TECHNICAL UNIVERSITY OF CRETE

DEPARTMENT OF ELECTRONIC AND COMPUTER ENGINEERING

THESIS PROJECT:

DESIGN AND IMPLEMENTATION OF A DIGITAL TRANSMITTER BASED ON THE MULTIBAND OFDM PHYSICAL LAYER PROPOSAL FOR UWB

Undergraduate student: Platsis Michalis

Board: Assistant Professor: Papaefstathiou Ioannis (supervisor)
Professor Dollas Apostolos
Associate Professor: Pnevmatikatos Dionisios

Chania 25/04/2007

I would like to thank my project advisor, Assistant Professor Ioannis Papaefstathiou, for his valuable help during the course.

I would also like to thank Professor Apostolos Dollas and Associate Professor Dionisios Pnevmatikatos for their presence in the presentation of this thesis.

Above all, I would like to thank my parents and family for their support and encouragement.

To my parents,

TABLE OF CONTENTS

CHAPTER 1 INTRODUCTION

UWB is a high data rate (up to 480 Mb/s), short range (up to 10 meters) technology for wireless personal area networks (WPANs). Operating in the range from 3.1 GHz up to 10.6 GHz band of the spectrum, UWB transmits the information over a minimum bandwidth of 500 MHz. Modern UWB systems use Modulation techniques such as OFDM (Orthogonal Frequency Division Multiplexing). OFDM is a multicarrier transmission technique that uses multiple frequencies to simultaneously transmit multiple signals in parallel. Orthogonality between neighboring frequencies (having overlapping spectrum) results in optimal usage of bandwidth. Several advantages include high spectral flexibility and resilience to RF interference and multipath effects. Furthermore, low complexity implementation is achieved by taking advantage of efficient FFT algorithms. After current developments in DSP and VLSI technologies, OFDM Modulation technique has become a feasible option and has been already applied successfully to several popular commercial communications systems including asymmetric digital subscriber line (ADSL), IEEE 802.11a, IEEE 802.11g, and IEEE 802.16a.

In this thesis the design of the digital part of the UWB transmitter is described. The MB-OFDM proposal is selected for our UWB system model. The main object is to create VHDL synthesible source codes for all the transmit fuctions applying the respective rate dependent modulation parameters for the support of the mandatory data rates of 53.3,110,200 Mbps. Also appropriate control logic is necessary to allow for transmitting samples in a continuous way with a sample frequency equal with the operation frequency of the transmitter. Since the required performance depends on the digital transmitter clock frequency which has to be the same with the signal bandwidth or sample frequency (528 MHz) we notice that the speed requirements of Ultrawideband baseband are very demanding. So a first goal is to explore how we can achieve performance as close as possible to the specifications, synthesizing the RTL VHDL description with current FPGA technologies. In this way we could estimate the achievable data rates of our design and decide the selection of the FPGA platform that could get a real-time speed in hardware according to the specifications. Initially Spartan-3 FPGA platform is chosen due to its low cost but its limited speed will leed us to the selection of Virtex-4 and even Virtex-5 FPGA. The synthesis results and the achievable transmission taxes using each FPGA platform are presented.

In chapter 2 an overview of the UWB technology is discussed, with its application trends. Also existing UWB Physical Layer Standards are presented and a comparison of UWB with other technologies is discussed. Chapter 3 gives the basic theory of OFDM modulation technique and a general description of our UWB OFDM model. In chapter 4 the design flow is explained as well as the specifications of the design. The hardware design of the digital UWB transmitter is described in Chapter 5. Chapter 6 contents the verification mechanism for our deisgn and some examples of the test vectors that we have applied for RTL simulation. In chapter 7 synthesis results are represented and the achievable data rates are given for each of the selected FPGA platforms (Spartan-3, Virtex-4, Virtex-5). Finally in chapter 8 conclusions are pointed out and also ideas for future work are recommended.

CHAPTER 2 ULTRA-WIDEBAND(UWB) TECHNOLOGY

2.1 UWB TECHOLOGY OVERVIEW

The origins of UWB technology begun in 1962 that was generally referred to as impulse radio, baseband or carrier-free communications. The term “ultrawideband” was first coined by the U.S. Department of Defense in 1989, and early applications leveraged the technology’s properties as ground-penetrating radar. Although it began as a military application dating from the 1960s, UWB has been redefined as a high data rate (480+ Mbps), short-range (up to 10 meters) technology for transmitting information spread over a large bandwidth (>500 MHz) that should, in theory and under the right circumstances, be able to share spectrum with other users.

Today,Ultra-wideband (UWB) communication technology is emerging as a leading standard for high data rate applications over wireless networks. Due to its use of a high frequency bandwidth, UWB allows the wireless connection of multiple devices at very high data rates. Ultra-Wideband (UWB) technology brings the convenience and mobility of wireless communications to high-speed interconnects in devices throughout the digital home and office. Designed for short-range, wireless personal area networks (WPANs), UWB is the leading technology for freeing people from wires, enabling wireless connection of multiple devices for transmission of video, audio and other high-bandwidth data.UWB, short-range radio technology, complements other longer range radio technologies such as Wi-Fi, WiMAX, and cellular wide area communications. It is used to relay data from a host device to other devices in the immediate area (up to 10 meters, or 30 feet).UWB technology specifically addresses emerging applications in the consumer electronics, personal computing and mobile device markets. When compared to other existing and nascent technologies capable of providing wireless connectivity, the performance benefits of UWB are compelling. For example, transferring a 1 Gbyte file full of vacation pictures from a digital camera to a photo kiosk would take merely seconds with UWB compared to hours using other currently available, lower speed technologies (i.e. Bluetooth) and consume far less battery power in doing so.

UWB differs substantially from conventional narrowband radio frequency (RF) and spread spectrum technologies (SS), such as Bluetooth technology and 802.11a/b/g. A traditional UWB transmitter works by sending billions of pulses across a very wide spectrum of frequencies several GHz in bandwidth. The corresponding receiver then translates the pulses into data by listening for a familiar pulse sequence sent by the transmitter. Specifically, UWB is defined as any radio technology having a spectrum that occupies a bandwidth greater than 20 percent of the center frequency, or a bandwidth of at least 500 MHz. UWB's combination of broader spectrum and lower power improves speed and reduces interference with other wireless spectra. In the United States, the Federal Communications Commission (FCC) has mandated that UWB radio transmissions can legally operate in the range from 3.1 GHz up to 10.6 GHz, at a limited transmit power of -41dBm/MHz. Consequently, UWB provides dramatic channel capacity at short range that limits interference.

Modern UWB systems use modulation techniques, such as Orthogonal Frequency Division Multiplexing (OFDM), to occupy these extremely wide bandwidths. In addition, the use of multiple bands in combination with OFDM modulation can provide significant advantages to traditional UWB systems.

2.2 UWB PHYSICAL LAYER STANDARDS

Ultra-wideband (UWB) has been proposed for the physical layer standard of the future high-speed wireless personal area networks (WPANs).The Federal Communications Commission (FCC) created a great opportunity in February 2002 when 7500 MHz of spectrum was allocated for unlicensed use of commercial UWB devices.Rather than requiring a UWB radio to use the entire 7.5 GHz band to transmit information, or even a substantive portion of it, the FCC defined a specific minimum bandwidth of 500 MHz at a -10dB level. This minimum bandwidth (in conjunction with other requirements of the FCC ruling) would substantially protect incumbent users of the spectrum. The flexibility provided by the FCC ruling greatly expands the design options for UWB communication systems. Designers are free to use a combination of sub-bands within the spectrum to optimize system performance, power consumption and design complexity. UWB systems can still maintain the same low transmit power as if they were using the entire bandwidth by interleaving the symbols across these sub-bands.

Given this option for a multi-band system, information can either be transmitted by the traditional pulse-based single carrier method or by more advanced multi-carrier techniques. Pulse-based single-carrier systems transmit signals by modulating the phase of a very narrow pulse. While this is a proven technology that only requires a very simple transmitter design, several inherent disadvantages exist. It is difficult to collect enough signal energy in a typical usage environment (with many reflecting surfaces) using a single RF chain; switching time requirements can be very stringent at both the transmitter and receiver; the receiver signal processing is very sensitive to group delay variations introduced by analog front-end components; and, spectral resources are potentially wasted in order to avoid narrowband interference.

In contrast, MB-OFDM(Multiband OFDM) transmits data simultaneously over multiple carriers spaced apart at precise frequencies. Fast Fourier Transform algorithms provide nearly 100 percent efficiency in capturing energy in a multi-path environment, while only slightly increasing transmitter complexity. Beneficial attributes of MB-OFDM include high spectral flexibility and resiliency to RF interference and multi-path effects. This has allowed designers to develop a specification, based on multiband OFDM, which meets the stringent market requirements: hundreds of megabits per second at low power and low cost. Several key organizations (MBOA, WiMedia, Wireless USB) have selected this design for their applications.OFDM modulation techniques have been already applied successfully to several other high-performance, popular commercial communications systems including asymmetric digital subscriber line (ADSL), IEEE 802.11a, IEEE 802.11g, and IEEE 802.16a.

MultiBand Orthogonal Frequency Division Multiplexing (MB-OFDM) combining Orthogonal Frequency Division Multiplexing (OFDM) with the above multi-band

approach is considered as the optimal UWB solution. An OFDM-based physical layer is one of the promising options for UWB devices due to its capability to capture multipath energy and eliminate inter-symbol interference and the fact that OFDM is considered by the industry as a mature and reliable technology.For these reasons, our focus will be on the OFDM-based modulation scheme for UWB communications.In addition, researches have shown that using pulsed-OFDM does not improve so much system performance in terms of BER compared to a robust MB-OFDM mode.Further, pulsed approach can only be used in low data rate applications..

2.2.1 IEEE 802.15.3

IEEE 802.15.3 is the IEEE standard for high data rate (20Mbit/s or greater) Wireless Personal Area Networks (WPAN) to provide Quality of Service (QoS) for real time distribution of multimedia content. IEEE 802.15.3 is accomplished by the IEEE P802.15.3 High Rate (HR) Task Group (TG3). The task group is charged with defining a universal standard of ultra wideband radios capable of high data rate over a distance of 10 meters using the 3.1GHz to 10.6GHz band (see Figure 1) for TVs, cell phones, PCs, and so forth. Besides a high data rate, the new standard will provide for low power, low cost solutions addressing the needs of portable consumer digital imaging and multimedia applications. In addition, ad hoc peer-to-peer networking, security issues are considered. When combined with the 802.15.3 PAN standard, UWB will provide a very compelling wireless multimedia network for the home.

The IEEE 802.15 Task Group 3a has been developing a physical layer standard for UWB technologies to support high data rates for wireless personal area networks (WPANs).Two merged technical proposals, referred to as multi-band orthogonal frequency division multiplexing (MBOFDM) and direct-sequence ultra-wideband (DS-UWB), are considered as strong candidates for the final high-speed WPAN standard.Unlike other committees who elected to release two methods as part of a single standard, in IEEE 802.15.3a many members believed that this approach would be destructive to the PAN(Personal Area Network) market because it operated over extremely short range (<10m) and so was particularly vulnerable to interference. Further more, having two different and incompatible standards to perform the same function would only confuse the market and delay the adoption of either of those technologies.

2.2.2 WiMedia UWB

Supporters of the Multiband OFDM proposal decided not to support an effort to split the standard. But it was also necessary that the delay in the standards process did not result in a delay in the entry of products into the market. To prevent this from happening, the OFDM supporters elected to continue the work on standardization outside of the IEEE 802.15.3a task group.Over a period of time, this outside group gradually formalized their relationship to provide a legal context in which to work. While starting as an organization called “Multiband OFDM Alliance” (or MBOA), eventually this group became known as the WiMedia Alliance. The organizational

structure is similar to that of the Wi-Fi Alliance, an organization that tests, certifies, and promotes interoperable IEEE 802.11 products.

The new WiMedia Alliance represents a combination of WiMedia with the Multiband OFDM Alliance SIG (MBOA-SIG).They will publish and manage the industry UWB specifications for rapid adoption by for mobile, consumer electronics and PC applications. The MBOA-SIG Promoter companies include Alereon, HP, Intel, Kodak, Microsoft, Nokia, Philips, Samsung Electronics, Sony, etc. MBOA member companies are actively engaged with IEEE standards process. The MOBA will announce its specifications for a physical layer ("PHY") and Media Access Control layer ("MAC") to enhance personal electronic devices mobility. The MBOA MAC and PHY specifications will serve as the common radio platform for industry standards. The MBOA MAC and PHY specifications, as published in Ecma-368, are intentionally designed to adapt to various requirements set by global regulatory bodies. The Multiband OFDM Alliance (MBOA) has devised its own media access control (MAC) layer, in effect rejecting the MAC mandated by the IEEE for the upcoming 802.15.3a standard. Enhanced support for mobility, mesh networking and management of piconets will be the key to the new MAC. Other application-friendly features in MBOA include the reduced level of complexity per node, long battery life, support of multiple power management modes and higher spatial capacity.

The UWB transmitter which is described in this thesis is based on the Multiband-OFDM Physical Layer Proposal for IEEE 802.15 Task Group 3a announced by the Multiband OFDM Alliance SIG (MBOA-SIG).

2.3 UWB APPLICATIONS

Today, most computer and consumer electronic devices—everything from a digital camcorder and DVD player to a mobile PC and a high-definition TV (HDTV)—require wires to record, play or exchange data. UWB could eliminate these wires, allowing people to eliminate the tangle of cables in their homes and offices.There are three overlapping target segments that could benefit from short-range wireless connections enabled by UWB:

- PC and peripheral devices
- mobile devices
- consumer electronics.

Many devices in each of these three segments frequently communicate significant amounts of data over very short distances with other complementary devices, usually by means of an interconnect cable. For example, a digital still camera, with a large storage capacity, typically requires a high-speed serial connection to the PC to transfer images. At the time of transfer, the distance between the PC and the camera is typically a few meters at most. UWB allows us to create a wireless link by enabling the necessary data rates in a radio suitable for cost-sensitive, battery-powered mobile devices, like a camera or PDA. Similar examples are smart phones, home entertainment centers, printers, handheld computers, camcorders, video projectors and MP3 players.For MBOA UWB, anticipated early applications include the exchange of media content over high-data consumer electronics devices including MP3 players,

personal media players (PMPs), set-top-boxes, digital cameras, hard-drives, printers/scanners, home-theater equipment, mobile phones, personal computers and video gaming platforms By eliminating the need for a physical cable connection, a new level of user convenience and mobility is provided.Specific usage examples are:

- An office worker could put a mobile PC on a desk and instantly be connected to a printer, scanner and Voice over IP (VoIP) headset.
- All the components for an entire home entertainment center could be set up and connected to each other without a single wire.
- A digital camcorder could play a just-recorded video on a friend's HDTV without anyone having to fiddle with wires.
- A portable MP3 player could stream audio to high-quality surround-sound speakers anywhere in the room.
- A mobile computer user could wirelessly connect to a digital projector in a conference room to deliver a presentation.
- Digital pictures could be transferred to a photo print kiosk for instant printing without the need of a cable.

The following figure illustrates some examples of short-range wireless connections enabled by UWB.

Figure 1.examples of short-range wireless connections enabled by UWB.

Nevertheless, UWB is more than a cable replacement technology. New User Models not possible in the Cabled World are created through UWB connecting PC, CE and Mobile Segments to share and exchange data (see figure 2). The ever increasing file sizes of contents to be downloaded by consumers create the need of High Speed (20+ Mb/s) LAN/WANs to avoid waiting to long for a file to transfer.

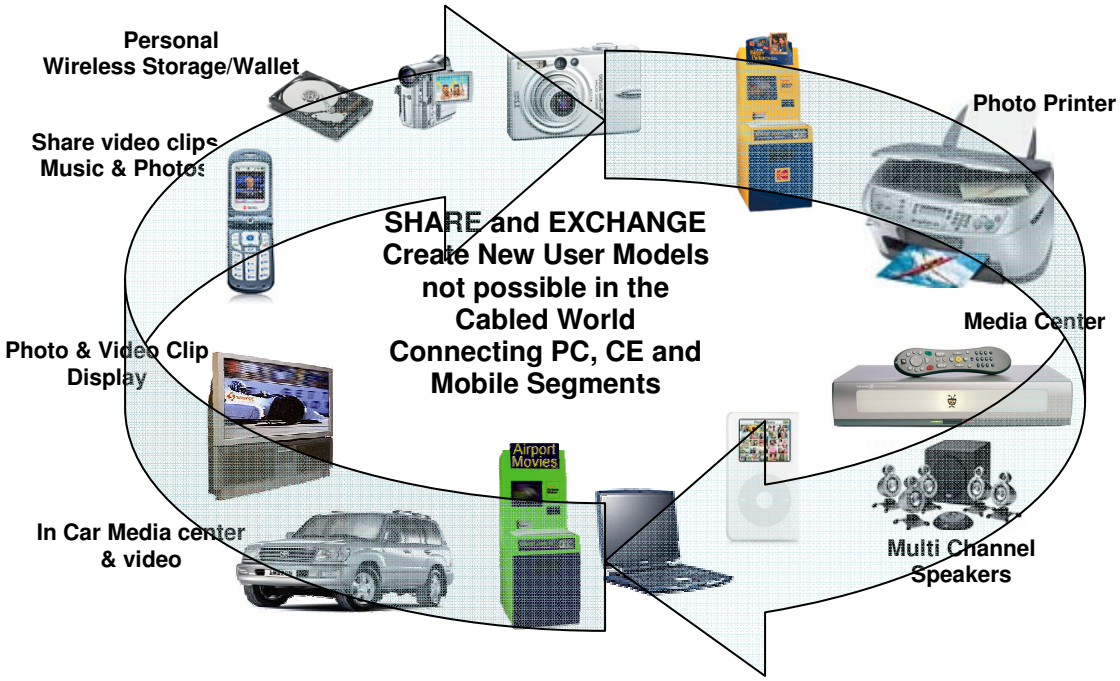


Figure 2.New user models through UWB connections

We outline here application trends where signal processing tools will probably have considerable impact in UWB system development.

▲ *Wireless personal area networks (WPANs):* Also known as in-home networks, WPANs address short-range (generally within 10–20 m) ad hoc connectivity among portable consumer electronic and communication devices. They are envisioned to provide high-quality real-time video and audio distribution, file exchange among storage systems, and cable replacement for home entertainment systems. UWB technology emerges as a promising physical layer candidate for WPANs, because it offers high-rates over short range, with low cost, high power efficiency, and low duty cycle.

▲ *Sensor networks:* Sensor networks consist of a large number of nodes spread across a geographical area. The nodes can be static, if deployed for, e.g., avalanche monitoring and pollution tracking, or mobile, if equipped on soldiers, firemen, or robots in military and emergency response situations. Key requirements for sensor networks operating in challenging environments include low cost, low power, and multifunctionality. High data-rate UWB communication systems are well motivated for gathering and disseminating or exchanging a vast quantity of sensory data in a timely manner. Typically, energy is more limited in sensor networks than in WPANs because of the nature of the sensing devices and the difficulty in recharging their batteries.Studies have shown that current commercial Bluetooth devices are less suitable for sensor network applications because of their energy requirements and higher expected cost. In addition, exploiting the precise localization capability of UWB promises wireless sensor networks with improved positioning accuracy. This is especially useful when GPSs are not available, e.g., due to obstruction.

▲ *Imaging systems:* Different from conventional radar systems where targets are typically considered as point scatterers, UWB radar pulses are shorter than the target dimensions. UWB reflections off the target exhibit not only changes in amplitude and time shift but also changes in the pulse shape. As a result, UWB waveforms exhibit pronounced sensitivity to scattering relative to conventional radar signals. This property has been readily adopted by radar systems and can be extended to additional applications, such as underground, through-wall and ocean imaging, as well as medical diagnostics and border surveillance devices.

▲ *Vehicular radar systems:* UWB-based sensing has the potential to improve the resolution of conventional proximity and motion sensors. Relying on the high ranging accuracy and target differentiation capability enabled by UWB, intelligent collision-avoidance and cruise-control systems can be envisioned. These systems can also improve airbag deployment and adapt suspension/braking systems depending on road conditions. UWB technology can also be integrated into vehicular entertainment and navigation systems by downloading high-rate data from airport off ramp, road-side, or gas station UWB transmitters.

.

2.4 UWB vs OTHER TECHNOLOGIES

In order to demonstrate a technology comparison of Ultrawideband with other technologies a quick overview of the main WPAN/WLAN/WMAN technologies is presented:

2.4.1 Ultra-Wideband (UWB)

- UWB is a wireless personal area network (PAN) technology
- UWB is a revolutionary wireless technology for transmitting digital data over a wide spectrum of frequency bands with very low power. It can transmit data at very high rates (for wireless local area network applications)
- Ideally, it will have low power consumption, low price, high speed, use a wide swath of radio spectrum, carry signals through obstacles (doors, etc.) and apply to a wide range of applications (defense, industry, home, etc.)
- Currently, there are two competing UWB standards. The UWB Forum is promoting one standard based on direct sequence (DS-UWB). The WiMedia Alliance is promoting another standard based on Multi-band Orthogonal Frequency Division Modulation (OFDM)
- operates in the range from 3.1 GHz up to 10.6 GHz, at a limited transmit power of -41dBm/MHz
- Each standard allows for data rates from approximately 0-500 Mbps at a range of 2 meters and a data rate of approximately 110 Mbps at a range of up to 10 meters
- The Bluetooth SIG announced in May 2005 its intentions to work with both groups behind UWB to develop a high rate *Bluetooth* specification on the UWB radio

2.4.2 Bluetooth Wireless Technology

- *Bluetooth* is a wireless personal area network (PAN) technology
- *Bluetooth* wireless technology is geared towards voice and data applications
- *Bluetooth* wireless technology operates in the unlicensed 2.4 GHz spectrum
- *Bluetooth* wireless technology can operate over a distance of 10 meters or 100 meters depending on the Bluetooth device class. The peak data rate with EDR is 3 Mbps
- *Bluetooth* wireless technology is able to penetrate solid objects
- *Bluetooth* technology is omni-directional and does not require line-of-sight positioning of connected devices
- Security has always been and continues to be a priority in the development of the *Bluetooth* specification. The *Bluetooth* specification allows for three modes of security
- The cost of *Bluetooth* chips is under $3

2.4.3 Certified Wireless USB

- Speed: Wireless USB is projected to be 480 Mbps up to 2 meters and 110 Mbps for up to 10 meters. Wireless USB hub can host up to 127 wireless USB devices
- Wireless USB will be based on and run over the UWB radio promoted by the WiMedia Alliance.
- Allows point-to-point connectivity between devices and the Wireless USB hub
- Intel established the Wireless USB Promoter Group in February 2004
- The USB Implementers Forum, Inc. (USB-IF) tests and certifies the "certified Wireless USB" based wireless equipment

2.4.4 Wi-Fi (IEEE 802.11)

- Wi-Fi is a wireless local area network (LAN) technology
- The Wi-Fi Alliance tests and certifies 802.11 based wireless equipment
- 802.11a: This uses OFDM, operates in the 5 GHz range, and has a maximum data rate of 54 Mbps
- 802.11b: Operates in the 2.4 GHz range, has a maximum data rate of 11 Mbps and uses DSSS. 802.11b is the original Wi-Fi standard
- 802.11g: Operates in the 2.4 GHz range, uses OFDM and has a maximum data rate of 54 Mbps. This is backwards compatible with 802.11b
- 802.11e: This standard will improve quality of service
- 802.11h: This standard is a supplement to 802.11a in Europe and will provide spectrum and power control management. Under this standard, dynamic frequency selection (FS) and transmit power control (TPC) are added to the 802.11a specification

- 802.11i: This standard is for enhanced security. It includes the advanced encryption standard (AES). This standard is not completely backwards compatible and some users will have to upgrade their hardware. The full 802.11i support is also referred to as WPA2
- 802.11k: Under development, this amendment to the standard should allow for increased radio resource management on 802.11 networks
- 802.11n: This standard is expected to operate in the 5 GHz range and offer a maximum data rate of over 100 Mbps (though some proposals are seeking upwards of 500 Mbps). 802.11n will handle wireless multimedia applications better than the other 802.11 standards
- 802.11p: This standard will operate in the automotive-allocated 5.9 GHz spectrum. It will be the basis for the dedicated short range communications (DSRC) in North America. The DSRC will allow vehicle to vehicle and vehicle to roadside infrastructure communication
- 802.11r: This amendment to the standard will improve users' ability to roam between access points or base stations. The task group developing this form in spring/summer 2004
- 802.11s: Under development, this amendment to the standard will allow for mesh networking on 802.11 networks. The task group developing this formed in spring/summer 2004

2.4.5 WiMAX (Worldwide Interoperability for Microwave Access and IEEE 802.16)

- WiMax is a wireless metropolitan area network (MAN) technology
- WiMax has a range of 50 km with data rates of 70 Mbps. Typical cell has a shorter range
- The original 802.16 standard operated in the 10-66 GHz frequency bands with line of sight environments
- The newly completed 802.16a standard operates between 2 and 11 GHz and does not need line of sight
- Delays in regulatory approval in Europe due to issues regarding the use of the spectrums in the 2.8 GHz and 3.4 GHz range
- Supports vehicle mobility for between 20 to 100+ km/hr. The 802.16e standard will allow nomadic portability
- The IEEE 802.16a and the ETSI HIPERMAN (High Performance Radio Metropolitan Area Network) share the same PHY and MAC. 802.16 has been designed from the beginning to be compatible with the European standard
- Created to compete with DSL and cable modem access, the technology is considered ideal for rural, hard to wire areas

2.4.6 WiBro (Wireless Broadband)

- Portable Internet Service (WiBro) is to provide a high data rate wireless internet access with PSS (Personal Subscriber Station) under the stationary or mobile environment, anytime and anywhere. Primarily based in South Korea based on TTA specifications.
- 2300-2400 MHz, TDD, OFDMA, channel bandwidth 10 MHz, etc.
- System shall support mobile users at a velocity of up to 60km/h
- Throughput (per user) Max. DL / UL = 3 / 1 [Mbps], Min. DL / UL = 512 / 128 [Kbps].

2.4.7 HiperLAN

- Speed: HiperLAN 2 = 54 Mbps, and has a 50 to 100 m capacity

2.4.8 HIPERMAN

- Fixed wireless access standard developed by the European Telecommunications Standards Institute (ETSI)
- Operates in the spectrum between 2 GHz and 11 GHz and is compatible/interoperable with the IEEE 802.16a-2003 standard

2.4.9 802.20

- Considered to be mobile wireless broadband wireless access.
- Maximum data rate expected to be 1 Mbps, operating in licensed bands below 3.5 GHz
- Supports vehicle mobility up to 250 km/hr

2.4.10 ZigBee (IEEE 802.15.4)

- Capacity of 250 Kbits at 2.4 GHz, 40 Kpbs at 915 Mhz, and 20 Kpbs at 868 Mhz with a range of 10-100 M
- Its purpose is to become a wireless standard for remote control in the industrial field
- The ZigBee technology is targeting the control applications industry, which does not require high data rates, but must have low power, low cost and ease of use (remote controls, home automation, etc.)
- The specification was formally adopted in December 2004
- Security was not considered in the initial development of the specification. Currently there are three levels of security

2.4.11 Advantages of UWB technoloy

The primary advantages of UWB are high data rates, low cost, and low power.Besides its blazing speed,UWB systems also consume very little power,with long battery life. This makes UWB practical for use in smaller devices, such as cell phones and PDAs, that users carry at all times. Among the most important advantages of UWB technology, however, are those of low system complexity and low cost. UWB systems can be made nearly "all-digital", with minimal RF or microwave electronics.UWB transceivers using low power short burst radio waves, they do not take as much planning to build. This makes them extremely easy and cheap to build compared to typical spread spectrum transceivers.

UWB also offers some other benefits worth considering.Since UWB waveforms are of short time duration, they have some rather unique properties. In communications, for example, UWB pulses can be used to provide extremely high data rate performance in multi-user network applications. For radar applications, these same pulses can provide very fine range resolution and precision distance and/or positioning measurement capabilities. In fact, multifunction architectures encompassing communications, radar and positioning applications have been developed. Due to the inherent RF simplicity of UWB designs and because UWB is spectrum hopping, and only for a tiny fraction of a second these systems has very little interference impact on other systems. RF simplicity of UWB designs make these systems highly frequency adaptive, enabling them to be positioned anywhere within the RF spectrum. This feature avoids interference to existing services, while fully utilizing the available spectrum.In addition, the relatively wide spectrum that UWB utilizes significantly minimizes the impact of interference from other systems as well

For the security issue, it is extremely hard to eavesdrop. It is like trying to track someone in a very busy street who continually changes different colors of clothes while running at extreme fast speed.

2.4.12 Disadvantages of UWB technology

The drawback is that the UWB speeds can be achieved only over short distances (10 meters). Communication speed is a function of bandwidth, power, and distance.Because of UWB's distance limitations, it will primarily be used for high-bandwidth local networks where the receiver can be plugged in, and not for cellular. As with any technology, there are always applications that may be better served by other approaches. For example, for extremely high data rate (10’s of Gigabits/second and higher), point-to-point or point-to-multipoint applications, it is difficult today for UWB systems to compete with high capacity optical fiber or optical wireless communications systems. Of course, the high cost associated with optical fiber installation and the inability of an optical wireless signal to penetrate a wall dramatically limit the applicability of optically-based systems for in-home or in building applications. In addition, optical wireless systems have extremely precise pointing requirements, obviating their use in mobile environments. Finally UWB is an RF wireless technology, and as such is still subject to the same laws of physics as every other RF technology. Thus, there are obvious tradeoffs to be made in signal-to-

noise ratio versus bandwidth, range versus peak and average power levels, etc.

2.4.13 UWB vs WiFi

Current implementations of WiFi were not designed for streaming audio and video, although they generally work well for the transfer of data.While efforts have been completed by the "e" Task Group to address quality of service (QoS) deficiencies in the 802.11 standard, Wi-Fi is being adapted to support multimedia applications through additions to an already relatively complex MAC. This complexity makes the technology inherently more expensive, and the requirement for backward compatibility places an ultimate limitation on WiFi efficiency and throughput. Also, both 802.11b and 802.11g are subject to interference and oversubscription issues that can result in latency and jitter because they operate in the noisy 2.4GHz ISM band and. Not only are transfer rates slower, topping out at 54Mbps if full data rates could ever be achieved, but power consumption also remains an issue requiring at least an order of magnitude higher than UWB for a comparable file transfer.

However the day is coming fast when home digital equipment will have wireless interfaces, implementing wireless transmission of high-definition television (HDTV) imagery and high-speed swapping of still pictures and audio content with portable gear. There are two major candidates for the wireless interface in home digital equipment: 802.11n, a next-generation wireless local area network (LAN) built on spatial multiplexing using multiple-input, multiple-output (MIMO) technology, and Ultra-WideBand (UWB) wireless technology. 802.11n offers a long range of up to 200m, and is viewed as the most likely contender for the home network backbone. UWB, on the other hand, is likely to make best use of its low-power, high-speed operation in short-range equipment interconnect, such as personal computers (PC) and portable equipment.By using 80% less power than 802.11a, UWB chipsets can work with smaller device such as PDAs and mobile phones without unduly burdening their batteries.

In conclusion,UWB function within a home or office environment is complementary to 802.11 WLAN which is best suited for long-range networking and Ethernet cable replacement. This synergy between WPAN and WLAN allows host connectivity with a plurality of portable peripherals while freeing bandwidth from the WLAN to support robust network connectivity between multiple PCs and intelligent hub devices within the home or office space. As many people believe Wi-Fi and UWB will complement each other rather than compete -- after all, at one time, many thought Bluetooth and 802.11 would fight it out and they now live (somewhat) in harmony. Likewise, 802.11 will most likely remain the preferred home data networking technology, with UWB covering the home multimedia networking arena.Finally 802.11 should also remain as the most effective technology for public access and enterprise markets, where power consumption issues are less important and data is still more important than multimedia.

2.4.14 UWB vs Bluetooth

Although designed as a low power replacement for cable technologies, Bluetooth consumes 50x as much power to transfer a single bit as UWB. The recently released Bluetooth Version 1.2 supports data rates only up to 2 Mbps but has implemented methods for enhanced voice quality and audio.Version 2.0 is under development, promising to extend rates up to 12 Mbps. However, that is still only a fraction of UWB rates. In addition, interference problems with WiFi have been common, since both technologies occupy the same 2.4GHz ISM band, and Bluetooth specifications do not require monitoring for traffic prior to transmitting.Moreover UWB chipsets are built in complementary metal oxide semiconductor, so they rival inexpensive Bluetooth price when produced in volume. As far as the question if UWB will replace Bluetooth is concerned the answer in no. Bluetooth is a complete, end-to-end communication standard. UWB, as a radio technology, can be used as a piece of an overall communication standard. Bluetooth defines how data is managed, formatted and physically carried over a wireless personal area network (WPAN). Ultra-Wideband is a specific type of RF signal that can be used to carry data between devices and it's not a complete communication standard.It is more likely that that UWB and Bluetooth could both be integrated into end-devices to serve different application spaces.

2.4.15 UWB vs WiMAX

There has been discussion of using UWB to provide cheap, fast, last mile wireless access systems, which would solve the interference issues that plague current spread spectrum-based Metropolitan Area Networks (MANs). These UWB systems could be set up in rural areas, bringing never seen before high-speed connectivity to those users.Personal Media Players (PMPs) or Personal Video Players (PVPs) equipped with wimax technology will be capable of downloading content from DVDs, PVRs, etc and some offer the ability to act as a content platform or portable PVR to project to big screen displays. This capability may compete directly with one or more of UWB’s primary target applications. WiMAX-enabled devices would be connected directly to the Internet and share data with other in–premises devices through that link.

2.5 CONCLUSIONS

UWB technology was designed specifically to support high speed, short range, point-to-point wireless communications. Right now the best killer application for UWB is home multimedia networking systems, where high bandwidth is crucial. UWB can support multiple channel multimedia streaming of broadcast quality video, making it the preferred technology to use when setting up a wireless home multimedia network UWB could connect virtually every multimedia device in your home without using any wires. Digital cameras and camcorders could wirelessly stream images and video to your TV or PC, DVD players and TV's could stream videos throughout your home, and flat screen monitors could wirelessly connected to computers, DVD players, or any other source you desire. Further, the industry has endorsed MB-

OFDM and is establishing design guidelines and standards based on this radio platform to ensure interoperability and coexistence between protocols for various interfaces including Wireless 1394, Wireless USB, and native IP-based applications. As WiFi has freed consumers from Ethernet cables, MB-OFDM UWB looks to UWB will very likely revolutionize the home multimedia scene and eliminate the mounds of tangled wires found behind home entertainment centers.Last but not least, intention is not to replace other device connection technologies, like Bluetooth, USB and 1394 but to allow for the migration of Bluetooth, USB and 1394 protocols in such a manner that a single radio can support each of these protocols. The following diagrams highlight the main differences between UWB and other technologies regarding data rate,range and spectrum utilization (see Figures 3,4,5). Finally table 1 gives a summary of the main features for the most important wireless technologies.

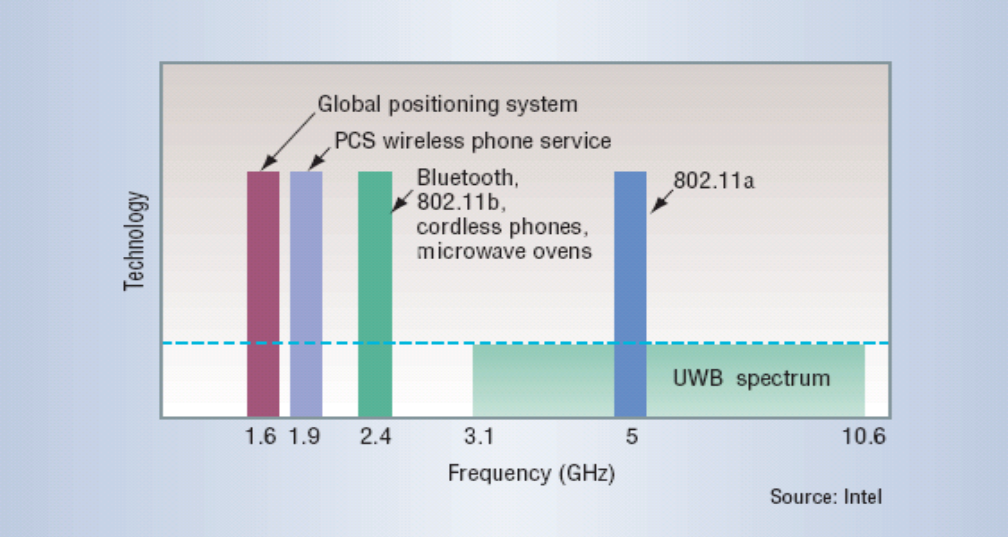


Figure 3 .Unlike conventional radio systems, which operate within a relatively narrow bandwidth, ultra-wideband operates across a wide range of frequency spectrum by transmitting a series of extremely narrow (10 - 1000ps) and low power pulses.

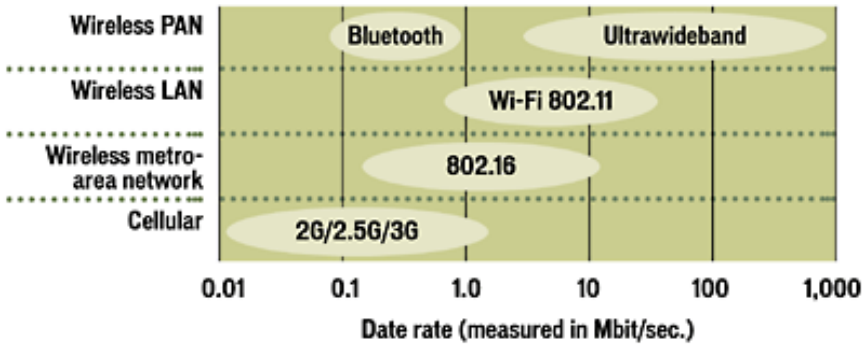


Figure 4.A technology comparison diagram regarding data rate

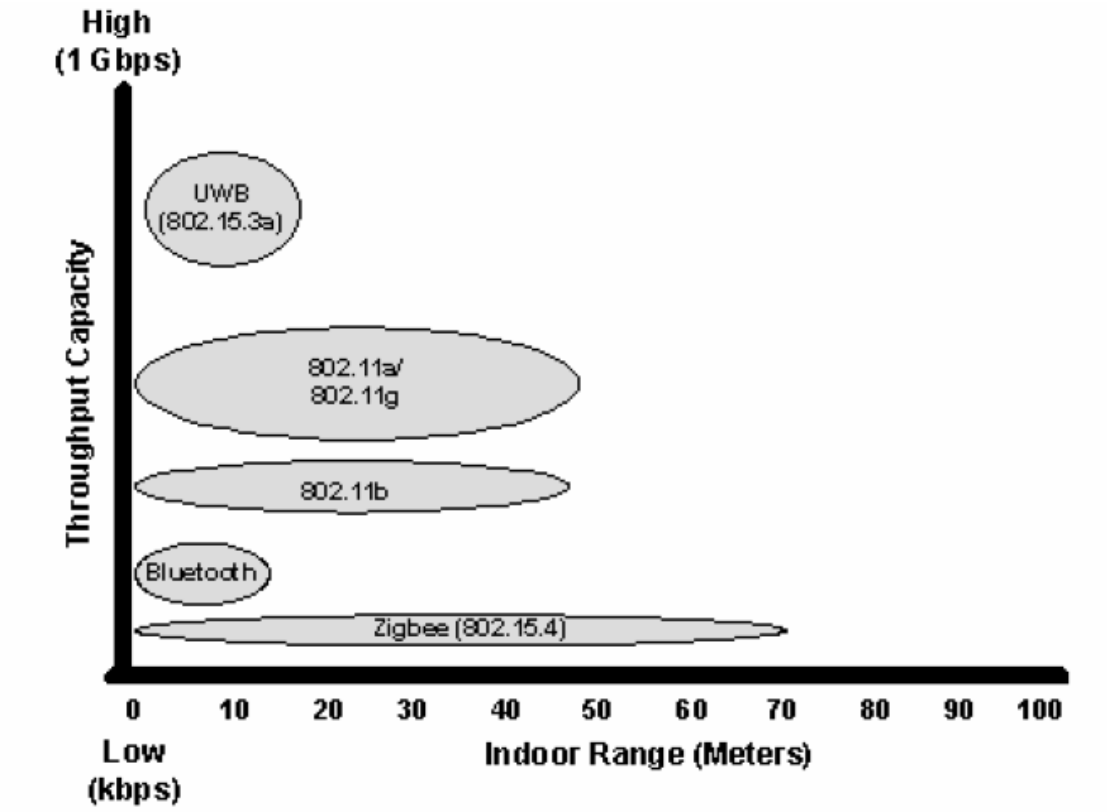


Figure 5.A technology comparison diagram regarding range and throughput capacity

Table 1:comparison between UWB and other wireless standards

Technology	Data Rate	Range	Cost	Power	Spectrum	Issues
UWB(802.15.3a)	**50-500 Mb/s**	**10-30 meters**	**Low**	**Low**	**3.1-10.6 GHz**	**High data rate for short range only**
Bluetooh	0.8-1.0 Mb/s	10 meters	Low	Low	2.4 GHz	Speed and Interference issues
802.11a	54 Mb/s	up to 50 meters	High	High	5.0 GHz	High Power consumption,High cost
802.11b	11 Mb/s	up to 50 meters	Medium	Medium	2.4 GHz	Speed and signal strength issues for more range
802.11g	54 Mb/s	up to 50 meters	High	High	2.4 GHz	Connectivity and range problems High cost
HyperLan2	Up to 54 Mb/s	50-100 meters	High	High	2.4 GHz	Only European standard,High cost.
WiMAX(802.16)	Up to 70 Mb/s	50 Km	High	Low	10-66GHz	Delays in regulatory approval in Europe due to issues regarding the use of the spectrums in the 2.8 GHz and 3.4 GHz range for the newly completed 802.16a standard that operates between 2 and 11GHz and does not need line of sight
ZigBee(802.15.4.a)	0.02-0.2 Mbits/s	less than 10 meters	Low	Low	2.4 GHz	Standard still under consideration very less communication range, law data-rate

CHAPTER 3 SYSTEM OVERVIEW

In this chapter the basic concept of OFDM is explained.We focus on the transmitter since the transmitter side of the UWB MB-OFDM system is implemented in this thesis.Also an overview of the Multiband OFDM standard for the IEEE 802.15.3 Task Group 3a is presented and its parameters are pointed out.

3.1.BASIC THEORY OF OFDM

Orthogonal frequency division multiplexing (OFDM) is a multi-carrier transmission technique that has been recently recognized as an excellent method for high speed bi-directional wireless data communication. Although the principles of orthogonal frequency division multiplexing (OFDM) modulation and some of the benefits have been known since 1960, it has recently become popular because of the low cost and availability of integrated circuits that can perform the high speed digital signal processing operations.In recent years, these techniques have quickly put into practise in modern communications systems and employed in data delivery systems over the phone line, digital radio and television, and wireless networking systems.

Orthogonal frequency-division multiplexing (OFDM), also sometimes called discrete multitone modulation (DMT), is a complex modulation technique for transmission based upon the idea of frequency-division multiplexing (FDM.),which is a technology that uses multiple frequencies to simultaneously transmit multiple signals in parallel. Frequency Division Multiplexing (FDM) extends the concept of single carrier modulation by using multiple subcarriers within the same single channel. (A typical single-carrier modulation spectrum is shown in Figure 6). A single carrier system modulates information onto one carrier using frequency, phase, or amplitude adjustment of the carrier. On the contrary in FDM the communication channel is divided into a number of equally spaced frequency bands. A subcarrier carrying a portion of the user information is transmited in each band. FDM systems usually require a guard band between modulated subcarriers to prevent the spectrum of one subcarrier from interfering with another. In OFDM the frequencies and modulation of FDM are arranged to be orthogonal with each other(independent of each other) which almost eliminates the interference between channels. In other words each subcarrier is orthogonal with every other subcarrier. The use of orthogonal subcarriers would allow the subcarriers' spectra to overlap, thus increasing the spectral efficiency. As long as orthogonality is maintained, it is still possible to recover the individual subcarriers' signals despite their overlapping spectrums (see figure 7).

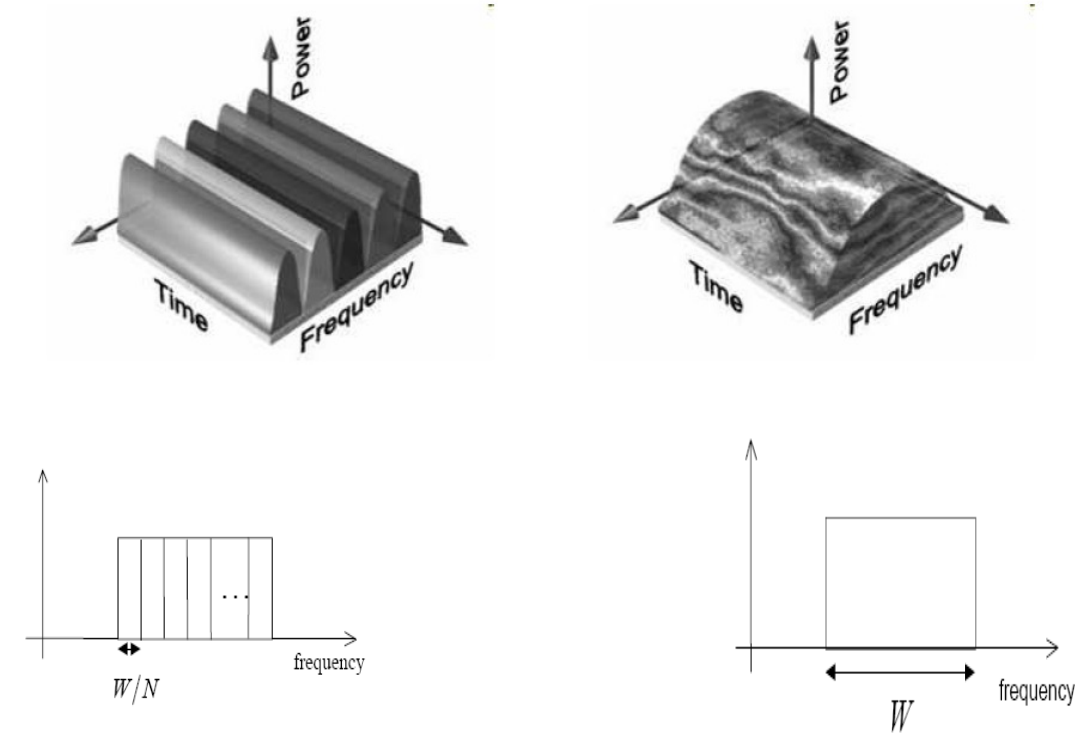


Figure 6: Multi-Carrier Modulation **Single-Carrier Modulation**

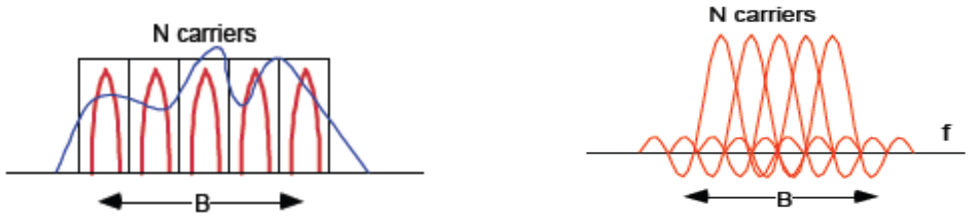


Figure 7: FDM **OFDM**

Since low-rate modulations (i.e modulations with relatively long symbols compared to the channel time characteristics) are less sensitive to multipath, it should be better to send a number of low rate streams in parallel than sending one high rate waveform. This is exactly what OFDM is doing. It divides the available frequency spectrum in subbands small enough so that the channel effects are constant (flat) over a given subband. An OFDM system takes a data stream and splits it into N parallel data streams,each at a rate 1/N of the original rate. Each stream is then mapped to a tone at a unique frequency and combined together using the inverse fast forward fourier transform (IFFT) to yield the time-domain waveform to be transmited. The information is modulated onto a tone by adjusting the tone's phase,amplitude or both.In the most basic form,a tone may be present or disabled to indicate a one or zero bit of information,however,either phase shift keying (PSK) or quadrature amplitude modulation(QAM) is typically employed.The following figure illustrates the OFDM signal spectrum.

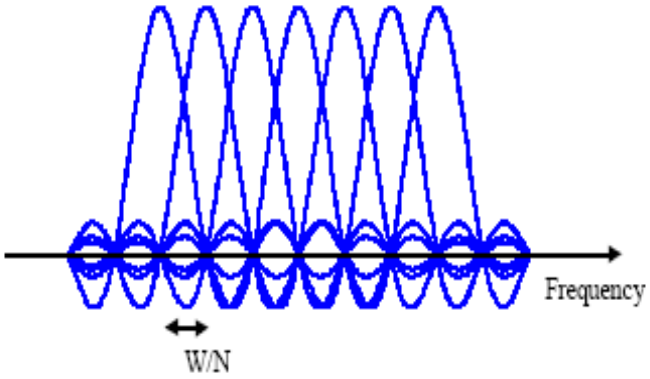


Figure 8.OFDM signal spectrum

In practice, OFDM systems are implemented using a combination of fast Fourier Transform (FFT) and inverse fast Fourier Transform (IFFT) blocks that are mathematically equivalent versions of the DFT and IDFT, respectively, but more efficient to implement. An OFDM system treats the source symbols (e.g., the QPSK or QAM symbols that would be present in a single carrier system) at the transmitter as though they are in the frequency-domain. These symbols are used as the inputs to an IFFT block that brings the signal into the time domain. The IFFT takes in N symbols at a time where N is the number of subcarriers in the system. Each of these N input symbols has a symbol period of T seconds. Recall that the basis functions for an IFFT are N orthogonal sinusoids. These sinusoids each have a different frequency and the lowest frequency is DC. Each input symbol acts like a complex weight for the corresponding sinusoidal basis function. Since the input symbols are complex, the value of the symbol determines both the amplitude and phase of the sinusoid for that subcarrier. The IFFT output is the summation of all N sinusoids. Thus, the IFFT block provides a simple way to modulate data onto N orthogonal subcarriers. The block of N output samples from the IFFT make up a single OFDM symbol. The length of the OFDM symbol is NT where T is the IFFT input symbol period mentioned above. At the receiver, an FFT block is used to process the received signal and bring it into the frequency domain. Ideally, the FFT output will be the original symbols that were sent to the IFFT at the transmitter.

The FFT and the IFFT of length N, is defned by the equations

$$X_k = \sum_{i=0}^{N-1} x_i W_N^{ki}, \quad 0 \le k < N$$

$$x_i = \frac{1}{N} \sum_{k=0}^{N-1} X_k W_N^{-ki}, \quad 0 \le k < N,$$

where $W_N = e^{-j\frac{2\pi}{N}}$
is called twiddle factor, *i* is the sample index, and *k* is the subcarrier index.

The OFDM time-domain samples at the output of IFFT are subsequently guard padded by enough samples to eliminate the interference between adjacent OFDM symbols caused by the delay-spread of the multipath channel. Hence, for each OFDM symbol, some of the time-domain samples of the OFDM symbol are cyclically added from its end to the beginning. Figure 9 indicates how the channel's delay spread causes Inter-Symbol Interference (ISI) and how the cyclic prefix (CP) eliminates this ISI. Interference from the previous symbol will only affect the samples in the CP so alternatively zero prefix and zero suffix (in the case of MBOA standard) can be used instead of copying the first *n* samples from the IFFT and placed them in the end of the OFDM frame. But there is another reason to insert a CP and then discard it at the receiver. Assuming that the CP is longer than the channel impulse response, the convolution between the data and the channel impulse response will act like a circular convolution and therefore no ICI (Inter-Channel Interference) will occur.

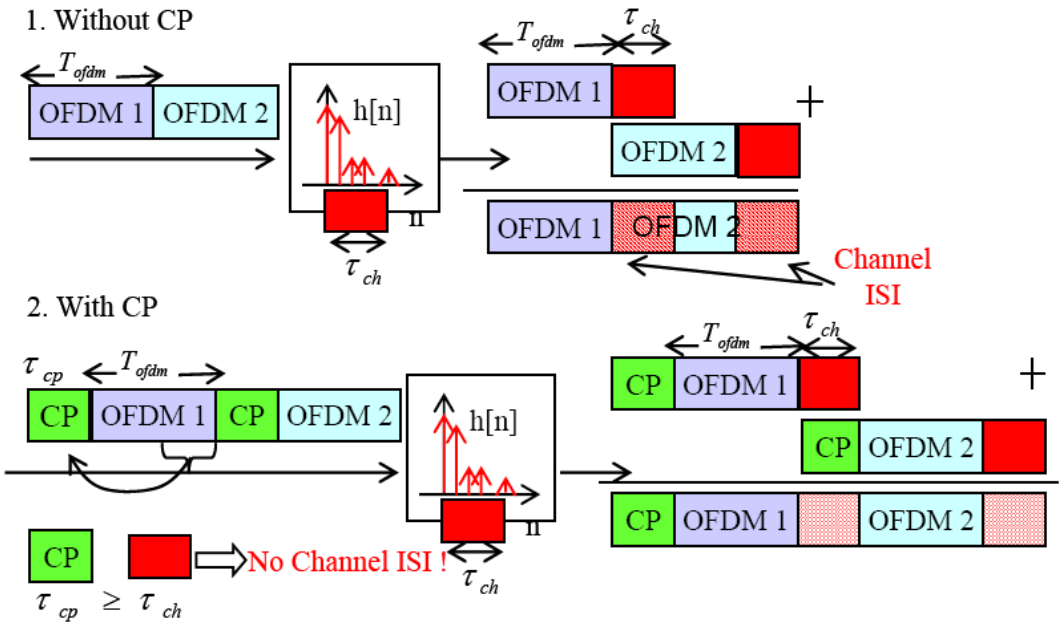


Figure 9: Cyclic prefix (CP) eliminates the channel Inter-Symbol Interference (ISI).

The typical OFDM system process is illustrated in figure 10 below.

- The incoming serial data is first converted from serial to parallel and grouped into x bits each to form a complex number. The complex numbers are modulated in a baseband fashion by the IFFT. And converted back to serial data for transmission.
- A guard interval is inserted between symbols to avoid intersymbol interference (ISI) caused by multipath distortion. The discrete symbols are converted to analog and lowpass filtered for RF up-conversion.
- The receiver performs the inverse process of the transmitter. One tap equalizer is used to correct channel distortion. The tap coefficients of the filter are calculated based on channel information.

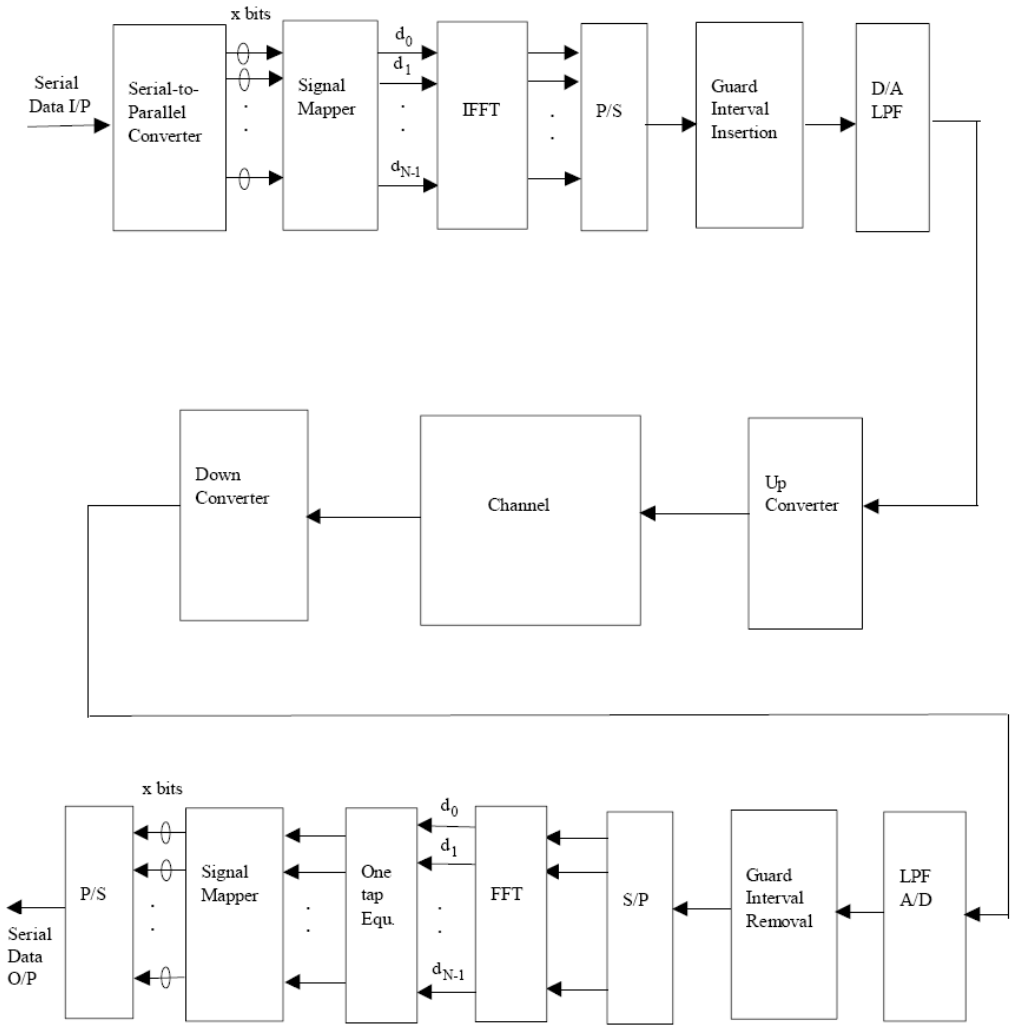


Figure 10.General OFDM system

3.2.IEEE 802.15.3a MultiBand OFDM Physical Layer

A number of important parameters are identified in the previous section such as the number of subcarriers which will decide the size of the FFT,the number of constellations,the bandwidth which corresponds to the sample time (T_s) and others. All these are defined by Multiband OFDM proposal for the implementation of the UWB OFDM physical.An overview of this proposal for IEEE 802.15 Task Group3a is presented first and afterwards various parameters of the protocol.

3.2.1 Introduction to MultiBand OFDM proposal

Multiband OFDM Physical Layer proposal for IEEE 802.15.3a specifies the PHY entity for a UWB system that utilizes the unlicensed 3.1 – 10.6 GHz. The UWB system provides a wireless PAN with data payload communication capabilities of 53.3, 80, 110, 160, 200, 320, 400, and 480 Mb/s. **The support of transmitting and receiving at data rates of 53.3, 110, and 200 Mb/s is mandatory**. The proposed UWB system employs orthogonal frequency division multiplexing (OFDM) splitting an information signal across a total of 122 modulated and pilot subcarriers using a total of 128 subcarriers. Twelve of the subcarriers are dedicated to pilot signals in order to make coherent detection robust against frequency offsets and phase noise and ten of the subcarriers at the edges of the occupied frequency band shall are guard subcarriers to allow for relaxing the specs on transmit and receive filters as well as possible performance improvements. The 100 data subcarriers are modulated using quadrature phase shift keying (QPSK).Forward error correction coding (convolutional coding) is used with a coding rate of 1/3, 11/32, 1/2, 5/8, and 3/4.

The transmitted RF signal is related to the baseband signal as follows:

$$r_{RF}(t) = \mathrm{Re}\left\{\sum_{k=0}^{N-1} r_k(t - kT_{SYM}) \exp(j2\pi f_{(k \bmod 6)} t)\right\},$$

where Re(·) represents the real part of a complex variable, $r_k(t)$ is the (possibly complex) baseband signal representing the kth OFDM symbol occupying a symbol interval of length T_{SYM}, and N is the number of OFDM symbols transmitted. The carrier frequency or band that the kth OFDM symbol is transmitted over is denoted as f_k.The exact structure of the *k*th OFDM symbol depends on its location within the packet:

$$r_k(t) = \begin{cases} r_{preamble,k}(t) & 0 \le k < N_{preamble} \\ r_{header,k-N_{preamble}}(t) & N_{preamble} \le k < N_{header} \\ r_{data,k-N_{preamble}}(t) & N_{header} \le k < N_{data} \end{cases}.$$

The structure of each component of , $r_k(t)$ as well as the offsets Npreamble, Nheader, and Ndata will be described in more detail later.All of the OFDM symbols $r_k(t)$ can be constructed using an inverse Fourier transform with a certain set of

coefficients Cn, where the coefficients are defined as either data, pilots, or training symbols:

$$r_k(t) = \begin{cases} \sum_{n=-N_{ST}/2}^{N_{ST}/2} C_n \exp(j2\pi m\Delta_f)(t) & t \in [0, T_{FFT}] \\ 0 & t \in [T_{FFT}, T_{FFT} + T_{ZP}] \end{cases}$$

The parameters Δf and N_{ST} are defined as the subcarrier frequency spacing and the number of total subcarriers used, respectively. The resulting waveform has a duration of $T_{FFT} = 1/\Delta f$. The time parameter T_{ZP} specifies a zero pad period for the OFDM symbol which is used to mitigate the effects of multipath as well as to provide a guard period to allow for switching between the different bands.

3.2.2 Multi-band OFDM PLCP sublayer

The PHY Sublayer Service Data Units (PSDU) of the Multi-band OFDM is converted to a PLCP Protocol Data Unit (PPDU). The PSDU is provided with a PLCP preamble and header to create the PPDU. At the receiver, the PLCP preamble and header are processed to aid in demodulation and delivery of the PSDU.The PPDU format of the is shown in figure 11 and it includes:

- the PLCP preamble: A standard or a streaming mode PLCP preamble shall be added prior to the PLCP header to aid receiver algorithms related to synchronization, carrier-offset recovery, and channel estimation (see chapter 5 section 10). Standard preamble contains 30 OFDM symbols (9.375 us duration) while optional streaming mode preamble contains 18 symbols (5.625 us duration).

- PLCP header includes
    1. the PHY header : 40 bits. Contains payload information-data rate and length(bytes).
    2. MAC header:80 bits.Combined with the PHY header the HCS is calculated.
    3. header check sequence(HCS): 16 bits used to detect Header errors.
    4. tail bits: 6 bits initialize the convolutional encoder to the “zero state”.
    5. pad bits. 52 bits to align the data stream on the OFDM symbol interleaver boundaries.

  The PHY layer first pre-appends the PHY header to the MAC header and then calculates the HCS over the combined headers. Six Tail bits are then inserted after the PHY header in order to return the convolutional encoder to the “zero state”. The resulting HCS is appended to the end of the MAC header along with an additional set of 6 tail bits. 52 Pad bits are finally added to the end of the tail bits in order to align the data stream on the OFDM symbol interleaver boundaries.PLCP Header contains 12 symbols (3.75 us duration).

- MAC frame body include:
    1. frame payload: Data.1 to 4095 bytes transmitted at 53-480 Mbits/sec.
    2. frame check sequence(FCS) :16 bits used to detect Payload errors.
    3. tail bits: 6 bits initialize the convolutional encoder to the “zero state”.
    4. Pad bits:Variable length. The number of pad bits that shall be inserted is a function of the code rate R and the number of bits in the frame payload. Pad bits ensures that the resulting frame boundaries are aligned with the boundaries of the OFDM symbol interleaver.

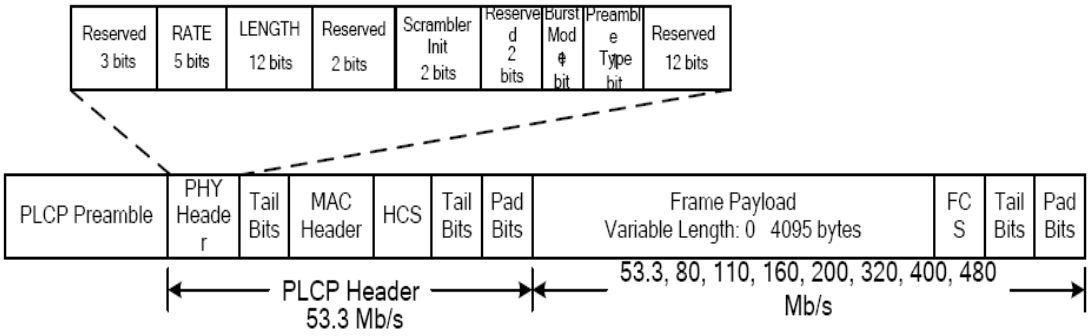


Figure 11.PLCP frame format

The PLCP preamble is sent first, followed by the PLCP header, followed by an optional band extension sequence, followed by the frame payload, the FCS, the tail bits, and finally the pad bits. As shown in Figure 11, the PLCP header is always sent at an information data rate of 53.3 Mb/s. The remainder of the PLCP frame (frame payload, FCS, tail bits, and pad bits) is sent at the desired information data rate of 53.3, 80, 110, 160, 200, 320, 400 or 480 Mb/s.

3.2.3 PHY header

The PHY header field shall be composed of 40 bits, as illustrated in Figure 12.

- Data Rate(RATE).Depending on the information data rate (RATE), bits R1–R5 shall be set according to the values in Table 2.
- PLCP length field (LENGTH): an unsigned 12-bit integer that indicates the number of bytes in the frame payload (which does not include the FCS, the tail bits, or the pad bits).
- PLCP scrambler field (SCRAMBLER):The MAC shall set bits S1–S2 according to the scrambler seed identifier value. This two-bit value corresponds to the seed value chosen for the data scrambler.
- Burst Mode (BM) field:Bit 26 shall encode whether the packet is being transmitted in “burst” or streaming mode.
- Preamble Type (PT) field: Bit 27 shall encode the preamble type (long or short preamble) used in the next packet if in streaming mode.

- The remaining 12 bits which are not defined in his section shall be understood to be reserved for future use and are set to zero.

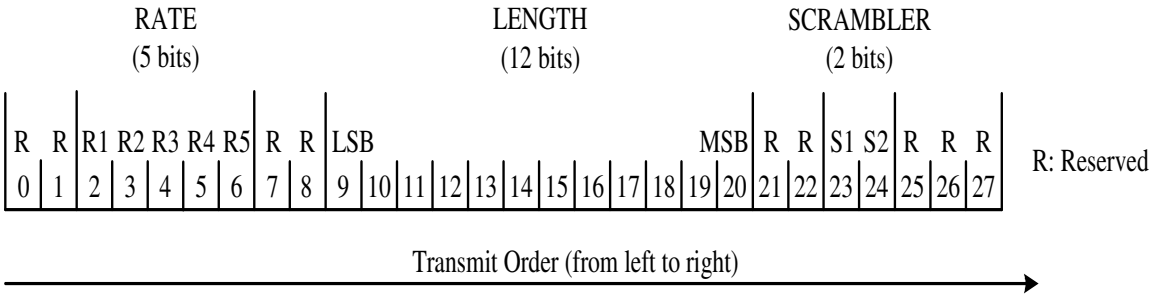


Figure 12 – PHY Header bit assignment

Table 2 – Rate-dependent parameters

Rate (Mb/s)	R1 – R5
53.3	00000
80	00001
110	00010
160	00011
200	00100
320	00101
400	00110
480	00111
Reserved	01000–11111

The number of bits in the DATA field is a multiple of N_{CBPS}, the number of coded bits in an OFDM symbol (100 or 200 bits). To achieve that, the length of the message is extended so that it becomes a multiple of N_{DBPS}, the number of data bits per OFDM symbol. At least 6 bits are appended to the message, in order to accommodate the tail bits. The number of OFDM symbols, N_{SYM} and the number of pad bits in N_{PAD}, are computed from the length of the PSDU (LENGTH) as follows:

$Npad=N_{DATA}–(8*(LENGTH)+FCS+tail\ bits)$

$N_{SYM} = Time\_Spreading\_Factor \times 6\times [1/R \times (8 \times bytes + FCS\_bits + pad\_bits)]/(6\times N_{CBPS})]$

The data rate-dependent modulation parameters are listed in Table 3 and the timing parameters associated with the OFDM PHY are listed in Table 4.

Table 3 – Rate-dependent parameters

Data Rate (Mb/s)	Modulation	Coding rate (R)	Conjugate Symmetric Input to IFFT	Time Spreading Factor (TSF)	Overall Spreading Gain	Coded bits per OFDM symbol (N_{CBPS})
53.3	QPSK	1/3	Yes	2	4	100
80	QPSK	1/2	Yes	2	4	100
110	QPSK	11/32	No	2	2	200
160	QPSK	1/2	No	2	2	200
200	QPSK	5/8	No	2	2	200
320	QPSK	1/2	No	1 (No spreading)	1	200
400	QPSK	5/8	No	1 (No spreading)	1	200
480	QPSK	3/4	No	1 (No spreading)	1	200

Table 4 – Timing-related parameters

parameter	value
N_{SD}: Number of data subcarriers	**100**
N_{SDP}: Number of defined pilot carriers	**12**
N_{SG}: Number of guard carriers	**10**
N_{ST}: Number of total subcarriers used	**122 (= N_{SD} + N_{SDP} + N_{SG})**
Δf: Subcarrier frequency spacing	**4.125 MHz (= 528 MHz/128)**
T_{FFT}: IFFT/FFT period	**242.42 ns (1/Δf)**
T_{ZP}: Zero pad duration	**70.08 ns (= 37/528 MHz)**
T_{SYM}: Symbol interval	**312.5 ns (T_{ZP} + T_{FFT} + T_{GI})**

- **Total subcarriers=(100 data+12 pilot+10 guard+6 null)samples+32 prefix+5 guard =128**
- **TFFT=128/528 MHz=242.42 ns**
- **Zero Prefix duration=32/528 MHz = 60.6 ns**
- **Guard duration=5/528 MHz=9.5 ns**
- **T_{SYM} = 242.42 ns + 60.6 ns + 9.5 ns=312.5 ns**

The data rate is defined by the following equations:

Data Rate=Symbol_Rate * Bits_per_Symbol * Coding_Rate/Spreading

Symbol_Rate= 528 MHz / 165 samples= 3.2 Msps
Bits_per_Symbol= 200 ,100 QPSK symbols or
100 , 50 QPSK symbols

3.2.4 System model

The individual blocks of UWB system model which perform the digital signal processing are shown in the following figure. In chapter 5 ,the purpose, the functionality and the design of every block is explained. In this thesis only the digital part of the UWB transmitter has been designed.

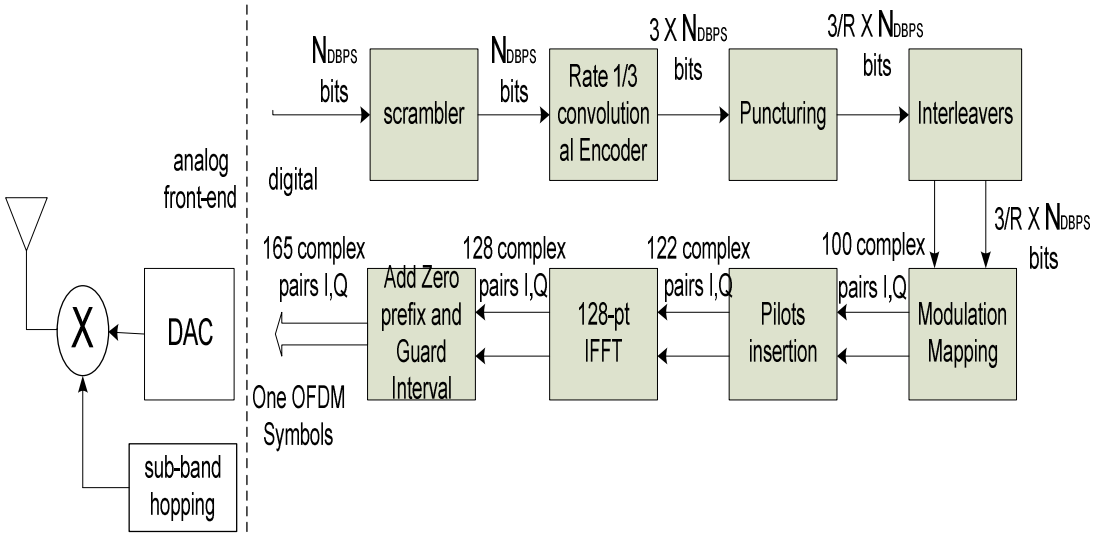


Figure 13.UWB baseband transmitter

CHAPTER 4 SYSTEM DESIGN

4.1.DESIGN FLOW

The design approach for the UWB transmiter is illustrated in figure 14. Early in the development cycle, different communication and signal processing algorithms are evaluated for their performance under different conditions like noise, multipath channel and radio non-linearity. First, a foating point model was specified using Matlab and Simulink .After functionality has been verified, a fixed point model is implemented.UWB Fixed-Point Model (Multiband OFDM) released by Mathworks was used as a reference model for our hardware design implementation.The importance of this model is that not only provides the functionality of the signal processing algorithms that are used for the hardware design but also defines the format or the representation of data. Floating point implementation results in higher hardware costs and additional circuits related with normalizing of numbers.Floating point representation is useful when dealing with data of different ranges. However, this is not true as the Baseband circuits have a fair idea of the range of values that they will work on. So a fixed-point representation will be more efficient. Further in fixed point a choice can be made between signed and 2's complement representation.We have selected 2's complement representation.Using the Ultrawideband fixed-point model(multiband OFDM) required wordlengths are determined to reach a certain signal to noise ratio. The wordlength is a critical parameter that affects speed, power, and area, and must thus be chosen with care. Simulations for different word lengths and scaling factor for the fixed-point representation of data are performed to determine which is the optimum word length and scale factor that maintain the required level of precision.16 bits was selected for implementation to have a good performance in all conditions.Since the UWB baseband was designed with tools like Matlab and Simulink,a verification mechanism which ensures that the hardware implementation (RTL) is same as the UWB simulink-model is crucial in the design flow.For this reason test vectors for later use are extracted from the UWB fixed-point model.

At this point, sufficient knowledge about the design obtained to start with an architectural description considering the restrictions that we posed regarding area,power and speed for the hardware implementation.The architecture is usually described using VHDL or Verilog which is a hardware description language (HDL).After the architecture is implemented in an HDL, functionality is verified with the generated test vectors in a simulation program.The next step is to synthesize the design into a netlist of components. After synthesis a first estimate on speed, area, and power consumption is obtained and post synthesis simulation is performed to make sure that the design still functions correctly. If the design does not meet the specified constraints, either the component library or the architecture has to be changed and the design re-synthesized. Finally, it is time to place and route the design. In this stage the physical chip layout is generated and interconnections between cells are routed. The place and route tool can automatically place cells either in a timing driven, connection driven, or random way but larger components like memories and pads are best placed manually.After this stage the final area is known and information about the actual location of the cells can be used to obtain a more accurate timing estimation that is

used in the post layout simulation. If the specifications are met,it is time for tape out, i.e. to send the design for fabrication.

Specifically,for our UWB transmitter implementation,VHDL was used to describe the hardware architecture.Matlab and Simulink was used for the fixed point model. Synthesis is performed with Xilinx ISE 7.1 Design Compiler on the VHDL model and all simulations are performed with Mentor Graphics: Modelsim. Place and route is done again with Xilinx ISE 7.1.

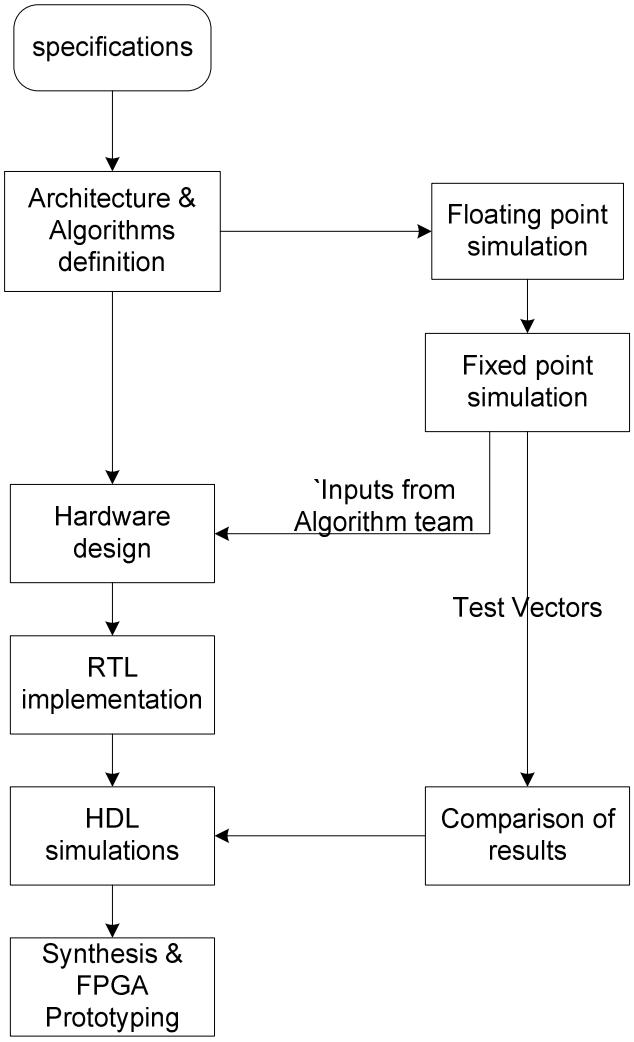


Figure 14: Design flow

4.2 SPECIFICATIONS

Our implementation aims to support the mandatory data rates of 53.3 Mbits/sec(for the PLCP header transmition) 110 and 200 Mbits/sec(for Payload transmition).The modulation parameters corresponding to these data rates are shown in Table 5.

Table 5. Modulation Parameters Corresponding to Data Rates.

Data Rate (Mb/s)	Modulation	Coding Rate (R)	Conjugate Symmetric Input to IFFT	Coded bits /OFDM symbol (N_{CBPS})
53.3	QPSK	1/3	Yes	100
110	QPSK	11/32	No	200
200	QPSK	5/8	No	200

The main goal of our design is to achieve the necessary throughput,to meet the mandatory data rates of 53.3,110,200 Mbits/sec.However there is a design trade-off between power, throughput, and area.If throughput is increased area and/or power will also increase.Since the speed requirements of Ultrawideband baseband are very demanding a first goal is to explore how we can achieve performance as close as possible to the specifications.Future work will put emphasis on power aspects and power optimizations.Another goal for the transmitter implementation is low cost.Initially for this reason Spartan-3 FPGA platform was used which is very cheap.However we noticed that the required performance depends on the digital baseband clock frequency which is the same with the signal bandwidth or sample frequency.In UWB case with bandwidth of 528 MHz or higher this is not reliazable with Spartan-3 FPGA platform.Therefore the selected FPGA platform should be Virtex-4 or even Virtex-5 to get a real-time speed in hardware according to the specifications.

4.2.1 Simulink UWB fixed point model features

4.2.1.1 Summary of model

- Based on IEEE 802.15.3a proposal (Sept 2003 version).
- End-to-end physical layer (streaming mode).
- Highest mandatory data rate (200 Mb/s).
- QPSK modulation, rate-5/8 forward error correction coding (convolutional + puncturing).
- OFDM transmission: 122 subcarriers, 22 pilots, 128-pt FFTs, zero prefix, guard period.
- Fixed-point model of OFDM TX/RX (16-, 12-, 10-, and 8-bit).
- 2x spreading across sub-bands.
- Data interleaving.
- PLCP preamble (packet sync, frame sync, and channel estimation).

- Viterbi decoding.
- Mode 1 frequency hopping (3 bands).

4.2.1.2 Model assumptions

- Baseband-equivalent model (no up/down RF conversion).
- Random data transmision (no data scrambling used).
- Fixed (selectable) number of data symbols per packet (no pad bits used).
- Continuous frame-to-frame operation (no coder state resetting via tail bits).
- Fixed transmit power level; link-SNR specified (on-the-fly).
- Idealized timing/frequency acquisition.
- Not modeled:
    - Other mandatory and optional data rates (55, 80, 110, 160, 200, 320, 480 Mb/s)
    - MAC/PHY interface and PLCP header (TXVECTOR/RXVECTOR, HCS/FCS)
    - Time windowing of OFDM symbols

The reference UWB MultiBand OFDM fixed-point model has some disadvantages regarding the specifications we have defined for our hardware design.First of all it does not support all the mandatory data rates that we are intended to put into practice.Scrambling is not used and also is based on an older version of IEEE 802.15.3a proposal.Finally PLCP Header and FCS,HCS are not modeled.For this reason we made the appropriate modifications at some of the blocks of the Simulink UWB model to adjust it to the most recently version that is MBOA-OFDM Physical Layer Proposal for IEEE 802.15.3a,September 14,2004.We added a scrambler and descrambler block and finally we manage to have a design that meets our functional requirements for the 200 Mb/s mode.

Moreover to obtain a functional description for the other two modes,rate 110 and 53.3 Mb/s we have used the same UWB simulink model modifying the parameters in the convolutional encoder and the puncture block in order to extract test vectors ideal for these two cases.

4.2.2 Hardware design assumptions

- Preamble: Only standard PLCP Preamble type (time-domain preamble pattern 1 and frequency domain training sequence results provided by the MBOA standard document).
- Data Padding: We assume all data padding and tail bits are added in the MAC layer.
- The formation of the PLCP Header (PHY header,tail bits,MAC header, HCS,tail bits,pad bits) and PLCP payload (Frame payload,FCS,tail bits,pad bits) is done by the MAC layer.Information bits are processed to the UWB digital transmited in the order according to the PLCP frame (see figure 11).
- HCS(Header Check Sequence) and FCS(Frame check Sequence) are calculated by the MAC Layer.
- No time-spreading across subbands.

- Continuous frame-to-frame operation (no coder state resetting via tail bits).
- Fixed-selectable number of OFDM symbols calculated by MAC layer.
- PHY/MAC interface not modeled.

CHAPTER 5 HARDWARE DESIGN

5.1 SCRAMBLER

Scrambling is used for Data Encryption making the transmition of data bits more secure.This is achieved by randomizing data and reducing the length of strings of 0s or 1s in the transmitted signal, since a long string of 0s or 1s may cause transmission synchronization problems.

A side-stream scrambler is used to randomize data for the the PLCP header and the entire PLCP frame body. The preamble symbols and the PHY header left unscrambled to allow direct correlation and detection while the rest of PLCP frame is scrambled (the combination of the MAC header, HCS, the tail bits following it and the pad bits as well as the frame payload**).**

The polynomial generator, $g(D)$, for the pseudo-random binary sequence (PRBS) is $g(D) = 1 + D^14 + D^15$, where D is a single bit delay element. This polynomial not only forms a maximal length sequence, but is also a primitive polynomial.A simple implementation of the scrambler consists of a 15-bit Linear Feedback Shift Register (LFSR) and 2 XORS as shown in figure15. The15 binary storage elements are implemented using D flip-flops.

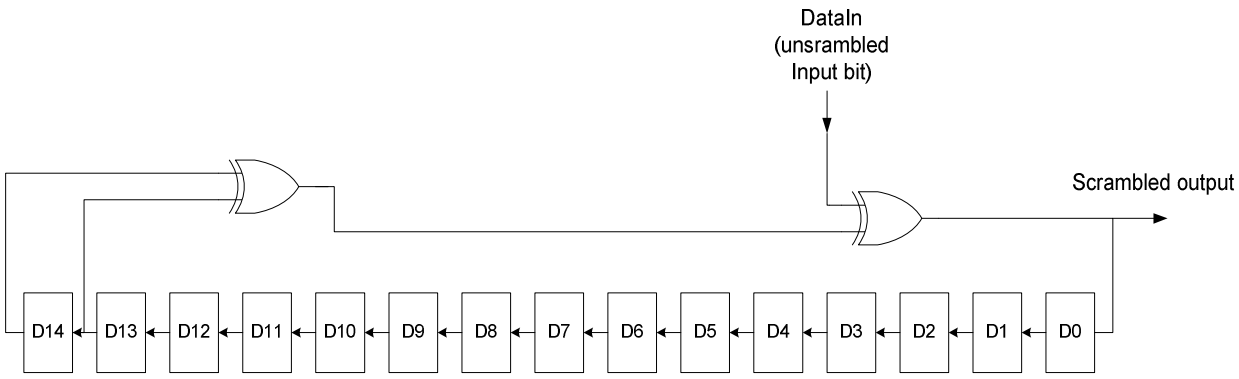


Figure 15.Data scrambler

The two 2-input,1-output XOR gates perform modulo-2 addition according to the generator polynomial. In this way the corresponding pseudo-random binary sequence PRBS,Xn,is generated as :

$$Xn=Xn\text{-}14 \oplus Xn\text{-}15 \ , \ n=0,1,2,...$$

where "$\oplus$" denotes modulo-2 addition.

So the scrambled data bits,Sn,are obtained as:

$$Sn=Xn \oplus dn \ , \ n=0,1,2,..$$

where dn represents the unscrambled data bits**.**

In other words bits 14 and 15 are XORed together and the result is XORed with the incoming data bit. Thus each incoming data bit is XORed with the current bit in the generated pseudo-random sequence of the scrambler and the result is output.As a consequence the input bit stream is randomized.

The shift register is initialized according to the seed identifier value contained in the PLCP header of the transmited frame. This value of the seed identifier shall be used to initialize the scrambler for both the PLCP header and the frame body. The 15-bit initialization vector shall correspond to the seed identifier as shown in the following Table 6 .

Table 6. scrambler seed selection

Seed identifier (s1,s2)	Initiliazation vector (X14 X13 ….X1 X0)
0,0	111111111111100
0,1	111111111111110
1,0	111111111111101
1,1	111111111111111

.

The signal description of the scrambler and the interface diagram as our VHDL implementation indicates are shown below:

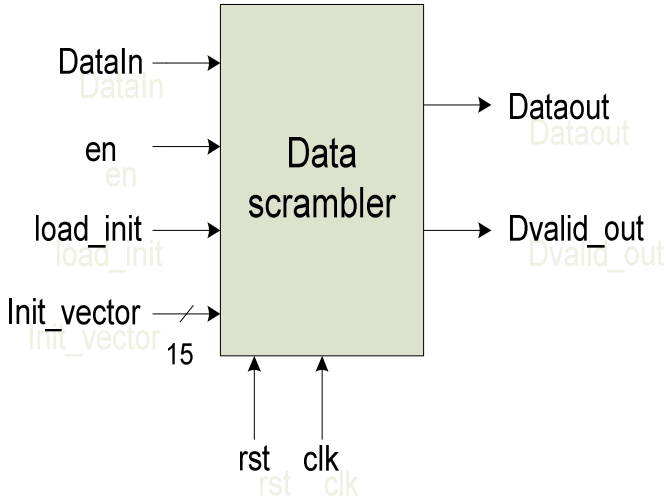


Figure 16. Interface Scrambler

Table 7. Scrambler signal description

signal	direction	width	description
clk	input	1	Global clock
rst	input	1	Asynchronous active high global reset
DataIn	input	1	Input data bit that is to be scrambled
en	input	1	Enables or disables the scrambling,when disabled the scrambler ignores the incoming input bits,remains in the current state and gives the output unchanged(not valid data)
load_init	input	1	When assert the initialization vector is valid and is loaded into the shift register of the scrambler.If low and scrambler is enabled valid scrambled data are generated
Init_vector	input	15	Initialization vector of the scrambler that defines the initial state of the scrambler
Dataout	output	1	Scrambled output data bits
Dvalidout	output	1	When high data bit on output is valid

The valid data stream that is generated by the scrambler feds the following encoder**.**

5.2.CONVOLUTIONAL ENCODER

A common design goal for a digital communication system is to transmit information through a channel and receive it with as few errors as possible. One technique which allows for "automatic" error correction is *forward error correction(FEC)* and a specific method to achieve that incorporates convolutional encoding with Viterbi decoding. Convolutional encoding is one way of performing channel coding in which redundant bits are used to help determine the occurrence of an error due to noise present on the transmission channel.Viterbi decoding is a way of performing channel decoding.In these methods errors can be "automatically" corrected (with some limitations) to recover the original information.

The Channel Encoder is utilised in the transmission path as a convolutional encoder which adds redundant information to the input data being transmited through the channel compensating for channel errors,providing improved error performance and making the wireless transmission more robust as a reliable wireless digital communication system demands. A convolutional encoder is made of a fixed numbers of shift registers. Each input bit enters a shift register and the output is derived by combining the bits of the shift register. The number of output bits depends on the number of modulo-2 adders used with the shift registers. Convolutional codes are usually described using two parameters:

- the code rate R=k/n
- the constraint length K=k(m+1)
- memory length m

The code rate, k/n, is expressed as a ratio of the number of bits iput into the convolutional encoder (k) to the number of channel symbols output by the convolutional encoder (n) in a given encoder cycle. The constraint length parameter, K, denotes the "length" of the convolutional encoder, i.e. how many k-bit stages are available to feed the combinatorial logic that produces the output symbols. Closely related to K is the parameter m, which indicates how many encoder cycles an input bit is retained and used for encoding after it first appears at the input to the convolutional encoder. The m parameter can be thought of as the memory length of the encoder.In addition the generator polynomial (g) characterizes the encoders connection. The selection of which bits(in the memory registers) are to be added(using modulo-2 adders) to produce the output bit is determined from the generator polynomial of that bit.

Specifically for the present implementation the convolutional encoder shall use the rate R = 1/3 code with generator polynomials, $g0 = 133$, $g1 = 165$, and $g2 = 171$, (octal definition) as shown in Figure 17. The bit denoted as "A" shall be the first bit generated by the encoder, followed by the bit denoted as "B", and finally, by the bit denoted as "C"**.**

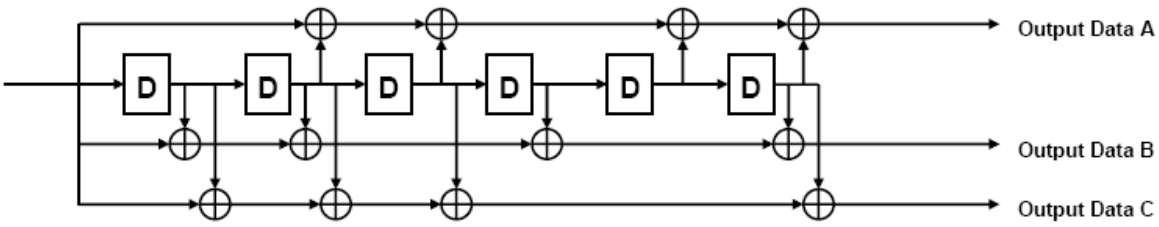


Figure 17. **Convolutional encoder: rate R = 1/3, constraint length K = 7**

The above figure denotes the parameters used for the specific convolutional encoder.

- k=1, number of input bits
- n=3, number of output bits
- R=1/3,code rate
- K=7,constraint length
- m=6,memory length of the encoder
- Total number of possible states =2^(K-1) = 2^(7-1)= 2^(6) = 64

For the m=6 memory registers,6 shift registers are used(storing 1-bit each)which are initialized to value 0. When a new input is loaded into the leftmost shift register three output bits are produced (outputA,outputB,outputC) by performing modulo-2 addition between certain bits in the memory registers for each output bit. Then all existing register values shift right and wait for the next input bit.12 XOR gates are used to operate modulo-2 sums according to the generator polynomials.

Generator polynomials representation shows the hardware connection of the shift register taps to the modulo-2 adders. A binary number representation defines a generator vector that represents the position of the taps for an output.A "1" represents a connection and a "0" represents no connection..The leftmost spot in the binary

number represents the current input, while the rightmost spot represents the oldest input that still remains in the shift register.Therefore g1=(1,0,1,1,0,1,1), g2=(1,1,1,0,1,0,1),g3=(1,1,1,1,0,0,1) is another representation of the generator polynomials which makes more obvious their purpose.

For example output bits are calculated as follows:

Data outputA = mod2 (inputdata + m_4 + m_3 + m_1+ m_0)
Data outputB = mod2 (inputdata + m_5 + m_4 + m_2+ m_0)
Data outputC = mod2 (inputdata + m_4 + m_4 + m_3+ m_0)

Where m_i , i=0,1,2,3,…,5 the memory shift registers and m_5 is the left most of them.

In order to verify that the binary representation has to do with the same polynomials as those we have mentioned before (octal representation) convert the binary representations into octal representations considering consecutive triplets of bits, starting from the rightmost bit. The rightmost bit in each triplet is the least significant. If the number of bits is not a multiple of three, then place zero bits at the left end as necessary.

For example interpret g1=(1,0,1,1,0,1,1) as 001 011 011 ↔1 3 3
g2=(1,1,1,0,1,0,1) as 001 110 101 ↔1 6 5
g3=(1,1,1,1,0,0,1) as 001 111 001 ↔1 7 1

5.3 PUNCTURER

Puncturing is a procedure for omitting some of the encoded bits in the transmitter and inserting a dummy “zero” metric into the convolutional decoder on the receive side in place of the omitted bits. In this way the number of transmitted bits is reduced and the coding rate is increased. The channel puncturer is utilised in order to reach code rates other that the nomimal code rate(1/3) given by the convolutional encoder. Mandatory code rates of 11/32 and 5/8 are achieved as well as code rate 1/3 when no puncturing is performed. The puncturing patterns for the code rates 11/32 and 5/8 respectively are illustrated in Figure 18 and Figure 19. In each of these cases, the tables shall be filled in with encoder output bits from the left to the right. For the last block of bits, the process shall be stopped at the point at which encoder output bits are exhausted, and the puncturing pattern applied to the partially filled block.

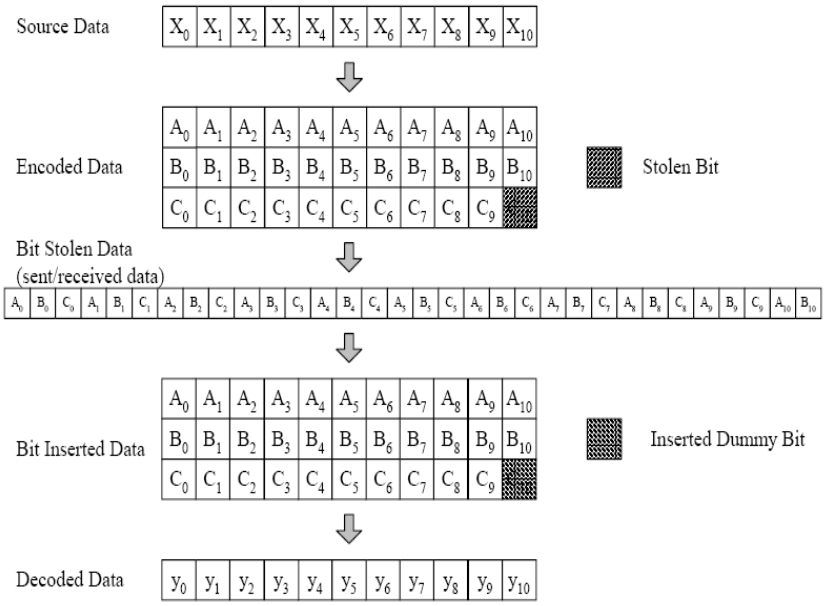


Figure 18. An example of the bit-stealing and bit-insertion procedure (R = 11/32)

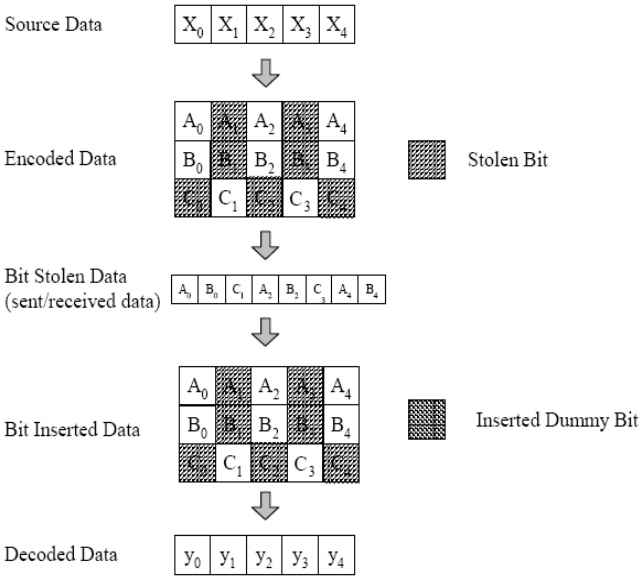


Figure 19 .An example of the bit-stealing and bit-insertion procedure (R = 5/8)

The principle of truncating bits at predefined positions in the data input stream(as illustrated in figure 18-19) is to store the encoded bits coming from the convolutional encoder in two shift registers(one for each case)and output only the necessary bits while discarding the bits to be punctured which as a consequence are not put into the output stream(see figure 20).

5.4.ENCODER RATE 1/3,5/8,11/32

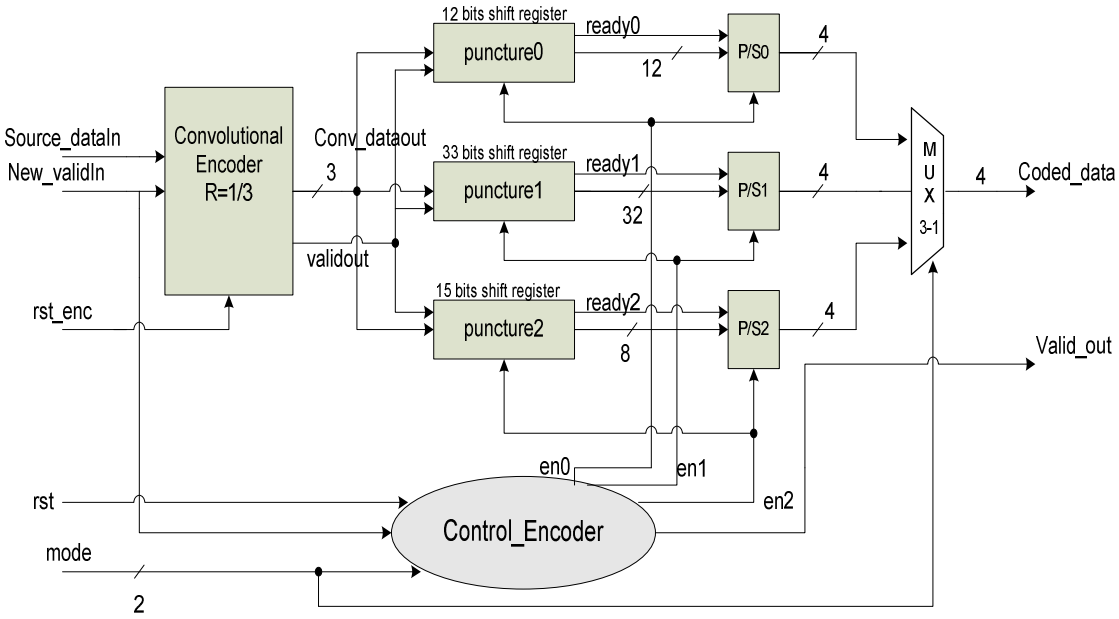


Figure 20. Encoder block diagram. Support for three modes of code rates of 11/32,5/8 and 1/3

*All the components in this block diagram are synchronous with the clk input.

The signal description of the Encoder block and the interface diagram as our VHDL implementation indicates are shown below:

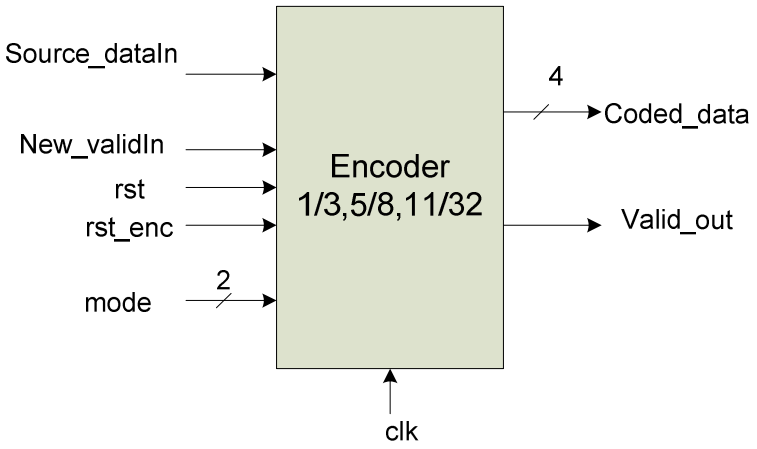


Figure 21.Encoder interface

Table 8. Encoder signal description

signal	direction	width	description
clk	input	1	Global clock
rts	input	1	Asynchronous active high global reset
rst_enc	input	1	Returns the convolutional encoder to zero state
Source_dataIn	input	1	Input data bit to be encoded
New_validIn	input	1	Indicates new valid data on Source_dataIn
mode	input	2	Determines the desired coding rate to be performed(R=1/3,5/8,11/32)
Coded_data	output	4	Output encoded data bits as vector of width 4
Valid_out	output	1	When high indicates valid data on output port

5.4.1 components description

5.4.1.1 convolutional encoder R=1/3:

The functionality of convolutional encoder has already been described above. Scrambled bits feed this encoder and the output signal Dvalidout from the data scrambler is ised to enable or disable the convolutional encoder. If a new valid input is loaded into the encoder enables it and 3 output valid bits are output in parallel(A,B,C). Else if the input bit is not valid it disables it,output is zero not valid bits and the convolutional encoder stays in the current state.

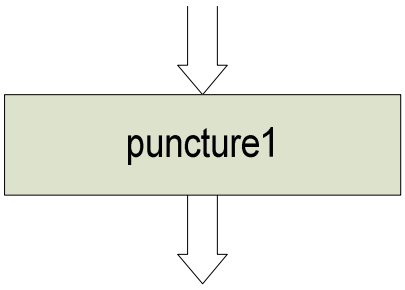
5.4.1.2 Puncture0:

A 12-bit Serial-to-Parallel shift-left register with positive edge clock. When signal validout is high 3 bits(encoded data)are loaded into the shift register every clock cycle until it fulls with 12 bits. Then outputs these 12 bits in parallel without any puncturing and feeds P/S0 in order to output encoded data in blocks of symbols with width 4.

5.4.1.3 Puncture code1:

It is implemented as a 33 bit Serial-to-Parallel shift-left register with positive edge clock.When signal validout is high 3 bits(encoded data)are loaded into the shift register every clock cycle until it fulls with 33 bits. Then outputs 32 bits,discarding the bit 33 the last in the data stream as shown in figure 18 and 22 (data-bit of shift_reg(0) is discarded). The same procedure is repeated iteratively for the remaing bits of the frame.

Input: [1 2 3 |4 5 6 |7 8 9 |10 11 12| 13 14 15| 16 17 18| 19 20 21| 22 23 24| 25 26 27| 28 29 30|31 32 33]



Bit 33 is removed when creating output.(shift_reg(0))
Bits 1 to 32 are preserved in output(shift reg(32 downto 1))
According to simulnk uwb model a puncture vector is defined where

- If Puncture vector(k) = 0, then the kth element of the input vector does not become part of the output vector.
- If Puncture vector(k) = 1, then the kth element of the input vector is preserved in the output vector.

In this case
Puncture vector1=[11111111111111111111111111111110] for 33 bits

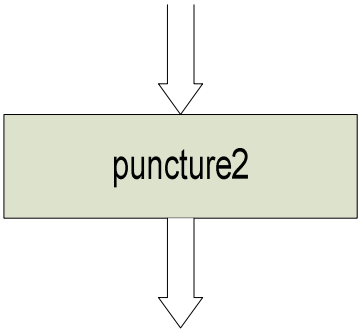
output: [1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32]

Figure 22. puncture1 example R=11/32

5.4.1.4 Puncture code2:

It is implemented as a 15 bit Serial-to-Parallel shift-left register with positive edge clock.When signal validout is high 3 bits (encoded data) are loaded into the shift register every clock cycle until it fulls with 15 bits.Then outputs 8 bits,discarding the bits 3,4,5,9,10,11,15 in the data stream as shown in figure 19 and 23(respectively data bits stored in shiftreg(12), shiftreg(11), shiftreg(10), shiftreg(6), shiftreg(5),shiftreg(4), shiftreg(0) are discarded). The same procedure is repeated iteratively for the rest data stream.

input: [1 2 3| 4 5 6| 7 8 9| 10 11 12|13 14 15]



Bits 3,4,5,9,10,11,15 are removed when creating output(shiftreg(12), shiftreg(11), shiftreg(10), shiftreg(6), shiftreg(5),shiftreg(4),shiftreg(0))
Bits 1,2,6,7,8,12,13,14 are preserved in output(shiftreg(14), shiftreg(13), shiftreg(9), shiftreg(8), shiftreg(7),shiftreg(3),shiftreg(2),shiftreg(1))
According to simulink uwb model a puncture vector is defined where

- If Puncture vector((k)=0,then the kth element of he input vector does not become part of of the output vector
- If Puncture vector((k)=1,then the kth element of he input vector is preserved in the output vector

In this case
Puncture vector2=[110001110001110] for 15 bits

output: [1 2 6 7 8 12 13 14]

Figure 23. puncture2 example R=5/8

5.4.1.5 P/S0:

A 12-bit Parallel-to-Serial shift-left register with positive edge clock and synchronous parallel load to output encoded data in blocks of 3 symblos with width 4, when code rate 1/3 is performed.

5.4.1.6 P/S1:

A 32-bit Parallel-to-Serial shift-left register with positive edge clock and synchronous parallel load to output encoded data in blocks of 8 symblos with width 4, when code rate 11/32 is performed.

5.4.1.7 P/S2:

A 8-bit Parallel-to-Serial shift-left register with positive edge clock and synchronous parallel load to output encoded data in blocks of 2 symblos with width 4, when code rate 5/8 is performed.

5.4.1.8 Control encoder:

For all the components of the Encoder that were described above there is some control effort to synchronize them and make them function properly using a finite-state-machine (FSM),a counter and combinational logic.The control unit accepts as inputs signals "mode" and "New_validIn" and activates the appropriate shift register depending on the desired mode.

- If "mode"=00 ,data from convolutional encoder are stored into the P/S0 shift register without puncturing. The control Unit enables only Puncture0 and P/S0 shift registers.Starting with the zero state,after 4 input bits have been received (4 cycles takes the loading phase of P/S0 shift register) 3 valid symbols(of width 4) are output after a certain delay(2 cycles)on Coded_data. The Valid_out signal indicates.the valid outputs. If the New_validIn signal is always high the data will be output in blocks of 3 symbols(width 4) and the Valid_out signal will be high for 3 clock cycles and low for the next 1 giving in total 12 valid bits output while 4 bits have been received.(**ratio=4/12=1/3**).

- If "mode"=01,puncture code1 has to be performed.Only puncture1 and P/S1 components are activated.Starting with the zero state,after 11 input bits have been received (11 cycles takes the loading phase of punture1 shift register) 8 valid symbols(of width 4) are output after a certain delay(2 cycles)on Coded_data. If the "New_validIn signal is always high the data will be output in blocks of 8 symbols(width 4) and the Valid_out signal will be high for 8 clock cycles and low for the next 3 giving in total 32 valid bits output while 11 bits have been received (**ratio=11/32**).

- If "mode"=10, puncture code2 has to be performed.Only puncture2 and P/S2 components are activated.Starting with the zero state,after 5 input bits have been received (5 cycles takes the loading phase of punture1 shift register) 2 valid symbols(of width 4) are output after a certain delay(2 cycles)on Coded_data..If the "New_validIn signal is always high the data will be output in blocks of 2 symbols(width 4) and the Valid_out signal will be high for 2 clock cycles and low for the next 3 giving in total 8 valid bits output while 5 bits has been received (**ratio=5/8**).

In any case if the “New_validIn” signal goes low the control unit stays in current state and the encoder retains its current information which is stored into the shift registers and can continue executing from that point when “New_validIn” goes high again.

Table 9. Coding Rate=Conv.Rate * Punc.Rate

mode	00	01	10
Conv.code rate	1/3	1/3	1/3
Puncture Rate	1/1	33/32	15/8
Coding rate	1/3	11/32	5/8

Every PLCP frame shall be encoded in the following manner. The PLCP header(consisting of the PHY header and associated tail bits, the MAC header plus HCS, and the associated tail bits, followed by the pad bits shall be encoded with a rate R = 1/3. The encoder shall be reset to the all-zero state following this. Next, the MAC frame body, tail bits and pad bits shall be encoded with a rate R =1/3, 11/32,5/8 corresponding to the desired data rate.

5.5 INTERLEAVER

Interleaver is a key component in wireless communication systems involving forward error correction (FEC) coding. Interleaving the encoded symbols provides a form of time diversity to guard against localized corruption or bursts of errors. This procedure can improve error correction capabilities of coding schemes over bursty channels. More, interleaver simply reorders input data bits (coded bits) to mitigate series of errors,so they are more evenly distributed over time. The coded bitstream is interleaved and the interleaver feeds the rearranged symbols to the modulator that follows.

The bit interleaving operation is performed in two stages: (i) symbol interleaving across the OFDM symbols, followed by (ii) intra-symbol tone interleaving.The symbol interleaver permutes the bit across OFDM symbols to exploit frequency diversity across the sub-bands, while the tone interleaver permutes the bits across the data tones within an OFDM symbol to exploit frequency diversity across tones and provide robustness against narrow-band interferers.

5.5.1 Symbol block interleaver:

In the first stage of interlaving process(interleaver1) the coded bits shall first be grouped together into blocks of 300 coded bits (corresponding to three OFDM symbols over the air). Each group of coded bits shall then be permuted using a block interleaver of size 300. Let the sequences$\{U(i)\}$ and $\{S(i)\}$, where i = 0,…, 299 represent the input and output bits of the symbol block interleaver, respectively. The input-output relationship of this interleaver shall be given by:

Equation 1. $S(i)=U\{ Floor(i/N_{CBPS})+(3* Mod(i,N_{CBPS})\}$

where the function Floor(·) returns the largest integer value less than or equal to its argument value, and where the function Mod(·) returns the remainder after division of i by N_{CBPS} (N_{CBPS}=100).

For example analyzing the equation 1 for a group of 300 coded bits

n=0 S(0)=U{ Floor(0/N_{CBPS})+(3* Mod(0,N_{CBPS})}=U(0+3*0)↔S(0)=U(0)
n=1 S(1)= U{ Floor(1/N_{CBPS})+(3* Mod(1,N_{CBPS})}=U(0+3*1)↔S(0)=U(3)
n=2 S(2)= U{ Floor(2/N_{CBPS})+(3* Mod(2,N_{CBPS})}=U(0+3*2)↔S(0)=U(6)
.
.
.
n=99 S(99)= U{ Floor(99/N_{CBPS})+(3*Mod(99,N_{CBPS})}=U(0+3*99)↔S(99)=U(297)
n=100 S(100)= U{ Floor(100/NCBPS)+(3*Mod(100,NCBPS)}=U(1+3*0)↔S(100)=U(1)
n=101 S(101)= U{ Floor(101/NCBPS)+(3*Mod(101,NCBPS)}=U(1+3*1)↔S(101)=U(4)
n=102 S(102)= U{ Floor(102/NCBPS)+(3*Mod(102,NCBPS)}=U(1+3*2)↔S(102)=U(7)
.
.
.
n=199 S(199)= U{ Floor(199/NCBPS)+(3*Mod(199,NCBPS)}=U(1+3*99)↔S(199)=U(298)
n=200 S(200)= U{ Floor(200/NCBPS)+(3*Mod(200,NCBPS)}=U(2+3*0)↔S(199)=U(2)
n=201 S(201)= U{ Floor(201/NCBPS)+(3*Mod(201,NCBPS)}=U(2+3*1)↔S(199)=U(5)
n=202 S(201)= U{ Floor(202/NCBPS)+(3*Mod(202,NCBPS)}=U(2+3*2)↔S(199)=U(8)
.
.
.
n=299 S(299)= U{ Floor(299/NCBPS)+(3*Mod(299,NCBPS)}=U(2+3*99)↔S(199)=U(299)

5.5.2 Tone block interleaver:

The output of the symbol block interleaver is then passed through a tone block interleaver(interleaver2).The outputs of the symbol block interleaver are grouped together into blocks of 100 bits and then permuted using a regular block interleaver of size 100. Let the sequences $\{S(i)\}$ and $\{T(i)\}$,where i = 0,…,99 represent the input and output bits of the tone interleaver, respectively. The input-output relationship of the tone block interleaver is given by:

Equation 2. $T(i)= S\{Floor(i/ N_{Tint})+(10* Mod(i\ N_{Tint})\}$

where N_{Tint} = N_{CBPS}/10=100/10=10 and function Mod(·) returns the remainder after division of i by $N_{Tint.}$

For example analyzing the above equation 2 for a group of 100 coded bits

n=0, T(0)= S{Floor(0/ N_{Tint})+(10* Mod(0,N_{Tint})}=S(0+10*0) ↔ T(0)=S(0)
n=1, T(1)= S{Floor(1/ N_{Tint})+(10* Mod(1,N_{Tint})}=S(0+10*1) ↔ T(1)=S(10)
n=2, T(2)= S{Floor(2/ N_{Tint})+(10* Mod(2,N_{Tint})}=S(0+10*2) ↔ T(2)=S(20)

.
.
.
n=9 T(9)= S{Floor(9/ N_{Tint})+(10* Mod(9,N_{Tint})}=S(0+10*9) ↔ T(9)=S(90)
n=10, T(10)= S{Floor(10/ N_{Tint})+(10* Mod(10,N_{Tint})}=S(1+10*0) ↔ T(10)=S(1)
n=11 T(11)= S{Floor(11/ N_{Tint})+(10* Mod(11,N_{Tint})}=S(1+10*1) ↔ T(10)=S(11)
.
.
.
n=19 T(19)= S{Floor(19/ N_{Tint})+(10* Mod(19,N_{Tint})}=S(1+10*9) ↔ T(19)=S(91)
n=20 T(20)= S{Floor(20/ N_{Tint})+(10* Mod(20,N_{Tint})}=S(2+10*0) ↔ T(20)=S(2)
T(21)=S(12)
.
.
.
T(99)=S(99)

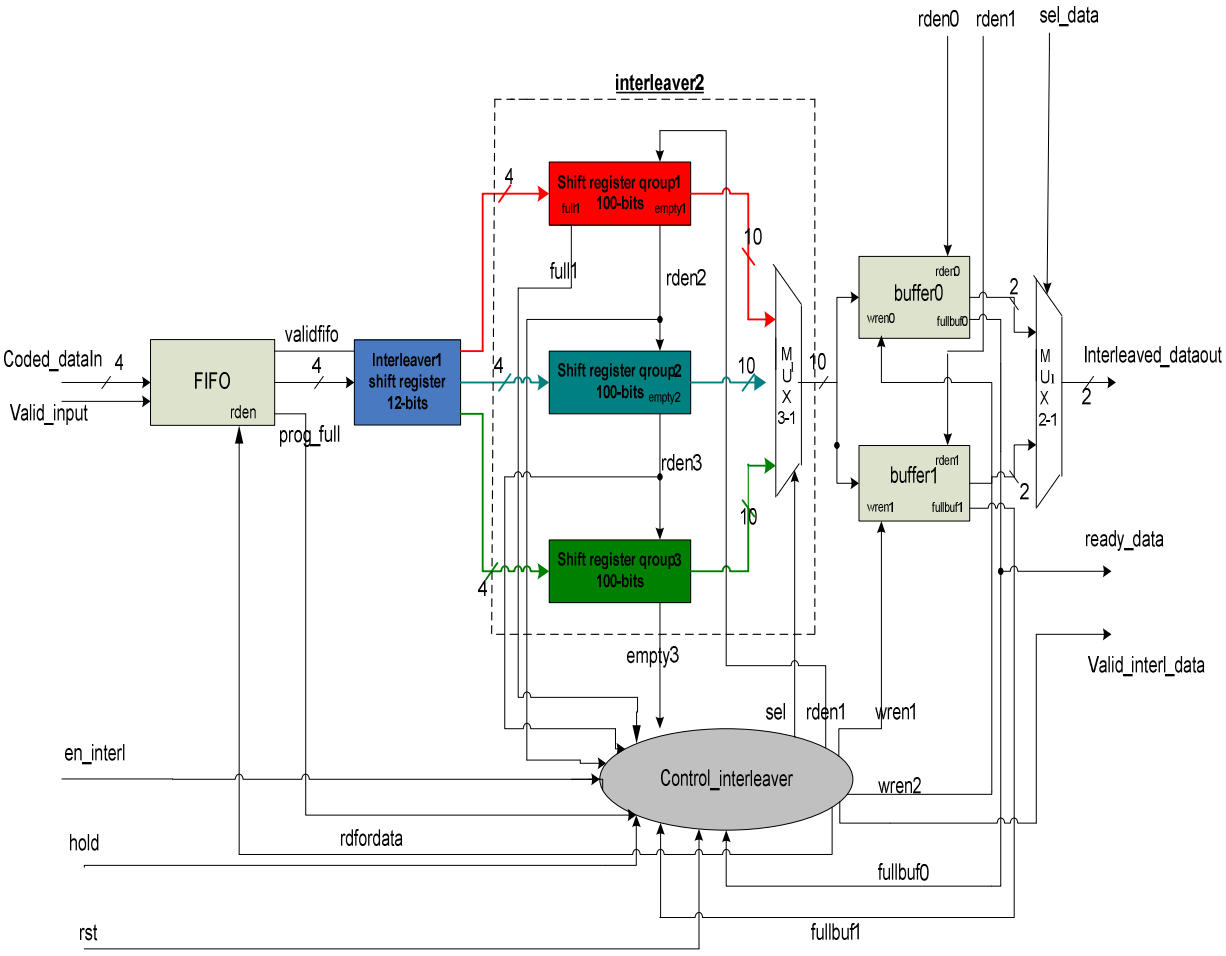


Figure 24.Interleaver block diagram

*All the components in this block diagram are synchronous with the clk input except for multiplexers.

The signal description of the Interleaver block and the interface diagram as our VHDL implementation indicates are shown below:

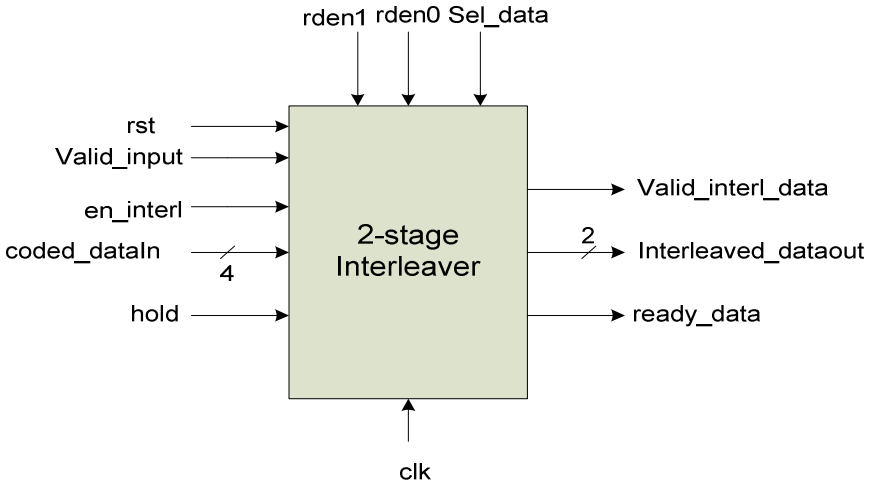


Figure 25 .Interleaver interface

Table 10. Interleaver signal description

signal	direction	width	description
clk	input	1	Global clock
rst	input	1	Asynchronous active high global reset resets Interleaver to the all zero state
en_interl	input	1	Enables interleaver when high
Valid_input	input	1	Valid coded data which are loaded into FIFO
hold	input	1	If high stalls Interleaving process and stays in the current state.All the contents of shift registers are preserved when hold is low.
Coded_dataIn	input	4	Encoded input data
rden0	input	1	Read buffer0 if high
rden1	input	1	Read buffer1 if high
sel_data	input	1	Selects output of mux2-1
Interleaved_dataout	output	2	Output interleaved data divided into groups of two bits.
valid_interl_data	output	1	Indicates valid interleaved data bits that are stored into buffers0,1.It is used by the Global Control to stall interleaver when it counts 600 bits that were generated.
ready_data	output	1	Is used to enable the following block(Mapper).When buffer0 is full with 600 bits(6 OFDM symbols) ready_data goes high and drives the following Mapper which starts reading buffer0 and receiving 2-bits vectors

5.5.3 First Stage Interleaver operation:(see figure 24-26)

A 12-bit shift register is used(depth=3,width=4) in which encoded data bits are stored and rearranged.When Interleaver is enabled(signal “en_interl” is high) the Contol Unit stays at the initial state and waits until 600 encoded bits (corresponding to 6 OFDM symbols over the air) has been writen to FIFO memory. At this time signal “progfull” goes high and the control indicates that interleaver process is ready to process new data (signal “rdfordata” goes high).As a result data are read from FIFO and loaded into the Interleaver1 shift register.After 3 loads of 4-bit encoded symbols the shift register is full with 12-bits and outputs bit in interleaved order according to equation 1(see figure 26).Then it continues to load and reorder data bits in the same way until 300 bits have been reordered by this operation and have been stored appropriately into the three shift registers of the interleaver2 block.This takes 75 cycles.At this time control exploits indication “full” of Interleaver2 shift registers and rdfordata goes low(stops reading FIFO). In this way 300 interleaved bits according to equation 1 have already been stored into groups of 100 bits,each group in a different shift register. So starts Interleaver2 block execution.

We have to mention that FIFO allows simultaneous read and write operation.While data bits are read from FIFO new encoded data are written with data rate equal to the desired encoder data rate(see section 5.4). Even if inteleaver stops reading data from FIFO write operation occurs and for this reason extra control logic has implemented for synchronizing interleaver,encoder and scrambler in order to allow continues data processing and to avoid FIFO to become full or empty. This control logic stalls scrambler and as a consequence Encoder for a number of cycles that is defined according to the "mode" and the difference between the number of clock cycles needed to write 600 bits to the input FIFO and the number of cycles needed to output 600 bits(300 vectors of width 2) by Interleaver.

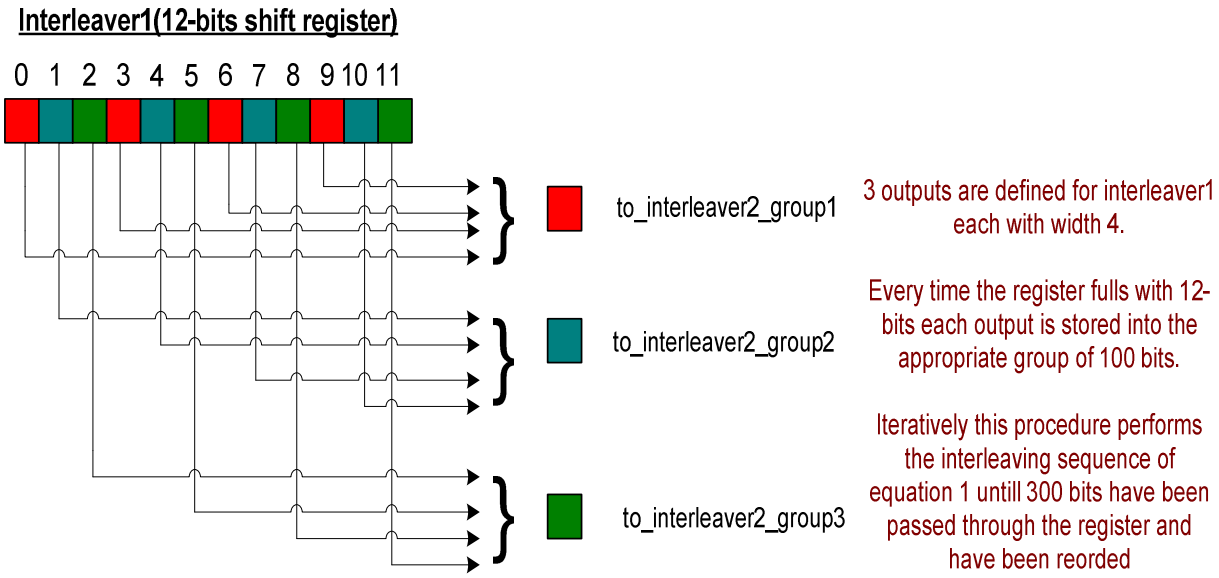


Figure 26. Interleaver1 shift register operation

5.5.4 Second Stage Interleaver operation:(see figure 24-27)

It consists of three 100-bit buffers(shift registers with depth=25,width=4) that work in the same way. As we have mentioned before after the interleaver1 process these buffers are full with 100-bits each. At that case tone block interleaving procedure begins. Control Unit activates the reading process("rden1" high)of the first shift register which contains the fisrt group of 100 bits while at the same time the contents of the other two shift registers remain unchanged. Shift register_group1 outputs 10 bits in parallel and is shifted left for 10 cycles until it becomes empty. In each shift it generates the interleaving sequence of the tone block interleaver as described by equation 2 and as shown in figure 27. At the same time the control unit assigns the corresponding value to signal "sel" for the 3-1 multiplexer to output bits from shift register_group1.

When the first qroup of 100 bits has been interleaved signal "empty1" goes high and is used to read shift register group2. With the same way shown in figure 27 the next 100 bits are interleaved until empty2 goes high and activates reading of shift register group3. Appropriate values are assigned to signal sel in each case. When

finally the interleaving process for the last 100 bits ends “rdfordata” goes high indicating that Interleaver is ready to process a new 300 bits block through interleaver1 and interleaver2 operation and starts reading FIFO again. Interleaving2 takes 30 cycles.

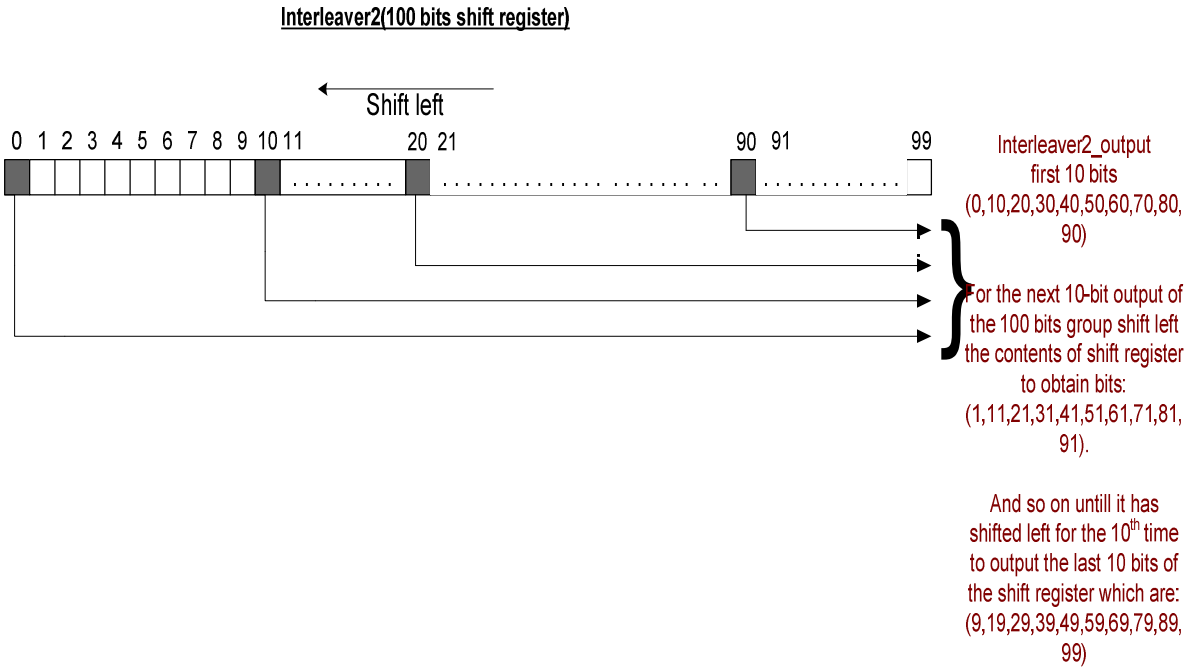


Figure 27.Interleaver2 shift register operation

5.5.5 Double buffering:

To ensure continuous data flow two buffers are required. When the first buffer is filled up with 600 bits((corresponding to 6 OFDM symbols over the air) the next block starts reading the first buffer while in parallel the second buffer is filled up with data bits for the next 6 ofdm symbols. These two buffers divide the encoded and interleaved data bits into groups of two bits 100 output vectors of two bits (200 bits) corresponding to one OFDM symbol for the case of transmiting with data rate=200 Mbits/s and 110 Mbits/sec and 50 output vectors of two bits(100 bits) correspond to one OFDM symbol when transmiting the PLCP Header with data rate=53.3 Mbits/sec. One vector(2 bits) is assigned to each OFDM data carrier and converted into complex numbers representing QPSK constellation points by the next block (Mapper-section 5.6).

The control unit controls also double buffering of interleaved data.When writing into buffer0, wren0 is high and if buffer0 is full, wren0 is low and wren1 is high writting to buffer1.In the same way when buffer1 is full wren1 goes low and wren0 goes high.Mapper controls signals rden0,rden1,sel and determines reading of these two buffers as will be described in the following section. In addition when buffer0 becomes high for the fist time signal full_buf0 is used by the Mapper to start executing.

5.6 MAPPER

This block consists of two sub-blocks. The first one converts each incoming vectors of two bits ,coming from the Interleaver ,into a complex number representing QPSK constellation points. The second is responsible for assembling OFDM symbols in a way that will be described above.

5.6.1 QPSK:

Since encoded and interleaved data are divided into groups of two bits, based on the input vector a complex pair which contains the real and the imaginary part is output by the QPSK block corresponding to QPSK constellation points.The conversion shall be performed according to the Gray-coded constellation mapping, illustrated in Figure 28, with the input bit, b0, being the earliest in the stream. The output values, *d*, are formed by multiplying the resulting (I + jQ) value by a normalization factor of K_{MOD}, as described in the following equation:

$$d = (I + jQ) \times K_{MOD}$$

The normalization factor, KMOD, depends on the base modulation mode,as defined in Table 11**.** For QPSK, b0 determines the I value, and b1 determines the Q value, as illustrated in Table 12.

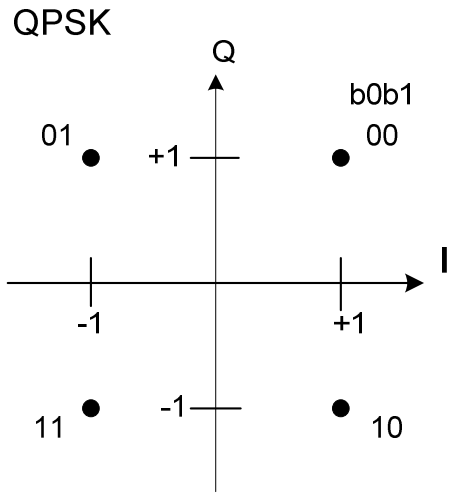


Figure 28 – QPSK constellation bit encoding

Table 11: Modulation-dependent normalization factor KMOD

Modulation	K_{MOD}
QPSK	$1/\sqrt{2}$

Table 12:QPSK encoding table

Input bit(b0 b1)	I-out	Q-out
00	+1	+1
01	−1	+1
10	+1	−1
11	−1	−1

In practice the QPSK modulation is implemented using a multiplexing block which is based on the input vector with width 2-bits and outputs the appropriate I,Q pairs in fixed-point representation. In particular it outputs the values I* K_{MOD}=−0.7071 or +0.7071 and Q * K_{MOD}=−0.7071 or +0.7071. To avoid using multipliers we have pre-computed the fixed-point representation values of the normalized constellation points I,Q.The calculation of those values has been done with the help of Matlab functions which convert a real number into a fixed-point binary number by specifying its word size (in bits) and the location of the binary point, as desired. Same as the uwb simulink reference model we have used 16 bits wordlength with 14 bits fraction part and 2 bits integer part (see figure 29). The actual QPSK encoding table is shown in Table 13.

Table 13: Practical QPSK encoding table

Input bit(b0 b1)	I-out normalized	Q-out normalized
00	+0.7071	+0.7071
01	−0.7071	+0.7071
10	+0.7071	−0.7071
11	−0.7071	−0.7071

: **+0.7071=00.10110101000001**
−0.7071=11.01001010111111

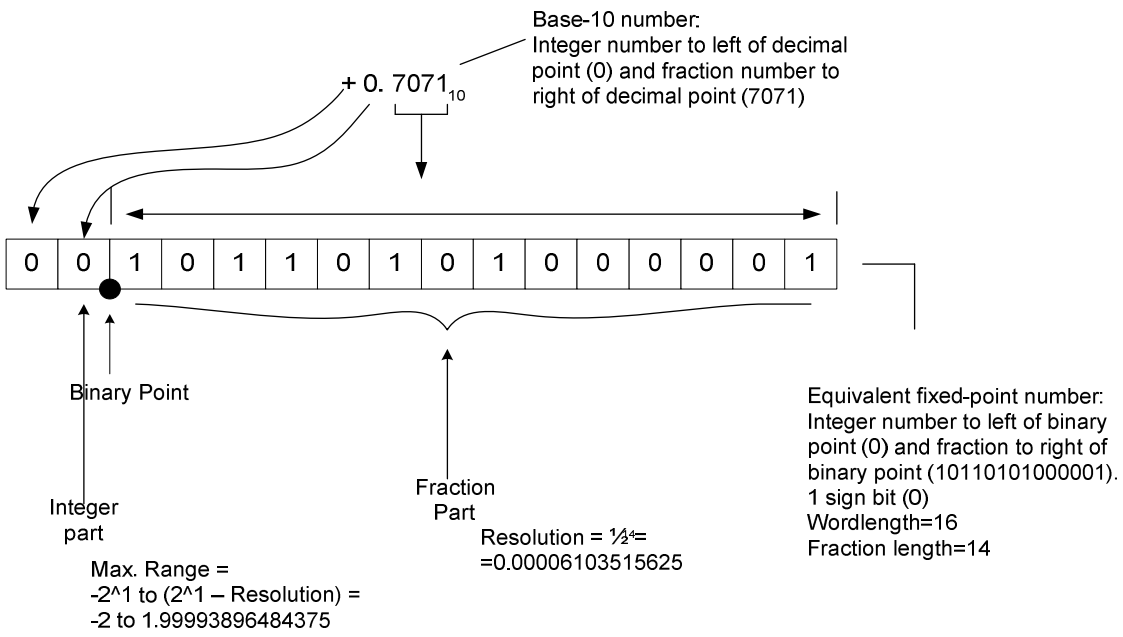


Figure 29. Fixed-point representation

The fraction-part of the fixed-point number = the binary equivalent of the decimal fraction divided by the resolution.Resolution is 1/2^E, where E is the number of bits to the right of the binary point. In our example, resolution is 1/2^14 = 0.00006103515625. Therefore, the fraction-part of the fixed-point number = the binary equivalent of 0.7071/0.00006103515625= 10110101000001.
Number –0.7071=11.01001010111111 is the two's complement of number +0.7071

The signal description of the QPSK block and the interface diagram as our VHDL implementation indicates are shown below:

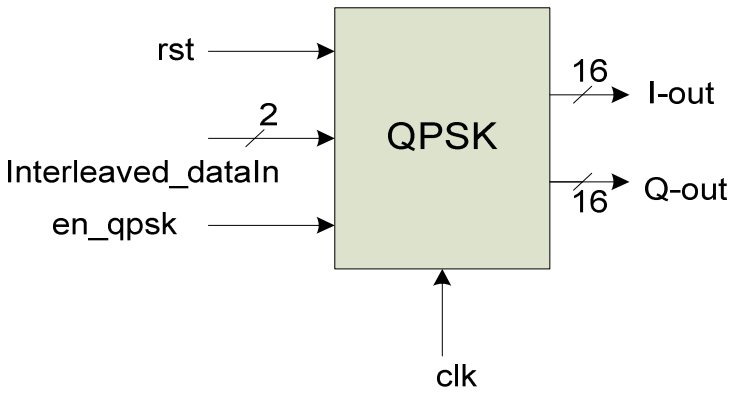


Figure 30 – QPSK interface

Table 14. QPSK signal description

signal	direction	width	description
clk	input	1	Global clock
rst	input	1	Asynchronous active high global reset
Interleaved_dataIn	input	2	Input data
en_qpsk	input	1	Enables constellation mapping
I-out	output	16	Real part of data out
Q-out	output	16	Imaginary part of data out

5.6.2.OFDM_SYMBOL ASSEMBLER:

This block is responsible for the formation of the OFDM symbol by using the modulation mapping fuction for the chosen modulation type.

5.6.2.1 OFDM Modulation

- For information data rates of 53.3, the stream of complex symbols is divided into groups of 50 complex numbers**.** We shall denote these complex numbers *Cn*,*k*, which corresponds to subcarrier *n* of OFDM symbol *k*, as follows:

$$c_{n,k} = d_{n+50\times k} \qquad n = 0,1,\mathrm{K}\ ,49, k = 0,1,\mathrm{K}\ ,\mathrm{N_{SYM}} - 1$$
$$c_{(n+50),k} = d^{*}_{(49-n)+50\times k}$$

where NSYM denotes the number of OFDM symbols in the MAC frame body, tail bits, and pad bits.

- For information data rates of 110,200Mb/s, the stream of complex numbers is divided into groups of 100 complex numbers. We shall denote these complex numbers *cn*,*k*, which corresponds to subcarrier *n* of OFDM symbol *k*, as follows:

$$c_{n,k} = d_{n+100\times k} \qquad n = 0,1,\mathrm{K}\ ,99, k = 0,1,\mathrm{K}\ ,\mathrm{N_{SYM}} - 1$$

where NSYM denotes the number of OFDM symbols in the MAC frame body,tail bits, and pad bits.

5.6.2.2 Pilot subcarriers

In each OFDM symbol following the PLCP preamble, twelve of the subcarriers are dedicated to pilot signals in order to make coherent detection robust against frequency offsets and phase noise. These pilot signals shall be put in subcarriers numbered –55, –45, –35, –25, –15 –5, 5, 15, 25, 35, 45, and 55.

$$P_{n,k} = p_{\mathrm{mod}(k,127)} \times \begin{cases} \frac{1+j}{\sqrt{2}} & n = 25, 55 \\ \frac{-1-j}{\sqrt{2}} & n = 5, 15, 35, 45 \\ 0 & n = \pm 1 \ldots, \pm 4, , \pm 6, \ldots, \pm 14, \pm 16, \ldots, \pm 24, \pm 26, \ldots, \pm 34, \pm 36, \ldots, \pm 44, \pm 46, \ldots, \pm 54, \pm 56 \end{cases}$$

- For modes with data rates less than 106.7 Mbps that is for the case of 53.3 Mb/s in our design:

$$P_{n,k} = P^{*}_{-n,k}, \qquad n = -5, -15, -25, -35, -45, -55$$

- For 106.7 Mbps and all higher rate modes that is for the case of 110 and 200 Mb/s in our design :

$$P_{n,k} = P_{-n,k}, \qquad n = -5, -15, -25, -35, -45, -55$$

In this numbering, *k=0* shall correspond to the first OFDM symbol following the PLCP preamble.

5.6.2.3 Guard subcarriers

Moreover in each OFDM symbol ten of the subcarriers at the edges of the occupied frequency band shall be termed guard subcarriers and defined as in this subsection. Implementations may exploit the guard subcarriers for various purposes, including relaxing the specs on transmit and receive filters as well as possible performance improvements. The relation of the power level of the guard subcarriers to the data bearing subcarriers shall be implementation dependent, except that the same relation shall be employed for all OFDM symbols that define guard subcarriers. Note that this includes both the CE sequence and payload. Thus, implementations may use reduced power levels for the guard subcarriers as long as the resultant transmit signal meets the local regulatory requirements of minimum occupied bandwidth, etc. There are five guard subcarriers on either edge of the OFDM symbol occupied band, located in subcarriers with indices –61, –60, …, –57, and 57, 58, …, 61. These guard subcarriers shall be created by copying over the five outermost data-bearing subcarriers from the nearest edge of the OFDM symbol as shown in Figure 27 (note the intervening pilot subcarrier is not copied).

More precisely, the guard subcarrier symbol definition for the *nth* subcarrier of the *kth* symbol shall be given as follows**:**

$$P_{n,k} = c_{m,k}, \quad l = 0,1,2,3,4; \; n = 57 + l; \; m = 95 + l$$
$$P_{n,k} = c_{m,k}, \quad l = 0,1,2,3,4; \; n = -61 + l; \; m = l$$

In this numbering, *k=0* shall correspond to the first OFDM symbol following the PLCP preamble

Figure 31 - Guard subcarrier creation based on edge subcarriers of the OFDM symbol

Table 15.Subcarriers within an OFDM symbols

Parameter	Value
N_{SD}: Number of data subcarriers	100
N_{SDP}: Number of defined pilot carriers	12
N_{SG}: Number of guard carriers	10
N_{ST}: Number of total subcarriers used	122 (= N_{SD} + N_{SDP} + N_{SG})

5.6.2.4 Ofdm_symbol_assembler design: (see figure 35)

Operates in two different modes according to the desired data rate.When data rate 53.3 Mb/s is indicated by the PHY header every Ncbps(coded bits per symbol)=100 output bits of the interleaver are converted into 50 complex numbers by the QPSK block(each complex number represents 2 output bits of the interleaver). When data rate 110 or 200 Mb/s is indicated by the PHY header every Ncbps (coded bits per symbol)=200 output bits of the interleaver are converted into 100 complex numbers by the QPSK block(each complex number represents 2 output bits of the interleaver).

In both cases,these complex numbers,consisted of a real value (Q) and an imaginary value (I), are temporarily allocated in dual memory dual port RAMs. Two dual-port block RAMs were used for each part of complex number(I,Q). This allowed the ofdm_symbol_assembler to simultaneously store new data(I,Q values) for the next OFDM symbol and read the data of the current OFDM symbol that is going to be modulated.Thus when the one buffer which stores I values is full, it could be read, while the second buffer which also stores I values is still storing the new data input. In this way continuous data flow is allowed and also avoidance of memory addresses collisions and data losses.

Before the complex numbers are processed by the 128-point IFFT a carrier sorting operation occurs,in which the incoming 50 or 100(depending on the data rate) complex pairs(I,Q) values are assigned to sub-carrier frequencies within the OFDM symbol.12 complex numbers representing the pilot signals and ten guard subcarrier

are inserted into the memory output through multiplexers at set positions. The pilot and guards are some predefined values as described before and their corresponding positions are illustrated in Figure 31. As with the values of the normalized constellation points I,Q, we used again the same fixed point representation for pilot values(+0.7071,0.7071).

Because an OFDM symbol consists of 128 sub-carriers:

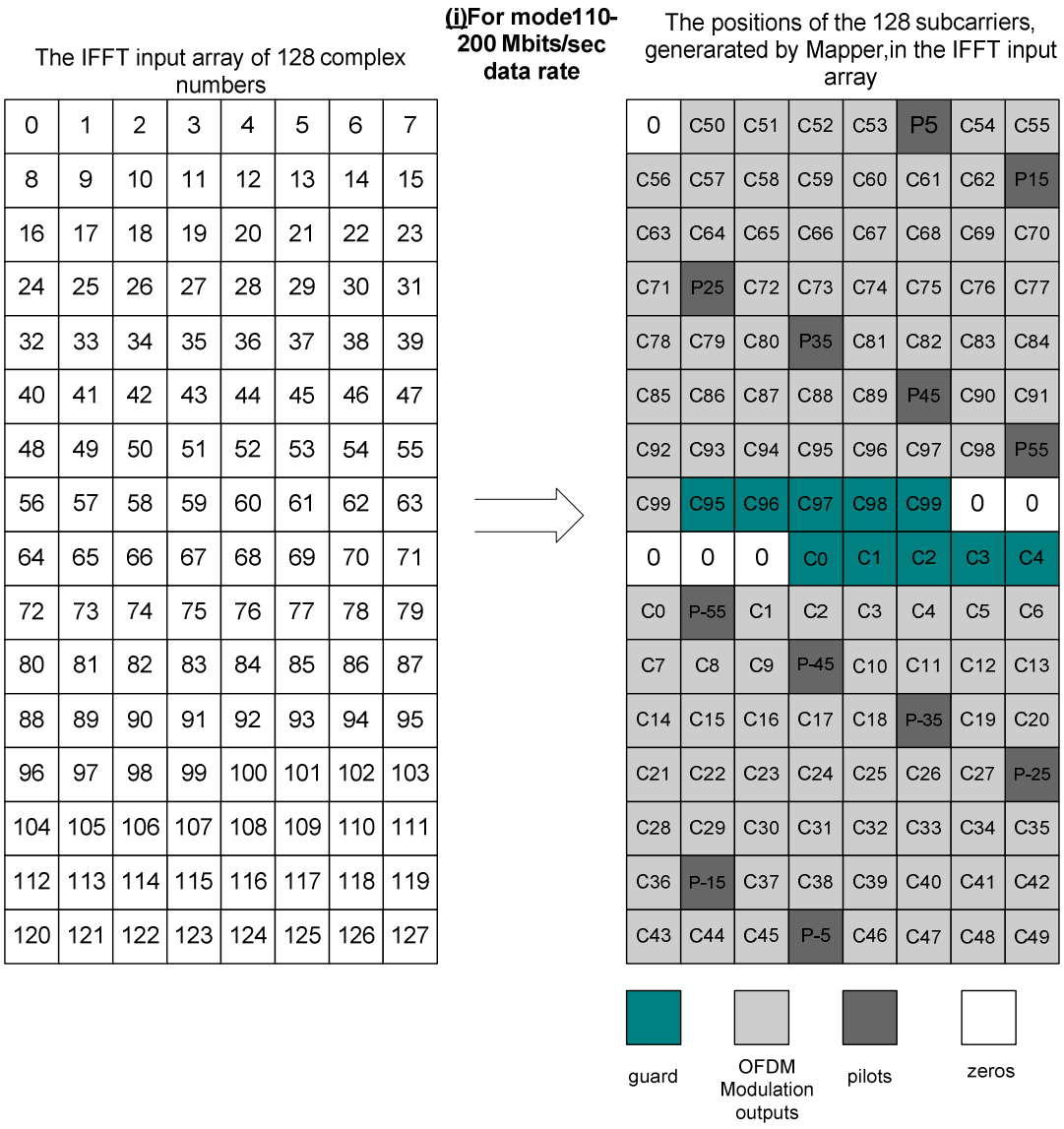
- For data rate of 53.3 Mb/sec,a conjucate symmetric process has to be performed to the 50 complex pairs(I,Q) leading to total 100 data subcarriers plus the12 pilots subcarriers plus the10 guard subcarriers leading to122 subcarriers.Finally there are also 6 zero carriers to consider that are inserted in the designated positions(see figure 31,34)
- For data rate of 110,200 Mb/sec,the 100 complex pairs assigned to 100 data subcarriers plus the12 pilots subcarriers plus the10 guard subcarriers leading to 122 subcarriers.Finally 6 zero carriers to consider inserted in the designated positions(see figure 31,33).

Appropriate carrier sorting is perfomed using an address counter multiplexers and control effort to set all 128-subcarriers(corresponding to 1 OFDM symbol at the designated positions) and output them in the order that has to be loaded into IFFT. Fig.32,33,34 illustrates the composition of the 128 complex IFFT inputs.

carrier	-61..-57	-56	-55	-54..-46	-45	-44..-36	-35	-34..-26	-25	-24..-16	-15	-14..-6	-5	-4..-1	0	1..4	5	6..14	15	16..24	25	26..34	35	36..44	45	46..54	55	56	57..61
position	67...71	72	73	74..82	83	84..92	93	94..102	103	104..112	113	114..122	123	127	0	1...4	5	6...10	15	16..24	25	26..34	35	36..44	45	46..54	55	56	57..61
type	guard	carrier	pilot	carrier	pilot	carrier	pilot	carrier	pilot	carrier	pilot	carrier	pilot	carrier	0	carrier	pilot	carrier	pilot	carrier	pilot	carrier	pilot	carrier	pilot	carrier	pilot	carrier	guard

Figure 32: Assignment of sub-carriers within an OFDM symbol

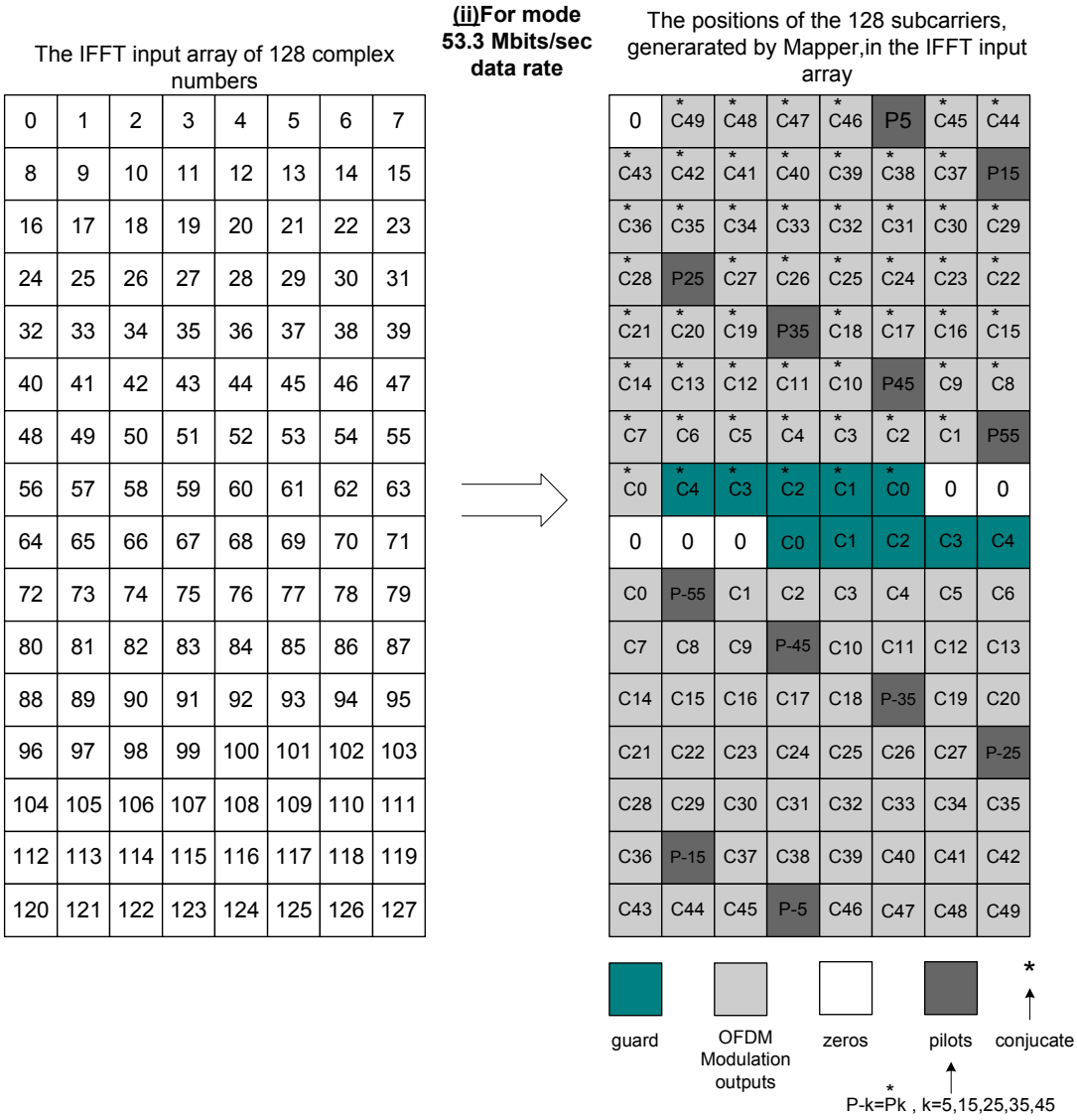
*positions 62…66 are filled with zeros



0	1	2	3	4	5	6	7
8	9	10	11	12	13	14	15
16	17	18	19	20	21	22	23
24	25	26	27	28	29	30	31
32	33	34	35	36	37	38	39
40	41	42	43	44	45	46	47
48	49	50	51	52	53	54	55
56	57	58	59	60	61	62	63
64	65	66	67	68	69	70	71
72	73	74	75	76	77	78	79
80	81	82	83	84	85	86	87
88	89	90	91	92	93	94	95
96	97	98	99	100	101	102	103
104	105	106	107	108	109	110	111
112	113	114	115	116	117	118	119
120	121	122	123	124	125	126	127

0	C50	C51	C52	C53	P5	C54	C55
C56	C57	C58	C59	C60	C61	C62	P15
C63	C64	C65	C66	C67	C68	C69	C70
C71	P25	C72	C73	C74	C75	C76	C77
C78	C79	C80	P35	C81	C82	C83	C84
C85	C86	C87	C88	C89	P45	C90	C91
C92	C93	C94	C95	C96	C97	C98	P55
C99	C95	C96	C97	C98	C99	0	0
0	0	0	C0	C1	C2	C3	C4
C0	P-55	C1	C2	C3	C4	C5	C6
C7	C8	C9	P-45	C10	C11	C12	C13
C14	C15	C16	C17	C18	P-35	C19	C20
C21	C22	C23	C24	C25	C26	C27	P-25
C28	C29	C30	C31	C32	C33	C34	C35
C36	P-15	C37	C38	C39	C40	C41	C42
C43	C44	C45	P-5	C46	C47	C48	C49

Figure 33.The composition of the IFFT inputs for the case of 110/200 Mb/s data rate



0	1	2	3	4	5	6	7
8	9	10	11	12	13	14	15
16	17	18	19	20	21	22	23
24	25	26	27	28	29	30	31
32	33	34	35	36	37	38	39
40	41	42	43	44	45	46	47
48	49	50	51	52	53	54	55
56	57	58	59	60	61	62	63
64	65	66	67	68	69	70	71
72	73	74	75	76	77	78	79
80	81	82	83	84	85	86	87
88	89	90	91	92	93	94	95
96	97	98	99	100	101	102	103
104	105	106	107	108	109	110	111
112	113	114	115	116	117	118	119
120	121	122	123	124	125	126	127

0	*C49	*C48	*C47	*C46	P5	*C45	*C44
*C43	*C42	*C41	*C40	*C39	*C38	*C37	P15
*C36	*C35	*C34	*C33	*C32	*C31	*C30	*C29
*C28	P25	*C27	*C26	*C25	*C24	*C23	*C22
*C21	*C20	*C19	P35	*C18	*C17	*C16	*C15
*C14	*C13	*C12	*C11	*C10	P45	*C9	*C8
*C7	*C6	*C5	*C4	*C3	*C2	*C1	P55
*C0	*C4	*C3	*C2	*C1	*C0	0	0
0	0	0	C0	C1	C2	C3	C4
C0	P-55	C1	C2	C3	C4	C5	C6
C7	C8	C9	P-45	C10	C11	C12	C13
C14	C15	C16	C17	C18	P-35	C19	C20
C21	C22	C23	C24	C25	C26	C27	P-25
C28	C29	C30	C31	C32	C33	C34	C35
C36	P-15	C37	C38	C39	C40	C41	C42
C43	C44	C45	P-5	C46	C47	C48	C49

Figure 34.The composition of the IFFT inputs for the case 53.3 Mbits/sec data rate

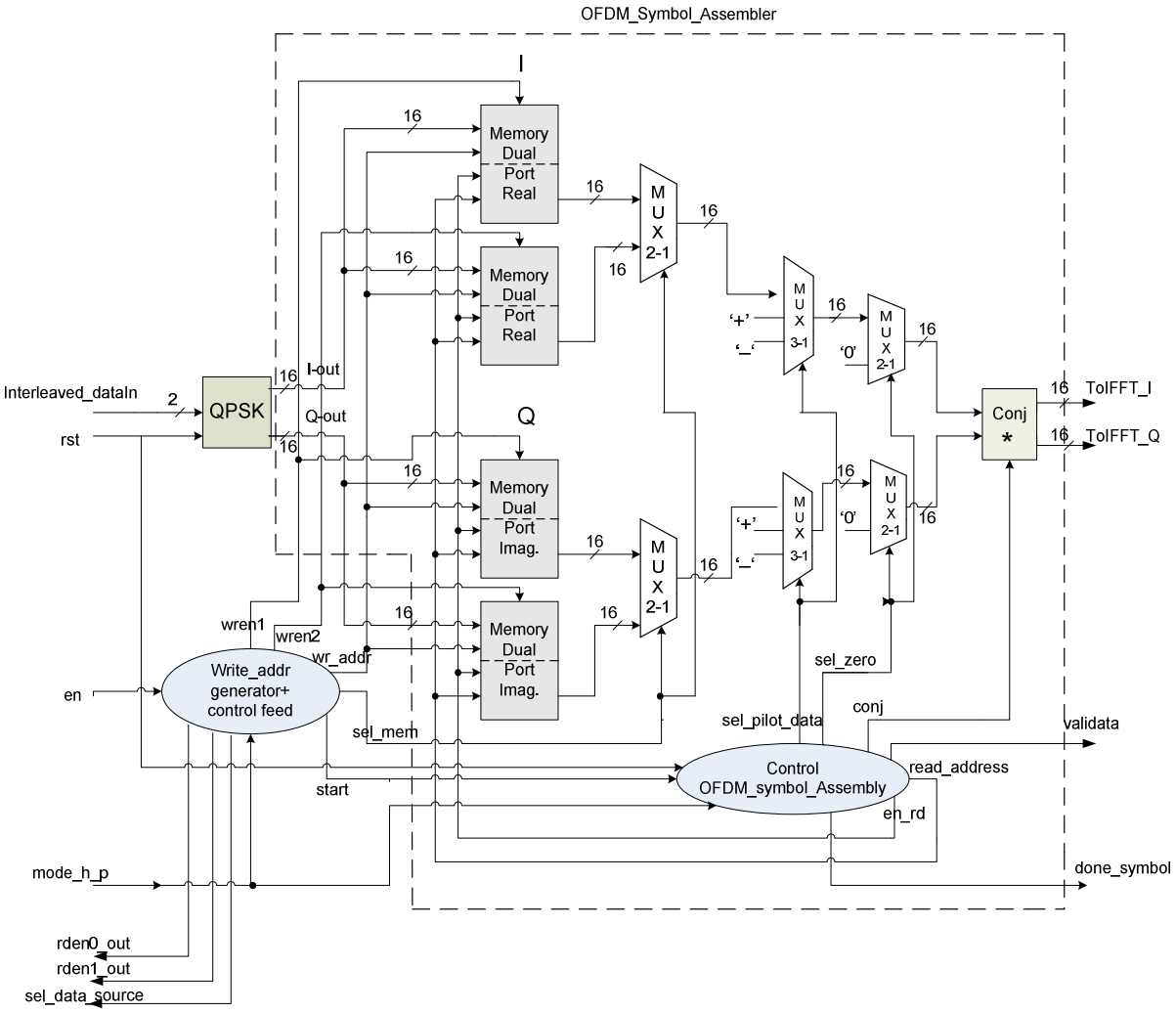


Figure 35. Mapper block diagram

*All the components in this block diagram are synchronous with the clk input except for the mux2-1.

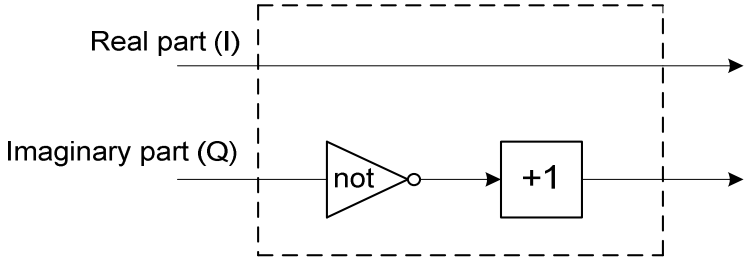


Figure 36.Conjucate

The signal description of the Mapper block and the interface diagram as our VHDL implementation indicates are shown below:

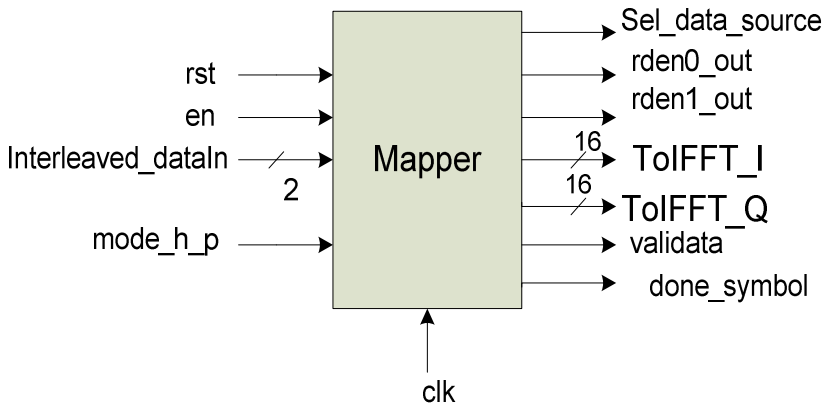


Figure 37 – Mapper interface

Table 16.Mapper signal description

signal	direction	widh	description
clk	input	1	Global clock
rst	input	1	Asynchronous active high global reset.
en	input	1	Enables Mapper when high
Interleaved_dataIn	input	2	Data input(vectors of interleaved data which are to be converted into complex numbers and assigned to sub-carrier frequencies within the OFDM symbol)
mode_h_p	input	1	Defines the operation mode of Mapper. • If low modulates for the case of transmiting PLCP header with rate=53.3 Mbits/sec. • If high modulates for the case of transmiting Frame Payload with rate=110 or 200 Mbits/sec
ToIFFT_I	output	16	Real part of output subcarrier
ToIFFT_Q	output	16	Imaginary part of output subcarrier
validata	output	1	Data valid on output(enables IFFT module)
rden0_out	output	1	Enables read out of buffer0 of the Interleaver
rden1_out	output	1	Enables read out of buffer1 of the Interleaver
sel_data_source	output	1	Defines the buffer of the Interleaver block which wiil provide input data to Mapper
done_symbol	output	1	When high indicates that one ofdm symbol has been assembled and 128 I,Q values have been passed to IFFT module

5.6.2.4.1 Write_addr generator & control feed

When the first 600 bits have been interleaved and stored into buffer0 of the Interleaver signal “ready_data” goes high and enables Mapper. More specificly write_addr generator & control feed unit starts operating and controlling reading of buffer0 and writing complex pair I,Q values (generated by QPSK block) into Dual Port RAMs I,Q respectively.

For data rate of 53.3 Mbits/sec rden0 signal is high until Mapper has received 50 vectors width 2 each,(thus 100 bits) corresponding to 50 complex pairs I,Q. Then reading is stoped and OFDM_Symbol_Assembler is activated.128 cycles are needed to assemble 1 OFDM symbol which consists of 128 subcarriers. As a consequence we had to stop feeding Mapper module with interleaved data for 78 cycles. In this way Mapper operates in blocks of 100 bits (50 I,Q values). Then reading of buffer0 starts again(“rden0” high) to feed Mapper with the next 50 input vectors with width 2 each and assembles the next OFDM symbol. When buffer0 is empty data are read from buffer1 which contains the next 600 interleaved bits that were generated in parallel while Mapper was operating. Sel_data_source value is controlled appropriately depending on which buffer has to be read. This process is continued in cascade until all OFDM symbols of the PLCP frame have been assembled.

In the case of 110 and 200 Mbits /sec the same process is used with the difference that we have to stall feeding of Mapper for 28 cycles after reading 100 vectors with width 2 each, (thus 200 bits) corresponding to 100 complex pairs I,Q that will be generated. In this way Mapper operates in blocks of 200 bits this time.

Write addresses for every I,Q complex pair are generated filling up Dual Port RAMs I,Q sequentially.Write address vary from 0 to 49 in case of “mode_h_p”=0 and 0 to 99 for “mode_h_p”=1.The two memory blocks for each I,Q writing process are used alternately.”Wren1” drives the first memory for I,Q and is high until data corresponding to 1 OFDM symbol have been stored.Then “wren1” goes low and “wren2” high and as a result the first memory is read out while the second one is filled up.This allows Mapper to continuously process data.

4.6.2.4.2 Control OFDM symbol Assembly

Controls reading data from the Dual Port Memories I,Q corresponding to data subcarriers.According to the desired mode controls signals “sel_pilot_data”, ”sel_zero”,”conj” are controlled to output through multiplexers the 128 subcarriers as illustrated in figure 33 and 34. Enables read out of the block memories I,Q as long as all data subcarriers for one OFDM symbol have been stored. Read address vary from 50 to 99 and 0 to 49 if rate=200,110 Mbits/sec is selected(mode_h_p=1) and from 49 to 0 and 0 to 49 if rate=53.3 Mbits/sec (mode_h_p=0) is selected in order to achieve appropriate sorting as illustrated in figures 33 and 34. The symbol Pilots and Zeros are inserted into the memory outputs through multiplexers and with the help of some control effort using counters and FSMs.Conjucate is perfomed if necessary to some outputs according to figure 36 only when mode 53.3 Mb/s is selected.”Sel_pilot_data” defines if data subcarrier or pilot will be output and “sel_zero” if zero will be output.

Mux_3-1 output

sel_pilot_data	Mux3-1 output
00	data subcarrier
10	positive pilot(+0.7071)
11	negative pilot(-0.7071)

Mapper feeds IFFT module with 128 I,Q values for each OFDM symbol continuously allowing IFFT to operate in streaming mode.

5.7 IFFT:

The IFFT block transforms transforms the subcarriers of an OFDM symbol into the time-domain.It implements the Inverse Fast Fourier Transform algorithm.The IFFT block is based on a 128-point IFFT in radix-2 structure and therefore 128 input values are necessary. The coefficients 1 to 61 (see Figure 38) are mapped to the same numbered IFFT inputs, while the coefficients –61 to –1 are copied into IFFT inputs 67 to 127. The rest of the inputs, 62 to 66 and the 0 (DC) input, are set to zero. The composition of IFFT inputs (128 subcarriers) was expained in the previous section.

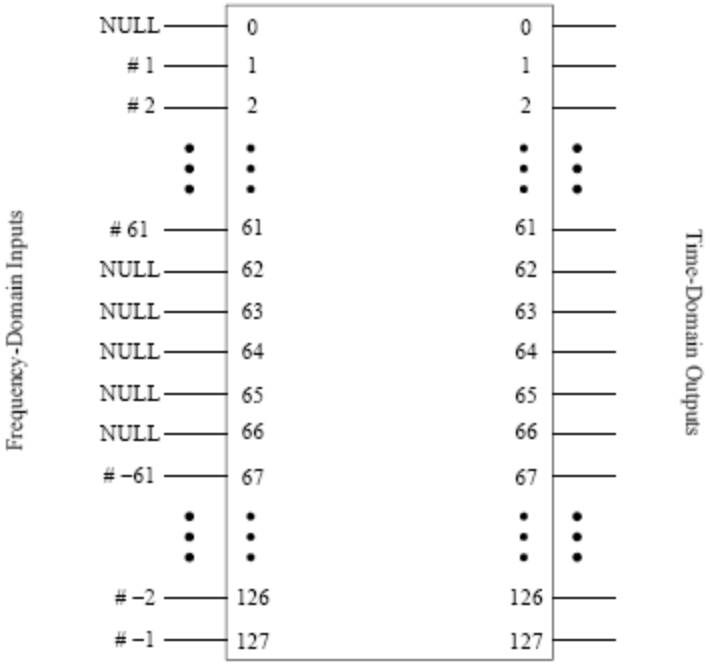


Figure 38. Input and outputs of IFF

5.7.1 IFFT design

For the reason that the IFFT core has a very complex structure and the design of this block is very time consuming we decided to use the IFFT core designed by Xilinx. using the Core Generator application in the Xilinx ISE software.The core that was used was the Fast Fourier Transform v3.2.

The specifications of our design indicate some constraints in the architecture option that we had to select.

- For real time operation, the IFFT calculation duration should be in the duration of an OFDM symbol including the guard interval.Thus the time available for a real time computation of the IFFT is limited to 165 clock cycles(structure of one OFDM Symbol (128 samples +37 guard samples).
- During the calculation of an IFFT, the data of the next OFDM symbol should have already been loaded into the input buffer of the IFFT block to allow a consecutive data transfer.

For these reasons,the architecture option that was selected was Pipelined, Streaming I/O which allows continuous data processing. This choice was the best we could do because the Pipelined, Streaming I/O solution pipelines several radix-2 butterfly processing engines to offer continuous data processing. Each processing engine has its own memory banks to store the input and intermediate data (Figure 39). The core has the ability to simultaneously perform transform calculations on the current frame of data, load input data for the next frame of data, and unload the results of the previous frame of data. The user can stream in input data and, after the calculation latency, can continuously unload the results.As a result the IFFT core,after a ceratin delay during the initial loading and calculation phase unloads results continuously. A new IFFT result vector(128 samples) will be available every 128 clock cycles after the first IFFT is completed. This satisfies the demand to operate within the duration of one OFDM symbol.

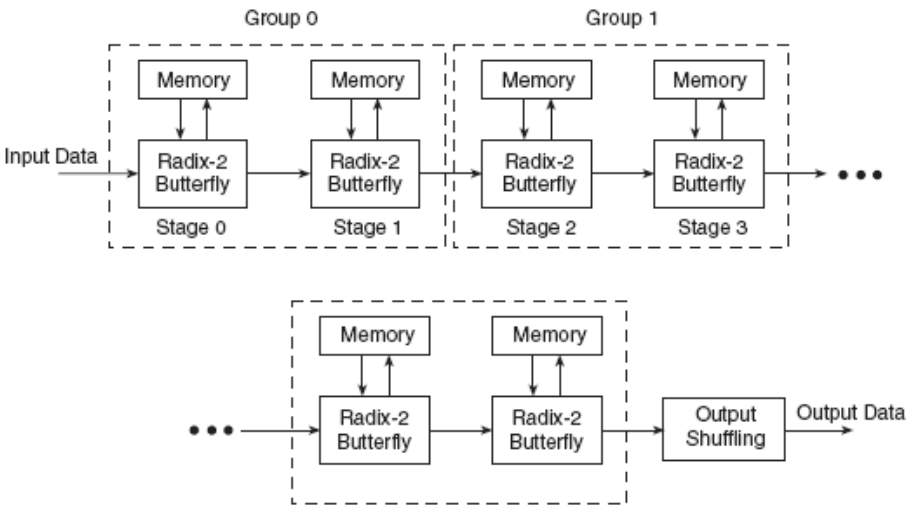


Figure 39. Pipelined, Streaming I/O

The IFFT core was customized with the following parameters:

- The pipeling streaming I/O architecture is used which pipelines several radix-2(DIF) butterfly processing engines to offer continuous data processing.It has log*2* (*N)* stages, with each stage containing, *N/2* radix-2 butterflies.(Figure 39-40).
- compute a N=2^7=128 point forward DFT or inverse DFT (IDFT).
- The input data is a vector of 128 complex values represented as 16-bit two'scomplement numbers – 16 bits for each of the real and imaginary components of the data sample.Similarly, the phase factors *is* 16 bits wide.
- The 128 element output vector is represented using 16 bits for each of the real and imaginary components of the output data. The output data width equals the input data width because scaled arithmetic is used.
- All memory is on-chip and was selected to use 2 Block RAMs in two stages for data and phase factor storage while the remaining stages use distributed memory.
- Input data are presented in natural order, and the output data in natural order
- Rounding after the butterfly.
- Scaled fixed-point, where the user provides the scaling schedule (see section 5.7.3 for more details).

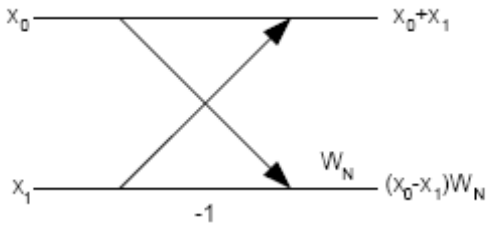


Figure 40. DIF butterfly

The above parameters we have selected for the Xilinx FFT core match with the function block parameters of the IFFT block used in the UWB simulink reference model.

5.7.2 Rounding Mode:

At the output of the butterfly, the LSBs in the datapath need to be trimmed. These bits can be truncated or rounded using convergent rounding, an unbiased rounding scheme.In our case convergent rounding was selected. When the fractional part of a number is equal to exactly one-half, convergent rounding rounds down if the number is odd (resulting in LSB=0), and rounds up if the number is even (resulting in LSB=1). Convergent rounding can be used to avoid the DC bias that would be introduced by truncation.

5.7.3 Scaling Schedule

The radix-2 FFT algorithm process an array of data by successive passes over the input data array.On each pass, the algorithm performs radix-2 butterflies, where each butterfly picks up two complex numbers and returns two complex numbers to the same memory.The numbers returned to memory by the processor are potentially larger than the numbers picked up from memory.A strategy must be employed to accommodate this dynamic range expansion. For radix-2, the values computed in a butterfly stage can experience a growth up to

$$1+\sqrt{2} \approx 2.414 .$$

This bit growth is handled by scaling at each stage using a fixed-scaling schedule. As a result of the scaling applied in the FFT implementation, the transform computed is a scaled transform.The scaling results in the final output sequence being modified by the factor *1/s*, where s is the scale Factor. For the inverse FFT, the output sequence is:

$$x(n) = \frac{1}{s}\sum_{k=0}^{N-1} X(k) e^{jnk2\pi/N} \qquad n = 0,\ldots,N-1$$

Scaled IFFT

$$s = 2^{\sum_{i=0}^{\log_2 N-1} b_i}$$

Scale Factor

where *bi* is the scaling (specified in bits) applied in stage *i*.

In the IFFT block of the UWB simulink model scaling occurs. Using fixed-point signals, the output of each butterfly of the IFFT is divided by two. Equivalently for radix-2, a scaling schedule of all 1's provides the factor of *1/N* in the case of FFT core by Xilinx.The scaling performed during successive stages can be set via the SCALE_SCH pin(see Table 17). For the pipelined streaming architecture, every pair of adjacent radix-2 stages is considered as a group. There are $\log_2$(point size)= $\log_2$(128)= 7 stages for Radix-2. That is, group 0 contains stage 0 and 1, group 1 contains stage 2 and 3, and so forth. The value of the SCALE_SCH bus is also used as pairs of bits [... N4, N3, N2, N1, N0]. Each pair represents the scaling value for the corresponding group of two stages. In each group, the data can be shifted by 0, 1, 2, or 3 bits which corresponds to SCALE_SCH values of 00, 01, 10, and 11. Groups are computed starting with group 0 as the two LSBs.

Considering all these in order to perform the scaling schedule of all 1's (data shifted by 1) which equals with the division by two of the output of each butterfly stage as uwb simulink model uses,we set the value of SCALE_SCH=[01 10 10 10].This translates to a right shift by 2 for group 0 (shift by one for stage 0 and shift by one for stage 1),a shift by 2 for group 1 (stages 2 and 3), a shift by 2 for group 3 (stages 4 and 5), and a shift of 1 for group 4 (stages 6). Note that when the point size is not a power of 4 (N=128), the last group only contains one stage,and the maximum

bit growth for the last group is one bit. Therefore, the two MSBs of the scaling schedule can only be 00 or 01.

Finally we have to mention that the core does not have a specific location for the binary point.The location of the binary point in the output data is inherited from the input data and then shifted by the scaling applied.In the end output data represented by 16 bits will have 2 bits for the integer part and 14 bits for the fractional part as sane as input data.

The signal description of the IFFT block and the interface diagram as our VHDL implementation indicates are shown below:

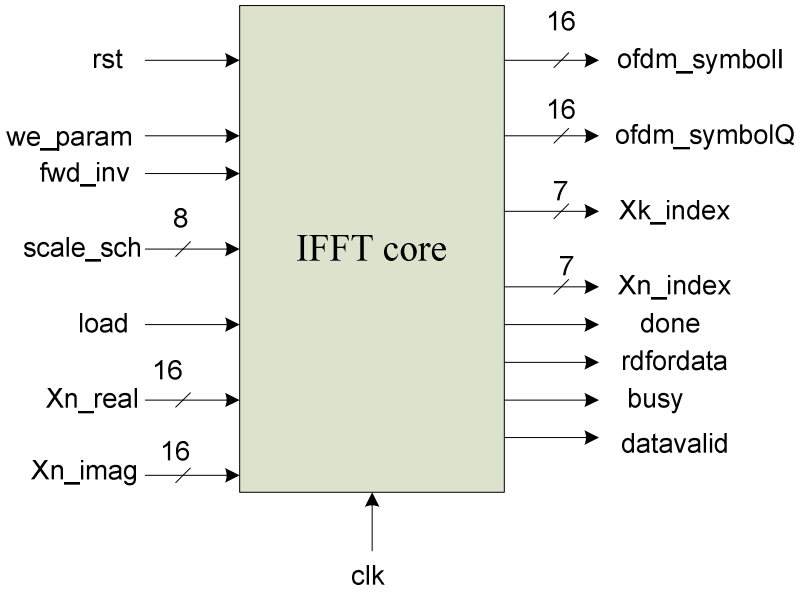


Figure 41.IFFT Interface

Table 17.IFFT signal description

signal	direction	widh	description
clk	input	1	Global clock
rst	input	1	Active high global reset.
Xn_real	input	16	Input data bus:Real component in two's complement format
Xn_imag	input	16	Input data bus: Imginary component in two's complement format
load	input	1	FFT start signal (Active High): For streaming I/O,load will begin data loading, which proceeds directly to transform calculation and then data unloading.
fwd_inv	input	1	Control signal that indicates if a forward FFT or an inverse FFT is performed. When fwd_inv =1, a forward transform is computed. If fwd_inv =0, an inverse transform is performed
scale_sch	input	8	Scaling schedule
we_param	input	1	Write enable for fwd_inv and scale_sch.
ofdm_symbolI	output	16	Output data bus: Real component in two's complement format.
ofdm_symbolQ	output	16	0utput data bus: Imaginary component in two's complement format.
xk_index	output	7	Index of output data.
xn_index	output	7	Index of input data.
rdfordata	output	1	Ready for data (Active High): is High during the load operation.
busy	output	1	Core activity indicator (Active High): This signal will go High while the core is computing the transform.
datavalid	output	1	Data valid (Active High): This signal is High when valid data is presented at the output.
done	output	1	FFT complete strobe (Active High): DONE will transition High for one clock cycle at the end of the transform calculation.

5.7.4 Timing for Pipelined Streaming I/O

Asserting load starts the data loading phase, which will immediately flow into the transform calculation phase and then the data unloading phase. Pulsing load once will allow the transform calculation for a single frame. Alternatively, holding load High will allow continuous data processing and that is exactly what we need. For this reason load is driven by signal "load_ifft" of Controller block output (see section 5.11) which is high as long as OFDM symbols are assembled and loaded continuously into the IFFT. Before the initial load,"we_param" must be asserted to register scale_sch and fwd_inv values internally. The scaling and transform type are constant through multiple frames,(that is, no new values are latched in) so registered values will apply for successive frames.Input data (Xn_real, Xn_imag) corresponding to a

certain “xn_index” should arrive three clock cycles later than the “xn_index” it matches. ”Rdfordata” will remain High with xn_index during the loading phase when it is valid to input data. “Busy” will go High while the core is calculating the transform. During the unloading phase, while valid output results are present on ofdm_symbolI/ofdm_symbolQ,datavalid (Data Valid) will be High.During unloading, “xk_index” will correspond to the ofdm_symbolI/ofdm_symbolQ being presented.

5.8 MULTIPLIERS

In order to have the same average power as the signals which are defined in the PLCP Preamble sequences (see paragraph 5.10), the output values I,Q of the IFFT block shall be normalized appropriately. For this reason we have used two multipliers that multiply the input I,Q by a constant value (11.5886) in the same way as the fixed point UWB simulink.

The binary fixed point representation of 11.5886 has precalculated with the help of Matlab functions and is “0101110010110101” using wordlength=16 and fractionlength=11 (5.11 fixed point format that means 11 bits of fraction,5 bits of integer). As we have mention before the fixed point representation of the I,Q output values of IFFT block has wordlength=16,fractionlength=14 (2.14 fixed point format that means 14 bits of fraction,2 bits of integer).

According to the fixed point arithmetic when you multiply two fixed point numbers in different fixed-points formats, (for example M.N format for the one and W.K for the other) the result is M+W.N+K. So multiplying IFFT I,Q values(2.14) with the constant gain(5.11) the result is a binary number with wordlength=32, fractionlength=25 (7.25). In order to have the same format for all the output samples of the transmitted (Preamble and I,Q results) ,as the UWB simulink model does, we chose to define 3.13 as fixed-point format. For this reason to get back to 3.13 fixed point format while the result of the multiplication is 7.25 we throw away(truncating) the bottom 12 bits of the fraction and the top 4 extra bits of the integer part of the number.

The signal description of the Mapper block and the interface diagram as our VHDL implementation indicates are shown below

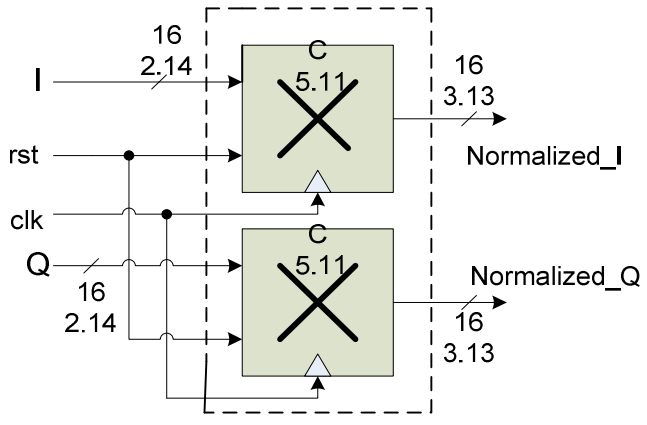


Figure 42.Normalize I,Q values using two registered multipliers

Table 18.Multipliers signal description

signal	direction	widh	description
clk	input	1	Global clock
rst	input	1	Active high global reset
I	input	16	Real part in two's complement format(2.14) which is to be multiplied by a constant value(11.5886)
Q	input	16	Imaginary part in two's complement format(2.14) which is to be multiplied by a constant value(11.5886)
Normalized_I	output	16	Output result of the multiplication I*C(3.13 format)
Normalized_Q	output	16	Output result of the multiplication Q*C(3.13 format)

5.9 ZERO PREFIX AND GUARD INTERVAL

Cyclic prefix is a crucial feature of OFDM to make the wireless communication system resistant to multi-path effects. To avoid symbol interference (ISI) and inter channel interference (ICI) successive.OFDM symbols are separated by guard band. In our case this is done by inserting zero prefix of 32 samples (1/4th of useful symbol period) at the beginning of each OFDM symbol. Also a guard interval of 5 zero samples at the end of each OFDM symbol (Figure 43) is added to allow the transmitter and receiver to switch from one sub-band to another.

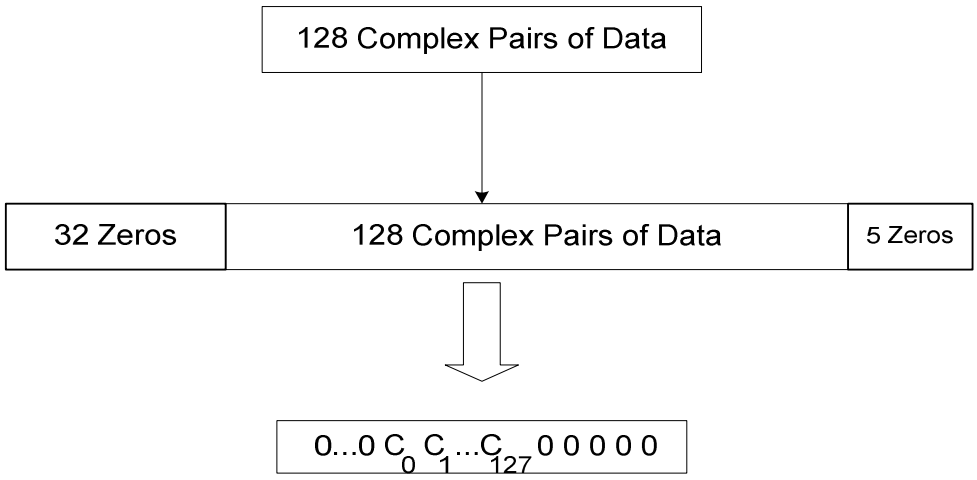


Figure 43.Zero prefix and Gurd Interval

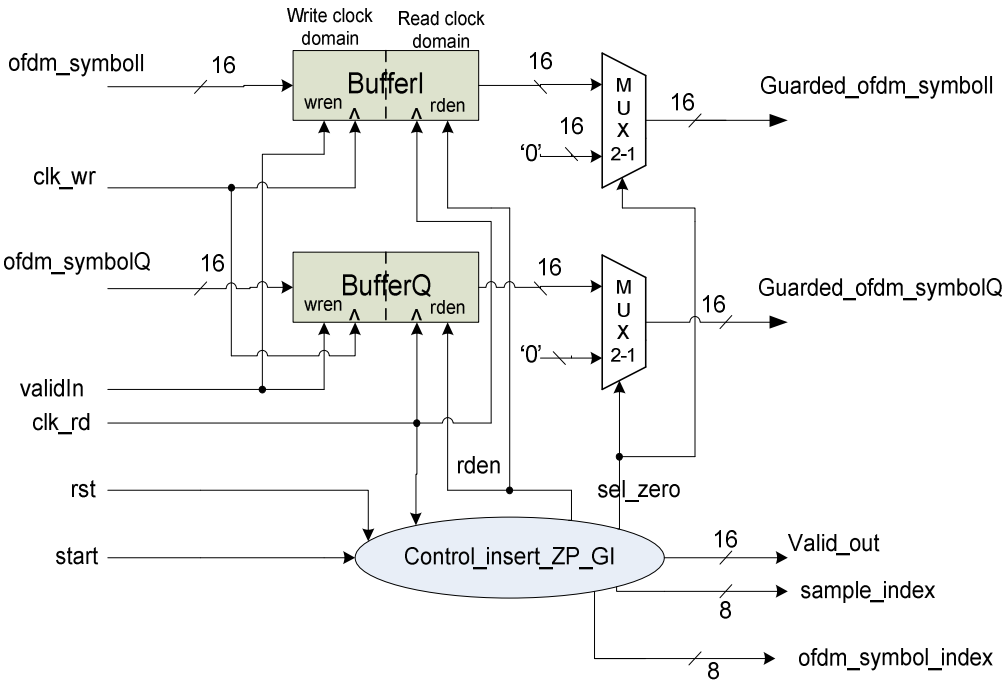


Figure 44.Zero prefix and Gurd Interval insertion block diagram

*All the components in this block diagram are synchronous with the clk input.

The signal description and the interface diagram as our VHDL implementation indicates are shown below:

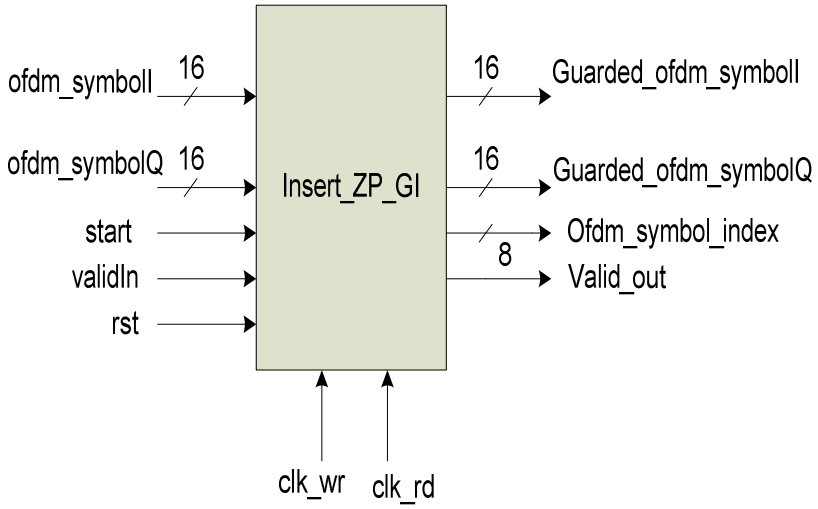


Figure 45.Zero prefix and Gurd Interval block Interface

Table 19. Zero prefix and Gurd Interval signal description

signal	direction	widh	description
clk_wr	input	1	Write clock
clk_rd	input	1	Read clock
rst	input	1	Asynchronous reset
ValidIn	input	1	Wren for Buffers I,Q to store valid OFDM_symbolI,Q
start	input	1	If high the process of transmiting data symbols with zero prefix and guard interval begins
ofdm_symbolI	input	16	Real input data stored into BufferI
ofdm_symbolQ	input	16	Imaginary input data stored into BufferQ
Guarded_ofdm_symbolI	output	16	OFDM symbol output with Zero prefix and Guard interval (real part)
Guarded_ofdm_symbolQ	output	16	OFDM symbol output with Zero prefix and Guard interval (Imaginary part)
Valid_out	output	1	Indicates valid samples that are transmited
ofdm_symblol_index	output	8	Index of trasmitted OFDM symbol

The purpose of this block is to prepend 32 zeros at the beginning of an OFDM symbol (128 complex pairs I,Q) and to append at least 5 zeros at the end. The total structure of the OFDM symbols becomes 165 samples (128 I,Q+37 zeros) which are to be transmited with a defined sample frequency of 528 MHZ. Due to hardware restrictions as far as the target device is concerned (Spartan-3 or Virtex-4) is not possible to achieve this sample frequency for this specific design (see conclusions and future work in section 8.1).

In addition input data (I,Q output results of streaming I/O IFFT operation) are generated continuously without any detours.These values are stored into bufferI and bufferQ respectively.The relative data input-output ratios for this module is 128 samples input (16-bit wide each) and 165 samples output (16-bit wide each) for both I and Q parts. To avoid full states of BufferI and BufferQ which will intercept continuous data processing and transmission of OFDM symbols we have precalculated the maximum number of OFDM symbols for a PLCP frame and we have used the maximum depth for the two buffers.

According to equation:

$$NSYM = 6\times [1/R \times (8 \times bytes + FCS\_bits + pad\_bits)]/(6\times NCBPS)].$$

the maximum number of OFDM symbols for a PLCP frame which is to be transmited with the mode of 110 Mbits/sec,thus R=5/8,bytes=4096(maximum number of payload information) and Ncbps=200.Finally this case gives 476 maximum number of OFDM symbols for the payload information adding the 6 OFDM symbols of the PLCP Header we have a total of 482 OFDM symbols in the worst case.

If the same clock source is used for read and write (figure 44), in order to read 128 samples from buffers and to insert 37 zeros through multiplexers it takes 165 clock cycles while simultanuesly 165 samples from IFFT will have been loaded into

each Buffer(I,Q).As a result the buffers will be filled up with 165-128=37 extra sample for each OFDM symbol. For this reason in order to ensure that FIFO buffers will never full in any case a maximum depth of 37*482=17834 should be used for BuffersI,Q.

Furthermore in order to exploit the simplicity of block insert_ZP_GI and to operate with the maximum frequency which will be the sample frequency of the transmitter we have considered using independent write and read clocks.The scale of write clock/read clock had to be larger or equal with 165/128 in order to avoid full or empty states for BufferI and BufferQ.

5.9.1 Control_ZP_GI :

In order to manage the insertion of zero values signal "sel_zero" is asserted for the first 32 cycles.Then starts reading Buffers I,Q asserting "rden" for 128 cycles (one OFDM symbol). After this, "rden" goes low and again 5 zeros samples are inserted at the output of each Buffer which is the guard interval.This procedure continuous iteratively until all OFDM symbols of a PLCP frame are transmitted(see figure 44).

5.10 PREAMBLE

A PLCP preamble contains some predefined values proposed in MultiBand OFDM Physical Layer Proposal for IEEE 802.15 Task Group 3a-September 14, 2004 that shall be added prior to the PLCP header to aid receiver algorithms related to synchronization, carrier-offset recovery, and channel estimation..The preamble signals are defined as real signals at the baseband – the baseband signal for lowest data rate modes is defined to be a real signal, and so this makes the preamble compatible with those modes. For the modes with data rates higher than 110 Mbps, where the transmitted signal is complex at the baseband, the PLCP preamble shall be inserted into the real part of the complex baseband signal.

In our case for data rates of 200 Mbps and lower, all the packets in the burst shall use the standard PLCP preamble**.** The standard PLCP preamble, which is shown in figure 33, consists of three distinct portions: packet synchronization sequence, frame synchronization sequence, and the channel estimation sequence (figure 46):

- The packet synchronization sequence shall be constructed by successively appending 21 periods, denoted as $\{PS_0, PS_1, \ldots, PS_{20}\}$, of a time-domain sequence.These time-domain sequences are precalculated values that are stored in a ROM and can be read out in time domain.This portion of the preamble can be used for packet detection and acquisition, coarse carrier frequency estimation, and coarse symbol timing.
- Similarly, the frame synchronization sequence shall be constructed by successively appending 3 periods, denoted as $\{FS_0, FS_1, FS_2\}$, of an 180 degree rotated version of the time-domain of packet synchronization sequence. Reading out the packet synchronization sequence which is stored in a ROM and performing inversion we managed to output very simply the frame synchronization sequence.This portion of the preamble can be used to synchronize the receiver algorithm within the preamble.

- Finally, the channel estimation sequence shall be constructed by successively appending 6 periods, denoted as $\{CE_0, CE_1, \ldots, CE_5\}$, of the OFDM training symbol. This training symbol has been converted into time domain using Matlab IFFT function before putting it into ROM , so no IFFT operation has to be performed on that preamble within the hardware. Also the IFFT results were normalized appropriately (multiplied by 11.586 again with the help of Matlab) to have the same average power as the Preample samples of PS and FS). This portion of the preamble can be used to estimate the channel frequency response, for fine carrier frequency estimation, and fine symbol timing.

Each period of the above sequence shall be constructed by pre-appending 32 "zero samples" and by appending a guard sinterval of 5 "zero samples" to the sequences.

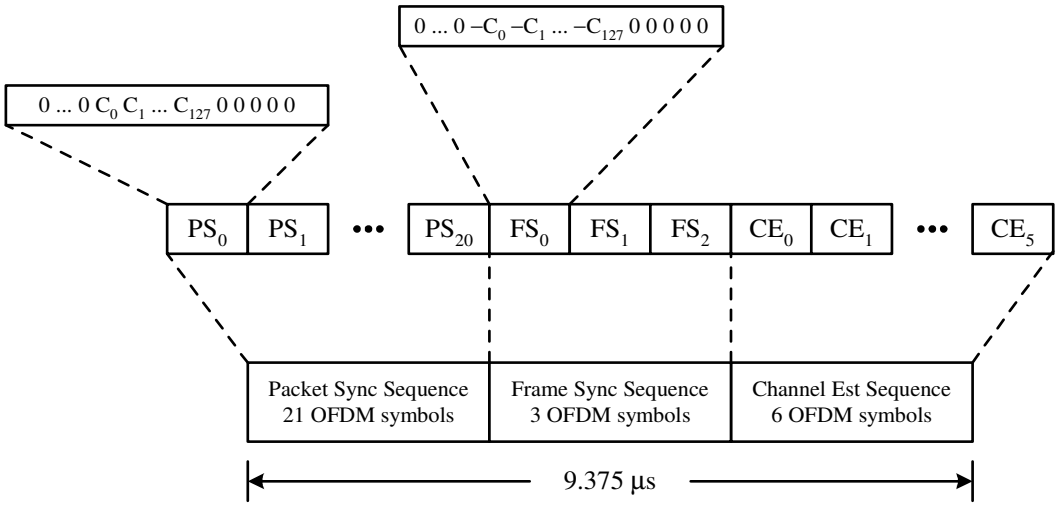


Figure 46 – Standard PLCP preamble format

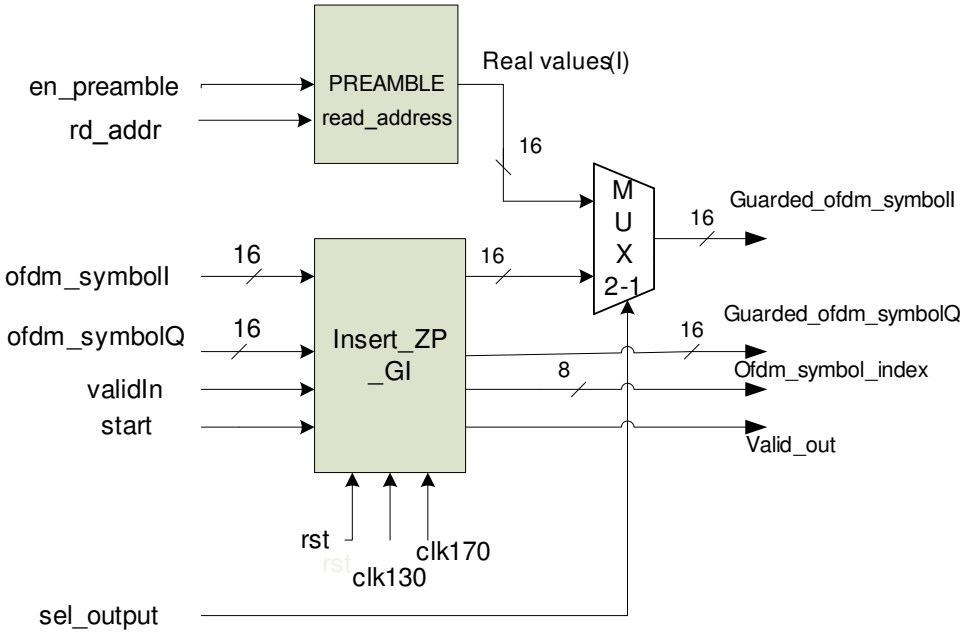


Figure 47 – Preamble insertion block diagram

Since the preamble is static for every message, we assume another module after the Transmitter will prepend the preamble. This module contains the ROM in which the preample paterns are stored,and a read address generator (figure 47). Reading of ROM is started right after a transmit request has occurred by pulsing “en_preamble” signal. Preamble pattern values are represented as 16 bits in fixed point format with 3 bits integer part and 13 bits fractional length as same as UWB simulink model (3.13 format).

5.11 CONTROL BLOCK

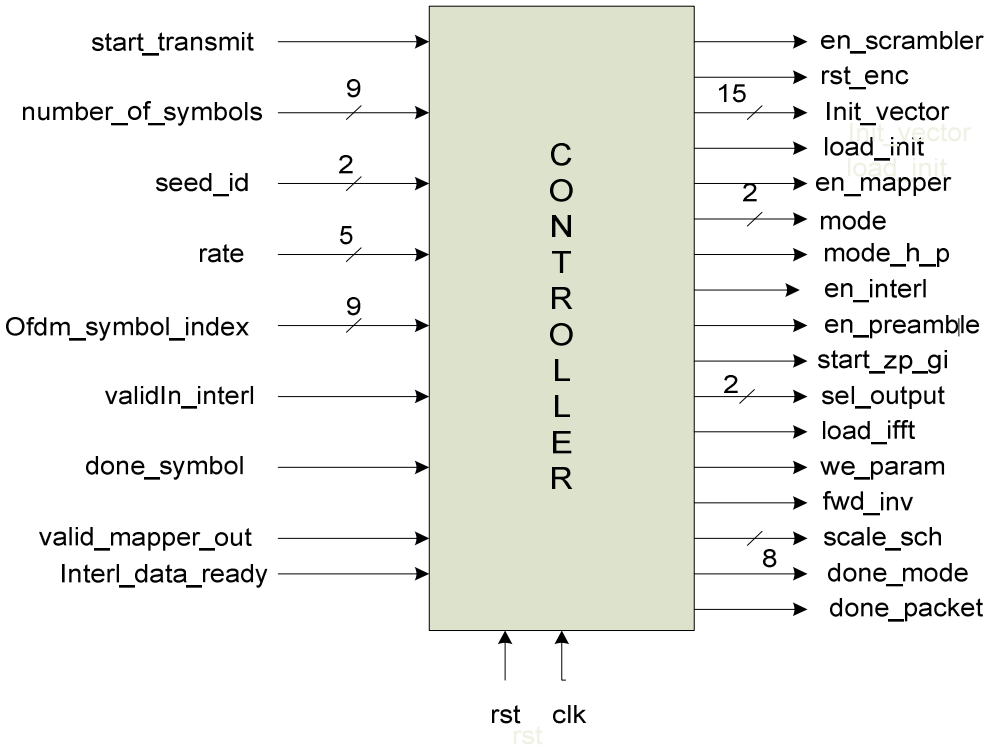


Figure 48 – Controller interface

5.11.1 Control process and signal description

The above block controller is used to define the most important control signals of every component of the transmitter allowing for operation on the desired data rate mode and for processing the entire PLCP frame appropriately (figure 48).

If “start_transmit” is asserted indicating a transmit request, the total number of OFDM symbols of the frame that must be transmited, calculated by the MAC layer, is provided to the controller as an input. Then Controller enables scrambler (“en_scrambler” goes high) and initializes it asserting “load_init” output signal and calculating the initialization vector (init_vector output signal) according to the “seed_id” input signal value. Output signals “mode” and “mode_h_p” are set to “00” and ‘0’ respectively to allow for Encoder and Mapper to operate on the 53.3 Mbits/sec data rate mode in order to process the PLCP header first. Simultaeously output signal “en_preamble” is asserted and output signal “sel_output” is 0 in order to start transmitting Preamble Patterns stored in ROM. When “validin_Interl” is asserted for the first time “en_interl” output signal goes high starting Interleaver process. The processing of the PLCP Header bits is continued until 6 OFDM symbols (600) bits have been interleaved. This is indicated by “ready_interl_data” input signal when it goes high. Then “en_scrambler” and “en_interl” goes low and starts the Mapper

operation by asserting “en_mapper” output signal. “Load_ifft”, “fwd”, “scale_sch” and “we_param” related to the IFFT operation were also initiliazed at that time. After loading 6 ofdm symbols (counting 6 pulses of “done_symbol” input signal to detect that), corresponding to the PLCP header information, loading phase of IFFT operation must be stoped. The output results of the IFFT operation are stored into BuffersI,Q and are not read yet since Preambles are still prepended. At this time Controler asserts “done_mode” signal to return all the components of the transmitter apart from block insert_zp_gi to the zero state. Then reads input signal “rate” which will define the mode for Payload transmission.(110 or 200 Mb/s). Signal ”mode” and “mode_h_p” are defined again accordingly. Also “en_interl” goes high again as soon as “validin_Interl” input signal goes high. This time after asserting“Load_ifft”a continuous processing is allowed for the rest OFDM symbols of the PLCP frame until all the OFDM symbols are laoded into IFFT (counting again pulses of “done_symbol” and comparing with input “numbers_of_symbols). ”Ofdm_symbol_index” input signal indicates the current OFDM symbols that is transmited. After transmitting 30 OFDM symbols which is the duration of the PLCP Preamble “en_preamble” goes low, “start_zp_gi “goes high and “sel_output” goes high. So OFDM symbols that were already stored into the Buffers I,Q start to be transmited. When the transmission of all ofdm symbols have been completed (“ofdm_symbol_index”= “number_of_symbols”) “done mode” is asserted reseting the system and “done_paket” also goes high indicating the completion of the packet transmission. At this time the system is in standby mode waiting for a new transmit request (“start_transmit”) and a new “number_of_symbols” value for the next paket. A general FSM (Finite State Machine) which describes the control of sample transmission is shown in figure 49.

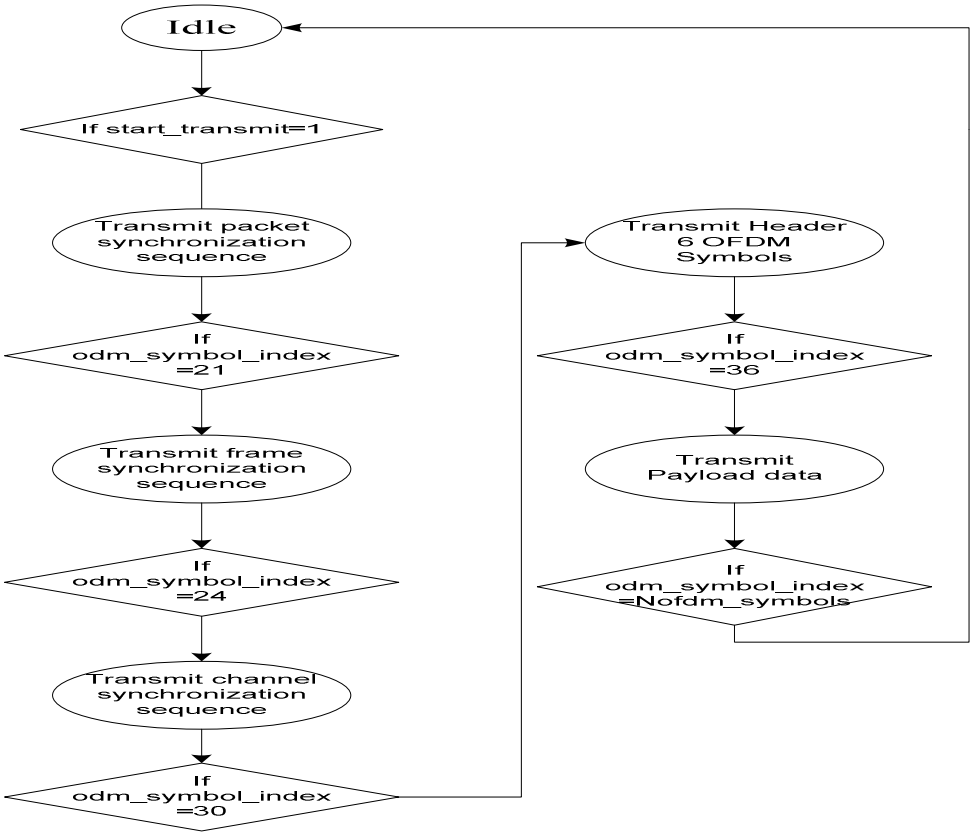


Figure 49 Transmit samples FSM

CHAPTER 6 RTL SIMULATION & VERIFICATION

After the hardware design of the UWB transmitter using VHDL has finished,RTL simulations are conducted to verify the functional correctness of all transmit Baseband functions for different data rates. When the individual modules testing and the system level testing has completed,the UWB transmited was synthesized using the Xilinx synthesis and design tool ISE 7.1. The target of the transmited implementation was initially a Spartan3-100 FPGA. Finally post-place and route simulation is done using Modelsim 6.0 to verify the functionality of our VHDL RTL code. Dowloading to the target FPGA has not done yet in order to test the design on a prototype board and this process is outside the scope of this report.

During the testing of the transmitter we have faced considerable difficulties for the reason that the reference UWB simulink model has many limitations and could not match exactly our hardware design which is more complete using the three mandatory modes (53.3,110,200 Mb/s) and other critical functions that were not part of the simulink model (scrambling,transmit header,control logic). Fortunately the reference model offered at least an excellent aid for the debugging of the individual components. Testing and evaluation is still in progress and the design of a similar receiver that could demodulate the transmited data and present the results would be the perfect solution for the system-level testing.

As shown in figure 14 (Chapter 4.1), applying the same set of vectors used in algorithm simulations to the RTL system and comparing the outputs it is verified that the different algorithmic blocks are implemented correctly in RTL.After each module (each transmit function) was designed in VHDL it was simulated (behavioural and post-place and route simulation) and the results were compared with the test vectors extracted by the simulink uwb model. Moreover sample results provided by the MBOA standard document were used to validate the functioning of the individual modules. For individual module testing we rely on the same input data used by simulink model and on output results comparison respectively (see Chapter 4.1). Some examples of the test scenarios that were applied are listed below:

- PLCP frames with maximum number of OFDM symbols for each mode:

  - rate= “00100”(mode 200 Mb/sec-coding rate=5/8)
  - number_of_symbols= “"111011100”(262)
  - seed_id= “01”

  - rate= “00000”(mode 53.3 Mb/sec-coding rate=1/3)
  - number_of_symbols= “"111101011”(491)
  - seed_id= “01”

  - rate= “00010”(mode 110 Mb/sec-coding rate=11/32)
  - number_of_symbols= “"100000110”(476)
  - seed_id= “01”

- PLCP frames with minimum number of OFDM symbols for each mode:

  - rate= “00100”(mode 200 Mb/sec-coding rate=5/8)
  - number_of_symbols= “000000001”(1)
  - seed_id= “01”

  - rate= “00000”(mode 53.3 Mb/sec-coding rate=1/3)
  - number_of_symbols= “000000001”(1)
  - seed_id= “01”

  - rate= “00010”(mode 110 Mb/sec-coding rate=11/32)
  - number_of_symbols= “000000001”(1)
  - seed_id= “01”
  -

- PLCP frames with selectable number of OFDM symbols for each mode with different seed_id values which initialize the scrambler in each case with a different vector:

  - rate= “00100”(mode 200 Mb/sec-coding rate=5/8)
  - number_of_symbols= “000001000”(8)
  - seed_id= “01”

  - rate= “00000”(mode 53.3 Mb/sec-coding rate=1/3)
  - number_of_symbols= “000001000”(8)
  - seed_id= “11”

  - rate= “00010”(mode 110 Mb/sec-coding rate=11/32)
  - number_of_symbols= “000001000”(8)”
  - seed_id= “10”

  - rate= “00100”(mode 110 Mb/sec-coding rate=5/8)
  - number_of_symbols= “000001000”(8)”
  - seed_id= “00”

- Succesive PLCP frames with different number of OFDM symbols each and different mode of transmission:

  PLCP frame1
  - rate= “00010”(mode 110 Mb/sec-coding rate=11/32)
  - number_of_symbols= “000011000”(24)
  - seed_id= “11”

  PLCP frame2
  - rate= “00100”(mode 200 Mb/sec-coding rate=5/8)
  - number_of_symbols= “000011110”(27)
  - seed_id= “10”

PLCP frame3

- rate= “00000”(mode 53.3 Mb/sec-coding rate=1/3)
- number_of_symbols= “000011111”(28)
- seed_id= “10”

In each case, 30 OFDM symbols (0 to 29) are transmitted first which correspond to the PLCP Preamble (21 periods of the packet synchronization sequence, 3 periods of the frame synchronization sequence and 6 periods of the channel estimation sequence).Then 6 OFDM symbols of the PLCP header are transmitted (only with mode 53.3 Mb/sec-coding rate=1/3). Finally the selectable number of Payload OFDM symbols follows and are transmited with the desired mode 53.3,110 or 200 Mb/sec.

Below a waveform is presented (figure 50) as an example of a test scenario that we have applied to the UWB transmitter.“Ofdm_symbol index”signal indicates the number of the current ofdm symbol that is transmitted. In this case the frame payload consists of 8 ofdm symbols that are transmited with rate of 200 Mbits/sec thus with coding rate=5/8.(rate= “00100”,seed_id= “01”, number_of_symbols= “000001000”). First the modulation of the PLCP header is performed using encoding rate=1/3 and conjucate symmetric operation.The I,Q results are stored into the output buffers I,Q until all the Preamble sequences have been transmited.After the PLCP Header modulation the system returns to zero state and starts modulating with the desire parameters(coding rate=5/8,no conjucate symmetric). At the output after 30 OFDM Symbols of the PLCP Preamble the 6 OFDM symbols of the PLCP Header follow and at the end the selectable number of Payload OFDM symbols(8).

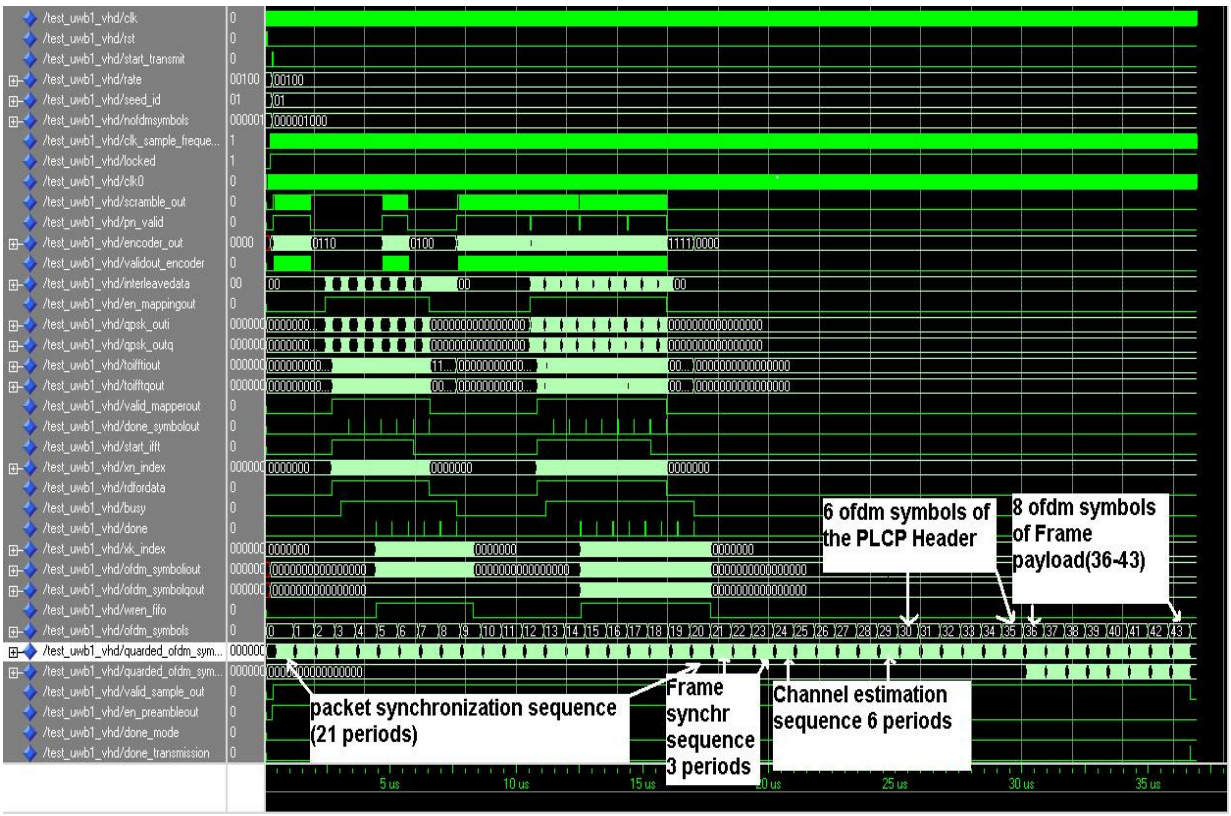


Figure 50. Waveform example of transmitting a PLCP frame with 8 OFDM Symbols Payload information.

CHAPTER 7 SYNTHESIS RESULTS

The design is to be loaded initially on an XC3S1000 device of SPARTAN 3 FPGA families, so there are some restrictions in the number of the logic devices that are supported by the FPGA. The gates, the output and input pins of each component, the multipliers, the flip flops and other logic components that are needed so as the model could be "run" in real time are defined by synthesizing. Synthesis results for the most important parts of the transmitter are shown below. The estimated clock frequency at which the FPGA can be run is also given.

1.Selected Device : 3s1000fg676-5
Speed Grade: -3

- **Convolutional encoder**

```
Device utilization summary:
---------------------------
```

Number of Slices:	7	out of	7680	0%
Number of Slice Flip Flops:	10	out of	15360	0%
Number of 4 input LUTs:	7	out of	15360	0%
Number of bonded IOBs:	8	out of	173	4%
Number of GCLKs:	1	out of	8	12%

Speed Grade: -5
Minimum period: 2.971ns (Maximum Frequency: 336.593MHz)

- **Encoder(convolutional encoder + puncturer)**

```
Device utilization summary:
---------------------------
```

Number of Slices:	144	out of	7680	1%
Number of Slice Flip Flops:	216	out of	15360	1%
Number of 4 input LUTs:	238	out of	15360	1%
Number of bonded IOBs:	12	out of	173	6%
Number of GCLKs:	1	out of	8	12%

Minimum period: 5.607ns (Maximum Frequency: 178.344MHz)

- **Bit interleaving**

```
Device utilization summary:
---------------------------
```

Number of Slices:	1128	out of	7680	14%
Number of Slice Flip Flops:	1709	out of	15360	11%
Number of 4 input LUTs:	1976	out of	15360	12%
Number of bonded IOBs:	17	out of	391	4%
Number of GCLKs:	1	out of	8	12%

Minimum period: 5.975ns (Maximum Frequency: 167.372MHz)

- **Mapper**

```
Device utilization summary:
---------------------------
```

Number of Slices:	191	out of	7680	2%
Number of Slice Flip Flops:	196	out of	15360	1%
Number of 4 input LUTs:	349	out of	15360	2%
Number of bonded IOBs:	75	out of	391	19%
Number of BRAMs:	4	out of	24	16%
Number of GCLKs:	1	out of	8	12%

Minimum period: 7.638ns (Maximum Frequency: 130.926MHz)

- **FFT+Multipliers**

```
Device utilization summary:
---------------------------
```

Number of Slices:	2116	out of	7680	31%
Number of Slice Flip Flops:	3511	out of	15360	24%
Number of 4 input LUTs:	2969	out of	15360	22%
Number of bonded IOBs:	96	out of	173	55%
Number of BRAMs:	4	out of	24	8%
Number of MULT18X18s:	14	out of	24	50%
Number of GCLKs:	1	out of	8	12%

Minimum period: 6.525ns (Maximum Frequency: 153.264MHz)

- **Insert_zp_gi**

```
Device utilization summary:
---------------------------
```

Number of Slices:	182	out of	7680	2%
Number of Slice Flip Flops:	191	out of	15360	1%
Number of 4 input LUTs:	317	out of	15360	2%
Number of bonded IOBs:	90	out of	391	23%
Number of BRAMs:	8	out of	24	33%
Number of GCLKs:	1	out of	8	12%

Minimum period: 5.395ns (Maximum Frequency: 185.352MHz)

- **Controler**

```
Device utilization summary:
---------------------------
```

Number of Slices:	100	out of	7680	1%
Number of Slice Flip Flops:	83	out of	15360	0%
Number of 4 input LUTs:	166	out of	15360	1%
Number of bonded IOBs:	75	out of	391	19%
Number of GCLKs:	1	out of	8	12%

Minimum period: 7.359ns (Maximum Frequency: 135.882MHz)

- **UWB Transmitter**

```
Device utilization summary:
---------------------------
```

Number of Slices:	4182	out of	7680	54%
Number of Slice Flip Flops:	6202	out of	15360	40%
Number of 4 input LUTs:	6540	out of	15360	42%
Number of bonded IOBs:	193	out of	391	49%
Number of BRAMs:	15	out of	24	62%
Number of MULT18X18s:	14	out of	24	58%
Number of GCLKs:	1	out of	8	12%

Minimum period: 8.308ns (Maximum Frequency: 120.372MHz)

Targeting at Xilinx Spartan-3 xc3s1000 and synthesising the transmitter design we noted that our implementation can sustain a clock frequency of up to 185 MHZ for block insert_ZP_GI and up to 120 MHZ for the rest blocks. So we could use 160 Mhz for the read clock and 120 Mhz for the write clock which create a scale larger and very close to 165/128(see paragraph 5.9). For this purpose we could use the single DCM module of Xilinx ISE 7.1 which allows the configuration of a clocking module using a single digital clock manager (DCM) and global buffers. .In this way we can operate at the maximum possible frequency of transmission for the specific target FPGA.

So using Spartan-3 xc3s1000 and having a clock frequency of 160 MHZ (sample frequency) successive ofdm symbols are transmited in a continuous way each having 128+37 samples(128 I,Q values+37 zeros). Every 165 clock cycles an new OFDM symbol is transmited with clock frequency of 160 Mhz.So the symbols rate is:

Symbol_Rate= F / #clock cycles=160/165= 0.96 Msps

Using the following equation:(without performing time spreading-Spreading=1):

*Data Rate=Symbol_Rate * Bits_per_Symbol * Coding_Rate/Spreading*

- For mode of 53.3 Mb/sec
  Bits_per_Symbol= 100 ,50 QPSK symbols
  Coding_Rate=1/3
  Spreading=1
  Data Rate=0.96*100*1/3=32 Mbits/sec

- For mode of 110 Mb/sec
  Bits_per_Symbol= 200 ,100 QPSK symbols
  Coding_Rate=11/32
  Spreading=1
  Data Rate=66 Mbits/sec

- For mode of 200 Mb/sec
  Bits_per_Symbol= 200 ,100 QPSK symbols

Coding_Rate=5/8
Spreading=1
Data Rate=120 Mbits/sec

Our UWB transmitter transmits data with the above data rates without performing time-spreading (repetition of the same ofdm symbol)which is mandatory. If we had perform time spreading and the same ofdm symbol was transmited twice (need twice duration for the same information) these values are divided by 2. For example in the case of 200 Mb/sec=>Data Rate=60 Mbits/sec. In later section we will explain a solution of performing time-spreading without decreasing the data rates of our system (chapter 8.2).

In practice it was very difficult our system to work with two different clock sources due to clock syncronization problems. Especially when a global controler was designed(see chapter 5.11) which defines control signals even for the block insert_zp_gi (which in theory could sustain the clock frequency of 160 MHZ) it is obvious that the whole system could not sustain a clock frequency of up to 160 MHZ. After Synthesis and place and route our transmitter work perfect with a clock frequency of 115 Mhz. Finally targeting at Xilinx Spartan-3 xc3s1000 we can achieve:

- For mode of 53.3 Mb/sec(without time-spreading)
  Data Rate=0.69*100*1/3=23.2 Mbits/sec

- For mode of 110 Mb/sec(without time-spreading)
  Data Rate=0.69*200*11/32=47.4 Mbits/sec

- For mode of 200 Mb/sec(without time-spreading)
  Data Rate=0.69*200*5/8=86.25 Mbits/sec

It is obvious that a different target FPGA(Virtex-4 or Virtex-5) should be used to achieve a better performance.

2.Selected Device : 4vfx40ff672-12
Speed Grade: -12

- **UWB transmitter**

```
Device utilization summary:
---------------------------
```

Number of Slices:	4343	out of	18624	23%
Number of Slice Flip Flops:	6215	out of	37248	16%
Number of 4 input LUTs:	6584	out of	37248	17%
Number of bonded IOBs:	193	out of	410	47%
Number of FIFO16/RAMB16s:	15	out of	144	10%
Number used as RAMB16s:	15			
Number of GCLKs:	1	out of	32	3%
Number of DSP48s:	14	out of	48	29%

Minimum period: 4.028ns (Maximum Frequency: 248.278MHz)

Another interesting result is the operation clock frequency that Vitex-4 FPGA gives.

- **Insert_zp_gi**
  Minimum period: 2.924ns (Maximum Frequency: 342.032MHz)

- **Mapper**
  Minimum period: 3.988ns (Maximum Frequency: 250.771MHz)

- **Controler**
  Minimum period: 4.028ns (Maximum Frequency: 248.278MHz)

Using Vitex-4 the same design allows to use a sample frequency of 340 Mhz at the insert_zp_gi block but the block controler can not sustain such a clock frequency. Finally we have used again the same clock for all the components of the transmitter and the operation frequency with which our transmitter run was 230 MHZ.

In this case: Symbol_Rate= F / #clock cycles=230/165= 1.39 Msps

- For mode of 53.3 Mb/sec
  Bits_per_Symbol= 100 ,50 QPSK symbols
  Coding_Rate=1/3
  Spreading=1
  Data Rate=1.39*100*1/3=46.46 Mbits/sec—23.32 Mbits/sec if time spreading occurs(Spreading =2)
- For mode of 110 Mb/sec
  Bits_per_Symbol= 200 ,100 QPSK symbols
  Coding_Rate=11/32
  Spreading=1
  Data Rate=95.83 Mbits/sec— 47.9 Mbits/sec if Spreading=2

- For mode of 200 Mb/sec
  Bits_per_Symbol= 200 ,100 QPSK symbols
  Coding_Rate=5/8
  Spreading=1
  Data Rate=174.24 Mbits/sec—87.12 Mbits/sec if Spreading=2

3.Selected Device : xc5vlx50-3ff676
Speed Grade: -3

- **UWB transmitter**

Logic Utilization	Used	Available	Utilization
Number of Slice Registers	6150	28800	21%
Number of Slice LUTs	6256	28800	21%
Number of fully used Bit Slices	4791	7615	62%
Number of bonded IOBs	193	440	43%
Number of Block RAM/FIFO	8	48	16%
Number of BUFG/BUFGCTRLs	2	32	6%
Number of DSP48Es	14	48	29%

Minimum period: 3.341ns (Maximum Frequency: 299.312MHz)

As shown above using Vitex-5 for the same design a sample frequency of 264 Mhz for all the components of the transmitter can be used. This operation frequency gives us the following estimated values:

Symbol_Rate= F / #clock cycles=264/165= 1.6 Msps

- For mode of 53.3 Mb/sec
  Bits_per_Symbol= 100 ,50 QPSK symbols
  Coding_Rate=1/3
  Spreading=1
  Data Rate=1.6*100*1/3=53.3 Mbits/sec—26.66 Mbits/sec if time spreading occurs(Spreading =2)
- For mode of 110 Mb/sec
  Bits_per_Symbol= 200 ,100 QPSK symbols
  Coding_Rate=11/32
  Spreading=1
  Data Rate=110 Mbits/sec— 55 Mbits/sec if Spreading=2
- For mode of 200 Mb/sec
  Bits_per_Symbol= 200 ,100 QPSK symbols
  Coding_Rate=5/8
  Spreading=1
  Data Rate=200 Mbits/sec—100 Mbits/sec if Spreading=2

Finally the duration of one OFDM symbol is 165*(1/115 MHz)=1.434 ms if the target device is Spartan3,165*(1/230 MHz)=717 ns if the target is Virtex4 and 165*(1/264 MHz)=625 ns

Table 20.implemented rate-dependent timing parameters

desired timing parameter values	Spartan3	Virtex4	Virtex5

Data Rate 53.3 Mb/sec	23.2 Mb/s	46.46 Mb/s	53.3 Mb/s
Data Rate 110 Mb/sec	47.4 Mb/s	95.83 Mb/s	110 Mb/s
Data Rate 200 Mb/sec	86.25 Mb/s	174.24 Mb/s	200 Mb/s
Sample frequency= 528 MHz	115 MHz	230 MHz	264 MHz
OFDM Symbol duration=312.5 ns	970 ns	717 ns	625 ns

Untill now the estimated timing parameters were calculated assuming that no time-spreading is performed. Otherwise performing time-spreading (repeating the same OFDM symbol at the output of the transmitter) will divide the above results with 2. Because time-spreading is mandatory for the UWB standard we have to find a solution to perform time-spreading without decreasing the data rates we have achieved.(see chapter 8.2 future work).

CHAPTER 8 CONCLUSIONS AND FUTURE WORK

8.1 CONCLUSIONS

The UWB transmitter is defined to have a bandwidth of 528 MHz. Traditionally the hardware implementations of digital baseband have been running about the same frequency that is the bandwidth of the system. That means that the operation clock frequency which is the sample frequency of the transmitter should be 528 MHz. Using Spartan3 FPGA as a target for our design due to its low cost , the operation clock frequency is limited at 120 MHz. Even if we have considered to exploit the simplicity of the last block of the transmitter (insert_zp_gi) using two different clock frequencies,160 Mhz for the last part of transmited that outputs OFDM symbols and 120 MHz for the rest parts, the complexity of the total transmitter design could not sustain a clock frequency of 160 MHz (limited at 120 MHz). Finally the lowest clock frequency was used for all the components of the transmitter to avoid clock synchronization problems and problems that were emerged in place and route simulation. The best we could do to achieve as higher transmission taxes as possible is to use a Virtex-4 or even Virtex-5 FPGA to achieve transmission rates as close as possible to specifications.Using Virtex-4 the operation clock frequency is 230 MHZ while using Virtex-5 is possible for our design to operate with 264 MHz clock frequency which is the half of the sample frequency of the transmitter. In this way we can achieve up to 200 Mbits/sec data rate but without performing time spreading. Because time-spreading is mandatory for the UWB standard we have to find a solution to perform time-spreading without decreasing the data rates we have achieved.

8.2 FUTURE WORK

First of all,in order to achieve a real-time UWB transmitter in the future we must focus on a different target FPGA, specifically choosing Virtex5. We have already discussed the limitations regarding the sample frequency that Spartan 3 FPGA poses. Virtex-4 is a satisfactory solution but is not enough to achieve the full rates. Higher data rates very close or equal to the specifications can be obtained using an operation clock frequency same as the bandwidth of the UWB transmitter (528 MHZ). Such a high clock frequency of operation that will be used as the sample frequency of the transmitter can possibly be given by a very simple circuit using Virtex-5. It is also crucial that time-domain spreading should be performed.

.

For data rates of 53.3,110 and 200 Mbps a time-domain spreading operation shall be performed with a spreading factor TSF =2, in order to improve frequency diversity and SOP performance. The time-domain spreading shall consist of transmitting the same information over two OFDM symbols. One idea of employing time-domain spreading and also to speed up the data throughput , even if slow Spartan3 is used, is the implementation of two parallel processing paths.Unfortunately the current device utilization of our design exceeds the 50% of the available resourses of Spartan3-xc3s1000. For this reason the idea of parallelization has rejected.

However, using two different buffers and reading them in cascade with 2x clock frequency than the rest of the design which stores the OFDM symbols twice, corresponds to a very simple circuit which will sustain a clock frequency of 528 MHz (using Virtex-5). Based on the idea of using 2 different clocks we can employ time spreading without dividing data throughput by 2 and the very data rates of the specifications can be achieved.(figure 51).

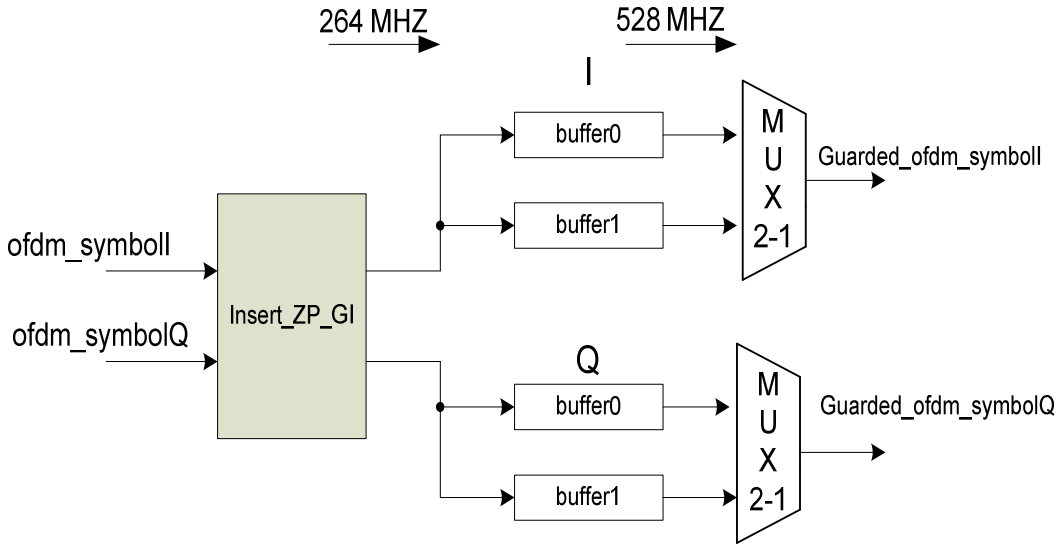


Figure 51-An idea for performing time-domain spreading and improving performance as well.

Other possible projects that can improve and extend the current implementation could be:

- Testing of the design on a prototype board.(Hardware debugging)
- Power and area optimization(probable VHDL source code optimization)
- Design and implementation of the Receiver part of UWB.Receive baseband functions has to be implemented:
  - Channel estimation and compensation,
  - FFT
  - Demapper,
  - Deinterleaver
  - Viterbi decoder
  - Descrambler
- Transmited interface between FPGA and analog front-end
- PHY/MAC interface

BIBLIOGRAPHY

[1] Xilinx official website, http://www.xilinx.com/

[2] Mathworks official website, http://www.mathworks.com/

[3] "Ultra-Wideband (UWB) Technology, One Step Closer to Wireless Freedom" , http://www.uwbcomm.com/

[4] "Ultrawideband: A Better Bluetooth" , UWB Forum , http://www.computerworld.com/mobiletopics/mobile/technology/story/0,10801,94896,00.html

[5] "Wimedia: The Ultra-Platform for Wireless Multimedia", 2006 , http://www.wimedia.org/en/resources

[6] Ed Thomas,"Ultra-Wideband:Four Years Later", 2006 , http://www.wimedia.org/en/resources

[7] Jing Jin , "Ultra-Wideband (UWB)", http://www.tml.tkk.fi/Opinnot/T-109.7510/2006/reports/UWB.doc

[8] "Ultrawideband: High-speed, short-range technology with far-reaching effects", Multiband OFDM Alliance Special Interest Group , 2004.

[9] "A technology comparison" , Memsen Corporation , http://wireless.fcc.gov/outreach/2004broadbandforum/ltrawideband.pdf

[10] Farinaz Edalat , "Behavioral Simulation of a Basic OFDM Transceiver using the CppSim Program" , Part I: Discrete-Time , May 13, 2004 , http://www-mtl.mit.edu/~perrott

[11] "Orthogonal Frequency Division Multiplexing (OFDM)-Applications for Wireless Communications with Coding" , http://www.comlab.hut.fi/opetus/311/ofdm_mod.pdf

[12] "E225C Lecture 10: Introduction of OFDM" , http://bwrc.eecs.berkeley.edu/Classes/EE225C/Lectures/Lec16_ofdm.pdf

[13] Dr. Jean Armstrong , "OFDM – Orthogonal Frequency Division Multiplexing" Department of Electronic Engineering , La Trobe University , http://www.ctie.monash.edu.au/ofdm/sample_files/armstrong_ofdm.pdf

[14] "MultiBand OFDM Physical Layer Proposal for IEEE 802.15 Task Group 3a" , Multiband OFDM Alliance Special Interest Group March, 2004.

[15] "MultiBand OFDM Physical Layer Proposal for IEEE 802.15 Task Group 3a" , Multiband OFDM Alliance Special Interest Group , September 14, 2004.

[16] “Part 15.3: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for High Rate Wireless Personal Area Networks (WPANs)” , IEEE Computer Society.

[17] Peter Aderson ,”The VHDL Cookbook”,first edition.

[18] Seyed Mohamad Sajad Sadough, Asad Mahmood, Emmanuel Jaffrot and Pierre Duhamel “Performance Evaluation of IEEE 802.15.3a Physical Layer Proposal Based on Multiband-OFDM”.

[19] A. Batra, J. Balakrishnan and A. Dabak, “Multiband OFDM: a new approach for UWB,” Internat. Symp. on Circuits Systems, May 2004.

[20] J.W. Cooley and J.E. Tukey, “An Algorithm for the Machine Calculation of Complex Fourier Series”, Math. Comput., Vol. 10, pp. 297-201, April 1965.

[21] Moises Serra, Juli Ordeix, Pere Marti, “OFDM Demonstrator: Transmitter”.

[22] S.B. Wicker, “Error Control Systems for Digital Communication and Storage”, Prentice Hall, 1995.

[23] Richard Van Nee and Ramjee Prasad, “OFDM for Wireless Multimedia Communications”, Artech House, Boston, 2000.

[24] Aseem Pandey, Shyam Ratan Agrawalla & Shrikant Manivannan from Wipro Technologies, “VLSI implementation of OFDM modem”.

[25] J.G.Proakis, D.G Manolakis, “Digital signal Processing”,Third Edition.

[26] Erick L.Oberstar, “Fixed Point Represantation and Fractional Math”, 2004-2005, Oberstar Consulting., http://www.superkits.net/whitepapers/Fixed%20Point%20Representation%20&%20Fractional%20Math.pdf

[27] Eldon Staggs , “Ultrawideband Radio Design , Sustem Analysis”, Presentation 1b, Worldwide Technical Workshop, Partners in design, Ansoft Corporation.

[28] Jossie Roivainen, Yan Zhan, Nils Rinaldi ,Christophie Devaucelle, Regis Cattenoz, Dirk Pannicke, “Implementation of the selcted PHY algorithms realized in VHDL and related resource allocation (HDR, VHDR)”, Integrated Projecr Pulsers.

[29] Fredrik Kristensen, “Design and Implementation of Flexible OFDM Hardware” , April 2004, Department of Electroscience , Lund University.

[30] Tonny Matos Siqueira, “Implementacao de um modem OFDM em FPGA”, April 2004, http://www.opencores.org.