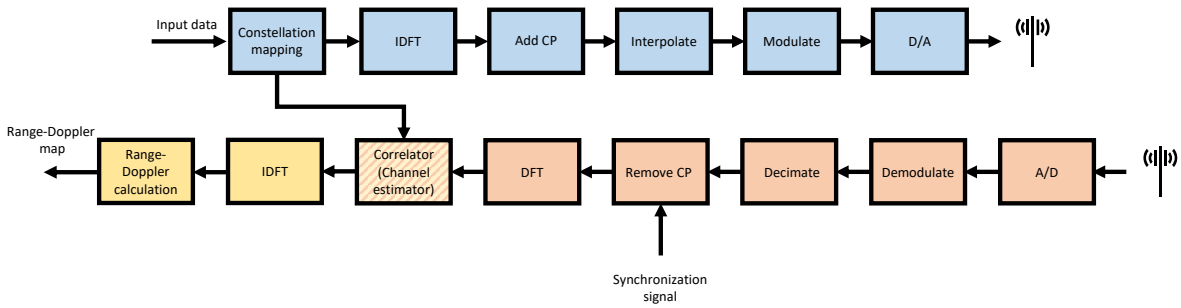
CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG



OFDM digital radar design for Integrated Sensing and Communication (ISAC) in 6G networks

Master's thesis in Embedded Electronic System Design

JORGE MUNOZ BAUTISTA

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2024

Master's thesis 2024

OFDM digital radar design for Integrated Sensing and Communication (ISAC) in 6G networks

JORGE MUNOZ BAUTISTA

UNIVERSITY OF
GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2024

OFDM digital radar design for Integrated Sensing and Communication (ISAC) in 6G networks
JORGE MUNOZ BAUTISTA

Supervisor: Per Larsson-Edefors, MC2
Advisor: Christoffer Fougstedt, Ericsson AB
Examiner: Lars Svensson, MC2

Master's Thesis 2024
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Proposed block design for a channel estimation-based FMCW radar.

Typeset in LaTeX
Gothenburg, Sweden 2024

OFDM digital radar design for Integrated Sensing and Communication (ISAC) in 6G networks
JORGE MUNOZ BAUTISTA
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

In this project a digital Frequency Modulated Continuous Wave (FMCW) radar was designed using a Orthogonal Freqeuncy Division Multiplexing (OFDM) pulse compatible with the 5G standard. A system level simulation was created to test the design using MATLAB. The design was then implemented using System Verilog. Two iterations of the design were created using purpose made modules and another one combining IP modules with purpose-made ones. The first iteration was tested on a MATLAB testbench based on the system-level simulation. Finally, the digital radar was combined with an existing project to create a testbench that was run on the ZCU111 development board. As a result of this project, it was found that 5G pulses offer good range resolution but poor speed resolution. Also, the effect of quantization noise for a 12 bit word length was found to be negligible. Finally, it was found that adequately scaling the signal in the FMCW OFDM radar design was fundamental to avoid overflow conditions.

Keywords: ISAC, radar, 5G, 6G, FMCW, RFSoC, Range-Doppler.

Acknowledgements

Many thanks to my company supervisors Henric Broström and Christoffer Fougsted for for their continuous support during the entire thesis. I am also thankful to my academic supervisor Per Larsson-Edefors for always helping me to see the big picture and asking the difficult questions. Thanks should also go to my examiner Lars Svensson for his feedback on the report. Lastly, I'd like to thank the Antenna and Microwave Hardware department in Ericsson Research for helping me to learn more about 5G and telecommunications.

Jorge Munoz Bautista, Gothenburg, June 2024

Contents

1

Introduction

The demand for higher data rates in mobile networks has pushed manufacturers to design systems that use higher frequency bands and to implement more complex multiplexing techniques such as Space-Division Multiplexing (SDM) using beamforming. As a result of this drive to increase performance in mobile networks, the hardware gap between telecommunications equipment and scanning radars has almost disappeared. Moreover, the access to higher frequency bands in communications makes it possible to design radar systems with down to to 1 cm resolution [1] in these bands. Industry is capitalizing on these developments for the new 6G standard. One of the key features of 6G networks will be Integrated Sensing and Communications (ISAC) [2]. The 6G network is expected to have the ability to perform radar-like sensing using the communication hardware. Sensing in 6G networks will be used to enhance performance of the network itself and will also be offered to clients who will be able to use the radar technology as an all-weather imaging system for, among other applications, traffic control or tracking autonomous vehicles in industrial settings [2].

In the context of mobile networks, it is important that sensing does not degrade the communication performance. With the 6G network expected to use some variation of the Orthogonal Frequency-Division Multiplexing (OFDM) pulse [1], a radar system compatible with this pulse is thus a very desirable design. The purpose of this Master's Thesis is to design and test a digital radar based on OFDM pulses.

A Frequency Modulated Continuous Wave (FMCW) radar prototype will be built as a Software Defined Radio (SDR) using a Radio Frequency System on Chip (RFSoC) development board. The RFSoC consists of a FPGA, a Microcontroller and high-speed Analog to Digital (A/D) and Digital to Analog (D/A) converters. The radar design will be implemented using System Verilog (SV) and the output will be a formatted vector containing the reflections of different targets at different ranges (range map). An optional goal of this project is to use an RF frontend to create a complete demonstrator of the technology.

Since the 6G standard is not fully specified at the time of writing this report, it has been decided to base the OFDM pulse on the 5G standard. At the physical layer[1], 5G and 6G are expected to have a high degree of commonality both for backwards compatibility reasons and because the 5G physical layer is already well optimized. Thus, the 6G standard will likely use the same waveform, the same frame length and

[1]The physical layer is used to transmit the raw bits of data. It includes pulse waveform, data modulation and timing

will support the same bandwidths as 5G [1] which are the most relevant parameters in the radar design.

1.1 Related work

ISAC has been identified as one of the new features for the 6G standard as part of the Hexa-X project, an EU funded project to investigate the requirements for 6G. The Hexa-X project has published deliverables [2], [3] and [4] underlining the importance of the ISAC technology and proposing potential use-cases.

The concept of combining radar sensing and communications has been studied before in academia and industry. The first mentions of OFDM radar, in particular, date back to 2006 [5]. Currently, the main body of work in the OFDM radar and ISAC areas centers on vehicular network applications [6] [7] [8]. However, the same fundamentals can be applied to mobile networks by increasing the weight of the communication in the system design and choosing suitable communication bands [1].

Using SoCs in SDR applications has become prevalent to the point that purpose-built devices such as the ZCU111 RFSoC [9] have been developed to meet this demand. The telecommunications industry makes extensive use of the different SDR architectures to implement standards ranging from Wi-Fi to 4G and 5G [10].

1.2 Goals and objectives

The goal of this project is to develop the digital backbone of an OFDM radar that will be used as part of a ISAC system in a 6G network. The project can be further divided into smaller objectives: choose a radar design that reuses the maximum number of components from a OFDM receiver, design a system-level simulation of the selected radar design which can be used to determine the performance of the system, implement a digital radar system using System Verilog, interface the system-level simulation with the Register Transfer Level[2] (RTL) design to test the RTL design and design a testbench with a known delay between the input and output ports and test the RTL implementation. Optionally, upgrade the testbench with a RF frontend.

The radar will be adapted from designs present in literature. These designs will be adapted to minimize the resource overhead when reusing an OFDM receiver.

The system-level simulation in MATLAB will be used to evaluate the radar system and as a basic for the HDL testbench. The simulation will also help to explain the concepts of the OFDM radar when presenting the project. An important feature of the system-level simulation will be the possibility to choose between different 5G specifications for bandwidth, number of subcarriers and cyclic prefix length. The different 5G parameters will yield different range and speed measuring performance

[2]The RTL design abstraction models synchronous digital circuits as interconnected registers in which data is stored and operated on.

for the radar and will impose different latency and throughput constraints on the RTL implementation.

1.3 Thesis outline

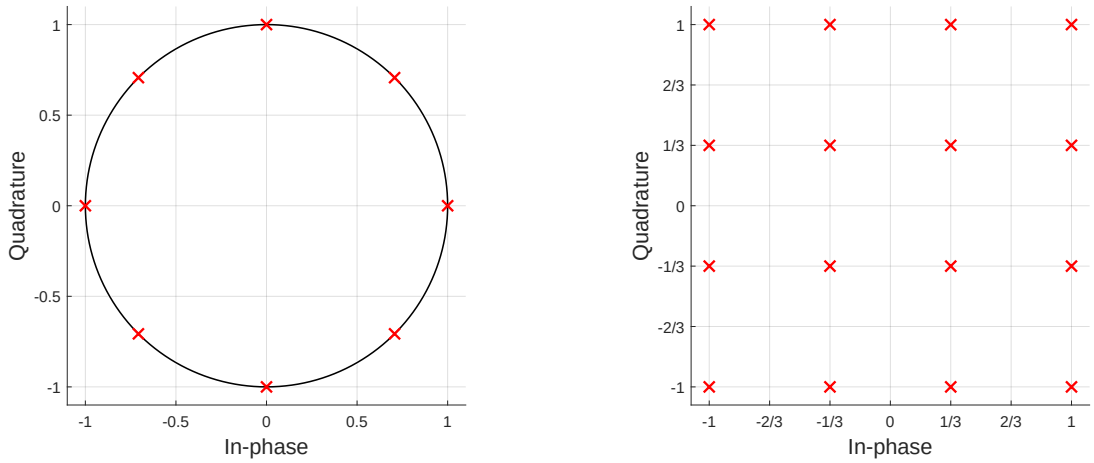
The report begins with an overview of the theoretical concepts that underline this project in Cahpter 2. Chapter 3 will include descriptions of the procedures used at the different stages of the project to develop and test the radar. Next, Chapter 4 will provide an overview of the design with emphasis on the trade-offs that lead to the final design. The results will be presented and discussed in Chapter 5. Finally, a conclusion will be drawn and further work will be outlined in Chapter 6.

2

Technical background

In this chapter the theoretical background relevant to this Master's Thesis will be covered. First, digital modulation will be explained. Next, OFDM pulses will be considered as well as the parts of an OFDM receiver and transmitter. An explanation of the radar that will be used in this project will follow as well as some common architectures. The 5G NR standard will also be explained emphasizing the OFDM pulse characteristics. An overview of the development board will be the next topic. The chapter will finish with an overview of IP blocks in the Xilinx library.

2.1 Digital modulation.



(a) 8-PSK constellation diagram.

(b) 16-QAM constellation diagram.

Figure 2.1: Digital constellation examples

Digital data can be transmitted using a wide range of modulation schemes [11]. A modulation scheme can be thought of as a mapping of a group of bits to a symbol that can be then transmitted. The most widely used modulations in mobile networks are phase shift keying (PSK) and quadrature amplitude modulation (QAM). Generally, the modulation scheme has a power of two number of symbols.

Digital communication systems generally use complex quantities to describe the symbols. The real part is referred to as the in-phase signal and the imaginary part

as the quadrature signal. The in-phase signal and the quadrature signal are two sine functions transmitted with a 90 degree phase offset, thus the name.

In PSK all symbols have the same amplitude and are distributed across the unit circle at equal phase intervals. Figure 2.1a depicts the constellation diagram for a 8-PSK modulation. Each symbol represents three bits and the symbols are 45° apart.

In QAM each symbol has a particular phase and amplitude associated to it. The symbols are equidistantly arranged in a grid. Figure 2.1b shows a 16-QAM constellation diagram.

2.2 Orthogonal Frequency Division Multiplexing

Orthogonal Frequency Division Multiplexing (OFDM) is a form of modulation where the data is encoded on mutually orthogonal subcarriers that are transmitted simultaneously [12]. The received pulse is then multiplied by the conjugate of the subcarriers to recover the data in a process that can be thought of as a demodulation. Since the subcarriers are orthogonal to each other, the data recovered during demodulation will be unique to each subcarrier.

2.2.1 The general OFDM pulse

The general OFDM pulse $x(n)$ can be expressed as:

$$x(n) = \sum_{k=0}^{N-1} a_k e^{j2\pi kn/N} \tag{2.1}$$

where N is the number of subcarriers and a_k represents the data to be modulated at the k-th subcarrier. Note that the OFDM pulse is periodic with period N. Moreover, the expression of the OFDM pulse is equivalent to the Inverse Discrete Fourier Transform (IDFT).

To extract the data from the OFDM we multiply the pulse by each of the subcarriers:

$$a_k = \sum_{n=0}^{N-1} x(n) e^{-j2\pi kn/N} \tag{2.2}$$

Note that the demodulation of the OFDM is equivalent to the Discrete Fourier Transform (DFT).

Another interpretation for the perfect recovery of the data is that the DFT is the inverse operation to the IDFT.

2.2.2 Cyclic prefix

In practical applications, a Cyclic Prefix (CP) is added to the beginning of each OFDM symbol to prevent Inter-Symbol Interference (ISI). Every time a new symbol

is sent the channel undergoes a transient period before reaching steady-state. In practise, this means that the first samples of the received OFDM symbol will contain a mix of whatever was in the channel before the symbol was transmitted and the transmitted symbol. To avoid this problem the CP is chosen such that its length is bigger than the transient response of the modelled channel. The first samples that contain the CP can be removed from the received signal and the data in the symbol decoded.

The CP consists of the last n samples of the pulse and it is generally placed at the beginning of the pulse although its actual location is irrelevant in digital communications. From a mathematical point of view, adding the CP makes the convolution of the channel and the transmitted pulse over the original duration of the pulse appear as a circular convolution. Besides the aforementioned removal of ISI the CP also reduces the cost of channel equalization.

2.2.3 OFDM Transmitter

The OFDM transmitter is used to map each data symbol to its corresponding subcarrier. In digital communications, each symbol represents a number of bits depending on the digital modulation scheme as seen in Section 2.1. Figure 2.2 shows a typical OFDM digital transmitter.

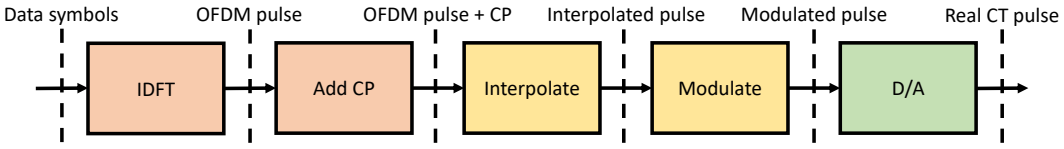


Figure 2.2: Digital OFDM digital transmitter.

The first step in the OFDM transmitter is mapping the data symbols to the subcarriers and combining the modulated data symbols to create an OFDM pulse using an IDFT. Next, the CP is added and the OFDM symbol is modulated with some complex carrier. In systems where the modulator is digital, an interpolation block is generally necessary between the CP block and the modulator to match the sampling frequencies of the OFDM pulse and the complex carrier. Finally, the OFDM pulse is converted to an analog signal in the D/A converter. If the D/A converter has only one channel, the quadrature part of the OFDM pulse needs to be discarded.

2.2.4 OFDM Receiver

The OFDM receiver is used to extract each data symbol from the subcarriers in the received OFDM pulse. Figure 2.3 shows a simple OFDM digital receiver.

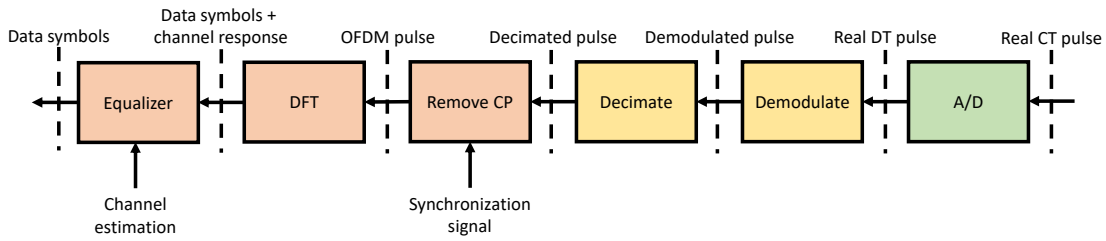


Figure 2.3: Digital OFDM digital receiver.

The OFDM pulse is first sampled using an ADC. For single channel ADCs, the real output is demodulated using the complex conjugate of the carrier frequency and decimated to save resources. The CP of the demodulated signal is then removed and the data symbols are extracted using a DFT. Generally, the receiver has no information about the start of the OFDM pulse so an additional synchronization step is generally needed before it is possible to start demodulating the symbols. After the symbols have been demodulated, the channel response and the data are separated using an equalization. The equalization consist in dividing the demodulated symbols by the inverse of the channel estimation with the objective of cancelling the channel contribution.

2.2.5 Equalization in the OFDM receiver

Equalization is a key operation in OFDM receivers as it removes the contribution of the channel to the received pulse leaving the data symbols and the noise at the output of the receiver. In general, equalization is performed using division:

$$X(w) + N(w) = \frac{Y(w)}{\hat{H}(w)} \tag{2.3}$$

where $X(w)$ are the data subcarriers, $N(w)$ is the noise spectrum, $Y(w)$ is the frequency domain received pulse and $\hat{H}(w)$ is the channel estimation.

The channel estimation can come from a number of sources. Generally, it is obtained using a set of pilot tones distributed across the OFDM symbol.

2.3 FMCW OFDM radar

In ISAC systems, the most common type of radar is the Frequency Modulated Continuous Wave (FMCW) radar with OFDM pulses [8]. This type of FMCW OFDM radar can be implemented using two architectures: the matched filter radar and the channel estimation radar. Both architectures are coherent meaning they can be used to measure target speed as well as target range.

2.3.1 Radar equations

Regardless of the architecture, all radars measure range using the delay between the transmitted pulse and the received pulse:

$$R = \frac{c \cdot t_{del}}{2} \tag{2.4}$$

where R is the range, c is the speed of light in air and t_{del} is the delay between the transmitted signal and the received echo.

In coherent radars, speed is measured using the non-relativistic Doppler effect:

$$f_D = \frac{2 \cdot v_{range}}{\lambda} \tag{2.5}$$

where f_D is the Doppler frequency, v_{range} is the range component of the speed of the target (The speed at which the target moves away from the radar antenna) and λ is the wavelength of the carrier frequency.

2.3.2 Matched filter radar

The coherent matched filter radar uses a matched filter to calculate the delay between the transmitted and received pulses.

A matched filter can be calculated as the time reversed conjugate of the transmitted pulse and it is the optimal solution for a detector in an Additive White Gaussian Noise (AWGN) channel. A detailed analysis of matched filter radars can be found in [13].

2.3.3 Channel estimation radar

Channel estimation radars are the most common architecture in ISAC literature [14]. This architecture removes the known transmitted data symbols from the received OFDM pulse using a process equivalent to the equalization in the OFDM receiver called system identification to obtain the frequency response of the channel. The time domain channel is obtained using an IDFT, delay terms in the channel correspond to reflections of the signal from targets. A detailed mathematical analysis of this architecture can be found in Appendix A.

For constant magnitude digital constellations, the channel estimation radar can be also interpreted as a matched filter in the frequency domain. Thus, in terms of performance, the two implementations can be considered equivalent under this condition.

2.3.4 Range-Doppler map

Radar speed measurements are only possible when multiple consecutive pulses are evaluated together as the phase due to Doppler shift remains virtually constant for the duration of the pulse. The general approach to calculating speed in FMCW

radars consists of creating a Range-Doppler matrix where the consecutive pulses are arranged in rows [15]. Conventionally, the time in the row dimension is called fast time because each entry is separated by a sampling period. The time across the column dimension is called slow time because each entry is separated by a pulse period.

Each consecutive pulse will have a different phase shift due to the Doppler effect and when combined, a tone corresponding to the Doppler frequency of the moving target will appear. These Doppler tones can be measured using a DFT across the columns of the Range-Doppler matrix to obtain a Range-Doppler map as seen in Figure 2.4.

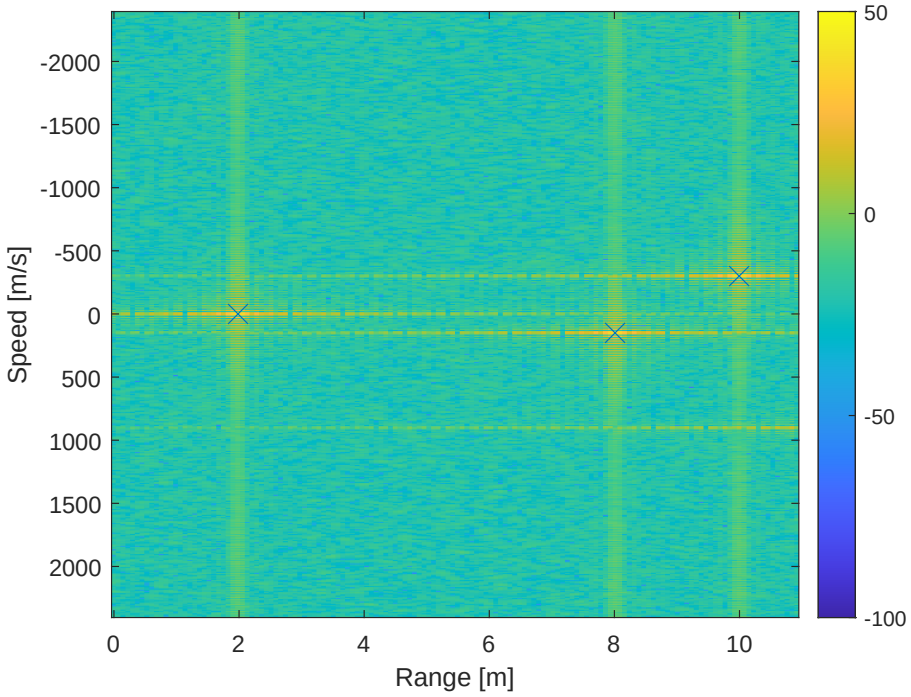


Figure 2.4: Range-Doppler map example.

2.4 5G NR

The 5G New Radio (NR) standard is a mobile network standard developed by the 3GPP project [16]. As its name implies, 5G represent the 5th generation of mobile networks. This standard utilizes the same OFDM pulse as LTE and builds upon its predecessor by increasing the flexibility of resource allocation and introducing higher bandwidth bands at higher frequencies [17].

The 6G standard is still in its infancy but work from the Hexa-X project suggests that many of the features of 5G will be inherited. Of particular relevance for this project, the OFDM pulse will still be part of the 6G standard [1]. The rest of the 5G concepts explained in this section will serve to justify decisions later on in the report as it is reasonable to assume the 6G will not be a radical departure from 5G in the aspect explained bellow.

2.4.1 Numerology and subcarrier spacing

Numerology is the term used to refer to the subcarrier spacing in 5G. For backwards compatibility reasons, 5G supports 15 kHz subcarrier spacing as its base subcarrier spacing, i.e. zero numerology. Higher numerologies indicate higher subcarrier spacing following:

$$\Delta f_{sc} = 15 \cdot 2^{\mu} \tag{2.6}$$

where Δf_{sc} is the subcarrier spacing in kiloHertz and μ is the numerology. Different subcarrier spacing means different symbol length so the frame structure in 5G becomes more complicated than in 4G.

2.4.2 Frame structure

The biggest transmission block in 5G is the radio frame. A radio frame lasts 10 ms and can contain a variable number of symbols. The radio frame is divided into slots each of which contains 14 symbols when using regular prefix and 12 symbols when using extended prefix. The number of slots in a frame and, therefore, their duration depends on the numerology following:

$$N_{slots} = 10 \cdot 2^{\mu} \tag{2.7}$$

where N_{slots} is the number of slots in a frame.

2.4.3 Timing parameters

Timing specifications in the 5G standard can be simplified by defining two basic timing units, T_c and T_s, and a ratio between the two κ:

$$T_c = \frac{1}{\Delta f_{max} \cdot N_f} = \frac{1}{480\text{kHz} \cdot 4096} = 0.509\text{ns}. \tag{2.8}$$

where Δf_{max} is the maximum carrier of 480 kHz spacing and N_f is the maximum Fast Fourier Transform (FFT) length. Note that a greater carrier spacing of 960 kHz is available but the maximum FFT length allowed in this configuration is 2048 so the timing unit remains unchanged.

$$T_s = \frac{1}{\Delta f_{ref} \cdot N_{f,ref}} = \frac{1}{15\text{kHz} \cdot 2048} = 32.552\text{ns}. \tag{2.9}$$

where Δf_{ref} is the LTE carrier spacing and $N_{f,ref}$ is the maximum FFT length in LTE.

$$\kappa = \frac{T_s}{T_c} = 64 \tag{2.10}$$

Using the two timing units and their ratio it is possible to define the length of a symbol as:

$$t_{symb} = N_{symb} \cdot T_c = (N_u + N^{\mu}_{CP,\bar{l}}) \cdot T_c \tag{2.11}$$

where N_u is the length of the symbol at the output of the OFDM modulation and $N_{CP,\bar{l}}$ is the normal length of the cyclic prefix. These two parameters are defined as:

$$N_u = 2048 \cdot \kappa \cdot 2^{-\mu} \tag{2.12}$$

$$N^{\mu}_{CP,l} = \begin{cases} 512 \cdot \kappa \cdot 2^{-\mu} & \text{extended cyclic prefix} \\ 144 \cdot \kappa \cdot 2^{-\mu} + 16 \cdot \kappa & \text{normal cyclic prefix } l = 0, l = 7 \cdot 2^{-\mu} \\ 144 \cdot \kappa \cdot 2^{-\mu} & \text{otherwise} \end{cases} \tag{2.13}$$

For the normal prefix, some symbols need to be longer than others to achieve the 10 ms frame length that is specified in the standard. Using the extended prefix such distinction is not necessary as the length of the symbols is a multiple of the frame length.

2.4.4 Resource blocks and frequency parameters

The 5G standard specifies two distinct frequency bands of operation, FR1 and FR2. Broadly speaking, FR1 is located in the centimeter region and FR2 in the millimeter region of the frequency spectrum. The particular frequencies used in each band depend on the country and are not relevant for our discussion. In this project we will focus on the FR2 band because it offers the most potential for radar sensing [1].

In the frequency domain, each subcarrier represents a resource element (RE) for the duration of the symbol. A resource block (RB) is made up of 12 REs and it is the smallest unit of data that can be transmitted in the 5G standard. The number of RBs per symbol is determined by the subcarrier spacing (numerology) and the bandwidth of the signal. Each bandwidth also includes provisions for some guard subcarriers. Table 2.1 represents the number of RBs in a symbol in the FR2 band.

μ	Δf	50 MHz	100 MHz	200 MHz	400 MHz	800 MHz	1.6 GHz	2 GHz
2	60	66	132	264	-	-	-	-
3	120	32	66	132	264	-	-	-
5	480	-	-	-	66	124	248	-
6	960	-	-	-	33	62	124	148

Table 2.1: Resource blocks in the FR2 band.

2.4.5 The 5G OFDM pulse

Finally, we can address the OFDM pulse structure. The following equation shows the time-domain specifications for the 5G OFDM pulse:

$$x_{\bar{l}}^{\mu}(t) = \sum_{k=\lfloor N_{RB}^{\mu} N_{sc}^{RB}/2 \rfloor}^{\lceil N_{RB}^{\mu} N_{sc}^{RB}/2 \rceil} a_{(k,l)}^{\mu} \exp(j2\pi k \Delta f (t - N_{CP,\bar{l}} T_c)) \tag{2.14}$$

where $x_{\bar{l}}^{\mu}(t)$ is the OFDM symbol at time t (including the CP), N_{RB}^{μ} denotes the resource blocks per symbol as specified in Table 2.1, N_{sc}^{RB} denotes the subcarriers in a resource block (12 subcarriers) and $a_{(k,l)}^{\mu}$ is the subcarrier modulation.

The 5G standard has a substantial beamforming component that is not being addressed in this project and that adds extra complexity to the OFDM waveform.

Note that, contrary to LTE and other communication standards the zero subcarrier is used to transmit information.

2.5 The ZCU111 Development Board

The ZCU111 [9] board is a development board for the ZCU28DR RFSoC. The ZCU28DR RFSoC can be divided into three subsystems: the Processing System (PS), the Programable Logic (PL) and the RF data converters. These three subsystems are enhanced by a number of hardened blocks designed for SDR applications.

The PS is composed of several processing units that can be used together with the PL to implement SDR systems.

The PL is, fundamentally, an FPGA optimized for SDR applications with purpose-built interfaces for the RF Data converters and the PS.

The RF data converters consist of eight 14 bit 6.554 GSPS DACs and eight 12 bit 4.094 GSPS ADCs. The ADCs are arranged in pairs forming a tile. Within each tile, both ADCs share the same clock source. The DACs are similarly arranged in tiles containing four DACs sharing the same clock.

2.6 IP blocks

IP blocks are a way to reduce resource usage and add features to the design that would have been very complex to implement using Hardware description Languages (HDLs) alone. The IP blocks used in this project are the FFT core, the RF data converter block, the ILA and the VIO. The FFT core and the RF data converter use the AXI-Stream interface for the data I/O.

2.6.1 The AXI-Stream interface

The AXI-Stream or AXI-S interface [18] is a synchronous communication interface used by many of the IP blocks developed by Xilinx. The simplest form of AXI-S

consists of a clock wire, two signal wires and a data bus. The two signal wires, READY and VALID, are used to perform a handshake operation between the controller and the peripheral. The data bus is used to send data from the controller to the peripheral and its width is a multiple of eight. The data bus uses little endian format where the zeroeth bit (first bit to be sent) in the bus corresponds to the zeroeth bit in the data.

Each data frame sent on AXI-S requires a handshake. The handshake occurs when the VALID and READY signal are high at the rising edge of the clock. The controller drives the VALID signal high and writes to the bus whenever it wants to send a new frame. The peripheral drives the READY signal high when it is ready to receive data. The controller checks for a successful handshake before sending a new data frame. Figure 2.5 depicts AXI-S transactions in a timing diagram.

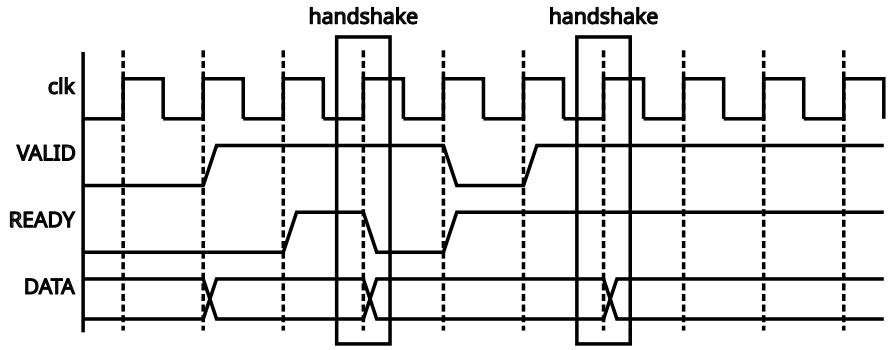


Figure 2.5: Timing diagram of AXI-S transactions.

2.6.2 The FFT core IP

The FFT core [19] is an IP block that computes FFTs on streams of samples. The input and output to the FFT core are AXI-S interfaces and the core can be configured using a separate AXI-S interface.

2.6.3 The RF data converter IP

The RF data converter IP block [20] is used as an interface between the PL and the RF tiles in the RFSoC. The RF data converters built in the RFSoC consist of the RF data converter itself and a DSP pipeline implemented on hardened silicon blocks. The Digital Signal Processing (DSP) pipeline contains all the necessary blocks to modulate and demodulate the baseband signal. The input and output of the block are AXI-S interfaces.

Different levels of synchronization are required in order to successfully utilize the RF data converters. In this project we used the simplest synchronization scheme available since we were only using a DAC for transmission and an ADC for reception. As previously explained, each tile has its own clock network. The clock for each tile is provided through a separate differential pin in the IP block. To synchronize multiple tiles together, an analog synchronization signal is used. Each tile has a

synchronization block that ensures the phase is constant across the design. Each tile also needs to be synchronized with the PL. A separate synchronization signal allows to synchronize the sampling clock to the clock used in the PL.

2.6.4 The ILA

The Integrated Logic Analyzer (ILA) [21] is an IP block that can be used to monitor signals inside a design. It includes diverse advanced features for triggering on user-specified events. In its simplest form, the ILA can be triggered manually through the Hardware Manager and the JTAG interface in Vivado to capture a user-defined number of system clocks. The captured data can be displayed in the waveform viewer and can be exported to a CSV file.

2.6.5 The VIO

The Virtual Input/Output (VIO) block [22] is used to monitor and drive signals inside a design in real time. The VIO connects to the Hardware Manager through the JTAG interface.

3

Methods

In this chapter, the methodology used in each of the stages of the project will be explained. First, the methods used to select the radar design will be reviewed. Next, the system-level simulation and testbench methodology will be analyzed. Finally, the HDL implementation, testing and optimization as well as the physical testbench design methodology will be explained.

3.1 Selecting the radar design

The first step in the project was to select the radar design that was going to be used. The radar system was based on a 5G compatible communication system. The communication system was designed following the OFDM transmitter and receiver designs specified in Section 2.2. Then, it was modified by removing unused blocks and adding the blocks needed for the radar operation. Two radar systems were implemented this way following the architectures described in Section 2.3: a coherent matched filter based radar and a channel estimation based radar. Finally, the designs were compared based on the considerations laid out in Chapter 1 and the most suitable design was chosen.

3.2 System-level simulation

The system-level simulation was programmed in MATLAB. Each of the versions of the simulation was compared to the expected theoretical result using plots of the OFDM pulse at different points of the simulation.

The system-level simulation was also used to generate stimuli vectors for testing the receiver and transmitter before the system testbench was ready.

3.3 The system-level HDL testbench

The system-level testbench was designed using a simulating framework developed at Ericsson Research that used Verilator and MEX to make verification of DSP blocks faster and more convenient. Verilator is an Open-Source simulator that compiles HDL code into highly optimized C++ models that can then be simulated using a C++ wrapper. MEX is a MATLAB API that allows MATLAB to access C++ functions. The combination of these two programs allowed for the creation of a

MATLAB function that could accurately represent the designed HDL block in a clock and bit-exact manner.

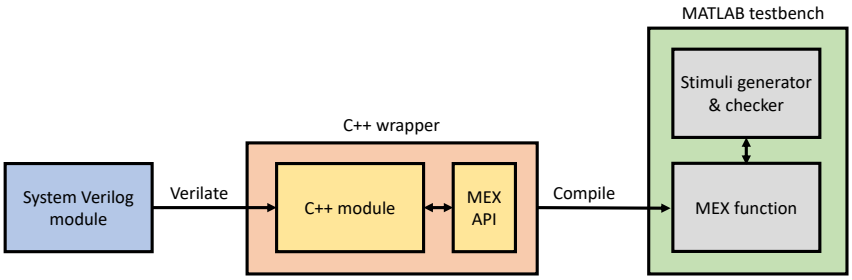


Figure 3.1: Testbench generation process.

The same process was followed to create all the MEX functions from HDL blocks. The first step was to verilate the HDL code into a C++ module. Next, the C++ module was instantiated in a wrapper together with the MEX API. The C++ wrapper determined the behaviour of the MEX function every time it was called in the MATLAB testbench. Finally, the C++ wrapper was compiled into a MEX function that could be used inside the MATLAB testbench. Figure 3.1 illustrates the process.

All of the MEX functions created from HDL blocks had the same behavior when called in the MATLAB testbench. First, the inputs provided to the MEX function were read and fed into the DUT. Next, the output of the DUT was evaluated. If the DUT was combinational, the output of the DUT was formatted and returned as the output of the MEX function. If the DUT was sequential, a full clock cycle was simulated. At the end of the clock cycle, the output of the DUT was formatted and returned as the output of the MEX function.

3.4 Radar HDL implementation

Each of the radar blocks was implemented in System Verilog (SV) using Vivado as the IDE and then combined in a system wrapper to create the radar. For every block, a Vivado project was created including a design file and a simple tesbench. After programming the modules, they were elaborated to check for syntax errors. Then, a pre-synthesis behavioral simulation was performed using the SV testbench. The test vectors were generated randomly on the testbench and the results were checked using the Waveform Monitor in Vivado.

The FFT module required a different methodology as it contained instances of other modules. The modules making up the FFT were tested individually using the method explained in the previous paragraph. The FFT module was tested using test vectors and golden vectors generated with a MATLAB script. The MATLAB script generated a file with random test vectors and the corresponding golden vectors and the testbench stimulated the DUT and checked the output.

3.5 Radar HDL testing

The HDL implementation was tested again using MATLAB testbenches. The second set of tests was more comprehensive thanks to their increased execution speed. Each module was first converted to a MEX function and then a MATLAB testbench was built around the MEX function to check the functionality. Finally, the different testbenches were combined to create a system-level testbench whose output could be compared to that of the system-level simulation.

The first step in the process to convert an SV module to a MEX function was "verilating" the module or compiling the module into C++ using Verilator. Next, a C++ wrapper was coded to allow the MEX API to interact with the C++ module. Finally, the wrapper was compiled into a MEX file that could be accessed by the MATLAB testbench. The process was automated using TCL scipts.

3.6 Radar HDL optimisation

The optimized radar HDL blocks were implemented following the same flow described in Section 3.4. However, instead of using a system wrapper, the radar system was implemented as a Block Diagram (BD) project. The BD tool in Vivado allowed for easier modification of the IP blocks.

Two testbenches were created to test the radar on the RFSoC. The first testbench contained the tranmitter while the second testbench contained the full radar.

The first testbench was used to test the configuration of the RF tiles. Two ILAs were used to evaluate the performance of the first testbench, one at the input to the DAC RF tile and another at the output of ADC RF tile. The output of the ILAs was saved to a CSV file and then plotted on MATLAB using a script.

The second testbench expanded upon the first testbench by adding the radar receiver and another ILA at the output of the receiver. The same MATLAB script was used to plot the output of the ILA.

3.7 RFSoC testbench

The radar BD was synthesised and implemented using a TCL script that configured the target board and the synthesis and implementation parameters. At the end of the synthesis and implementation, a hardware configuration file was generated. The hardware configuration file together with some initialization scripts for the PS were loaded into Vitis and used to program the RFSoC

The RFSoC testbench was monitored using the Hardware Manager. The ILAs were configured for a manual trigger and the length of the recording was set to capture two full symbols at the output and the input of the radar. The output of the ILAs was dumped to a CSV file and plotted using a MATLAB script.

4

Design

In this chapter the different design stages of the project will be investigated. First, the proposed designs for the radar system will be analyzed. Next, the system-level simulation will be explained. The HDL implementation of the radar will follow starting with the initial design and finishing with the optimized design. Finally, the RFSoC testbench will be described.

4.1 The radar design

Two radar designs were compared for the project, one for each implementation described in Section 2.3: the matched filter radar and the channel estimation radar. Both designs used the OFDM transmitter described in Section 2.2.3 and parts of the OFDM receiver described in Section 2.2.4. Both radar designs used a Range-Doppler calculator block which can calculate the Range Doppler matrix of the output as described in Section 2.3.4.

Figure 4.1 shows the system using the matched filter architecture and Figure 4.2 shows the system using the channel estimation architecture. The blue blocks represent the OFDM transmitter, the orange blocks represent the OFDM receiver and the yellow blocks are the radar.

4.1.1 Matched filter radar

The matched filter radar reused the least amount of blocks from the OFDM receiver. A matched filter block was connected to the output of the decimator and to the output of the add CP block in the transmitter. The output of the add CP block was used to calculate the matched filter which was then convolved with the output of the decimator. The result could then be used to calculate the Range-Doppler matrix.

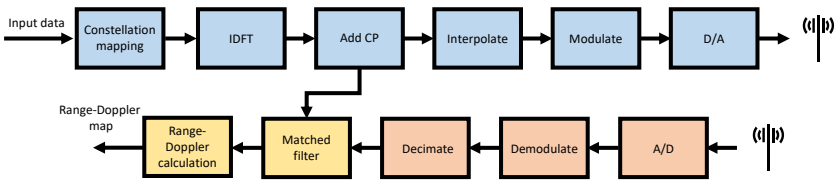


Figure 4.1: Matched filter radar block diagram.

4.1.2 Channel estimation radar

The channel estimator radar reused all of the blocks in the OFDM receiver. The channel estimation block could be implemented using the equalizer block in the OFDM receiver by substituting the frequency domain channel estimation by the data subcarriers as seen in Section 2.3.3. The time domain channel estimation could be calculated using a IDFT and then used to calculate the Range-Doppler matrix.

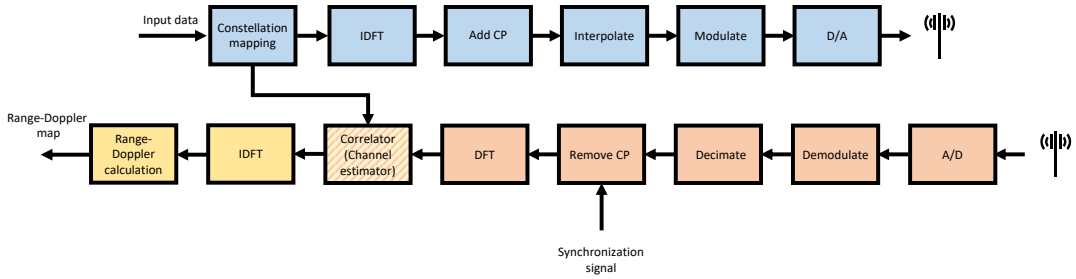


Figure 4.2: Channel estimation radar block diagram.

4.1.3 Selecting a design

Both designs offered the same performance for a QPSK constellation shown in Section 2.3.3. The matched filter design offered higher range as the length of the input to the radar was not constrained to the size of an OFDM symbol. On the other hand, the channel estimation radar offered a lower latency as the frequency domain convolution is generally faster than its time domain equivalent for high tap counts. Moreover, the channel estimation radar had the added advantage that it produced a precise channel estimation that can be used for the communications system.

For this project, latency was considered a more important parameter than range. In the proposed applications of the radar [1], the range was expected to be short while the ability to quickly detect dangerous situations was emphasized. Thus, the channel estimation radar was selected.

4.2 Designing system-level simulation for the channel estimation radar

Once the radar design was chosen, a system-level simulation was designed to verify the design. Each block in Figure 4.2 was implemented using independent MATLAB functions such that each block could be tested individually. The block architecture also allowed for the creation of a system-level testbench in which the MATLAB blocks were substituted for equivalent RTL blocks. Figure 4.3 shows a block diagram of the simulation.

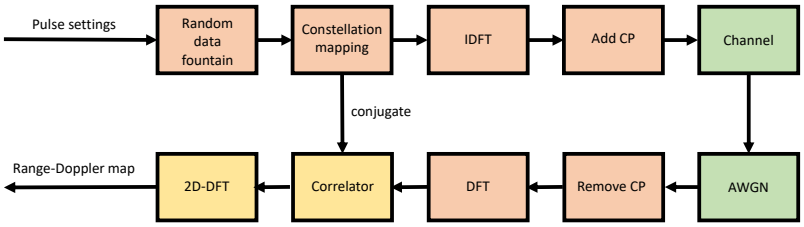


Figure 4.3: MATLAB simulation block diagram.

4.2.1 Simulated parameters

The simulation was designed to meet the 5G NR standard in the FR2 band. A comprehensive support of the 5G NR standard was useful to understand how the different parameters affected the SNR of the output. This understanding was used at the HDL implementation stage to simplify the design. The following 5G NR parameters were supported:

- Numerology: 2,3,5 and 6.
- Subcarrier spacing (derived from numerology): 60 kHz, 120 kHz, 480 kHz and 960 kHz.
- Symbol Badwidth: 50 MHz, 100 MHz, 200 MHz, 400 MHz, 800 MHz and 1600 MHz.
- Resource blocks: See Table 2.1.
- Prefix type: regular or extended.

4.2.2 Iterations

The system-level simulation underwent four major iterations. The basic functionality of the radar was implemented in the first iteration. In the second iteration, the channel options were expanded, a 5G compliant pulse was coded and the ability to generate Range-Doppler maps was added. In the third iteration, all the options for OFDM pulses in the FR2 band were included. Finally, in the fourth iteration, the simulation was simplified based on the results from the previous version to include only the relevant parameters for the HDL testbench.

The first iteration of the system-level simulation consisted on the radar processing blocks, an arbitrarily modulated OFDM pulse with 1024 subcarriers and a CP length of 256 samples and a single simulated target. The first step in the design of the first system-level simulation was to determine the MATLAB functions that would be performing each of the processing steps in the radar transmitter and receiver. Next, each block was tested individually to ensure that the particular processing step was performed successfully. Once the MATLAB blocks were verified, they were combined to form the transmitter and the receiver and tested using the OFDM pulse and the simple channel.

The second iteration of the system-level simulation added more channel options. First, the option to simulate multiple targets was included and tested. Then,

Doppler effect was modelled for each target. Also, the option to generate Range-Doppler maps was included. The Range-Doppler maps allowed testing of moving targets. Finally, AWGN was added.

The third iteration implemented the option to select between a number of OFDM pulse specification of the FR2 band. The frequency related specifications were coded first as they required the least modification to the simulation. Next, the time domain specifications, namely the normal and extended cyclic prefix options, were added. The normal prefix option required extensive modification of the simulation as the symbol length was not constant inside a frame.

The fourth iteration was used as a basis for the HDL testbench. The implementation set some limitations to the parameters that could be realistically implemented. Thus, a simplified simulation was possible.

4.2.3 Generating test vectors

The system-level simulation required two sets of test vectors, a data input and a channel.

The data input test vector was a stream of random bits. Each bit was generated independently using a uniform random distribution. In real communication applications, data does not necessarily follow a uniform distribution. However, for radar applications, this data model was advantageous because it resulted in low correlation between different pulses which could have lead to false positives.

The channel test vector was designed to model the presence of multiple targets moving at different speeds. Optionally, AWGN could be added to the output of the channel to degrade the input signal to the radar. A more complex modeling of the channel was possible but it was not implemented as it was beyond the scope of this project. The channel test vector was implemented as an FIR filter of the same length as the CP of the OFDM symbol because the maximum range of the radar is given by the length of the CP.

4.2.4 Emulating moving targets

Emulating moving targets was necessary to fully showcase the Range-Doppler map. For simplicity, the Doppler effect was only modelled for the carrier frequency as seen in Equation A.10. Each target was suitably delayed and then multiplied by the Doppler tone corresponding to the speed of the target.

4.3 Implementing the channel estimation radar on HDL

The channel estimation radar was implemented using System Verilog. The degree of parameterization required to fulfil all the specifications of the 5G standard was deemed beyond the scope of this project. A subset of the 5G specifications were chosen that could be reasonably implemented and that were compatible with the

available hardware. Each block in Figure 4.2 was implemented separately and a complete design was created in a hierarchical structure. Some glue logic and pipeline registers were added to control the data flow and make the design meet timing constraints respectively.

4.3.1 Constraining the design

A subset of specifications was chosen that provided a good compromise between performance, resource usage and implementation complexity:

- Extended cyclic prefix. The extended cyclic prefix allowed for uniform CP length in all the symbols and longer range, thus reducing complexity.
- Symbol bandwidth: 50 MHz to 100 MHz. The symbol bandwidth was determined by the clock frequency of the system.
- FFT Points: 1024. The number of FFT points was chosen as the minimum number that could compute all symbols bellow 100 MHz.

The remaining specifications were constrained by the previous three.

4.3.2 Fast Fourier Transfrom (FFT)

The initial implementation of the FFT was a direct translation of the FFT algorithm into RTL. This parallel structure was considered optimal for a low latency application. The main disadvantage of this implementation was the high resource usage and the need to input all samples simultaneously. Later in the project, other options were found that offered sufficient latency for much lower resource usage.

The complex multipliers were implemented using the Gauss algorithm for complex multiplication which uses three real multiplications instead of four, thus, reducing the number of DSP slices needed from four to three.

To overcome the natural growth of the data samples as they were operated on, after each butterfly stage, the result was shifted by one bit to the right to divide it by two. This scaling schedule guaranteed that no overflow would be present at the output for a constant width system.

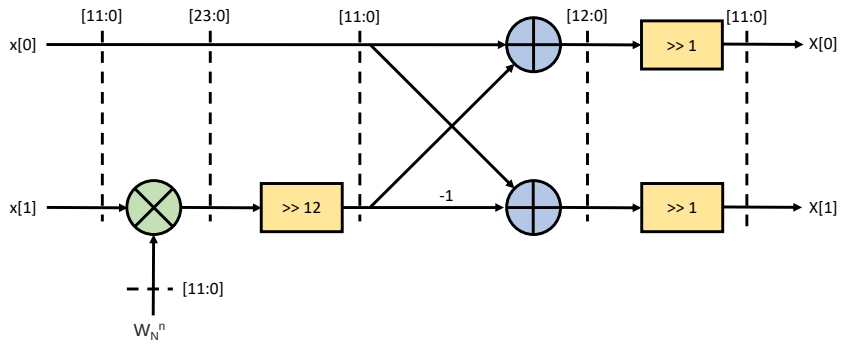


Figure 4.4: Butterfly module.

The recursive nature of the FFT was leveraged to simplify the HDL description. Using a single butterfly structure it was possible to generate the complete design. Figure 4.4 depicts a detailed butterfly module.

The twiddle factors (W_N^n) were stored in a ROM that was connected to the different butterflies in the top module. The ROM values were calculated at synthesis using the built in cosine and sine functions in SV. Since the only difference between the FFT and the IFFT are the twiddle factors, changing the definition of the ROM yielded the IFFT without further modifications.

4.3.3 The transmitter

The transmitter consisted of a constellation mapping block implemented as a LUT, an IFFT and a block to add the CP. Since the IFFT block required all samples to be provided simultaneously, a serial to parallel (SIPO) register was needed at the input and a parallel to serial (PISO) register was needed at the output. The CP block and the PISO register were combined for convenience. Figure 4.5 shows the transmitter implementation.

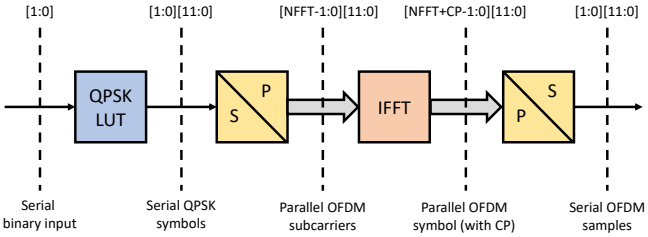


Figure 4.5: Radar transmitter.

The input data to the transmitter was a bitstream sourced from a ROM memory and the output consisted of complex samples that could then be fed into the RF tiles in the RFSoC.

4.3.4 The receiver

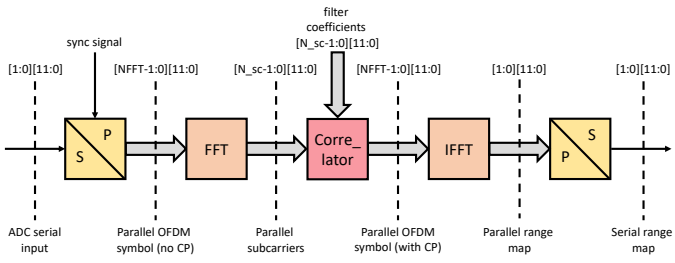


Figure 4.6: Radar receiver.

The receiver consisted of a block to remove the CP, a FFT block, a correlator (implemented using the same complex multipliers used for the FFT block) and a

IFFT block. A SIPO register was used at the input of the FFT to parallelize the samples as well as to remove the CP. The correlator was implemented as a parallel structure at the output of the FFT. After the IFFT, a POSI register was added to create a serial stream of signed complex numbers representing a range map. Figure 4.6 shows the receiver implementation.

It was decided that the Doppler map would be calculated during post-processing as the timing requirements were less strict. Thus, it was possible to reduce the complexity of the receiver with a 2D-FFT and a BRAM.

4.3.5 Synchronizing the transmitter and the receiver

Synchronization between the transmitter and receiver was fundamental to ensure correct range measurements. In order to guarantee correct synchronization, three slow clocks were generated from the system clock. Each slow clock had a high time of one system clock period and remained low for the rest of the slow clock period, thus, it could be used as enable signals to time the sequential blocks in the transmitter and receiver. The three slow clocks were:

- The sample clock was used to synchronize the ADC and DAC sampling as well as the serial sections of the design.
- The transmitter symbol clock was a multiple of the sample clock and it spanned the length of an OFDM symbol. It was used to time the IFFT block in the transmitter.
- The receiver symbol clock had the same period as the transmitter symbol clock but it was delayed by some empirically phase to remove delays due to the transmitter and receiver in the RFSoC. This clock was used to indicate the beginning of a new symbol to the receiver.

Figure 4.7 shows an example timing diagram where the three slow clocks are depicted together with the system clock. Note that the frequency of the clocks is arbitrary.

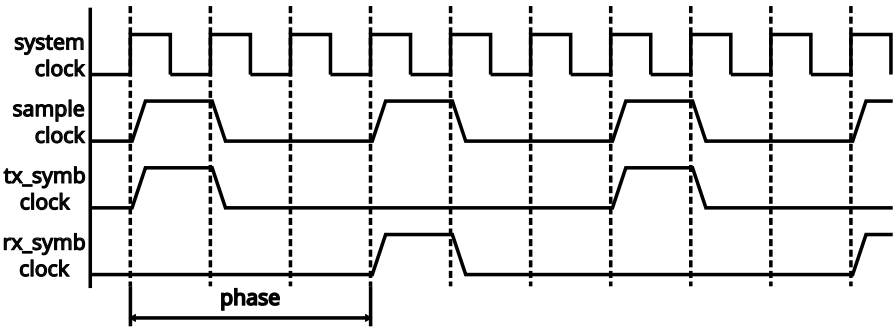


Figure 4.7: Radar clock timing diagram.

4.4 Optimizing the channel estimation radar implementation

After testing the initial radar implementation, it was observed that the design did not completely fulfill the requirements and that some improvements could be made to the overall implementation. It was decided that IP blocks would be used to reduce the resource usage and the design was modified into a BD project to facilitate IP integration.

4.4.1 Problems with the implementation

The two main problems found in the original implementation were excessive DSP slice usage and difficulty interfacing the system to the RF data converters.

The FFT implementation used in the original design required $N/2 \log_2 N$ complex multipliers per block and the correlator required N complex multipliers (where N was the length of the FFT). Each complex multiplier used three DSP slices bringing the total DSP slices needed to implement the transmitter alone, in theory, up to 11264 DSP slices for the 1024 point FFT. From the beginning it was known that this resource-intensive implementation would be problematic. However, in order to have a system-wide testbench in MATLAB, it was necessary to implement the FFT algorithm in HDL.

Interfacing the RF data converters with the radar proved challenging. Successful synchronization of the different converters required very specific control signals and interaction with the built-in micro-controller in the RFSoC. It was decided that modifying an existing project would allow to keep the scope of this project focused on the radar implementation while simplifying synchronization.

The IP blocks required AXI-S interfaces for data transfer. Thus, the radar blocks also needed to be modified to support the AXI-S communication protocol. Another modification motivated by the use of IP blocks was increasing the word length from 12 to 16 bits.

4.4.2 The transmitter

The transmitter design was substituted by a ROM with a suitable AXI-S interface. It was decided that a single OFDM pulse waveform was going to be used for the testbench to make it easier to characterize performance. The OFDM pulse was generated using a MATLAB script and it was then written to a memory file which was read during synthesis.

4.4.3 The receiver

The receiver resource usage was reduced by substituting the FFT blocks designed in Section 4.3.2 for FFT IP blocks. The correlator was substituted for a single complex multiplier and AXI-S interfaces were added to the I/O of all modules. Figure 4.8 shows the optimized receiver.

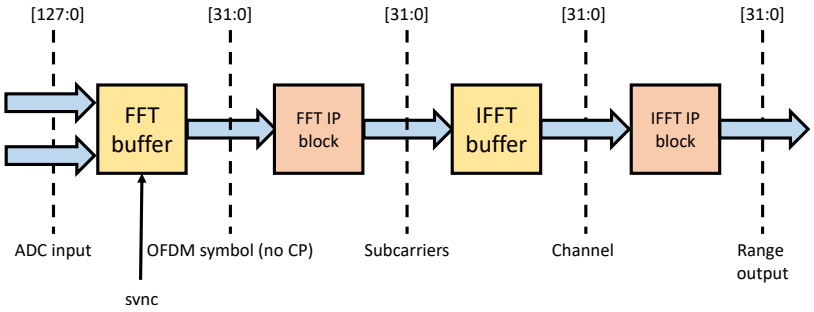


Figure 4.8: Optimized radar receiver. The connections between blocks are AXI-S buses and the widths specified above the dashed lines refer to the data bus width in the AXI-S interface.

Further modifications to the receiver design included the addition of two buffers, one at the input of the FFT and one at the input of the IFFT connecting the FFT to the IFFT. The first buffer, the FFT buffer, substituted the SIPO register in Figure 4.6 and also performed the CP removal. The IFFT buffer substituted the correlator, and the POSI was removed as it was no longer needed.

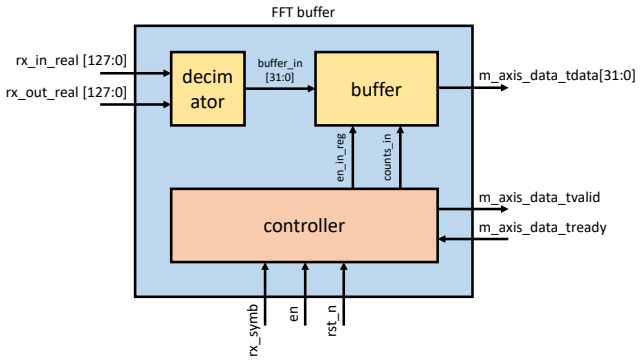


Figure 4.9: Block diagram of the FFT buffer.

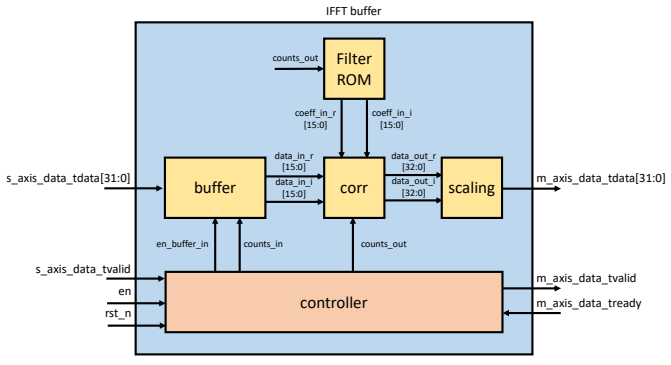
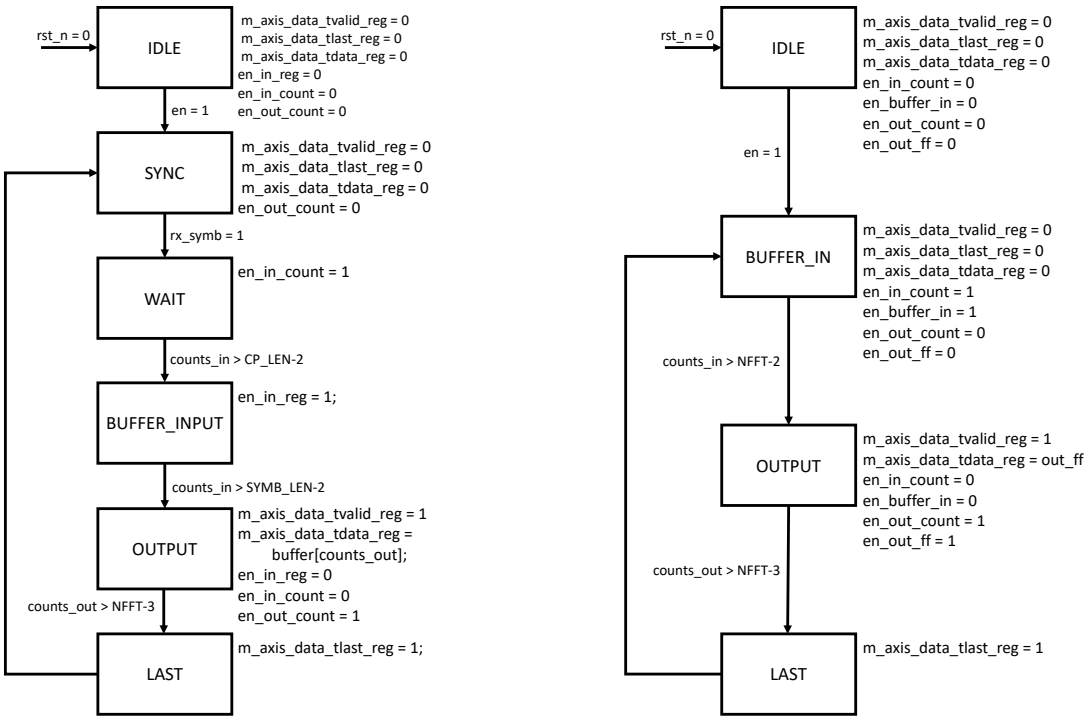


Figure 4.10: Block diagram of the IFFT buffer.

The buffers were needed in this new design to store data because the FFT IP block processed data in a batch way. The FFT buffer was controlled by the state machine

in Figure 4.11a and the block diagram can be seen in Figure 4.9. Likewise, the IFFT buffer was controlled by the state machine in Figure 4.11b and the block diagram can be seen in Figure 4.10.



(a) FFT buffer. **(b)** IFFT buffer.

Figure 4.11: FSM diagrams of the buffers in the receiver

Upon receiving a reset signal, the buffer goes into the IDLE state where all the modules get disabled. When enabled, the FFT buffer waits for a synchronization signal indicating a new OFDM symbol has been transmitted. Next, a counter is started to count the number of samples received by the buffer. The first samples in the input AXI-S bus correspond to the cyclic prefix and are discarded while the last samples are stored in the buffer. After the symbol has been read, the buffer writes the symbol to the AXI-S bus at the output.

The IFFT buffer has the same behavior to the FFT buffer but with a simpler FSM (Figure 4.11b) because it does not need to synchronize to the incoming symbols.

4.4.4 The controller

The controller was used to first configure the FFT IP blocks and then enable the radar components as specified in [19]. Figure 4.12 shows the FSM diagram of the controller.

Upon power up of the system, the controller moved into the IDLE state and, in the next clock cycle, to the READ state. In the READ state, the scaling schedule of the FFT and IFFT blocks was selected. Once selected, the load_scaling signal

was set HIGH to start the configuration. First, the FFT was configured in the CONFIG_FFT state. After detecting a successful handshake in the FFT, the IFFT was configured in CONFIG_IFFT. Next, the WAIT_CONFIG state waited for a clock cycle to ensure the configuration was successful. Finally, the transmitter and the buffers were enabled in the DONE state. To reconfigure the FFT blocks, the load_scaling signal had to be set LOW.

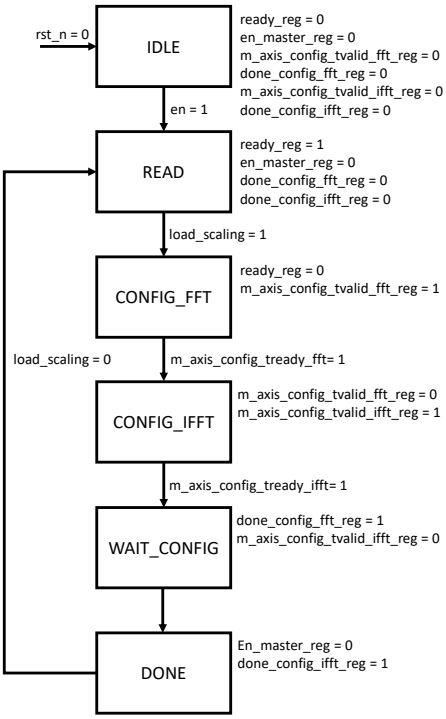


Figure 4.12: FSM diagram of the controller.

4.5 The RFSoC testbench

Figure 4.13 shows a complete block diagram of the RFSoC testebench.

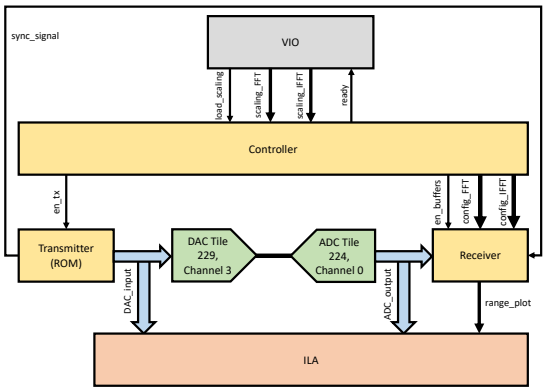


Figure 4.13: Block diagram of the RFSoC testbench.

4.5.1 Clock configuration

The development board clock was configured to 491.52 MHz, the ADC and DAC clocks were configured to 3932.16 MHz and the PL was clocked at 245.76 MHz. The DAC and ADC were configured for a x2 interpolation and decimation, respectively, such that the sampling frequency at their input or output was 1966.08 GHz. This was the sampling frequency for a standard 5G OFDM as seen in Equation 2.8.

4.5.2 RF tile configuration

The radar design required a DAC for the transmitter and an ADC for the receiver. Therefore, two RF tiles were needed, one with a DAC and another one with an ADC. The RF tiles for the receiver and the transmitter were synchronized to each other using an analog synchronization signal generated in the development board. The tiles were also synchronized to the PL using a synchronization signal generated in the template project.

The DAC tile was configured with an x2 interpolation factor so that the sampling frequency of the input to the DAC tile corresponded to the 5G sampling frequency. Since the PL frequency was 245.76 MHz [1], the AXI-S input to the DAC tile was configured with an eight sample wide data bus to meet the data throughput required by the DAC. Additionally, the mixer inside the DAC tile was enabled so that the I/Q input to the tile could be modulated with a 0.8 GHz carrier. Finally, the inverse sinc filter was also enabled to reduce the distortion from the Zeroeth Order Hold (ZOH) at the output of DAC.

The ADC tile was also configured with x2 decimation. Since the FFT IP block does not support parallel inputs, an extra decimation block was added at the output of RF tile to match the sample rate of the ADC output to the sample rate of the FFT (which is the sample rate of the PL). The extra decimation reduced the maximum bandwidth of the OFDM pulse down to 245.76 MHz, still within the specifications in Section 4.3.1. The mixer inside the ADC tile was enabled to demodulate the input to the ADC using a -0.8 GHz signal.

4.5.3 Transmitter configuration

The AXI-S bus in the transmitter ROM was configured with an eight I/Q sample wide data bus. Each sample was composed of an I and a Q component 16 bits wide. The width of the I/Q samples was determined by the DAC tile.

The OFDM pulse was chosen to meet the design specifications outlined in 4.3.1. The OFDM subcarrier spacing was chosen to be 120 kHz (numerology 3) and a 100 MHz pulse bandwidth was selected. This resulted in an OFDM pulse with 792 subcarriers as seen in Table 2.1. Each subcarrier was modulated with a random QPSK symbol following a uniform random distribution.

[1] The PL frequency was controlled by a script from the template project running in the PS

4.5.4 Receiver configuration

The FFT cores in the receiver were configured for 2048-point FFTs running from the 245.76 MHz clock and with a 50 MSPS throughput. The scaling schedule was configured during testing to maximize the range at the different stages of the receiver.

The filter ROM in the IFFT buffer was configured from a memory file that contained the conjugate of the transmitted OFDM pulse. The memory file was generated in the same MATLAB script as the OFDM pulse.

4.5.5 Controlling and monitoring the testbench

An ILA block was used at the input and output of the RF tiles to monitor the input to the DAC and the output of the ADC so that synchronization errors could be easily detected while plotting these signals. The same ILA block was used at the output of the transmitter to read the range plot output.

The VIO block was used to control the testbench. The VIO was used to change the scaling schedule of the FFT and IFFT blocks and trigger the reconfiguration as well as to check the state of the controller.

4.5.6 The RFSoC physical tesbench

The physical RFSoC testbench consisted of the ZCU111 development board connected to a SMA breakout board. The input and output corresponding to the channel 3 in the DAC tile 229 and the channel 0 in the ADC tile 224, respectively, were connected using coaxial cables of different lengths. Figure 4.14 shows an example of the setup.

A spectrum analyzer was used to check the output of the DAC and a RF signal generator was also available to generate test inputs to the ADC. The RF signal generator was not used as the ADC did not present any issues during the setup of the testbench.

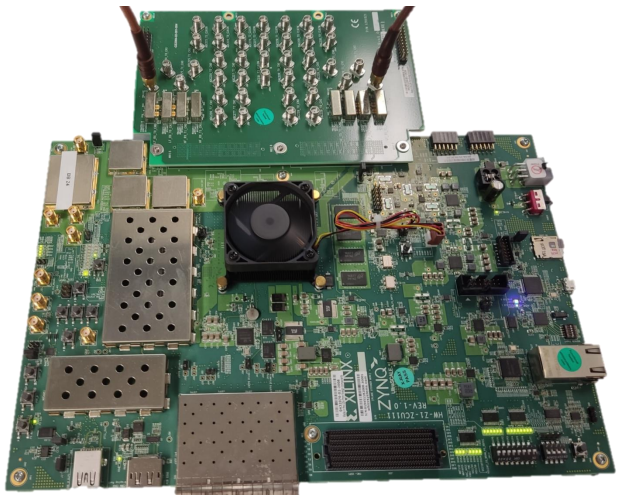


Figure 4.14: RFSoC physical testbench.

5

Results

In this chapter, the results from the project will be shown and discussed. First, the system-level simulation results will be addressed followed by the results from system-level HDL testbench. Next, a resource usage comparison between the initial HDL design and the optimized design will be performed and discussed. Finally, the physical testbench results will be shown. The first test only includes the transmitter while the second test includes the whole radar system.

5.1 System-level simulation results

As explained in Section 4.2, the system-level simulation simulation was designed to comply with the 5G standard in the FR2 band. In this section, one of the test runs used to verify the simulation will be analyzed.

The OFDM pulse parameters used for this simulation run were:

- Subcarrier spacing: 120 kHz (Numerology 3).
- Bandwidth: 100 MHz (792 data subcarriers).
- Extended prefix.
- Subcarrier modulation: QPSK.

Four moving targets were simulated as seen in Table 5.2.

Target	Distance [m]	Speed [m/s]
1	17	0
2	20	10
3	60	-30
4	35	90

Table 5.1: Targets used for the simulation.

AWGN was added to the received signal to obtain a 20dB SNR in the input to the receiver.

Figure 5.1 shows the magnitude and phase of the transmitted OFDM pulse. A random QPSK symbol was used to modulate each of the data subcarriers as explained in Section 4.2. Thus, the magnitude of the pulse in the data subcarriers region is

always one and the phase alternates randomly between $\pi/4$, $3\pi/4$, $-3\pi/4$ and $-\pi/4$ as it is expected from a QPSK constellation.

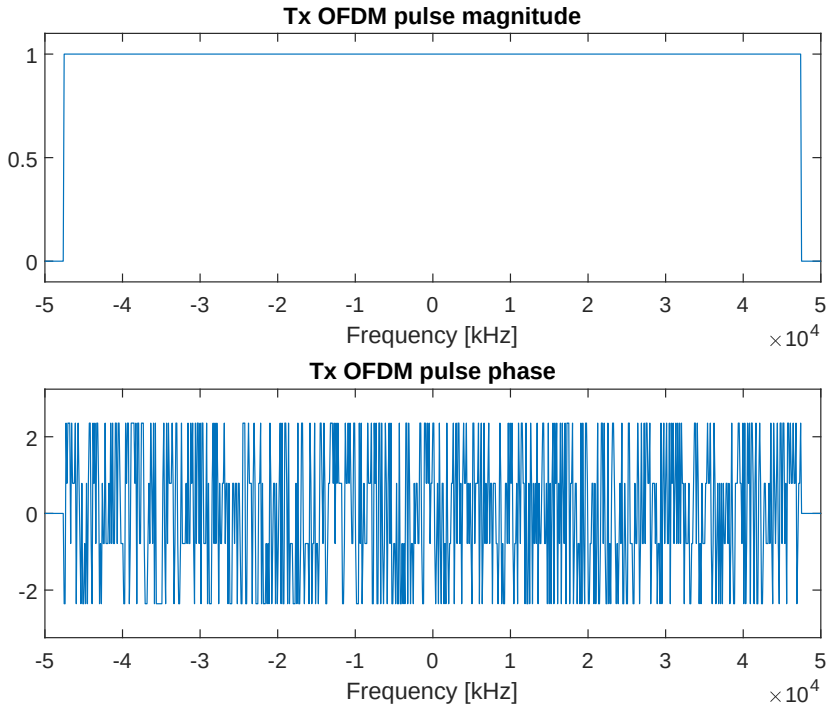


Figure 5.1: Frequency domain Tx OFDM pulse.

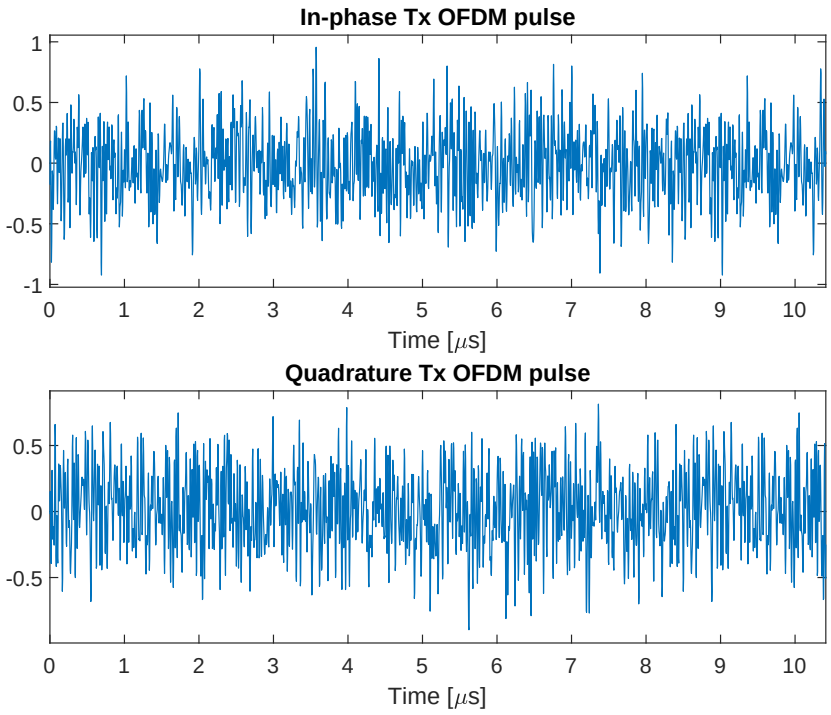


Figure 5.2: Time domain Tx OFDM pulse including CP.

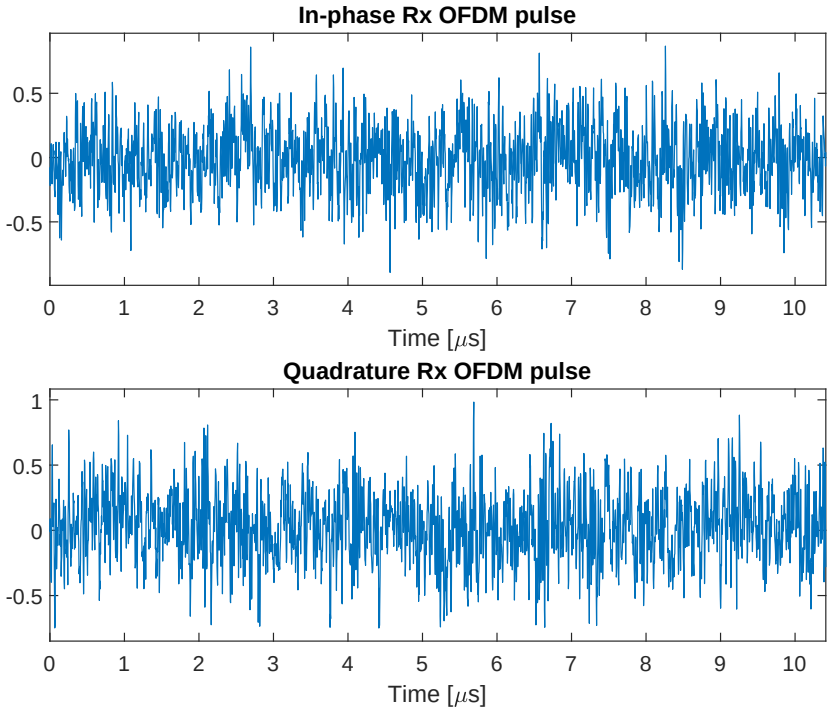


Figure 5.3: Time domain Rx OFDM pulse including CP.

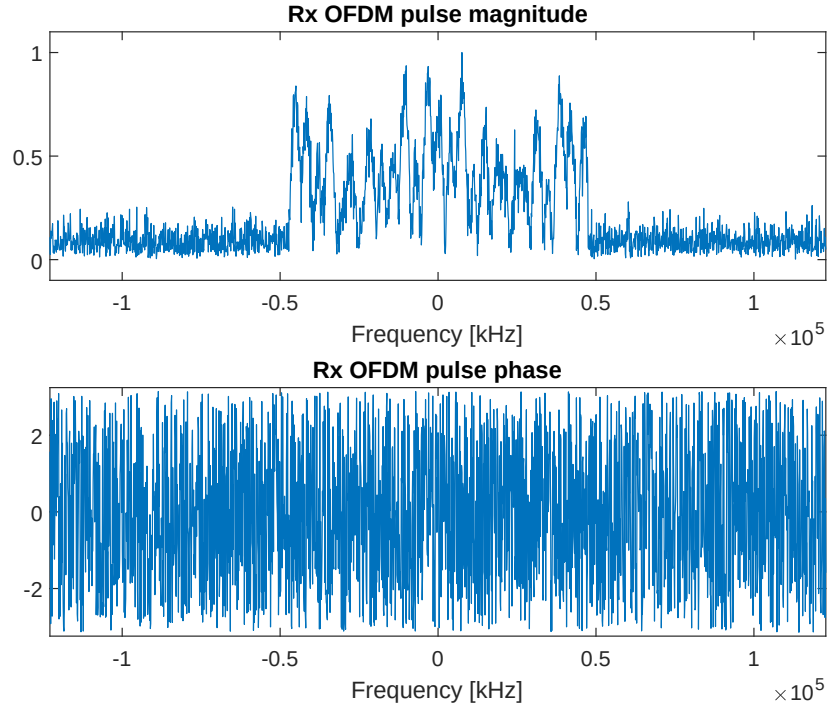


Figure 5.4: Frequency domain Rx OFDM pulse.

Figure 5.2 shows the time domain OFDM at the output of the transmitter (including

the CP). On close inspection, it is possible to see that the end and the beginning of the OFDM pulse are the same in the In-Phase plot and the Quadrature plots.

Figure 5.3 shows the input to the receiver after the transmitter pulse has traveled through the channel. No apparent trends can be deduced from this figure.

Figure 5.4 depicts the input to the receiver in the frequency domain. Comparing the magnitude plot of the received signal with the magnitude plot of the transmitted signal in Figure 5.1 the effect of the channel is apparent. Moreover, some noise has appeared in the guard subcarriers as a result of the AWGN.

Figure 5.5 shows the matched filter used in the receiver. The phase plot reflects that the matched filter is the conjugate of the transmitted symbol in Figure 5.1.

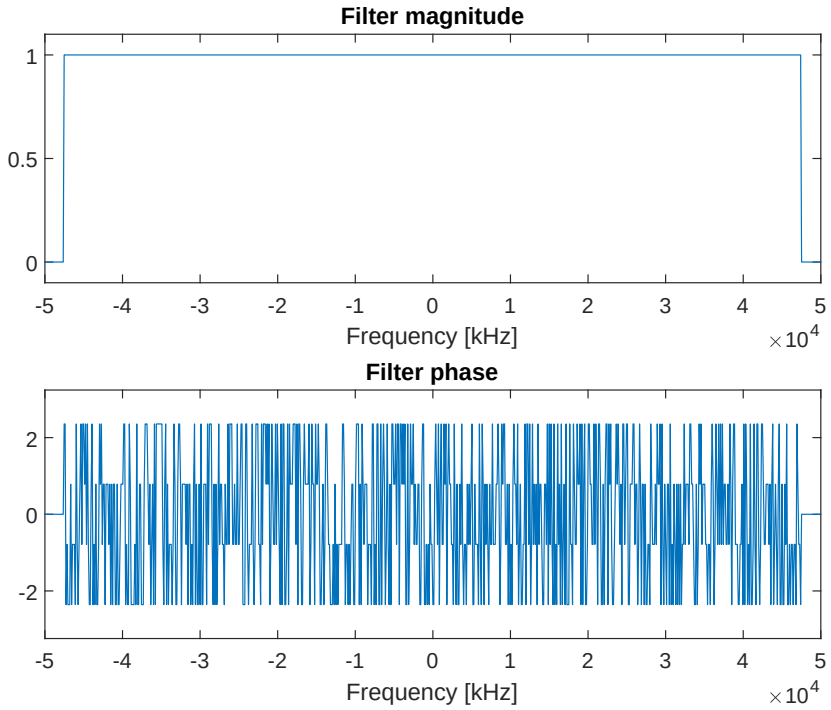


Figure 5.5: Frequency domain matched filter.

Figure 5.6 shows the output of the correlator in the receiver. The magnitude phase plot shows an interference pattern from the different tones corresponding to the simulated targets.

Figure 5.7 shows the real and imaginary output of the correlator. In this figure it is possible to see the superposition of the different tones corresponding to each target.

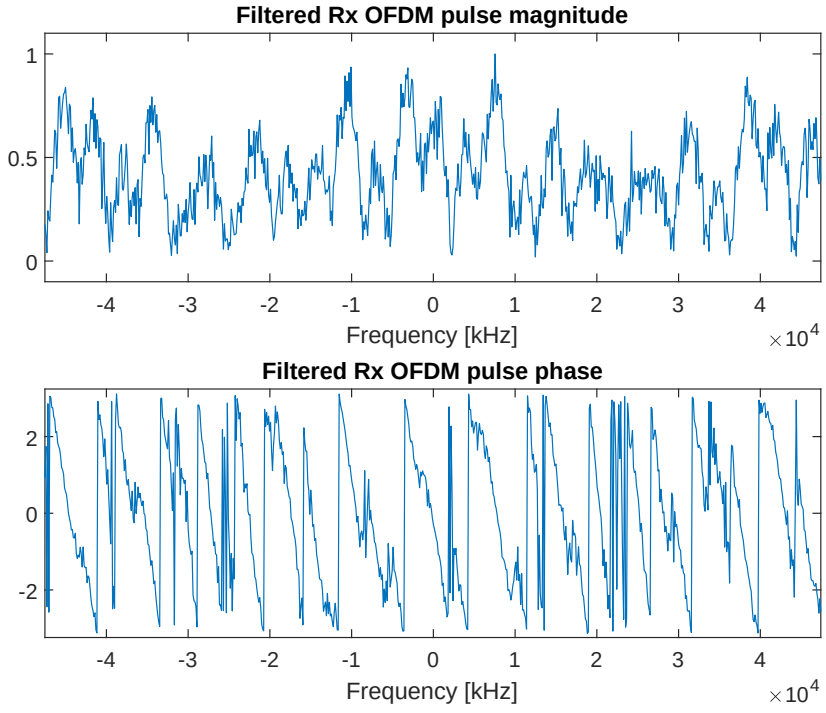


Figure 5.6: Filtered frequency domain Rx OFDM pulse magnitude and phase.

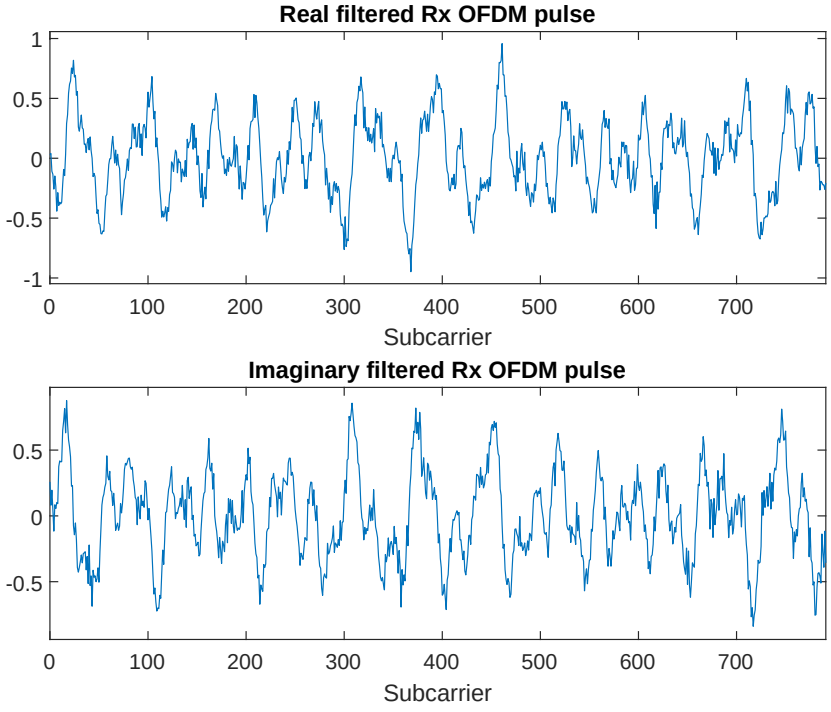


Figure 5.7: Filtered frequency domain Rx OFDM pulse real and imaginary components.

Figure 5.8 shows the output of the correlator for a simulation run where only Target

2 in Table 5.2 was simulated. In this figure, the tone corresponding to the target can be easily identified. The period of the tone can be measured to be approximately 60 samples and, using Equation A.10 the range to target two can be calculated to be 20.83 m. The result is close to the 20 m range in Table 5.2 considering the inaccuracy from measuring the period of the noisy sinusoid by hand.

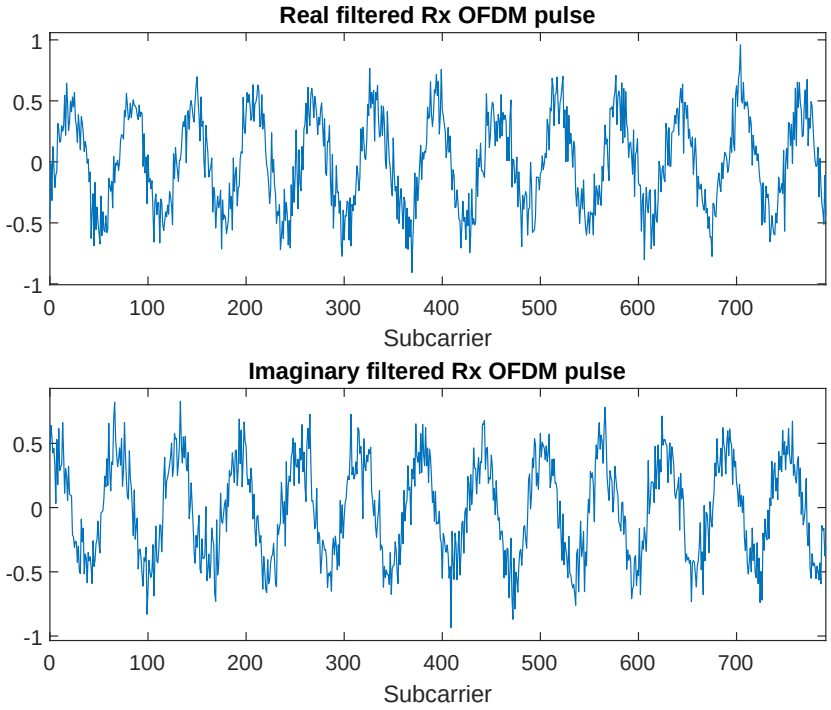


Figure 5.8: Filtered frequency domain Rx OFDM pulse for Target 2.

Figure 5.9 shows the Range-Doppler map at the output of the radar. The crosses represent the real range and speed of the simulated targets. Table 5.2 shows a detailed comparison between the real and measured ranges and speeds.

Table 5.2 shows that the errors in range and speed are minimal because the only source of error is the size of the range and speed bins. These bins limit the resolution of the radar but, as seen in the table, the size is sufficient. The size of the bins in the range axis is given by the number of points in the IFFT used to compute the range and the CP length. The size of the bins in the speed axis is given by the symbol duration and the length of the Coherent Processing Interval[1] (CPI).

In Figure 5.9 the targets are clearly visible, because the only noise source was a small AWGN added to the channel. In more realistic scenarios, the noise floor is expected to be higher due, primarily, to interference signals. Interference signals can arise from radar echoes beyond the CP length of the OFDM symbols or OFDM symbols coming from neighboring radars.

[1]The CPI is the number of pulses that are combined to generate a Range-Doppler map

Target	Real distance [m]	Measured distance [m]	Real speed [m/s]	Measured speed [m/s]
1	17	17.09	0	0
2	20	20.14	10	10
3	60	59.81	-30	-30
4	35	34.79	90	90

Table 5.2: Targets used for the simulation.

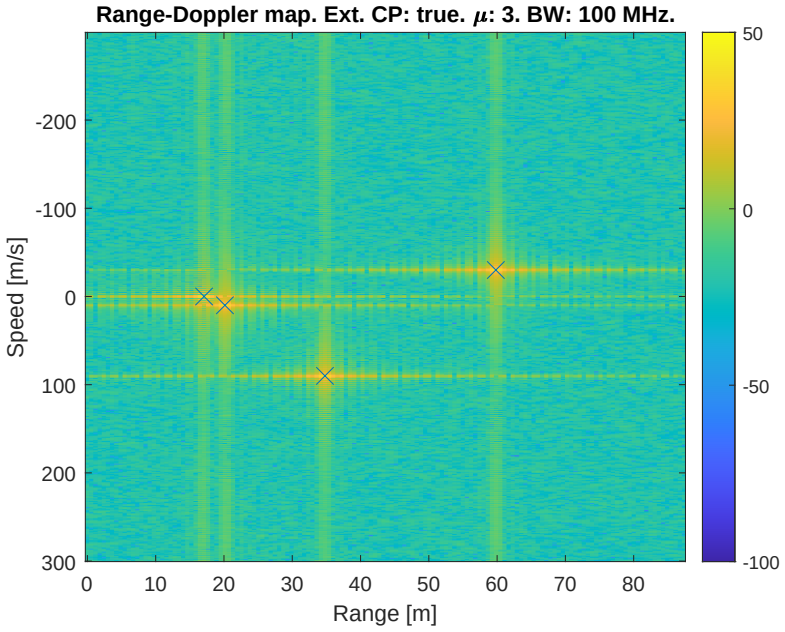


Figure 5.9: Range Doppler map.

The target signals in Figure 5.9 are sinc functions instead of delta functions. This discrepancy is caused by windowing. In the range dimension, windowing is a byproduct of the filtering done in the correlator: the received OFDM pulse spans 2048 subcarriers, however, the filter is only applied to the 792 data subcarriers while the rest are suppressed. In the speed dimension the windowing is a result of the length of the CPI not being a multiple of the Doppler period.

A range plot can be obtained by non-coherently integrating across an arbitrary number of symbols. The range plot in Figure 5.10 was obtained from non-coherently integrating 256 symbols. Note that non-coherent integration increases the logarithmic SNR by approximately $10 \log_{10} N$, where N is the number of symbols integrated. Again we can see the sinc shape of the target signals due to windowing.

The speed resolution seen in Figure 5.9 shows that the radar is not well suited for slow moving targets. The reason is that the pulse length is very short which means that, for a slow moving target, a large number of pulses need to be integrated for the Doppler frequency to be detected. In other words, the phase change is so slow compared to the sampling frequency that the target looks stationary for

reasonable length observation windows. This limitation is inherent to the timing characteristics of the 5G pulse and will remain a problem in 6G as the pulse length is expected to remain the same or be reduced. To work around this limitation, the comparatively high range resolution of the system can be leveraged. As the target moves, its position will change in the range axis such that, using some post-processing the speed could be derived as the differential in range over time. Higher frequency systems have shorter pulses which makes the speed resolution problem worse but also have lower latency and more range resolution which result in better speed estimations using the range differential method described before.

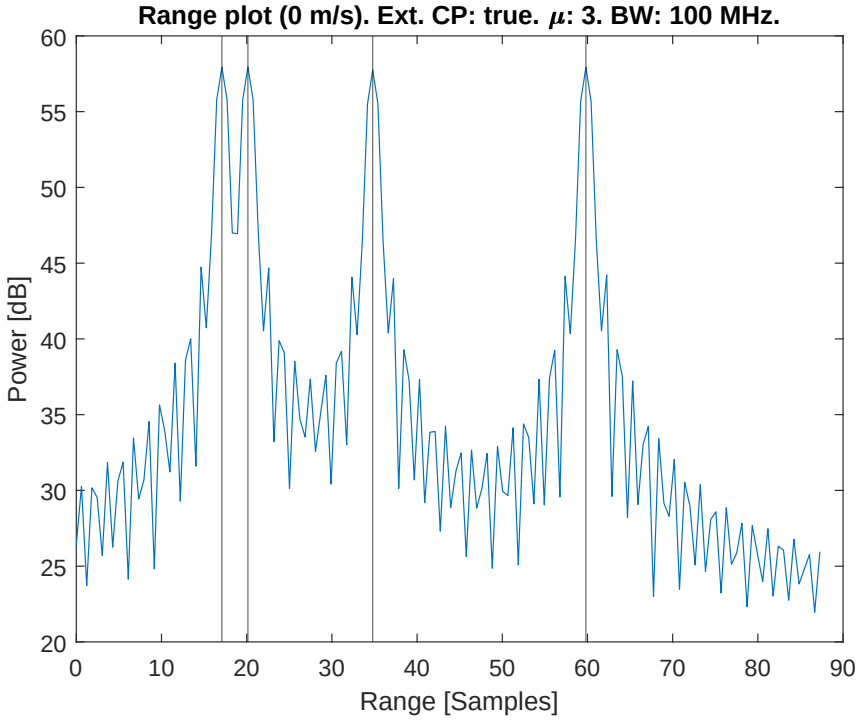


Figure 5.10: Range plot at 0 m/s.

5.2 Simulation results from system-level HDL testbench

The results from the system-level testbench will be explained in this section.

The OFDM pulse parameters used for this simulation run were:

- Subcarrier spacing: 60 kHz (Numerology 2).
- Bandwidth: 50 MHz (792 data subcarriers).
- Extended prefix.
- Subcarrier modulation: QPSK.

The FFT blocks were configured for a 1024 point FFT to save resources. The output of the transmitter was suitably interpolated to obtain an OFDM symbol with 2048

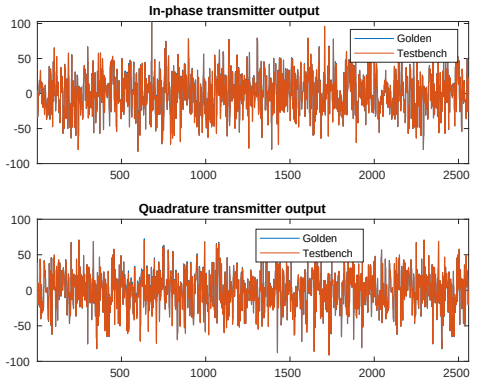
samples (not including the CP). Also, the input to the receiver was decimated to obtain an OFDM symbol with 1024 samples.

Four stationary targets were simulated as seen in Table 5.3.

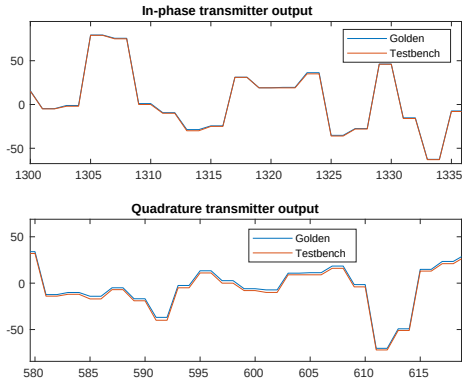
Target	Distance [m]
1	60
2	250
3	403
4	578

Table 5.3: Targets used for the testbench.

Figure 5.11a shows a comparison between the in-phase and quadrature outputs of the reference transmitter and the HDL transmitter and Figure 5.11b is a zoomed in version.

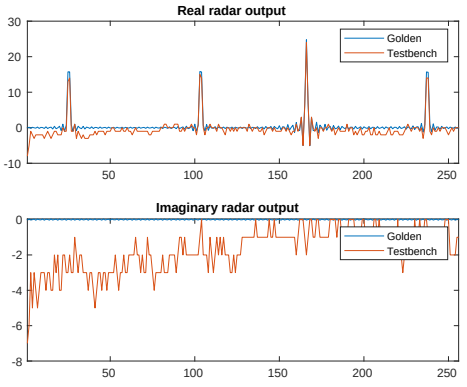


(a) Complete transmitter output.

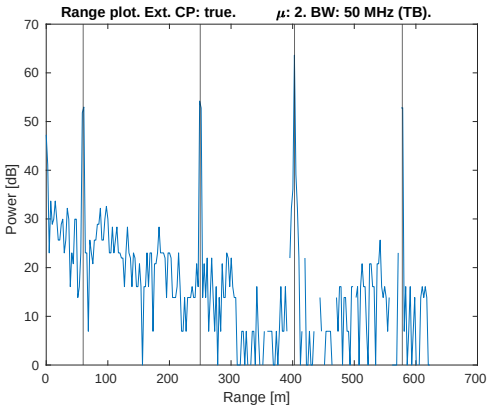


(b) Zoomed transmitter output.

Figure 5.11: Transmitter output.



(a) Raw radar output.



(b) Range plot.

Figure 5.12: Radar output.

The in-phase and quadrature outputs of the reference and HDL transmitters follow each other closely in Figure 5.11a. However, Figure 5.11b shows that the HDL transmitter signal is shifted in the y axis compared to the reference transmitter signal due to the sample rounding strategy used in the HDL transmitter. The HDL transmitter used a truncation rounding strategy which slowly converges towards minus infinity. The error is not big enough to justify the substantial increase in hardware overhead that arises from more advanced rounding strategies such as flooring.

Figure 5.12a displays the comparison between the output of the reference radar and the output of the HDL radar. The offset due to rounding discussed previously is more apparent in this figure because of the larger number of processing steps that the signal has undergone. The noise in the imaginary output is due to artifacts in the FFT implementation. The channel is real so the imaginary output should be zero because the output of the radar, as explained in Section 2.3.3, is an estimation of the channel.

Figure 5.12b shows the Range map obtained from the output of the reference radar. This processing step was not implemented in the HDL radar as it is a visualization tool. The peak at zero meters is caused by the rounding offset. The remaining four peaks represent each of the four targets. Table 5.4 shows a comparison between the real and measured range as well as the error. The error is small because the range bins are sufficiently small. If more performance is required, it would be possible to use a smaller FFT at the expense of range resolution.

Target	Real distance [m]	Measured distance [m]	error (%)
1	60	61.04	1.73
2	250	249.02	0.39
3	403	402.83	0.04
4	578	578.61	0.11

Table 5.4: Target range comparison.

5.3 Resource usage of the radar implementations

In this section we will compare the resource usage for the two radar implementations. We will compare the receivers and the transmitters separately as the resource overhead for the RF tiles and the control logic is the same regardless of the implementation. The initial implementation was synthesized with no timing constraint to obtain a best-case estimate of the resource usage and the optimized implementation was synthesized using a 241.76 MHz clock which is the system clock in the PL as configured in the template project. Tables 5.5 and 5.6 show a comparison between the resource usage for the initial transmitter and the optimized transmitter and Tables 5.7 and 5.8 show a comparison between the resource usage for the initial receiver and the optimized receiver.

Design	CLB LUT	CLB Registers	CARRY8	DSP slices
Initial	1080091	325998	153167	4272
optimized	6	14	2	0

Table 5.5: Total resource usage comparison for the transmitter designs.

Design	CLB LUT	CLB Registers	CARRY8	DSP slices
Initial	253.97	38.33	288.12	100
optimized	< 0.01	< 0.01	< 0.01	0

Table 5.6: Percentual resource usage comparison for the transmitter designs.

Tables 5.5 and 5.6 shows the expected results when comparing a full transmitter with a ROM-based transmitter. On the one hand, the initial transmitter requires more resources than there are available in the ZCU111 while, on the other hand, the optimized transmitter barely uses a small percentage of the available resources. The initial design offers more flexibility at the cost of resource usage. If this flexibility was needed a similar approach to that used for the optimized receiver could be realized thus reducing the resource usage.

Design	CLB LUT	CLB Registers	CARRY8	DSP slices	BRAM
Initial	2676580	533506	637558	3750	0
optimized	109736	136662	218	21	7.5

Table 5.7: Total resource usage comparison for the receiver designs.

Design	CLB LUT	CLB Registers	CARRY8	DSP slices	BRAM
Initial	629.37	62.72	691.42	87.78	0
optimized	25.8	16.07	0.41	0.49	0.69

Table 5.8: Percentual resource usage comparison for the receiver designs.

Tables 5.7 and 5.8 are particularly interesting when comparing the resource usage of the direct FFT implementation with the resource usage of the FFT IP core. The initial design using the direct FFT implementation requires more resources than are available in the ZCU111 and almost consumes all of the available DSP slices. In contrast, the optimized design using IP blocks consumes only a fourth of the CLB LUTs and only a 0.5 % of the DSP slices. Achieving this level of improvement without IP blocks would have required a thorough redesign of the FFT blocks that would have shifted the scope of the project from the radar system to the optimization of this particular block. The BRAM usage in the optimized design comes from the FFT IP blocks and the ROM used to store the filter coeffcients.

5.4 Physical testbench results

The physical testbench was tested first using the transmitter to ensure the setup was correct and then the whole system was tested.

5.4.1 Testing the setup

The transmitter testbench was used to test the transmitter ROM as well as the ADC and DAC tile configuration. The complex test sinuosid in Figure 5.13 was stored in the transmitter ROM and transmitted in a cyclical fashion. The complex test sinusoid had a frequency of 120 kHz, a length of 1280 samples at a sampling frequency of 122.88 MHz and was interpolated 16 times using a ZOH to match the required sampling frequency of the DAC (1.96 GHz).

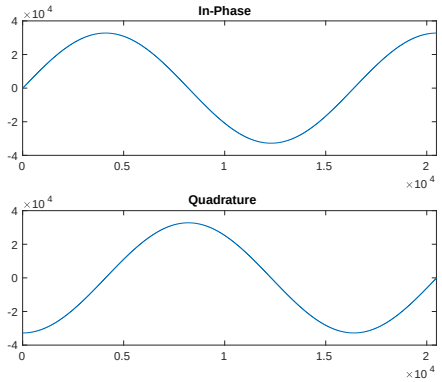


Figure 5.13: Test sinusoid signal at 120 kHz (sampling frequency: 1.96 GHz).

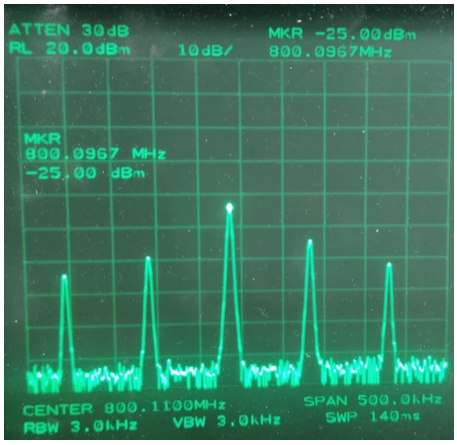


Figure 5.14: Spectrum of the DAC output.

The DAC output was tested using a spectrum analyzer as seen in Figure 5.14. The peak at 800.0967 MHz is the result of modulating the output frequency of the ROM with a 800 MHz carrier frequency and the harmonics are the result of the discontinuity that appears at the end of a writing cycle of the ROM. The complex

sinusoid frequency was 120 kHz but this peak is obscured by the larger 96.7 KHz peak previously described.

After the output of the DAC was verified, the DAC and the ADC were connected together using a coaxial cable as seen in Figure 4.14. Figure 5.15 shows a comparison between the output and the input of the radar. The amplitude of both pulses has been normalized for easier comparison.

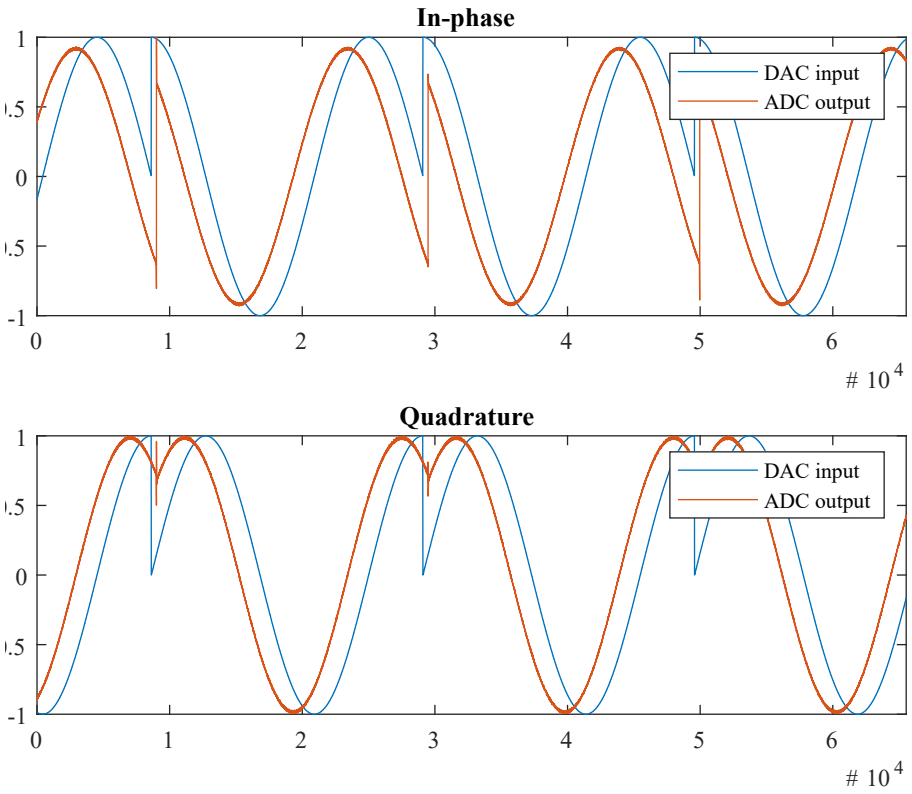
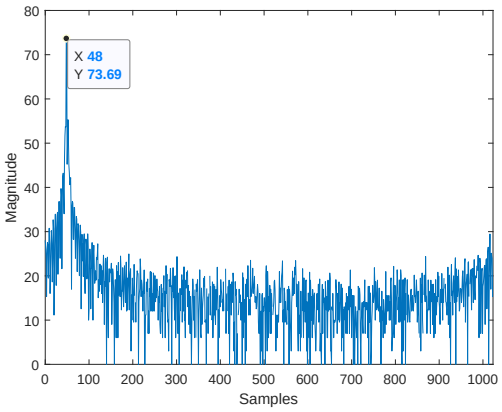


Figure 5.15: Comparison between the radar output (DAC input) and the radar input (ADC output).

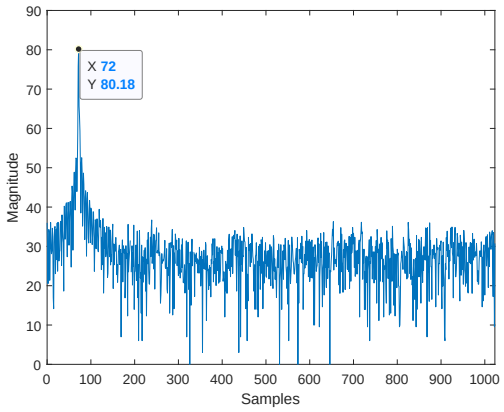
Figure 5.15 shows that the input of the radar is a delayed version of the output of the radar as it has travelled through the coaxial cable. Also, the initial transient response of the channel is visible whenever a new symbol is transmitted in the form of spikes. Finally, the noise in the received signal is a result of amplification and discretization at different points of the signal path.

5.4.2 Testing the radar system

The radar system was tested using two coaxial cables to simulate targets at different distances: 1.5 m and 20 m. Figures 5.16a and 5.16b show the range plot for the 1.5 m cable and the 20 m cable respectively.



(a) 1.5 m cable.



(b) 20 m cable.

Figure 5.16: Range plots

Figure 5.16a shows a peak at 48 samples of delay representing the delay of the signal after traveling trough the coaxial cable. Figure 5.16b shows a peak at 72 samples of delay representing the longer delay of the signal after traveling trough the longer coaxial cable.

Comparing Figures 5.16a and 5.16b we can deduce that there is some baseline delay in the system that creates an offset. This delay is caused by the RF tiles as well as the traces and components that the signal needs to travel through to reach the coaxial cable. This offset could be eliminated in post-processing or adding a delay in the receiver synchronization.

Table 5.9 shows a comparison between the expected delay and the measured delay. The expected delay in samples was calculated using Equation 2.4. The speed of light in a coaxial cable was taken as 2/3 of the speed of light in a vacuum and the sampling frequency of the system was 247.45 MHz. The results of Equation 2.4 were floored as it is the rounding strategy used in the radar system.

Cable length (m)	Theoretical delay (samples)	Measured delay (samples)	Error (samples)
1.5	2	48	46
20	27	72	47

Table 5.9: Radar system results.

The system offset was calculated using the average of the error in Table 5.9 and subtracted from the measurements to obtain Table 5.10.

Cable length (m)	Theoretical delay (samples)	Measured delay (samples)	Error (samples)
1.5	2	1.5	0.5
20	27	25.5	1.5

Table 5.10: Radar system results without offset.

Table 5.10 shows that the observed peaks correspond the delayed signal after traveling through the coaxial cable. If the speed of light in each cable was known, the error should be zero once the offset was removed. However, since the datasheet for the coaxial cables was not available, an approximation was necessary which is the source of the error.

6

Conclusion

In this project a working digital radar using 5G OFDM pulses has been successfully implemented and tested. A suitable radar architecture has been found in the FMCW OFDM radar that reuses most of the components in a standard OFDM receiver. A system-level simulation of the selected design has been implemented to demonstrate the underlying principles of the FMCW radar and its compatibility with the 5G standard. The system-level simulation has also been successfully used to design a system-level testbench that has been used to verify the FMCW OFDM radar implementation. The FMCW radar has been implemented on HDL first using a complete custom design for all the components and, later, a mix of custom and IP blocks to create a hardware optimized design. The optimized design has been combined with an existing project to create a radar system that could be tested using the ZCU111 development board. Finally, the radar has been successfully tested using simulated targets through the use of different length coaxial cables.

6.1 Results discussion summary

Using the system-level simulation results it has been possible to determine that 5G pulses on FMCW radars offer a satisfactory level of range resolution. However, the speed resolution has been found to be too low for slow moving targets. Range resolution can be leveraged using post-processing to more precisely estimate speed as the differential of range.

The system-level simulation has proven that, for 12 bit word lengths, the FMCW OFDM radar does not suffer from excessive quantization noise. The HDL implementation, however, suffers from a small negative offset due to the rounding strategy used. In the current implementation this error can be easily removed from the output as it only occupies the DC bin.

During the implementation, it has been found that a direct implementation of the Coley-Tuckey FFT algorithm requires excessive resources and that more advanced implementations are required. In this project, the advanced implementations have been sourced from IP blocks.

The tests performed on the ZCU111 RFSoC development board have proven that the design works as expected and can successfully measure the delay in a coaxial cable. The tests have also underlined the importance of configuring the scaling in the system correctly to avoid distortion due to overflow.

6.2 Further work

The system-level simulation in this project has shown that FMCW radars using 5G OFDM pulses have good range resolution but poor speed resolution. Thus, computing a Range-Doppler matrix is not worth the extra processing effort. A better solution to increase SNR and find the speed of the targets is to first non-coherently integrate a number of range plot, and then use post-processing to find the speed of the target as a range differential. Figure 6.1 shows the proposed receiver design.

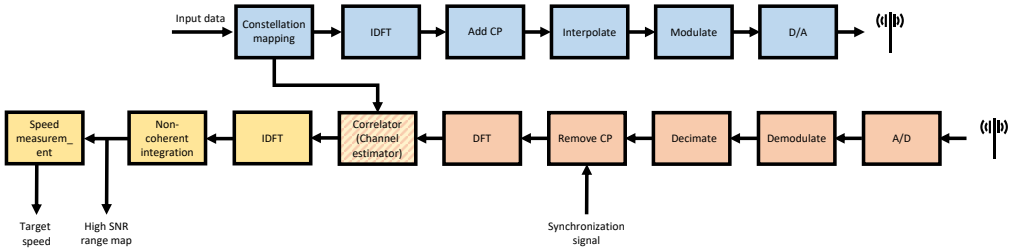


Figure 6.1: Proposed receiver design.

The implementation work in this project has shown that the FFT blocks are the most resource intensive blocks and that future design hardware optimization will need to address this issue. The design optimization of this block will need to focus, specially, on the trade-off between latency and hardware utilization.

The testing has shown that overflow can be a problem in the system as the attenuation is dependent on the range. A manual approach has been employed for this project but a more advanced adaptive scaling mechanism will likely be needed if this type of radar is to ever be used. This scaling mechanism will need to introduce a known delay in the system to maintain accuracy in the measurement.

Bibliography

[1] H. Andersson. (2021) Unpacking joint communication and sensing in 6G. [Online]. Available: https://www.ericsson.com/en/blog/2021/10/joint-sensing-and-communication-6g

[2] H. Wymeersch, D. Shreshta, K. Schindhelm, A. Wolfgang, S. Lindberg, C. Morais de Lima, A. Ramirez, V. Yajnanarayana, T. Svensson, H. Chen, M. A. Nazari, H. Kim, Y. Ge, M. F. Keskin, T. E. Abrudan, A. Behravan, and R. Baldemair, "Localisation and sensing use cases and gap analysis," Hexa-X, Tech. Rep., 2023.

[3] E. M. Taghavi, H. Wymeersch, A. Behravan, K. Schindhelm, A. Wolfgang, V. Yajnanarayana, S. Lindberg, M. F. Keskin, H. Chen, T. Svensson, H. Guo, M. A. Nazari, A. Ramirez, Y. Ge, R. Schroeder, T. E. Abrudan, and M. H. Moghaddam, "Initial models and measurements for localisation and sensing," Hexa-X, Tech. Rep., 2023.

[4] H. Wymeersch, A. Stavridis, K. Schindhelm, H. Chen, H. Guo, M. F. Keskin, S. Lindberg, J. M. Mateos-Ramos, M. H. Moghaddam, M. A. Nazari, I. Perera, A. Ramirez, R. Schroeder, T. Svensson, A. Wolfgang, and V. Yajnanarayana, "Final models and measurements for localisation and sensing," Hexa-X, Tech. Rep., 2023.

[5] B. J. Donnet and I. D. Longstaff, "Combining MIMO radar with OFDM communications," in *2006 European Radar Conference*, 2006, pp. 37–40.

[6] C. Sturm and W. Wiesbeck, "Waveform design and signal processing aspects for fusion of wireless communications and radar sensing," *Proceedings of the IEEE*, vol. 99, no. 7, pp. 1236–1259, 2011.

[7] C. Aydogdu, H. Wymeersch, and M. Rydstrom, "Can automotive radars form vehicular networks?" in *2020 IEEE Radar Conference (RadarConf20)*, pp. 1–6.

[8] J. B. Sanson, D. Castanheira, A. Gameiro, and P. P. Monteiro, "Fusion of radar and communication information for tracking in OFDM automotive radar at 24 GHz," in *2020 IEEE/MTT-S International Microwave Symposium (IMS)*, pp. 1153–1156.

[9] (2023) Zynq UltraScale+ RFSoC ZCU111 evaluation kit. [Online]. Available: https://www.xilinx.com/products/boards-and-kits/zcu111.html

[10] L. Crockett, D. Northcote, and R. W. Stewart, *Software Defined Radio with Zynq UltraScale+ RFSoC*. Strathclyde Academic Media, 2023, ch. Introduction.

[11] J. B. Anderson, *Carrier Transmission.* Wiley-IEEE Press, 2005, pp. 79–160.

[12] D. Allan, *Software Defined Radio with Zynq UltraScale+ RFSoC.* Strathclyde Academic Media, 2023, ch. OFDM: Orthogonal Frequency Division Multiplexing.

[13] M. N. Cohen, *Pulse Compression in Radar Systems.* SciTech Publishing, 1987, pp. 465–501.

[14] K. M. Braun, "OFDM radar algorithms in mobile communication networks," Ph.D. dissertation, 2014.

[15] (2024) Range-doppler response. [Online]. Available: https://se.mathworks.com/help/phased/ug/range-doppler-response.html

[16] E. Dahlman, S. Parkvall, and J. Skold, *5G NR : the Next Generation Wireless Access Technology.* Elsevier Science amp; Technology, 2018, ch. What is 5G?

[17] ——, *5G NR : the Next Generation Wireless Access Technology.* Elsevier Science amp; Technology, 2018, ch. Overall transmission structure.

[18] *UG1037 Vivado Design Suite: AXI Reference Guide*, AMD.

[19] *PG109 Fast Fourier Transform LogiCORE IP Product Guide*, AMD.

[20] *PG269 Zynq UltraScale+ RFSoC RF Data Converter v2.6 Gen 1/2/3/DFE LogiCORE IP Product Guide*, AMD.

[21] *PG261 System Integrated Logic Analyzer v1.0 Product Guide*, AMD.

[22] *PG159 Virtual Input/Output v3.0 Product Guide*, AMD.

A

Mathematical derivation of the channel estimation-based radar architecture

To better understand the channel estimation architecture it is convenient to analyze the radar signals at different points in the radar. To make the analysis simpler we will use signals in time domain and we will utilize a generic OFDM pulse. We will begin by looking at the OFDM pulse before modulation:

$$x(t) = \sum_{k=0}^{N_{sc}-1} a_k \exp(j2\pi\Delta f k t) \tag{A.1}$$

After modulation with the carrier signal, we obtain the transmit signal:

$$x_{tx}(t) = \sum_{k=0}^{N_{sc}-1} a_k \exp(j2\pi\Delta f k t) \exp(j2\pi f_c t) \tag{A.2}$$

For the sake of tidiness, let us define $\omega_k = 2\pi\Delta f k$ and $\omega_c 2 = \pi f_c$. Equation A.3 then becomes:

$$x_{tx}(t) = \sum_{k=0}^{N_{sc}-1} a_k \exp(j\omega_k t) \exp(j\omega_c t) \tag{A.3}$$

The received signal can be expressed as the convolution of the transmitted signal and the channel as seen in Equation A.4. Generally, in literature descriptions of the received signal, the Doppler shift is neglected. However, for radar applications the Doppler shift contains important information about speed. Therefore, we have chosen to include the Doppler term:

$$x_{rx}(t) = \sum_{k=0}^{N_{sc}-1} a_k \cdot H(\omega_k) \exp(j(\omega_k + \omega_{k,D})t) \exp(j(\omega_c + \omega_{c,D})t) \tag{A.4}$$

where $\omega_{k,D}$ is the Doppler shift in each subcarrier and $\omega_{c,D}$ is the Doppler shift in the carrier frequency.

After demodulation we are left with the original OFDM pulse convolved with the channel and multiplied by the Doppler shift as seen in Equation:

$$x_{rx}(t) = \sum_{k=0}^{N_{sc}-1} a_k \cdot H(\omega_k) \exp(j(\omega_k + \omega_{k,D})t) \exp(j\omega_{c,D}t) \tag{A.5}$$

Assuming narrowband conditions (The bandwidth of the pulse is much smaller that the carrier frequency), we can further simplify Equation A.5 by considering the Doppler shift in the subcarriers negligible. Finally, we are left with:

$$x_{rx}(t) = \sum_{k=0}^{N_{sc}-1} a_k \cdot H(\omega_k) \exp(j\omega_k t) \exp(j\omega_{c,D}t) \tag{A.6}$$

Following demodulation we can recover the data in the subcarriers using a DFT. For simplicity we will assume the Doppler term is constant for the duration of the symbol. Instead of t we will express the time variable for the Doppler shift as τ to denote the difference:

$$s_k(\tau) = \mathcal{F}\{x_{rx}(t)\} = \exp(j\omega_{c,D}\tau) \cdot a_k \cdot H(\omega_k) \tag{A.7}$$

Assuming the transmitter and receivers are synchronised, we could obtain the frequency response of the channel by removing the data modulated in the subcarriers. Note that this operation is equivalent to the equalization explained in Section 2.2.5 but instead of removing the channel contribution we are removing the data contribution. The general result is:

$$h_k(\tau) = \frac{s_k(\tau)}{a_k} = \exp(j\omega_{c,D}\tau) \cdot H(\omega_k) \tag{A.8}$$

A delay in time domain is a phase shift in frequency domain. Suppose we have a target at distance R, the delay of the echo will be $t_{del} = 2R/c$. Using the Fourier transform, $\mathcal{F}\{x(t - t_{del})\} = X(\omega) \cdot exp(-j\omega t_{del})$.

Assuming there is only one target in the signal path, we can substitute the channel frequency response in Equation A.8 by a phase and a complex envelope:

$$h_k(\tau) = \exp(j\omega_{c,D}\tau) \cdot E_k \cdot \exp(-j\omega_k t_{del}) \tag{A.9}$$

where E_k is the complex envelope and t_{del} is the delay due to range. Substituting $\omega_k = 2\pi\Delta f k$ and $t_{del} = 2R/c$ we get:

$$h_k(\tau) = \exp(j\omega_{c,D}\tau) \cdot E_k \cdot \exp(-j\frac{4\pi R\Delta f k}{c}) \tag{A.10}$$

Equation A.10 shows that the range to the target will appear as a tone with period $T_{range} = \frac{c}{2R\Delta f}$. Combining many consecutive channel estimations we will be able to measure the frequency of the Doppler shift. The result will be a radar matrix where

the rows represent the channel estimation for each OFDM symbol and each column corresponds to a particular subcarrier.

Using a IDFT across the rows will yield a range map and using a DFT across the columns will result in a Doppler map. Combining the results will produce a range-Doppler map.