Cairo University

VISIBLE LIGHT COMMUNICATION SYSTEMS OVER FPGA

A Major Qualifying Project Report:

SUBMITTED TO THE DEPARTMENT OF ELECTRONICS

AND ELECTRICAL COMMUNICATIONS

ENGINEERING

OF CAIRO UNIVERSITY

IN PARIAL FULFILLMENT OF THE REQUIREMENTS FOR

THE DEGREE OF BACHELOR

By

Mohamed Amir Aly

Mohamed Ibrahim El-Desouky

Moaaz Roushdy

Amr Hamdy

Mosaab Mahmoud

Supervised By:

Dr. Hassan Mostafa

Dr. Tawfik Ismail

July, 2015

Table of Contents

Table of Figures

Abstract

Nowadays, if we have a look on the radio frequency spectrum we will find that it is becoming more crowded and the traffic demand is rising exponentially, so people must start to think about having an alternative means to wireless communication which is necessary to accommodate this rising demand. Visible light communication systems provide an alternative to the current standards of wireless transfer of information by using light from light-emitting diodes (**LEDs**) as the communication medium. In these systems, LEDs blink at a high rate such that the human eye cannot notice the change in light intensity, but a sensitive photodiode will be used to detect the on-off behavior and the FPGA starts to decode the information embedded within it.

Firstly, there are various issues and problems with current wireless communication systems that are going to be analyzed in this project. This project discusses how these issues could be resolved by the visible light communication. Then, some simulations and calculations for the power distributions and signal to interference ratio. After that the hardware design and implementation processes of the visible light communication system are described in detail, including a value analysis of the parts, components and the building blocks used to build the prototype, as well as the necessary steps to wire and/or code each functional block of the design. The attained results of the system, including transmission distance and speed, as well as quality of transmission and type of data are discussed. Finally, all processing made on data will be discussed in deep details.

Acknowledgements

All thanks to God, the **Almighty**, for helping us by giving us the patience and courage to introduce this work.

With deep appreciation we would like to thank *Dr. Hassan Mostafa* and *Dr. Tawfik Ismail* for their aid and support all over the year and for all of the help they have given us throughout the course of this project. They gave us constant feedback on our work, as well as providing helpful suggestions to fix issues that arise.

1 Chapter 1

1.1 Introduction

Over the last century communications using the radio frequency have been the preferred way to transmit data wirelessly. Although, wireless optical communications has been used long before radio communications was first considered. Nowadays, there is a huge capacity shortage for wireless data communications and this is a serious problem that we are facing these days, so free-space optical communication is being considered as a reliable candidate for the widespread wireless communications applications. Although, the data demand is increasing exponentially, there is a limited available radio spectrum. To accommodate this needs of wire-free communication systems, it is necessary to think about some alternatives to overcome the increasing demand. With the widespread use of the light emitting diodes (**LED)** light bulbs, visible light communications (**VLC**) has become the forerunner in the current optical wireless communications field.

This chapter will illustrate the problems of current wireless communication systems and alternatives to these systems, as well as motivations and possible applications for visible light communications.

1.2 Historical background

The use of light to send messages is a very old idea. Fire and smoke signaling were used in ancient civilizations. For example, the ancient Greeks used polished shields to reflect sunlight to signal in the battle and Roman records indicate that polished metal plates were used as mirrors to reflect sunlight for long distance signaling. The Chinese started using fire beacons followed by the Romans and American Indians using smoke signals [1].

In the early 1800s, the US military used a wireless solar telegraph called “Heliograph” that signals using Morse code flashes of sunlight reflected by a mirror. The flashes are produced

by momentarily pivoting the mirror, or by interrupting the beam with a shutter. The navy often uses blinking lights, i.e. Aldis lamps, to send messages also using Morse code from one ship to another. In 1880, the first example of VLC technology was demonstrated by Alexander Graham Bell with his "photophone" that used sunlight reflected off a vibrating mirror and a selenium photo cell to send voice on a light beam [2].

The Visible Light Communications Consortium (**VLCC**) is established in 2003 to develop, plan, research and standardize Japan's own visible light communication systems.

In 2009, IEEE 802.15 TG7 Task Group seven was chartered to write standards for free-space optical communication using visible light.

1.3 Motivation

To meet the high demand on wireless communication and radio frequencies, wireless technology needs to expand and to be improved, as societal dependence upon wireless systems continues rising. Phones, laptops, and global positioning systems are all devices that implement certain forms of wireless communication to be able to send information to another location or exchange some information with each other. However, the availability of current forms of wireless is very limited, and it is not necessarily safe to implement wireless radio, making it necessary to explore other alternatives to wireless communication to allow continued expansion upon communication systems and ensure safe use.

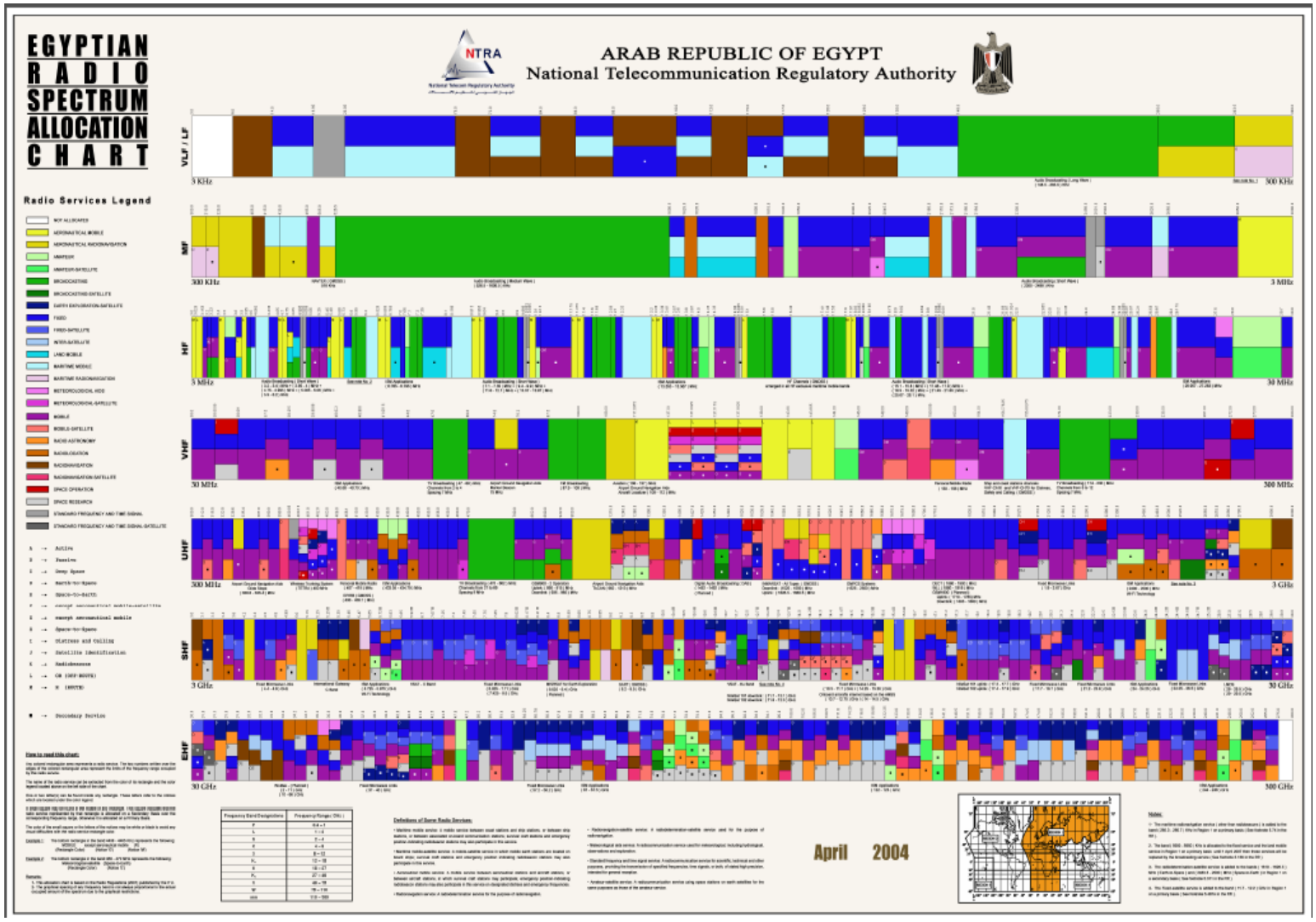


Figure 1-illustrates the frequency allocations of the radio spectrum in Egypt

Source:
http://www.tra.gov.eg/english/DPages_DPagesDetails.asp?ID=138&menu=1[3]

As we can see the figure above shows the frequency allocations of the radio spectrum in the Arab Republic of Egypt. The National Telecommunication Regulatory Authority (**NTRA**) regulates many wireless applications in the US, including radio, television, wire, satellite, and cable. From **Fig.1 & Fig.2**,we could notice the frequency band is distributed among various applications such that each application is given a certain band so that it can operate within this frequency band allow efficient use of the available frequency spectrum. As we can see it is quite evident that this spectrum is very crowded. At the same time, there is a huge growth in demand in the limited radio frequency spectrum.

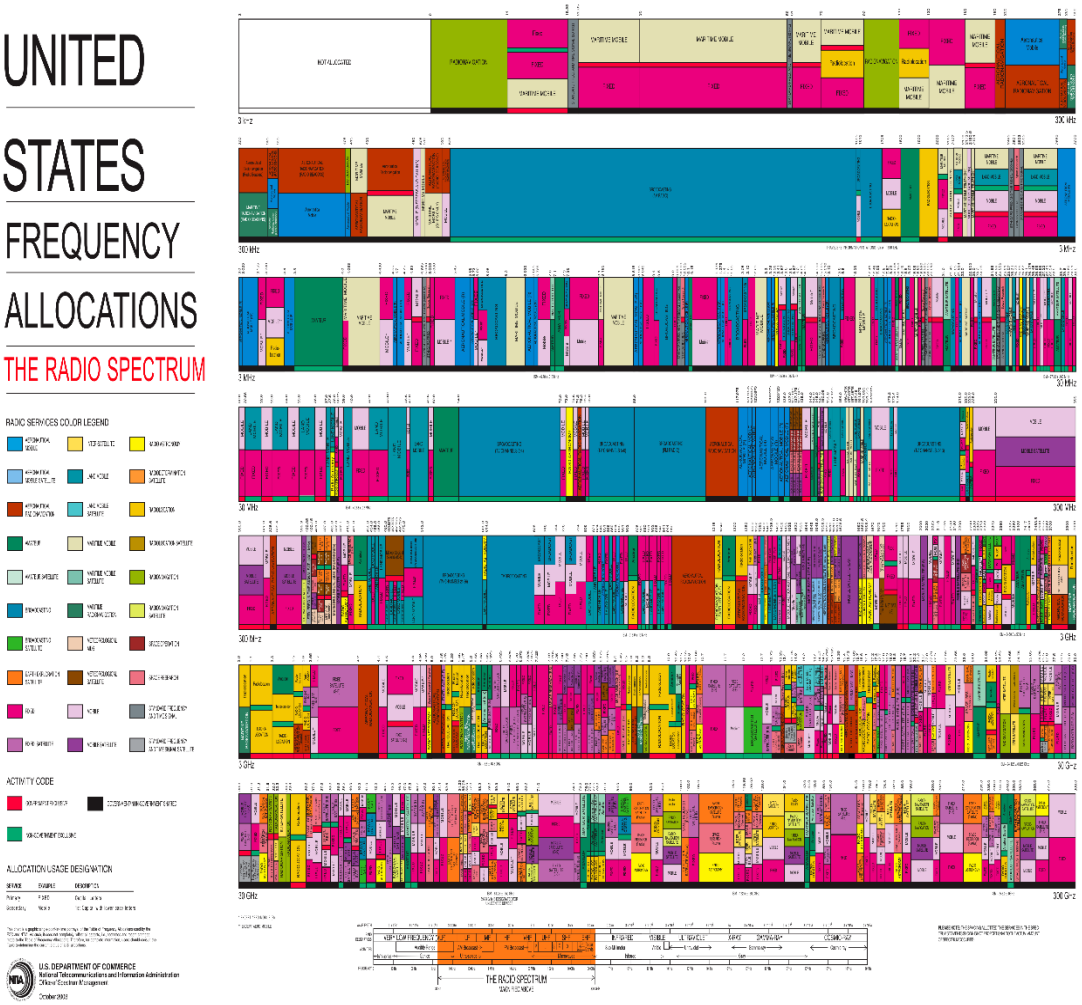


Figure 2- illustrates the frequency allocations of the USA

Source: https://en.wikipedia.org/wiki/Frequency_allocation[4]

Let us have a look on the frequency allocations the Federal Communications Commission (**FCC**) regulates many wireless applications in the US, FCC recently published a paper about the traffic demand of frequency and its rapid growth, this paper states that By 2014 there will be a significant deficit of available spectrum to cope with rapidly growing wireless data traffic. Even if we assume flat voice traffic, with 3500% growth in data traffic between 2009 and 2014(see **Fig.3** & **Fig.4**), there is an estimation that there will be a spectrum gap of 275 MHz.

Cost of delivering data traffic in the absence of additional spectrum is building new cell-sites, over 200,000 of them, each at an estimate cost of $550K including the operational cost. Therefore, a more efficient way of utilizing radio frequency is necessary.

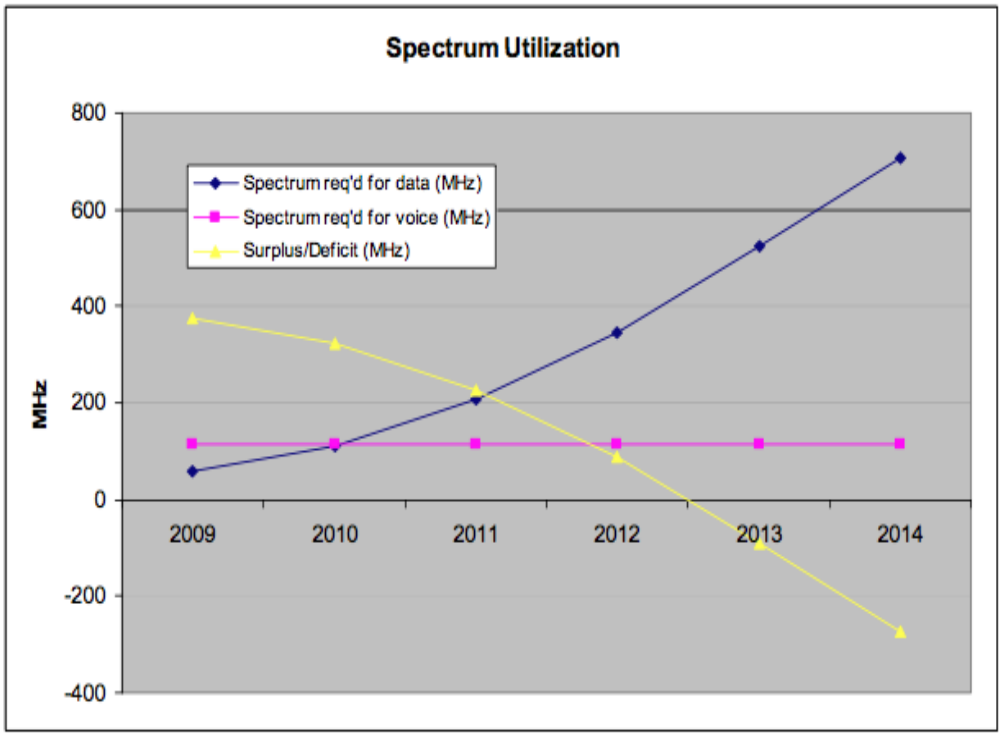


Figure 3- shows the spectrum requirements and deficit

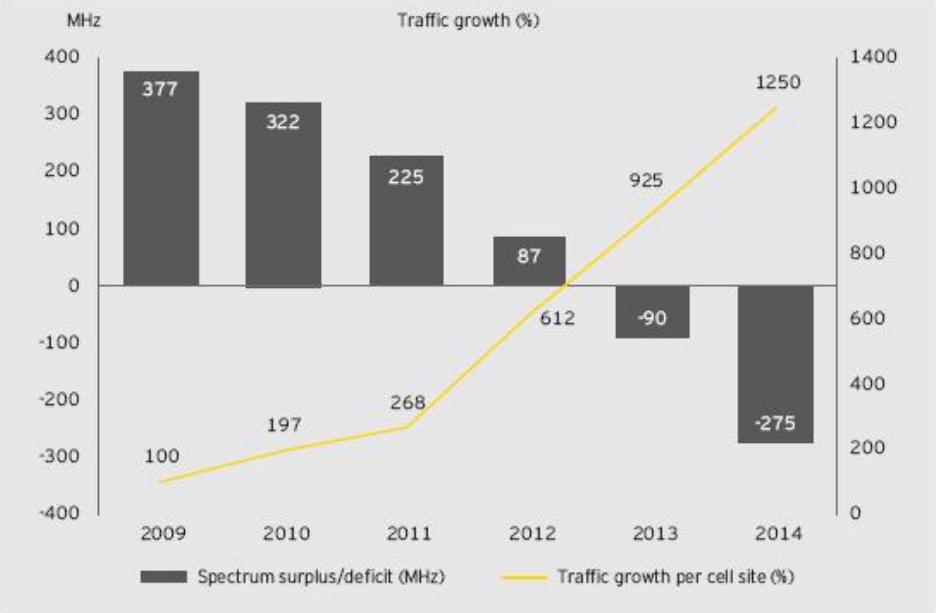


Figure 4- shows the growth of traffic demand

Interference is also considered a concern for many existing wireless systems, in addition to the crowding of the frequency spectrum. Any simultaneous use of a frequency band will cause interference due to the electromagnetic nature of most wireless devices, which could result in incorrect or loss of information for those users involved. A prime example of this is the use of mobile devices on planes, which directly affects safety.

The Federal Aviation Administration (FAA) argues that these wireless devices cause interference to the aircraft's navigation and communication systems, and the Federal Communications Commission (FCC) argues that mobile devices used on aircrafts will disrupt cellular towers on the ground. Other studies indicate that use of mobile electronics on aircrafts can exceed permissible emission levels for safety with regard to some avionics [5]. Regardless of the reason, it is clear that it is not feasible to use wireless devices in certain environments in which safety, data integrity, and accuracy are highly important.

VLC systems have more flexible than other communication systems in many regards. Since the medium for transmission or the communication channel in VLC systems is visible light and not RF waves that can penetrate walls, that is considered a high level of security because light cannot leave the room, containing data and information in one location. There

is no way to retrieve and access the information unless a user is in a direct path of the light being used to transmit the data. In addition, the abundance of the raw materials like LEDs which are highly efficient and becoming more durable, adding to the integrity of these systems. High power white LEDs are expected to replace the existing lighting technologies in near future which are also suggested for visible light communication (**VLC**).

1.4 Alternatives in progress

Many engineers spend their time and effort trying to determine solutions for the crowded frequency spectrum. Given that this is a major issue in wireless communication.

Currently, there are some alternatives to the radio frequency communications exist. For example, there are *Laser communications* and the *Cognitive radio*.

1.4.1 Cognitive Radio:

As we have mentioned before that the problem that we are facing in wireless communications is the crowded frequency spectrum. As the current trend continues, devices that normally would not be able to wirelessly communicate, such as lamps or temperature sensors, will be connected to some type of wireless network. This will increase the number of end users and further increase the traffic of the frequency spectrum. Next Generation (**xG**) communication networks, also known as Dynamic Spectrum Access Networks (**DSANs**) as well as cognitive radio networks, will provide high bandwidth to mobile users via heterogeneous wireless architectures and dynamic spectrum access techniques. The inefficient usage of the existing spectrum can be improved through opportunistic access to the licensed bands without interfering with the existing users. **xG** networks, however, impose several research challenges due to the broad range of available spectrum as well as diverse Quality-of-Service (**QoS**) requirements of applications. These heterogeneities must be captured and handled dynamically as mobile terminals roam between wireless architectures and along the available spectrum pool. The key enabling technology of **xG** networks is the cognitive radio.

Software defined radios are evolving as platforms for the communications systems. A ‘‘Cognitive Radio’’ is a radio that can change its transmitter parameters based on interaction with the environment in which it operates [6]. The cognitive radio, based on a software-defined radio, is defined as an intelligent wireless communication system that is conscious of its environment and uses the methodology of understanding by building to learn from the environment and adapt to statistical deviations in the input stimuli, with two primary objectives in mind which are highly reliable communication whenever and wherever needed and efficient operation of the radio spectrum. So the difference between the cognitive radio and a typical radio system is that a cognitive radio is programmed to adapt to its surroundings. Software radio provides an ideal platform for the realization of cognitive radio. Dynamic spectrum access techniques allow the cognitive radio to operate in the best available channel. More specifically, the cognitive radio technology will enable the users to [7] determine which portions of the spectrum is available and detect the presence of licensed users when a user operates in a licensed band *spectrum sensing*- which is Detecting unused spectrum and sharing the spectrum without harmful interference with other users-, select the best available channel *spectrum management*- which is Capturing the best available spectrum to meet user communication requirements-, coordinate access to this channel with other users *spectrum sharing*-which is Providing the fair spectrum scheduling method among coexisting **xG** users., and vacate the channel when a licensed user is detected *spectrum mobility*-which is Maintaining seamless communication requirements during the transition to better spectrum.

1.4.2 Laser communications:

Another form that utilizes the wireless connections through the atmosphere is the laser communication systems, they work similar to fiber optic cable systems except the beam carrying transmitted data is through free space by shooting a laser. This form of wireless communication can be effective because it is not regulated by the government as it operates in a near infrared spectrum, hence avoiding any additional overcrowding of the spectrum with this form of communication. This allows for quick establishment of communication links, as it does not need to go through the various regulatory processes that would be necessary to set up an RF system. The carrier used for the transmission of this signal is generated by either a high power LED or a laser diode. The laser systems

operate in the near infrared region of the spectrum. The laser light across the link is at a wavelength of between 780 – 920 nm. Two parallel beams are used, one for transmission and one for reception (**Fig.7**). The system can work for a distance of up to 6 km with bitrates up to 1.25 Gbps [8]. The system also uses relatively low power and has a low noise ratio. It is also secure, as any sort of eavesdropping on the data transmission will require viewing directly into the transmitter path, causing an interruption in transmission.

Figure 5-MAGNUM 45 High-Speed Laser-Communication System

Unfortunately, the system requires a line-of-sight (**LOS**) path from the transmitter to receiver. This renders the two functional blocks relatively immobile. If the path is not calibrated precisely and accurately, the laser could miss the receiver by a large distance, resulting in no data transmission. Additionally, since no radio interference studies are necessary, the systems are quickly deployable. The narrow laser beam-width prevents interference with other communications systems. In addition, although invisible to the unaided eye, the lasers used could result in damage to one's eye if there is an extended exposure to the laser. [9][10][11]

There are some further remarks, as an example if heavy fog, snow or smoke blocks the **LOS** between the units or the sun light is interfering the laser beam. Unfortunately, there are not many countermeasures to improve the situation in such cases. Also the link performance is sensitive to vibration, wind sway, and thermal expansion of the equipment.

2 Chapter 2

2.1 Visible light communications

The focus of this project will be on Visible Light Communications (**VLC**). We aim to understand the field of **VLC** very well to be able to investigate this system by designing our own analog circuit which is considered the hardware that will be connected to the FPGA which is responsible for the processing done on the data upon sending or receiving to integrate with a computer, and then sending some form of data using visible light LEDs from a transmitter, and decoding it with the FPGA connected to the receiver and the second computer.

Suppose that we have two computers "**A**" and "**B**"; where Computer A is the source of the information that will be transmitted and computer B is the receiver one.

Information will be converted into bits or packets through some software programs, then data will be sent or transmitted from computer A to the first FPGA using Ethernet or USB-to-UART serial port. Now comes the role of the transmitter circuit which is connected to that FPGA, after that data will be transmitted with blinking LEDs that are allied to this circuit, the blinking LEDs will not be visible to the human naked eye as they are blinking at a high frequency.

On the receiver side, photodiodes will be used to detect the fluctuations and blinking of the LEDs in the transmitter, after that these detected variations will be sent to the second FPGA to be decoded as some processing could be done on it to determine the originally transmitted information message, finally this data will be sent to computer B.

The whole system will be powered by computer/FPGA combination, also Batteries might be used in some cases.

2.1.1 The use of LEDs

With the exponentially increasing abundance of the LED in the last period of time. LEDs are getting more and more involved in many technologies in our life (see **Fig.6**). As

per LED Inside, the LED lighting segment is estimated to increase from around $1.5 billion in 2012 to approximately $8 billion by 2015, a CAGR of over 70%.[12]

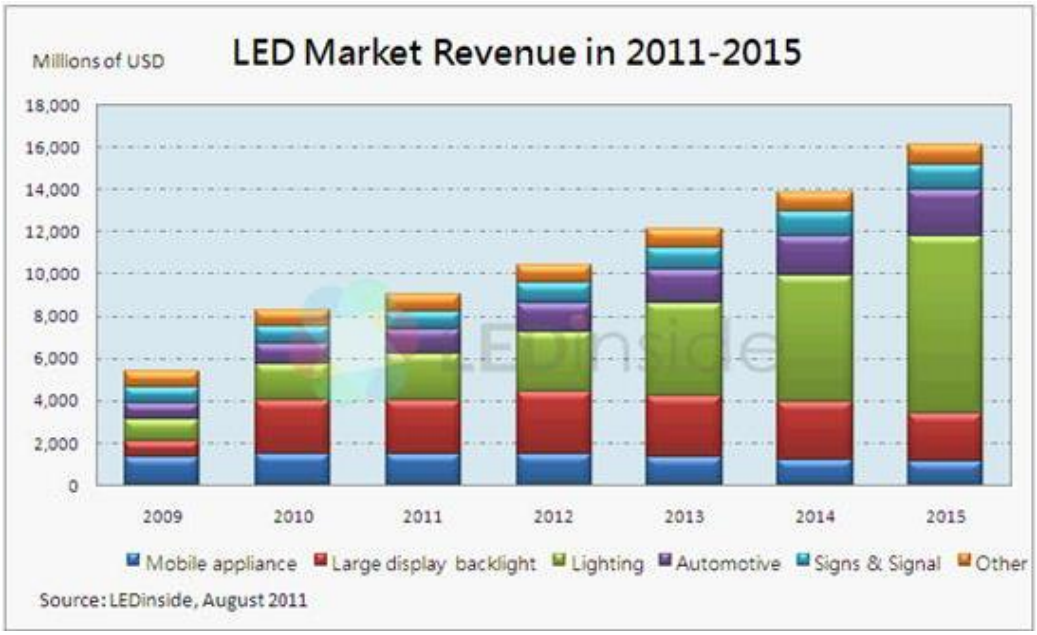


Figure 6-LED capacity growth in market

LEDs are considered a strong candidate for future illumination devices as they are more efficient and have longer lifetime such that LEDs could last for 100,000 hours. Moreover, LEDs are mercury free, a hazardous and poisoning material that is hard to be recycled in the current illumination industries which is an added benefit to the LEDs. Comparing the LEDs with the fluorescent lamps and incandescent bulbs, the LEDs have many advantages over the other two types. Table 1 shows a comparison between LEDs, incandescent and fluorescent lamps.

	LEDs	Incandescent bulbs	Fluorescent lights
Electricity used	6~8 Watts	60 Watts	13~15 Watts
Contain mercury	No	No	Yes
Turns on instantly	Yes	Yes	No
On/Off cycling effect	None	Some	Yes
Failure modes	Not typical	Some	Yes
Heat emitted	3.4 btu/hr.	85 btu/hr.	30 btu/hr.

Sensitive to humidity	None	Some	Yes
Fragility	Durable	Not durable	Not durable

Table 1: comparison between three types of illumination sources.

Solid state lighting based on LED technology enables an information signal to be superimposed onto an LED driver signal in order to carry data. This is a bit like the method used in radio broadcasts where an audio signal is superimposed onto the radio frequency carrier in order to transmit the information. In the case of the LED it is the wavelength of the light emitted that automatically provides the signal carrier and the intensity of the light is varied at high speed (modulated) to superimpose the data. At the receiver a photo-detector is used to remove the data from the optical signal.

Visible Light Communication (**VLC**) with Light Emitting Diodes (**LEDs**) as transmitters and receivers provide a novel approach to enable low bitrate wireless ad hoc networking for short distances. This case is for LED-to-LED communication which can be used over a free space optical link or channel for a short distance, this full-duplex way of communication is done to have less complexity in the design of the system, but it introduces some interference problems and cause the system to be more slower as this system needs some synchronization (see **Fig.7**). [13] However, the half-duplex way of

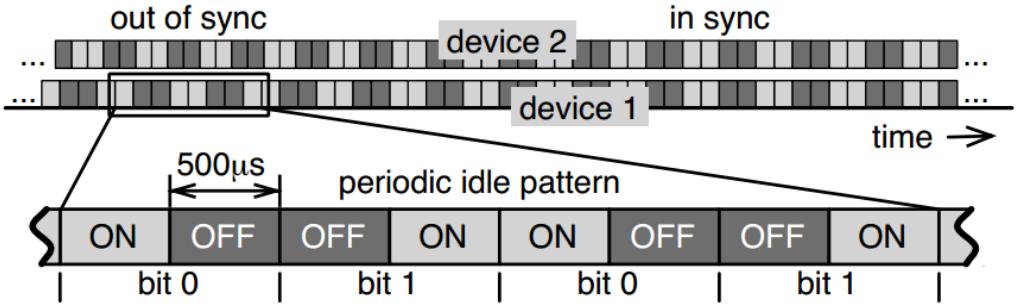


Figure 7-shows synchronized and none synchronized bits

communication is used to have high data rates and longer distances that could reach several meters, using the LEDs to transmit data and the photodiodes as a receivers which will detect the rapid changes in the transmitting LEDs this technique of modulation is called ON-OFF-Keying (**OOK**), when the LED is off; no light is emitted from it, this means 0'b (digital 0). When the LED is ON; light is emitted from it, this means 1'b (digital 1), there will be no need for any synchronization as the photodetector will understand the incoming signal rather it is above a certain threshold (digital 1) or below it (digital 0) this data will

be modulated by the (OOK) technique. Moreover, we could obtain different channels or mediums by making a good use of the LEDs, this is made by using the Red-Green-Blue (**RGB**) LEDs that will provide more security and more immunity for the transmission of the information through the channel of the system.

2.1.2 Advantages

Increasingly, RF wireless networks are compromised by the fact that in many buildings the three independent WLAN frequency bands are multiply occupied, this occupation leads to collisions, interference and miss match among the data packets. Researches and studies are made, these studies showed that the speed of the Internet connection can be reduced within the home by more than 30% when using a wireless broadband router due to interference. The manufactures advice to overcome this phenomena is to reduce the number of other wireless devices in the home, such as cordless phones, TV remotes and baby monitors that also cause some interference. In a situation like this, visible light, as a license-free medium and an unused spectrum band, offers a suitable alternative.

As visible light has a few advantages over other standard wireless transmissions so it should be considered as the medium for wireless transmission. The visual light's frequency spectrum bandwidth ranges from 430 THz to 750 THz [14], this is the first reason to consider in visible light as this bandwidth is much larger than that of the radio frequency, which ranges from 3 KHz to 300 GHz [15]. With this larger bandwidth and by setting some regulations on it, it is possible to potentially achieve higher transfer rates of data and accommodate more users because each user could be given a larger portion of bandwidth to use for the information transmission with very low levels of interference.

If the communication system will be used in hospitals, the transmissions will not occur in the Industrial, Scientific, and Medical (**ISM**) band which are used in other than telecommunication applications and machines, these devices have very powerful emissions that can cause disturbance to radio communication at the same frequency due to the creation of electromagnetic interference, so these devices were limited to certain bands of frequencies, therefore not interfering with medical devices must be considered. On top of having a higher bandwidth, the frequency spectrum has less regulation than the radio

spectrum. With little regulation, the user will be able to choose any frequency to transfer information. If visible light communication systems become more popular, regulations could be placed on these forms of data transmission for the same reasons that they were placed for the radio spectrum.

Visual Light Communication offers many benefits over RF wireless communications. One of these benefits is that it has an unlicensed spectrum, six orders of magnitude larger than RF, also it is free and unregulated wavelengths short wavelength. Moreover; visible light is harmless to humans and appealingly pleasing, little harm as visible and eye safe, little electromagnetic interference (**EMI**) compared to RF equipment, high gain antenna and high quality links.

One of the biggest advantage supported by the visible light systems over the other communication systems is its abundance, light sources is already installed everywhere, this infrastructure can support two different functionality lighting places in addition to data transition, companies, stores, hyper markets, schools, restaurants, etc. will have lights on for at least the duration of hours of operation, of which could be used for visible light communications.

Visible light in some standard situations also has few drawbacks that could be potentially considered or used as advantages for the VLC system. Light cannot propagate through walls on the contrary to the radio waves. Since light cannot propagate out of an enclosed room, then the only way to access the transmitted information is if the receiver is in the same room; thus, any source that is placed outside this room which the transmitter is placed in will not be able to acquire the information. Additionally, more security levels could be reached when more than one receiver exists in the same room by making use of the **RGB** light colors and some optical filters to differentiate between these different wavelengths, as we could use the Red light for the downlink or to communicate with one of the receivers in the room and the Green light could be used for the uplink or to communicate with another receiver in the same room. This could be done without the knowledge of any of the receivers in this room that there is an optical link transmitting information between the communicating transmitter and the receiver that is receiving the

data upon a specific light color. Therefore, light sources are more secure than radio waves because they are not broadcasted for external sources to receive.

Table 2 shows how RF wireless networks compare to LED lighting with optical communications. A key aspects of the LED method is the bandwidth speed and extra level of security, making it a much more secure means of transferring information between devices.

Attribute	RF @ 2.4 GHz	LED optical	Advantage
Security/Privacy	Penetrates walls.	Does not penetrate walls and prevent snooping.	LED optical.
Available bandwidth capacity	Signals at same frequency can interfere with another and thus be limited by contention; signals degrade from peak.	Light can be directed (smart light sources) and can be tuned to adapt to different environments and narrow footprints.	LED optical.
Cost of additional bandwidth spectrum	Very high when available.	None, **yet**!	LED optical.
Interference	Multiple users on same frequency slow transmission speed; ISM sources.	Visible natural (solar) and manmade light (non-LED lamps) slow transmission speed.	Varies.
Multipath fading	Destructive interference: RF wave bounce off surfaces and can be out of phase.	Interference appears as noise; no signal cancelling.	LED optical.
Path redundancy	Achieved with multiple access points.	Achieved with multiple LEDs.	LED optical.
Transmission speed	100 Mbits/second deployed.	Comparable; but with reuse of volume for higher aggregate speed.	LED optical.
Cost	High.	Low.	LED optical.

Table 2: Comparisons between VLC and RF wireless communication methods. (Source: Boston University)

Visible light was chosen for a variety of reasons, but primarily because it will not add to the cluttering of the radio frequency spectrum, which is heavily regulated by the FCC, and also because it will avoid the issue of interference in sensitive settings such as hospitals and airplanes. Fig.8 shows the wavelength range of visible light.

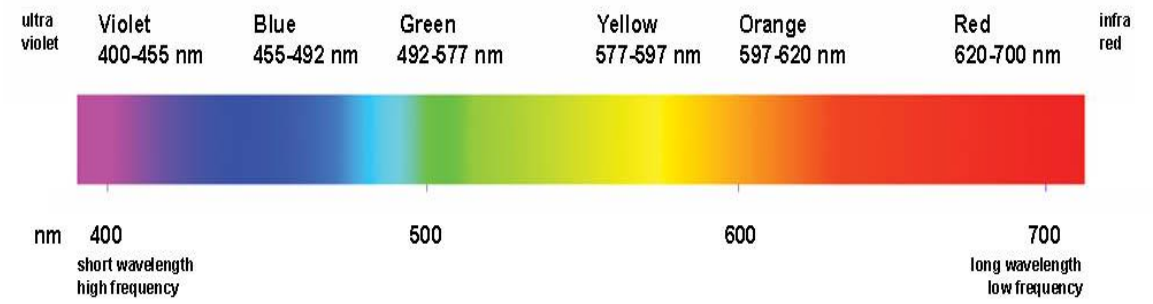


Figure 8- Visible Light Spectrum

Source: **http://nextgenlite.com/images/VisibleLightSpectrumGradientForWeb.jpg**

From these wavelengths, the frequency range can be calculated by the following equation:

$$f = \frac{c}{\lambda}$$

Where f is the frequency, c is the speed of light, and λ is the wavelength. Thus, it can be shown that the range of frequencies for visible light is around 400-800 THz.

2.1.3 Disadvantages

Limitations and drawbacks that we have to consider include noise from ambient light and the line-of-sight of the system. If the intensity of ambient light is greater than that of the light from our system, the signal-to-noise ratio (**SNR**) is low, which will distort transmitted data. To compensate for this, the SNR will be maximized by setting thresholds on the FPGA based on voltage signals produced by the ambient light in conjunction with the transmitter signal. Also, the system will only be maximized when the LEDs are directly facing the sensor. If the angle is changed even slightly, the maximum range of the system will decrease significantly. The easiest solution is to ensure that the transmitter and receiver are facing directly at each other.

2.2 Modulation techniques

Sending data over the visible light spectrum could be done by using many different methods and techniques.

On-off keying (**OOK**): As the name suggests the data is conveyed by turning the LED off and on. In its simplest form a digital '1' is represented by the light 'on' state and a digital '0' is represented by the light 'off' state. The beauty of this method is that it is really simple to generate and decode. However, this method is not optimal in terms of illumination control and data throughput.

Another techniques that could be used such as; pulse width modulation (**PWM**) and pulse position modulation (**PPM**), these methods conveys information encoded into the duration of pulse or using the position of the frame in which more than one bit can be transmitted in each pulse, but it requires more duration or a longer frame rather than the OOK single bit. For the pulse amplitude modulation (**PAM**) which can carry more data in each pulse than OOK, this modulation technique is more complex and more susceptible to noise on the optical link between the transmitter and the receiver.

Color shift keying (**CSK**) could be used when RGB type LEDs are used for the illumination of the system, where different colors of light are combined so the output data will be carried by the color itself. Hence, the intensity of output will be constant. The disadvantage of this system is the complexity of both the transmitter and receiver.

The visible light communication system could reach very high data rates, the implementation of the PHY II 96 Mbit/s standard would require an LED with 120 MHz bandwidth, which is currently unrealistic. Siemens have achieved 500Mbit/s using a specialist RCLEDs in laboratory conditions. The University of Edinburgh D-Light project uses OFDM with a standard single OSRAM Ostar phosphor coated LED with 17 MHz bandwidth and claim they can achieve 100 Mbit/s.

2.3 Potential applications of visible light communications

Visual light provides several opportunities to apply visible light communications as light in the visible spectrum is used everywhere. Many applications could be done in this field that will be certainly useful. VLC could be applied in many different fields that results in several applications including medical and health applications, which could utilize the systems for more secure transfer of data; Also, there are many outdoor applications in which VLC could be useful such as traffic lights, transportation and vehicle to vehicle communication which could utilize systems to optimize traffic flow; then we have the indoor applications like the indoor positioning systems, television sets which could supply a user with information on current show listings.

2.3.1 Hospitals and Healthcare

There are a lot of risks that result from the use of mobile phones and Wi-Fi and in certain parts of the hospitals, especially about scanners around MRI scanners and in operating theatres, so medical equipment requires isolation from Electromagnetic Interference (EMI) or Radio Frequency Interference (RFI). Hospitals need wireless technology to updated information maintaining patient records, collecting data as a real-time handheld patient monitor to detect changes in a patient's condition, or even observing medical images via ultrasound. However, many concerns follow with the use of wireless technology in hospitals, and must be addressed when implementing a wireless communication system in such a sensitive environment. Operational efficiency is necessary to ensure reliability and short delay time between two communicating devices. Moreover, data accuracy and security must be concerned in such applications. Many medical devices are sensitive to waveform distortion, and any sort of electromagnetic interference between a wireless communication device and a medical instrument could cause an unexpected automatic shutdown or restart of the instrument; so interference is perhaps the most significant concern in the hospital environment.

VLC is considered a safe and secure alternative in hospitals, where its use has its many advantages because of VLC does not emit EMI or RFI, so it does not interfere with

medical instruments and also it does not interfere with MRI scanners. Visible light communication systems do not allow for high mobility through obstacles, providing a relatively secure method of transferring information between a transmitter and receiver. Only those directly nodes facing one another will be able to obtain any information. Hence VLC provides equipment and staff communications with no problems.

2.3.2 Outdoor applications

Many modern applications uses the visible light to portrait specific information. Using the VLC in tandem with these devices could increase their functionality. An example of a device that can benefit from a visible communication system is a traffic or stop light. The traffic lights are used to maintain the traffic flow. Because these lights are common in major cities, incorporating some sort of communication system in them to allow our society to stay connected and up to date with all sorts of information improves overall efficiency through multitasking. The use of VLC comes when dealing with traffic lights, a driver or pedestrian remains idle while waiting for their turn to proceed. The majority of the time, this time is simply wasted by remaining idle. If a visible light communication system was connected to a traffic light, the user could potentially use his/her phone or car head lights to connect to the traffic lights and retrieve some form of information. The information may be about local traffic, or even directions to a specific location. The system could even be used as a local connection to access the internet. By doing this, the user can have an alternative means of accessing data instead of his/her costly and limited 3G or 4G data connection.

There is a big imposition for applications in this field because of the availability of the infrastructure where cars containing LED lamps addition to traffic lights and traffic signs that adopt the LED technology. Using this technology, cars can communicate with each other, helping to prevent or reduce traffic accidents as well as traffic signs that can communicate with cars to provide them with some information about the road such as safety of road, Whereas Traffic lights and many cars use LED-based lights Which contributed to the use of visible light communication as shown in **Fig.11**.

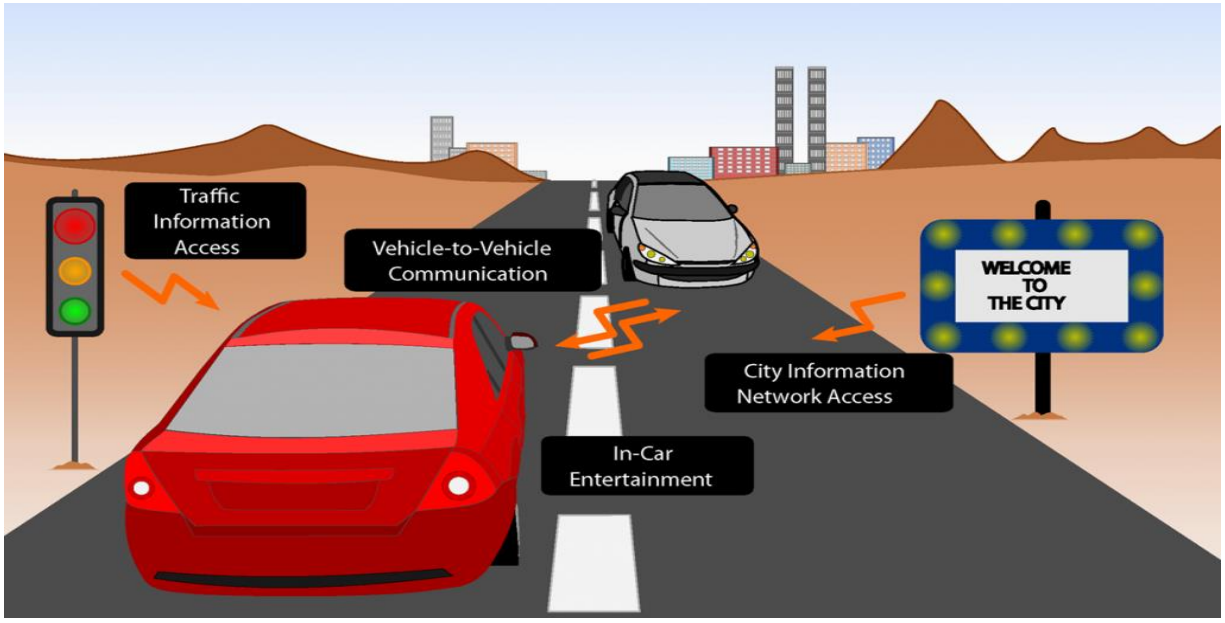


Figure 9- outdoor application

Source: Source: http://visiblelightcomm.com/top-10-visible-light-communications-applications/

Furthermore, visible light communications could be used for smart lighting as smart lighting is an important requirement for smart homes and buildings. Using VLC with smart lighting makes it more efficient. In addition to providing the infrastructure needed for illumination, light sources can be used to transfer signals which control lighting in a smart manner. This will reduce the cost of wiring as well as the energy consumption in homes and buildings.

2.3.3 Indoor application

Each visible light information source can be uniquely identified, so the location of any VLC device can be identified quickly and accurately.

Transmission of a unique ID is all that is required for basic positioning for providing local information in museums, communications for civil contingencies. Multiple LED light bulbs can be used with relative location for more accurate indoor positioning and navigation. Also in the malls we can provide VLC tags for positioning and localization purpose.

One of the major applications of VLC, especially in the medical or industrial field, consists of estimating one's location or a certain box containing goods in a stock. VLC is very good for indoor positioning. Propose a scenario for visually handicapped people or for the goods. Location estimation is put to use in this scenario to guide people through a series of hallways. All hallways are assumed to be illuminated by fluorescent lights which are capable of transmitting a unique ID via VLC. Estimating the current location consists of two steps: Firstly, the distance to each fluorescent light in reach is computed and secondly, the current position is estimated based on the previously computed distances. The distance to each light source is computed by first measuring the angle of incident light with assistance of a photo sensor that is attached to the person's shoulder. Then, using some trigonometric functions, the distance between.

Another piece of modern technology that uses visible light to portray information is a television. Unlike a traffic light, a television contains thousands of pixels that are constantly changing colors to project an image to its viewers. There are many individual LEDs in a television so it could be possible to allocate to a few of them the task of transmitting information through a visible light communication system. When a user is watching television, there is a possibility that the user may wish to see what else is airing on other channels. To do this with today's technology, the user will have to either constantly switch the channels to see shows that are currently airing on other channels or minimize what was being watched to bring up the TV guide. If the user has access to a smartphone or a computer, he/she could use that to look at the guide. Unfortunately, this requires internet access. Instead of using the internet connection, the smartphone or computer could also incorporate a visible communication system and retrieve the information from the television and display it on the second device, and not affect what is occurring on the television. Also, if the user is really intrigued by what he or she is currently watching but does not know what it is, they could use the communication system to transmit the program information to their other device. One drawback to using a visible communication system on a television is the fact that a few pixels are dedicated to transmission and potentially could affect what is being displayed. To not disrupt the user experience, the LEDs must be placed somewhere that will not affect what is being displayed. One way to accomplish this is to place the LEDs away from the display, or use

the LED to indicate that the television is ready to transmit the information. Similar to the traffic light scenario, the receiver will need to minimize the noise that may come from other light sources. This could be accomplished by filtering out all but a few light color frequencies.

2.3.4 Other applications

In some hazardous environments, such as mines, petro-chemical plants and oil rigs, RF cannot be used because of the risk of explosions. Wireless communications are very important in such places to communicate between workers, and transfer sensors data and control signals. VLC is a good and safe alternative to RF in such environments. Underwater communications are used in many important applications such as communicating between divers, submarines and underwater wireless sensor networks (WSNs). Wireless underwater communication is a challenging task. The attenuation in water for RF, especially in electrically more conductive salt water, is very high due to strong signal absorption in water. This makes RF communication impractical under water. VLC can be used to provide a high data rate under water communication.

2.4 Goals and Features

To ultimately be able of sending and receiving data from one point to another using only the visible light; is the goal of this system. This system would be able to transmit any type of data at high rates. However, the success of this design does not depend on the creation of a new type of communication system that will instantly replace all other means of data transfer. The objective of this system is to be able to send data reliably and accurately over a short distance at a fair speed.

Initial goals for the functionality of this system include being able to send text, pictures or videos over a distance of approximately one meter at a data rate of at least 2 Mbps. To do this, the transmitter circuit of the design would receive a signal from the FPGA after some processing which will be done to enhance the signal, the FPGA will receive its signal from the computer using a specific software. The transmitter circuit will control the flashing of the LEDs to send bits to the receiver which will pass this data to the

FPGA for the decoding and reprocessing of the bits to detect only the desired data neglecting the noise, then displaying a computer.

Additional functions that would enhance the project but are not mandatory goals of the design include sending video, sending data at a distance more than one meter and transmitting data with very high rates that could reach X Gbps. There are many other features that could be added to the design to increase its functionality and its flexibility like using different colored LEDs simultaneously to increase number of users and/or increase the rates of data transfer, also the use of some modulation techniques by doing some processing on the FPGA will increase the integrity and the performance of the designed system. The main reason for not including these features in the designed system is the time constraints. Furthermore, there are some stability issues when data is being transmitted at higher frequencies as some electronic components become less ideal.

3 Chapter 3

3.1 MATLAB simulations

A simulation program for indoor visible light communication system based on MATLAB. This simulation aims to demonstrate the power distributed over a certain area, in order to accomplish network coverage. The program considers the position of the transmitters and the effect of all of them on the transmitted signal by one of them that transmit data at a definite time slot, the illumination of a light-emitting diode (LED) is used not only as a lighting device, but also as a communication device for transmitting any type of information.

3.1.1 Modeling of optical channel

There are some physical parameters for developing the simulation program at specific conditions. The size of the office room size is 15m x 15m x 3m and the LEDs are installed on the ceiling of the room, the height of desk in which we assume receiving at is 0.85 m and the receiver is placed on the working plane surface in the room model. The simulation parameters are listed in Table 3

Semi-angle at half power	**30[deg.]**
Center luminous intensity	0.73[cd]
Number of LED each group	10
Field of view	30[deg.]
Area of room	15x15m
Height of the simulated device	0.83m
Illumination of one LED	723.2 LX

Table 3: simulation parameters

Semi-angle at half power is the angle between the perpendicular and the line that contains half the power in our case is 30[deg.] see **Fig.10 and Fig.11**. Center luminous

intensity is the intensity of light measured at the center of the led in our case is 0.73, Number of LED in each group fixed in the ceiling is 10 LEDS each the illumination of one LED 732.2 LX.

The distribution of luminance at a working plane is plotted. It is assumed that the source of emission and the reflected points on wall have a lambertian radiation pattern. Lambertian emission means that the light intensity emitted from the source has a cosine dependence on the angle of emission with respect to the surface normal. Following the function for an optical link the luminous intensity in angle **φ** is given by:

$$\mathbf{I(\varphi) = I(0)\cos^{m}(\varphi).} \qquad \textbf{Eqn. 3-1}$$

Where I(0) is the center luminous intensity of the group LEDs, φ is the angle of irradiance, m is the order of Lambertian emission and is given by the semi-angle at half illuminance of the LED φ ½ where it is a 60 degree cone that contains half the power as:

$$\mathbf{M = -\ln 2 / \ln(\cos \varphi 1/2).} \qquad \textbf{Eqn. 3-2}$$

$\Phi 1/2 = 60^{o}$ then, M=1.

A horizontal illuminance (**E_{hor}**) at a point (x, y, z) on the working plane is given by:

$$\mathbf{E_{hor}(x, y, z) = I(0)\cos^{m}(\varphi) / D^{2}_{d} * \cos(\psi).} \qquad \textbf{Eqn. 3-3}$$

Where **$D_{d\ is}$** the distance between transmitter and receiver, **ψ** is the angle between line incidence of light and the perpendicular of the array of LEDs.

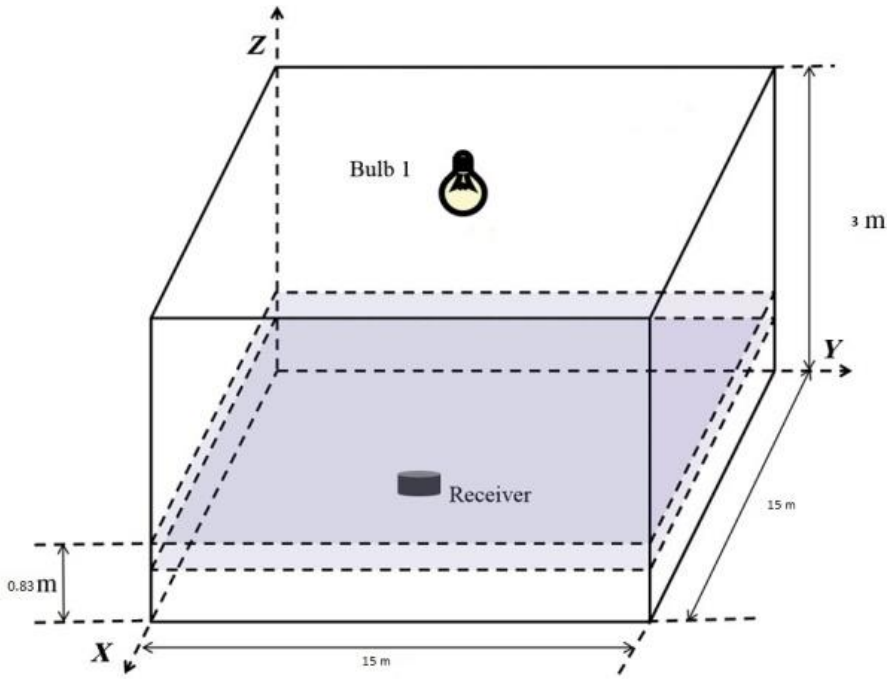


Figure 10-the position of transmitters on the ceiling

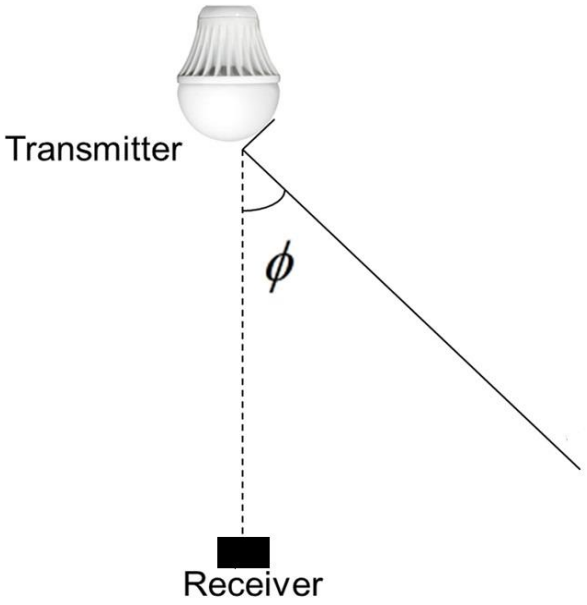


Figure 11-transmitting bulb – semi angle

To survey the illuminance distribution of one LED in the system where it is located at the center of the room and the receiver is moved over a height of 0.83 m as referred in **Fig. (10, 11)** in the horizontal plane to cover the whole area then by calculating the power illuminance distribution and then plotting it.

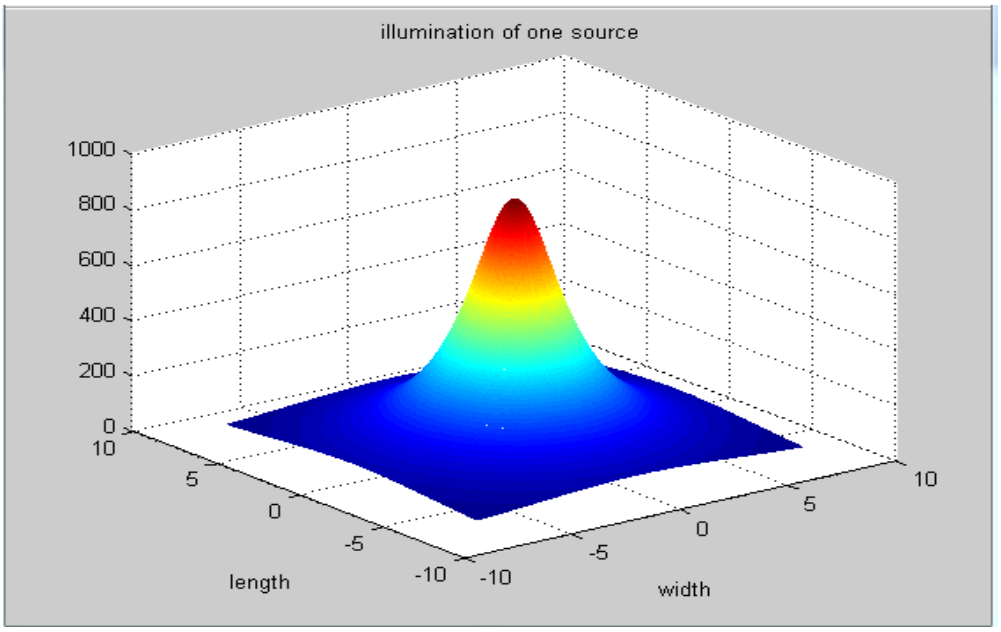


Figure 12-Illumination of only one light source

The distribution of illuminance of only one source that consists of ten LEDs (**see Fig.12**), is plotted using mesh plotting as $\mathbf{E_{hor}}$ **(x, y, z)** where it has maximum power of 7232 ill. At peak, and minimum power at the corners of the room where there is no coverage signal so, a group of LEDs cannot cover the whole area defined above so we need another solution as regards to this problem. A solution could be obtained by using many sources attached to the room ceiling, this term of multiple carrying data LEDs introduce a new type of problem. Interference is inevitable, as Signal to interfering noise ratio (**SNIR**) will increase significantly, as the noise becomes two terms the ambient light, light that is only used for illumination and LEDs that carry different types of data, this introduced problems must be dealt with each in one time.

The ambient light if it is relatively small in respect to the carrying data LEDs, it can be filtered with a high pass filter. As OOK modulation is used in this communication

system, the difference in received power is what does really matters, so as long as we can detect the light source with our eyes we should be able to detect that high frequency flickering signal using photodiodes, by this way we can separate the useful signal from the ambient light.

The second problem is the interference between two sources of carrying data LEDs, can be solved by time domain multiple access TDMA where time is divided into slots and activating each of carrying data LEDs in different slots, this is way no data interface happens at all unless a miss in choosing the time slot, the other way is frequency division multiple access FDMA, where each carrying data LEDs is operates at different frequency where it's filtered by a band pass filter.

In case of the receiver in x-y plane and has angle of incidence which is the angle between perpendicular at the light source and the straight beam of light given as **Θ** (see **Fig.13**):

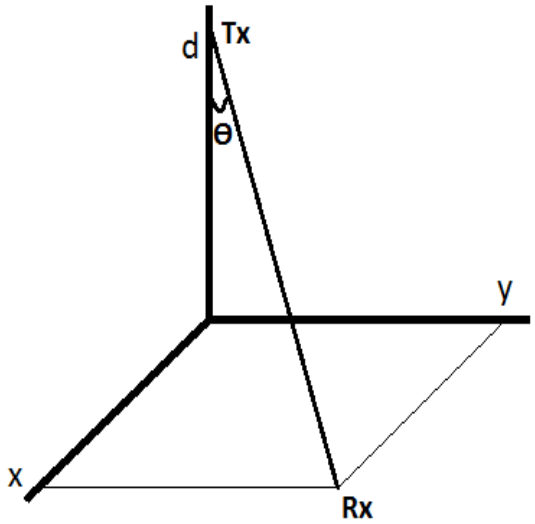


Figure 13- X-Y plane related to the angle of incidence

$$\Theta=\cos^{-1}\left(\frac{d^2}{d\sqrt{X^2+Y^2+d^2}}\right). \quad \textbf{Eqn. 3-4}$$

Straight beam of light that hit the receiver, X is length of the room, Y is width of the room. And d is the vertical distance between led and the plan of the receiver (Height of the roof~0.83).

Direct distance between Transmitter and receiver is given by the relation:

$$\mathbf{D_d=\frac{\sqrt{x^2+y^2}}{\sin\theta}} \qquad \textbf{Eqn. 3-5}$$

And the height of the roof is 3 m, **X** is set to 7.5m and **Y** is set to7.5m.

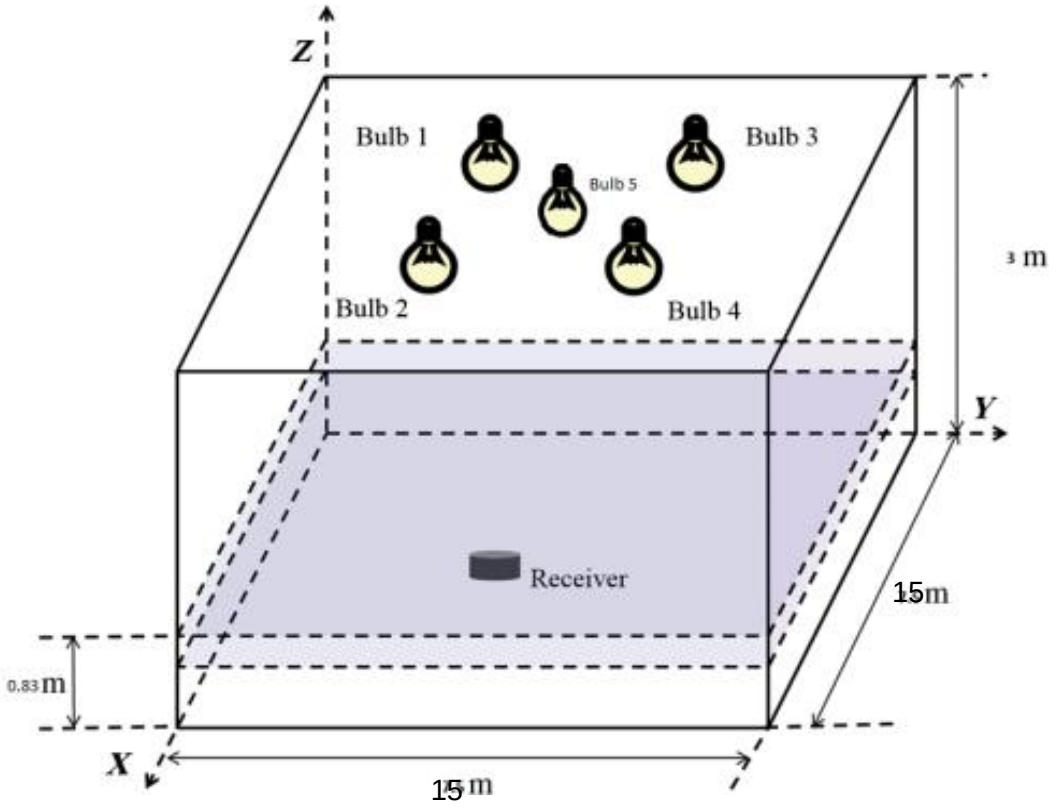


Figure 14- five source of light

By simulating horizontal illumination of 5 sources of LED groups in the room distributed as shown in **Fig.14** by assuming TDMA is used so one group of LED is considered as ambient when it's not transmitting any data, but when it starts transmitting data the OOK technique is used at 50% of power The distribution of illuminance the whole

power is accumulating at the center in this is part is our DC component it is considered to have has maximum power of 7232 ill. The transmitter is also considered at the same height of 0.83 m.

$$E_{hor}(x, y, z)\_Total = E_{hor0}(x, y, z) * 0.5 + E_{hor1}(x, y, z) + E_{hor2}(x, y, z) + E_{hor3}(x, y, z) + E_{hor4}(x, y, z) \quad \textbf{Eqn.3-6}$$

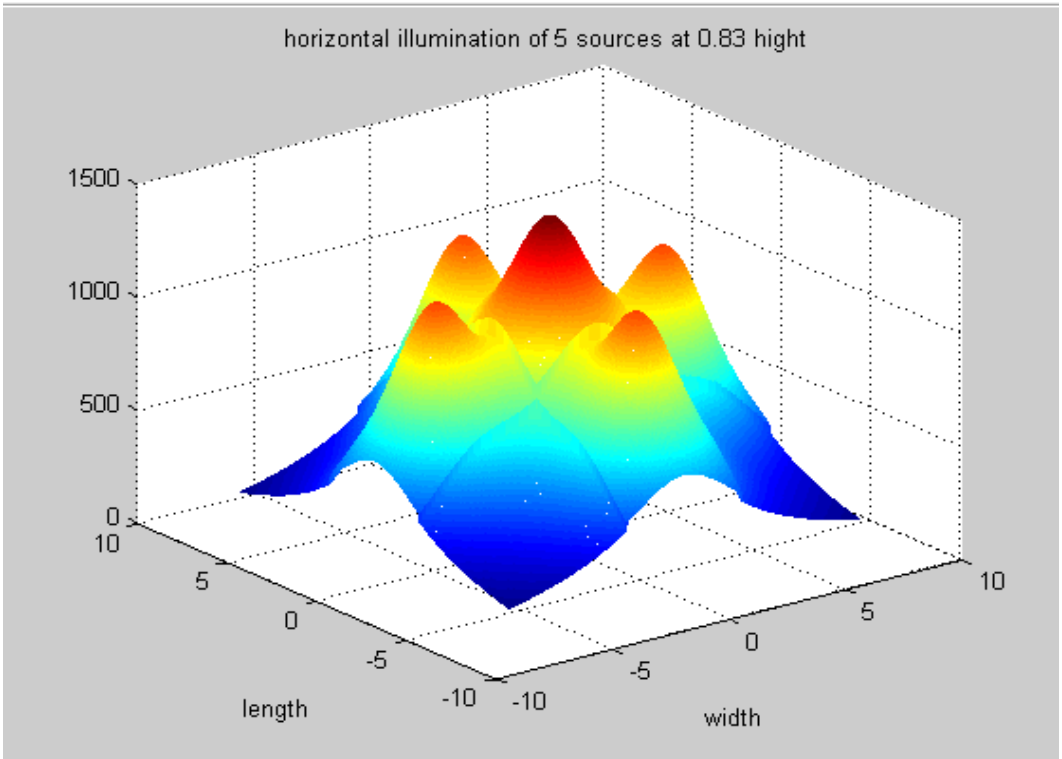


Figure 15- illumination of five sources

$E_{hor}(x, y, z)\_Total$ is the DC component of the system where four of the sources is at full power(see **Fig.15**), but source E_{hor0} is the transmitting source where 50% of the power is just an ambient light so **$E_{hor}(x, y, z)\_Total$** is the unused part in the system, while the used part is the other 50% of this light By dividing the used power by the unused power we get the signal to interference noise ratio (**SNIR**) as shown in **Fig.16** the high part of this graph is the area where group zero of LEDs is covering for networking, other areas are covered by a similar shape like this graph so the system must be designed to get coverage

at the intersection between all simulated graphs to cover the whole area, So this is our limitation in terms of SNIR.

The LED transmitting power is not turned off completely it only has a depth of 50% of its full power.

$$\mathbf{I(0)\_depth = I(0) * 0.5} \qquad \textbf{Eqn. 3-7}$$

$$\boldsymbol{SIR} = \frac{\mathbf{Ehor(x,y,z)\_AC}}{\mathbf{Ehor(x,y,z)\_Total}} \qquad \textbf{Eqn. 3-8}$$

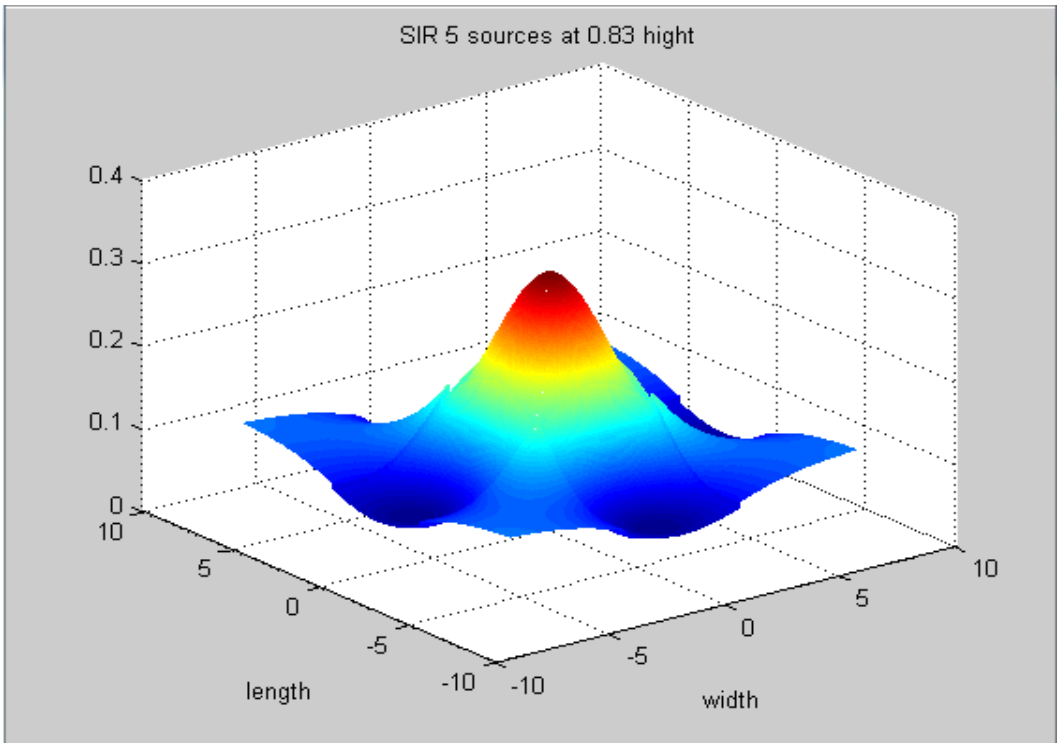


Figure 16-signal to interference ratio at a height of 0.83 meter.

4 Chapter 4

4.1 Design approach

This chapter will illustrate the functional block diagram of the system, the required specifications for each block of the system architecture and how this system is implemented. The functional blocks of the transmitter and the receiver side have different functionalities and implementations. The blocks include power sources, analog circuitry, a FPGA, and a computer.

4.2 Functional block diagram

Fig17. shows the overall block diagram of the whole system. The transmitter side consists of the source of the signal, FPGA and an analog circuit which incorporates the LEDs. The receiver side is similar to the transmitting side, containing an analog circuit which incorporates the photodetectors, FPGA and a device which is capable of understanding the data and show the output.

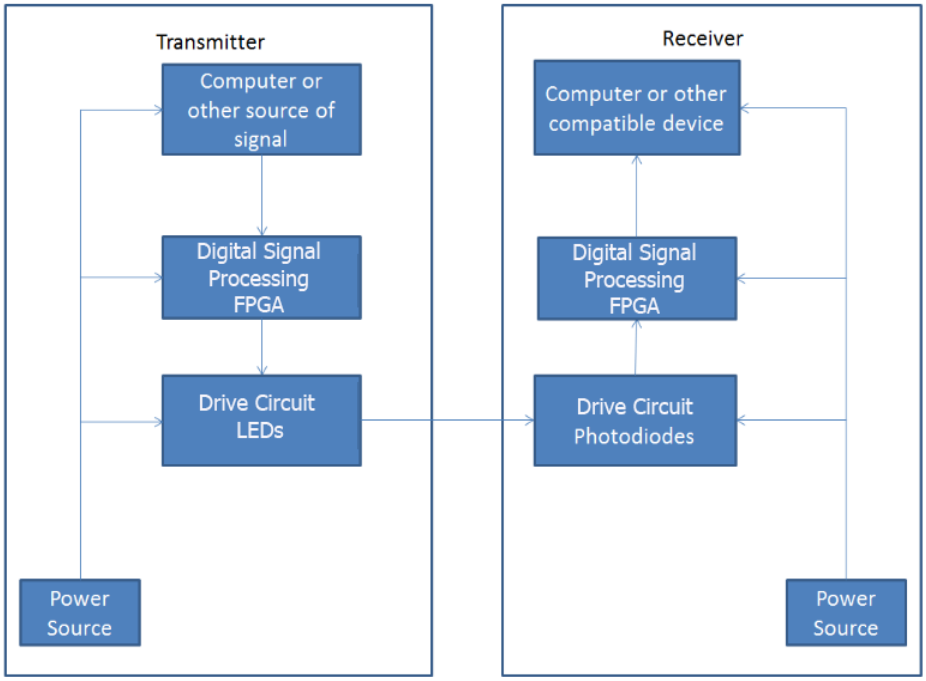


Figure 17- Functional block diagram

4.3 Transmitter circuit

VLC transmitter driver circuit is an electro-optical device that uses visible light to transmit data over wireless medium. The transmitter driver circuit is used to drive the current needed to operate an array of LEDs. The input to this circuit is the binary data signal coming from FPGA which is small signal. So, the function of this circuit is to amplify the current of the data signal to be able to operate the LED array. Transmitted data rate is limited by the switching speed of the transmitter LEDs and other components, while the distance between transmitter and receiver is limited by the transmitted power and ambient light sources. Now we are going to illustrate some of the used components in the transmitted circuit.

4.3.1 Light Emitting Diodes (LEDs)

In VLC, visible light is used to transmit data between transmitter and receiver. This visible light is produced using light emitting diodes (LEDs). LEDs have many effective parameters must be considered when designing the circuit. The most important parameter is the brightness of LEDs which is measured in units of Lumens.

Another important parameter is the amount of current drawn by the LED. This current affects the power dissipated by the LED. Power dissipated by each LED is calculated by the following equation:

$$P_{LED} = I_F V_F \qquad \textbf{Eqn. 4-1}$$

Where I_F and V_F are the forward current and forward voltage of the LED respectively. In this design, an array of 16 LEDs is used to transmit data. Referring to the datasheet of the used LEDs, the typical forward current and voltage of each LED is 30 mA and 3.3 V respectively. This means that the power dissipated by each LED is approximately 100 mW, and the total power dissipated by the array is approximately 1600 mW.

Also, frequency response of the LED is a very important parameter. Transmitted data rate is limited by the frequency response of the LED.

4.3.2 Operational amplifiers (op-amps)

Op-amps are used in this design to amplify current of the data signal. There are two main specifications required in the used op-amp. The first one is the slew rate. High slew rate is important to achieve a high transmission data rate. The other specification is the op-amp output current. It is required to drive a high current needed in the LED array.

Initially, the op-amp **OPA2677** is used in the transmitter driver circuit. The current drawn from this op-amp is not enough to operate the LED array. So, another op-amp is used which is **LT1206**. LT1206 is a current feedback amplifier with high output current drive capability. Another difference between OPA2677 and LT1206 is the IC packaging. OPA2677 has a surface mount package but LT1206 has a DIP one which is easier to deal with and test. Some problems are faced while dealing with surface mounted IC due to soldering problems. Table 4 and **Fig.18** compare between the two op-amps.

	OPA2677	**LT1206**
Slew Rate	1800 V/µs	900 V/µs
Max. Output Current	500 mA	1200 mA
Supply Voltage	±5 to ±6 V	±5 to ±15 V
IC Package	Surface mount	DIP
Price	$11.30	$14.18

Table 4: Comparison between OPA2677 and LT1206 op-amps

Figure 18-OPA2677 and LT1206 ICs

4.3.3 Circuit simulations

There are two designs simulated for the transmitter driver circuit. Simulation is done using NI Multisim™ simulation software. More details about both designs are explained in the following sections.

4.3.4 Initial design

The initial design of transmitter driver circuit consists of two cascaded inverting amplifiers with unity voltage gain. The supply voltage of op-amps is ±5 volts. A 3.3 volts, 5 MHz signal generator is used to model the data signal coming from FPGA. LEDs are connected in parallel to ensure that enough voltage is supplied to each LED. **Fig.19** shows the circuit diagram of the initial design, and **Fig.20** shows the simulated input and output voltages.

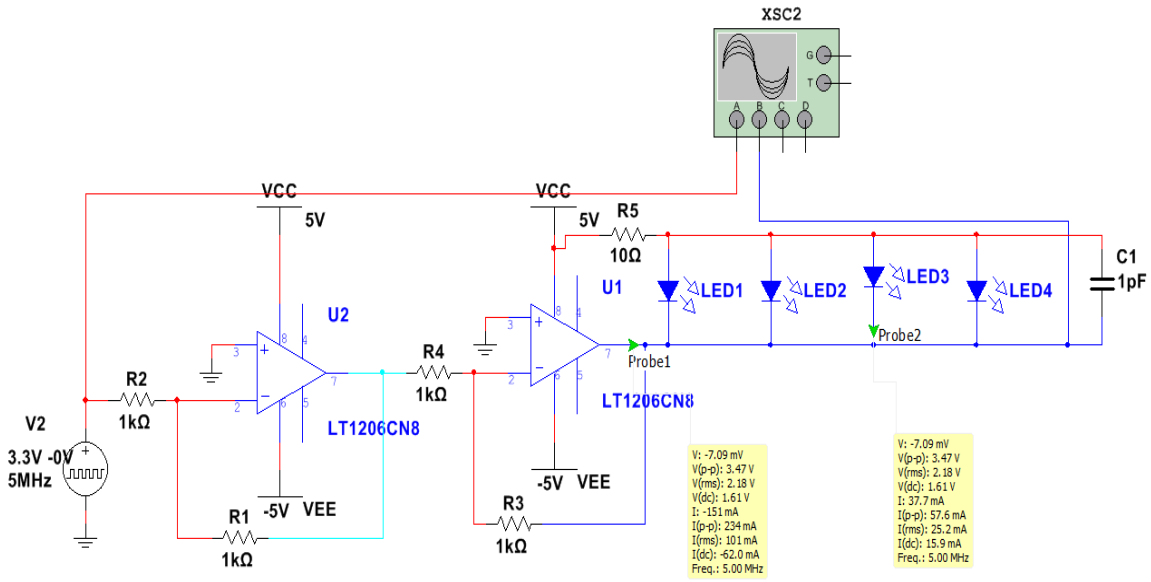


Figure 19-Initial transmitter circuit diagram

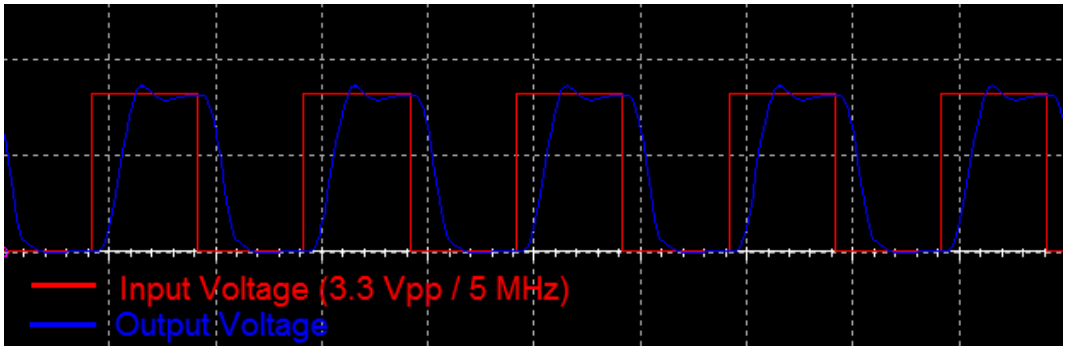


Figure 20-Initial design simulation results

The output current of this circuit is not large enough to operate the required number of LEDs. So, another design is required to achieve enough current.

4.3.5 Final design

In this design a comparator circuit is used to compare input signal with a reference voltage. Reference voltage is assigned to be the middle voltage between high and low voltage values of the input signal. So, if the used voltages for logic '1' and '0' are 3.3 V and 0 V respectively, then the reference voltage is assigned to be 1.7 V. The used supply voltage of the op-amp is ±10 volts. This wide supply range enhances slew rate and response of LEDs. The output voltage is expected to equal to the positive supply voltage if the input is greater than the reference voltage, and equal to negative supply voltage if the input is less than the reference voltage. In this design, the output current is increased, so it can be able to operate a larger number of LEDs. **Fig.21** shows the circuit diagram of the final design, and **Fig.22** shows the simulated input and output voltages.

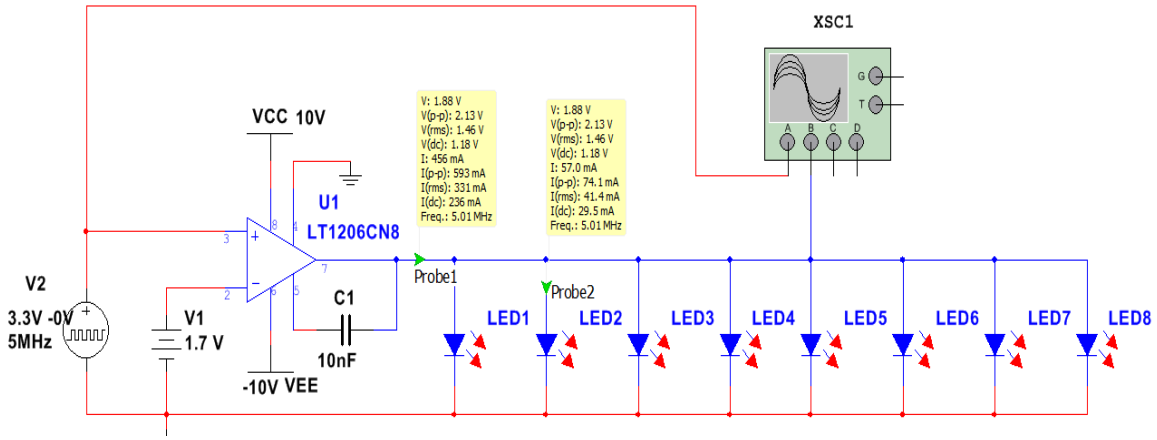


Figure 21-Final transmitter circuit diagram

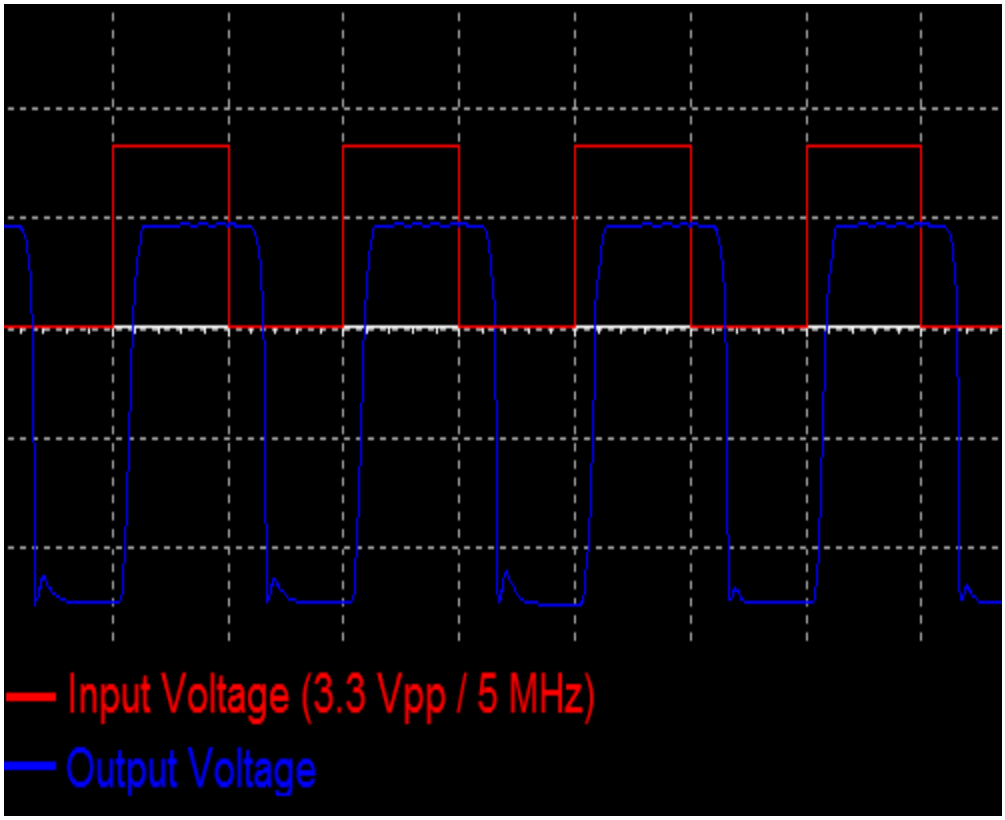


Figure 22-Final design simulation result

4.3.6 Hardware implementation

Hardware of transmitter driver circuit is tested initially on a breadboard and then implemented on a strip-board. To test the circuit, function generator is used to generate a square wave used as input signal. In addition, oscilloscope is used to view the output voltage signal across the LED array. **Fig.23** shows the implementation of the circuit on both breadboard and **Fig.24** shows the same implementation but on a strip-board. **Fig.25** shows testing the circuit with the function generator and an oscilloscope.

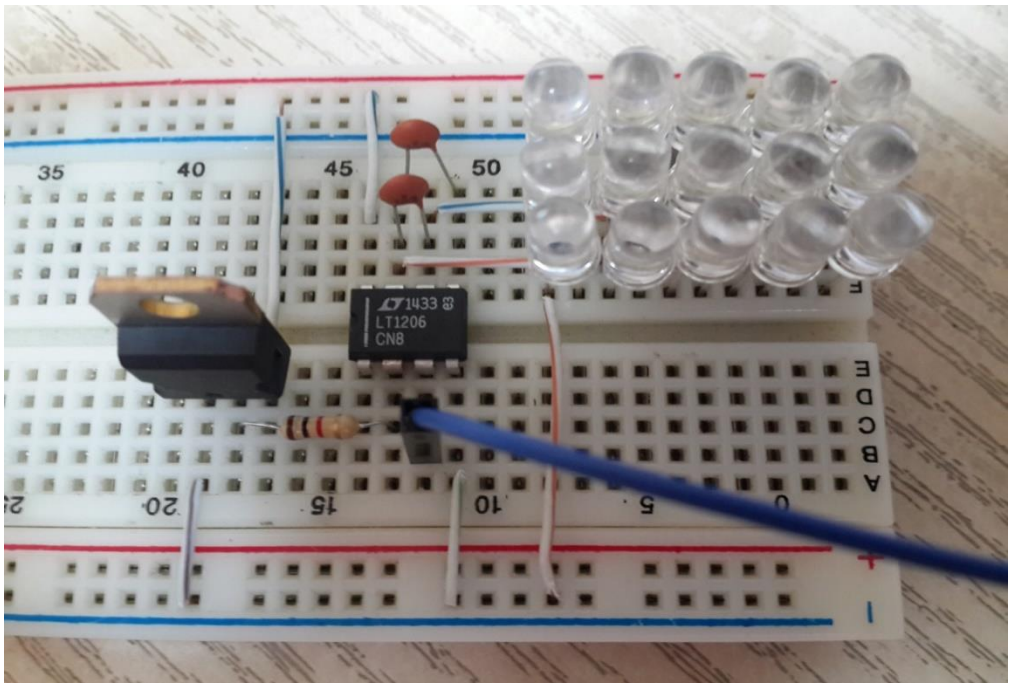


Figure 23-Implementation on a bread-board

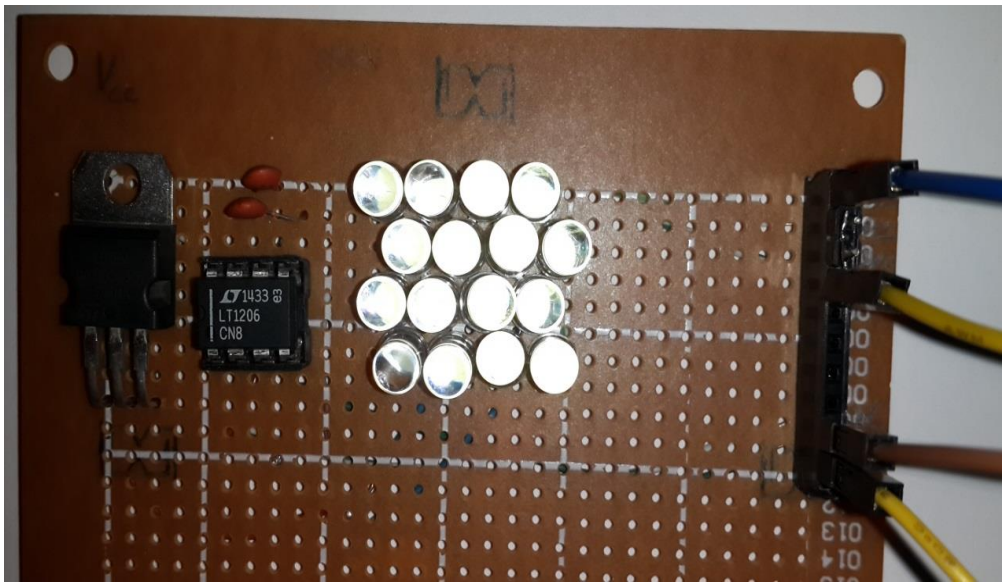


Figure 24-Implementation on a strip-board

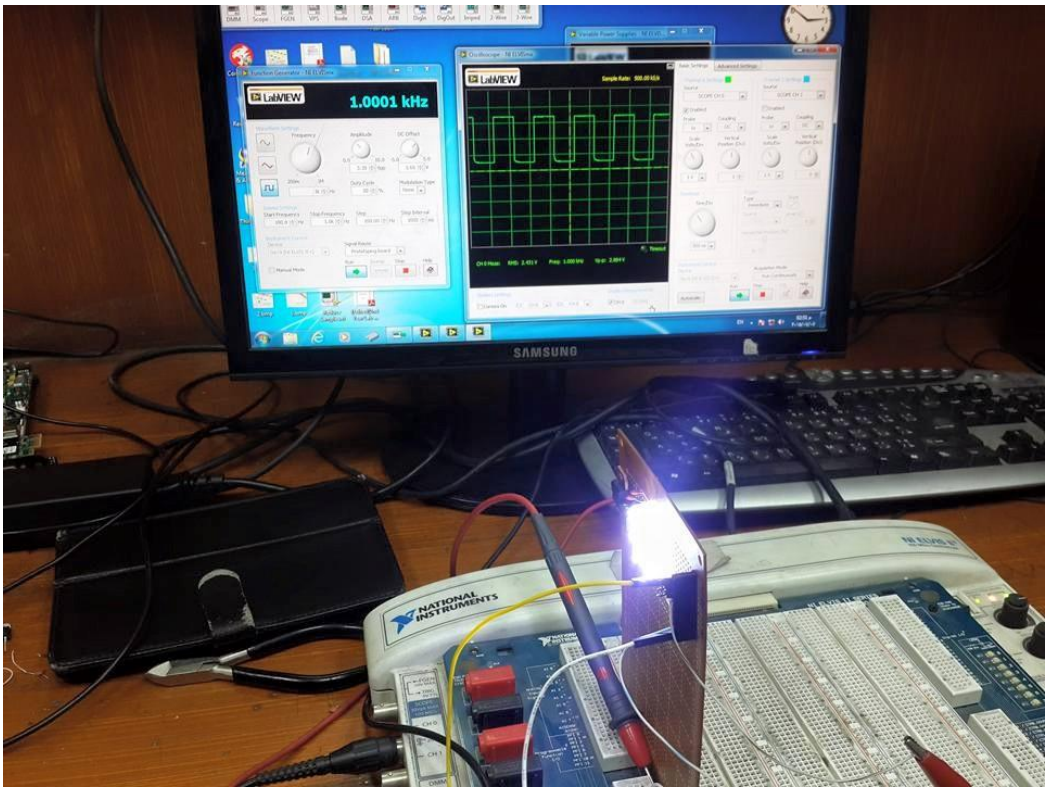


Figure 25- TX cct. Connected to Fn. generator and an oscilloscope

The maximum transmission bit rate achieved in the hardware circuit is 4 Mb/s. The output signal with the maximum bit rate achieved is shown in **Fig.26**. Limitation on transmission bit rate has many reasons related to hardware. Some of these reasons are the

Figure 26-Transmitter output signal with bit rate 4 Mb/s

switching speed of LEDs, frequency response of the used components and parasitic capacitances due to soldering points.

4.4 Receiver circuit

As explained before in the block diagram after transmitting the signal with the transmitter signal, a receiver needs to capture the signal, the main two crucial points about the receiver is the maximum data rate that could be handled, and the other point is the distance in which the receiver can operate at. And there comes the tradeoff between the distances versus the data rate. A lot of experiments have been done to test both distance and the data rate, this details will be demonstrated in this chapter. Main components used are photodiodes, zener diodes and op-amps.

4.4.1 Photodiodes

Photodiodes are considered a devices that converts light into electrical charges. Photodiodes can produce voltage output as well as current output. However, using it as voltage output (photovoltaic mode) produce nonlinear response and a very restricted bandwidth. This later problem can be overawed by using the photodiode as an output current (photoconductive mode) then transforming it using current to voltage op-amp circuit, this allow the circuit to operate at a very high frequency.

Our system is operating at Visible light spectrum (400nm-700nm). So **spectrum sensitivity** of the photodiode must support the visible light spectrum. Another important parameter in the photodiodes called **Dark Current**, which is an existing current in the absence of light; this current arises when the photodiode is operated in *photoconductive* mode. Dark current is generated due to saturation current of the semiconductor junction, this is a serious problem as it shortens the distance which the system can operate at, and this noise also has a frequency gain which needs to be filtered in order to retrieve the signal. **Response time** is also considered for the photodiode, the generation of current inside the photodiode under the effect of electric field moving the electron-holes released when photon is absorbed by semiconducting. The resistance and the capacitance exists in the photodiode affects the response time by time constant $\tau = RC$. Noise equivalent power

NEP is about the minimum detectable power a low NEP is an indication to compare different photodiodes a small NEP is a good indication for a better noise resistance.

Different photodiodes have been put to the test and their key variables were experimented from their data sheet is shown in Table 5.

	SFH203	**Bpw34**	**unit**
Spectrum sensitivity	400-1100	430-1100	nm
Response time	5	100	ns
Dark Current	1	2	nA
Noise equivalent power	2.9x10^-14	4x10^-14	w/√HZ
Price	1	1	$

Table 5: Photodiodes comparison

From this comparison it is clear that SFH203 is more suitable for this system usage as it has a very high response time also it has a smaller noise power and a good relative spectral sensitivity like shown in **Fig.27**.

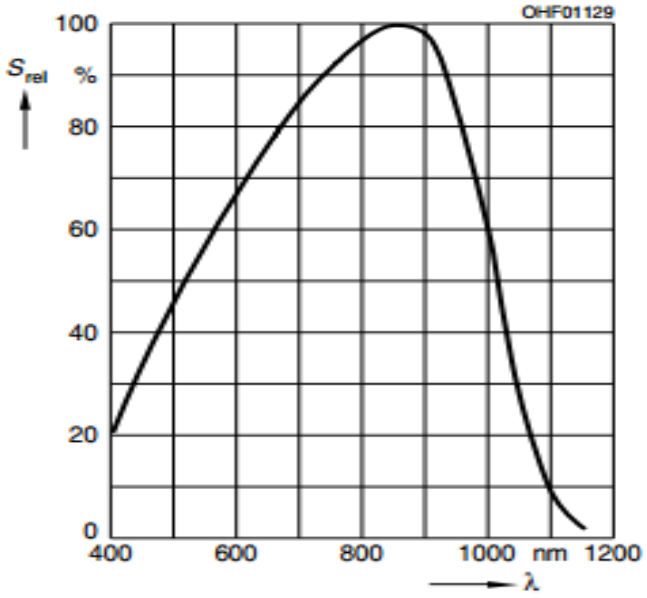


Figure 27-Relative Spectral Sensitivity SFH 203

4.4.2 Operational amplifiers (Op-amps)

Op-amp is such a device that is extremely used in analog building blocks. The op-amp has the potential to have a gain of hundreds of thousands as a differential gain, this is considered a strong advantage that make the op-amp more preferable than transistors. The operational amplifier is superior in two different modes of operation voltage to voltage or current to voltage, it also has nearly infinite input resistance. As the photodiodes output

current is very small. There are multiple circuit configurations that are experimented in this chapter.

There are many of variables in the op-amp that must be considered. **Slew Rate** is the key for choosing the op-amp for this system; it represents the maximum rate of change of the output voltage response, so for 10 MHz with 5v output; a slew rate of 315 v/us. **Input offset current** and **input offset voltage** are not desirable values that get amplified as they have small values but after the huge gain of the op-amp it becomes a significant value with milli-volts, so it has to be considered in the design as the main signal is with milli-volts. Input noise voltage is the minimum signal we can detect surveying for different op-amps as in table.2 to put in test. Table 6 shows a comparison between the op-amps that have been tested and used in the receiver system.

	LM348N	LMH6703	LT1221CN8	LT1360CN8	units
Slew rate	0.5	4500	200	800	v/us
Gain-bandwidth	1	1200	150	50	MHZ
Input offset voltage	7500	1500	600	1000	Uv.
CMRR	90	47	114	92	dB
Input resistance	2.5	1	45	5	MΩ
Input noise voltage	60	2.3	6	9	nv/√Hz
price	1	10	14	4.5	$

Table 6: op-amps comparison

First, for the op-amp **LM348N** is implemented as it will be explained later it reached speed up to 10 KHz, cascaded LM348 could be done to increase data rate, maximum distance was 30cm, **LMH6703** is a very high speed op-amp, on simulation it has reached 30 MHz, this op-amp LMH6703 is surface-mounted, after soldering it on a PCB, there was a coupling between voltage supply pin and one of the op-amp inputs, may be their was a strong coupling that prevents the circuits to have a big gain at the beginning, So **LT1360CN8** was implemented, as it is a 8-DIP package it also has bigger input resistance than the later op-amp. This LT1360CN8 op-amp gives up to 2MHz clock frequency, with filters it has a distance 200 cm with 8 LEDs array using one photodiode. At last **LT1221CN8** was tested as a photoconductive mode (current –to-voltage) amplifier it reached up to 12MHz clock frequency at a distance 10 cm.

With one LED and one photodiode, after 10cm it is found that there is a need to implement band pass filter in order to remove the frequency gain produced by this configuration.

4.4.3 Circuit simulations

First, the receiver circuit is designed based on photovoltaic mode as photodiode is treated as a voltage source that's what exists in the solar cells, so the photodiode is replaced by a voltage source, this circuit consists of two stages of cascaded amplifier to amplify the signal. In **Fig.28,** a simulation to the amplification part and the slew rate of the real electronic circuits it shows that with op-amp **LMH6703** on Multisim simulator. The circuit reached for 30MHz frequency response with a very little attenuation at the output voltage, But things does not turn out as we always wish, when trying to implement this circuit on an electronic board there was one major problem, a gain no more than two has been obtained this problem could be because of soldering the surface mounted op-amp that result in a coupling between the supply and the input, so another op-amps have been bought and used that operates at a high response as it will be shown in the next pages. The output of this circuit is shown in **Fig.29** the input is 300 mv green signal line while the red line is the amplified signal at output stage of the second op-amps.

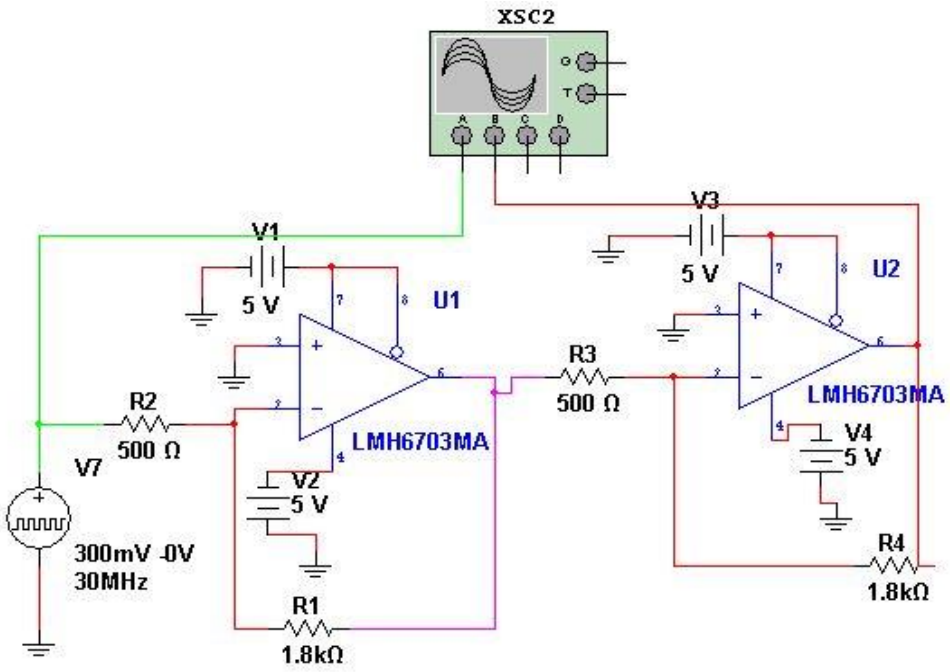


Figure 28-the receiver circuit based on LMH6703 op-amp

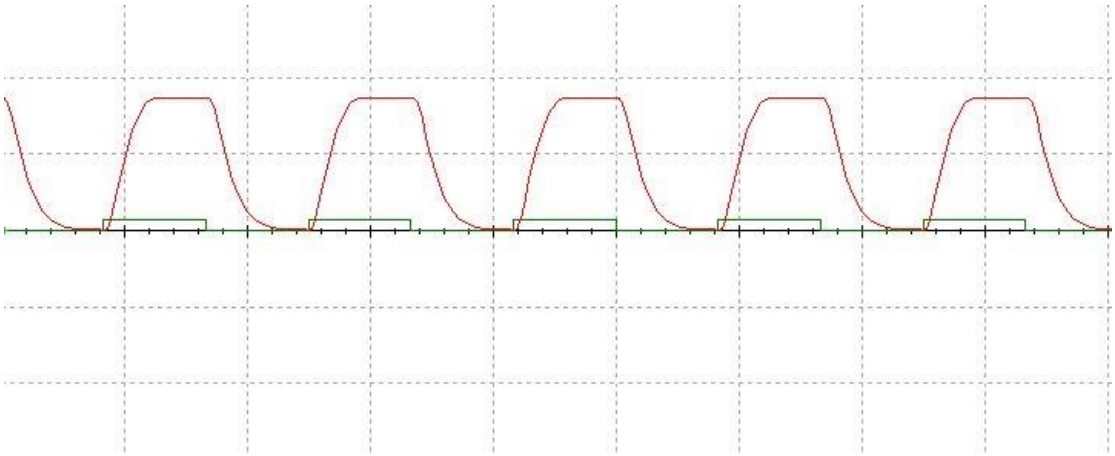


Figure 29-The receiver simulation results based on LMH6703 op-amp

4.4.4 High speed circuit implementation

As mentioned before **LT1221CN8** is implemented, it supported speed up to 12MHZ clock frequency zeros and ones, with OOK modulation 12MHz means that data can be sent by a data rate up to 24 Mbit/sec. The schematic circuit as shown in **Fig.30** and the bread-board implementation shown in **Fig.31**, uses the photoconductive mode (current-to-voltage) amplifier, first the photodiode is fetched with high-pass filter to remove the DC input of the diode voltage effect (0.7 volt). Then an amplifier with a very high input resistance is used to amplify the current, this op-amp is connected as a negative feedback to amplify the current, another small resistance is used on the positive input of the op-amp to solve the problem of the input current offset this circuit couldn't operate without this resistance. Then the output of the first op-amp is connected to a comparator to detect zero crossings of the signal and detect zeros and ones. the supply voltage used in this circuit is +10 volt and -10 volt, this high voltage used increase the slew rate of the circuit dramatically as shown in **Fig.32** from the datasheet of a similar op-amp having same features of this op-amp to reach such a high bitrate is its high gain bandwidth, its low Noise equivalent power and also its low input offset voltage.

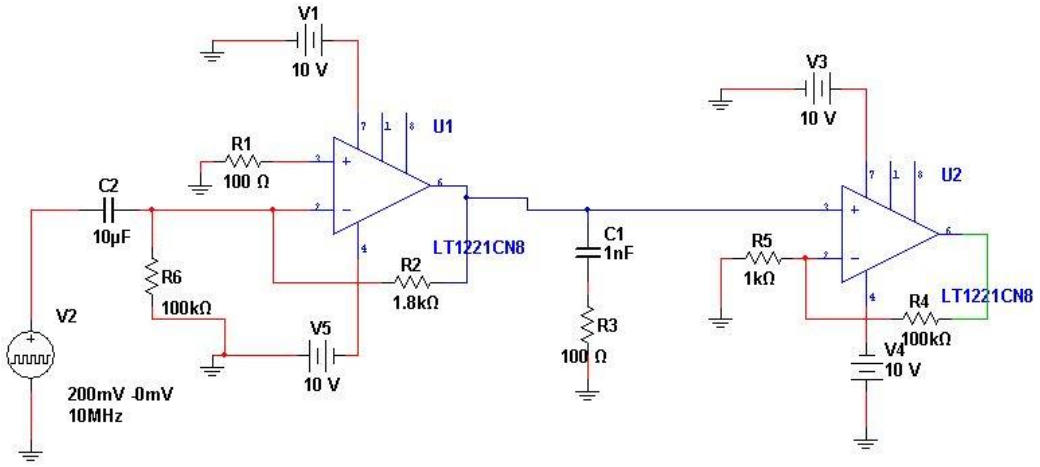


Figure 30-Schematic of the high speed circuit

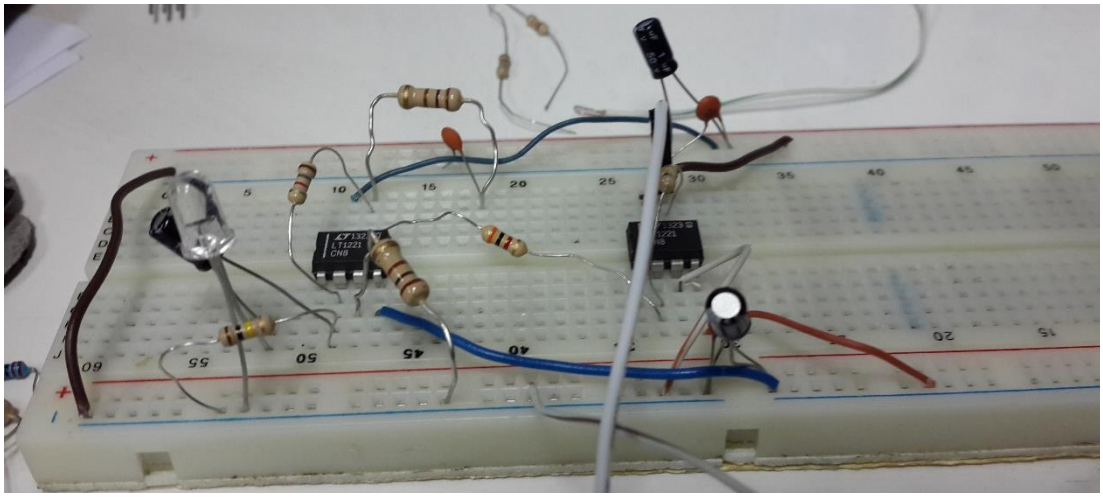


Figure 31-High speed receiver circuit that support 24Mbit/sec

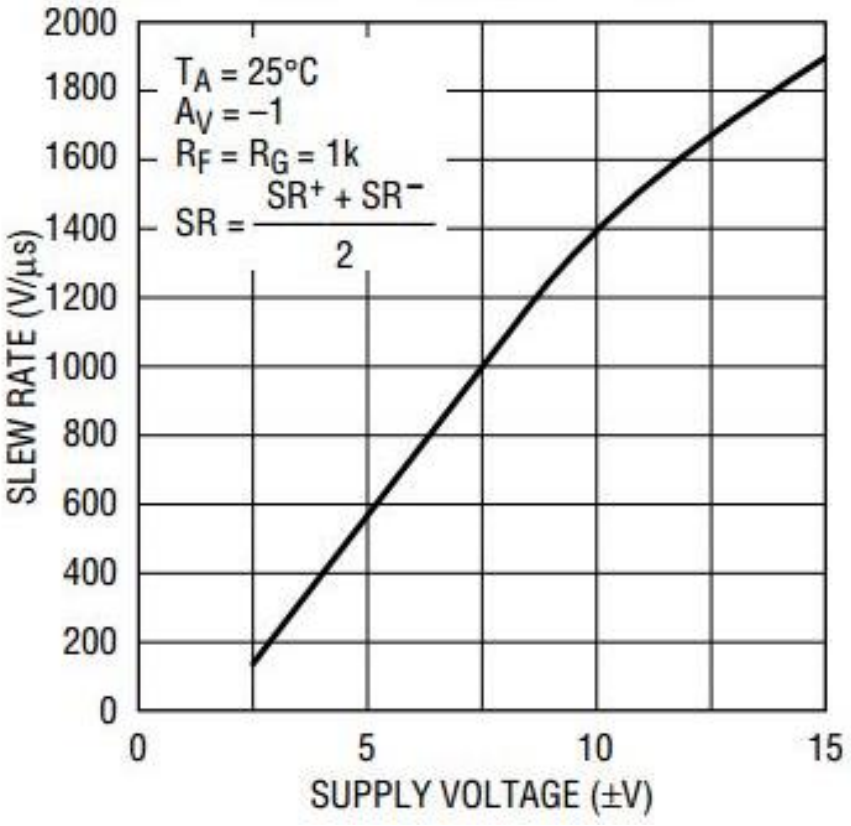


Figure 32-slew rate vs supply voltage of LT1221CN8 op-amp

For the sensor the higher the rates the lower peak to peak to be detected that is the limitation set by the photodiode that limit both of maximum distance and the data rate, after just one amplifier that have gain of 26.5db their peak to peak voltage is as shown in **Fig.33**, **Fig.34** and **Fig.35** at 5MHZ 6MHZ and 10MHZ, all these readings are taken at a very small distance for about one cm.

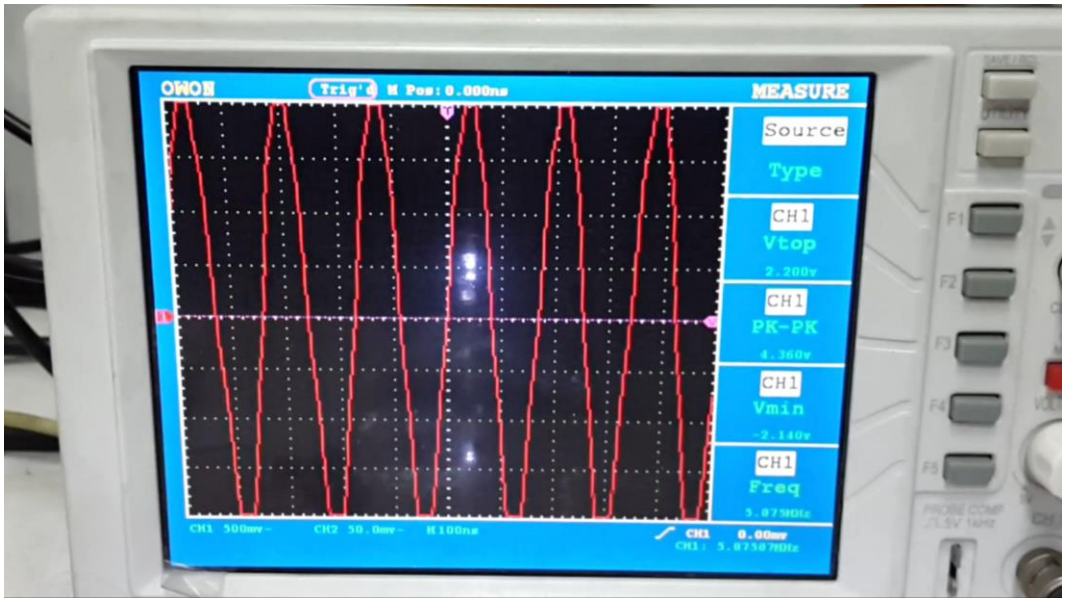


Figure 33-Sensor after one amplifier at frequency 5MHZ with 4.5v pk-pk

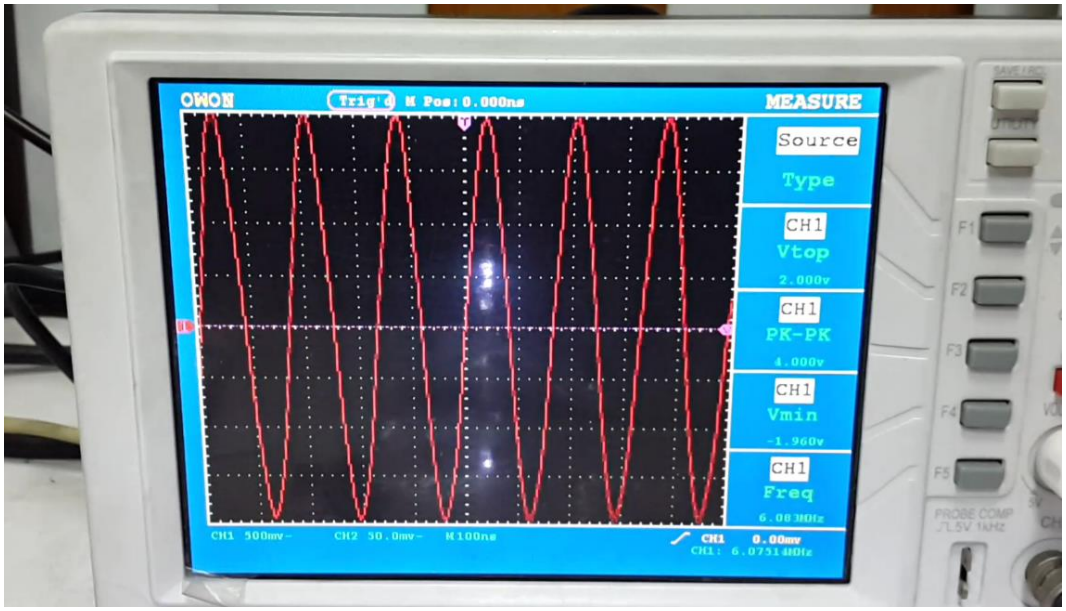


Figure 34-Sensor after one amplifier at frequency 6MHZ with 4v pk-pk

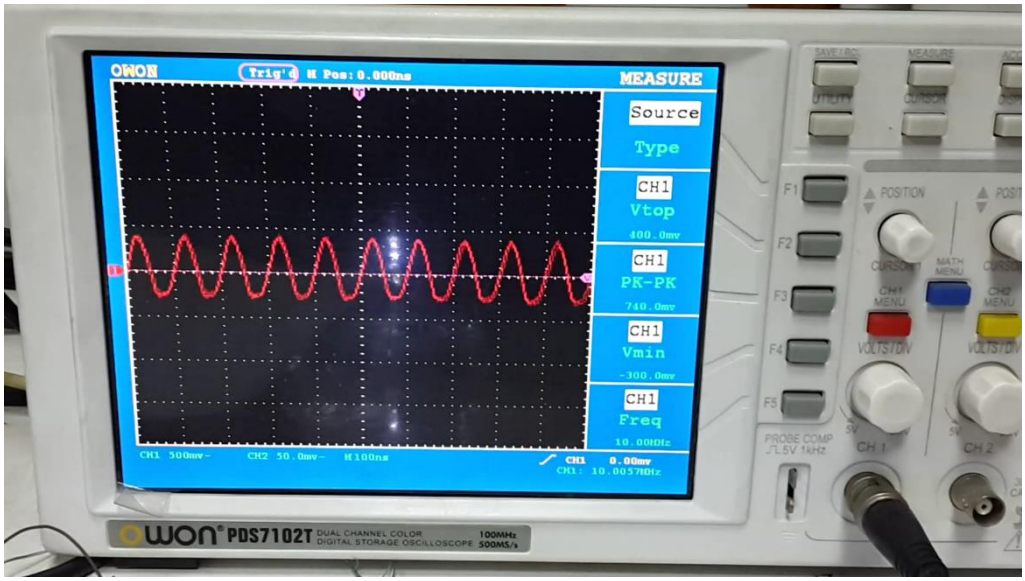


Figure 35-Sensor after one amplifier at frequency 10MHZ with 400mv pk-pk

4.4.5 Final design

The previous circuit only operates at a very short distances as the photodiode requires very high intensity to operate, so *photovoltaic mode* is less sensitive to light intensity so it can operate at relatively large distance, which is more important in this project. This circuit is implemented as shown in **Fig.36 and Fig.37,** the photodiode is connected as a voltage source, it has a voltage drop of 0.7v like a diode, as shown in **Fig.38**. This problem is solved using a high-pass filter as shown in **Fig.39** it has a very large noise frequency, this frequency increase with increasing the signal frequency the as shown in **Fig.40** the signal output of is estimated to be 5 mv peak-to-peak.

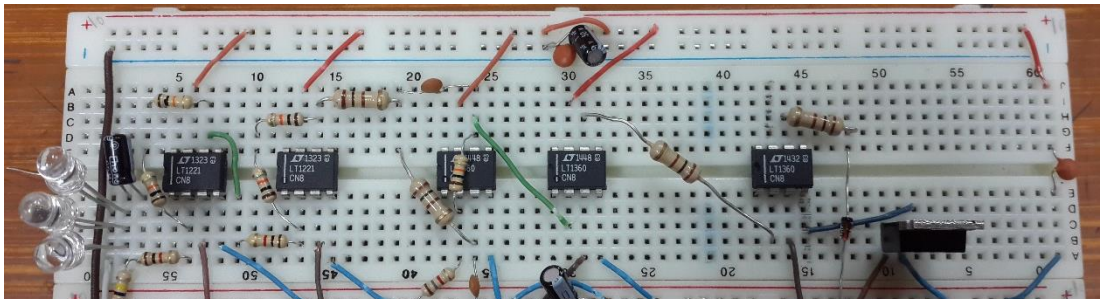


Figure 36-The final circuit used in the receiver circuit on a brad-board

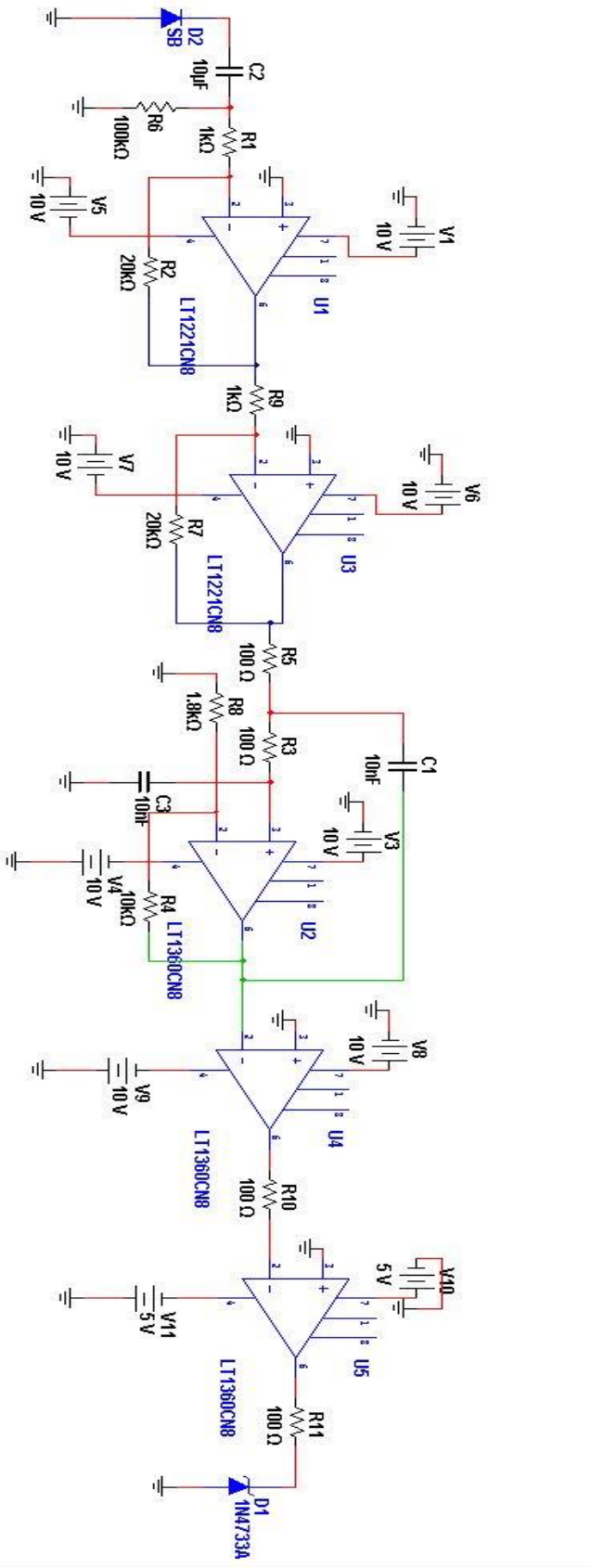


Figure 37-Schematic of the final circuit used in the receiver circuit

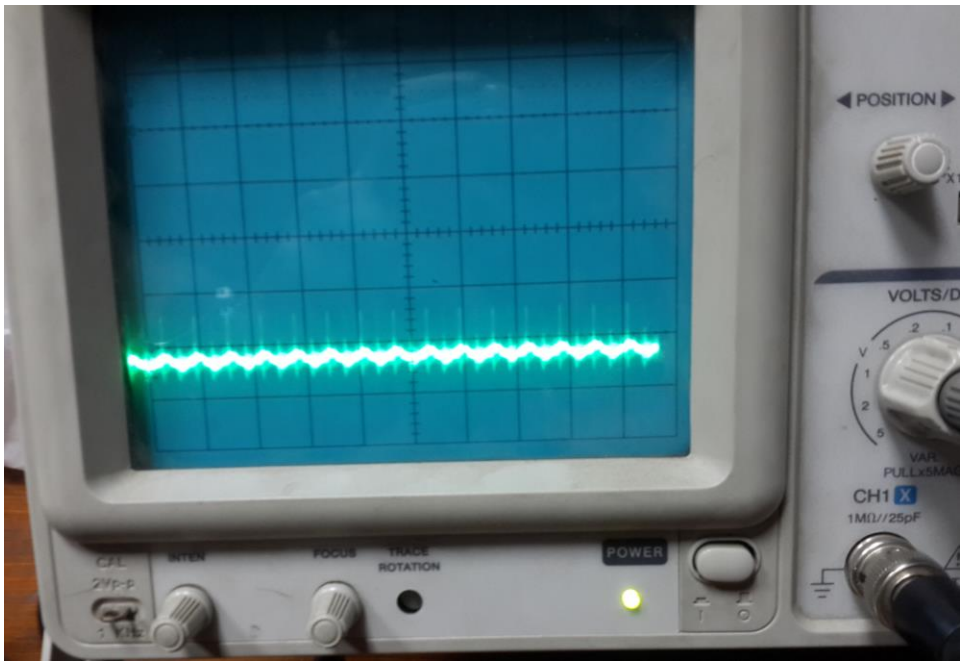


Figure 38-The sensor received signal with 0.7 volt dc shift

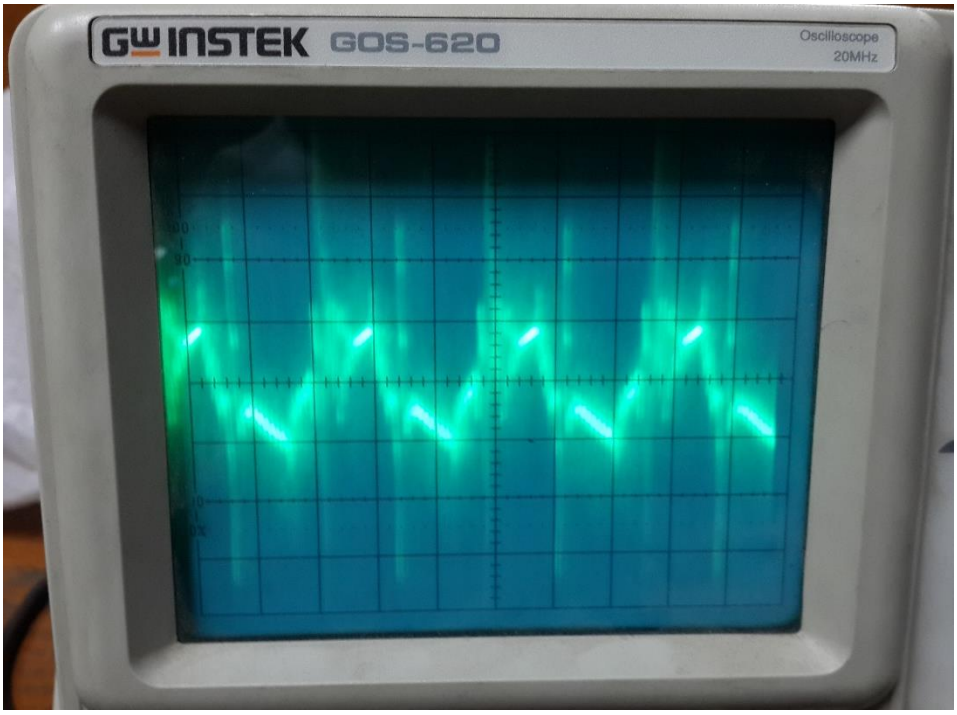


Figure 39-The signal after filtering the dc gain it has 20mv peak to peak and gain 26db

LT1221CN8 is used in this circuit after the photodiode directly as an amplifier then another LT1360CNis used as a cascaded amplifier to give a gain of 52db as shown in **Fig.40.** The signal is modulated by the noise found in the photodiode.

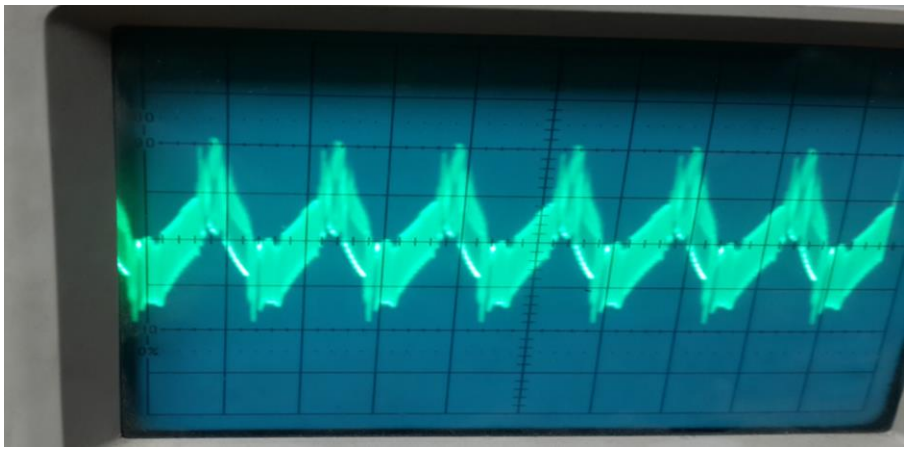


Figure 40-The signal is amplified with the noise

Then the signal is filtered using sallen key filter as low pass filter to remove this noise with the high frequency component and gain with 13db with -3db point of 159 KHz in this circuit design as shown in **Fig.41**.

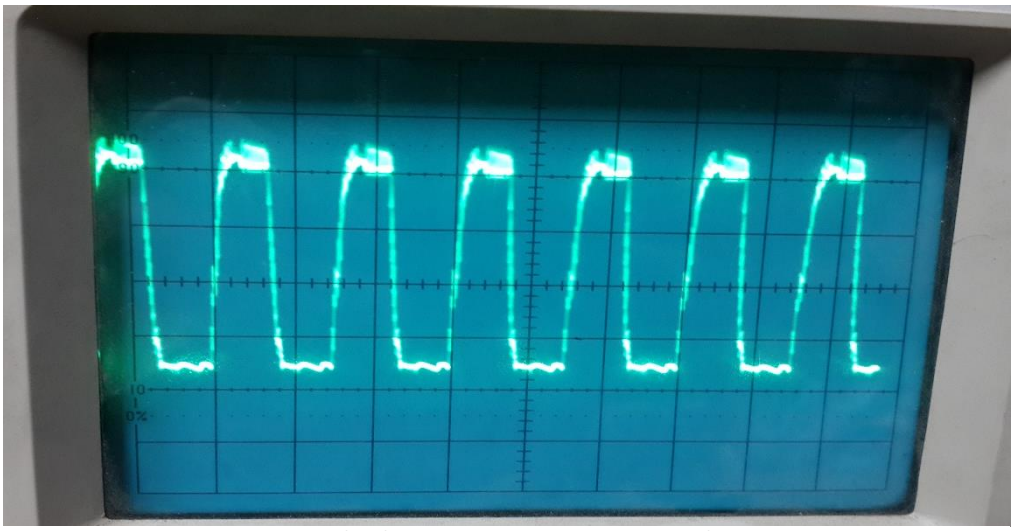


Figure 41-Signal after filtering the noise using low pass filter sallen key

After using sallen key, a comparator is used to digitalize the signal and detect the zero crossing then the last op-amp is used as a level shifter to shift the volt down to suit the FPGA as 3.3 volt as input one and a -0.7 volt as an input zero as shown in **Fig.42** using the op-amp to down convert the voltage to plus\minus 5 volts, then a zener diode in the breakdown is used to convert to 3.3v.

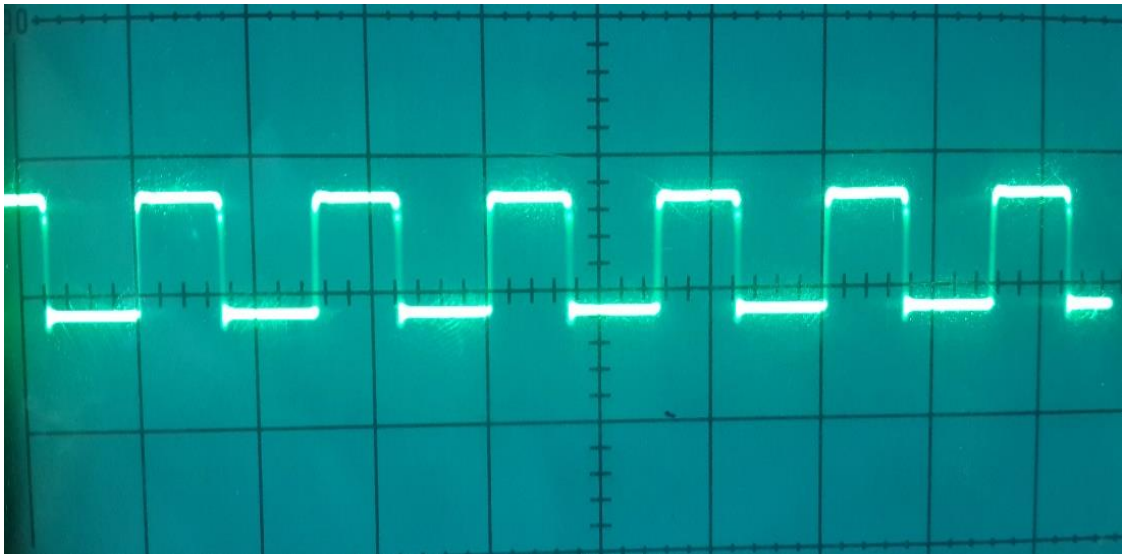


Figure 42-The final output that enter the FPGA with 3.2v, -0.7v

4.4.6 Final system

The overall system shown in **Fig.43** monitors the full hardware implementation, as the transmitter circuit is connected to a function generator producing a signal of frequency 2 MHz letting us to reach a rate of 4 Mbps. The receiver circuit captures the data sent from LEDs using the photodiodes, this circuit is connected to an oscilloscope to observe the reached frequency and bit-rate of the received signal which are the same of the transmitter circuit.

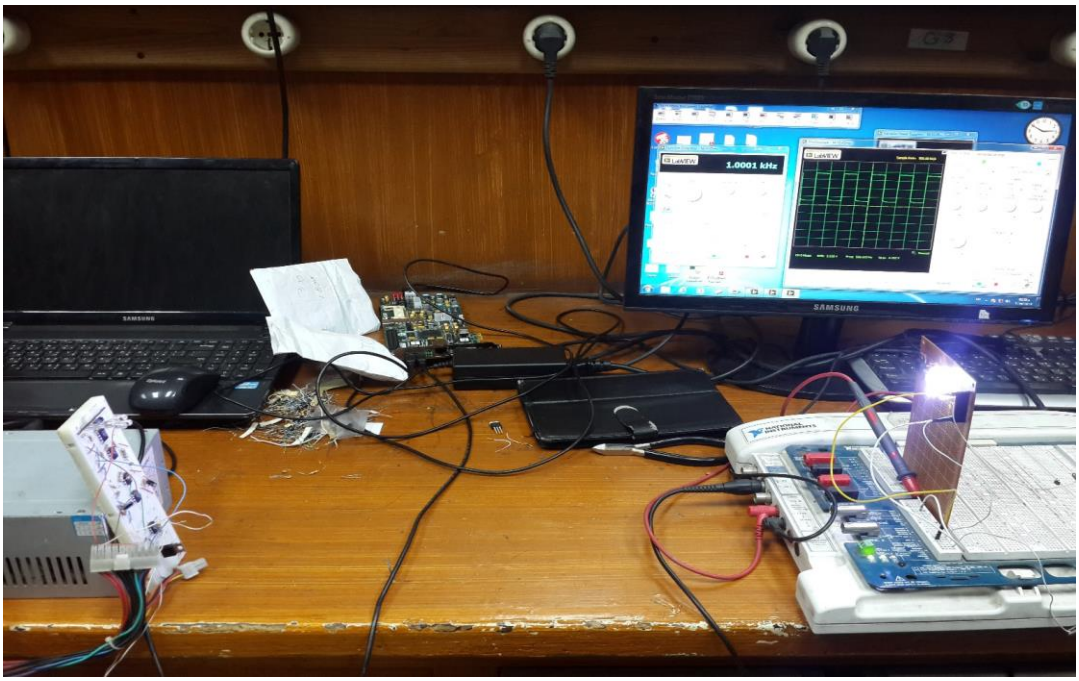


Figure 43- Final hardware full system implementation

5 Chapter 5

5.1 Field Programmable Gate Array (FPGA)

In Visible light communication system as mentioned before a lot of digital processing must be done before producing a prototype or a full system. A lot of solutions are available for such a concern like digital processors, microcontrollers and FPGA kits. One of the suitable microprocessors is MSP430BT5190 which can do the job, but its sampling rate is 1000ns which is relatively low for our system 10Mb/sec is required to be achieved.so processors are inconvenient with this kind of application, as it will be very limited.

FPGAs are high speed devices, an FPGA can reach very high speed, also after writing the code it can be manufactured as a system on chip using VLSI, so FPGAs have a very different perspective from microprocessors. FPGAs has been used to implement this system as it will be indicated later in this chapter.

There are a lot of different types of FPGAs. Spartan 6 FPGA family are chosen for this project as it has low power consumption as long as it is relatively cheaper than other FPGAs, it has less power consumption by 42%, while it also has a higher performance for communication systems by 12% , low cost and low risk during work on it, it has high safety factors. Spartan 6 FPGA family is better than another types where it introduces advanced power management technology. It can support advanced memory, also it has a lot of logic cells up to 150K logic cells and 250MHz crystal clock.

SPARTAN-6 FPGA sp605 Evaluation kit which have been selected from Xilinx family to design VLC system. There are a lot of advantages in this kit. It can be used in higher-level system to design applications which need to implement high-speed serial transceivers. SP605 kit can develop network applications with 10-100-1000 Mbps Ethernet and it has also 128MB DDR3 Component Memory.so Spartan 6 sp605 represents a good candidate for our project.

5.2 Communicating with FPGA

In the Spartan 6 sp605 evaluation kit there are many ways to use the FPGA on this kit. To communicate with this FPGA to control the information needed to be sent after some processing; are the Ethernet and the USB-to-UART ports. (See **Fig.44** and **Fig.45**)

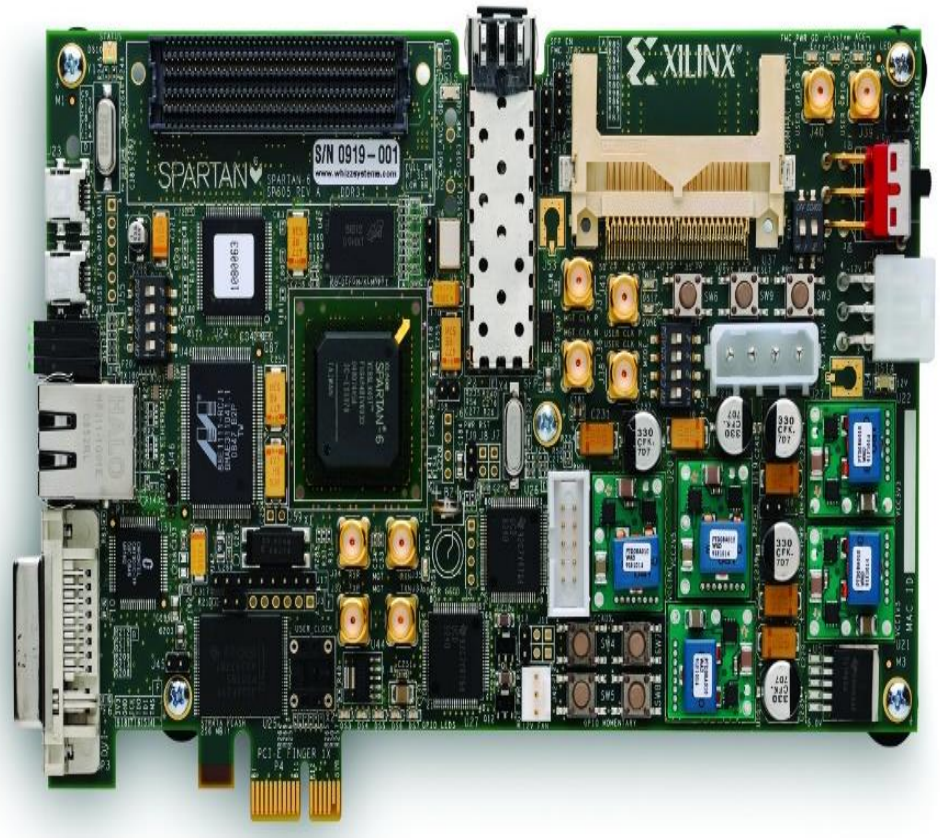


Figure 44- Spartan 6 sp605 Evaluation kit

BOARD FEATURES

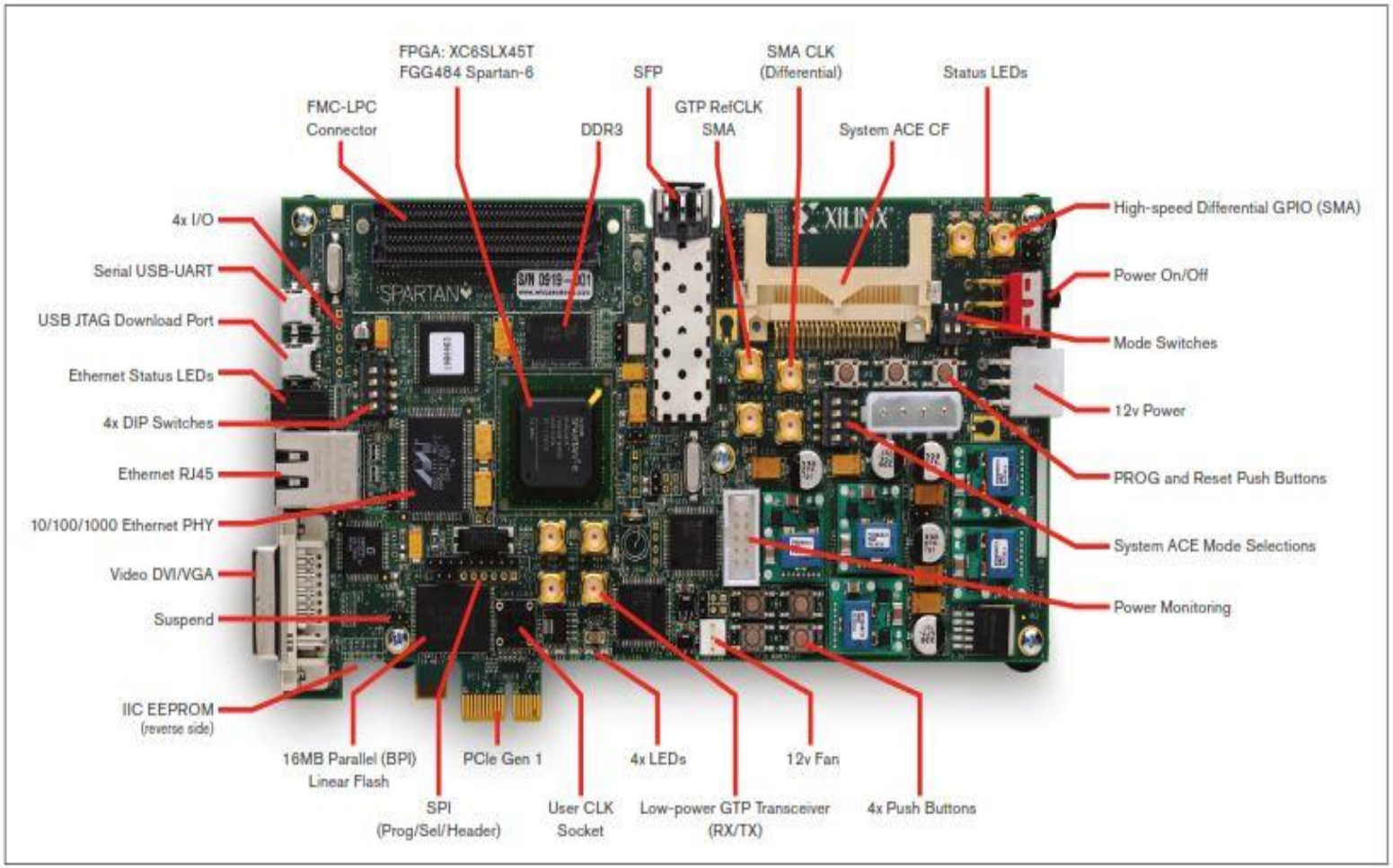


Figure 45- Labeled sp605 kit

5.2.1 Using Ethernet

In this project we were targeting to use Ethernet protocol to connect the FPGA with the computer on both sides as Ethernet is more advanced and can reach very fast speeds, but the FPGA have been arrived lately so to use Ethernet on the kit we were having time constraints. But this in this part we will show how we start using the Ethernet.

The RJ-45 connector is not connected directly to the FPGA IC. first it is connected to PHY device called (88E1111) Marvell Alaska for Ethernet communications at 10,100, or 100 Mb/s. there is a table for the connections between the PHY device and the FPGA pins you can find in SP605_Hardware_User_Guide_ug526 in the detailed description of the Ethernet as shown in **Fig.46** and **Fig.47** as long as the PHY device IC pins as shown in **Fig.48**.

U1 FPGA Pin	Schematic Net Name	U46 M88E111	
		Pin Number	Pin Name
V20	PHY_MDIO	33	MDIO
R19	PHY_MDC	35	MDC
J20	PHY_INT	32	INT_B
J22	PHY_RESET	36	RESET_B
N15	PHY_CRS	115	CRS
M16	PHY_COL	114	COL
P20	PHY_RXCLK	7	RXCLK
U20	PHY_RXER	8	RXER
T22	PHY_RXCTL_RXDV	4	RXDV
P19	PHY_RXD0	3	RXD0
Y22	PHY_RXD1	128	RXD1
Y21	PHY_RXD2	126	RXD2
W22	PHY_RXD3	125	RXD3
W20	PHY_RXD4	124	RXD4
V22	PHY_RXD5	123	RXD5
V21	PHY_RXD6	121	RXD6

Figure 46-Ethernet PHY Connections

U1 FPGA Pin	Schematic Net Name	U46 M88E111	
		Pin Number	Pin Name
U22	PHY_RXD7	120	RXD7
AB7	PHY_TXC_GTPCLK	14	GTXCLK
L20	PHY_TXCLK	10	TXCLK
U8	PHY_TXER	13	TXER
T8	PHY_TXCTL_TXEN	16	TXEN
U10	PHY_TXD0	18	TXD0
T10	PHY_TXD1	19	TXD1
AB8	PHY_TXD2	20	TXD2
AA8	PHY_TXD3	24	TXD3
AB9	PHY_TXD4	25	TXD4
Y9	PHY_TXD5	26	TXD5
Y12	PHY_TXD6	28	TXD6
W12	PHY_TXD7	29	TXD7

Figure 47-Ethernet PHY Connections (Cont'd)

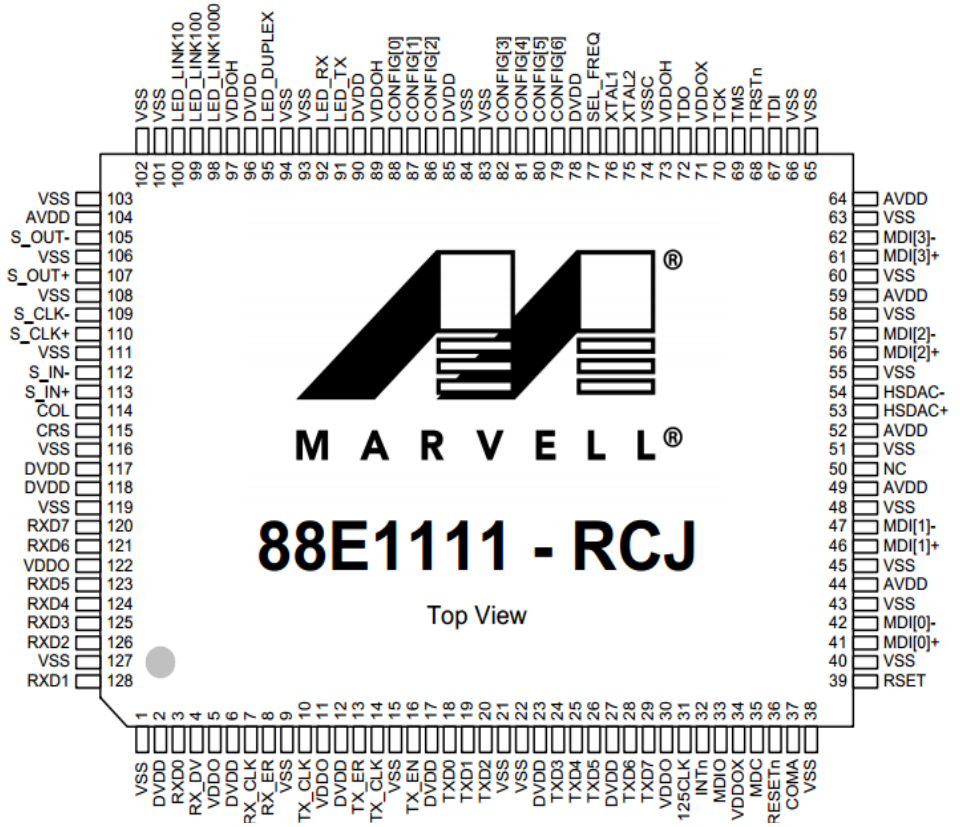


Figure 48-(88E1111) Device 128-pin package from its data sheet

The Ethernet uses two types of main interfaces, Media Independent Interface (**MII**) which is the standard interface used to connect a fast Ethernet it is standardized by **IEEE 802.3u** it uses clock 25 MHZ and 4 data lines to achieve (100Mbit/s), the other interface is the Reduced Media Independent Interface(**RGMII**), it uses 2 data lines TXD0 and TXD1 for transmitting data, while RXD0 and RXD1 for receiving data, while the Gigabit Media independent interface (**GMII**) uses 125 MHZ with 8 data lines to reach 1Gbit/s. First to operate at Ethernet on the FPGA, IPcore is needed to be generated on ISE. Tri-state Ethernet IPcore is generated as follows.

At first to create a new project on the ISE 14.7 Project Navigator as shown in the next screenshots.

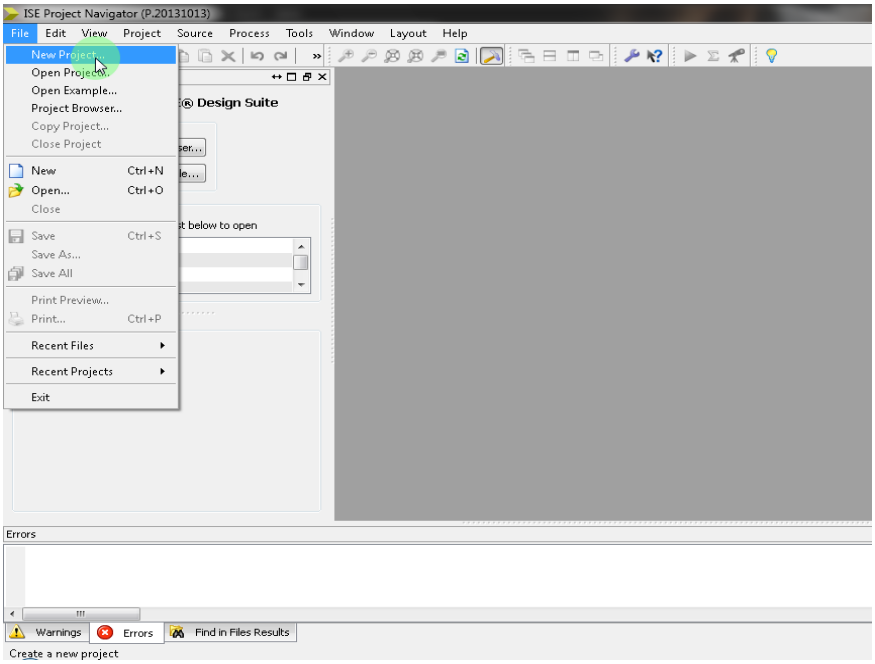


Figure 47-Instantiate the name of the project

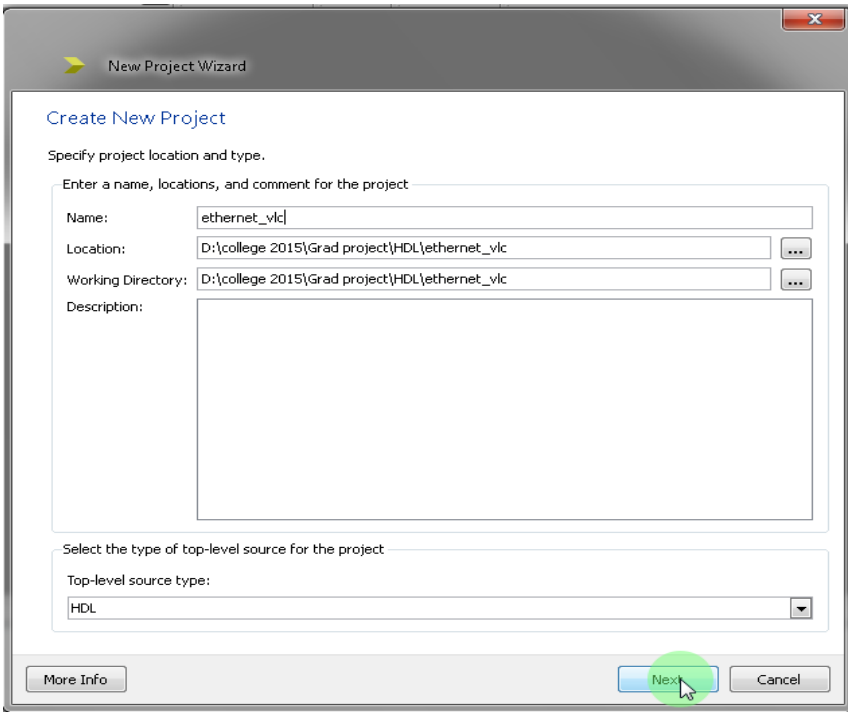


Figure 48-choose the name of the evaluation development board to be Spartan sp605

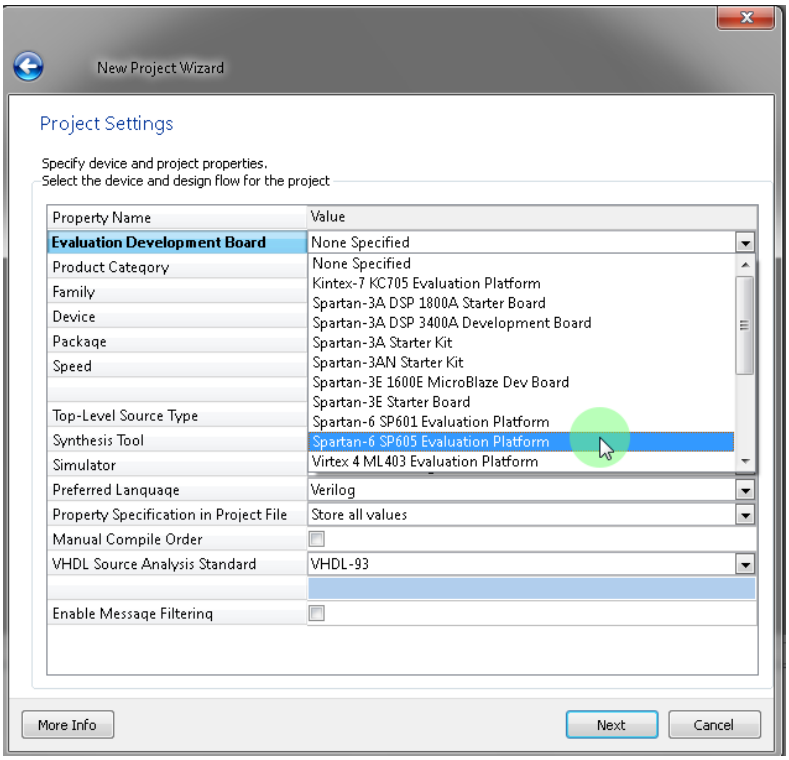


Figure 49-choose new source to add the main programming file to the project

Then in the new source wizard if you are going to use Verilog choose Verilog module to write the main program that you want to use on the FPGA. (See **Fig.50**)

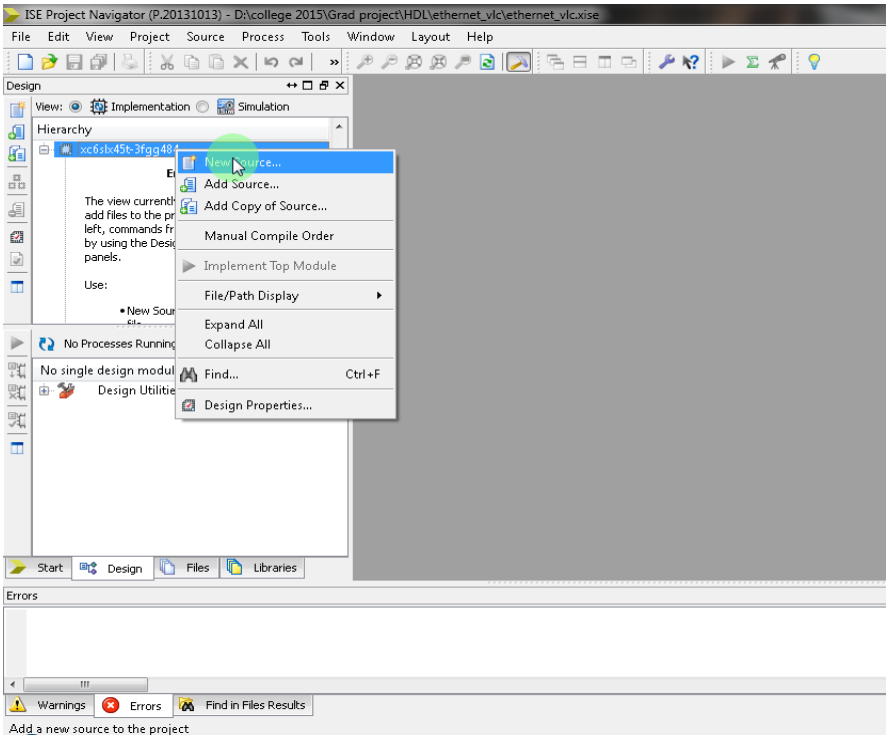
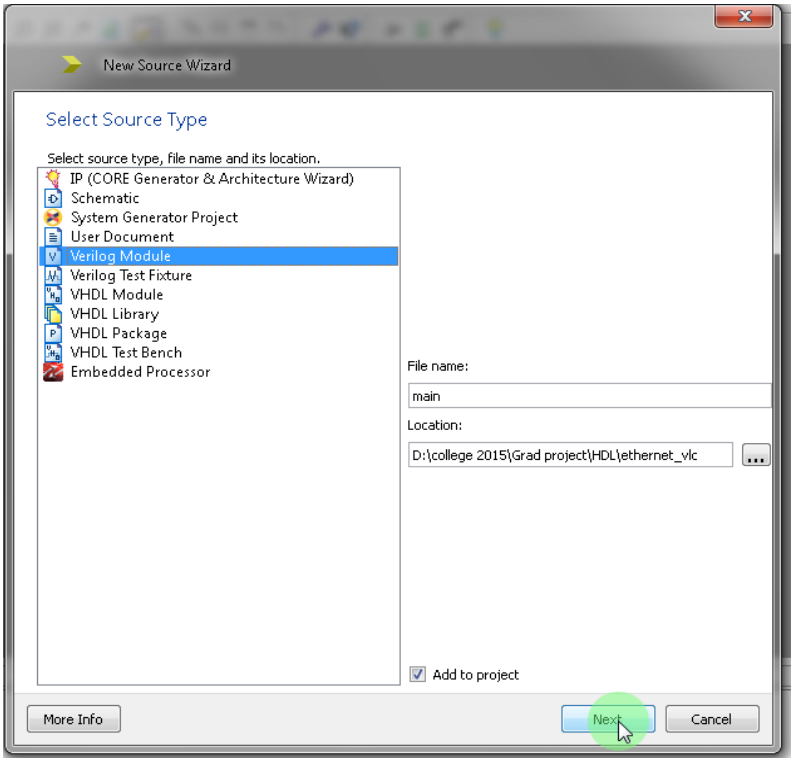


Figure 50- New Wizard



Then the next step is to enter your FPGA inputs and outputs and click next

Add another new source wizard to choose the IPcore needed to be generated then choose communication then Ethernet then Tri-state Ethernet or whatever you need.

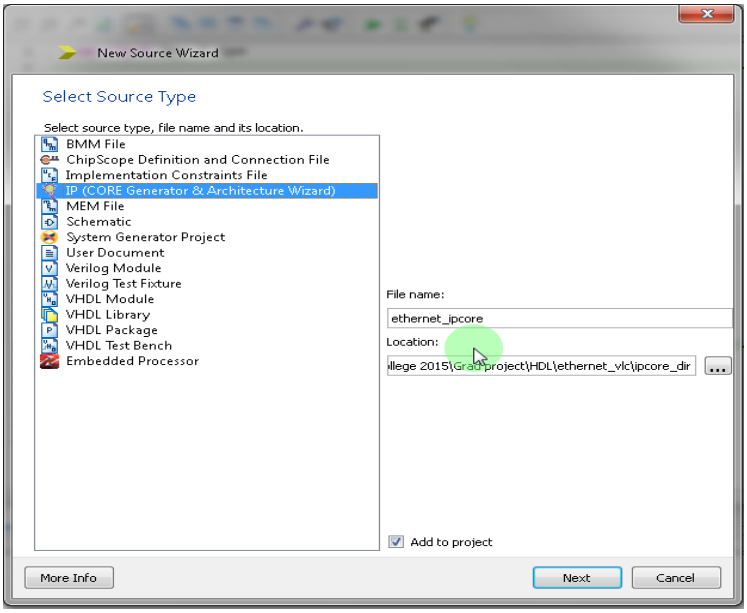


Figure 51-New Wizard

Then choose your configuration you need. Then generate the code like the PHY interface you need the speed you are operating. (See **Fig.52**)

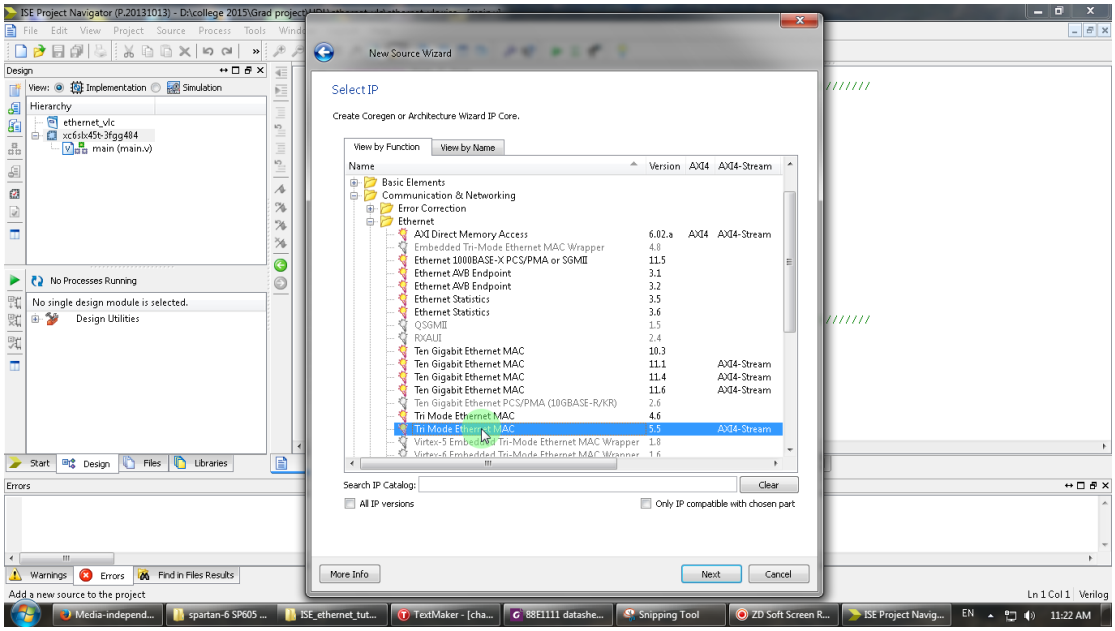


Figure 52- Generating the code

After generating the code click on the file called **Ethernet-IPcore** then go down and choose view HDL instantiation template then copy that template and paste it in your main code, and program the configuration needed to operate the Ethernet protocol. This generated code is attached in the code appendix. (See **Fig.53** and **Fig.54**)

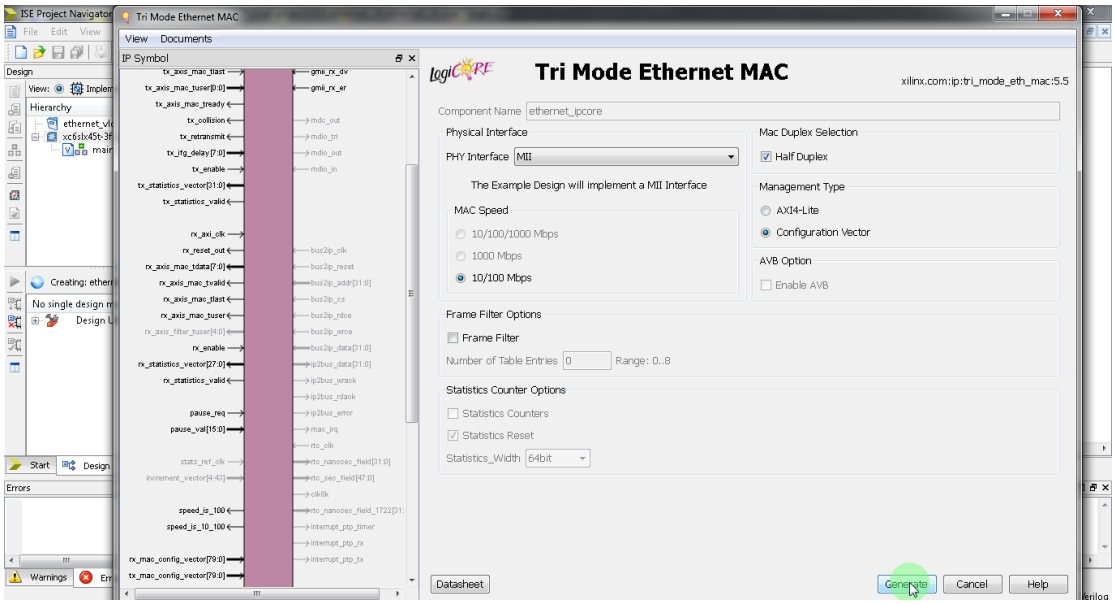


Figure 53-Ethernet IPcore

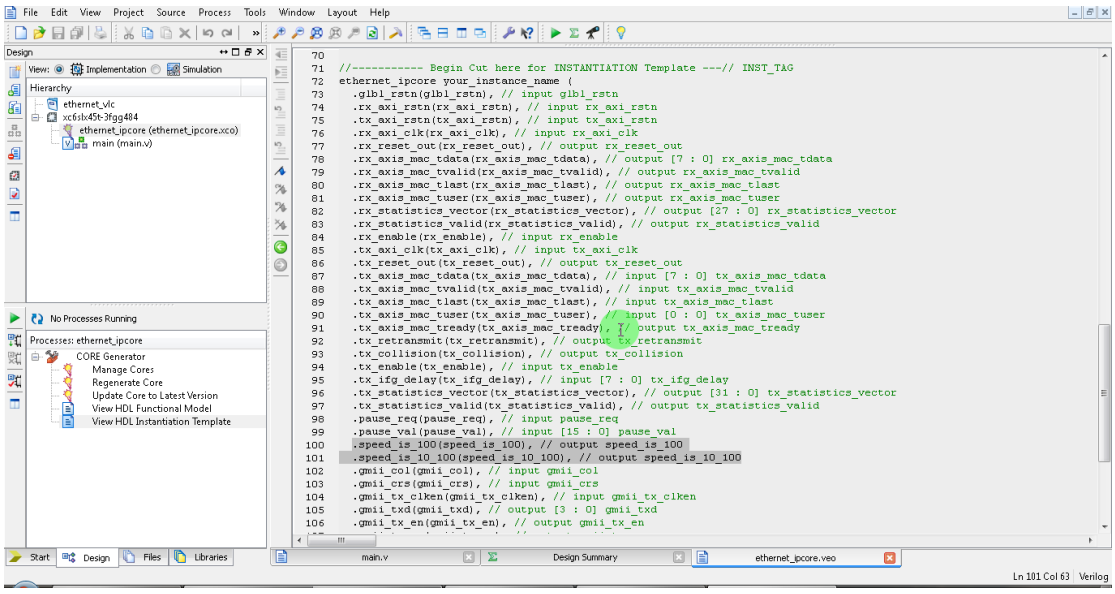


Figure 54- IPcore

These configuration could have taken months, since the FPGA have arrived in the last month, due to time constraints we didn't have the time to program this kind of complex code, so we had to step forward to another possible solution, which were available in the Spartan kit, which is USB-to-UART.

5.2.2 Using USB-to-UART

First Trials:

USB-to-UART was the available alternative for Ethernet, so it was used to test the communication system, FPGA has module for converting easy UART protocol to a USB to be connected to the host computer, and this CP2103GM module of the USB-to-UART then the computer has a virtual COM port which is supported from Silicon Labs as a royalty-free driver. The FPGA is connected to this Module as the following table where **Tx** on CP2103GM is the data output connected to the FPGA so it is an input for the FPGA and the schematic Net name is "**USB_1_TX**" as shown in **Fig.55.** While the **Rx** on CP2103GM is the data input connected to the FPGA so it is an output for the FPGA and the schematic and the schematic Net name is "**USB_1_RX**", there are another two signals called clear to send (**CTS**) and request to send (**RTS**). This is the part of the UART from the perspective of FPGA.

U1 FPGA Pin	UART Function in FPGA	Schematic Net Name	U30 CP2103GM Pin	UART Function in CP2103GM
F18	RTS, output	USB_1_CTS	22	CTS, input
F19	CTS, input	USB_1_RTS	23	RTS, output
B21	TX, data out	USB_1_RX	24	RXD, data in
H17	RX, data in	USB_1_TX	25	TXD, data out

Figure 55 -UART-to-USB pins

The USB has four different wires, signal gnd, +5 voltage supply from the host, the negative-direction and the positive-direction for the bidirectional differential serial data. This is the connection between the host computers as USB. CP2013 handle both protocols

to connect between the FPGA as a UART to the host computer as USB, its schematics is as shown in **Fig.56.**

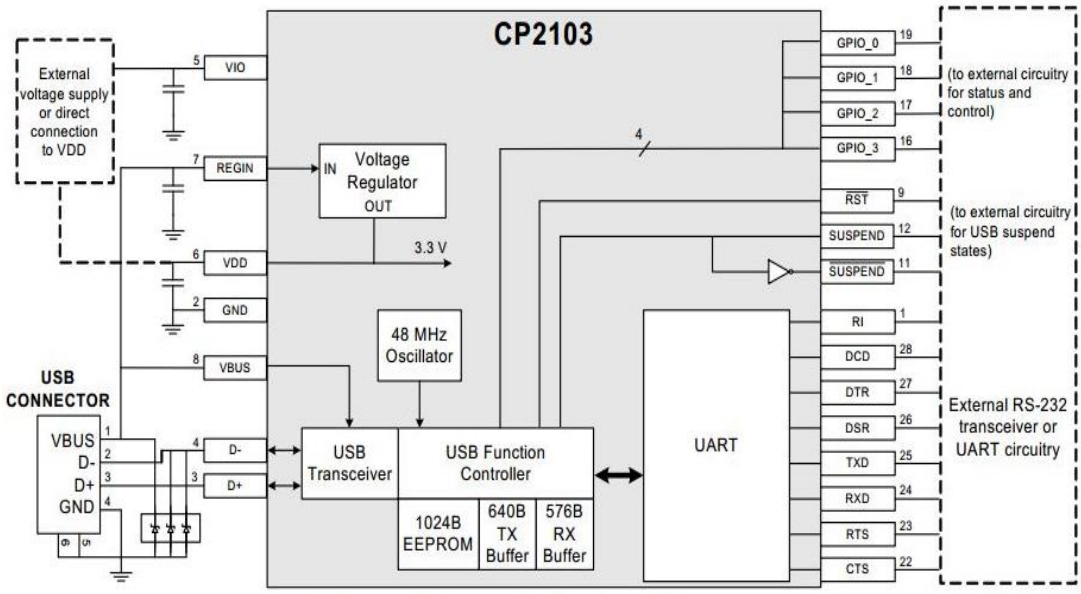


Figure 56- CP2103 schematic

so we are dealing with the UART on the FPGA, we didn't find a driver for the UART written on FPGA, and there was no IPcore for this function to operate, so we started writing our driver to read the data and convert it to a suitable format for our system. The UART mainly consists of some formats we are working with one bit as a start bit, one bit as a stop bit and eight bits of data as shown in **Fig.57**. However, there was a huge shortage in the rest of UART pins they were not connected to the FPGA, we tried to fix the baud rate (bits per second in UART) at 300 bits the code works fine but when trying to increase the baud rate to 9600 bits or 115200 bits; the code doesn't function well, we doubt that the problem without a control signal is as shown in **Fig.57,** where the transmitting rate is a little bit slow so there was a shifting in data that we were not be able to read the data correctly, both codes of the UART at 300 and 9600 baud rate were attached to the code appendix. The algorithm we had tried is like the flow chart shown in **Fig.58.**

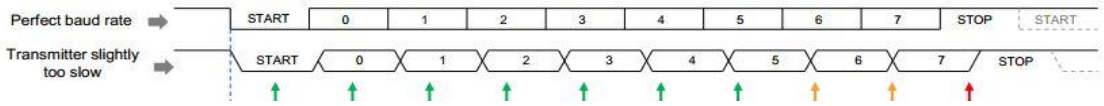


Figure 57-the transmitter is slightly too slow while the sampling is perfect

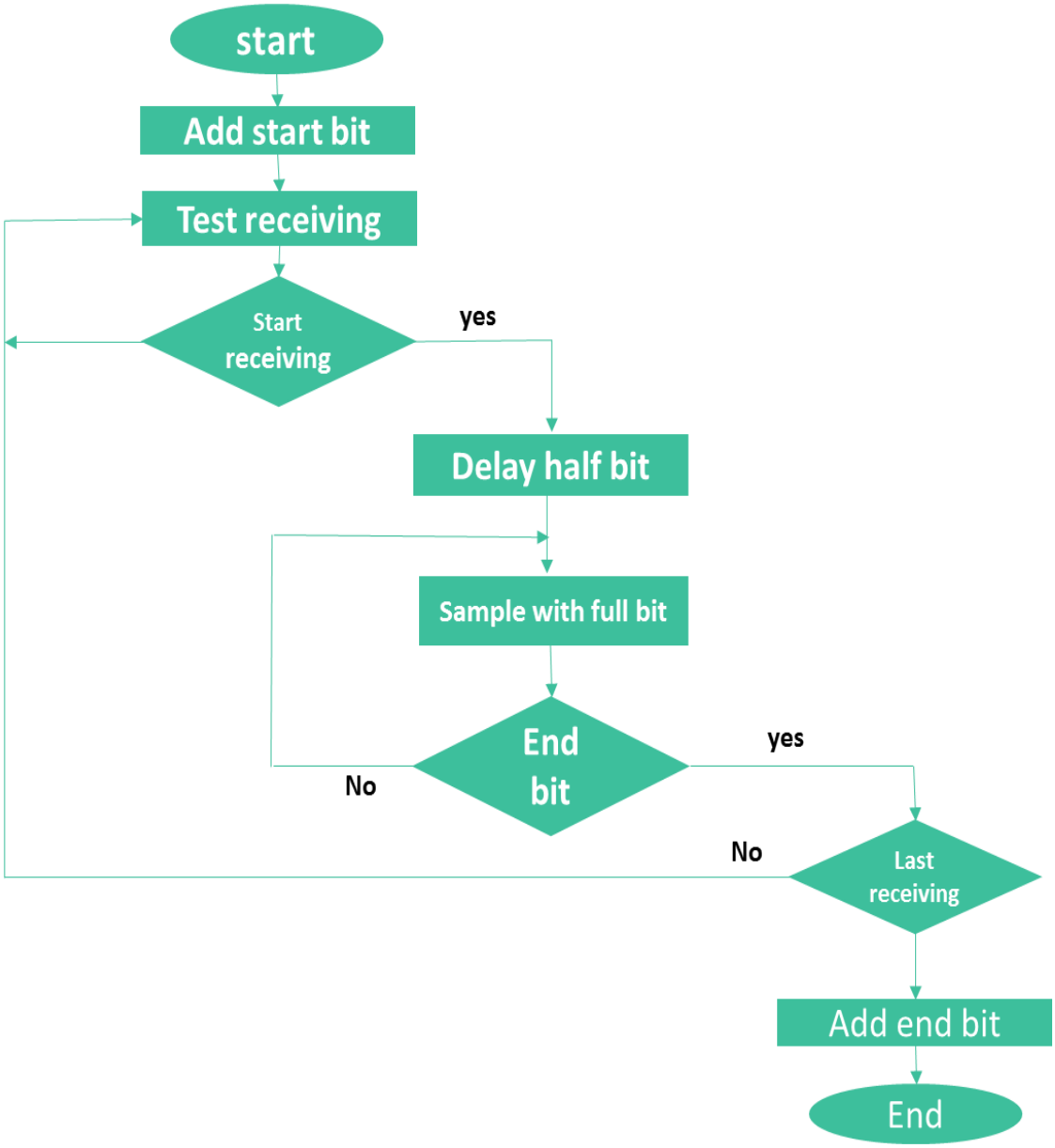


Figure 58-Flow chart for the code

After the failure to be able to sample the data correctly with such an algorithm, we think about another solution to use the UART, this solution must include that there is no data being read, as we fail to read the data. This solution also must include the protocol we are using for our communication system.

Final code:

Before we start to develop a working code, we need to be certain that if we send data from the FPGA using a certain baud rate the computer will be able to read this data. So we have tried to send this data by another code we have developed. This time we succeeded to send a character stored on the FPGA repeatedly at baud rate 9600.

At first, the clock generated by the crystal on the FPGA kit equals 27 MHz which is only divisible by 300 to give 90000 clk for each bit to count, for 9600 it gives 2182.5 clk for every bit so as the data rate increases the number of equivalent clocks on the FPGA decrease, increasing the chance of errors. There were two solutions for this problem; the first we have tried is that we count 2182 clock the first bit and 2183 the next bit and then loop this counter, by this way there won't be errors every 218 character, as we have already compensated for the 0.5 fraction that cause the bits to be synchronized as shown in **Fig.57** above, the clock of the FPGA must be modified to a clock value that is divisible by 9600, which by ISE using an IPcore called **Clock Wizard**, it will be shown how to develop all the program later including the clock wizard.

There is thermal noise and other noises generated by the LED that has all the frequency component this noise is not due to the ambient light at all, it has all components up to 11 MHz, this noise is what is limiting us in the distance that we aim for so it is needed to control this noise and eliminate them at least for the frequency band we are operating at. However, this noise amplified along with the signal and if the data is not being sent then the receiver received a noise data which is acceptable to let this kind of noise without at least recognizing that, so a start bits of 30bit at least and 30bit for end bits is needed to recognize the data that matters, we should keep that in mind.

In order to create a suitable protocol for our communication system we need to define it first using a good representation one of the good representations we have studied is the finite state machine, as shown in **Fig.59,** first we define the first state which is the system is **idle** no data received at this case the LEDs must be turned on all the time waiting for the data to be sent, the variables that control the states are **startflag** which will be one when the host computer send data to the FPGA. and the **endflag** which will be active high when no data is being received by the host computer, the next state is when the start-bit

needs some delay for the data to transmit 30 bits as **startbits** at first, this can be achieved using an array of registers and the data is shifted every clock, the size of the array is delayed for the data by terms of clock. Then starting to transmit the real **data**. After the data received by the stopping **endbits** are needed to be transmitted, then after sending the end bits the system goes again to the **idle** state, before doing this finite state machine, the system was untraceable, there were a lot of errors in operating the code, and we couldn't tell what is going wrong.

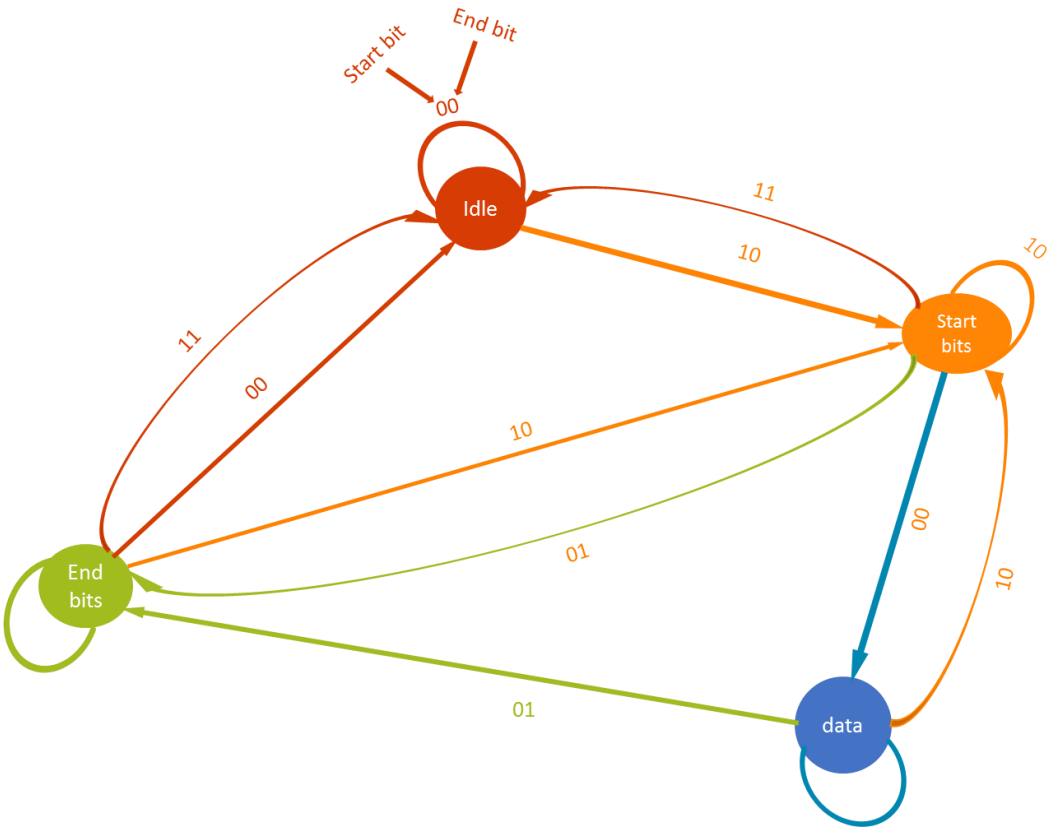


Figure 59- finite state machine for the communication protocol of the system

After we have tried the code written in according to this finite state machine. There was some problems.

The LED was not always on all the time this would cause the LED to turn off during idle state, while turn on when start sending data, so we force the idle signal to be active high in order to be ON all time.

There was another problem the clock we have used in our early time line was being transmitted along for longer than 2 meters was transmitted perfectly, So the FPGA works very well for 2 meters when sending the character 'U' the upper case of the letter 'u', as its ASCI code is "0101010101" this is equivalent to the clock we have worked on in the lab. However, when trying to send data that contains two successive zeros and\or ones which is equivalent to a duty cycle 33%; the noise found in the photodiode modulates the one signal when the distance increased over 10 cm, the output is shown as **Fig.60,** so this problem can be solved using Manchester code this will reduce the noise to minimal, it also can be solved by decreasing the gain of our circuit.

After solving the first problem another one appeared, that when the LEDs are IDLE their intensity was high and when sending data its intensity is lowered, due to time spent when sending zeros this act as pulse width modulation that control the intensity one of the purposed solutions was to modulate the signal in Manchester code so the circuit is always transmitting high and low but we have tried again to capture the data then transfer it to Manchester coding, but lots of errors appeared, so this problem was currently needed to be optimized.

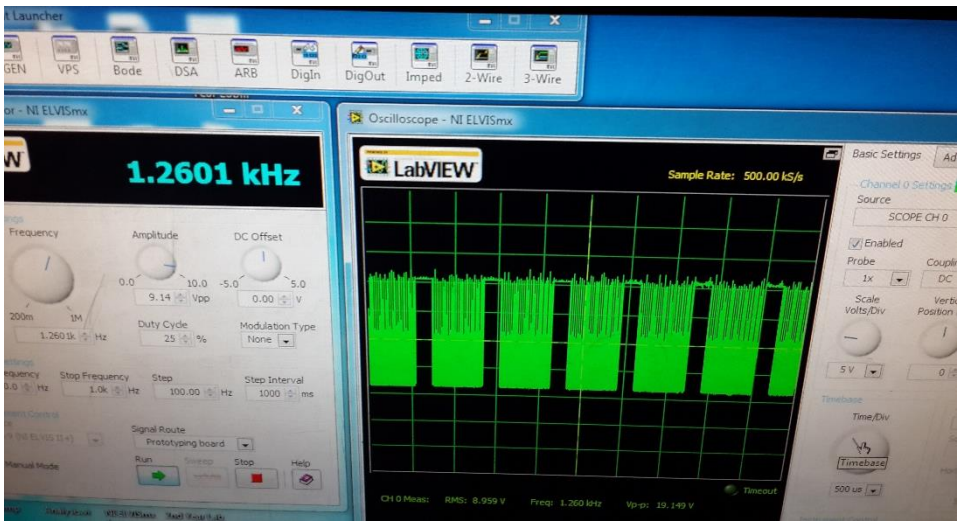


Figure 60-sensor is far away from the led, the gain was very high, so no data received

We have used the Teraterm to transmit data and receive the data after being transmitted through the FPGA then the LED transmitter then the photodiode transmitter using baud rate 9600. (See **Fig.61**)

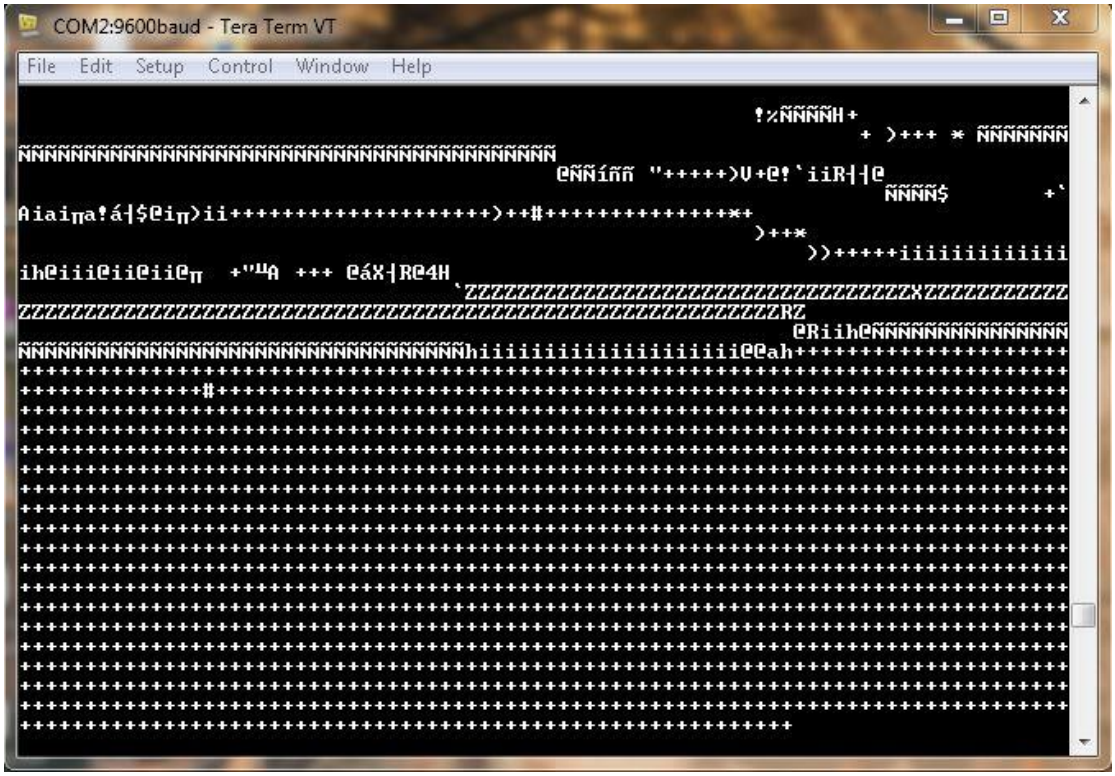


Figure 61-first time using Teraterm to receive and transmit data using serial port in the project

5.3 Final system implementation

In **Fig.62**, after attaching the whole system parts together and sending data through the computer using the Teraterm software. This data is passed to the transmitting circuit through the GPIO pin on the evaluation kit, then data is transmitted using the VLC to the receiving circuit, then to the FPGA again through another GPIO pin to be monitored on the computer.

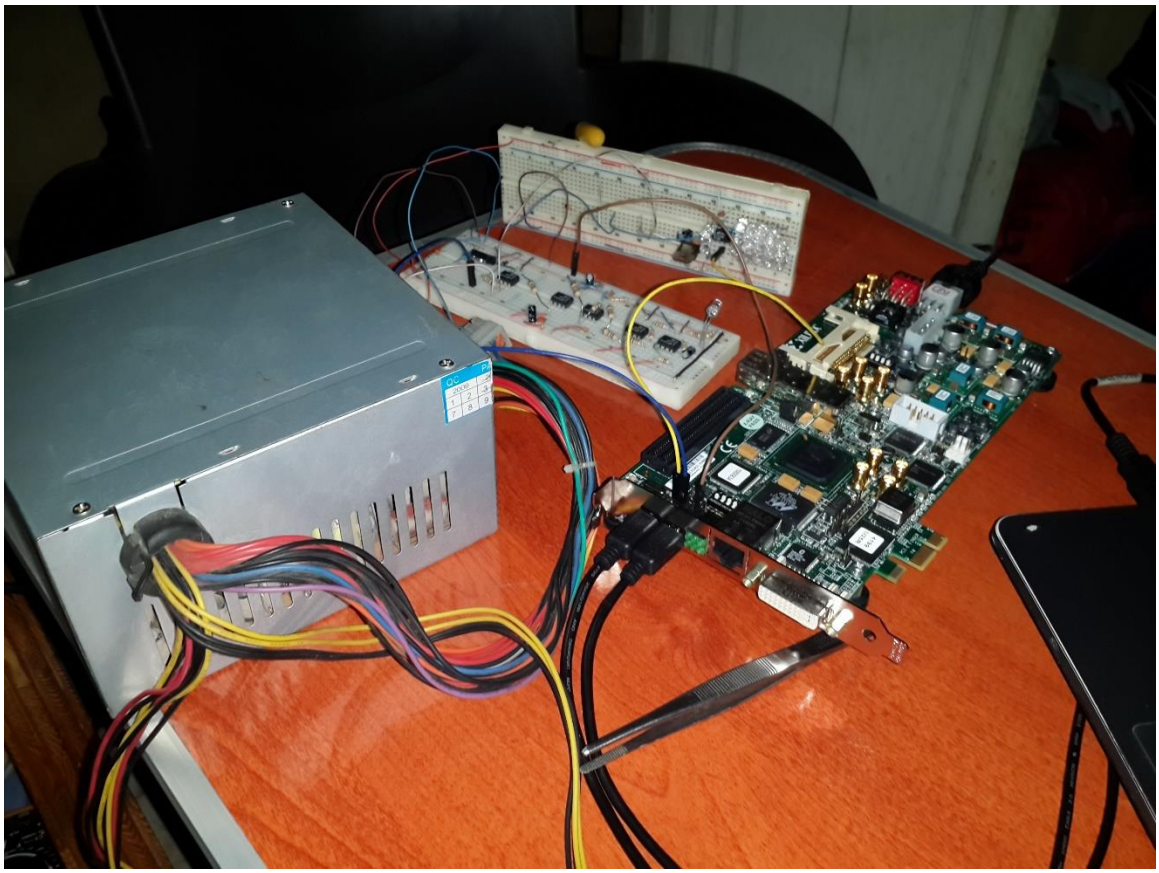


Figure 62- Final system

5.4 ISE and coding the FPGA

First you setup up the FPGA configuration Mode switches and the System ACE Mode Selections to be ready to burn the code this can be found in the Hardware setup guide and the Hardware user guide.

You must also check every jumper's configuration then after installing ISE 14.7 and creating a new project then clicking a new.

For the setting the desired clock that your work will be based on; add the IPcore of the **clock wizard** then input the clock crystal frequency embedded on the FPGA and choose other option as shown below in **Fig.63.**

Figure 63-clock wizard

Then click next and add the output clock you need to be generated by the clock wizard and turn off the rest of the output clock. (See **Fig.64**)

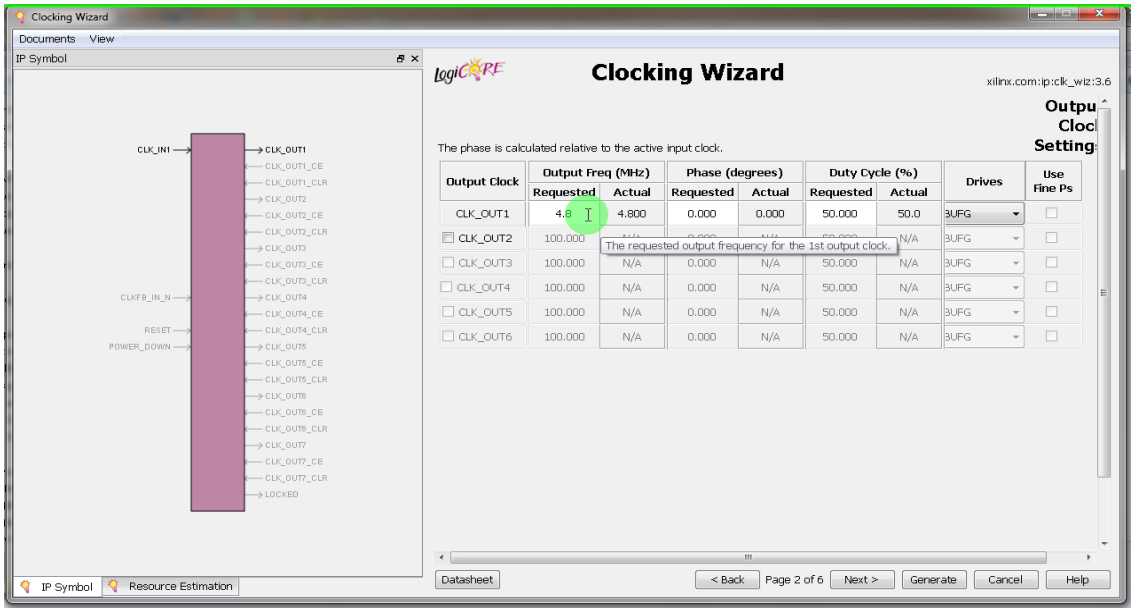


Figure 64-clock wizard

Then uncheck the reset and looked and finally click Generate.

Then go to the generated clock wizard file (.xco) instance and choose View HDL instantiation Template and select the part shown below and copy it into your code the variable between the packets is the variable you would rename by yourself and use it in your code. (See **Fig.65**)

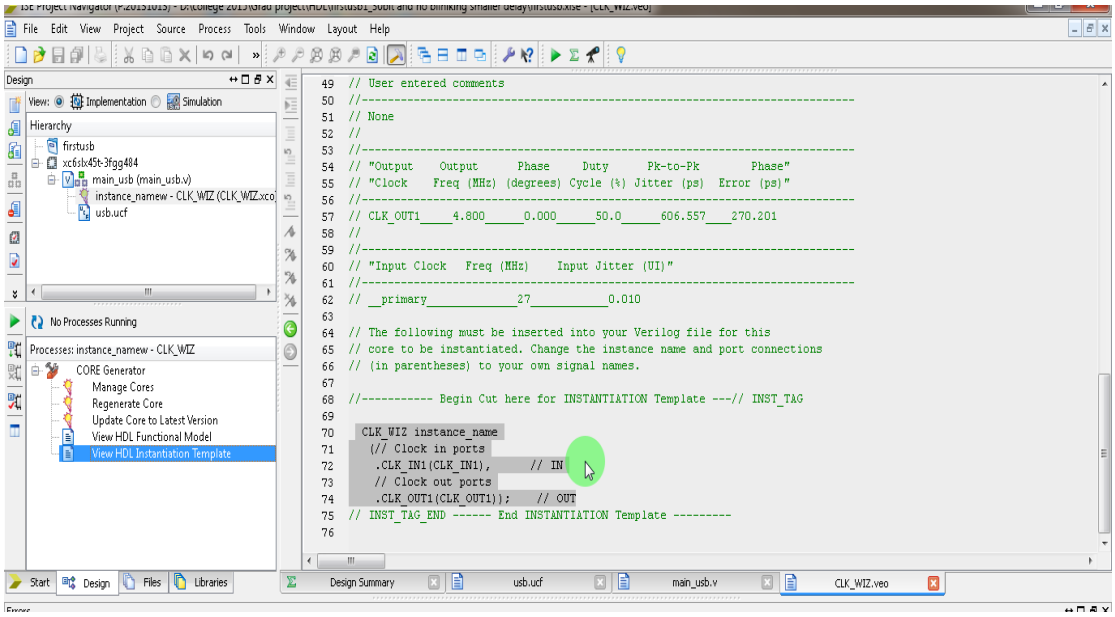


Figure 65- (.XCO) file

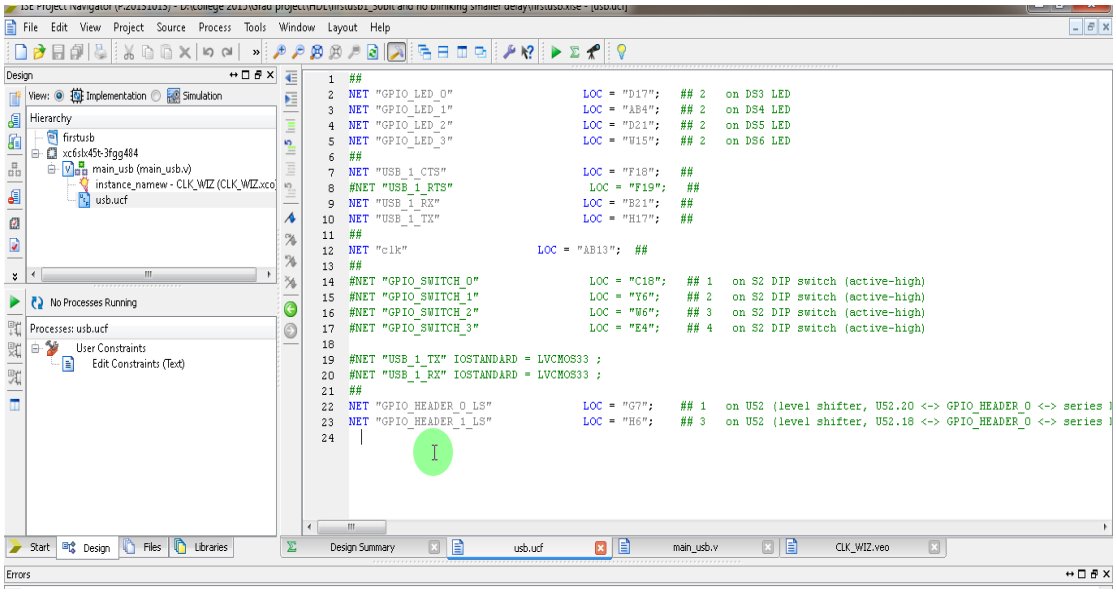


Figure 66- (.ucf) file

Then you must define your pin that you have used in(.ucf) file by adding a new source file as shown before then choose Implementation Constraints File and then get the pin name and assignment from the file called "**SP605_Master_UCF**" on the flash drive of the FPGA you must define all the inputs and outputs of the FPGA. (See **Fig.66**)

Then finally you make sure you choose view Implementation then select the main file "**main_usb.v**" double click Generate Target PROM file. (See **Fig.67**)

Then click ok after generating the main_usb.it file as shown below

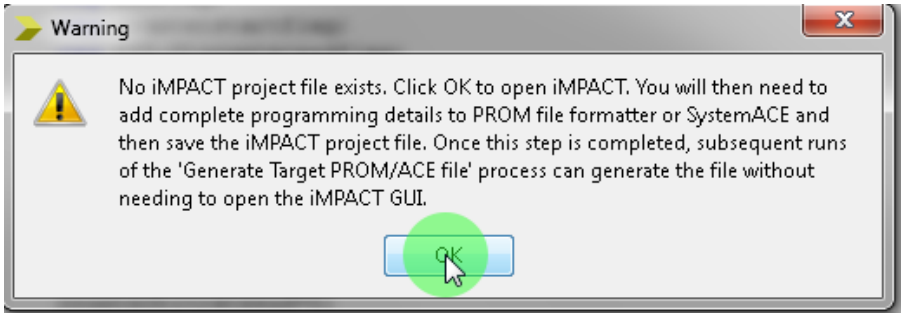


Figure 66

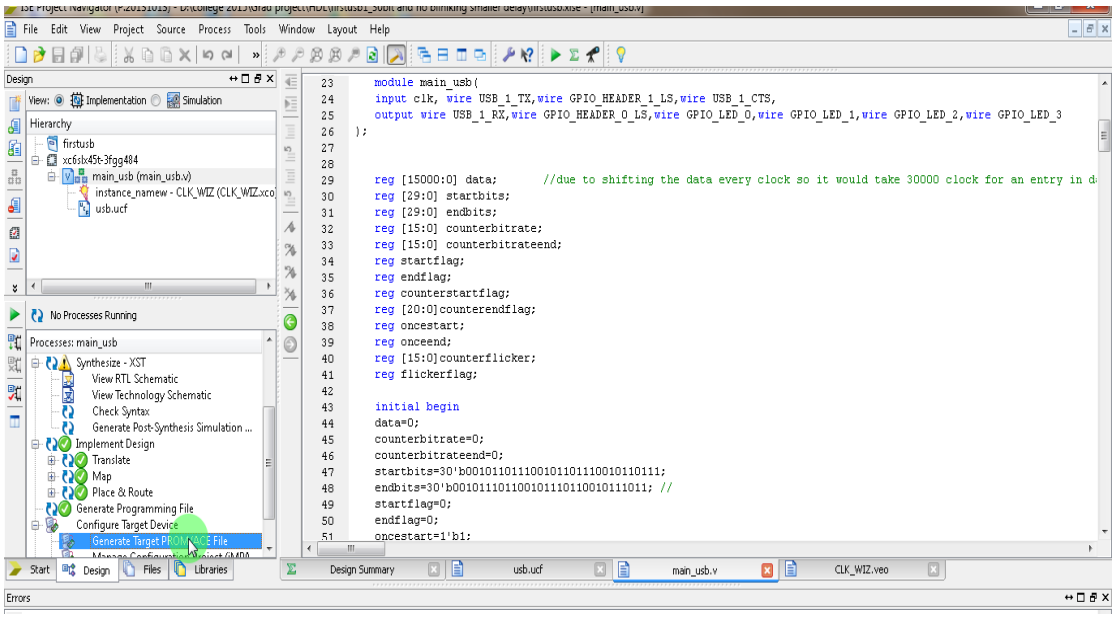


Figure 65- (.v) file

After clicking ok, make sure that **USB_JTAG** cable is connected to the computer in order to be able to read the FPGA and also turn the FPGA power switch on to be ready to burn the (.bit) file.

You must also know that the memory FPGA is volatile as the power goes OFF, the programmed file is erased from the FPGA. After that **ISE IMPACT** window opens automatically, double click on Boundary scan, to scan the connected FPGA. Then right click on the white background and choose initialize chain. (See **Fig.69**)

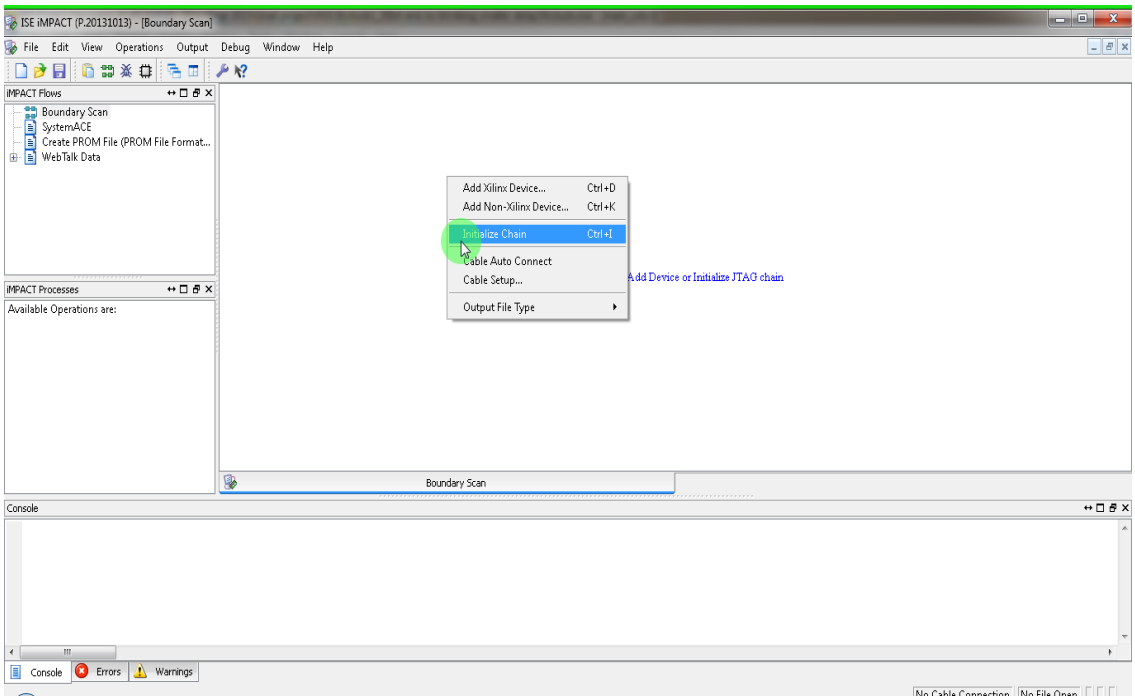


Figure 67- initializing chain

Then click ok now the FPGA is identified successfully. To assign the (.bit) file right; click on the FPGA IC then choose Assign new configuration file and choose "**main_usb.bit**" file to be burned. (See **Fig.70**)

After assigning the configuration file; right click on the FPGA IC and click program or choose program from the side toolbar. (See **Fig.71**)

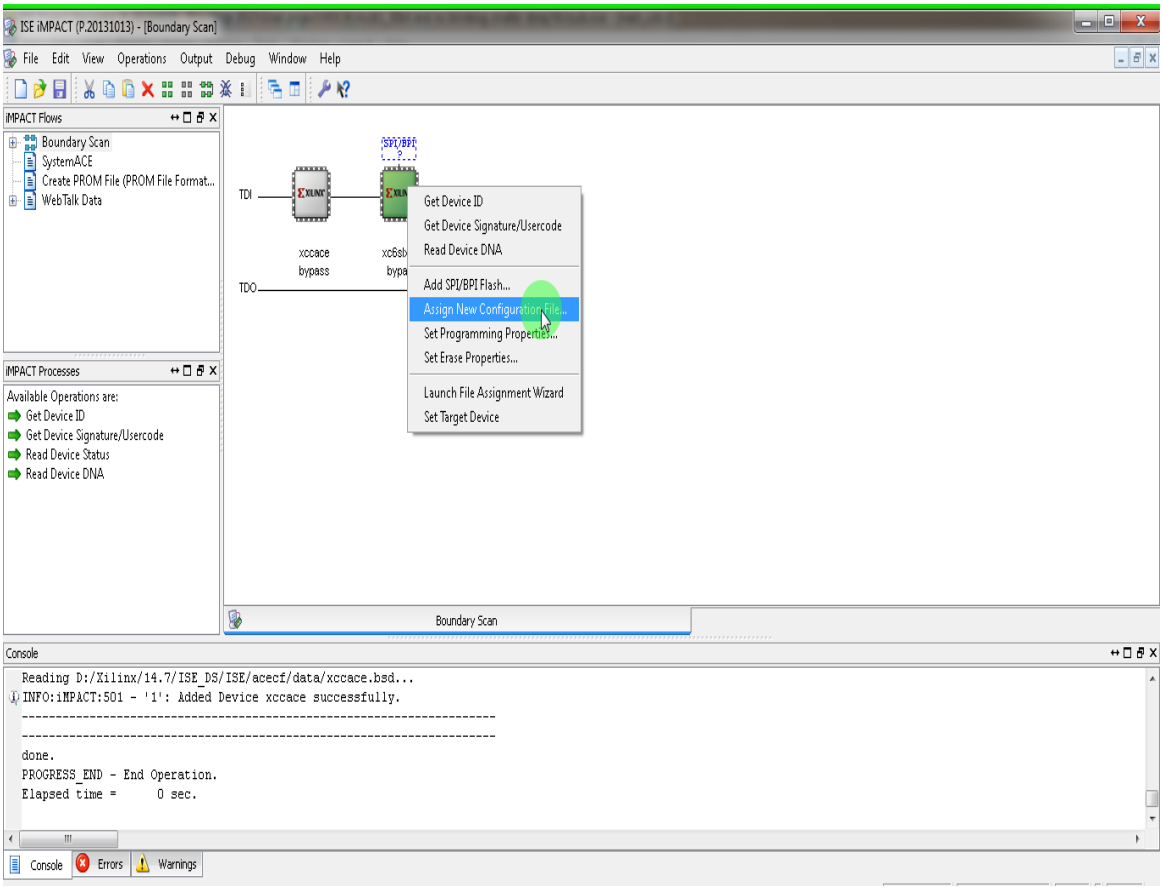


Figure 68- (.bit) file

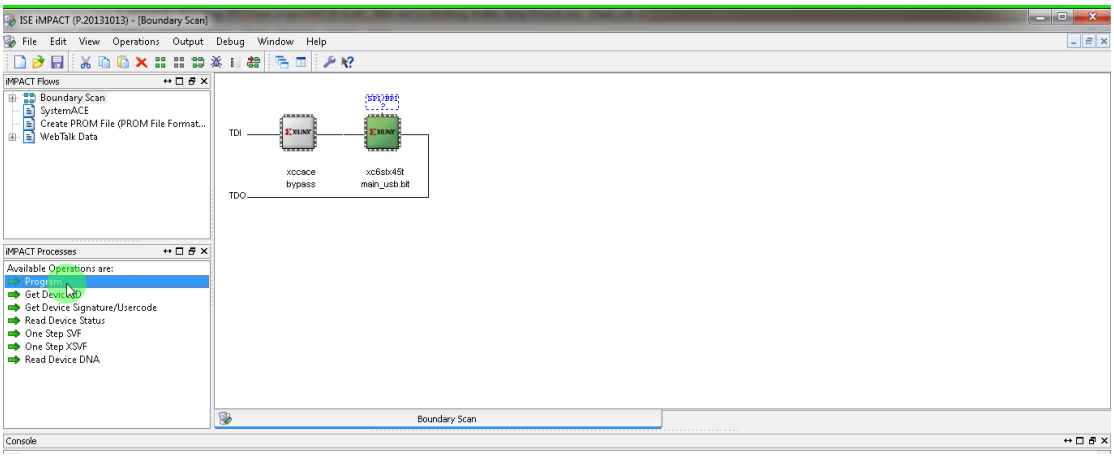


Figure 69- burning on FPGA

Then a message "**program Succeeded**" is displayed on the screen. (See **Fig.72**)

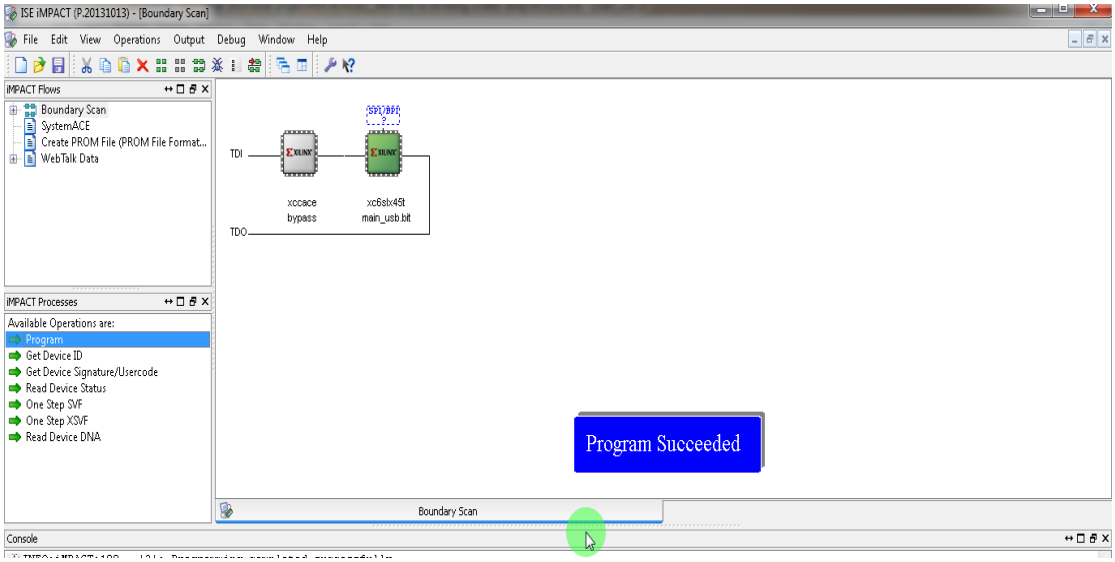
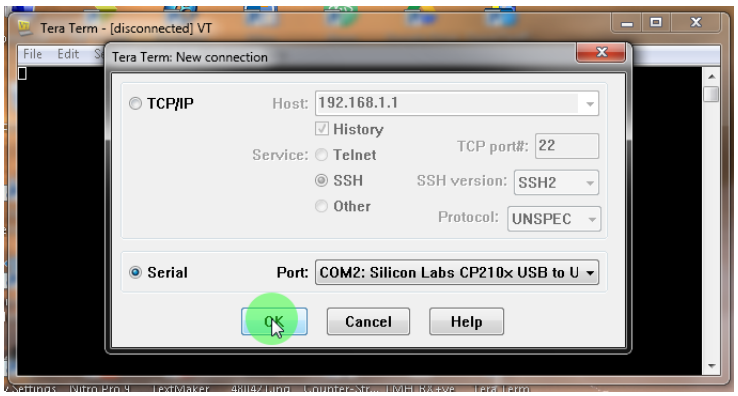
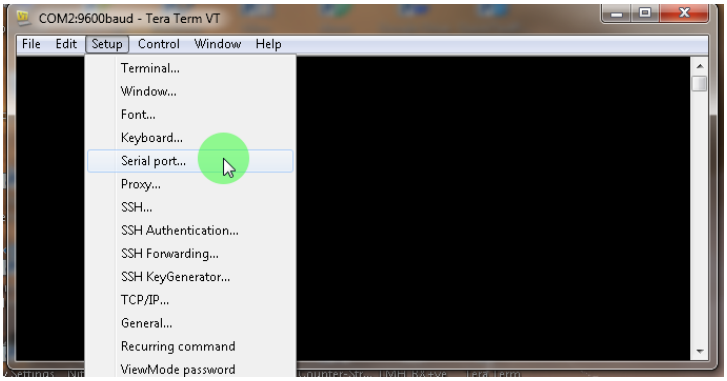
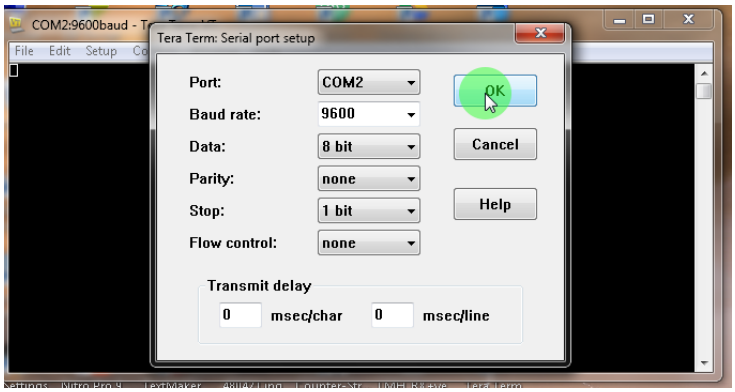


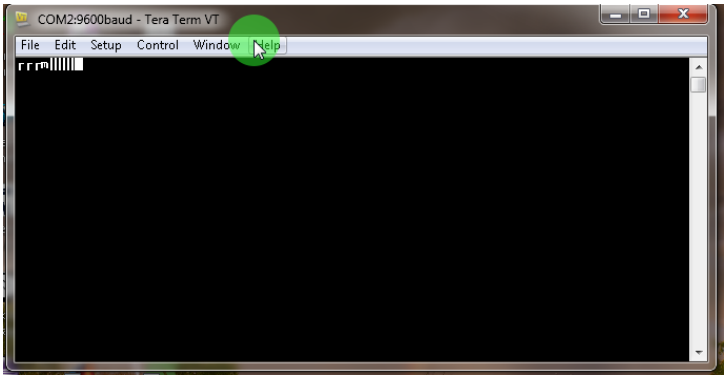
Figure 70- program succeeded

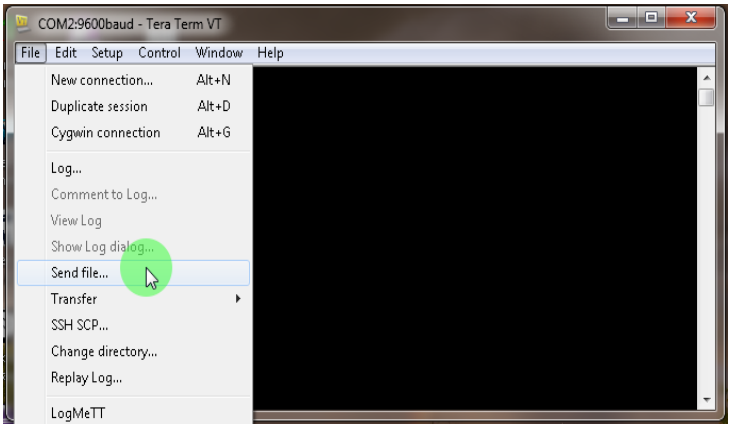
After burning the program, install CP210x driver for the virtual port com. then choose the parameter you need to operate at UART. From device manager choose ports and select a name for the com port and choose the baud rate as below. Then open the TERTERM to monitor the serial port and check serial. Note that FPGA must be connected to the FPGA and to computer.

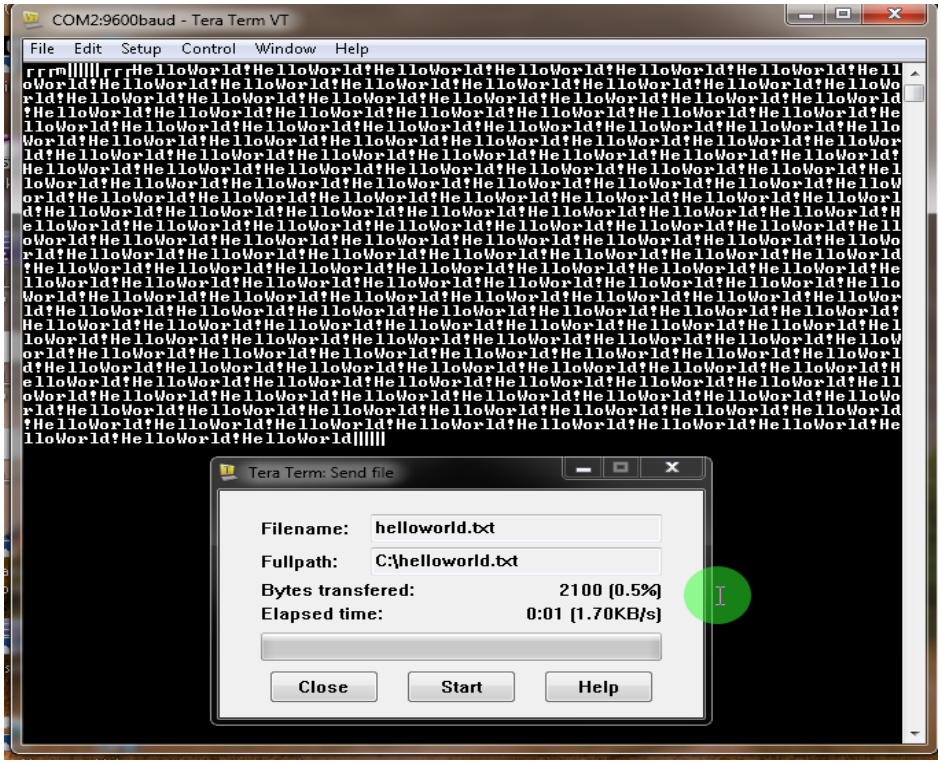



COM2:9600baud - Tera Term VT
File Edit Setup Control Window Help
Terminal...
Window...
Font...
Keyboard...
Serial port...
Proxy...
SSH...
SSH Authentication...
SSH Forwarding...
SSH KeyGenerator...
TCP/IP...
General...
Recurring command
ViewMode password


Tera Term: Serial port setup
Port: COM2
Baud rate: 9600
Data: 8 bit
Parity: none
Stop: 1 bit
Flow control: none
OK
Cancel
Help
Transmit delay
0 msec/char
0 msec/line


COM2:9600baud - Tera Term VT
File Edit Setup Control Window Help


COM2:9600baud - Tera Term VT
File Edit Setup Control Window Help
New connection... Alt+N
Duplicate session Alt+D
Cygwin connection Alt+G
Log...
Comment to Log...
View Log
Show Log dialog...
Send file...
Transfer
SSH SCP...
Change directory...
Replay Log...
LogMeTT



The file has been sent startbits and endbits are as shown on the terminal.

References

[1] G. Holzmann and B. Pehrson, "The Early History Of Data Networks," ed, 1994, p. 292.

[2] A. G. Bell, "Selenium and the Photophone," in *Nature* vol. 22, ed, 1880, pp. 500--503.

[3] http://www.tra.gov.eg/english/DPages_DPagesDetails.asp?ID=138&menu=1

[4] https://en.wikipedia.org/wiki/Frequency_allocation

[5] http://en.wikipedia.org/wiki/Mobile_phones_on_aircraft#The_debate_on_safety

[6] Demestichas, P.; Vivier, G.; El-Khazen, K.; Theologou, M. "Evolution in wireless systems management concepts: from composite radio environments to reconfigurability", Communications Magazine, IEEE, on page(s): 90 - 98 Volume: 42, Issue: 5, May 2004

[7] Yiping Xing; Mathur, C.N.; Haleem, M.A.; Chandramouli, R.; Subbalakshmi, K.P. "Real-time secondary spectrum sharing with QoS provisioning", Consumer Communications and Networking Conference, 2006. CCNC 2006. 3rd IEEE, on page(s): 630 - 634 Volume: 1, 8-10 Jan. 2006

[8] Baschirotto, A.; Castello, R.; Campi, F.; Cesura, G.; Toma, M.; Guerrieri, R.; Lodi, A.; Lavagno, L.; Malcovati, P. "Baseband analog front-end and digital back-end for reconfigurable multi-standard terminals", Circuits and Systems Magazine, IEEE, On page(s): 8 - 28 Volume: 6, Issue: 1, First Quarter 2006

[9] Roberts, M.L.; Temple, Michael A.; Mills, R.F.; Raines, R.A. "Evolution of the air interface of cellular communications systems toward 4G realization", Communications Surveys & Tutorials, IEEE, On page(s): 2 - 23 Volume: 8, Issue: 1, First Quarter 2006

[10]http://www.bakom.admin.ch/dokumentation/zahlen/00545/00547/00554/index.html?lang=en&download=NHzLpZeg7t,lnp6I0NTU042l2Z6ln1ad1IZn4Z2qZpnO2Yuq2Z6gpJCDdH18gGym162epYbg2c_JjKbNoKSn6A—

[11] http://lasercommunications.weebly.com/index.html

[12] http://www.sahyadri.edu.in/e-journal/laser.pdf

[13] Winburn D.C., „Practical Laser Safety. Occupational Safety and Health“

Marcel Dekker, Inc. 1985

[14] http://www.trefis.com/stock/cree/articles/173710/why-cree-will-continue-growing-its-led-market-share/2013-03-21

[15] S. Schmid, G. Corbellini, S. Mangold, and T. Gross. An LED-to-LED Visible Light Communication system with software-based synchronization. In Optical Wireless Communication. Globecom Workshops (GC Wkshps), 2012 IEEE, pages 1264–1268, Dec. 2012.

[16] http://csep10.phys.utk.edu/astr162/lect/light/spectrum.html

[17] http://www.fcc.gov/what-we-do

Appendix

MATLAB code

```
%distrubution of illumination indoor vlc assuming plane of reciver is
at
%a table 0.83 m from ground.
clear all; clc; close all;
[length,width] = meshgrid(-7.5:0.05:7.5);
height = 3;
q=1.602176487E-19;
d = height-0.83;                    %distance btw Rx Tx
psi = 30;      %Semi-angle at half power 30
Io=7232;       % perpindicular illimunation power
%calculating distance btwn TX & RX
temp_theta=sqrt((length.^2)+(width.^2)+d^2);
theta=acos((d^2)./(d.*temp_theta));
dd=sqrt((length.^2)+(width.^2))./sin(theta);
Ehora=Io*cos(pi/3)./((dd.^2).*cos(pi/6)); %Horizontal power
%%
Ehorb=Ehora;
%%
%summing all the Results of dc
Ehor=Ehora;

%b(5,0)
  for i = 1:size(length)-101
       for j = 1:size(width)

           Ehor(i,j)=Ehor(i,j)+Ehorb(i+101,j);

       end
  end

  %d(-5,0)
    for i = 102:size(length)
       for j = 1:size(width)

           Ehor(i,j)=Ehor(i,j)+Ehorb(i-101,j);

       end
    end

    %c(0,5)
    for i = 1:size(length)
       for j = 102:size(width)

           Ehor(i,j)=Ehor(i,j)+Ehorb(i,j-101);
```

```
        end
    end

    %(0,-5)
     for i = 1:size(length)
       for j = 1:size(width)-101

           Ehor(i,j)=Ehor(i,j)+Ehorb(i,j+101);

       end
     end
mesh(length,width,Ehor);
xlabel('width');
ylabel('length');
title('horizontal illumination of 5 sources at 0.83 hight');
%%
%calclating ac value
Io_depth=Io*0.5;
temp_theta=sqrt((length.^2)+(width.^2)+d^2);
theta=acos((d^2)./(d.*temp_theta));
dd=sqrt((length.^2)+(width.^2))./sin(theta);
Ehor_AC=Io_depth*cos(pi/3)./((dd.^2).*cos(pi/6)); %Horizontal power
SIR=Ehor_AC./Ehor;for i = 1:size(length)
       for j = 1:size(width)
           if(SIR(i,j)==1)
           SIR(i,j)=0.14;
           end
       end
     end

figure;
mesh(length,width,SIR);
xlabel('width');
ylabel('length');
title('SIR 5 sources at 0.83 hight');
figure;
mesh(length,width,Ehora);
xlabel('width');
ylabel('length');
title('illumination of one source');
figure;
mesh(length,width,Ehor_AC);
xlabel('width');
ylabel('length');
title('depth of half the power');
```

Verilog code

Sending only one character generated by FPGA to a host computer

```
`timescale 1ns / 1ps
module main_usb(
    input clk,USB_1_TX,
    output reg USB_1_RX
 );

reg [20:0] temp;
reg [3:0] sendbit;

initial begin
temp=0;
sendbit=0;
end

always @(posedge clk) //send data baud rate 115200
begin
if (temp==2812)begin
temp=0;
 case (sendbit)
      4'b0000: begin
                  USB_1_RX<=0;
               end
      4'b0001: begin
                  USB_1_RX<=1;
               end
      4'b0010: begin
                  USB_1_RX<=0;
               end
      4'b0011: begin
                  USB_1_RX<=0;
               end
      4'b0100: begin
                  USB_1_RX<=0;
               end
      4'b0101: begin
                  USB_1_RX<=0;
               end
      4'b0110: begin
                  USB_1_RX<=0;
               end
      4'b0111: begin
                  USB_1_RX<=1;
               end
      4'b1000: begin
                  USB_1_RX<=0;
               end
      4'b1001: begin
```

```
                USB_1_RX<=1;
              end

      default: begin
                        sendbit=0;
            end
   endcase
        sendbit=sendbit+1;
end
else
temp=temp+1;
end

endmodule
```

300 baud rate code

```
`timescale 1ns / 1ps
module main_usb(
    input clk,wire USB_1_TX,
    output reg USB_1_RX
 );


reg [1024:0] data=0;
reg [10:0] count;
reg [10:0] count1;
reg [40:0] temp;  //wait 30 secend and start sending
reg [30:0] temp2;
reg [30:0] temp3;
reg [30:0] temp4;
reg [30:0] temp5;
reg Rx_enable;
reg Tx_enable;

initial begin
count=0;
count1=0;
Rx_enable=0;
Tx_enable=0;
temp=45000;
temp2=0; //flag for baud rate Rx
temp3=0;//flag of Rx
temp4=0;//flag for first delay in Rx
temp5=0;
end

//Rx enable
always @(posedge clk)
begin
 if (Tx_enable==0)begin
 Rx_enable=1;
 temp3=0;
 end
```

```
 if (temp3==810000)begin   //90000*9bit
  Rx_enable=0;
  temp3=0;
  temp4=0; //for the first delay to be reset
end else
temp3=temp3+1;
end



//Rx
always @(posedge clk) //300 baud rate
begin
if(Rx_enable)begin
    if (temp4==45000)begin

       if (temp2==90000)begin
       temp2=0;
       data[count]=USB_1_TX;
         count=count+1;
       end
       else
       temp2=temp2+1;
     end
      else
      temp4=temp4+1;
end

end


//Tx baud rate 300
always @(posedge clk)
begin
 if (temp==900000000)begin

     if (temp5==90000)begin
      temp5=0;
        USB_1_RX<=data[count1];
        count1=count1+1;
    end
    else
    temp5=temp5+1;

 end
  else begin
  temp=temp+1;
  end
end

endmodule
```

9600 and 115200 baud rate code

```
`timescale 1ns / 1ps
module main_usb(
    input clk,wire USB_1_TX,
    output reg USB_1_RX
 );

reg [1024:0] data;
reg [10:0] datacounterrx;
reg [10:0] datacountertx;
reg startbitflag;                 //indicate that start bit exists
during the current symbol..this is the key to sample the data
reg [4:0]currentbitrecived;     //pointer to the current bit
received...indicate that 8 bits is done

//reg [12:0] halfbitequivelantbaud=1406;
//reg [12:0] bitequivelantbaud=2812;      //27000000%9600=2812.5
reg [20:0] halfbitequivelantbaud=45000;
reg [20:0] bitequivelantbaud=90000;      //27000000%300=90000
reg [20:0] counthalfstart;
reg [20:0] counterbitequivelantbaudrx;
reg [20:0] counterbitequivelantbaudrxend;
reg [20:0] counterbitequivelantbaudtx;


initial begin
data=0;
datacountertx=0;
datacounterrx=0;
startbitflag=0;
currentbitrecived=0;
counterbitequivelantbaudrx=0;
counterbitequivelantbaudrxend=0;
counterbitequivelantbaudtx=0;
counthalfstart=0;
end


//receiving
always @(posedge clk) //300 baud rate
begin
// start bit receiving
    if(USB_1_TX==0 && startbitflag==0)begin  //----------->
       if (counthalfstart==halfbitequivelantbaud)begin
         data[datacounterrx]<=USB_1_TX; //save start bit
         datacounterrx<=datacounterrx+1;

         startbitflag<=1;        ///////////////must be held to zero
after 8bits...even during the end bit[done1]
```

```
          counthalfstart<=0;
        end
        else
          counthalfstart<=counthalfstart+1;
     end  else
     counthalfstart<=0;
     //-----------<


   //end bit test
       if ( startbitflag==1 && currentbitrecived==8)begin  //----------
->
              if(counterbitequivelantbaudrxend==bitequivelantbaud)begin
                data[datacounterrx]<=USB_1_TX; //save end bit
                datacounterrx<=datacounterrx+1;


                        if(datacounterrx==1020)   //buffering data for
102 bytes
                          datacounterrx<=0;

                startbitflag<=0;                //[done1]
                currentbitrecived<=0;          //[done2]
               counterbitequivelantbaudrxend<=0;
                 end else

counterbitequivelantbaudrxend<=counterbitequivelantbaudrxend+1;
      end  //------------<

//Rx data
       if (startbitflag==1 && currentbitrecived<8)begin   //------>
               if(counterbitequivelantbaudrx==bitequivelantbaud)begin
                 data[datacounterrx]<=USB_1_TX;
                datacounterrx<=datacounterrx+1;
                currentbitrecived<=currentbitrecived+1;
//////////need to be zero at end bit[done2]
               counterbitequivelantbaudrx<=0;
                 end else

counterbitequivelantbaudrx<=counterbitequivelantbaudrx+1;

      end   //------<

end

//Tx
always @(posedge clk) //300 baud rate
begin
   if (startbitflag==0)begin//---------->
            if(datacountertx<datacounterrx)begin    //22
                if(counterbitequivelantbaudtx==bitequivelantbaud)begin
///11
                       USB_1_RX<=data[datacountertx];
                       datacountertx<=datacountertx+1;

                end else      ///11
```

```
counterbitequivelantbaudtx<=counterbitequivelantbaudtx+1;
            end     //22

     end  //---------<
endendmodule
```

Very simple code for testing

```
 `timescale 1ns / 1ps

module main_usb(
    input wire USB_1_TX,wire GPIO_HEADER_1_LS,
    output wire USB_1_RX,wire GPIO_HEADER_0_LS
 );

//assign USB_1_RX=USB_1_TX;
assign GPIO_HEADER_0_LS=USB_1_TX;
assign USB_1_RX=GPIO_HEADER_1_LS;

endmodule
```

Manchester trial code

```
`timescale 1ns / 1ps

     module main_usb(
    input clk, wire USB_1_TX,wire GPIO_HEADER_1_LS,wire USB_1_CTS,
    output wire USB_1_RX,wire GPIO_HEADER_0_LS,wire GPIO_LED_0,wire
GPIO_LED_1,wire GPIO_LED_2,wire GPIO_LED_3
 );


    reg [10000:0] data;
    reg  datarecieved;
     reg [9:0] startbits;
     reg [9:0] endbits;
     reg [15:0] counterbitrate;
     reg [15:0] counterbitrateend;
     reg startflag;
     reg endflag;
     reg counterstartflag;
     reg [20:0]counterendflag;
     reg oncestart;
     reg onceend;

     reg [8:0]counterhalflifezero;
     reg [8:0]counterhalflifeone;
     reg [12:0]counterrx;
```

```
      initial begin
      data=0;
      counterbitrate=0;
      counterbitrateend=0;
      startbits=10'b0010110111;
      endbits=10'b0010111011;

      startflag=0;
      endflag=0;
      oncestart=1'b1;
      onceend=1'b0;
      counterendflag=0;

      counterhalflifezero=0;
      counterhalflifeone=0;
      datarecieved=1;

      end



      /*always @(posedge CLK_OUT1)
      begin
      if(counterclk10==250) begin     //1000=250
      clk10=~clk10;
      counterclk10=0;
      end else
      counterclk10=counterclk10+1;
      end
      */


      //DELAY AT THE BEGIN OF SENDING
      always @(posedge CLK_OUT1)
      begin

      //shift register
         data =data << 1;
      //data[0] = USB_1_TX;

      //manchester coding
      if(oncestart ==1'b0)begin

      if (USB_1_TX==1'b1)begin
      if(counterhalflifeone<250)begin
      data[0]=1'b0;
      end else if(counterhalflifeone>=250 &&
counterhalflifeone<499)begin
      data[0]=1'b1;
      end else
      counterhalflifeone=0;
      end


     if(USB_1_TX==1'b0)begin
      if(counterhalflifezero<250)begin
```

```
data[0]=1'b1;
end else if(counterhalflifezero>=250 && counterhalflifezero<499)
data[0]=1'b0;
end else
counterhalflifezero=0;
end

//endflag after 100000 clk idle
if(USB_1_TX==1'b1) begin   //------>
     if(onceend==1'b1)begin
          if (counterendflag==5000)begin
            oncestart=1'b1;
            onceend=1'b0;
            counterendflag=0;
            endflag=1;    //must be zero when starting again
           counterbitrateend=0;
          end else
          counterendflag=counterendflag+1;
       end
end else
counterendflag=1'b0;


//initialize startflag when start receiving data
if(USB_1_TX==1'b0 && oncestart==1'b1 ) begin   //------>
startflag=1;
oncestart=1'b0;      //oncestart is initialized by 1
onceend=1'b1;
counterbitrate=0;
end   //-------<

//start bits sending
if(startflag==1)begin  //----------->
if(counterbitrate<500) begin
data[9999]=startbits[9];
data[9998]=startbits[9];
counterbitrate=counterbitrate+1;
end else if (counterbitrate>=500 && counterbitrate<500*2)begin
data[9999]=startbits[8];
data[9998]=startbits[8];
counterbitrate=counterbitrate+1;
end else if (counterbitrate>=500*2 && counterbitrate<500*3)begin
data[9999]=startbits[7];
data[9998]=startbits[7];
counterbitrate=counterbitrate+1;
end else if (counterbitrate>=500*3 && counterbitrate<500*4)begin
data[9999]=startbits[6];
data[9998]=startbits[6];
counterbitrate=counterbitrate+1;
end else if (counterbitrate>=500*4 && counterbitrate<500*5)begin
data[9999]=startbits[5];
data[9998]=startbits[5];
counterbitrate=counterbitrate+1;
end else if (counterbitrate>=500*5 && counterbitrate<500*6)begin
data[9999]=startbits[4];
data[9998]=startbits[4];
counterbitrate=counterbitrate+1;
```

```
     end else if (counterbitrate>=500*6 && counterbitrate<500*7)begin
     data[9999]=startbits[3];
     data[9998]=startbits[3];
     counterbitrate=counterbitrate+1;
     end else if (counterbitrate>=500*7 && counterbitrate<500*8)begin
     data[9999]=startbits[2];
     data[9998]=startbits[2];
     counterbitrate=counterbitrate+1;
     end else if (counterbitrate>=500*8 && counterbitrate<500*9)begin
     data[9999]=startbits[1];
     data[9998]=startbits[1];
     counterbitrate=counterbitrate+1;
     end else if (counterbitrate>=500*9 && counterbitrate<500*10)begin
     data[9999]=startbits[0];
     data[9998]=startbits[0];
     counterbitrate=counterbitrate+1;
     end else
     startflag=0;
     end  //-----------<


     // //end bits sending
     if(endflag==1)begin  //----------->
     if(counterbitrateend<500) begin
     data[0]=endbits[9];
     counterbitrateend=counterbitrateend+1;
     end else if (counterbitrateend>=500 &&
counterbitrateend<500*2)begin
     data[0]=endbits[8];
     counterbitrateend=counterbitrateend+1;
     end else if (counterbitrateend>=500*2 &&
counterbitrateend<500*3)begin
     data[0]=endbits[7];
     counterbitrateend=counterbitrateend+1;
     end else if (counterbitrateend>=500*3 &&
counterbitrateend<500*4)begin
     data[0]=endbits[6];
     counterbitrateend=counterbitrateend+1;
     end else if (counterbitrateend>=500*4 &&
counterbitrateend<500*5)begin
     data[0]=endbits[5];
     counterbitrateend=counterbitrateend+1;
     end else if (counterbitrateend>=500*5 &&
counterbitrateend<500*6)begin
     data[0]=endbits[4];
     counterbitrateend=counterbitrateend+1;
     end else if (counterbitrateend>=500*6 &&
counterbitrateend<500*7)begin
     data[0]=endbits[3];
     counterbitrateend=counterbitrateend+1;
     end else if (counterbitrateend>=500*7 &&
counterbitrateend<500*8)begin
     data[0]=endbits[2];
     counterbitrateend=counterbitrateend+1;
     end else if (counterbitrateend>=500*8 &&
counterbitrateend<500*9)begin
     data[0]=endbits[1];
```

```
     counterbitrateend=counterbitrateend+1;
     end else if (counterbitrateend>=500*9 &&
counterbitrateend<500*10)begin
     data[0]=endbits[0];
     counterbitrateend=counterbitrateend+1;
     end else
     endflag=0;
     end  //-----------<



      //tx decoding manchester
      if(oncestart ==1'b0)begin

         if(counterrx<100)begin
         counterrx=counterrx+1;
         datarecieved=~GPIO_HEADER_1_LS;
         end else if(counterrx>=100 && counterrx<499)
         counterrx=counterrx+1;
         else
         counterrx=0;

      end






     end //always end



   wire CLK_OUT1;
  CLK_WIZ instance_namew
   (// Clock in ports
    .CLK_IN1(clk),      // IN  27MHZ
    // Clock out ports
    .CLK_OUT1(CLK_OUT1));   //4.8MHZ


assign GPIO_HEADER_0_LS=data[9999];
assign USB_1_RX=datarecieved;
assign GPIO_LED_0=oncestart;
assign GPIO_LED_1=onceend;
assign GPIO_LED_2=endflag;
assign GPIO_LED_3=USB_1_CTS;
endmodule
```

⇨ Final code

Startbits, Endbits and Delay

```
`timescale 1ns / 1ps


     module main_usb(
    input clk, wire USB_1_TX,wire GPIO_HEADER_1_LS,wire USB_1_CTS,
    output wire USB_1_RX,wire GPIO_HEADER_0_LS,wire GPIO_LED_0,wire
GPIO_LED_1,wire GPIO_LED_2,wire GPIO_LED_3
 );


    reg [30000:0] data;
     reg [29:0] startbits;
     reg [29:0] endbits;
     reg [15:0] counterbitrate;
     reg [15:0] counterbitrateend;
     reg startflag;
     reg endflag;
     reg counterstartflag;
     reg [20:0]counterendflag;
     reg oncestart;
     reg onceend;

     initial begin
     data=0;
     counterbitrate=0;
     counterbitrateend=0;
     startbits=30'b001011011100101101110010110111;
     endbits=30'b001011101100101110110010111011; //
     startflag=0;
     endflag=0;
     oncestart=1'b1;
     onceend=1'b0;
     counterendflag=0;

     end
     //DELAY AT THE BEGIN OF SENDING
     always @(posedge CLK_OUT1)
     begin
     //shift register
     data =data << 1;
     data[0] = USB_1_TX;
     //endflag after 50000 clk idle is eqivelant to 10 character
```

```
     if(USB_1_TX==1'b1) begin   //------>
          if(onceend==1'b1)begin
               if (counterendflag==50000)begin
               oncestart=1'b1;
                 onceend=1'b0;
               counterendflag=0;
                 endflag=1;    //must be zero when starting again
                counterbitrateend=0;
               end else
               counterendflag=counterendflag+1;
            end
     end else
     counterendflag=1'b0;
     //initialize startflag when start receiving data
     if(USB_1_TX==1'b0 && oncestart==1'b1 ) begin   //------>
     startflag=1;
     oncestart=1'b0;      //oncestart is initialized by 1
     onceend=1'b1;
     counterbitrate=0;
     end   //-------<
     //start bits sending..........500clk is equivelant to 1 bit in
9600 baud rate
     if(startflag==1)begin  //----------->
     if(counterbitrate<500) begin
     data[29999]=startbits[29];
     data[29998]=startbits[29];
     counterbitrate=counterbitrate+1;
     end else if (counterbitrate>=500 && counterbitrate<500*2)begin
     data[29999]=startbits[28];
     data[29998]=startbits[28];
     counterbitrate=counterbitrate+1;
     end else if (counterbitrate>=500*2 && counterbitrate<500*3)begin
     data[29999]=startbits[27];
     data[29998]=startbits[27];
     counterbitrate=counterbitrate+1;
     end else if (counterbitrate>=500*3 && counterbitrate<500*4)begin
     data[29999]=startbits[26];
     data[29998]=startbits[26];
     counterbitrate=counterbitrate+1;
     end else if (counterbitrate>=500*4 && counterbitrate<500*5)begin
     data[29999]=startbits[25];
     data[29998]=startbits[25];
     counterbitrate=counterbitrate+1;
     end else if (counterbitrate>=500*5 && counterbitrate<500*6)begin
     data[29999]=startbits[24];
     data[29998]=startbits[24];
     counterbitrate=counterbitrate+1;
     end else if (counterbitrate>=500*6 && counterbitrate<500*7)begin
     data[29999]=startbits[23];
     data[29998]=startbits[23];
     counterbitrate=counterbitrate+1;
     end else if (counterbitrate>=500*7 && counterbitrate<500*8)begin
     data[29999]=startbits[22];
     data[29998]=startbits[22];
     counterbitrate=counterbitrate+1;
     end else if (counterbitrate>=500*8 && counterbitrate<500*9)begin
     data[29999]=startbits[21];
```

```
data[29998]=startbits[21];
counterbitrate=counterbitrate+1;
end else if (counterbitrate>=500*9 && counterbitrate<500*10)begin
data[29999]=startbits[20];
data[29998]=startbits[20];
counterbitrate=counterbitrate+1;
end else if (counterbitrate>=500*10 && counterbitrate<500*11)begin
data[29999]=startbits[19];
data[29998]=startbits[19];
counterbitrate=counterbitrate+1;
end else if (counterbitrate>=500*11 && counterbitrate<500*12)begin
data[29999]=startbits[18];
data[29998]=startbits[18];
counterbitrate=counterbitrate+1;
end else if (counterbitrate>=500*12 && counterbitrate<500*13)begin
data[29999]=startbits[17];
data[29998]=startbits[17];
counterbitrate=counterbitrate+1;
end else if (counterbitrate>=500*13 && counterbitrate<500*14)begin
data[29999]=startbits[16];
data[29998]=startbits[16];
counterbitrate=counterbitrate+1;
end else if (counterbitrate>=500*14 && counterbitrate<500*15)begin
data[29999]=startbits[15];
data[29998]=startbits[15];
counterbitrate=counterbitrate+1;
end else if (counterbitrate>=500*15 && counterbitrate<500*16)begin
data[29999]=startbits[14];
data[29998]=startbits[14];
counterbitrate=counterbitrate+1;
end else if (counterbitrate>=500*16 && counterbitrate<500*17)begin
data[29999]=startbits[13];
data[29998]=startbits[13];
counterbitrate=counterbitrate+1;
end else if (counterbitrate>=500*17 && counterbitrate<500*18)begin
data[29999]=startbits[12];
data[29998]=startbits[12];
counterbitrate=counterbitrate+1;
end else if (counterbitrate>=500*18&& counterbitrate<500*19)begin
data[29999]=startbits[11];
data[29998]=startbits[11];
counterbitrate=counterbitrate+1;
end else if (counterbitrate>=500*19 && counterbitrate<500*20)begin
data[29999]=startbits[10];
data[29998]=startbits[10];
counterbitrate=counterbitrate+1;
end else if (counterbitrate>=500*20 && counterbitrate<500*21)begin
data[29999]=startbits[9];
data[29998]=startbits[9];
counterbitrate=counterbitrate+1;
end else if (counterbitrate>=500*21 && counterbitrate<500*22)begin
data[29999]=startbits[8];
data[29998]=startbits[8];
counterbitrate=counterbitrate+1;
end else if (counterbitrate>=500*22 && counterbitrate<500*23)begin
data[29999]=startbits[7];
data[29998]=startbits[7];
```

```
      counterbitrate=counterbitrate+1;
      end else if (counterbitrate>=500*23 && counterbitrate<500*24)begin
      data[29999]=startbits[6];
      data[29998]=startbits[6];
      counterbitrate=counterbitrate+1;
      end else if (counterbitrate>=500*24 && counterbitrate<500*25)begin
      data[29999]=startbits[5];
      data[29998]=startbits[5];
      counterbitrate=counterbitrate+1;
      end else if (counterbitrate>=500*25 && counterbitrate<500*26)begin
      data[29999]=startbits[4];
      data[29998]=startbits[4];
      counterbitrate=counterbitrate+1;
      end else if (counterbitrate>=500*26 && counterbitrate<500*27)begin
      data[29999]=startbits[3];
      data[29998]=startbits[3];
      counterbitrate=counterbitrate+1;
      end else if (counterbitrate>=500*27 && counterbitrate<500*28)begin
      data[29999]=startbits[2];
      data[29998]=startbits[2];
      counterbitrate=counterbitrate+1;
      end else if (counterbitrate>=500*28 && counterbitrate<500*29)begin
      data[29999]=startbits[1];
      data[29998]=startbits[1];
      counterbitrate=counterbitrate+1;
      end else if (counterbitrate>=500*29 && counterbitrate<500*30)begin
      data[29999]=startbits[0];
      data[29998]=startbits[0];
      counterbitrate=counterbitrate+1;
      end  else
      startflag=0;
      end  //-----------<
//end bits sending
      if(endflag==1)begin  //----------->
      if(counterbitrateend<500) begin
      data[0]=endbits[29];
      counterbitrateend=counterbitrateend+1;
      end else if (counterbitrateend>=500 &&
counterbitrateend<500*2)begin
      data[0]=endbits[28];
      counterbitrateend=counterbitrateend+1;
      end else if (counterbitrateend>=500*2 &&
counterbitrateend<500*3)begin
      data[0]=endbits[27];
      counterbitrateend=counterbitrateend+1;
      end else if (counterbitrateend>=500*3 &&
counterbitrateend<500*4)begin
      data[0]=endbits[26];
      counterbitrateend=counterbitrateend+1;
      end else if (counterbitrateend>=500*4 &&
counterbitrateend<500*5)begin
      data[0]=endbits[25];
      counterbitrateend=counterbitrateend+1;
      end else if (counterbitrateend>=500*5 &&
counterbitrateend<500*6)begin
      data[0]=endbits[24];
      counterbitrateend=counterbitrateend+1;
```

```
     end else if (counterbitrateend>=500*6 &&
counterbitrateend<500*7)begin
     data[0]=endbits[23];
     counterbitrateend=counterbitrateend+1;
     end else if (counterbitrateend>=500*7 &&
counterbitrateend<500*8)begin
     data[0]=endbits[22];
     counterbitrateend=counterbitrateend+1;
     end else if (counterbitrateend>=500*8 &&
counterbitrateend<500*9)begin
     data[0]=endbits[21];
     counterbitrateend=counterbitrateend+1;
     end else if (counterbitrateend>=500*9 &&
counterbitrateend<500*10)begin
     data[0]=endbits[20];
     counterbitrateend=counterbitrateend+1;
     end else if (counterbitrateend>=500*10 &&
counterbitrateend<500*11)begin
     data[0]=endbits[19];
     counterbitrateend=counterbitrateend+1;
     end else if (counterbitrateend>=500*11 &&
counterbitrateend<500*12)begin
     data[0]=endbits[18];
     counterbitrateend=counterbitrateend+1;
     end else if (counterbitrateend>=500*12 &&
counterbitrateend<500*13)begin
     data[0]=endbits[17];
     counterbitrateend=counterbitrateend+1;
     end else if (counterbitrateend>=500*13 &&
counterbitrateend<500*14)begin
     data[0]=endbits[16];
     counterbitrateend=counterbitrateend+1;
     end else if (counterbitrateend>=500*14 &&
counterbitrateend<500*15)begin
     data[0]=endbits[15];
     counterbitrateend=counterbitrateend+1;
     end else if (counterbitrateend>=500*15 &&
counterbitrateend<500*16)begin
     data[0]=endbits[14];
     counterbitrateend=counterbitrateend+1;
     end else if (counterbitrateend>=500*16 &&
counterbitrateend<500*17)begin
     data[0]=endbits[13];
     counterbitrateend=counterbitrateend+1;
     end else if (counterbitrateend>=500*17 &&
counterbitrateend<500*18)begin
     data[0]=endbits[12];
     counterbitrateend=counterbitrateend+1;
     end else if (counterbitrateend>=500*18 &&
counterbitrateend<500*19)begin
     data[0]=endbits[11];
     counterbitrateend=counterbitrateend+1;
     end else if (counterbitrateend>=500*19 &&
counterbitrateend<500*20)begin
     data[0]=endbits[10];
     counterbitrateend=counterbitrateend+1;
```

```
     end else if (counterbitrateend>=500*20 &&
counterbitrateend<500*21)begin
     data[0]=endbits[9];
     counterbitrateend=counterbitrateend+1;
     end else if (counterbitrateend>=500*21 &&
counterbitrateend<500*22)begin
     data[0]=endbits[8];
     counterbitrateend=counterbitrateend+1;
     end else if (counterbitrateend>=500*22 &&
counterbitrateend<500*23)begin
     data[0]=endbits[7];
     counterbitrateend=counterbitrateend+1;
     end else if (counterbitrateend>=500*23 &&
counterbitrateend<500*24)begin
     data[0]=endbits[6];
     counterbitrateend=counterbitrateend+1;
     end else if (counterbitrateend>=500*24 &&
counterbitrateend<500*25)begin
     data[0]=endbits[5];
     counterbitrateend=counterbitrateend+1;
     end else if (counterbitrateend>=500*25 &&
counterbitrateend<500*26)begin
     data[0]=endbits[4];
     counterbitrateend=counterbitrateend+1;
     end else if (counterbitrateend>=500*26 &&
counterbitrateend<500*27)begin
     data[0]=endbits[3];
     counterbitrateend=counterbitrateend+1;
     end else if (counterbitrateend>=500*27 &&
counterbitrateend<500*28)begin
     data[0]=endbits[2];
     counterbitrateend=counterbitrateend+1;
     end else if (counterbitrateend>=500*28 &&
counterbitrateend<500*29)begin
     data[0]=endbits[1];
     counterbitrateend=counterbitrateend+1;
     end else if (counterbitrateend>=500*29 &&
counterbitrateend<500*30)begin
     data[0]=endbits[0];
     counterbitrateend=counterbitrateend+1;
     end else
     endflag=0;
     end  //-----------<
     end
wire CLK_OUT1;
  CLK_WIZ instance_namew
   (// Clock in ports
    .CLK_IN1(clk),      // IN  27MHZ
    // Clock out ports
    .CLK_OUT1(CLK_OUT1));   //4.8MHZ
assign GPIO_HEADER_0_LS=data[29999];
assign USB_1_RX=GPIO_HEADER_1_LS;
assign GPIO_LED_0=oncestart;
assign GPIO_LED_1=onceend;
assign GPIO_LED_2=endflag;
assign GPIO_LED_3=USB_1_CTS;
endmodule
```

Configuration file used for FPGA Spartan kit SP605

```
##UCF file
NET "GPIO_LED_0"                LOC = "D17";   ## 2   on DS3 LED
NET "GPIO_LED_1"                LOC = "AB4";   ## 2   on DS4 LED
NET "GPIO_LED_2"                LOC = "D21";   ## 2   on DS5 LED
NET "GPIO_LED_3"                LOC = "W15";   ## 2   on DS6 LED
NET "USB_1_CTS"                 LOC = "F18";   ##
#NET "USB_1_RTS"                 LOC = "F19";   ##
NET "USB_1_RX"                  LOC = "B21";   ##
NET "USB_1_TX"                  LOC = "H17";   ##
##
NET "clk"               LOC = "AB13";  ##
##
#NET "GPIO_SWITCH_0"             LOC = "C18";   ## 1   on S2 DIP switch (active-
high)
#NET "GPIO_SWITCH_1"             LOC = "Y6";    ## 2   on S2 DIP switch (active-
high)
#NET "GPIO_SWITCH_2"             LOC = "W6";    ## 3   on S2 DIP switch (active-
high)
#NET "GPIO_SWITCH_3"             LOC = "E4";    ## 4   on S2 DIP switch (active-
high)
#LVCMOS for high voltage 3.3volt output pin for FPGA, default is 2.2v
#NET "USB_1_TX" IOSTANDARD = LVCMOS33 ;
#NET "USB_1_RX" IOSTANDARD = LVCMOS33 ;
##
NET "GPIO_HEADER_0_LS"          LOC = "G7";    ## 1   on U52 (level shifter,
U52.20 <-> GPIO_HEADER_0 <-> series R280 200 ohm <-> 1 on J55
```

NET "GPIO_HEADER_1_LS" LOC = "H6"; ## 3 on U52 (level shifter,
U52.18 <-> GPIO_HEADER_0 <-> series R281 200 ohm <-> 2 on J55