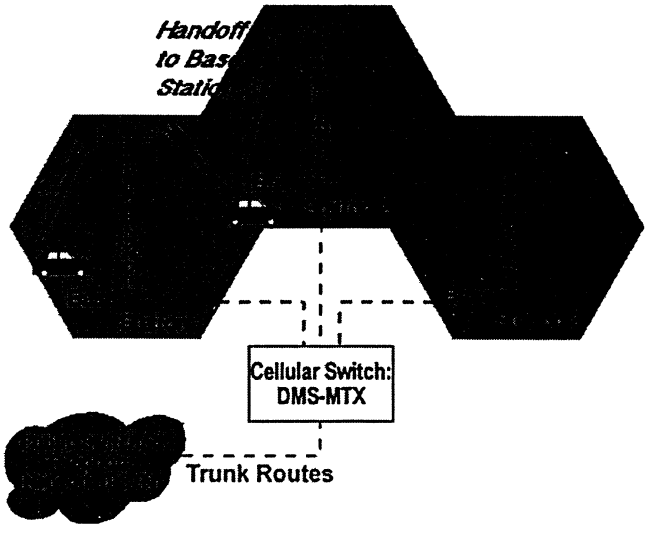


Figure 19 Transfer of users from different cell base stations

Users are automatically transferred from different base stations as illustrated in Figure 19. Although base stations work fairly well outdoors, foreign objects can dramatically attenuate RF signals. Thus, concrete used in buildings easily attenuates RF signals down to a bare minimum in cases where it is difficult to achieve a good RF signal indoors. By installing distributed antenna systems (DAS) throughout a building, the RF signal inside the building is regenerated from the base station and as such achieves good RF signals indoors. This is particularly useful for crowded shopping malls to justify the extra costs for installation of ROF systems.

3.2 Adaptive digital predistortion techniques

The implemented ASIC handles digital predistortion for ROF systems that use LASERs and PIN photodiodes. Basically, the signal is predistorted to the inverse of the ROF link characteristics so that when the signal traverses the ROF link it gets equalized by the channel characteristics and produces a good signal at the receiving end. The predistortion is carried out only for the transmission end and not the returning path with the assumption that signal feedback at the receiving end exists for the digital predistortion circuit. Figure 20 shows a basic diagram of the predistortion system.

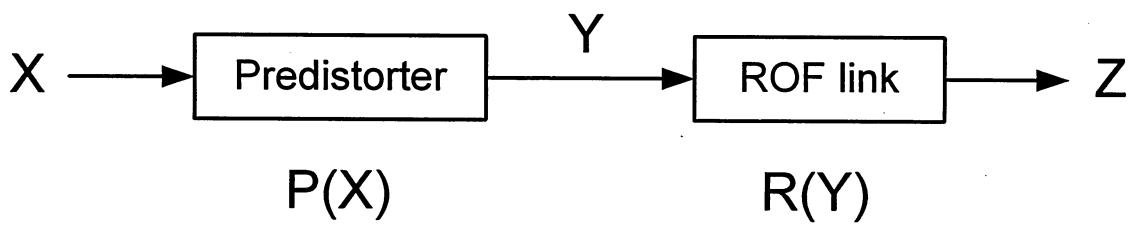


Figure 20 Basic predistortion system technique [1]

Assuming the transfer function of the predistorter is P(X) and the transfer function of the ROF link is R(Y), then the function becomes

$$X = R(P(X)) = Z$$

Since X = Z, the transfer function P is the inverse of R or $R = P^{-1}$. The goal of the predistorter is to ensure that the function P is the inverse of R and that the link equalizes the signal as it passes through. Two methods of implementation include analog filters and circuits as well as digital electronics. Due to the progression and ease of use of digital electronics, digital predistortion presents itself as an easier and cheaper way of creating digital predistortion techniques before the signal is sent through the ROF link.

Predistortion techniques require feedback from the signals at the receiving end of the circuit to ensure that it is indeed predistorting the signal correctly. Any transfer function has a magnitude portion and a phase portion. For both transfer functions to be the inverse of the other, the magnitude must be the inverse

$$1 = |R|(|P|)$$

thus, $|R| = 1/|P|$. The phase or the angle must also be the inverse of one another, i.e.

$$\theta R = -\theta P$$

thus, the angle or phase distortion created by the predistorter is the negative of the ROF link phase distortion. Once both angle and magnitude values are exactly the inverse of each other such that the input matches the output signal a good predistortion system is achieved. In reality however, a feedback signal is required to ensure that the predistorter does indeed track the inverse transfer function of the dynamically changing ROF link transfer function [1].

3.3 Coordinate Rotation Digital Computer Algorithm (CORDIC)

The CORDIC algorithm was created by Volder in 1959 for computation of trigonometric functions, multiplication, division and datatype conversion of numbers. Two basic CORDIC modes, the rotation mode and the vectoring mode, lead to the computation of different functions .

Digital signal processing algorithms exhibit the need for efficient implementation of complex arithmetic operations as current devices become more and more complex. The computation of trigonometric functions, polar to rectangular transformations, and rectangular to polar is almost always used in modern DSP algorithms. These tasks can all be efficiently implemented using the CORDIC algorithms. CORDIC enables digital electronics to calculate all the desired functions in a rather simple and painless way [12].

The algorithm can be realized for both modes as an iterative sequence of additions, subtractions, and shift operations. The shift operations are a fixed rotation angle but with variable rotation direction. This algorithm is very well suited for VLSI implementation and DSP but the iterations are not always perfect.

Figures 21 and 22 show examples of CORDIC algorithm rotation trajectory for the circular and linear coordinate systems respectively.

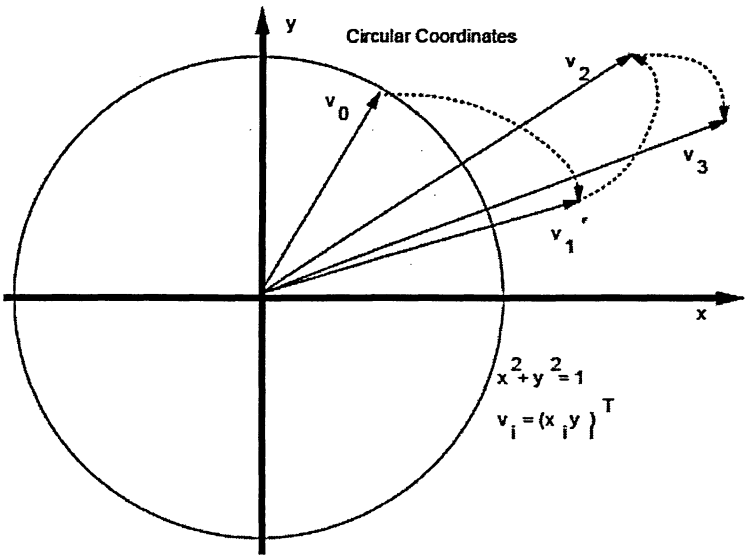


Figure 21 Circular coordinate system

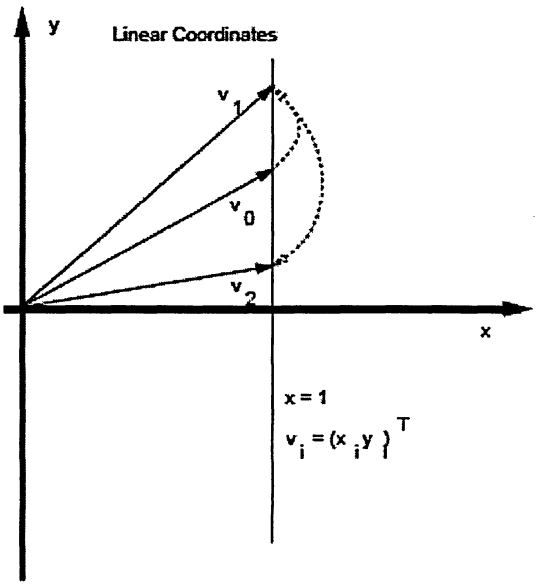


Figure 22 Linear coordinate system

The standard expression for a CORDIC algorithm [1]:

$$X_{new} = X\cos(\phi) - Y\sin(\phi)$$
$$Y_{new} = Y\cos(\phi) + X\sin(\phi)$$

The final equation that can be used for rectangular coordinate to polar coordinate conversion is as follows [1]:

$$X = X \qquad Y = Y \qquad Z = 0$$
$$[X_{new}, Y_{new}, Z_{new}] = [P\sqrt{X^2 + Y^2}, 0, \arctan(\frac{Y}{X})]$$

For the purposes of this project, the CORDIC system is only used to convert from rectangular to polar form and vice versa. The assumption is that the incoming signals will be received in rectangular format, processed in polar format and then reconverted to the appropriate rectangular format before being sent out to the ROF link.

Chapter 4 Adaptive digital predistortion / ROF Techniques

4.1 Adaptive digital predistorter overview

The adaptive digital predistortion system as shown on Figure 23 is discussed briefly in this section (including an overview of the design and building blocks).

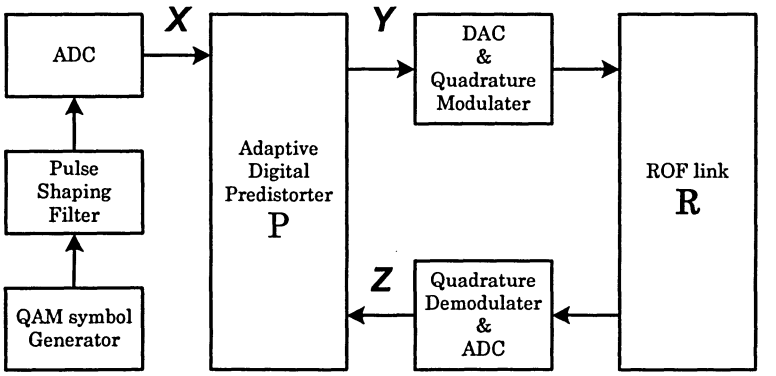


Figure 23 Adaptive digital predistortion system overview [1]

4.1.1 QAM symbol generator

The QAM symbol generator generates the signals to be transmitted onto the ROF link as data by the users. It is comprised of two signals in rectangular coordinate form that are combined and transmitted in QAM format.

4.1.2 Pulse shaping filter

The pulse shaping filter shapes the pulse into a more rectangular or square wave formed signal for easier processing.

4.1.3 ADC

The ADC converts the analog signal of the QAM into a digital signal that can be interpreted and processed by the digital adaptive processing element. This is required by the adaptive predistortion technique that processes digital data only.

4.1.4 DAC and QAM modulator

The digital to analog converter or DAC and the QAM modulator take the predistorted rectangular form output signal from the adaptive predistortion element and convert it into an analog signal. The two analog signals are then modulated into one signal using QAM to be transmitted through the ROF link.

4.1.5 ADC and QAM demodulator

The analog to digital converter or ADC and the QAM demodulator take the final received waveform output signal from the ROF link, demodulate it and convert it into a digital signal. The two rectangular form digital signals are then used as feedback to the internal workings of the adaptive predistortion module to constantly update its memory correction elements and to adapt to the dynamically changing ROF link characteristics.

4.1.6 ROF Link

The ROF link is a simulated model of the actual ROF link being used in the real system. It takes in the signal and manipulates its phase and magnitude so that it appears as if it has traversed an actual ROF link. Figure 24 shows the transfer functions of the ROF link and how the magnitude and the phase of the signal are distorted as the input power of the signal varies. As input power increases, the output power saturates onto a high point and the phase distortion approaches 80 degrees [1].

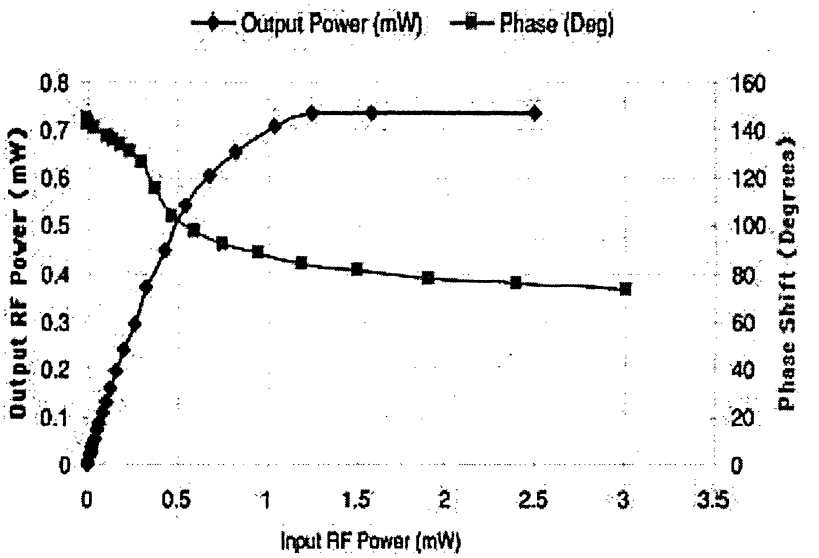


Figure 24 ROF link magnitude and phase characteristics [1]

4.1.7 Adaptive digital predistorter

The adaptive digital predistorter is the heart of the whole design. It is the digital processing element and storage that predistorts the incoming input signal and sends out the distorted signal onto the ROF link. It also has the built in feedback signal that allows it to dynamically update and correct the memory contents of the FIFO block to ensure that the correction being done to the output signal still corresponds to that of the current ROF link characteristics. Figure 25 shows the building blocks being used by the ADP for this project [1].

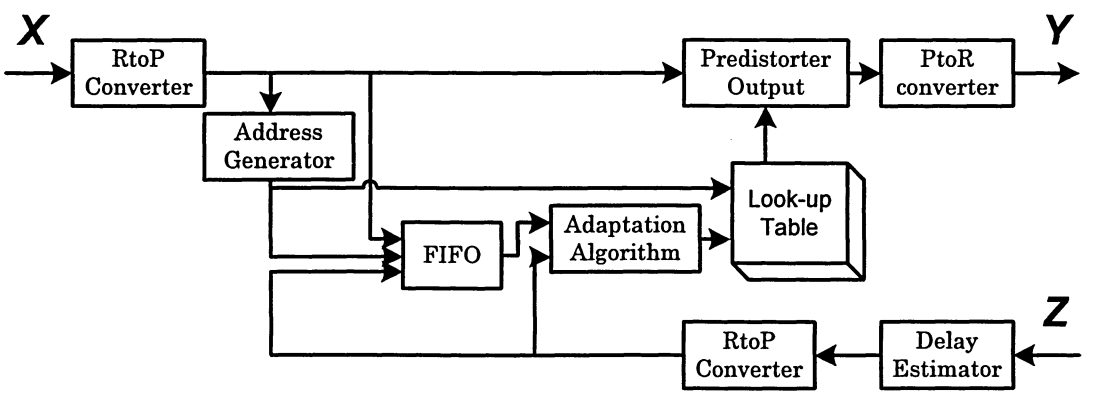


Figure 25 Proposed adaptive digital predistorter [1]

4.1.7.1 RtoP converter

The RtoP converter is merely a block that converts rectangular form to polar form since the data being kept inside the look up table is in polar form. This is implemented using the CORDIC algorithm for easier realization in hardware. X and Z signals are in rectangular form by default and the output of the RtoP and PtoR converters convert the X and Z signals into their appropriate polar forms and vice versa.

4.1.7.2 PtoR converter

The PtoR converter is merely a block that converts polar form to rectangular form since the data being sent is actually a rectangular form signal. This is implemented using the CORDIC algorithm for easier realization in hardware. X and Z signals are in rectangular form by default and the output of the RtoP and PtoR converters convert the X and Z signals into their appropriate polar forms and vice versa.

4.1.7.3 Delay estimator

The delay estimator is a block that ensures that the data being received is delayed appropriately so that during the training of the adaptation block, the design matches the data sent to the data received and obtains their differences. This is required so that the adaptation block knows if the output signal is indeed being properly predistorted. This block is needed in order to ensure a match between the correct input data and ROF link received data.

4.1.7.4 LUT

The corrective values are stored in the look up table and, depending on the input power (i.e. address values), the appropriate data is taken out from the LUT with the proper address. The data coming from the LUT is then used accordingly to correct the incoming data.

Table 2 Example of a LUT [1]:

Input Power	Output Amplitude	Output Phase
R1	Ro1	θRo1
R2	Ro2	θRo2
....	....	....
Rn	Ron	θRon

The input power determines the LUT address for the output amplitude and output phase. The stored values inside the LUT are corrected every so often by the adaptation algorithm to ensure that the values inside the LUT for distorting the signal are up to date.

4.1.7.5 Predistorter output

The predistorter output module takes in the current input signal power and the corrective values from the LUT. The block then applies the corrective values and the input signal power to get the final distorted output signal. This block does the addition or subtraction of the phase margins using Figure 25 to ensure that the final phase is what was intended. The same operations are done with the magnitude portion of the signal to ensure that both the phase and magnitude portions of the received signal match those of the input signals.

4.1.7.6 Address generator

The address generator divides all the possible amplitude or input power values into ranges for simplification. Since there could be an infinite number of points in any signal, a division into range allows the address generator to store realistic values for the input power and its corresponding corrections in the look up table. For this design, the range was divided into 16 regions. The address generator gives the correct address depending on which input power range so that the LUT values may be used.

4.1.7.7 First-in-First-Out (FIFO)

The FIFO block is used to store the data for later comparison with the received delay feedback signals from the delay estimator. When the delayed signal is required to be read then a read signal is sent to the FIFO by the delay estimator so that the previous input signal and the received delayed signal may be compared. Updates can be made to the LUT after the comparison to ensure that the input signal matches the received output signal. If the initial input data is not stored and can not be compared to the received delayed signal, there would be no way to update the LUT data to ensure that the distorted output signals still match those of the ROF link characteristics.

4.1.7.8 Adaptation algorithm

The adaptation algorithm generates the most current LUT coefficients by comparing the input signals and the received output signals. The adaptation algorithm ensures that the LUT coefficients are correct so that the distorter output block always applies the correct amount of distortion to the output signal. If the LUT coefficients do not ensure that the input signal matches the received output signal, they are updated by the adaptation algorithm.

Chapter 5 Digital ASIC IC Design Process Theory

5.1 Theory

The two main types of ASICs are standard cells and structured ASICs (cell based array ASICs). Standard cells contain pre-designed circuit functions at the VLSI level of complexity and can be joined by interconnecting cells. The designer defines all the components and interconnections to build the design. Cell based arrays on the other hand, consist of pre-designed transistor cells of varying size depending on drive requirements. This allows circuits to be created faster as the designer must only interconnect the transistors properly depending on the drive capabilities. However, cell based arrays use more gates and are less flexible in terms of special blocks and circuits [13].

Table 3 Quick comparison of cell based arrays and standard cells:

	Non-recoverable cost	Turnaround time	Transistor efficiency	Flexibility
Standard Cells	High	Long	High	High
Cell Based Arrays	Low	Short	Medium	Low

Before designing an ASIC, an engineer must consider the tradeoffs involved and the exact requirements of the ASIC. Major tradeoffs include: area, turnaround time, cost, performance, flexibility and special blocks.

Area limitations dictate how big the ASIC die can become, thus, the amount of circuit blocks, special blocks and interconnects allowed in the ASIC are limited. Turnaround time determines how long it takes to develop the ASIC from start to finish, an indicator of how quickly the product can be marketed. ASIC cost increases as die size increases. Therefore, the smaller the ASIC die the cheaper it is. Performance of the ASIC

also depends on the ASIC design including the transistors used and the drive strength. Less fan out and logic levels allow for faster ASIC speeds. Flexibility refers to how easy it is for the designer to add features or make changes to the ASIC after it has been designed. In contrast to FPGAs, ASICs have very little if any flexibility as once fabricated, a new mask must be generated. Special blocks define the numbers and types of specialized blocks required by the ASIC such as RAM, PLLs, ROM, and microcontrollers.

Table 4 Quick summary of ASIC choices depending on ASIC requirements:

ASIC Requirements	Standard Cells	Cell Based Array
Flexibility		X
Number of Logic Blocks		X
Performance	X	
Turnaround Time		X
Die Size	X	
Special Blocks	X	
Mass Production	X	

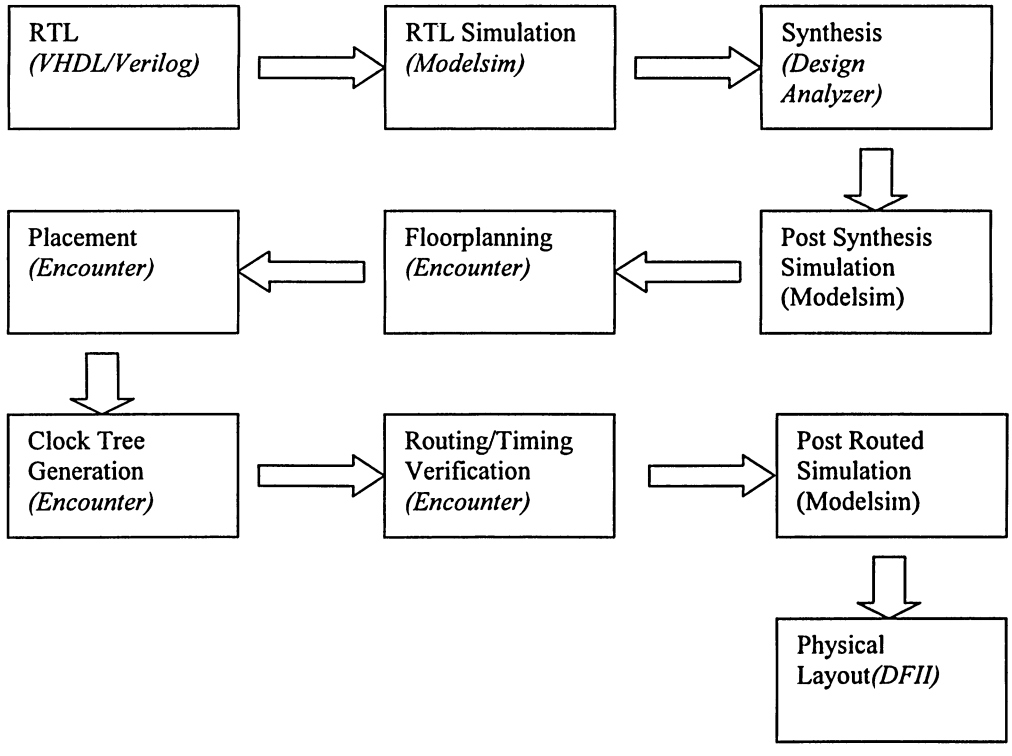


Figure 26 Digital ASIC design flow process

The basic digital ASIC design flow followed in this project is presented in Figure 26. The project commenced at the synthesis of the whole design and followed through until the physical layout of the ASIC was completed.

5.1.1 RTL

The RTL design of a digital system is very critical to the rest of the design. This is the stage where constraints are made and the structure of the project is analyzed and determined. RTL, the register transfer level, defines the basic building blocks of any digital system such as flip-flops and combinatorial elements such as gates, muxes, adders, etc. The design is created using hardware description languages (HDLs) such as VHDL and Verilog to implement the design. HDLs describe the design in terms of the basic digital building blocks using text for ease of use and design. Building a design using all the gates in schematic form would take not only an incredible amount of designers but also greatly lengthen the design time. HDLs allow almost any type of hardware building block to be modelled and characterized such as gate delays and transport delays.

The reason for using VHDL as a CAD tool in the design of digital circuits is due to the high level of design complexity. Although similar to C and normal programming languages, VHDL statements can be executed in parallel with other VHDL statements in the same way any type of digital block can run in parallel with other digital blocks. In C or other sequential languages, statements cannot be executed in parallel. This is one of the reasons for the debate whether HDL programming is a hardware type of design or a software type of design. HDLs are also capable of determining when a certain statement is executed after a certain time delay. Generally, this does not apply in sequential programming. Figure 27 shows an example of an RTL diagram [3].

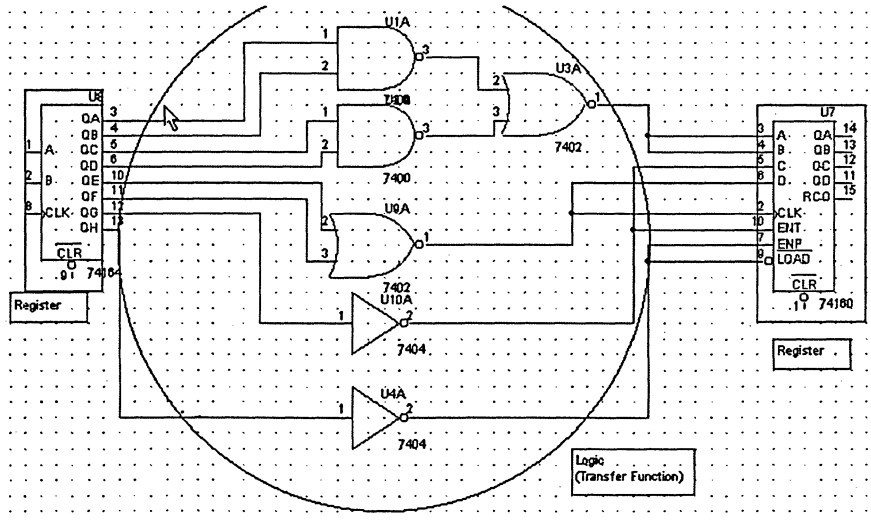


Figure 27 Register transfer level diagram

Example of an inverter defined in HDL:

```
Module inv(in, out);
  Input in;
  Output out;
    assign in = not out;
endmodule;
```

5.1.2 RTL simulation

Another great aspect of HDL coding is not only that a circuit can be described in register-transfer level but also that it can be simulated. This code is called a test bench and involves applying test signals or input stimulus to the circuit and checking the outputs for the expected response. The basic analogy is that the circuit described in VHDL represents an IC chip and the test bench represents a test fixture. The only difference between HDL and the actual circuit is that HDL is much more flexible.

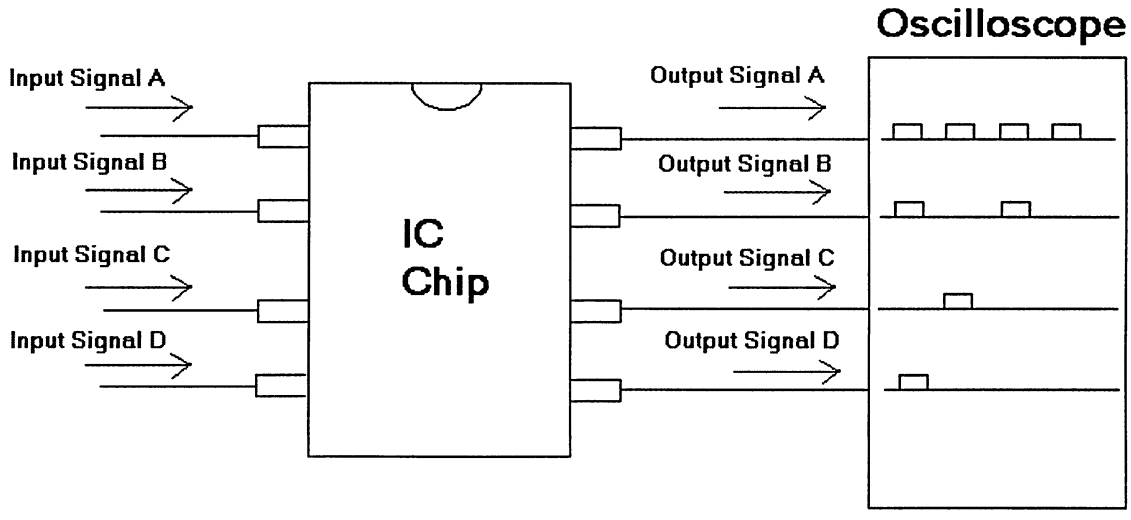


Figure 28 Practical real world test

Figure 28 shows a practical test where the IC chip gets input stimuli and the IC's output is then taken and analyzed. If the outputs match the supposed output determined by the input stimuli then the IC is working. The unfortunate part of this test is that if the IC doesn't work then it will be very difficult to tell which part of the design is not working properly. Fortunately for normal RTL design using HDL coding, it is fairly simple to analyze the internal workings of the circuit. The test bench can be made fairly extensive such that the internal signals within the chip can be monitored and analyzed. This allows for ease of testing and faster debugging of the IC being designed.

Different types of simulation include low level simulations such as transistors, RTL level simulations which are the gate level simulations, and even system level simulations for bigger more complex systems. Simulation involves constructing computer models of the device under test and then using the model to analyze the circuit's behavior. All simulations in this project were performed on register-transfer-level (RTL), post-synthesis and post-routed simulations. The main goal of simulation at this level is to notice state transitions, analyze circuit functionality and determine if any timing errors exist.

In digital circuit simulation using binary logic, signals are denoted as either a '0' which stands for "low" or '1' which stands for "high". Defined signals can be of input nature (feeding into the gate) and of output nature (coming out of the gate). Before an entire circuit can be simulated, a simulator must be able to fully model the behavior of a single gate. The simplest way to model a gate is by using a truth table. Table 5 illustrates a truth table for a simple 2-input AND gate.

Input_A	Input_B	Output
'0'	'0'	'0'
'0'	'1'	'0'
'0'	'X'	'0'
'1'	'0'	'0'
'1'	'1'	'1'
'1'	'X'	'X'
'X'	'0'	'0'
'X'	'1'	'X'
'X'	'X'	'X'

Table 5 Truth table for a 2-input AND gate.

Note: 'X' signal represents an unknown signal value or uninitialized signal.

The real world is not perfect and therefore signals propagating through gates will incur delays. Logic gates are modeled by a fixed delay at the gate's output, which means that the effect of an input signal will be seen at the output after this fixed delay. In order to have a switching effect on the output, an input signal must have a certain pulse width, thus an inertial delay is introduced. All these unwanted effects take their toll on the computers processing power which makes simulations slow.

Event-driven simulation means that the computation of signal propagation is only carried out when a signal is actually changing. This is the most efficient simulation method to increase simulation speeds. In event-driven simulation, the event queue or event list is the central data structure. The event queue stores the events that are occurring in the circuit that is being simulated. The simulation algorithm of the given simulator adds and retrieves events from the event queue. When the simulation is taking place, events in the event queue

are fetched, signal changes associated with those events are processed and new events that result from the current events are created and stored at proper time in the event queue [3].

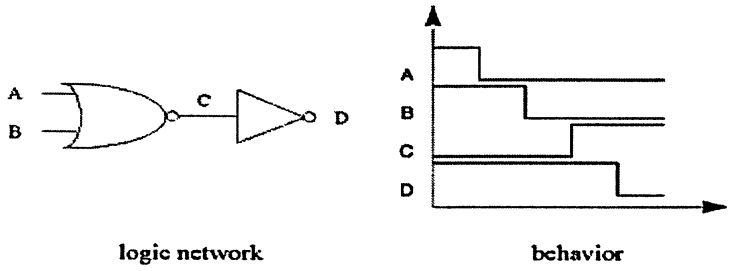


Figure 29 Event-driven simulation example

Figure 29 illustrates event-driven simulation, with a NOR gate connected in series with an inverter. From the behavior of the circuit, it is observed that signal A changes at time t1 and signal B changes at time t2. Signal C changes at time t3 when both A and B are at logic level 0. In event-driven simulation, there is no need to compute signal C if neither signal A nor B changes. This allows for faster simulations since only the signals that have changed need to be processed by the simulator.

5.1.3 RTL synthesis

As discussed in the previous sections of this report, HDL coding represents digital building blocks in terms of text files written in code. The actual hardware hasn't been decided and is not known to the designer at this point. This step in the digital IC design, referred to as logic synthesis, specifies exactly how the hardware should look and become. During the logic synthesis process, the synthesis tool uses a specified technology library coming from a foundry (in this case CMC's 0.18 um technology) and determines from the available logic blocks in that library, the best way to create the digital circuit from the code. Logic synthesis is normally broken down into two main sections, sequential logic synthesis and combinatorial logic synthesis [3].

Sequential logic synthesis is the synthesis of code that becomes flip flops and latches in any digital design. These are called "sequential" due to the fact that flip-flops require a clock signal to trigger the data to their outputs properly. Figure 30 illustrates how a sequential code is synthesized into a D-flip-flop. Sequential circuits are created using combinatorial gates with feedback loops which create memory elements.

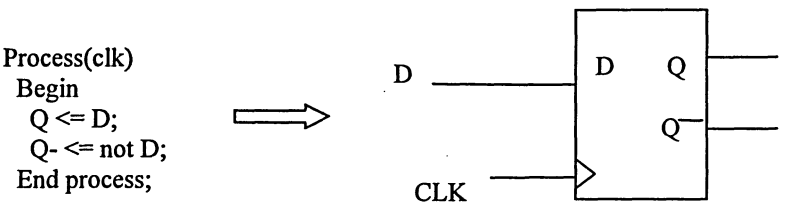


Figure 30 Sequential code synthesis

Combinatorial logic synthesis is the synthesis of code that becomes AND gates, NAND gates, muxes, or any type of digital block that is not level triggered or edge triggered. These combinatorial blocks change their outputs with regard to their inputs after the gate delay time of the signal passing through the gates. Figure 31 shows a good example of how a combinatorial code gets synthesized. Combinatorial circuits are created using combinatorial gates without feedback loops to ensure no memory elements are created.

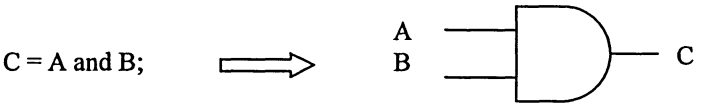


Figure 31 Combinatorial code synthesis

As shown on Figure 31, two inputs A and B go to a single output C. It can be seen from A and B, that there are 4 possible values of C if we allow the inputs A and B to be either a logic 1 or logic 0 which is standard. Unfortunately, by increasing the number of inputs variables x, the number of possible output combinations (also called Boolean space) also increases in the order of 2^x. This dictates the complexity of the synthesis algorithm required to synthesize the design. The larger the design, the harder and more complex the synthesis algorithm has to be. To fix this problem, reduced ordered binary-decision diagram (ROBDD) is used for logic synthesis.

Reduced ordered binary decision diagram allows for more compact representations of Boolean functions to help solve the exponential problem of Boolean spaces. To understand how ROBDD works, one must first be introduced to OBDD, which stand for ordered binary-decision diagram. Figure 32 shows an example of representing a Boolean function using an OBDD tree. Table 6 shows the Boolean function that generated the OBDD in Figure 32 [3].

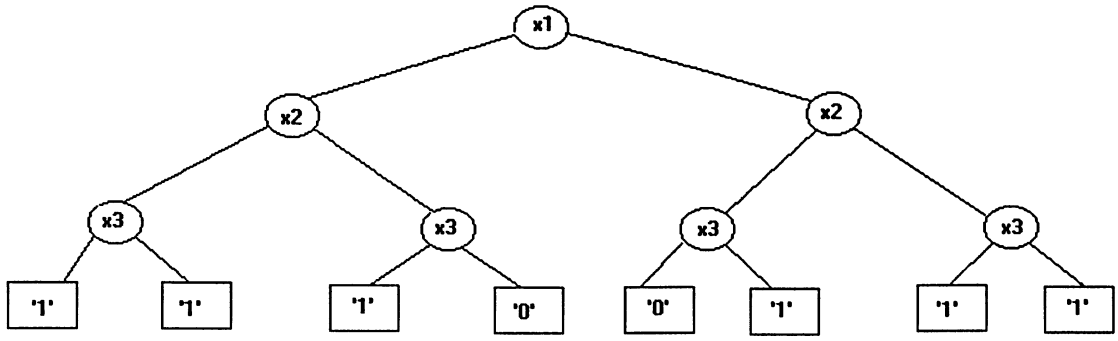


Figure 32 Sample ordered binary decision diagram (OBDD) tree

X1	X2	X3	Output
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

Table 6 Output boolean logic function for the circuit of Figure 32

An OBDD tree is a directed tree where all vertices except for the root and leaf vertices, possess one edge incident to them and two edges incident from them. The left vertices from the root always depict a choice of 0 while the right sides of the vertices always depict a choice of 1. The leaf cells determine the actual output of the functions. Table 6 depicts the Boolean function that was generated from Figure 32. Once an OBDD tree is obtained, a ROBDD tree can be obtained. ROBDD trees are similar to doing a Karnaugh map on the diagram itself and redrawing the minimized Boolean function as a binary tree.

Figure 33 shows the ROBDD tree equivalent to the OBDD tree from Figure 32.

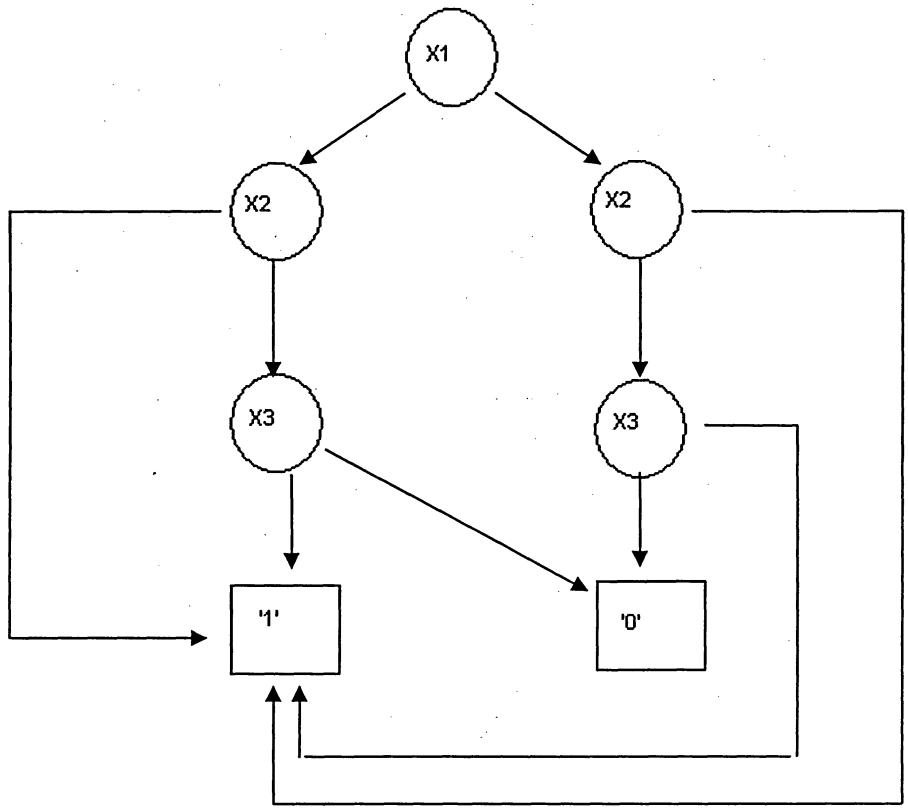


Figure 33 Sample ROBDD tree

5.1.4 Post synthesis simulations

Post synthesis simulations are another critical part of the design. These simulations are very similar to normal simulations with the main difference that this simulation includes all the gate delays from the actual library components used. Although they are theoretically identical, an added approximate gate delay has been imposed on the timing parameters of the simulation. Thus, the output of the gates does not change at the same instant as the inputs change but rather after a specified gate delay time. An example of post synthesis simulation is shown in Figure 34. In normal RTL simulations, if inputs A and B change to a logic '1', the output signal C should instantly change to a logic '1'. In post synthesis simulations, however, a gate delay is added before the actual signal output on C changes. Other than this detail, normal RTL simulations and post-synthesis simulations are pretty much identical. The gate delay methodology is added to ensure that even with gate delays in the system, the whole design still works.

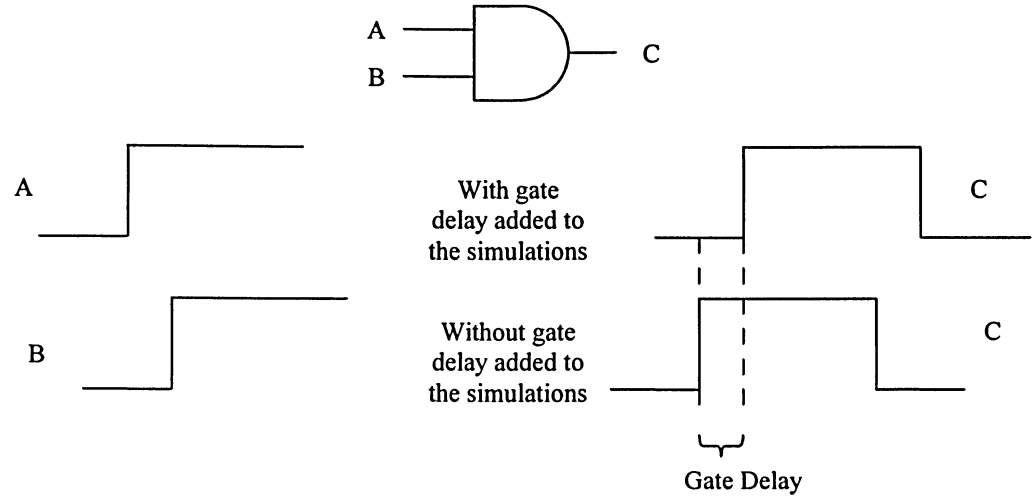


Figure 34 Gate delay simulation example

5.1.5 Floorplanning

As an increasing number of gates are being compressed onto integrated circuits, the need to layout the circuit more efficiently has never been more critical. Chip area is very expensive and as such, it is unacceptable to layout chips without properly thinking and utilizing the whole chip to its maximum potential. An unnecessarily large die for an IC chip will make it operate slower due to its longer distances between blocks. An excessively small die on the other hand will make routing very tedious and complex. Floorplanning allows a fix for all these issues by approximating and allocating distinct regions of the chip to specific sub blocks of the design and providing a quick estimate of the interconnection pattern for the design. The shape of the die as well as the aspect ratio is defined in the floorplanning stages along with the positions of the input and output pins [3].

In floorplanning for small and simple designs the whole design is either flat or only one level. Complex and bigger designs on the other hand cannot be flattened and must maintain their hierarchical design format. It is unwise to separate sub blocks with similar or related functionality all over the chip which could slow the design speed even more. Hierarchical design involves having to handle sub blocks or cells that comprise bigger cells depending on which cells are on which level of hierarchy. Cells in the lowest hierarchical level are called leaf cells while cells containing other cells are called composite cells. The parent cell is the main cell that contains all the other cells for the whole design and as such there can only be one parent cell in a design. There are different types of floorplan methodologies such as the slicing floorplan methodology and the wheel methodology. An example of a wheel floorplan (Figure 35) shows that the cells look like a wheel after they have been floorplanned [3].

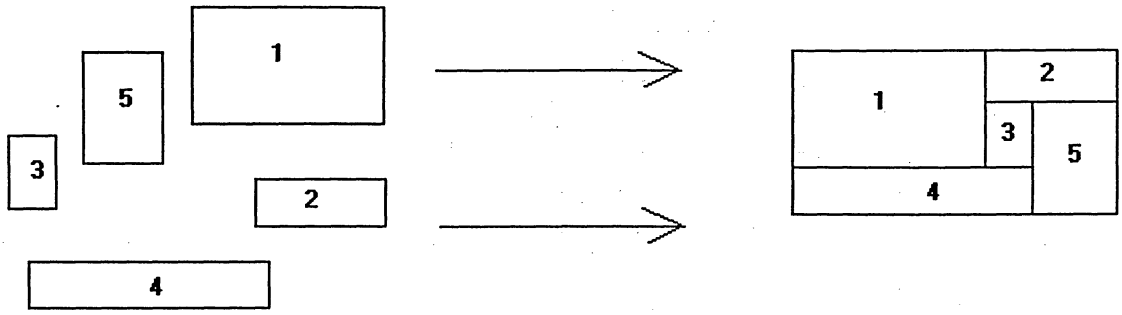


Figure 35 Sample wheel floorplan

In a slicing floorplan methodology, the shape of a composite cell is obtained from the shape function of its children cells. A sample slicing floorplan methodology is as follows [3]:

1. Construct the shape function of the top-level composite cell in a bottom-up fashion starting with the lowest level shape functions then moving upward.
2. Choose the optimal shape of the top-level cell.
3. Propagate the consequences of the choice for the optimal shape down the slicing tree until the shapes of all leaf cells are fixed.

5.1.6 Placement

The next step after floorplanning is the placement stage, where the actual digital component blocks are placed onto the die. The goal of this step is to minimize the area required. Standard cells are the hardest ASICs to place properly onto the die as these cells are predesigned and specialized. Cell based ASICs are easy and require almost no placement due to the fact that the whole die is basically nothing more than a sea of transistors that need to be properly routed. The two main placement algorithms can be categorized into constructive placement and iterative placement. Constructive placement algorithms do not allow cells that have been placed in position on the die to be moved. Iterative placement, on the other hand, allows a cell to move to another location. The best placement method combines both methods by using constructive placement algorithms to generate the initial placement and the iterative placement algorithms to iterate until an acceptable placement solution is found. One of the common constructive placement algorithms is the min-cut algorithm which splits the circuit into two separate circuits at a point where there are a minimum number of nets connecting both sub blocks. This allows for minimization of the interconnecting nets and fairly easy routing, the main goal of placement [3].

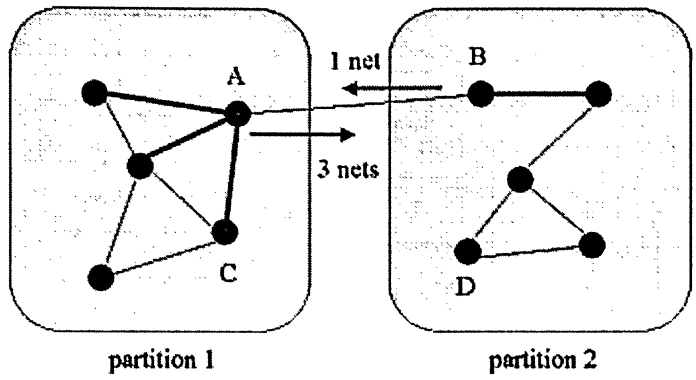


Figure 36 Example of min cut design

In the above example of the min-cut algorithm (Figure 36) the design partitions are separated at the point where only four nets cross both designs. This algorithm can be repeated until a satisfactory number of wires travel between partitions or modules. The iterative placement process follows the constructive placement algorithm. The iterative placement algorithm upgrades the placement of the design to improve the placement.

5.1.7 Clock Tree Generation

5.1.7.1 Theory and Design

Clock tree generation is one of the most critical parts of the design since the clock tree is the highest fan out signal which also operates at the highest frequency. The clock in every design is the signal net with the most restrictions. A problem with a design's clocking scheme could make the design totally useless. Not only are the timing delays important, but clock skew and the impedance of the routing for the clock nets are critical as well. This is the reason why only this signal net has a separate step just to estimate the clock routing path delays including the skews for them.

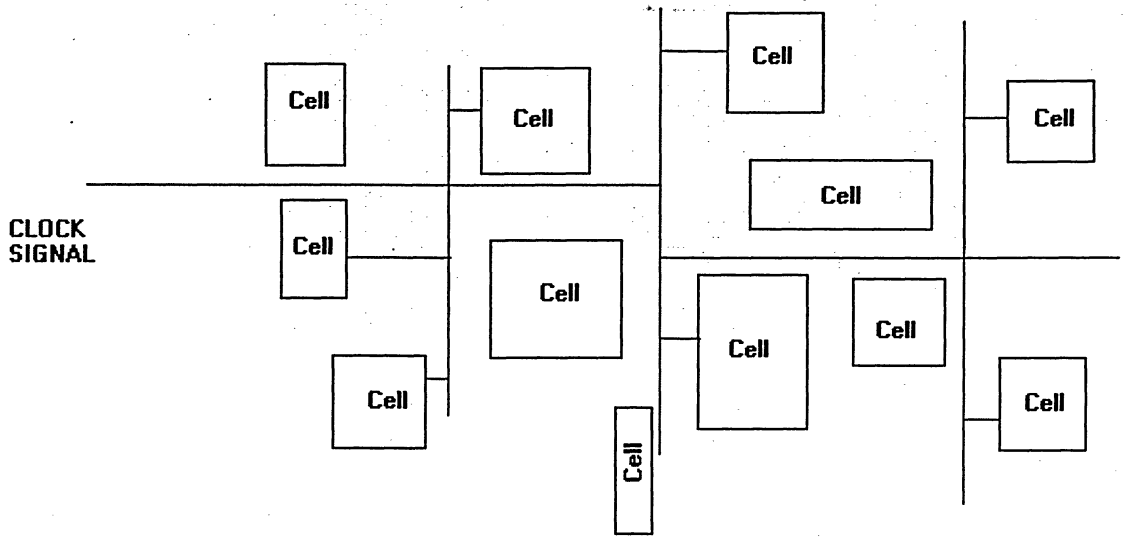


Figure 37 Example of a clock tree

As evident from the clock tree in Figure 37, the nets of the clock signal are similar to tree branches, hence its name clock tree. The clock signal arriving at the different cells must be more or less aligned with minimal skew levels and delays. This design of clock tree routing is most widely used. The clock tree was designed with maximum delay parameters of 5ns, maximum transitions of 200ps, skew of 100ps and depth of 10 levels [3].

5.1.8 Routing/Timing

Once the clock tree process is complete, the next step is to route the design. This is one of the most complicated process steps in the ASIC design process. The routing of wires is tedious since thousands if not millions of wires need to be routed all over the chip die while still ensuring that the timing parameters are met. The routing goal is to connect cells that have been assigned positions by the placement step with the best possible timing and area consumption. Usually, designs are routed in three steps: clock, power, and then the remaining signals. Clocks are routed first since they are the most critical portion in the design. Each routing step consists of two stages namely global routing and detailed routing steps. Global routing determines which channels the wires are to be routed on. Detailed routing is a lot more complex in the sense that it specifies the actual path and route that the net takes to the destination cell. Global routing is nothing more than a good estimate and preparation for detailed routing. Routing the design can be made easier or harder depending on the maximum allowable metal layers that can be used per design. The more routable metal layers, the easier it is to route the design since more paths are opened to enable more channels for routing. Unfortunately, even

though more metal layers increase the routing channels, complexities also arise through the required vias that need to connect them. Gridded routing is another issue in routing since routing is only allowed to be either in the vertical direction or the horizontal direction. This can be a major issue when lots of cells or wires are blocking the path of one of the nets to be routed. Figure 38 shows a primitive and inefficient way of gridded routing [3].

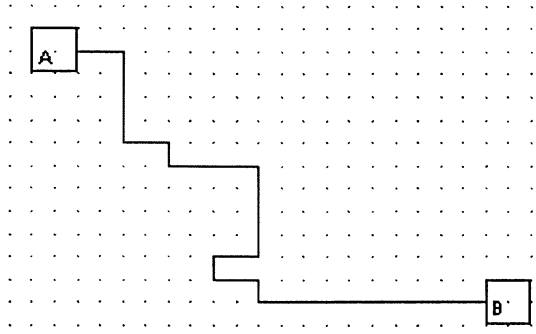


Figure 38 Example of gridded routing

To solve issues with gridded routing, an algorithm known as Lee algorithm or path connection algorithm is used to allow for efficient routing in gridded routes along with the ability to detect and avoid obstacles. The algorithm uses wave like patterns to analyze the available paths before it actually routes the net. It finds the shortest path without going into obstacles by using cost functions with the main cost being the number of grids it has to pass through. An obstacle in the path means cost = infinity. Figure 39 shows the ideal gridded routing wherein minimum distance between points A and B is achieved.

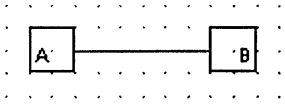


Figure 39 Example of cost functions in the Lee routing algorithm

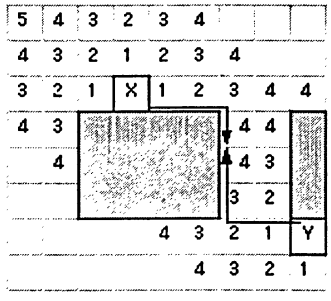


Figure 40 Example of Lee routing algorithm

Figure 40 shows the Lee maze-running algorithm. The algorithm starts from X and tries to find a path to Y by emitting a wave from both the source and the target. The cost for running through the grid is labeled by the grid itself which determines which minimum cost path to take. Once the target is reached, the path with the lowest cost for arriving at the target is used as the path. If more than one metal layer is used, the same concept can be applied to multiple metal layers although it would naturally increase the complexity of the routing algorithm.

5.1.9 Post routed simulations

Post routed simulations are another critical part of the design. Although very similar to post synthesis simulations, this simulation includes all the routing delays from the nets that have already been routed as well. Although both types of simulation are theoretically the same, an added approximate wiring delay to the timing parameters of the simulation has been imposed. In reality, all nets have delays as the signal traverses through the net before it reaches the destination. Routing delays are not included in the post synthesis simulations since they are not known until after routing. The wiring net delay methodology is added to ensure that the whole design still works even with wiring delays in the system.

5.1.10 Physical layout

The final process step involves the physical layout of the chip where the actual poly layers as well as N+ and P+ diffusions is done. There are multiple design rules that need to be followed when laying out the actual physical circuit design. Since the actual physical design involves placing the transistors on the silicon, it is even more critical to ensure that all rules are followed due to the millions if not billions of transistors on the chip. The design rules can be separated into four main categories: Minimum Width Rules for trace durability and connectivity, Minimum Separation Rules for shorting, Minimum Extension Rules for connectivity, and Overlap Rules for connectivity. These are strictly implemented since there are tolerances when creating any IC chip. In other words, the design should always account for such errors to ensure it still works. An example of such a design rule is the Minimum Width Rule for the poly layer. This width defines the technologies being used for the current CMOS devices such as 0.35u, 0.18u, 0.13u, 0.9u and even 0.045um [3] [14].

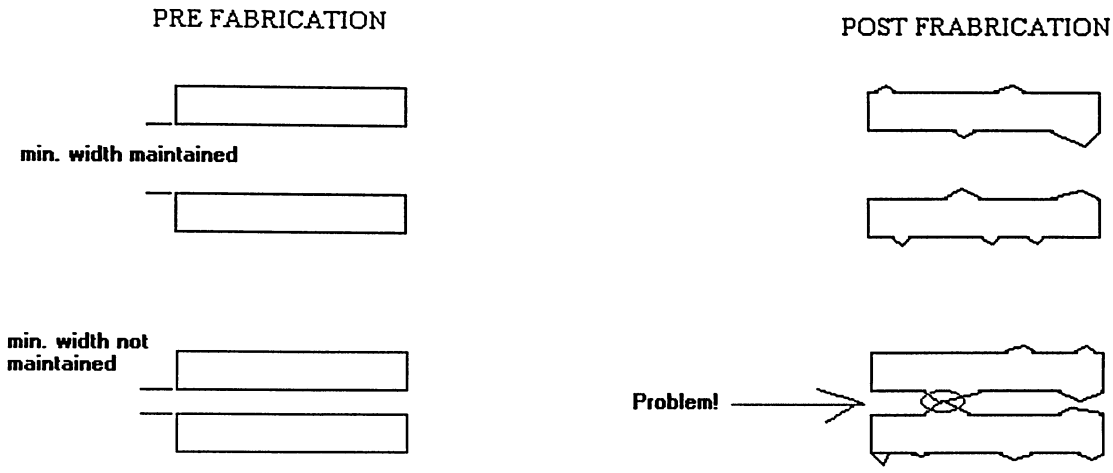


Figure 41 Example of Minimum Separation Rule

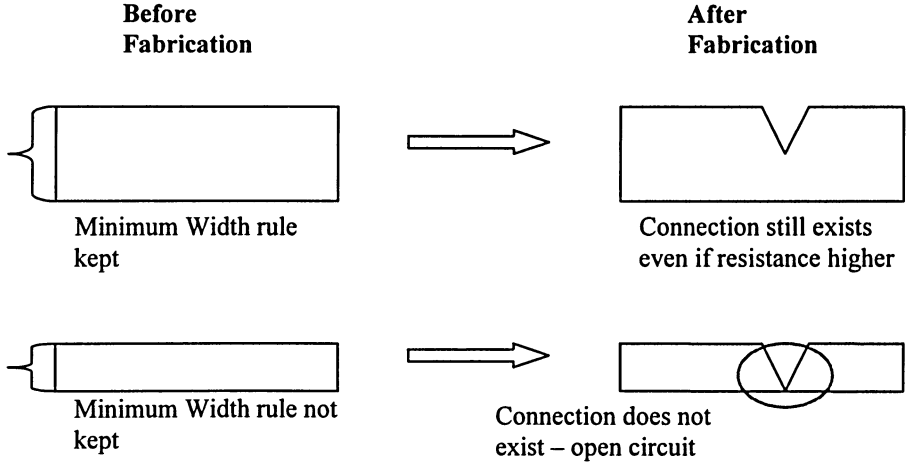


Figure 42 Example of Minimum Width Rule

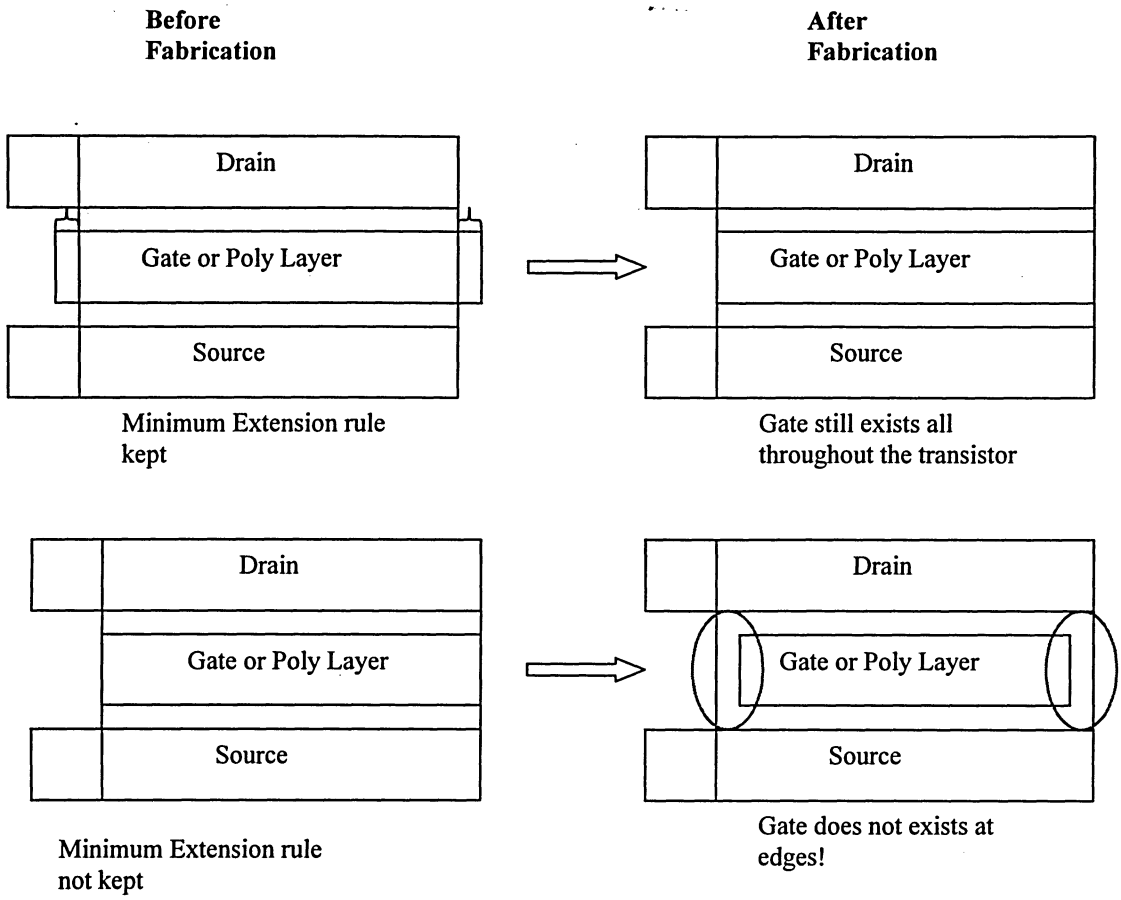


Figure 43 Example of Minimum Extension Rule

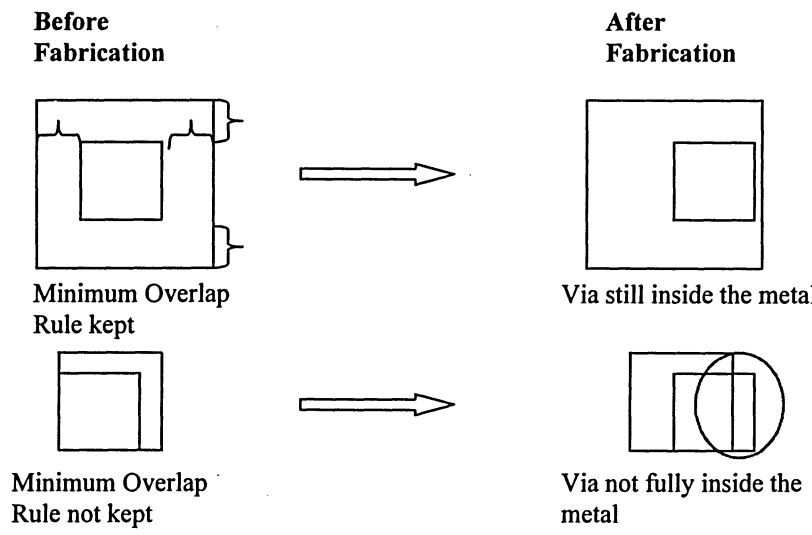


Figure 44 Example of Minimum Overlap Rule

Figures 41 to 44 show the different scenarios of errors that could occur if the basic physical layout rules are not followed. Violation of any of these basic rules will result in performance degradation and less

durability. Figure 45 illustrates an example of the actual layout for an NMOS transistor with the basic design rules being met.

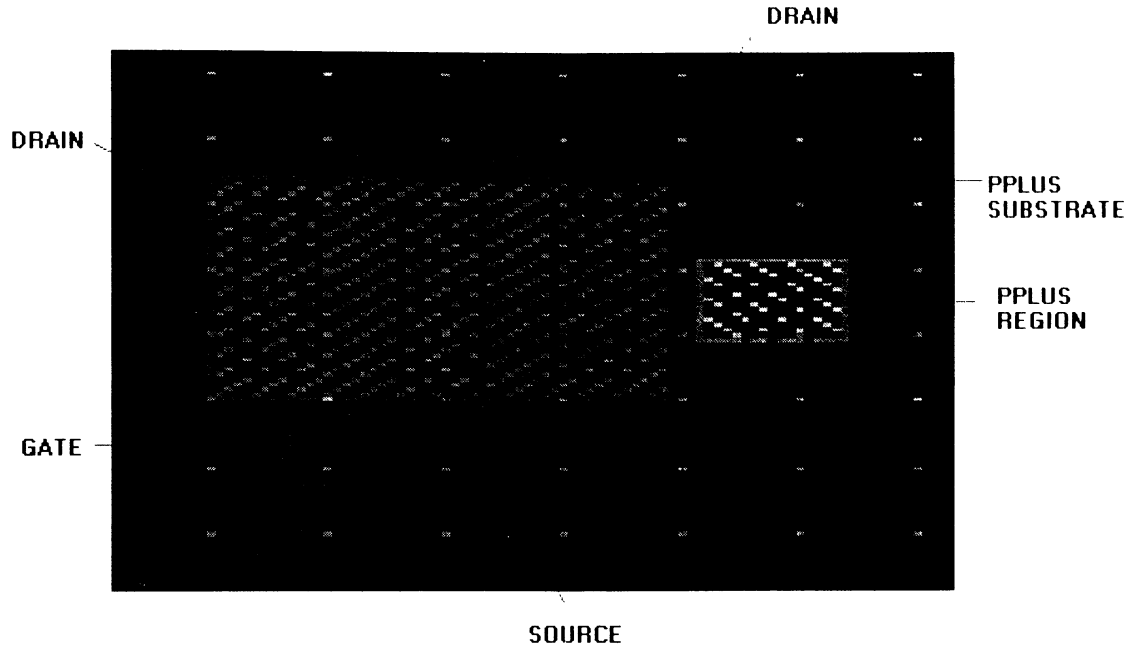


Figure 45 Example of NMOS physical layout

Chapter 6 Digital ASIC IC Design Process Techniques

6.1 ASIC tools and implementation

Application Specific Integrated Circuit (ASIC) design requires extensive knowledge of each of the circuit design methodologies and the algorithm behind the tools. The two major architectural structures for an ASIC are standard cell and cell based array (structured ASICs). Cell based arrays technically consist of a sea of transistors which allow for lower NRE costs, shorter prototyping time at lower efficiency and less flexibility to the design. Standard cells are designed literally from ground, i.e. nothing is predefined. NRE costs are higher and prototyping is long but the design is more efficient and flexible.

This project is created with Ryerson University's Mentor, Synopsys and Cadence tools for ASIC design and debugging purposes. The tools used include Modelsim from Mentor, design analyzer from Synopsys as well as NC-Verilog, Encounter, and IC Designer from Cadence. Modelsim and NC-Verilog were used to debug, simulate and verify the RTL code's functionality as well as to verify the gate simulation with back annotated timing. The design analyzer synthesizes the RTL code and generates the appropriate netlist and timing file using the modules available in the technology library specified. Cadence's Encounter handles the placement, floorplanning, clock tree generation and routing for the design while the IC designer is responsible for the actual physical layout of the IC onto the die along with the metal layers.

6.1.1 RTL

Appendix A contains many design guidelines and recommendations for the practical design side of RTL design.

6.1.2 RTL simulation

The RTL simulation of this project was implemented using Mentor's Modelsim. This tool is very flexible yet powerful since it allows for mixed HDL simulations and has a wide variety of features that allow ease of use. The main design files used to create this project are: core_adp.vhd, adp.vhd, fifo.vhd, dpmem.vhd, ind.vhd, index.vhd, and refer.vhd.

The hierarchical structure of the design is as follows:

- Core_adp.vhd
  - Adp.vhd
    - Fifo.vhd
      - Dpmem.vhd
    - Ind.vhd
    - Index.vhd
    - Refer.vhd

The input and output test files for input stimulus and output signal comparisons are: delay_est.vhd, laser_model.vhd, multisigned.vhd, p2r_corproc.vhd, p2r_cordic.vhd, p2r_cordicpipe.vhd, r2p_corproc.vhd, r2p_cordicpipe.vhd, r2p_pos.vhd, r2p_pre.vhd, and signal_input.vhd.

Core_adp is the top level module under which the rest of the files are sub-modules. In RTL simulation, all the files created for the RTL design are required for the design to be simulated properly. Later chapters will discuss why top level modules need to be added later on for simulations and how some other files such as libraries and timing constraint files are required to do the post synthesis and post routed simulations. RTL simulation only verifies functionality of the design: it includes no timing constraints and the libraries to define the actual components that become the digital building blocks are not yet defined. As depicted in the structure above, the whole design was added into the Modelsim project and simulated with the test input files while being analyzed by the user on the waveform viewer.

Figure 46 shows the output waveform of the top level core_adp.vhd module where it is evident that the design works properly and as expected. This simulation only displays the functional simulation of the RTL design. Since this is a synchronous design, a simple RTL functional simulation would do but this project implements post-synthesis and post routed simulations as well. The critical signals monitored were the i_out, q_out and drdy_ld signals which control the ADP blocks.

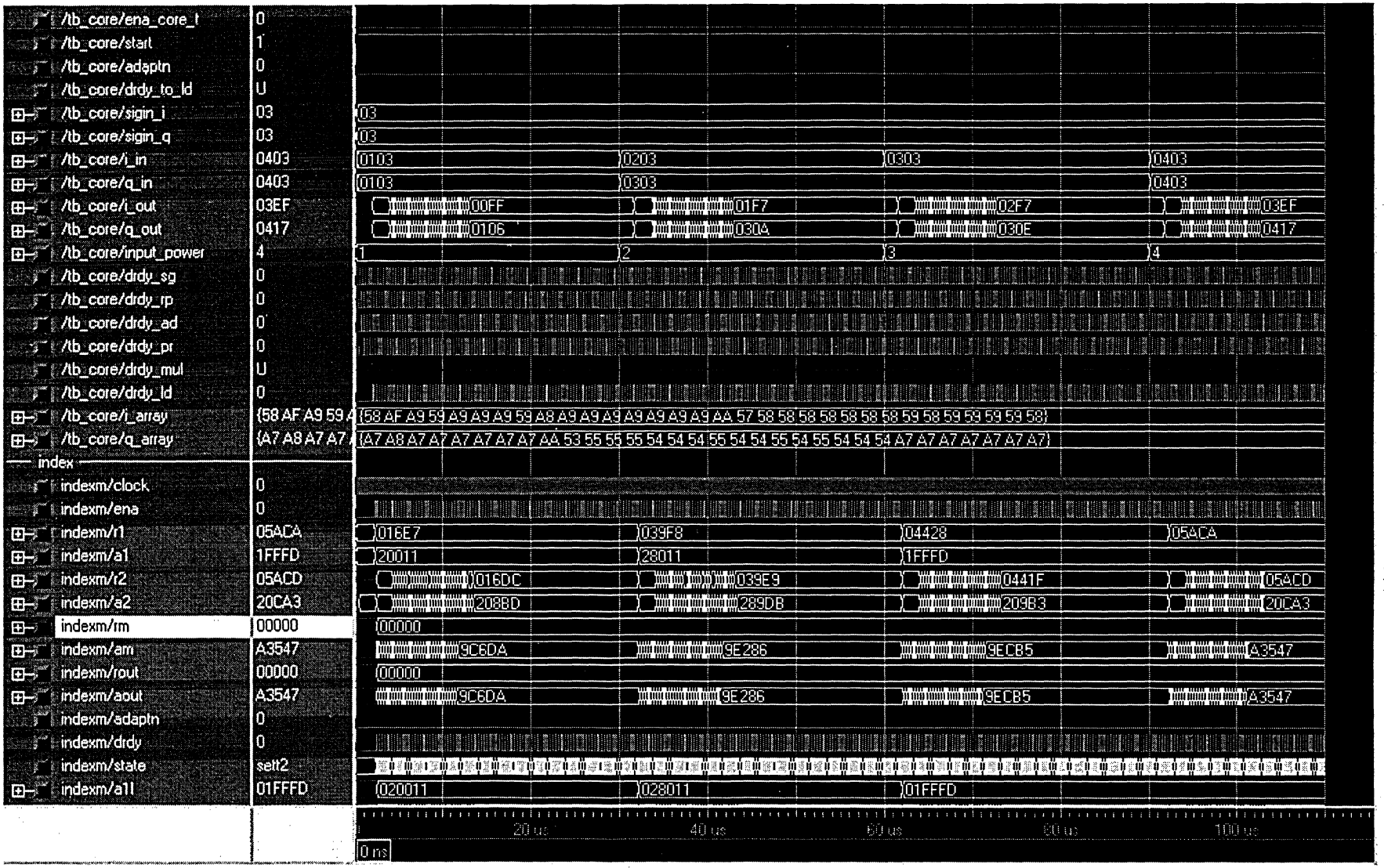


Figure 46 Design RTL simulation output waveform

6.1.3 RTL synthesis

Before design synthesis, several constraints must be specified, including clock signal absolute maximum skew, clock frequency, I/O load capacitance, I/O pad types, and mapping schemes. Other optimization techniques allow for the use of special blocks such as multipliers, fifo's, RAMs, and ROMs. For more complex ASIC IC designs specific restrictive timing constraints on delayed signals are added in the design.

1. Clock Signal Absolute Maximum Skew

Clock skew refers to the difference between the actual clock rising edge and the circuit's received clock rising edge. In any complex digital circuit, the main clock being routed through the whole design does not arrive exactly at the same time; there are skews between received clocks from different flip-flops as shown in Figure 47. There must therefore be a limit on the specified clock skew. Clock skews only make sense for flip-flips that cascade to each other in the same clock domain. Other than this scenario, clock skews are meaningless [3]. For this design, the clock skew is constrained to 0.5 ns for both positive and negative skews.

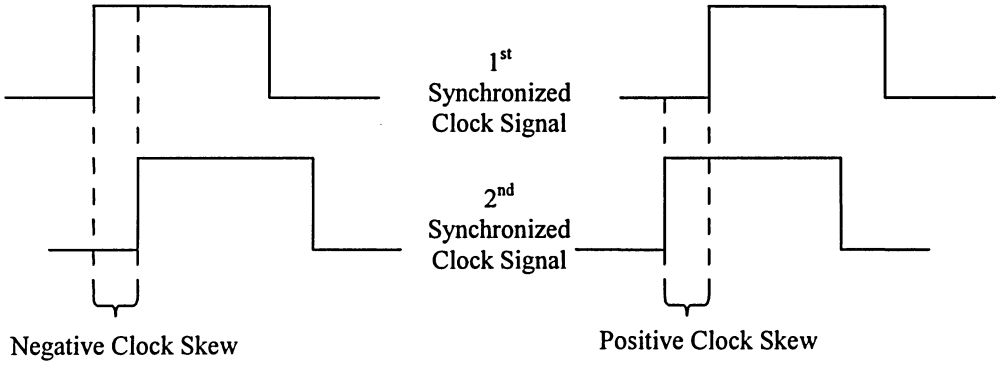


Figure 47 Clock skew example

2. Clock Operating Frequency

Clock operating frequency is another major constraint that needs to be defined in the synthesis section. The clock frequency defines the maximum frequency the whole design can operate at and limits the propagation time between combinatorial signals. This means that the critical path within the system must have a combinatorial and path delay that is less than the minimum clock period [3]. For this design, the clock frequency was set to 50 MHz.

3. Input/Output Load Capacitance

Input/output load capacitance also needs to be specified as this constraint defines the type of I/O pad or buffer that needs to be used. It also enables the synthesis tool to know how many fan outs are required by the system. Excessive load capacitance on the outputs of the IC pins causes a delayed signal rise time and as such limits the speed that the signal can toggle on the output [3]. For this design the load was assumed to be 20 pF. Figure 48 shows an example of parasitic load capacitances on the circuit while Figure 49 shows the effects of such parasitic capacitances to the circuit performance.

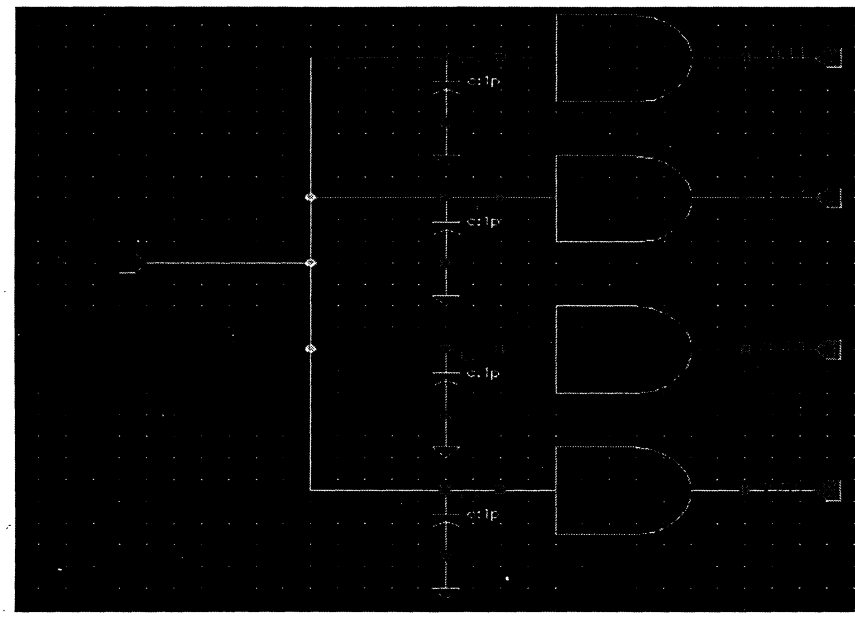


Figure 48 Parasitic load capacitance

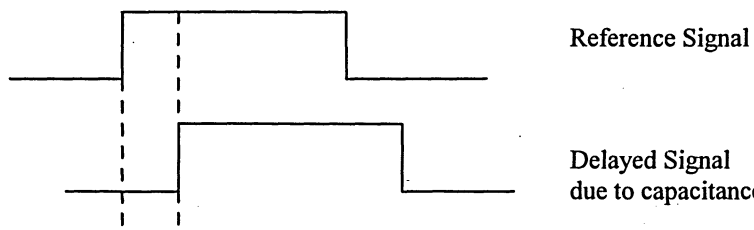


Figure 49 Delayed output signal due to load capacitance

4. Input/Output Pad Type

Input/output pad type is also defined in this stage since it is necessary for the designer to determine the critical signals and the pads to which they should be assigned. There are special input and output pads for specific signals requiring fast transitions such as clock signals or even for signals with high fan out requiring large drive currents. Rather than letting the synthesis tool do this, it is recommended that the designer look at the specified pads available from the foundry libraries and add them accordingly [3]. The input pads used for this design were PDIDGZ due to the chosen clock frequency of 50 MHz while the output pad chosen for the design was PDO08CDG which satisfies the functionality.

5. Mapping optimization

Mapping optimization defines whether or not a flat design is to be implemented or a hierarchical design is to be made. A flat design means that no grouped blocks will be saved onto the design and only a flat design of all gates and flip-flops will be synthesized. This is good for small designs which allow the synthesizer to optimize the design more efficiently since most synthesizers have a hard time optimizing through hierarchical blocks. However, hierarchical design mapping is better for larger designs even though the whole design might not be fully optimized. Functionally similar circuits are grouped together to allow for better placement and routing later in the design. A situation where a whole flat design and the inverter attached to the input of the flip-flop is on the leftmost end while the flip-flop is on the right most end would be disastrous for the critical path in the design [3]. This design was mapped using the hierarchical mapping method.

6.1.3.1 Design figures from synthesis

The synthesis of the design was implemented using a design analyzer. The figure below depict the outputs from the synthesis process.

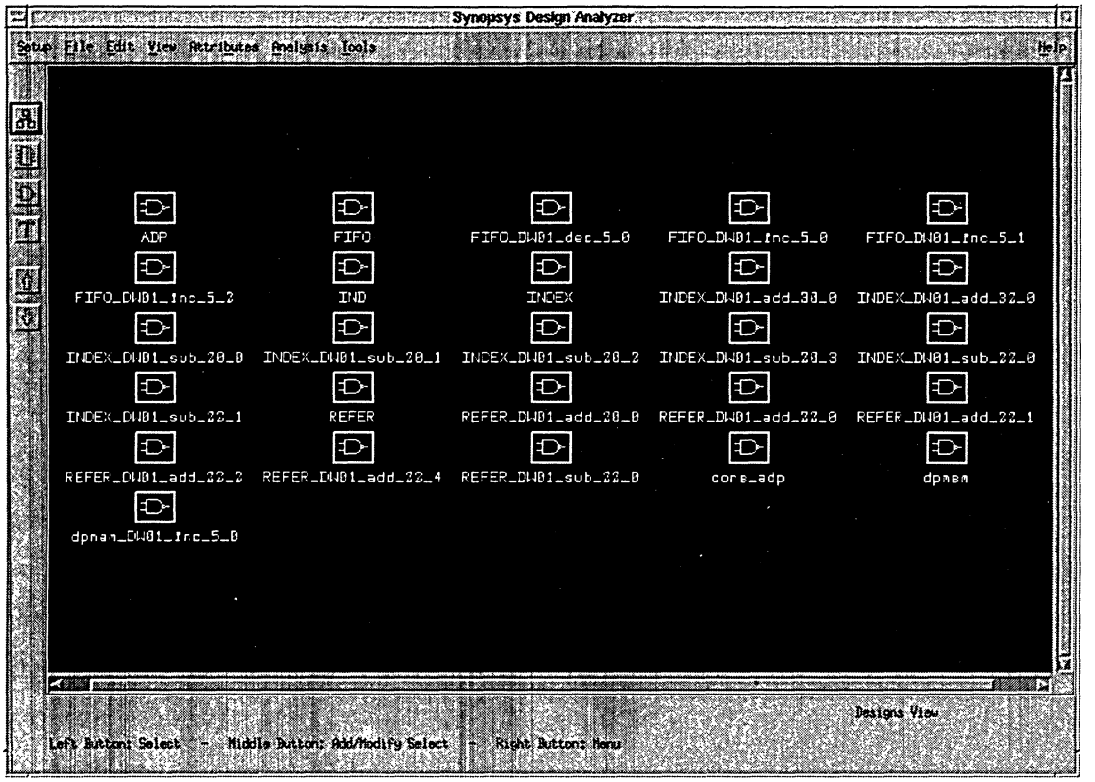


Figure 50 Design modules

The design modules were read into the design analyzer and synthesized into separate blocks for each one of the required instantiated block designs. Figure 50 shows multiple refer blocks due to multiple instantiations of this block in the design. The index block was instantiated multiple times. A script named tutorial.build was used to generate and synthesize the whole design with the constraints specified in the previous section.

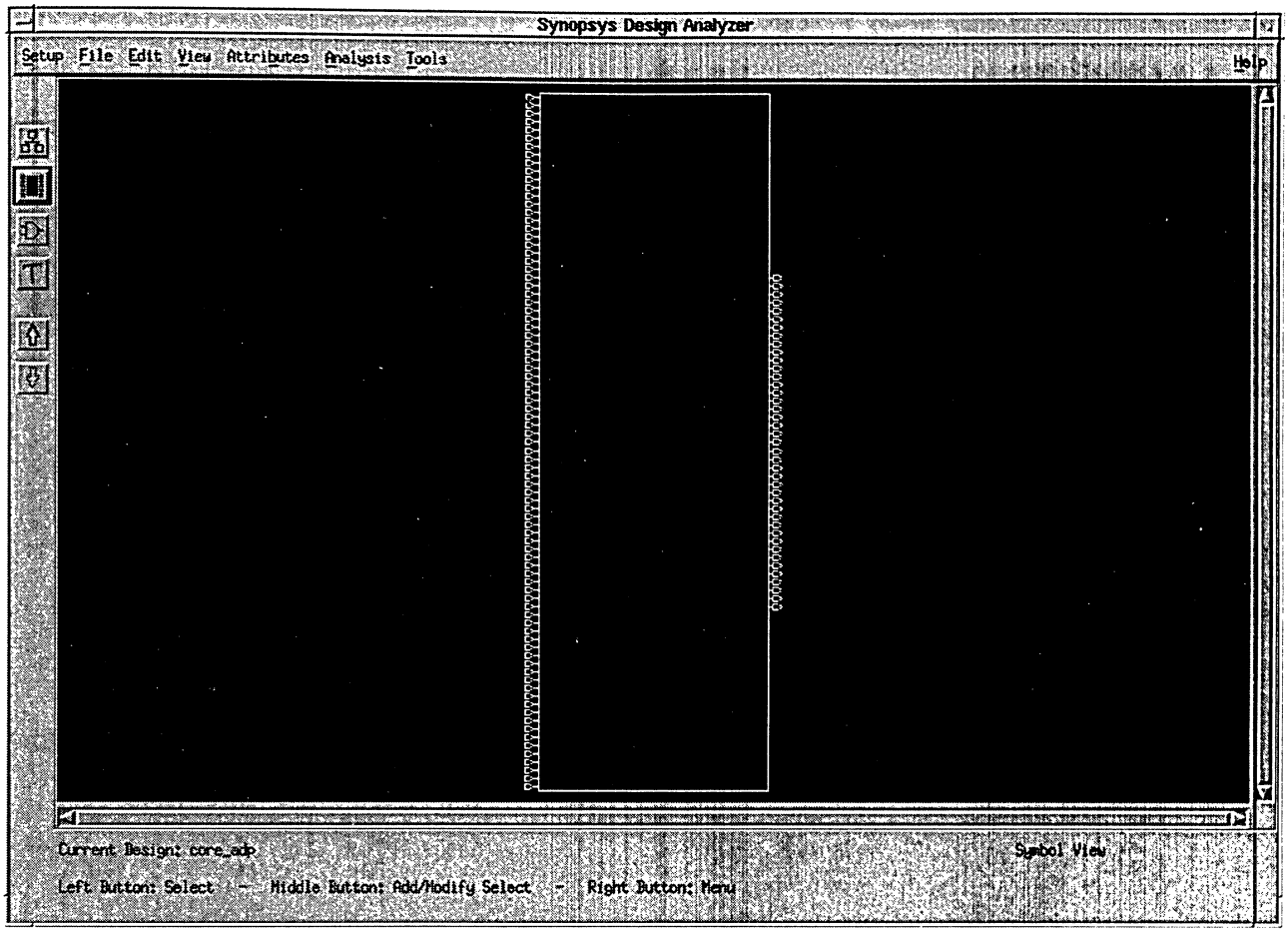


Figure 51 Top level symbol

Figure 51 shows the top level block diagram of the design after synthesis, with the proper constraints. A waveform on the second pin from the top identifies this signal pin as the clock pin. The skew and operating frequency constraints are already defined.

6.1.4 Post synthesis simulations

Modelsim from Mentor was used for the post synthesis simulation in this design. The tool is has the capability to read sdf (standard delay format) files to specify any timing constraints on the design. The design file used was core_adp.vhd. Unlike the previous design files, core_adp.vhd is no longer an RTL design described in code, but rather a netlist showing the interconnections between blocks. This netlist is the synthesized and constrained code generated by the synthesizer. The core_adp.sdf file is also used to define the timing constraints for all the blocks used inside core_adp.vhd. All these timing files and synthesized code are used to generate the post synthesis simulation which includes the gate delay timing.

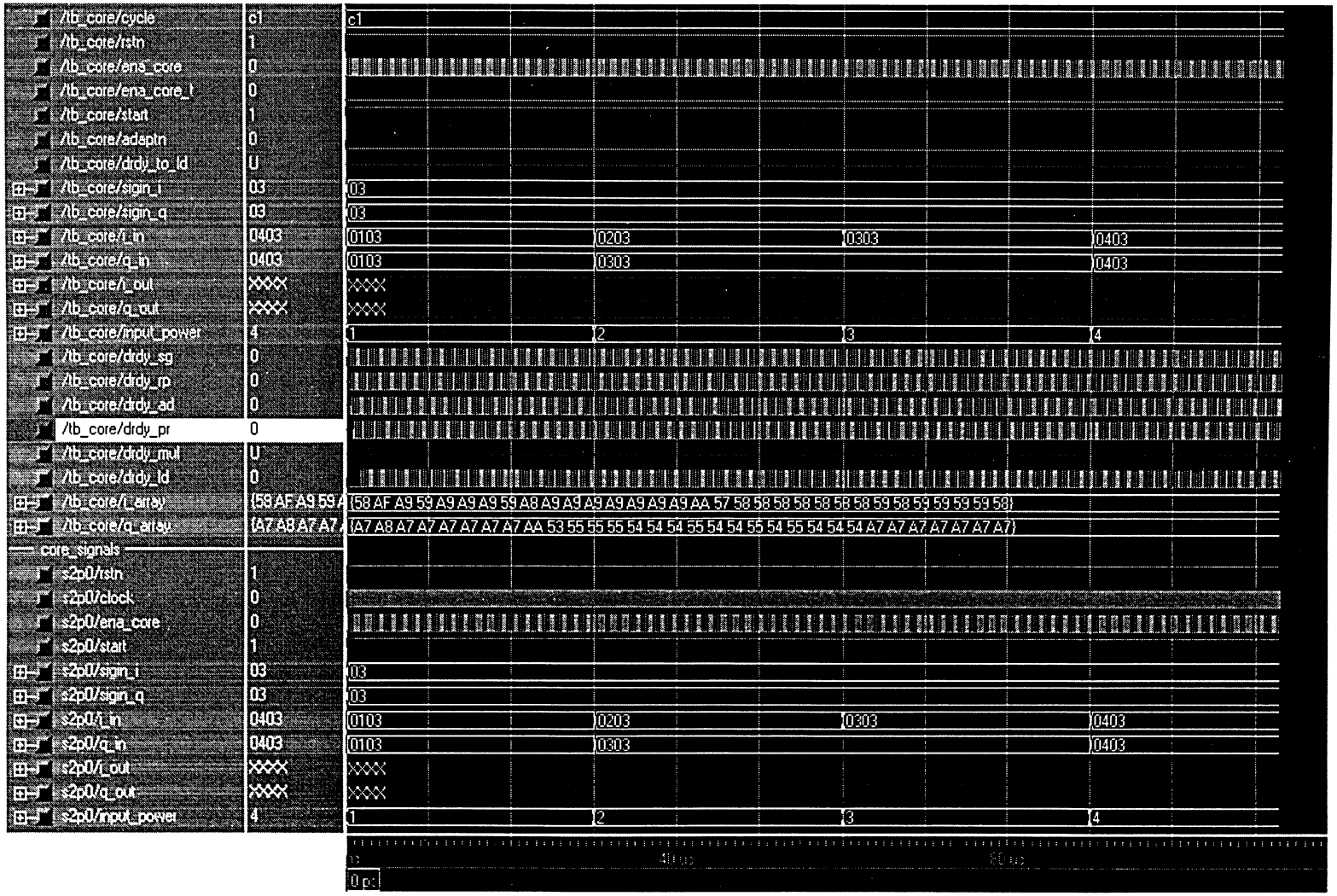


Figure 52 Gate delay simulation example

Figure 52 simulation waveforms for post synthesis simulation may not appear identical to the RTL simulation in the previous section. In the previous design described in RTL code, it was possible to initialize signals in the code itself since no actual hardware was involved. After post synthesis, however, RTL code no longer exists and only the actual netlist for the blocks remains. As such the signals cannot be initialized as easily. In normal practice, all sequential hardware elements such as flip-flops are reset into a known default value if the input reset signal is high. Therefore, the post synthesis simulations show undefined signals as all the memory elements in the design are not defaulted to any value. Also, since feedback exists in the memory elements and in the design, the flip-flops cannot get to a known state just by toggling the input signals to the circuit. This problem exists at the post simulation results as well.

6.1.5 Floorplanning

The floorplanning tool used to implement the design is Cadence's Encounter. This design uses an aspect ratio of 1 to make the x dimensions and the y dimensions identical or the same. The power pads for both the IC's IO pads and core circuits are distributed evenly on the four sides of the integrated circuit. The width and spacing of the power rings and power stripes also need to be defined. For this design, the power ring width was set at 2.88 with a spacing of 3.38. The same values were used for the power stripes. The IO pad placement was also specified by a created placement file, which indicated to the floorplanning software exactly where to place the pads and what pads to use. The libraries used in the floorplanning stages of the design were CMC's 0.18um technology library. The timing parameters and files associated with them were also imported to the Encounter tool. The main files to be imported before the actual floorplan is implemented are the netlist files, timing constraint files, IO placement file, libraries and timing libraries.

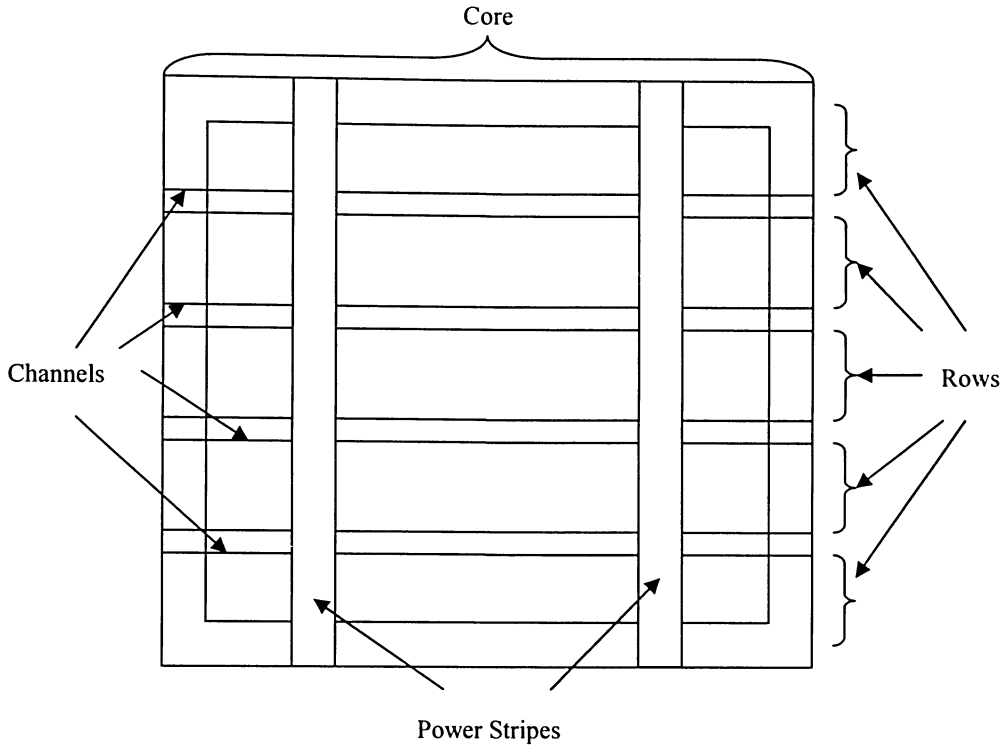


Figure 53 Power stripes, rows, channels of the chip die

Figure 53 indicates where the power stripes are normally located in the chip's die design including the rows where the actual component cells will later reside. The channels are the routing channels to ensure that the design is still routable. This is a good example of what a normal ASIC design would look after it has been floorplanned.

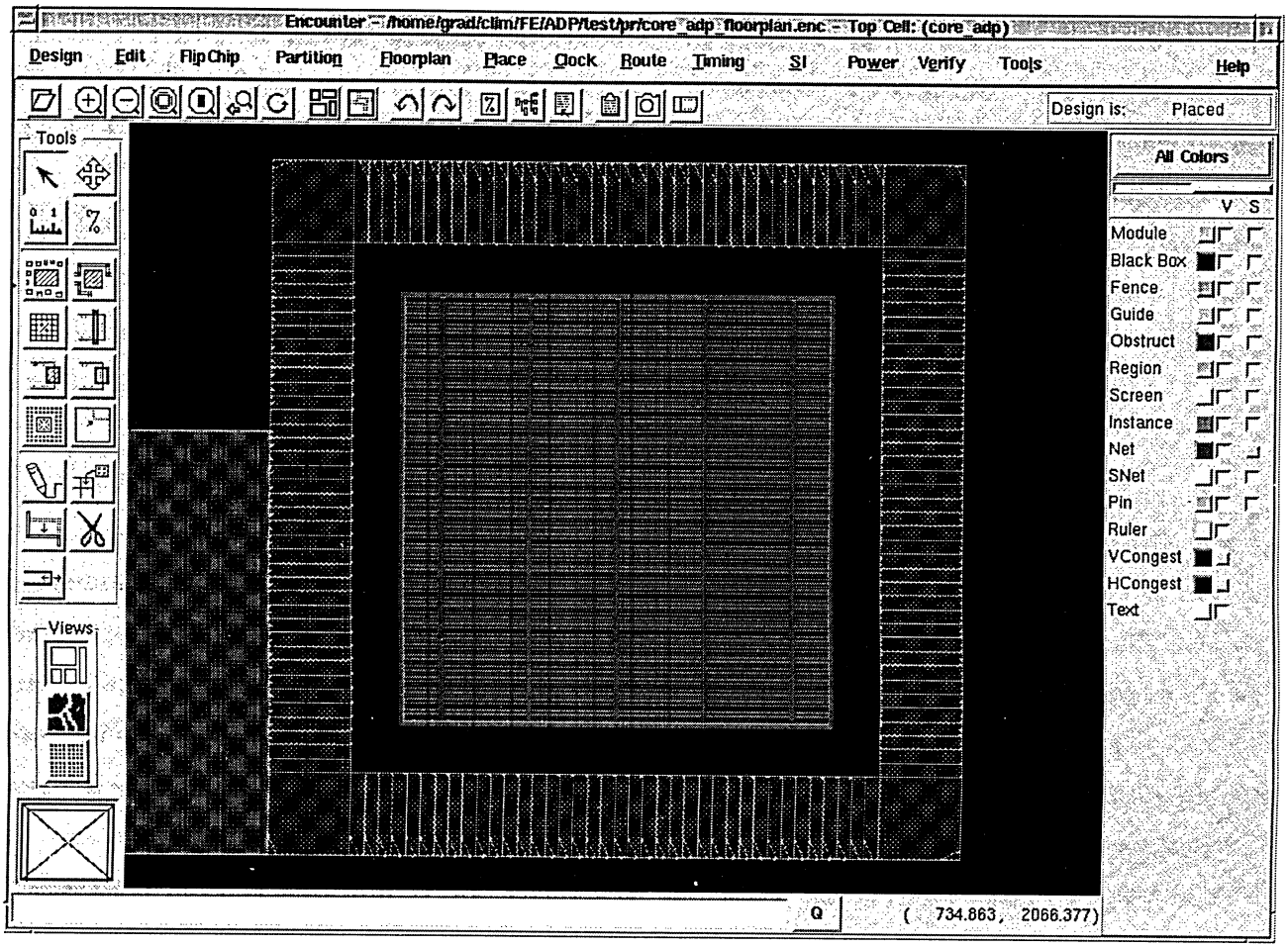


Figure 54 Floorplanned design after being imported into Encounter

Figure 54 depicts the finished floorplanned design in Encounter. The vertical and horizontal lines around the core are the power rings and power stripes that supply power to the core circuitry. The I/O pads of the design can be seen on the 4 corners including the feeder cells to ensure that the aspect ratio of the design could be made to 1. The component names of the I/O pads aren't visible but by examining the floorplan closely, the actual blocks used for the I/O pads can be determined.

6.1.6 Placement

The Encounter tool from Cadence was used for this design to implement the placement of the cells in the design. The placement effort was set to medium for the design.

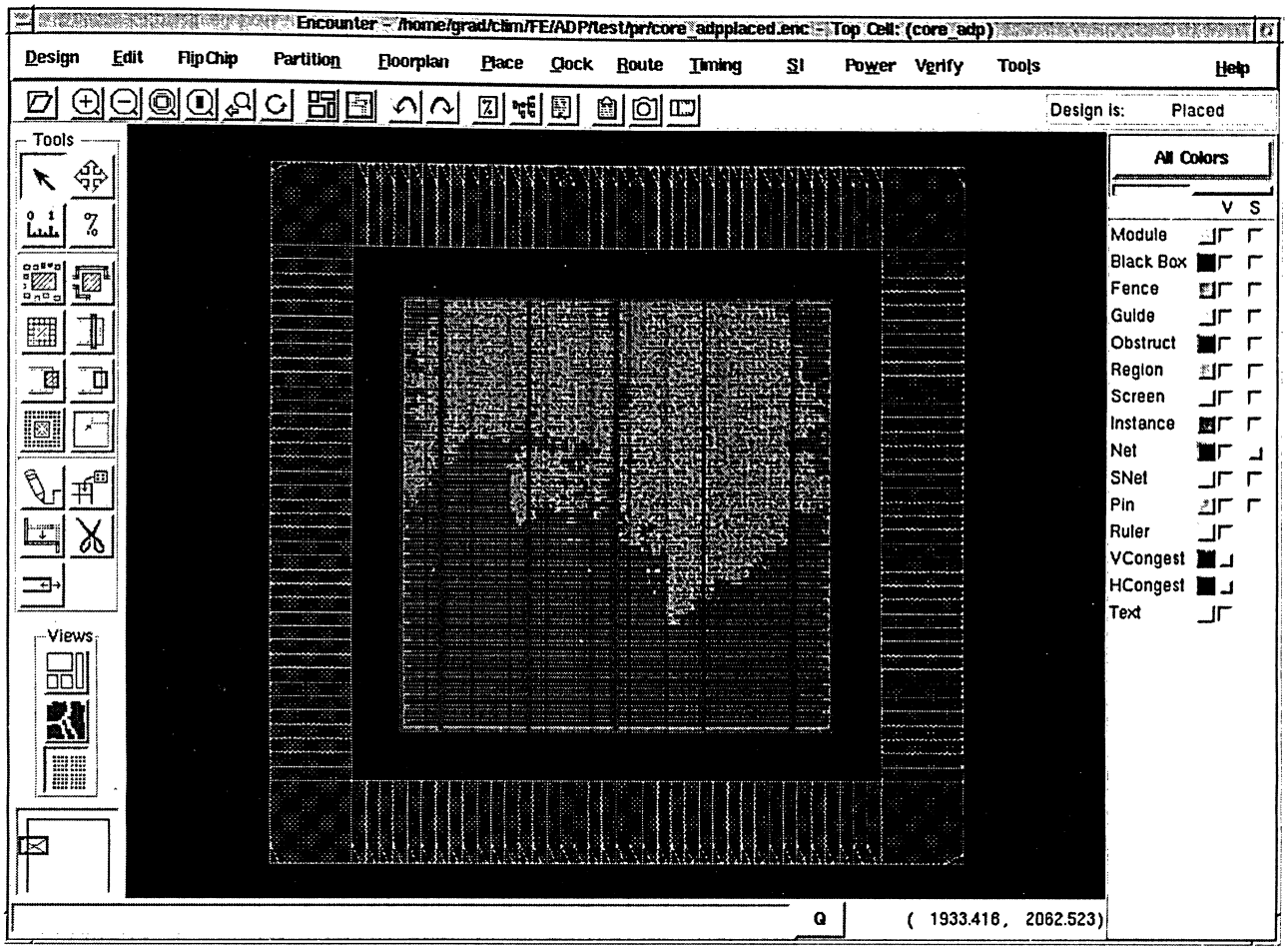


Figure 55 Final placed design

Figure 55 shows the design after the completed placement of all the sub-modules. Examined in greater detail, it is actually possible to look at each cell and its properties along with all the wiring connections to the other blocks. It is also possible to define manually where the hierarchical blocks are being placed through partitioning.

6.1.7 Clock Tree Generation

Please see section 5.1.7 for theory and design of clock trees.

6.1.8 Routing/Timing

The design was routed using the clock, then the power was routed, followed by the rest of the signals. Encounter was used to route the clock, power signals and the rest of the remaining signals. Figure 56 shows the design after the power routing and clock routing has been done. The core and the ring powers were already routed to the IC pads. Figure 57 shows the whole design after it was routed. The different colors in the design show the different metal layers that were required to route the whole design. The technology used for this design is a 6 metal layer process at 0.18um technology.

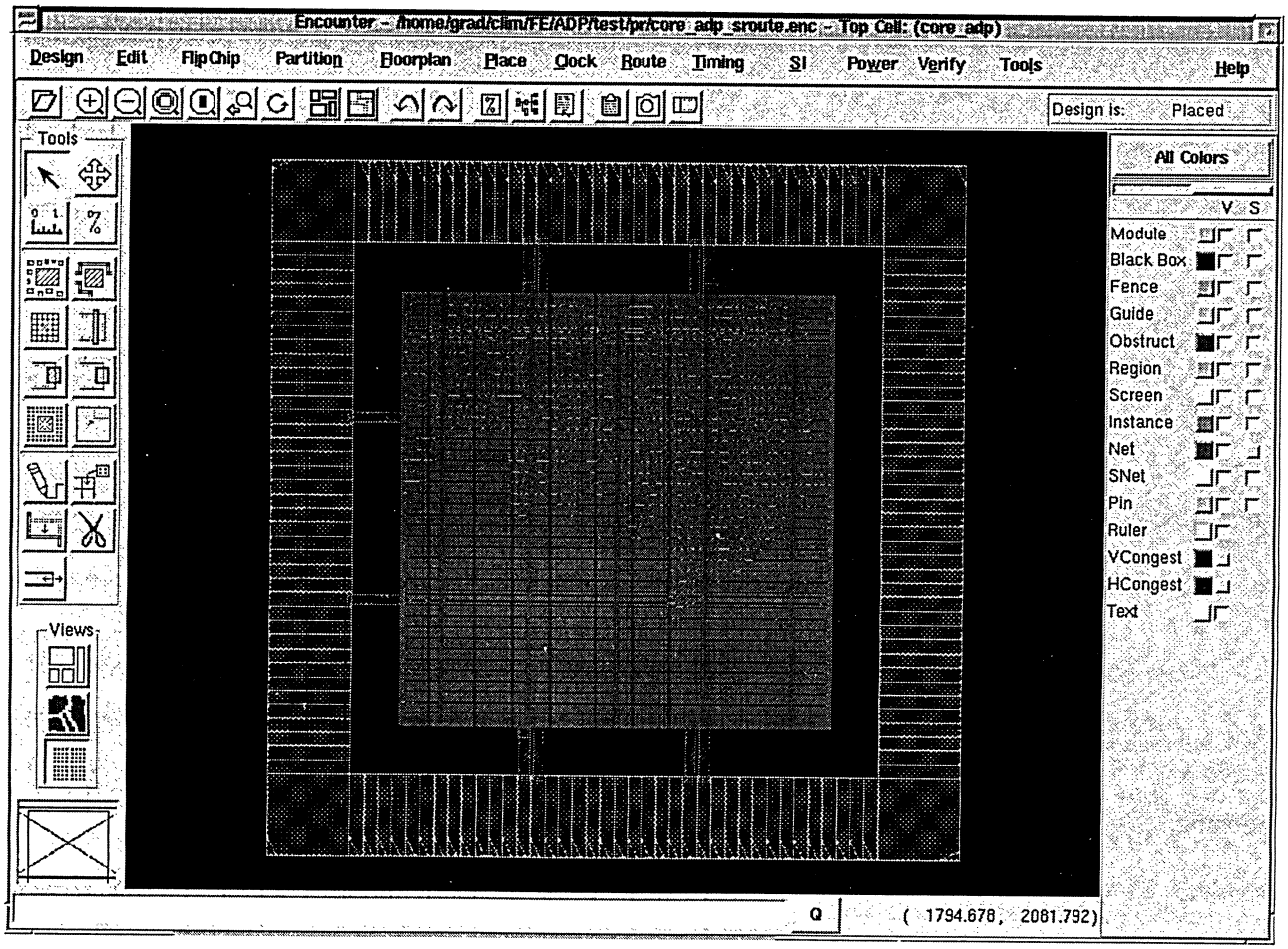


Figure 56 Power routed design

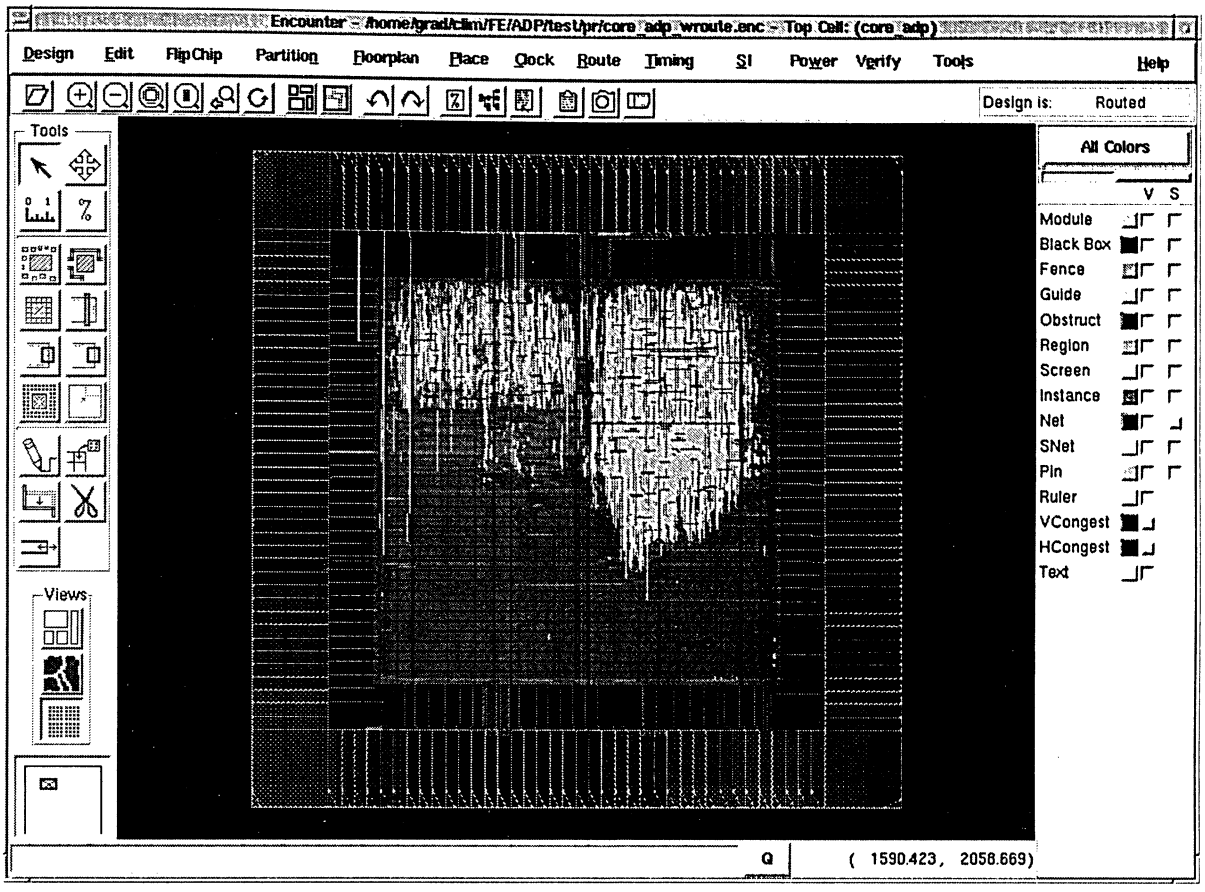


Figure 57 Routed design

6.1.9 Post routed simulations

The post routed simulation in this design was also done using Mentor's Modelsim. The tool has the capability to read sdf(standard delay format) files to specify any timing constraints on the design. The design file used was core_adp_netlist.v, which now has all the necessary instantiated library design blocks inside. Unlike the previous design files, core_adp_netlist.v is no longer an RTL design described in code, but merely a netlist showing the block interconnections along with the routed wires. The core_adp.sdf file is also used to define all the timing constraints for all the blocks used inside core_adp_netlist.vhd. Libraries describing all the instantiated modules in the netlist have been added to the simulation as well to ensure proper simulations of the design.

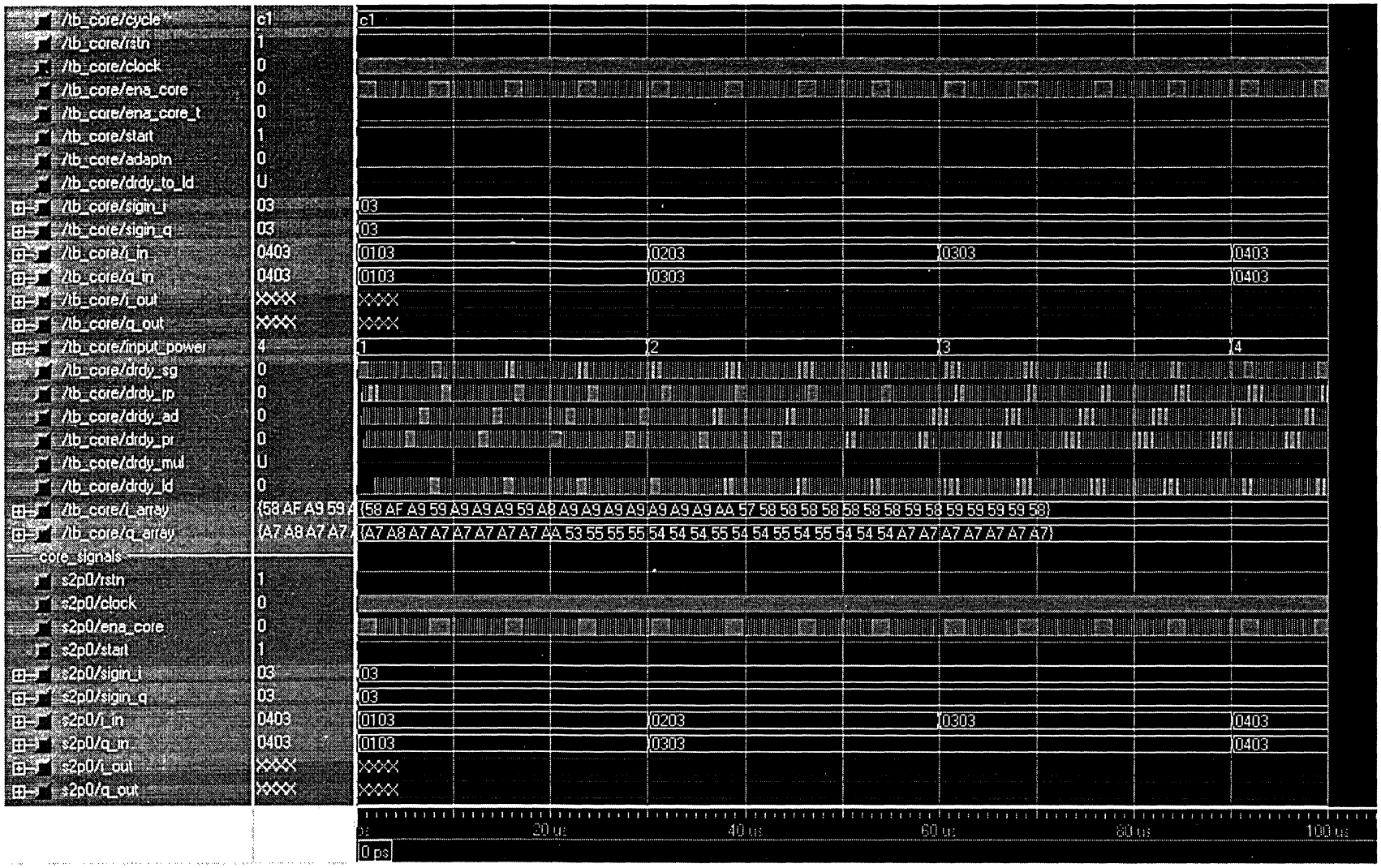


Figure 58 Post routed simulation waveforms

The simulation waveforms shown in this post routed simulation (Figure 58) may not look identical to the RTL simulation in the previous section but it does look identical to Figure 52. The differences between simulations are explained in Section 6.1.4.

6.1.10 Physical layout

The actual final layout of the whole design, including the layout of the routed wires as well as the transistors and the different metal layers, is now complete. This final step utilizes the Cadence DFII software using the output files from the Encounter layout tool. It includes the importing of the necessary files onto DFII, such as the verilog netlist, the lef files to ensure all the proper libraries for the design are inside DFII, and the def files from Encounter. The tool then automatically ensures that it follows all the layout rules depending on the technology to create a working ASIC. Figure 59 shows the full schematic view of the whole design including interconnects with the block diagram and top level connections.

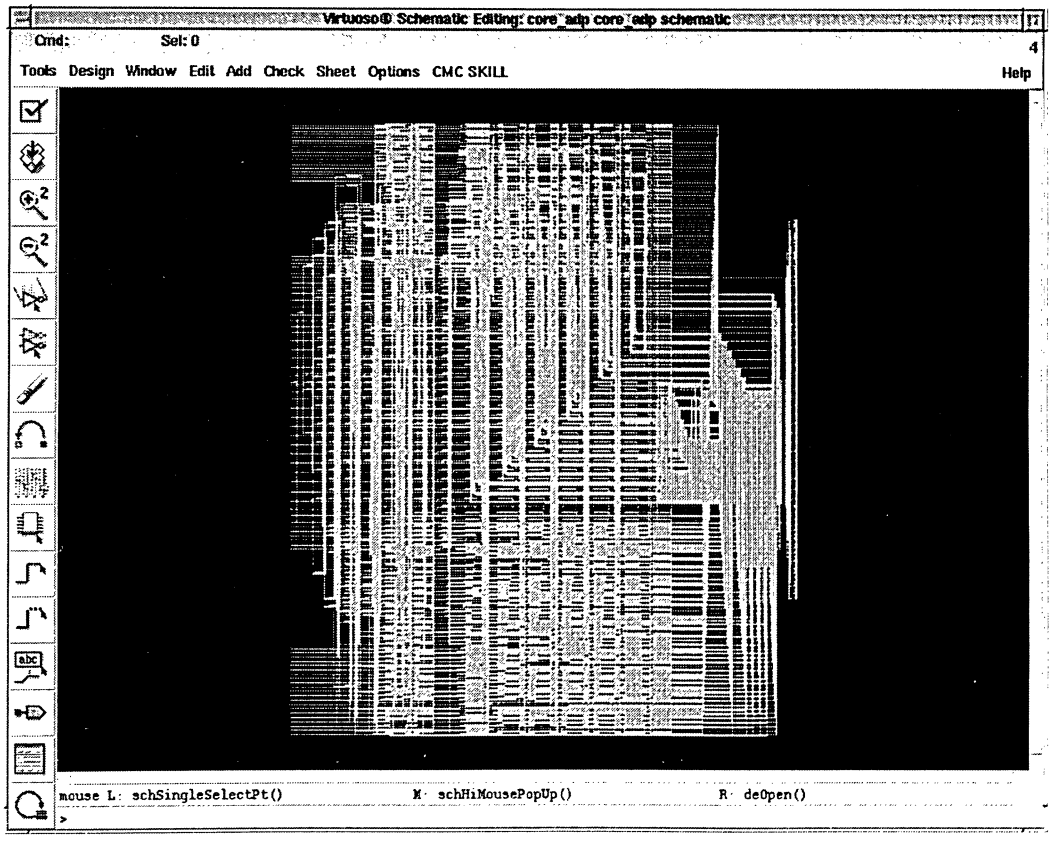


Figure 59 Schematic view of design

Figure 60 shows the autolayout diagram of the design that came from Encounter's output *.def file. This figure includes the routed design, pads, blocks and the power of the whole design without any of the parasitics inherently included in the design.

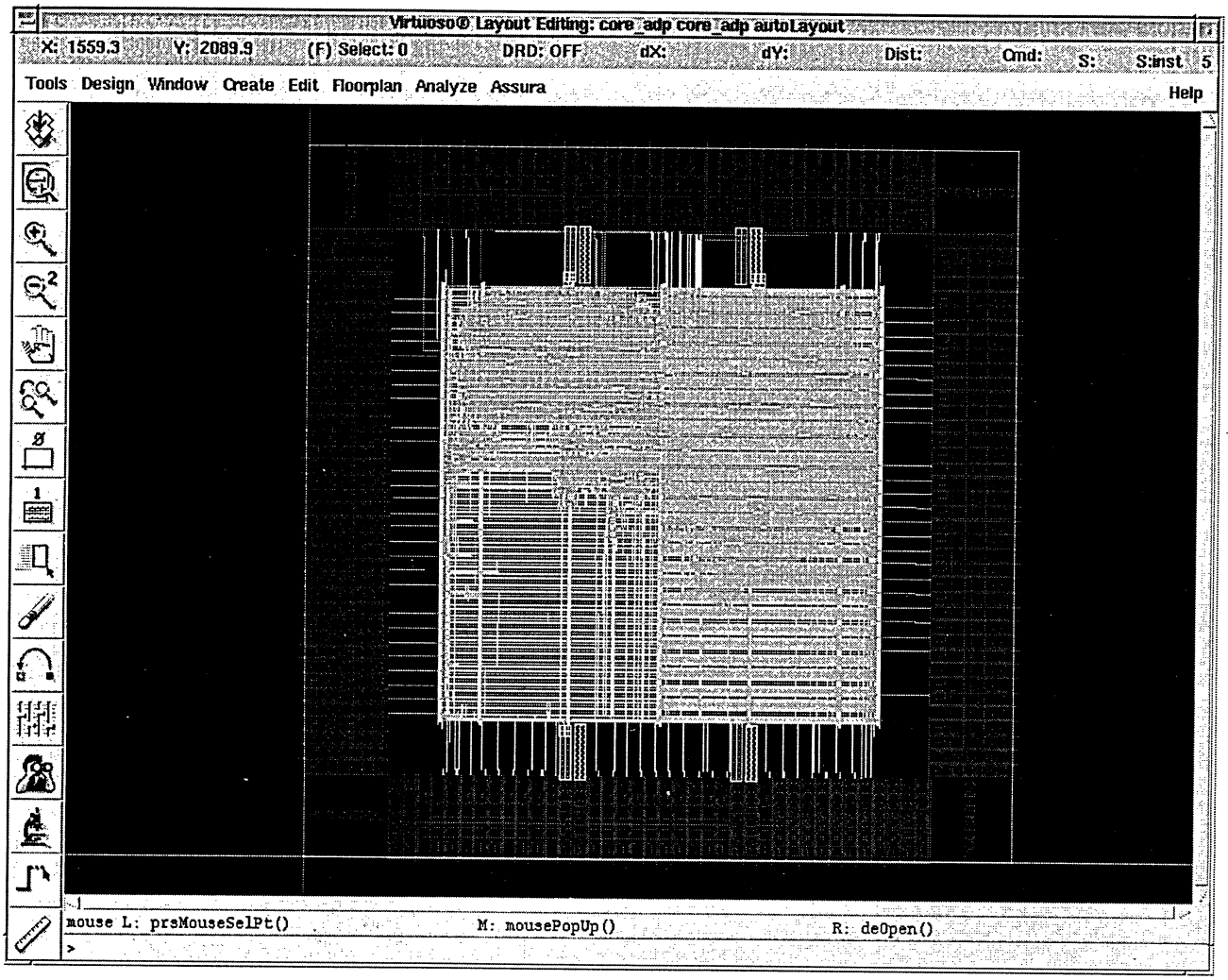


Figure 60 AutoLayout diagram of design

The extracted view which includes the extracted parasitics of the design is presented in Figure 61. Every design, including digital IC designs, contains the desired or undesired parasitic effects of extra added resistance or capacitance. Figure 61 is the closest model to the real ASIC that we can get from the tools and represents the real ASIC fairly accurately.

Chapter 7 Conclusion

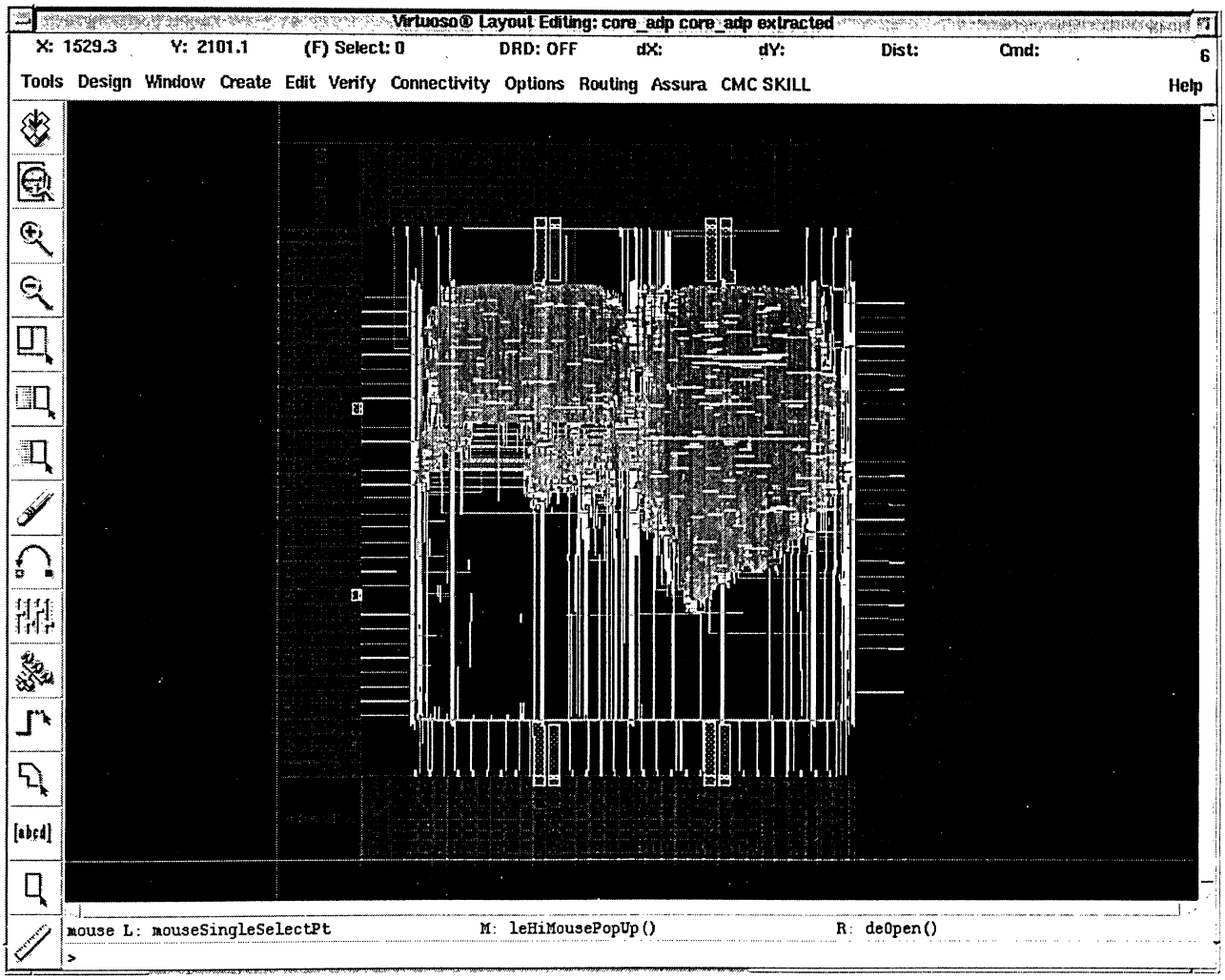


Figure 61 Full view of final extracted design

The ASIC design was completed and fully implemented. The digital IC design process was designed as shown in Figure 26. The RTL synthesis has minimal warnings which can be optimized out. The floorplanning, placement, clock tree generation and routing of the device are automated since all the design constraints where already loaded in from the sdf file generated by the synthesis tool. The actual layout of the device went well without errors as all the libraries where found and components matched to each other. The new tools have a fairly steep learning curve but once the user understands the tool, it can be a very powerful tool to help design complicated IC designs.

Chapter 8 References

[1] Moon, H., Sedaghat, R., "FPGA-based Adaptive Digital Predistortion System for Radio over-Fiber Links", International Journal of Microprocessors and Microsystems, Science Direct, Elsevier, Accepted and in print, Nov. 2005

[2] Wake, D., "A Survey of Current Radio Over Fiber Technologies for Wireless Communications Applications", Microwave Photonics, (http://www.microwavephotonics.com/knowledge/application/pdf/rofapplicationnote.pdf), 2002

[3] Lim, C., Hac, P., Sedaghat, R., "Digital Integrated Circuit Design: Implementation of a Regenerator Section Overhead (RSOH) for an SDH Chip", Thesis Report, Ryerson University Library, 2003

[4] Allan, G., Derome, J.L., Liu, F., Khalili, Al, "Digital IC Design Flow: A Tutorial on RMC's Digital Design Flow (Version 3.1 for Cadence 2003a)", Royal Military College of Canada, RMC Microelectronics Lab, 2003

[5] ARC Electronics, "The Basics of Fiber Optic Cable", (http://www.arcelect.com/fibercable.htm), 2005

[6] Richard, J., "Fundamentals of Fiber Optics: An Introduction for Beginners", (http://www.machinevisiononline.org/public/articles/voip_FiberOpticsRevAa.pdf), 2000

[7] Purdie, I., "Antenna Basics", (http://www.electronics-tutorials.com/antennas/antenna-basics.htm), 2000 - 2002

[8] Carlsson, C., Martinsson, H., Larsson, A., Alping, A., "High performance microwave link using a multimode VCSEL and a high bandwidth multimode fiber", Int. Topical Mtg. on Microwave Photonics, MWP'01, Tech Digest, January 2002, pp.81-84

[9] WebEE, The Electrical Engineering Homepage, "A Primer on Photodiode Technology", (http://www.web-ee.com/primers/files/photodiodes.htm), 2005

[10] The International Engineering Consortium, WebForum Tutorials, "Cellular Communications", (http://www.burnsidetelecom.com/whitepapers/ cell_comm.pdf), 2005

[11] Dawid, H., Meyr, H., "CORDIC Algorithm and Architectures", Synopsys Inc., Aachen University of Technology, Chapter 24, pp.1-11, (http://www.eecs.berkeley.edu/~newton/classes/EE290sp99/lectures/ee290asp996_1/cordic_chap24.pdf) , 2005

[12] Bishop, P., Naish, P., "Designing ASICs", Ellis Horwood Limited, (http://web.ukonline.co.uk /paul.naish/DA/contents.htm), 1998

[13] Canadian Microelectronics Corporation, "Digital IC Design Flow Tutorial (Document ICI-134, Version 1.0)", (http://www.cmc.ca), 2004

[14] Bashker, J., "Verilog HDL Synthesis: A Practical Primer", Star Galaxy Publishing, ISBN 0-9650391-5-3, Chapters 1 and 2, pp. 1 – 107, 1998

[15] John, M., Smith, S., "Application Specific Integrated Circuits", Addison-Wesley, VLSI Systems Series, ISBN: 0-201-50022-1, pp. 930 - 985, 1997

[16] Fernando, X., Sesay, A., "Adaptive Asymmetric Linearization of Radio Over Fiber Links for Wireless Access", IEEE Transactions on Vehicular Technology, vol. 51, no. 6, November 2002, pp. 1576-1586

[17] Banker, J., Shanbhag, A., Sherwani, N., "Physical Design Tradeoffs for ASIC Technologies", Department of Computer Science Western Michigan University, IEEE 1993, 0-7803-1375-5, pp. 70 - 78

[18] Smith, M.J.S., "ASIC Technologies", Compass Design Automation and University of Hawaii, IEEE 1992, 0-7803-0768-2, pp. 97 -106

[19] Nadarajah, S., Fernando, X., Sedaghat, R., "Adaptive Digital Predistortion of Laser Diode Nonlinearity for Wireless Applications", Department of Electrical and Electronic Engineering Ryerson University, CCECE 2003, CCGEI 2003, IEEE 2003, 0-7803-7781-8, pp. 159 - 162

[20] Angarita, F., Perez-Pascual, A., Sansaloni, T., Valls, J., "Efficient FPGA Implementation of CORDIC Algorithm for Circular and Linear Coordinates", Departamento de Ingeniería Electrónica, Universidad Politécnica de Valencia, IEEE 2005, 0-7803-9362-7, pp. 535 - 538

[21] Brasen, D., Saucier, G., "ASIC Prototyping With Reprogrammable Implementations Of Large ASICs", MINC-IST, EUROPOLE, IEEE 1996, 0-8186-7603-5, pp. 127 - 132

[22] Wolf, W., MODERN VLSI DESIGN: SYSTEM-ON-CHIP DESIGN, Department of Electrical Engineering, Princeton University. (http://www.ee.princeton.edu/~wolf/modern-vlsi/Overheads.html), 2003

Chapter 9 Appendix A

9.1 RTL design guidelines

These basic guidelines serve the novice ASIC designer.

9.1.1 Use synchronous designs

Synchronous designs imply that a main clock edge triggers all the different processes and blocks inside the system. This allows for better reliability and simpler design since critical handshaking between asynchronous designs is not a concern. The design is less susceptible to process variations unlike asynchronous designs.

9.1.2 Minimize hazard circuit designs

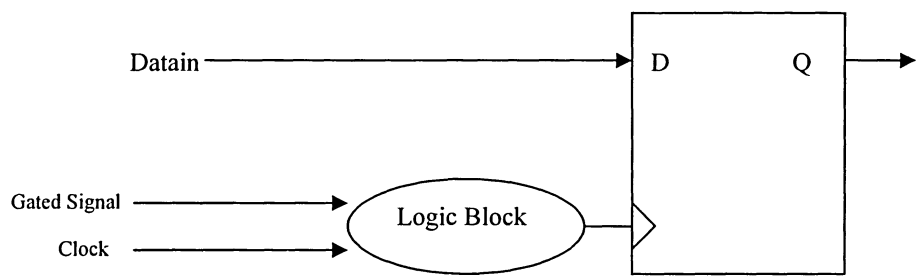
Combinatorial logic is the main cause of glitches in any type of system. Propagation delays between logic blocks i.e. gates should be taken into account and the output should never glitch on any type of input combination when the clock is latching.

9.1.3 Avoid race conditions

Race conditions are situations in which the same signal travels separate paths to the same block or in which two different signals triggered by one signal (i.e. clock) follow different paths to the same block. This could cause one signal to arrive earlier than the other signal causing race conditions and making the design not as reliable.

9.1.4 Minimize gated clocks

A gated clock is a clock signal that is gated (i.e. combined) with a logic signal using any type of combinatorial block before it goes on to the clock pin of the block. Passing the clock through any combinatorial block adds delay and skew to the clock and will cause the system to become unreliable. Gated clocks should be used as seldom as possible.

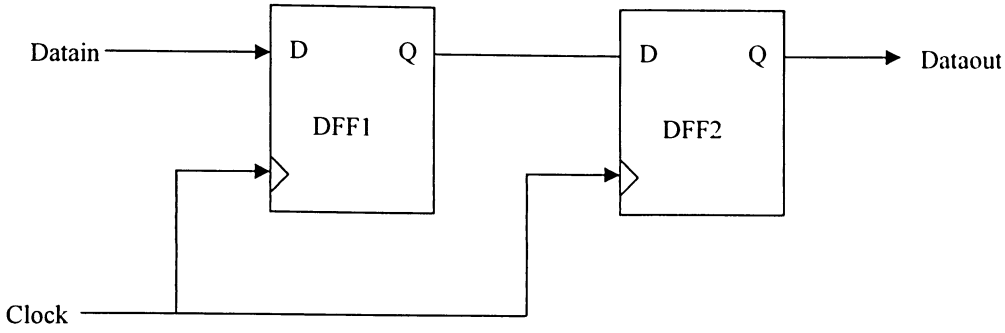


9.1.5 Use synchronous reset/clear

Synchronous resets ensure that the whole system is reset on the clock edge at the same time. Asynchronous reset signals could lead to one section of the circuit resetting earlier than another and could also potentially cause one section of the circuit not to fully come out of reset properly. Synchronized reset signals prevent this from happening as the designer can predefine exactly how many clock cycles the reset state is held for and ensures they all come out of reset state at the same time.

9.1.6 Shoot through consideration

A path between two D-flip flops without any combinatorial logic between them could potentially be hazardous and cause a shoot through condition. This means that the input signal could potentially go through DFF1 and DFF2 in one clock cycle and appear on dataout. Such situations should be avoided.



9.1.7 Do not use half-clock cycles

Some newer designs maximize the use of the clock by clocking on both the rising edge and falling edges of the clock. The creation of designs using both clock edges should be avoided, as clocking on both edges requires the critical path of the circuit to be less than half a clock cycle width. Due to process variations, it can not be guaranteed that the duty cycle of the clock is even.

9.1.8 Minimize gated preset/clear signals

Gated preset and clear signals should be avoided as they could cause improper timing, similar to the gated clock design issues.

9.2 Designing for area or performance?

The ASIC designer should understand the differences between designing an ASIC for area or designing it for speed. The design goal of the system dictates exactly how the design should be created. Designing for speed would require a larger area while designing for area would lower the speed of operation.

Some guidelines to optimize designs for performance:

9.2.1 Avoid extra delays

By ensuring that floorplanning is done properly and optimized, extra interconnection delays can be prevented. To ensure proper optimization, partitioning of the blocks into sub-blocks must be done carefully.

9.2.2 NAND logic versus NOR logic

Nand logic gates are faster than NOR logic gates. This is a direct result of PMOS transistors having longer delay times than NMOS transistors including the design of a standard NAND gate and NOR gate.

9.2.3 Avoid large buffers

Large buffers generally have a large capacitance associated with them. Unnecessary use of such buffers will add extra parasitic capacitance and cause more loading than required on the circuit. This will slow down the switching capabilities and overall system clock speed of the circuit.

9.2.4 Divide the chip into hierarchical blocks

By dividing the chip into blocks the signals associated with each block are within close proximity of the other signals and blocks with which they must interact. As such, better optimization and less delay can be achieved by the tools as more efficient routing is possible.

9.2.5 Minimize critical path

Minimizing the critical path requires that the logic depth between flip flops be controlled as the critical path defines the maximum speed of the whole system.

Some guidelines to optimize designs for area:

9.2.6 Use memory cells

For the most part, designs made with memory cells are smaller than those made with standard cells. This is dependent on design and architecture.

9.2.7 Minimize input connections to VDD/VSS

Cell input to VDD or VSS should be avoided in preference to a cell with the correct functions.

9.2.8 Large blocks vs. multiple small ones

Blocks and peripherals require routing, and as such, using a single large block as opposed to multiple smaller blocks can save on routing area usage and result in a smaller die.