DAB Processing Chain for Passive Radar Applications

Prepared by:
Callum Rhys Tilbury

Prepared for:
Dr Stephen Paine
Department of Electrical Engineering
University of Cape Town

Submitted to the Department of Electrical Engineering at the University of Cape Town
in partial fulfilment of the academic requirements for a
Bachelor of Science degree in Electrical and Computer Engineering

November 11, 2020

Declaration

1. I know that plagiarism is wrong. Plagiarism is to use another's work and pretend that it is one's own.
2. I have used the IEEE convention for citation and referencing. Each contribution to, and quotation in, this report from the work(s) of other people has been attributed, and has been cited and referenced.
3. This report is my own work.
4. I have not allowed, and will not allow, anyone to copy my work with the intention of passing it off as their own work or part thereof.

Callum Rhys Tilbury

Date

Acknowledgements

> No man is an island entire of itself; every man is a piece of the continent, a part of the main.
>
> —*John Donne*

This project—and the degree that it concludes—could not have been completed without the love and support received along the way. I am incredibly privileged and I feel overwhelmingly blessed. I wish to thank *all* of those who have been a part of the journey, regardless of the role played. Nonetheless, some special notes of appreciation are due.

Firstly, to the wonderful benefactors that have enabled me to embark on this university experience—the Allan Gray Orbis Foundation, the Klaus-Jürgen Bathe Leadership Scholarship, PricewaterhouseCoopers South Africa, Peralex Electronics, and the UCT Engineering Faculty: thank you for your faith in me, and for your unending generosity.

To my supervisor, Dr Stephen Paine: thank you for your patience and wisdom during this project, and for your willingness to help—even in the late hours of the night.

To Mark Cammidge: thank you for all of your assistance over the years. Thank you, too, for meticulously proofreading this report so cheerfully.

To my passionate lecturers at UCT, and my equally-passionate teachers back at Fish Hoek High School: thank you for dedicating so much of your time and energy for my sake. Thank you for inspiring me, challenging me, and helping me grow.

To my wonderful and caring friends: Nic, thank you for the deep friendship we share. Kate, thank you for the constant support and encouragement you so readily provide to me—your companionship has made this tough year more bearable. Matt, thank you for your quirky sense of humour, which makes my life brighter. Tom, Justin, and everyone else—thank you for being an integral part of this tumultuous yet gratifying escapade.

To my lift-club partner, Mark Keeling: thank you for the hours and hours that we shared in commute, and the many kilometres we added to your car's odometer.

To my loving family: Mom and Dad, thank you for the innumerable sacrifices you have made for me, without a hint of complaint. Your unwavering support has been the bedrock of my life. Gran and Papa, thank you for your love and unceasing kindness. My dear siblings—Bron and Hennie, Jes and Dave—thank you for being some of my closest friends with whom I can enjoy life. My second family—Claire, Sara, Omi—thank you for welcoming me into your lives so graciously.

Finally, to my best friend, Gemma: thank you for all the adventures we have shared, and for the love you continue to show me. Thank you for teaching me so much about the world.

Abstract

Passive Radar (PR) is a variant of radar technology in which no dedicated transmitter is required for operation. Instead, PR systems use so-called Illuminators of Opportunity (IOO)—such as existing radio-station broadcasts—for the detection of objects in a scene. In the past decade, there has been an increased interest in the use of *digital* broadcasting signals as IOOs, since these signals offer a variety of benefits in a PR context. One of the digital broadcasting standards that has garnered attention for use in PR is the Digital Audio Broadcasting (DAB) format—an increasingly-popular mode of digital radio.

The aim of this project was to research, design, and implement a DAB processing chain, with the intention that it eventually be integrated into a PR system. Though prior work has been done on DAB-based PR implementations, the literature lacks a clear description of the DAB pipeline used for such scenarios. Moreover, the work that covers DAB signals more comprehensively focuses too much on the intricacies of the DAB standard, in a way that is unnecessarily complex for PR applications. Therefore, there is a need for a clear and thorough exposition of the design procedure for a DAB processing chain, specifically within the context of PR.

This report aims to document the project's design steps, and its associated results. After a brief surveillance of the available literature, a comprehensive consideration of the relevant aspects of the DAB standard is provided, including the key theoretical concepts that underpin it—namely, Coded Orthogonal Frequency Division Multiplexing (COFDM) and Differential Quadrature Phase-Shift Keying (DQPSK). Thereafter, a detailed unpacking of the designed DAB processing chain is provided, starting from a high-level overview, and then zooming in to the individual functional blocks. The designed DAB chain is finally validated, illustrating that it functions correctly and is suitable for use in a PR system.

Contents

List of Figures

Abbreviations

AM Amplitude Modulation

ARD Amplitude-Range-Doppler

ASK Amplitude-Shift Keying

ATSC Advanced Television Systems Committee

BPSK Binary Phase-Shift Keying

CAZAC Constant-Amplitude Zero-AutoCorrelation

COFDM Coded Orthogonal Frequency Division Multiplexing

DAB Digital Audio Broadcasting

DFT Discrete Fourier Transform

DPSK Differential Phase-Shift Keying

DQPSK Differential Quadrature Phase-Shift Keying

DSI Direct Signal Interference

DVB-T Digital Video Broadcasting - Terrestrial

DVB-T2 Digital Video Broadcasting - Second Generation Terrestrial

ECM Electronic Countermeasures

EM electromagnetic

ETSI European Telecommunications Standards Institute

FDM Frequency Division Multiplexing

FEC Forward Error Correction

FFT Fast Fourier Transform

FM Frequency Modulation

FSK Frequency-Shift Keying

ICASA Independent Communications Authority of South Africa

IFFT Inverse Fast Fourier Transform

IOO Illuminators of Opportunity

IQ In-phase and Quadrature

ISI Intersymbol Interference

MCM Multicarrier Modulation

NS Null Symbol

OFDM Orthogonal Frequency Division Multiplexing

PR Passive Radar

PRS Phase Reference Symbol

PSK Phase Shift Keying

QPSK Quadrature Phase-Shift Keying

RADAR Radio Detection and Ranging

SFN Single-Frequency Networks

SKA Square Kilometer Array

SNR Signal-to-Noise Ratio

Chapter 1

Introduction

> Philosophers have hitherto only interpreted the world in various ways; the point is to change it.
>
> —*Karl Marx*

This report describes a project involving the implementation of a processing chain for Digital Audio Broadcasting (DAB) signals, specifically oriented towards the context of Passive Radar (PR). The project consisted of research into the DAB standard, design and implementation of a corresponding processing chain, and a series of validation steps taken to ensure its correctness. This cursory chapter aims to outline the background for the project, followed by a listing of the project's key objectives, requirements, and scope.

1.1 Background

Fundamentally, radar[1] is a fairly straightforward concept, one that is analogous to the biological mechanism of *echolocation*—used by animals such as bats and dolphins. In the simplest form, a radar system consists of a co-located transmitter and receiver. If the transmitter sends out an electromagnetic (EM) signal—a short pulse, for example—this signal will propagate through space, hitting objects as it does. The collection of objects is often called a *scene*. Depending on the properties of the objects the EM waves hit, some of the energy will reflect off them, thus travelling back to the receiver. There will be a variety of time delays between sending out the signal and receiving the reflections, depending on the positions of the objects. Since these waves are travelling at the speed of light, which is known,[2] the time delays can be converted to distances—thus revealing the ranges to the various objects. Additional complexity can be introduced to calculate the objects' actual positions, velocities, and so on. This is the basis for an *active* radar system—because the transmitter is *actively* producing the EM waves.

A PR system, on the other hand, is different: instead of having a transmitter and receiver, it only has the latter. Rather than actively transmitting signals, it uses existing EM transmissions—sometimes called Illuminators of Opportunity (IOO)—that are being transmitted by some other operator—a radio station, for example. While the mathematics is slightly more complicated for this approach, the core

[1] The word *radar* was originally derived from the acronym, Radio Detection and Ranging; however, it has subsequently become a common noun, and is used in this report as such.

[2] The speed of light in a vacuum, c, is known exactly, as our measurement of distance is actually defined in terms of c, not the other way around. The speed of light in air, however, is slightly slower than this—and is thus an approximation. Nevertheless, this difference is usually negligible.

idea remains the same: reflections of the transmitted signal are measured, and using the calculated time delays, the locations of objects are calculated. This approach is sometimes called Passive Coherent Location, Passive Bistatic Radar, or Commensal Radar. The simple term, PR, will be used throughout this report.

For the past century of radar development, much of the focus has been on *active* radar systems, despite PR systems appearing as early as 1935. However, in recent decades, the use of PR has been shown to have a host of advantages over its active counterpart. As a result, there has been a renewed interest in the field, across a broad range of use cases.

A substantial amount of the research done in PR has involved the use of analogue broadcasting signals as IOOs, such as Frequency Modulation (FM) signals—likely because they are so ubiquitous worldwide. Nonetheless, these transmissions have a variety of downsides when implemented in a PR system, such as inconsistent range-resolution in the case of FM. In comparison, *digital* broadcasting signals avoid many of these issues. If a received signal was modulated using an openly-accessible digital scheme, the signal can be perfectly 'reconstructed', by going through a process of demodulation and subsequent remodulation. This procedure removes noise and other interference from the signal, which can result in superior PR performance. Furthermore, the use of digital signals as IOOs enables one to use a single antenna for the entire PR system, compared to the two antennas required for an analogue-based system.

One such digital broadcasting standard, DAB, has been shown to yield promising results. However, though work has been published involving the use of DAB signals in PR situations, there are no clear or comprehensive guides discussing the salient details of such work. The design of the DAB processing chain is usually glossed over in favour of the actual PR results. On the other hand, existing DAB-specific literature focuses too much on the intricacies of the DAB standard, complicating matters unnecessarily for PR applications. Therefore, there is a need for a focused investigation on using DAB signals in these contexts.

1.2 Objectives

The broad objective of this project was to design and build a DAB processing chain that could be integrated into a larger PR system. The chain was to be tested and validated, and well documented in the form of a report. Note, however, the goal here was not to build a *fully-fledged* DAB demodulator or decoder—one which could extract the audio signal from a provided DAB recording, for example. Instead, the intention was to create the components of a DAB chain that would be required by a PR system, and nothing more.

1.3 System Requirements

Based on the objectives above, a set of general system requirements was defined. The designed DAB processing chain was required to have the ability to:

- Pre-process a DAB signal from a provided recording.
- Demodulate the frames of the DAB signal.
- Remodulate the demodulated data, back into DAB frames, but perfectly reconstructed.

1.4 Scope & Limitations

This project was considerably time-bound, with approximately fourteen weeks from its genesis to its submission. Moreover, the ongoing COVID-19 pandemic created additional barriers for both students and staff during this period. Owing to this, the study was intentionally kept fairly narrow in scope.

Formally, the scope of the project centred around an implementation of the required DAB functionality—pre-processing, demodulation, and remodulation—in MATLAB. This code was to be separated into a set of functions, and submitted in conjunction with this report. Motivations for the design decisions made were to be well justified and documented. Block diagrams for each function were also to be documented, as well each function's associated component-steps and algorithms. An array of validation tests was expected to be formulated, including some tests using provided reference data. These tests were to demonstrate convincingly that the chain indeed worked.

In terms of the project's limitations, there were several. Firstly, within the DAB chain itself, some processing steps could be omitted for simplicity. These being: frequency synchronisation, and the implementation of error correction. Secondly, the DAB processing chain was not necessarily required to be *optimised*, and its efficiency, execution times, and other related metrics were not to be considered whatsoever. Thirdly, only pre-recorded data was to be used in the demodulation and remodulation chain; in contrast, data would probably be live-streamed in a practical system. Finally, the processing chain was not intended to be integrated into a real-world PR system during the course of the project. In fact, no PR processing was required to be tested at all.

1.5 Report Outline

This report begins with a broad surveillance of the relevant literature for the project, in Chapter 2. Thereafter, an unpacking of the relevant aspects of the DAB standard is presented in Chapter 3, including a detailed explanation of the theoretical concepts that underpin the core DAB functionality. With the standard then established, Chapter 4 provides a thorough exhibition of the design of the DAB processing chain, including the relevant block diagrams and sample outputs. To ensure the correctness of the designed chain, Chapter 5 outlines the validation steps that were taken for the project. After this, a short illustrative segment is provided in Chapter 6, showing how the DAB processing blocks would fit into a larger PR system. Finally, conclusions are made and recommendations provided, in Chapters 7 and Chapter 8 respectively. Appendices follow with miscellaneous information that appeared during the report.

Chapter 2

Literature Review

> If you wish to make an apple pie from scratch, you must first invent the universe.
>
> —*Carl Sagan*

This chapter aims to establish the context for this project, by considering its place within a broader literary sphere. Fundamentally, if one wanted to implement a conventional DAB processing chain in software, one would only need to consult the DAB standard document from the European Telecommunications Standards Institute (ETSI) [1], and perhaps one or two more papers for assistance. However, in doing so, one might erroneously overlook the reasons for which the chain here is being designed: integration into a PR system. Consequently, the resulting chain would likely be ill-suited for the specific needs of such a system. Instead, it is important to understand the situations in which this chain is intended to operate, and use this as a foundation for the chain's design, testing, and validation. Therefore, the surveillance of the literature done here hopes mostly to paint a backdrop for the reader, providing a general insight into the motivations for the project, and the existing work that relates to it.

The chapter begins by examining PR and digital broadcasting methods—such as DAB—in two separate sections, as independent concepts. Each concept has an interesting history, a rich theoretical foundation, a unique set of advantages and disadvantages, and deserves to be considered in isolation. With the stage then set, the integration of digital broadcasting into PR is discussed, including the motivations for such integration, and a broad overview of the work done so far in this domain. Finally, a short critique of the literature is presented, outlining the proposed value of this report.

2.1 Passive Radar

2.1.1 Brief History

The history of PR naturally begins as a history of radar itself. Like most scientific breakthroughs, it is difficult to attribute the development of radar solely to one individual or group, and frankly, it would be disingenuous to do so—it arrived via a long line of successive discoveries, sometimes even with multiple independent discoveries of the same thing, as highlighted by Brown [2]. A good starting point for this history is in 1865, when the Scottish mathematical physicist, James Maxwell, published his seminal work [3], in which the now-famous *Maxwell's equations* were first demonstrated, albeit in a verbose form. In doing so, Maxwell correctly predicted that EM waves travel through free space at the

speed of light [4]. Three decades later, through a host of experiments done between 1885 and 1889, Heinrich Hertz was able to verify these equations and, more relevantly, demonstrate the *reflection* of EM waves [5, 6], amongst other things. In the subsequent years, this work was continued by many more scientists—each of whom helped construct the stage on which radar technology could develop.

Then, in 1904, a 22-year-old German engineer named Christian Hülsmeyer filed a patent for the so-called *Telemobiloskop*—which he publicly demonstrated on at least two occasions that year [7]. This device was marketed as an anti-collision device for ships, able to detect nearby obstacles using the reflection of EM waves [8]. Interestingly, the device failed commercially, and there was little interest in it; however, despite its failure, Hülsmeyer is today regarded as the father of radar.[1] For a thorough history of Hülsmeyer and his achievements, the reader is encouraged to see [9] and [11].

Some twenty years later, after a period with no tangible developments of Hülsmeyer's work, the concept of radar was essentially rediscovered independently, on the other side of the Atlantic, in the United States. Leo Young and Hoyt Taylor, two radio engineers at the U.S. Naval Research Laboratory, noticed the reflections of high-frequency EM waves from nearby ships, as these ships moved in between a radio transmitter and receiver [2]. More time passed before further developments occurred; but, eventually, fully-fledged radar systems began to emerge.

The first appearance of a *passive* radar system, as retold by Kuschel and O'Hagan in [12], occurred in 1935, in the well-known Daventry Experiment. It was here that Sir Robert Watson-Watt and Arnold Wilkins used an existing shortwave BBC transmission for the detection of a plane flying 13 kilometres away. However, despite the early appearance of PR, interest primarily shifted to active radar developments for the following decades. That was until a renewed interest arose in the mid-1980s, such as in Griffiths et al.'s work [13]. It was here that the authors resurfaced the idea of using existing EM transmissions as so-called IOO, for PR detection. Since then, the field has further grown, and continues to grow still.

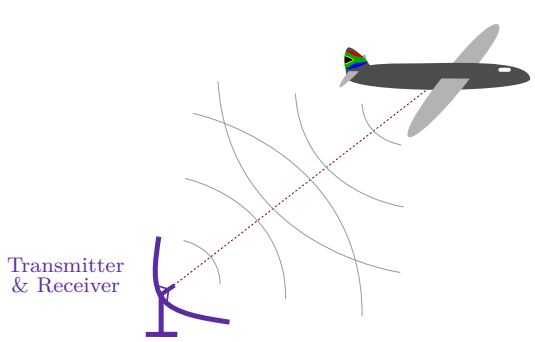
2.1.2 Radar Geometries

Radar systems can be divided into three broad categories, based on the geometry of their receiver/s and transmitter/s: *monostatic*, *bistatic*, and *multistatic*. Each of these is depicted conceptually in Figure 2.1.

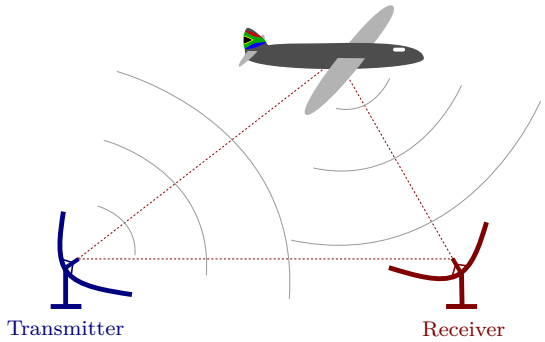
It is clear from Figure 2.1a that a monostatic geometry is the simplest of the three categories, where the transmitter is co-located with the receiver. This need not be *exactly* so, provided the distance between the transmitter and receiver is much smaller than the distance to the target. However, in most modern situations, the same antenna is used for both transmission and reception, made possible by the duplexer, invented in 1936 [12].

In contrast, the bistatic geometry, shown in Figure 2.1b, is slightly more complicated. The geometry itself is straightforward, where the receiver is simply not co-located with the transmitter; however,

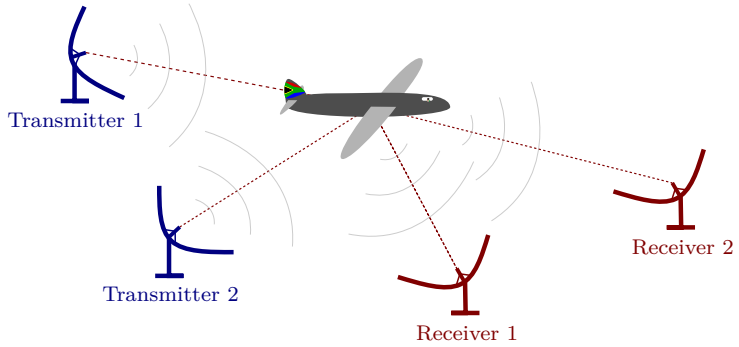
[1] Semantically, some scholars have argued that the *Telemobiloskop* was, in fact, not a RADAR—radio detection and *ranging*—device, as it could not calculate the distance to obstacles, along with some other petty concerns regarding its poor performance [9]. Nevertheless, in 2019, Hülsmeyer was officially honoured posthumously for his radar contributions, in an IEEE Historic Milestone event [10].



(a) Monostatic radar geometry



(b) Bistatic radar geometry



(c) Multistatic radar geometry

Figure 2.1: Conceptual depictions of the three broad radar categories, based on their respective transmitter-receiver geometries.

the mathematics consequently becomes more complicated. To illustrate, suppose the radar system detects that the received signal occurs t_d seconds after it was transmitted. This time delay can easily be converted to the distance that the EM wave travelled, d, using the speed of light, c, where $d \approx ct_d$. In a monostatic case, this range can be combined with a bearing—the direction in which the antenna is pointing—in order to determine the location of the object. Notice, though, in the bistatic case, knowing only the distance creates an elliptic trace of possible positions—sometimes called an *isorange contour*—with the transmitter and receiver as the two foci of the ellipse respectively. This situation can be seen in Figure 2.2, where two possible target points are shown as examples. Of course, the actual target could be anywhere on the contour.

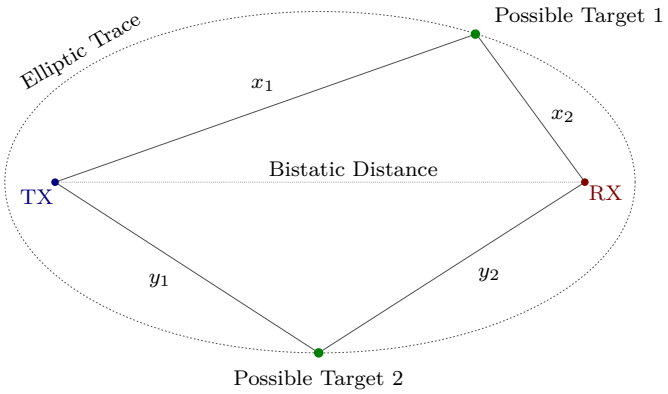


Figure 2.2: Elliptic trace of possible target positions as measured by a bistatic radar system.

Notice in the figure that $d = x_1 + x_2$ and $d = y_1 + y_2$, even though $x_1 \neq y_1$ and $x_2 \neq y_2$. This demonstrates an example of how two completely different locations can result in the same reflected time delay. As a consequence of this, bistatic radars require additional measurements in order to locate a target accurately. For example, one can use the doppler shift of the object in order to determine its velocity, and thus pinpoint its position on the elliptic trace.[2] Alternatively, multiple bistatic radar systems can be used simultaneously, and the intersection of the resultant ellipses then indicates the target's location. This latter approach is termed a *multistatic* radar, and was depicted previously in Figure 2.1c. Note that a multistatic arrangement is loosely defined, and may actually comprise multiple bistatic and/or monostatic radar systems.

Intuitively, in PR situations, it is highly unlikely that the IOO's transmitter will be co-located with the system's receiver; thus, such systems are almost always at least bistatic, or are otherwise multistatic. Interestingly, Kuschel and O'Hagan have suggested that the convenience and simplicity of monostatic radars—enabled largely by the invention of the duplexer—shifted focus away from PR technology in the early years of radar [12].

2.1.3 Advantages & Disadvantages of Passive Radar

The use of PR is attractive for a variety of reasons, hence the renewed interest therein over the past three decades. Many scholars have discussed these advantages comprehensively, and they are summarised nicely by O'Hagan [15], whose reasons will be briefly mentioned here. Fundamentally, a PR system is advantageous because, by definition, it does not use or need its own transmitter—it simply hitchhikes off existing transmissions. Several benefits arise from this simple fact.

Firstly, transmitting EM signals requires energy, and such energy is not free. A useful active radar transmitter may need to transmit kilowatts of power when operating, which naturally has an associated electrical cost. Licensing for the usage of the EM spectrum is also not cheap—especially not in recent years, where there has been increasingly-fierce competition and political lobbying for spectrum allocation. Moreover, the antennas used for these transmissions are often required to be bulky. Therefore, a PR system can be both cheaper to operate and maintain, and more compact than an active counterpart.

In a military context, the lack of a transmitter also enables *covertness* of the system. An active radar can be easily detected by an enemy via the radiation that is emitted, and thus puts it at risk of sabotage by Electronic Countermeasures (ECM) or otherwise. A PR system, on the other hand, is far more difficult to detect, and is therefore less susceptible to such attacks. Furthermore, O'Hagan points out that the use of the VHF frequency band—which is not allocated for radar purposes—in a PR system enables certain counter-stealth benefits for so-called *low-observable* targets [15].

Another interesting advantage is demonstrated in a real-world use-case. Peralex Electronics, along with work done by Paine et al., deployed a PR system at one of the Square Kilometer Array (SKA) sites in South Africa [16]. The SKA is a massive radio-telescope, which is so extremely sensitive that the zones surrounding it need to be electromagnetically 'clean' within the telescope's operating bands. Unfortunately, small aircraft regularly fly over the site, thereby introducing EM interference from their

[2] For a thorough description of the signal processing required for Passive Bistatic Radar systems, see [14].

weather radars, transponders, and other electrical equipment. It is vital that the telescope does not operate during these instances. However, using an *active* radar system to detect these aircraft would be pointless, as the radar system itself would then interfere with the telescope. The solution here was to use a PR system that hitchhikes on surrounding FM broadcasts. Since the FM spectrum falls below the telescope's operating band, the radar can detect incoming aircraft, while the telescope continues to operate normally.

Naturally, some disadvantages exist for PR systems, which are also discussed in [15]. Ideally, the waveforms used in a radar system's transmitter should have certain characteristics that facilitate detection in its receiver. But, in a PR context, one has no control over the content that is broadcast on a particular IOO, possibly making the system less reliable, depending on the IOO used. Furthermore, PR systems can also suffer from Direct Signal Interference (DSI)—a scenario in which the signal travelling directly from the transmitter to the receiver overpowers the reflected signals from the scene, thereby limiting the radar's 'sensitivity'.

It is also important to note that PR systems do not have outright *immunity* against ECM. In fact, there is ongoing research into the possible electronic attacks that can be taken against such a system. For example, Giusti et al. have looked at jamming Orthogonal Frequency Division Multiplexing (OFDM)-based signals [17], and Schüpach and Böniger have experimented in specifically jamming DAB signals [18], both in PR contexts. For a broad overview of this topic, see [19] by O'Hagan et al. For a more comprehensive look at applying ECM specifically to FM and Digital Video Broadcasting - Second Generation Terrestrial (DVB-T2) signals, see Paine's work in [20].

2.2 Digital Broadcasting

When considering the innumerable technological advancements of the past century, one cannot overlook the enormous impact of wireless broadcasting. The advent of mass communication, first via radio and later via television, has undoubtedly shaped our culture, our politics, and frankly, our entire lives. *Digital* broadcasting, in contrast to *analogue* broadcasting, involves the use of digital signals for the modulation of the carrier waves in a broadcast. Of course, the EM signals that are actually transmitted are themselves analogue in nature, but the information encoded in them is discretized to a finite set of possible values.

Understanding the development of digital broadcasting—why it has or has not been adopted in its various instances—is an important question in the context of PR, for the following reason: a PR system that uses a digital broadcast as an IOO relies entirely on that digital broadcast existing in perpetuity. When a particular broadcasting standard ceases to be relevant, the systems designed around that standard become useless. Therefore, it is vital to gauge the availability of a particular standard to contemplate the development effort required to integrate it into a practical PR system.

This section attempts to show briefly the history of and motivations for digital broadcasting. Since this report focuses specifically on DAB, the entire chapter following this one is dedicated to the *theory* behind the DAB standard and its associated concepts. To avoid needless repetition, these specific concepts will not be covered comprehensively here. Instead, digital broadcasting will be considered in

a generic sense.

2.2.1 Brief History

Around the same time that Hülsmeyer et al. were experimenting with early radar systems, there was development occurring in wireless *audio* transmissions. For example, in the final days of 1906, Reginald Fessenden successfully demonstrated the first wireless broadcast by wishing a Merry Christmas to nearby ships of the U.S. Navy, as Belrose reminisces in [21]. In the decades that followed, radio broadcasting became a staple of society.

For most of the twentieth century, radio was operated solely via analogue broadcasting techniques, firstly with Amplitude Modulation (AM), and later with FM too. The theoretical foundations for digital broadcasting did exist for some time, arguably beginning with Chang's seminal work on OFDM in 1966 [22], followed by Weinstein and Ebert's insight into using the Discrete Fourier Transform (DFT) for OFDM in 1971 [23]. However, the computational power of the computers at the time was simply inadequate for the demands of a digital modulation scheme. Nonetheless, as computers matured, the scene was set to change; investigations into digital broadcasting techniques could begin. For example, in 1985, McNally wrote an article entitled, 'Digital Audio in Broadcasting,' for the IEEE's Acoustics, Speech, and Signal Processing magazine [24]. Two years later, Alard and Lassalle [25, 26] discussed the modulation and channel coding requirements for an OFDM system; and in 1988, Weck and Theile [27] laid further foundations with their proposal to use convolutional codes and Viterbi decoding in such contexts. Just before the end of the decade, Rault et al. [28] then released their work on Coded Orthogonal Frequency Division Multiplexing (COFDM), which enabled robust mitigation of multipath effects in a mobile digital receiver.

Naturally, these developments spurred on a flurry of research into digital broadcasting, for both television [29, 30], and audio [31, 32, 33] purposes. Around the same time, work was being done on the *Eureka 147* project [34] in Europe, which later became the DAB standard—the focus of this report. In the years that followed, many more concepts, designs, and possible architectures were released for digital broadcasting technology. For a richer history of digital radio, specifically looking at DAB, see [35]; for a history of digital television, see [36].

2.2.2 Advantages & Disadvantages of Digital Broadcasting

The benefits and drawbacks of digital broadcasting are somewhat dependent on the particular use-case in question. Nonetheless, there are a handful of consistent advantages when using a digital scheme compared to an analogue equivalent. Gandy [37], and Witherow and Laven [38]—both on behalf of the BBC—highlight some of the upsides of a DAB system, though many of these can be extended to other modes of digital broadcasting too. A handful of them will be highlighted below.

One of the most useful features of digital broadcasting is the spectral efficiency that it can achieve. As mentioned previously in the motivations for PR, the EM spectrum is a fiercely contested resource. By enabling more content to be transmitted using the same bandwidth, these broadcasting techniques can ease the pressure on regulators, bring down the cost per programme, and ultimately, provide more

options for the consumer. For example, Gandy considers the situation in the United Kingdom, stating that *six* audio channels in a DAB broadcast can fit into the bandwidth of a single FM broadcast. Note, too, that the actual *quality* of the digital signals is also far superior. DAB can achieve high-quality audio, with a fidelity comparable to the compact disc. Digital television, too, can achieve high-definition broadcasts.

Digital broadcasts are also more robust in the presence of multipath—which is a massive challenge in FM radio. Various error-correction techniques and other protective mechanisms—such as guard intervals in OFDM transmissions—can be used with great success. This further enables the use of Single-Frequency Networks (SFN), where nearby transmitters can use the same carrier frequencies for their transmissions. Consequently, these broadcasts can be even more efficient when considering the spectra and power they require.

Finally, digital broadcasting also offers more functionality to the end-user. A DAB signal, for example, can include a multitude of media types other than audio, including basic graphics, textual information for traffic or station updates, and so on [1].

Despite the many benefits of digital broadcasting systems—of which only some were mentioned—analogue broadcasting systems remain popular around the world. This reality is less about the *disadvantages* of moving towards digital broadcasting, and more about the practicalities of doing so. Since analogue broadcasting has existed for many decades, the receivers for AM, FM, and similar transmissions remain cheaply and widely available. Accordingly, the transition from analogue to digital requires an enormous amount of political will and regulatory effort, as demonstrated by South Africa's failed switch-over. Here, the *Broadcasting Digital Migration Policy* was issued by the Department of Communications in 2008 [39], where the Independent Communications Authority of South Africa (ICASA) was tasked with rolling-out digital television, such that there would be an 'analogue switch-off' in 2011. As of late-2020, after countless delays, this has yet to occur.

2.3 Digital Broadcasting-based Passive Radar

With the foundations of both PR and digital broadcasting presented, it is now interesting to look at the use of the latter in the former—that is, the use of digital broadcasting signals as the IOOs for a PR system.

2.3.1 Motivation

When Griffiths et al. first resurfaced the idea of PR, their work was done using analogue television signals—both terrestrial [13] and satellite-borne [40]—as the IOOs. As the field grew, however, the majority of the research focused on using FM signals instead. This makes sense, considering the ubiquity of FM radio across the globe. Consequently, much of the existing work has been done in this domain. For a straightforward example, consider [41]; for a more thorough elucidation, see the dissertations prepared by O'Hagan [15], Brown [42], and Tong [43] respectively. The present state of

FM-based PR is fairly advanced, and some authors have even demonstrated successful systems that solely use readily-available, 'commercial off-the-shelf' components, such as in [44] and [45].

Using FM signals as IOOs has some downsides, though. It can be shown that the range resolution of a radar system is a function of the bandwidth of the signal used for detection. In turn, since the bandwidth of an FM signal is time-variant—it is a function of its programme content—the range resolution of a FM-based PR system is also time-variant. This naturally results in unreliable system performance [46]. Moreover, O'Hagan [41] points out that FM-based systems still have to overcome the major problem of DSI. While various suppression techniques do exist, the FM signal itself offers no assistance.

Digital signals, on the other hand, have superior performance in these dimensions where FM signals fall short. For example, COFDM-based signals—such as DAB and DVB-T2—have time-*invariant* bandwidths, regardless of the instantaneous programme content that they are transmitting. Consequently, such signals facilitate a reliable range-resolution when used as IOOs for PR [46]. Griffiths and Baker [47] have also shown definitively, for similar reasons, that the ambiguity functions for digitally-modulated signals are far more auspicious than those for analogue signals. This implies that such signals are well-suited for radar applications. Specific analysis of the ambiguity functions for OFDM signals has also been done in [48].

Furthermore, digital broadcasts have another important advantage. Since digital broadcasts usually adhere to an openly-accessible standard—such as ETSI's DAB standard [1]—a received signal can be demodulated, and then *remodulated*, thereby creating a perfect reconstruction of the received signal—without noise and interference. In doing so, the requirements for the suppression of DSI become less stringent. This approach was put forward in [46], and is also discussed in [49].

This 'perfect reconstruction' method has also been taken one step further. A conventional PR system requires two signals: a *reference* signal, which is a measurement aimed directly at the original transmitter, and a *surveillance* signal, which is a measurement aimed at the scene to record the various reflections of the transmitted signal. Usually, these signals must be recorded by two independent antennas. However, as proposed by Searle et al. [50], digital broadcasting-based PR systems can generate the reference data *from* the surveillance data, by reconstructing it perfectly.[3] This is massively advantageous, and forms the central motivation for this project.

2.3.2 Existing Implementations

There have been many implementations of digital broadcasting-based PR described in the literature, beginning with Poullin's work [52] back in 2005, which dealt with the use of COFDM transmissions in PR. Since then, numerous papers have used digital *television* signals as IOOs, probably because of the widespread adoption of digital television around the world. Earlier works used the older Digital Video Broadcasting - Terrestrial (DVB-T) standard, such as in [53, 54, 55]; but, more recently, the newer DVB-T2 standard has also been used [56, 57].

[3] This approach was also put forward in [51], where the Advanced Television Systems Committee (ATSC) standard—a non-OFDM digital television approach—was used.

There have been implementations using DAB too, such as from Yardley and Coleman [58, 59]. More recently, Schüpach et al. have taken on various DAB-based PR projects [60, 45].

Some progress has also been made using *multi-illuminator* systems, where several IOOs are used simultaneously, to leverage the benefits of multiple approaches at once. For example, Daun and Koch [61] tested a multistatic approach that used DAB and DVB-T signals together. See also [62] and [16], for work done using FM and DVB-T2 together.

2.4 Literature Critique

It has been clear from this chapter that a comprehensive amount of literature exists on the topics of PR, digital broadcasting, and the integration of the two. Much has already been written about the motivations for these fields and their theoretical underpinnings. In the case of DAB, papers and books have demonstrated successful receiver architectures, outlined audio extraction techniques, and considered error-correction thoroughly. In the case of PR, many real-world tests have been done that consider accuracy, efficiency, reliability, and more. Furthermore, DAB-based PR systems have also been demonstrated and tested.

However, the literature lacks a thorough exposition of the processing chain required for the use of DAB signals in PR. Existing PR work omits many details regarding the design choices made for such systems, and existing DAB work focuses too much on the data perspective—the playback of audio, the processing of incoming traffic information, etc. This report, then, hopes to unpack the DAB standard insofar as a PR context would require, and thereafter, detail the processing chain design for DAB-based PR applications.

Chapter 3

Digital Audio Broadcasting: Standard

> Where the waters do agree, it is quite wonderful the relief they give.
>
> —*Jane Austen, Emma*

3.1 Overview

This chapter aims to outline the salient aspects of the DAB standard, as prescribed by the ETSI in [1]. Fundamentally, the DAB standard was built upon two important communication technologies: COFDM, and Differential Quadrature Phase-Shift Keying (DQPSK). Without the innovations witnessed in these domains in the latter half of the twentieth century, it is unlikely that the DAB system could have been adequately designed, while maintaining a suitable degree of spectral efficiency and information robustness. Each of these concepts will thus be thoroughly discussed in the following sections. For the sake of clarity, the provided explanations attempt to be as graphical as possible, leaving out the sometimes-obfuscating mathematical details. Thereafter, the DAB frame structure will be outlined, with the key regions highlighted where appropriate.

Note, however, that this project focused on DAB signals within the context of PR; therefore, it was unnecessary to provide a complete and thorough description of the DAB format itself. Instead, information was considered only insofar as it was relevant to the broader picture of a PR processing chain. As a consequence, the details surrounding DAB's audio coding, configuration information, and other data features were omitted for simplicity.

3.2 Coded Orthogonal Frequency Division Multiplexing

COFDM is the first pillar upon which the DAB standard was built. As its name suggests, this technology is a special case of OFDM, which itself is a special case of Frequency Division Multiplexing (FDM). Each of these concepts will be explored in the coming text, along with other important considerations surrounding the implementation of COFDM, starting with the motivation for it. The reader is also encouraged to see [28] and [63] for a deeper exploration of COFDM.

3.2.1 Motivation: The Problem of Multipath

The DAB standard was designed to replace the conventional FM and AM analogue modes of broadcasting, with the intention of audio and data reception in both fixed and mobile environments [1]. Of these two listening contexts, the more common one was—and still is—certainly the latter. Despite living in an era saturated by computers and the internet, radio has remained a staple for many people in their cars—enabling them to listen to content while commuting. The reception environment for a DAB signal was thus known beforehand as one that would be highly dynamic and somewhat unpredictable. This situation was made worse by the reality that many receivers would be in urban centres, which contain numerous high-rise buildings and other 'clutter.'

To understand the challenges of such environments, consider the following simple scenario: suppose one transmitted a set of symbols, $\mathcal{D}$, using an arbitrary digital modulation scheme with a carrier frequency ω_0, to a mobile receiver in an urban setting. Suppose too that each of the symbols modulated the carrier wave in succession for a finite period, t_s, and call this the 'symbol period'. Once modulated, the carrier wave would be transmitted by a stationary antenna and travel through the environment, at times bouncing off surrounding objects, such as buildings, cars, mountains, and trees. Provided there was a direct path from the transmitter to the receiver, the signal—the 'direct signal'—would travel along this path. Additionally, due to the surrounding clutter, duplicate versions of the signal—'delayed signals'—would also arrive at the receiver. This effect is called *multipath*. Figure 3.1 illustrates this scenario with dashed lines indicating the various signal paths.

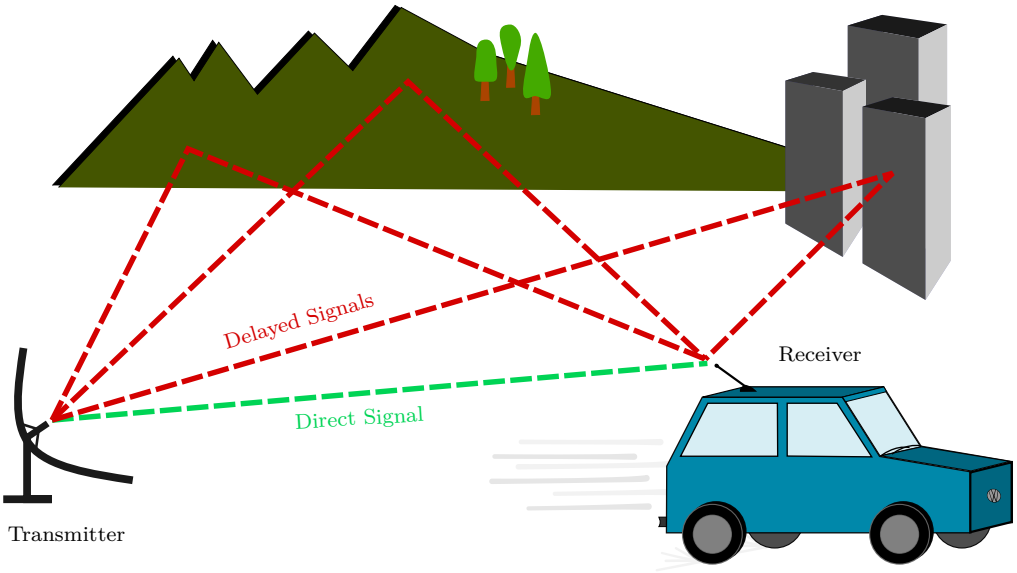


Figure 3.1: Illustration of an environment where multipath occurs

By definition, the direct signal would take the shortest path from the transmitter to the receiver, and thus the time to receive this signal would be shorter than for all the others—it would 'arrive' first at the receiver. The delayed signals, each taking a longer route than the direct path, would be received later, each after a certain amount of time. The number of delayed copies that were received, and how long these delays were, would depend on the clutter contained within the scene. Importantly, the receiving antenna would *not* be able simply to separate the direct signal and the delayed signals;

rather, it would record a superposition of them, as a single 'received signal.'

For the sake of simplicity, suppose only one delayed signal is received, with a delay period of t_d. Consider the two possible situations that could arise in such a scenario for the received signal—the sum of the direct and delayed signals. Firstly, the delay period could be *shorter* than the symbol period; that is, $t_d < t_s$. Figure 3.2 depicts such a situation graphically, with three symbols—blue, pink, green—used for illustration.

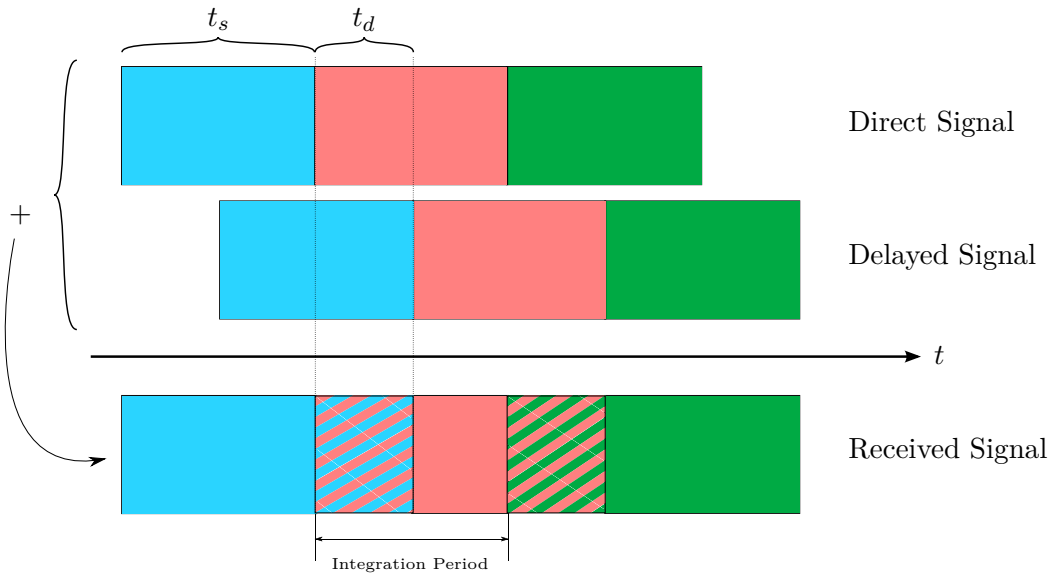


Figure 3.2: Multipath situation with $t_d < t_s$

For the receiver's demodulation system to extract the information of an incoming signal, it must measure the data over a particular 'integration period'—in this case, equal to the symbol length, t_s. Notice in the figure how the extraction of the second, pink symbol from the direct signal is affected by the presence of the delayed signal. For the first half of the integration period for this symbol, the blue and pink symbols overlap, causing Intersymbol Interference (ISI)—depicted as a pattern of striped blue and pink in the received signal. The severity of this interference depends on the magnitude of the delayed signal, compared to the magnitude of the original signal. In any case, recovering the correct information from this region becomes unreliable. Fortunately, though, since the symbol period is longer than the delay period, some of the original, direct, pink symbol symbol is superimposed with some of the pink symbol in the delayed signal, creating a 'pink-only' region within the integration period. While the demodulator is not guaranteed to extract this symbol successfully—since some of the integration period is still corrupted—it remains possible. Naturally, the less of the symbol period that is corrupted, the more likely correct demodulation will occur. An additional strategy for improving these chances is by adding a so-called 'guard interval,' which will be covered in a later section.

In contrast, consider the second situation for the received signal, where the delay period is *longer* than the symbol period; that is, $t_d \geq t_s$. An illustration for this situation is provided in Figure 3.3, with the same three coloured symbols shown.

In this case, the entire pink symbol from the direct signal is superimposed with the entire blue symbol from the delayed signal. The result of this is a completely corrupted symbol in the received signal,

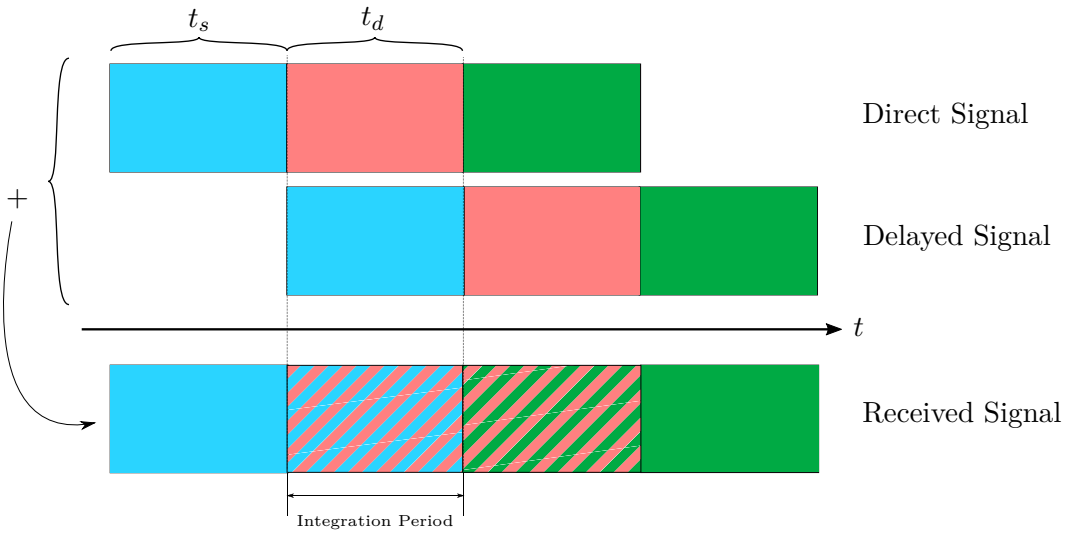


Figure 3.3: Multipath situation with $t_d \geq t_s$

over the whole integration period. Consequently, the original information in this symbol cannot be recovered reliably.

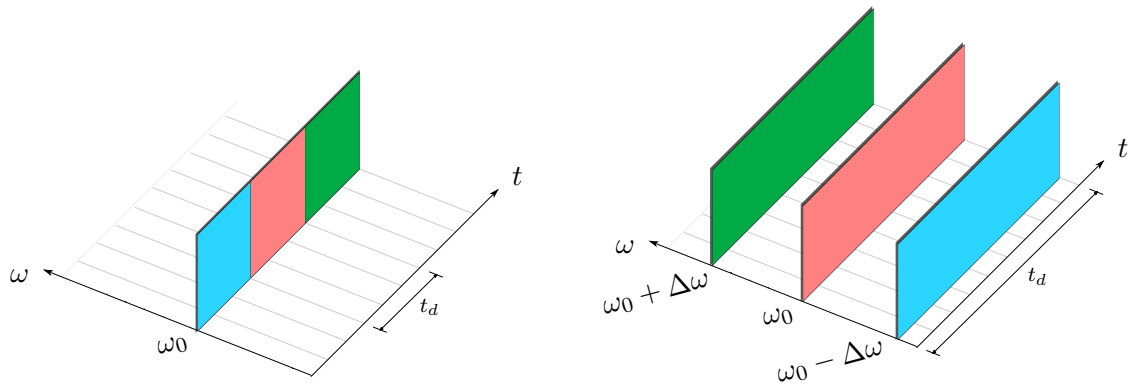
From this example, it is clear that the symbol length, t_s, determines the maximum allowable multipath delay, t_d. It follows that for reliable communication in mobile and urban environments, for which the DAB standard was primarily designed, a longer symbol length is preferred—equivalently, a lower data-rate. The longer the symbol—i.e. the slower the data-rate—the more robust a receiver system will be to the effects of multipath

3.2.2 Frequency Division Multiplexing

The discussion from the previous example assumed a 'serial' approach, where a *single* carrier wave was modulated with successive elements of $\mathcal{D}$. Instead of this, one could take a 'parallel' approach, by splitting $\mathcal{D}$ into n subsets, and then using these subsets to modulate n independent carriers respectively—sometimes called 'sub-carriers'. As a result, each symbol could be n times longer—increasing its multipath resilience—while maintaining the original, faster data-rate. This approach is termed FDM[1]. Figure 3.4 illustrates these situations with two simplified spectrograms. For the sake of clarity, the frequency components are shown to be perfectly impulsive, and no modulation parameter is depicted.

Figure 3.4a shows the serial case, where all three symbols—blue, pink, and green—are successively modulated (in some unspecified way) on the carrier wave with a frequency of ω_0. Figure 3.4b, on the other hand, shows the parallel case, with the three symbols split into three separate groups, respectively modulating sub-carriers with frequencies of $(\omega_0 - \Delta\omega)$, ω_0, and $(\omega_0 + \Delta\omega)$. Notice the symbol period, t_d, for the latter situation is three times longer than that of the former. Nonetheless, it is clear from these

[1]Strictly speaking, FDM is the fairly general concept of sharing a communication channel between multiple frequency bands—including, for example, transmitting multiple FM radio stations at different centre frequencies simultaneously. On the other hand, transmitting multiple *bitstreams* by multiplexing them in the frequency domain is sometimes specifically termed Multicarrier Modulation (MCM) [64]—a special case of FDM. Nonetheless, since the DAB standard simply refers to the concept as FDM, the distinction will not be further considered.



(a) Single carrier with a shorter symbol period.

(b) Multiple carriers with a longer symbol period.

Figure 3.4: Simplified spectrograms for the two approaches to transmitting a set of symbols, $\mathcal{D}$.

plots that in each scenario, the same amount of data is transmitted over the same amount of time—i.e. the data-rate is the same. Therefore, the FDM signal can be more resilient to multipath effects, while not compromising on the rate of transmission of the data.

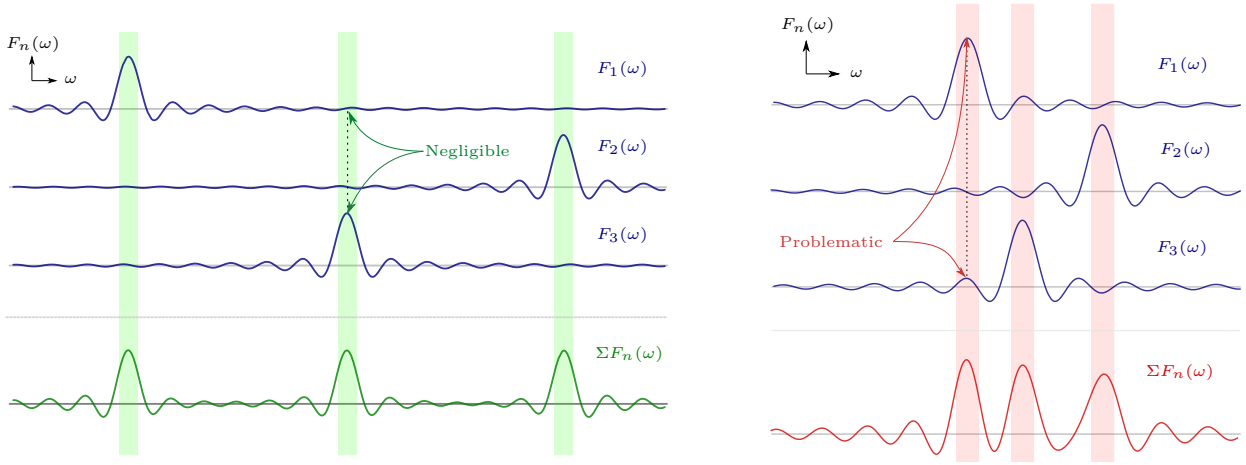
However, there is an important caveat to note with this example—one which presents the tradeoff that exists when using FDM. The frequency components, as shown in the example, were assumed to be impulsive. This was done for the sake of simplicity, to convey the advantage of using multiple sub-carriers instead of only one carrier. Yet, this assumption implies that adjacent sub-carriers can be infinitely close to each other without any crosstalk—that is, without interference between them. If this were true, the increased bandwidth requirements for an FDM signal, compared to a single carrier wave, would be negligible. Unfortunately, this cannot be the case when considering the time-domain signals over a finite integration period. Instead, each sub-carrier has an associated *spectrum*—which largely depends on the window function used in the system. For example, the straightforward rectangular window used in this project, $w(t)$, with an integration period of T_u, has the following spectrum in the frequency domain:

$$w(t) = \text{rect}\left(\frac{t}{T_u}\right) \overset{\mathcal{F}}{\Longleftrightarrow} W(\omega) = T_u \cdot \text{Sa}\left(\frac{\omega T_u}{2}\right) \tag{3.1}$$

where $\text{Sa}(x) = \frac{\sin x}{x}$. Since multiplication with this function in time is equivalent to convolution with its spectrum in frequency, the previously-impulsive frequency components for the FDM signal were, in fact, shaped like $\text{Sa}(\omega)$ functions. This is important because such spectra have large sidelobes, which can potentially cause interference between adjacent sub-carriers.

For example, suppose one uses three carrier waves, each with a $\text{Sa}(\omega)$ spectrum, in an FDM system. If the centre frequencies of these carriers are sufficiently distant from each other, there will be negligible interference from one spectrum's sidelobes to another spectrum's main lobe, and there will be minimal crosstalk. This is depicted in Figure 3.5a. However, if the carriers are placed too close to each other in frequency—as depicted in Figure 3.5b—such interference will no longer be negligible, and the robustness of the system will degrade.

Clearly, for a reliable system using FDM, the sub-carriers must be sufficiently spaced from each other in the frequency domain. Consequently, there exists another tradeoff. As shown previously, if the number of sub-carriers increases, the system's resilience to multipath will improve; yet, as the number of



(a) Three spectra spaced sufficiently far away from each other

(b) Three spectra spaced insufficiently far away from each other

Figure 3.5: Depiction of two possible FDM situations.

sub-carriers increases, the total bandwidth required for the FDM system similarly increases. Since the carriers need to be placed far apart in frequency, these bandwidth requirements can become untenable for a large number of sub-carriers.

3.2.3 Using Orthogonal Carriers

FDM is not a particularly efficient mode of transmitting parallel streams of data, because of the aforementioned bandwidth limits. However, in 1966, Robert W. Chang published an incredibly important paper [22], outlining the idea of so-called OFDM—a special case of FDM using 'orthogonal' carrier waves. In essence, by choosing specific sub-carrier frequencies for a system, one can reduce the total bandwidth requirements massively, and eliminate the concern of crosstalk between sub-carriers. For a given integration period of T_u, the requirement is that the spacing between adjacent carrier waves, $\Delta\omega$, must be set to

$$\Delta\omega = \frac{2\pi}{T_u} \tag{3.2}$$

The reason that the sub-carriers can be placed so close together, without crosstalk, becomes evident when viewed graphically. Consider the plot for three $\mathrm{Sa}(\omega)$ spectra arranged in this way, as shown in Figure 3.6.

Notice how the peak of each spectrum, occurring at the respective carrier frequencies of ω and $\omega \pm \Delta\omega$, is perfectly aligned with the zero crossings of the other spectra. This relationship results in zero cross-talk at those frequencies, despite the spectra being packed together tightly. Mathematically, these signals are said to be 'orthogonal' over the integration period—for a detailed proof of this fact, see Appendix B.

Therefore, by arranging the carrier frequencies in this specific way, many sub-carriers can be used within a limited bandwidth, creating an efficient method of transmitting bitstreams in parallel. A collection of such sub-carriers over a single integration period is usually termed an OFDM 'symbol.'

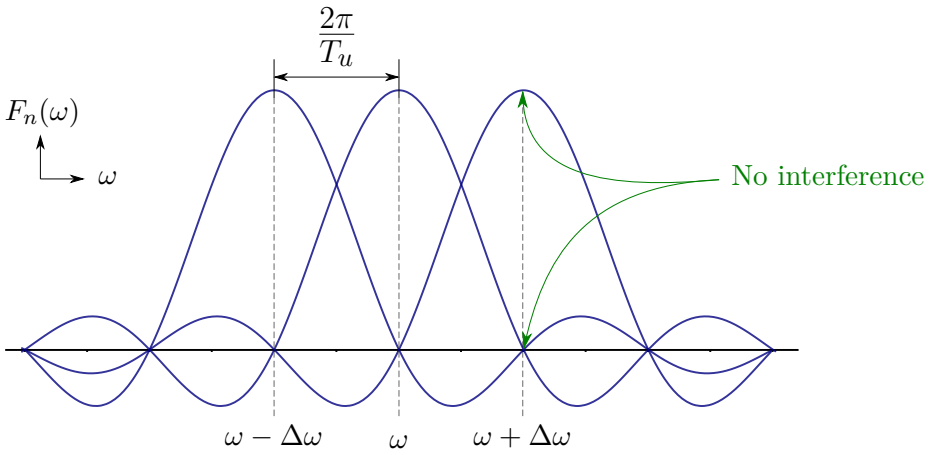


Figure 3.6: Plot showing three $\mathrm{Sa}(\omega)$ spectra arranged orthogonally.

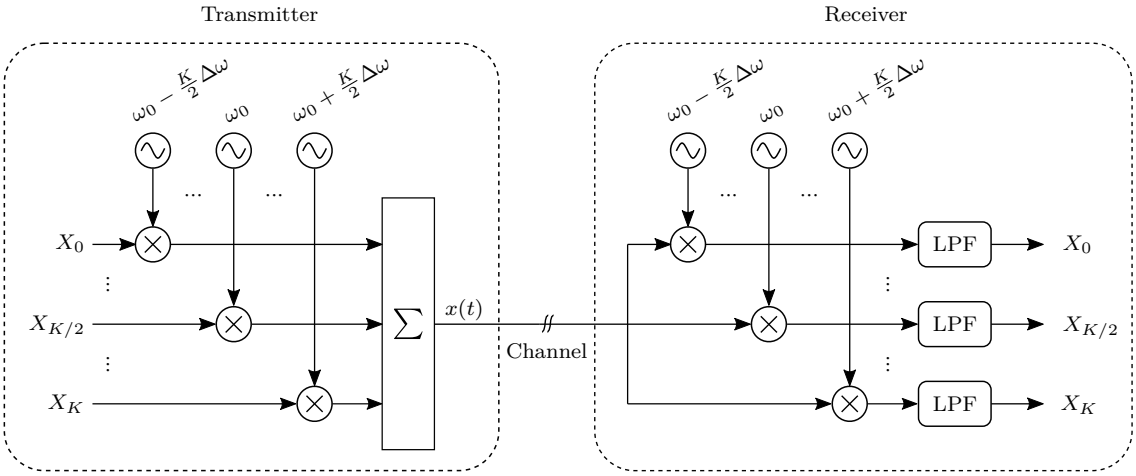


Figure 3.7: Depiction of a multiplier approach to OFDM.

3.2.4 Leveraging the Discrete Fourier Transform

While the OFDM method is a conceptually clever one, it could not be easily adopted for some time due to major practical constraints. Suppose one had K independent bitstreams to be modulated onto K sub-carriers arranged orthogonally, in an OFDM transmission. A naïve approach to this problem would involve using K independent oscillators—with each one tuned uniquely to one of the sub-carrier frequencies—and K independent multipliers. On the receiving end, a similar set-up would be required, along with a bank of narrowband low-pass filters. Such a set-up is depicted in Figure 3.7.

While this approach would indeed work theoretically, it would be expensive and impractical—the number of possible OFDM sub-carriers used would depend on both the money and space available for the transmitters and receivers. As a consequence, such systems are limited to a small number of sub-carriers, making them fairly useless for anything that requires both a high data-rate and resilience to multipath effects.

Fortunately, in [23], Weinstein and Ebert demonstrated how one could perform the same functionality

as shown in Figure 3.7, using only the DFT algorithm. By using the more-efficient DFT-equivalent, the Fast Fourier Transform (FFT) algorithm, one can implement a system that uses a large number of sub-carriers—for example, over one thousand sub-carriers in the DAB standard—yet remains small, efficient, and affordable.

3.2.5 Cyclic Prefixing & Guard Intervals

As discussed previously, the use of a longer symbol period enables a more robust signal reception process within a multipath environment. Furthermore, the use of OFDM symbols enables an additional protective measure: that being, the implementation of so-called 'guard intervals.' A guard interval is a section of a given symbol that is prepended to the symbol itself, via a process called 'cyclic prefixing.' This is illustrated in Figure 3.8.

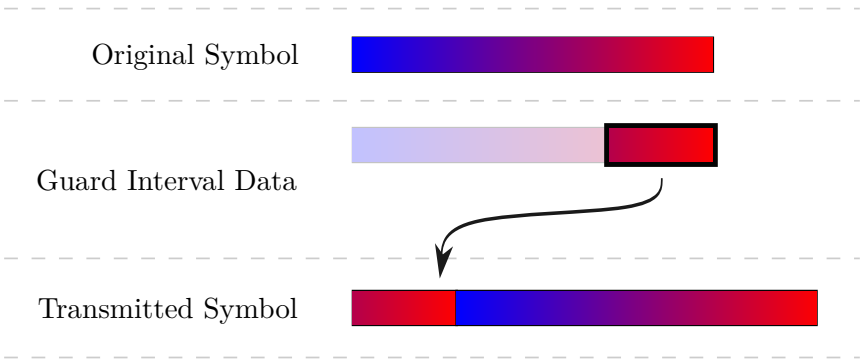


Figure 3.8: Illustration of the creation of a guard interval via cyclic prefixing.

There are several things to note here. Firstly, notice that the guard interval data is defined from the *ending* of the original symbol. That is, a guard interval with a period of T_g will be equal to the last T_g time-units of its original symbol. Secondly, observe that when this guard interval is prepended to the original symbol, the order of the data is maintained—this is demonstrated by the gradient shown in the illustration. Together these approaches are termed 'cyclic prefixing.' Thirdly, one can clearly see that the transmitted symbol is longer than the original symbol.

Importantly, though, the integration period—the time over which the symbol is considered—remains the same, equal to the length of the original symbol, despite the transmitted symbol's additional length. To understand why this is the case, and why the inclusion of a guard interval is helpful, consider the graphic shown in Figure 3.9.

This illustration shows two received signals in a multipath situation—named, as before, the 'direct' and 'delayed' signals—one of which is received slightly later than the other. Two symbols are shown, p and q, together with their delayed equivalents, $\tilde{p}$ and $\tilde{q}$, and the guard intervals are indicated with Δ characters[2]. Observe that the integration period spans the original symbols exactly. Importantly, notice that the inclusion of guard intervals has ensured that the first symbol of the delayed signal, $\tilde{p}$, does not interfere with the second symbol of the direct signal, q. In other words, ISI has been avoided.

[2]The Δ character is used to notate the duration of the guard interval in the DAB standard [1].

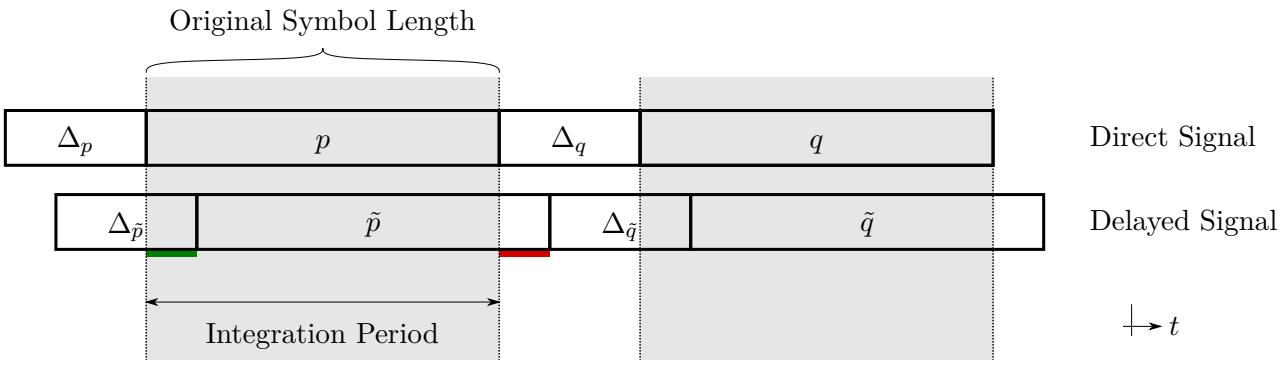


Figure 3.9: Depiction of the usefulness of a guard interval.

Additionally, recall that the guard interval is defined via cyclic prefixing. As a consequence of this, sections of a delayed symbol that fall outside of an integration period will be in the guard interval. For example, consider the chunk of $\tilde{p}$ that falls outside of the integration period—underlined in red. Because of the guard interval, this chunk is identical to the chunk underlined in green, which *is* included in the integration period. This result ensures that the guard interval of the delayed signal, $\Delta_{\tilde{p}}$, does not interfere with the original symbol, p.

Of course, the mere inclusion of the end chunk of the delayed symbol (the red section) in the integration period (as the green section) does not necessarily mean that the net received signal, the sum of the direct and delayed signals, will be correctly demodulated. This is where another strength of OFDM lies: since all of the carrier waves are, by definition, orthogonal over the symbol period, it can be shown that cyclicly shifted versions of a symbol will yield the same OFDM results when demodulated, except for a potential phase-shift. The mathematical details of this fact are unimportant for this context, but the point remains.

Guard intervals are hence a powerful mechanism to mitigate the effects of multipath. Once again, though, there exists a tradeoff. For ISI to be avoided, the maximum delay period between signals must be less than the guard interval period. Thus, increased robustness via an increased guard interval length has a cost of a decreased data-rate, since a longer guard interval means less actual data transmitted per unit of time.

3.2.6 Frequency Interleaving

There is another challenge to overcome in multipath situations that has not yet been discussed. When signals travel along different paths to a receiver, the superposition thereof could occur in various ways. Consider the two extreme cases: firstly, the signals could arrive in phase with each other, and thus add purely 'constructively'—resulting in a larger received signal; alternatively, the signals could arrive out of phase with each other, and add purely 'destructively'—resulting in no received signal. In reality, some intermediate situation would likely occur, where the received signal is either somewhat attenuated, or somewhat amplified.

Whatever the case may be, the nature of this observed superposition is clearly dependent on the *phases* of the waves received. Because a wave is made up of many frequency components, a particular

multipath-prone scene may cause a received signal to be amplified at some frequencies, and attenuated at others. The latter phenomenon is called *frequency selective fading*, and can prove to be problematic for OFDM-based systems. Naturally, this fading will often occur in localized regions of a particular spectrum—since nearby frequencies will have similar phases. Consequently, in the case of an OFDM symbol, the fading can cause clusters of sub-carriers to be degraded simultaneously. Figure 3.10 illustrates such an effect.

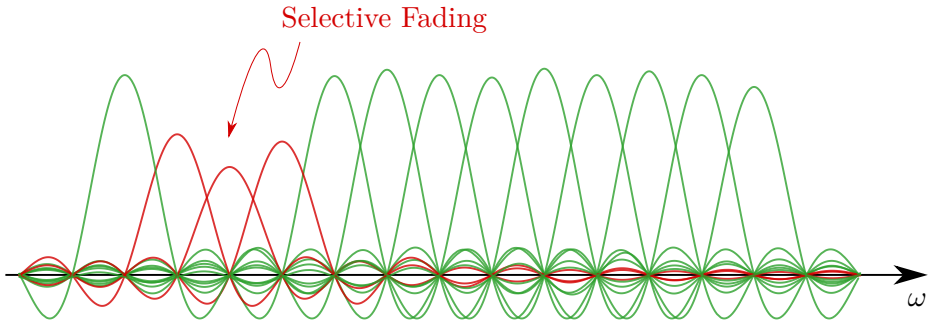


Figure 3.10: Illustration of frequency selective fading in an OFDM spectrum.

While the selective fading of sub-carriers cannot itself be completely avoided, one *can* reduce its impact on the overall system performance. Ideally, if errors are going to arise in the transmission of a signal, it is better for these errors to occur randomly. This is partly motivated by the nature of the error correction algorithm used in COFDM. Since selective fading would consistently affect certain clusters of carriers, causing repeated errors in certain bitstreams, the idea of 'frequency interleaving' is introduced.

Essentially, before the data is modulated for transmission, it is distributed—i.e. interleaved—amongst the OFDM sub-carriers. For example, the first data point might end up modulating the 100th carrier, while the 100th data point might modulate the 60th carrier, and so on. A specific, known mapping is used to interleave the data, the details for which are given in the DAB standard. Each DAB transmission mode has a unique mapping. The inverse process of deinterleaving can then applied at the receiver's end, undoing the effects of interleaving, and thus extracting the original bitstreams. In doing this, the errors caused by frequency selective fading will be less localized and more sporadic in the signal—making the system more robust.

3.2.7 Forward Error Correction

The final consideration for COFDM is the 'coded' part of *coded*-OFDM. In essence, this approach is almost equivalent to OFDM, except that the data is convolutionally encoded before modulation. Redundant data is consequently added to the signal—a method which is broadly termed 'Forward Error Correction.' At the receiver's end, a Viterbi decoder can then be used to predict the most likely bits that were transmitted, thus hopefully correcting any errors that arose during transmission.

The scope of this project did not enable a thorough analysis or design of the error coding methods used in COFDM. It is included here for completeness' sake, and further investigation is certainly recommended.

3.3 Differential Quadrature Phase-Shift Keying

The second pillar upon which the DAB standard was built is DQPSK. Whereas COFDM defines how the sub-carriers are arranged in a DAB symbol, DQPSK defines how data is modulated onto these sub-carriers over consecutive symbols. To understand the mechanisms behind DQPSK, one should first understand the concepts of Phase Shift Keying (PSK), Quadrature Phase-Shift Keying (QPSK), and differential modulation. Since these ideas are somewhat simpler than those in the previous section, they are covered briefly in the coming text, followed by a description of DQPSK itself. Importantly, the DAB standard uses a slight variation of conventional DQPSK, termed $\pi/4$-DQPSK, which will also be discussed.

3.3.1 Phase-Shift Keying

Consider a sinusoidal carrier wave with a frequency of ω_0 and an amplitude of A. Unlike in Amplitude-Shift Keying (ASK) or Frequency-Shift Keying (FSK) scheme, where the amplitude or frequency of the carrier wave is modulated, a PSK scheme keeps both of these parameters constant over time. Instead, the *phase* of the carrier wave is modulated. Mathematically, one can think of the modulated wave as follows:

$$s(t) = A \cdot \sin(\omega_0 t + \phi[n]) \tag{3.3}$$

where $n = \lfloor \frac{t}{T} \rfloor$, with T as the symbol period; or, in other words, where $\phi[n]$ is the corresponding phase for the symbol at the discrete-time step n.

In the simplest case, one could take a bitstream of two possible values, either 0 or 1, and correspondingly modulate the carrier with a phase of either 0 or π radians. Such an approach is called *binary* phase-shift keying, since only two values are used. Figure 3.11 shows a simple illustration of this process, with an arbitrary bitstream.

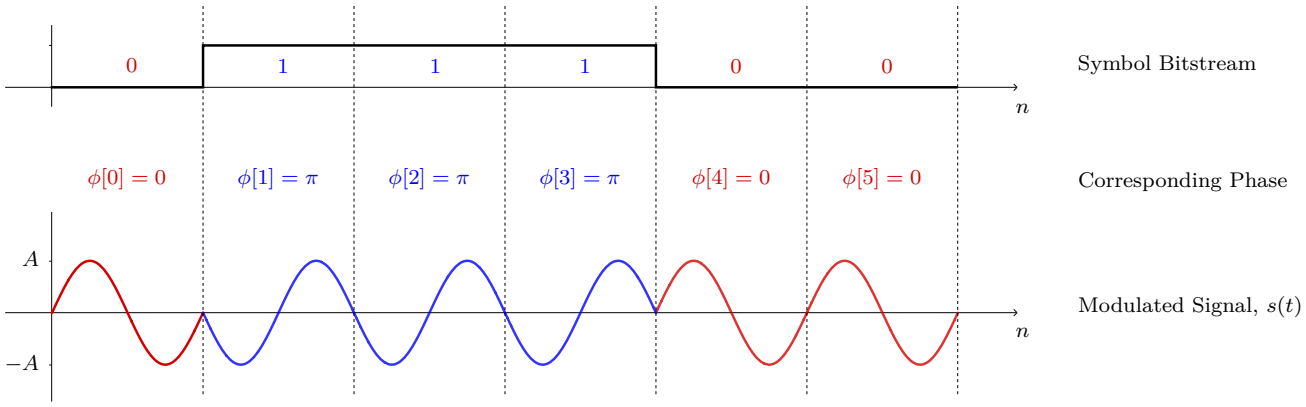


Figure 3.11: Illustration of binary phase-shift keying with an arbitrary bitstream.

Notice in the figure how the value at each point in the bitstream directly affects the corresponding phase of the modulated signal—when the bit is 0, the phase is 0 radians; when the bit is 1, the phase is π radians. These mappings are emphasised by the colours of red and blue respectively.

3.3.2 Quadrature Phase-Shift Keying

The idea of QPSK is simply an extension of Binary Phase-Shift Keying (BPSK), where four possible angles are used instead of two—hence the word *quadrature*. Having four phase-values enables one to transmit two bits of data with each symbol—for example, the bit-pair '00' could map to an angle of 0 radians, the bit-pair '01' to an angle of $\frac{\pi}{2}$ radians, and so on. Intuitively, this approach can send twice as much information as a BPSK approach, in the same amount of time and constrained by the same bandwidth.

3.3.3 Differential Modulation

Consider the task of demodulation for a PSK-modulated signal, whether BPSK, QPSK, or of a higher order. On the surface, it may be tempting simply to split the incoming data into chunks, and measure the phase values of each of these chunks. Unfortunately, though, when a modulated signal travels from a transmitter to a receiver, various nonideal effects can alter the signal adversely—the details of which fall outside of the scope of this report. Consequently, such signals require a demodulation procedure which is 'coherent,' as explored thoroughly in [65]. This requirement results in receivers that are more complicated and more expensive than they would be otherwise.

A straightforward alternative to PSK which solves this problem is called Differential Phase-Shift Keying (DPSK). In essence, instead of using the actual phase value of a carrier wave for modulation, a DPSK scheme uses the *difference* between the phase values of consecutive symbols. For example, if one was using differential-BPSK—where two possible bits are available—one could map the bit 0 to a phase-difference of 0 radians between symbols, and the bit 1 to a phase-difference of π radians. Figure 3.12 illustrates this example.

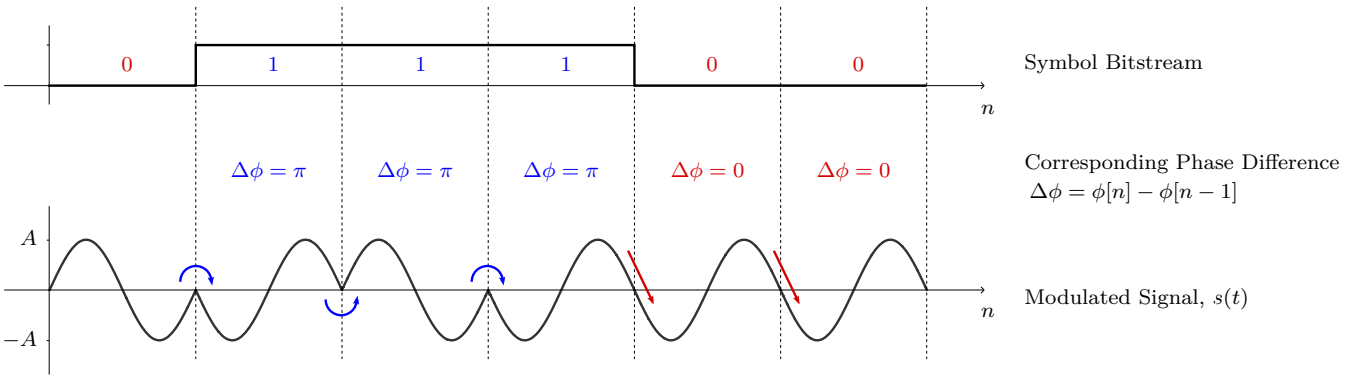


Figure 3.12: Illustration of a differential binary phase-shift keying with an arbitrary bitstream.

Notice carefully the distinction between Figure 3.11, which illustrated a BPSK scenario, and the figure here, which is showing a differential-BPSK scenario. In the latter's case, no information can be garnered from the actual phase of the modulated signal, $s(t)$, at any point in time. Instead, one must consider a symbol's phase, and compare it to the phase of the symbol before it. In this way, the demodulation process no longer needs to be coherent, which enables cheaper and more efficient receiver systems.

3.3.4 Adding a $\pi/4$ Phase-Offset Between Consecutive Symbols

The modulation scheme used by the DAB standard is essentially DQPSK, with one minor revision: the four possible phase values are rotated by $\frac{\pi}{4}$ radians after each symbol. Figure 3.13 illustrates this approach.

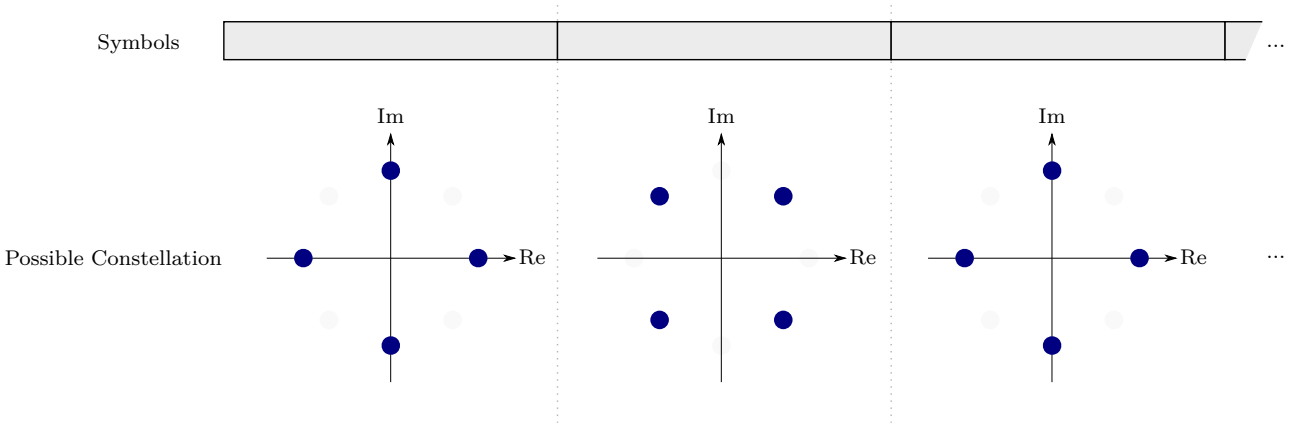


Figure 3.13: Illustration of a $\pi/4$-DQPSK modulation scheme.

Accordingly, because of the *differential* modulation, the possible DQPSK values for this approach are:

$$\Delta\phi \in \left\{\frac{\pi}{4}, \frac{3\pi}{4}, -\frac{\pi}{4}, -\frac{3\pi}{4}\right\} \tag{3.4}$$

This scheme is usually termed $\pi/4$-DQPSK, but hereafter this distinction will not be made.

3.4 Transmission Frame

The previous two sections have detailed the two core technologies that underpin the DAB standard: COFDM, and DQPSK. It is important now to discuss how these concepts are practically implemented in a DAB signal. This section begins with an overview of a DAB 'symbol', followed by the structure of a DAB 'transmission frame' in the time domain. The three relevant regions in this structure are then highlighted briefly. Relevance in this context reflects that which is useful in a broader PR system, not necessarily what is relevant in a DAB radio receiver system.

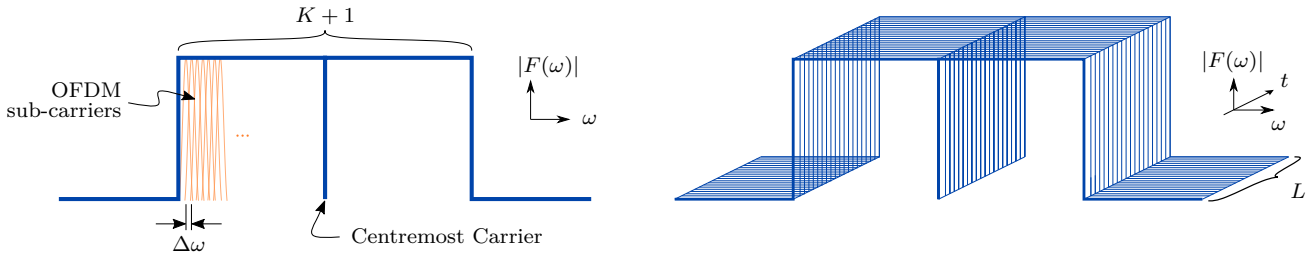
Note that the dimensions provided in the coming text are all generic, independent of the DAB transmission mode used. Appendix C provides a table showing the actual values for these dimensions, for each of the possible modes.

3.4.1 Overview

Fundamentally, a DAB signal is constituted of a series of DAB transmission 'frames', where each frame is a collection of DAB 'symbols' (i.e. OFDM symbols). In the frequency domain, each DAB symbol contains K adjacent OFDM sub-carriers, arranged symmetrically around a central frequency, which

itself is not a sub-carrier. Hence, the DAB spectrum spans $(K+1)$ sub-carriers, with a spacing of $\Delta\omega$ between each of them[3]. The magnitude of the frequency spectrum of a single DAB symbol is depicted in Figure 3.14a, along with various annotations of the aforementioned information.

Every DAB frame contains L DAB symbols (as well as the Null Symbol (NS), which will be discussed shortly), where the K OFDM carriers from symbol to symbol are differentially modulated using DQPSK. The first of these symbols is always a known reference symbol, aptly called the Phase Reference Symbol (PRS), with specifically defined phase values. Since these phase values are known, the subsequent $(L-1)$ symbols can be differentially modulated, and later differentially demodulated, deterministically. The symbols from a frame can be viewed as a surface plot, as illustrated in Figure 3.14b.



(a) A single DAB symbol, with annotations.

(b) The L symbols of a DAB frame.

Figure 3.14: Illustrations of the magnitude of a DAB symbol's frequency spectrum.

At this point, it is important to make a note regarding nomenclature. The plots given in Figure 3.14 were shown from a frequency-domain perspective. Of course, the time-domain plots of these symbols would still represent the same underlying physical signal in the real-world, only viewed with a different lens. But, for the sake of clarity, a minor distinction will be made for this report. Notice that the OFDM sub-carrier values, when viewed from the frequency-domain perspective, are readily available for consideration without further transformation. That is, if one wanted to know the value of a certain sub-carrier, ω_n, one could simply index that sub-carrier in the frequency-domain array. In that sense, the signal is 'demultiplexed' when in this perspective. In contrast, when the signal has been generated in the time-domain via the FFT algorithm—as explained in Section 3.2.4—the numerous sub-carrier signals are summed into one complex signal. Therefore, this perspective can be considered as 'multiplexed.'

In the time domain, a DAB frame begins with a NS, which has a period of T_{null}. This is followed by the PRS, the first of the L DAB symbols. The remaining $(L-1)$ symbols come next, and can be broadly termed the 'data-carrying' symbols. Each of these L DAB symbols have a 'useful' period of T_u. Prepended to each useful interval is a guard interval, with a period of T_g (this period is sometimes notated as Δ). The total symbol period, T_s, is therefore the sum of these two periods: $T_s = T_u + T_g$. The entire frame has a period of T_f, which includes the NS and the L DAB symbols. Figure 3.15 illustrates all of this information, as annotated on a single DAB frame.

Each of the three regions of the DAB frame is explained briefly in the coming subsections.

[3]Recall that, for orthogonality to hold over an integration period of T, the relationship of $\Delta\omega = \frac{2\pi}{T}$ must hold.

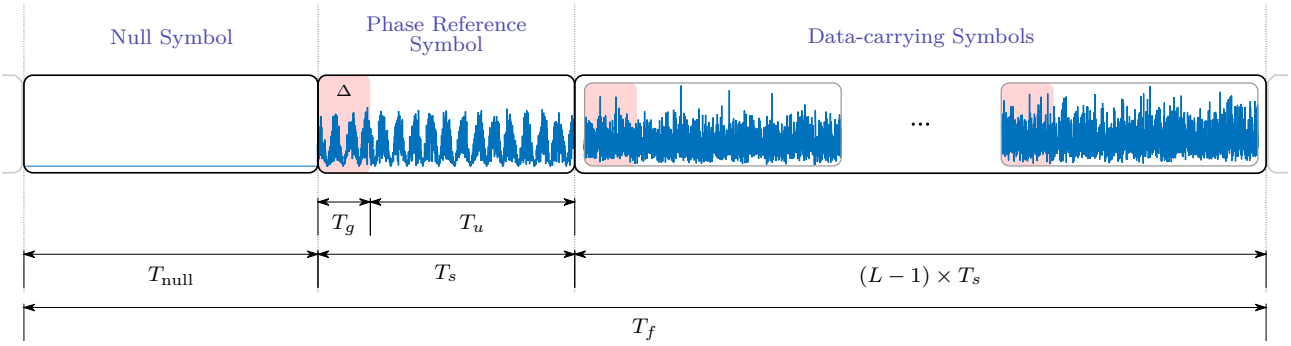


Figure 3.15: Illustration of a DAB transmission frame.

3.4.2 Null Symbol

The NS is a period that begins the DAB frame, where all OFDM sub-carriers are 'turned off'—i.e. set to a value of 0. Hence, during this time, no power is transmitted. The purpose of this symbol is for 'coarse' synchronisation of the frame. That is, to detect the approximate beginning of a DAB frame.

3.4.3 Phase Reference Symbol

The PRS is the first of the L DAB symbols within a frame, and is arguably the most important. Fundamentally, it serves two purposes. Firstly, as mentioned previously, it provides a reference for the differential modulation of the subsequent symbols. Secondly, it serves as a mechanism for 'fine' synchronisation of the DAB frame. Because the PRS consists of known values, detecting the location of these values in a received signal can reveal the precise beginning of a frame. Together, the NS and the PRS are called the 'Synchronisation Channel.'

The PRS is generated via a specifically-defined process outlined in the DAB standard, which uses various formulae and look-up tables [1]. Note that the values of the PRS are not arbitrary: they create a so-called Constant-Amplitude Zero-AutoCorrelation (CAZAC) waveform in time, which facilitates robust detection in the presence of noise and other unwanted effects [37].

3.4.4 Data-carrying Symbols

The 'data-carrying' symbols are all those that follow the PRS in a DAB frame. Note that this terminology is not used in the DAB specifications document whatsoever, and is introduced here for the sake of simplicity. The formal DAB standard delves into the technicalities of these symbols, dividing them into a 'Fast Information Channel,' and a 'Main Service Channel.' It further defines concepts such as 'Fast Information Blocks,' 'Common Interleaved Frames,' and so on.

Fortunately, because of the context of this project, one can ignore these details. Recall that a PR system only requires the DAB processing chain to *reconstruct* a given DAB signal perfectly according to the standard. Since no actual data (audio, station information, etc.) needs to be extracted from the signal, the specifics of the data-carrying symbols are safely omitted in this report.

3.5 Summary

This chapter provided an insight into the important concepts surrounding the DAB standard, as defined in [1]. This included a detailed motivation for and description of both COFDM and DQPSK, which are fundamental concepts used in DAB signals. Thereafter, an overview of the relevant aspects of the DAB transmission frame was outlined, based on the specific intention of eventual integration into a PR system.

Chapter 4

Digital Audio Broadcasting: Processing Chain Design

> Nobody can build you the bridge over which you must cross the river of life, nobody but you alone.
>
> —*Friedrich Nietzsche*

This chapter details the core design component of the project: a DAB processing chain. The fundamental task at hand was to demodulate and remodulate a given DAB signal, in order to create a perfect reference signal of given recorded data. Fortunately, as outlined in the previous chapter, not all of the intricacies of DAB signals were required to be understood or even considered for integration into a PR chain.

The design of the chain involved studying the relevant aspects of the DAB standard document from the ETSI [1], as well as using perfect reference data for initial testing. For some functionality, real-world data was also used for design and testing. So far as possible, the chain's processing steps were crafted independently of the DAB transmission mode used. All of the necessary MATLAB functions were designed with a `dab_mode` input to allow this generic approach. However, the provided DAB data was only of Transmission Mode I type; therefore, the chain was only practically tested in this mode.

This chapter begins with an overview of the processing chain, and then unpacks the three main blocks of the chain into respective sections. Each section then dives deeper into the different blocks, looking at their sub-blocks and associated functionality. Block diagrams are used extensively, and the dimensions provided on these diagrams are based on the generic DAB parameters. The corresponding parameters assigned to each mode are given in Appendix C. The variable names shown on the diagrams match directly to the those used in the MATLAB code.

4.1 Overview

Within the DAB processing chain, three main functional blocks were identified: pre-processing, demodulation, and remodulation. The first block reads in a DAB recording of In-phase and Quadrature (IQ) data saved as a binary file, and outputs a single DAB frame from this file. This frame can then be input to the `demodulate` block, which extracts the DAB data from it. This data is not the decoded information from the DAB recording—such as the audio signal—rather, it is simply an extracted

bitstream. This bitstream can then be perfectly reconstructed as a DAB frame, by the `remodulate` block. A diagrammatic overview of this chain is given in Figure 4.1.

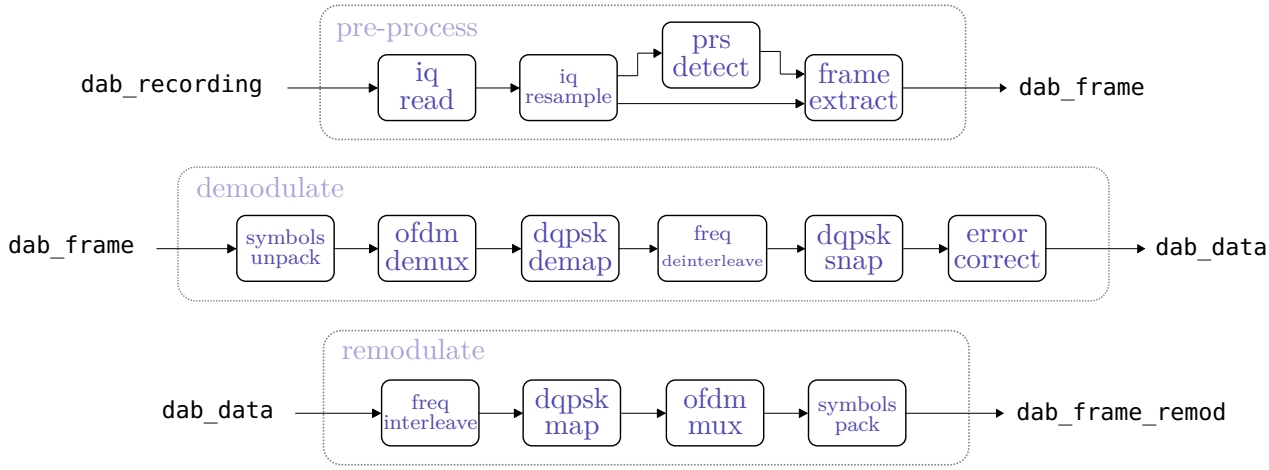


Figure 4.1: Block diagram showing entire DAB processing chain.

Notice in the figure that each of the aforementioned blocks—the `preprocess`, `demodulate`, and `remodulate` blocks—was created from a collection of smaller sub-blocks. These sub-blocks were designed to perform simpler tasks individually, and were written as relatively short MATLAB functions. This atomic-function approach had a host of benefits for the design, debugging, and validation of the chain, and added layers of abstraction for the larger system.

Each of the three blocks will be unpacked in the sections that follow this one, with subsections dedicated to the relevant sub-blocks.

4.2 Pre-processing

4.2.1 Overview

The pre-processing chain was designed to read in an IQ data file, synchronise it appropriately, and extract a single frame from it. Note that in an actual PR system—where this DAB processing chain would be only a small part of a larger chain—this block would be markedly different. If nothing else, a real-world chain would probably stream the IQ data, rather than read it from a pre-recorded file. Moreover, a real chain might process several DAB frames at once. However, such details are largely context-specific, and it would be superfluous to include every use-case in this report. Importantly, the other two blocks—used to demodulate and remodulate—would mostly remain the same in various other scenarios. Thus, to make matters simpler, the pre-processing block was designed to be straightforward: to read in data and extract a single frame. That is not to say the block was trivial, and there were some important design decisions made for it.

The pre-processing block had four key stages: reading in IQ data from a binary file, resampling this data if necessary, detecting the location of the PRS and thus synchronising the recording, and then

extracting a DAB frame accordingly. This pipeline is shown graphically in Figure 4.2.

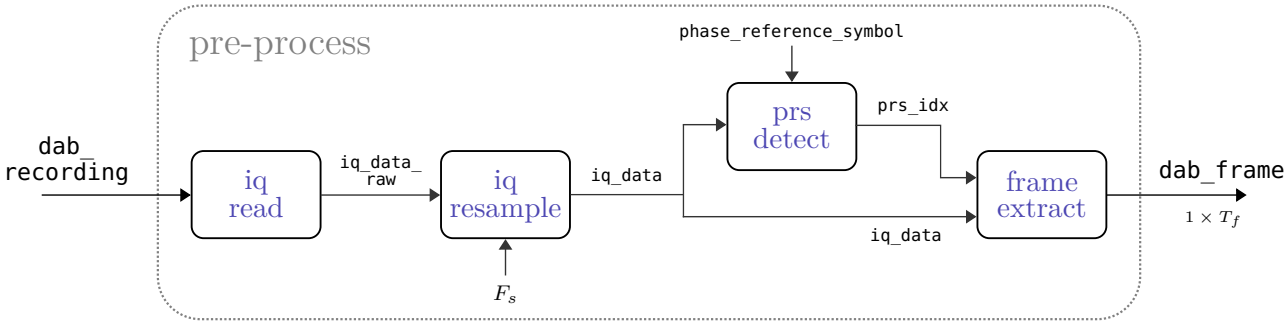


Figure 4.2: Block diagram showing preprocessing section of processing chain.

4.2.2 Reading IQ data from file

The first sub-block was called `iq_read`, and, as mentioned previously, would likely be very different in a real PR chain. Nonetheless, here it read in a DAB recording from a binary file, and output so-called 'raw' IQ data. No changes were made to the data in this function, and it essentially just abstracted MATLAB's file-handling functionality conveniently.

As a minor note, one must remember that a DAB recording contains IQ data, which is complex-valued. Since a binary file cannot store such a datatype, it will usually be stored as pairs of numbers. To read this correctly in MATLAB, one can use the simple code shown in Listing 4.1.

```
1 iq_data_raw = data_from_file(1:2:end) + j*data_from_file(2:2:end);
```

Listing 4.1: MATLAB code for creating a complex data array with the values read from a binary file.

Though it is not shown in the pre-processing block diagrams, the MATLAB function for `iq_read` also included some other input parameters for convenience, including the data-type of the binary file, the offset to skip in the file's data, the number of frames to read, and so on.

A perfect DAB recording that would be output from the `iq_read` function is shown in Figure 4.3, showing a handful of DAB frames. Notice the separation of successive frames by the distinct null symbols.

4.2.3 Resampling IQ data

In order for one to utilise the FFT to demultiplex an OFDM symbol, certain requirements must be met. In DAB Transmission Mode I, the window length for a single symbol, notated as T_u, is 2048 samples. This means that a 2048-point FFT must be used, with a resulting 2048 points in the frequency domain. Recall also that OFDM has a strict requirement for the spacing between adjacent carriers—1 kHz for DAB Mode I. Notice, then, if the sample rate for the original signal is precisely set at $F_s = 2.048\,\text{MHz}$,

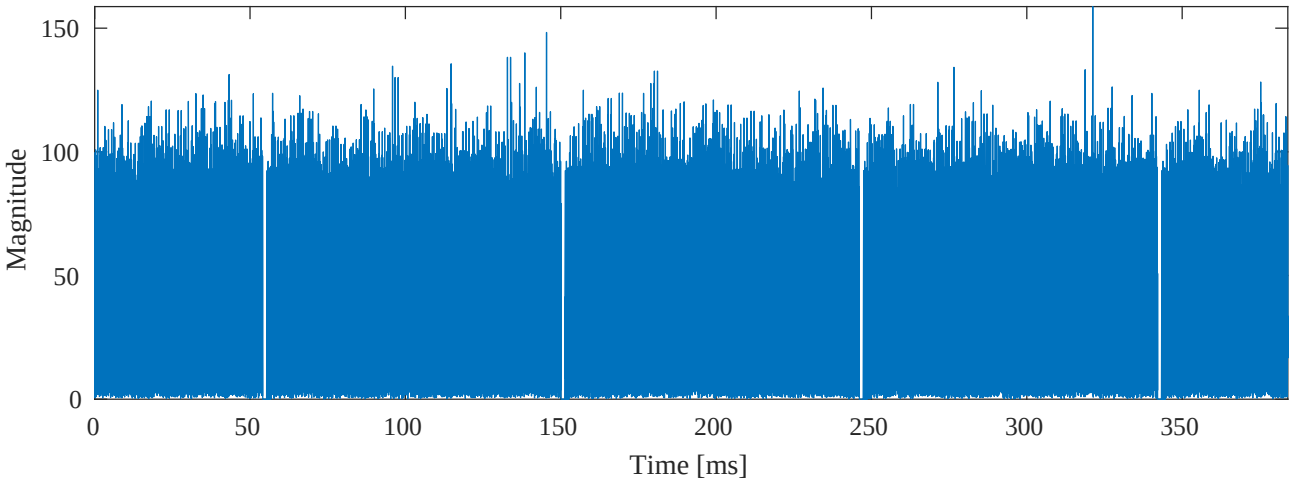


Figure 4.3: Time-domain plot of a handful of frames of a perfect DAB signal.

for DAB Mode I, then the bins from the FFT will be perfectly aligned on the OFDM carriers—enabling proper demultiplexing.

However, in a real-world scenario, the desired sampling rate might not necessarily be used in the receiver. Thus, in such a situation, the IQ data must be resampled. Consider an original sampling rate of F_s, a desired sampling rate of $\tilde{F}_s = 2.048\,\text{MHz}$, with $F_s \neq \tilde{F}_s$, and with

$$\frac{\tilde{F}_s}{F_s} = \frac{P}{Q} \tag{4.1}$$

where P and Q are integers—preferably chosen such that P/Q is in its most reduced fractional form. Resampling, then, is equivalent to upsampling by a factor of P, and thereafter downsampling by a factor of Q.

Fortunately, MATLAB provides a neat abstraction of this process with the `resample(X, P, Q)` function, where X is a uniformly sampled signal, and P and Q are as defined above.

4.2.4 Frame Synchronization via PRS Detection

One of the most important parts of the entire DAB processing chain was correct time-synchronisation.[1] If the frame from the pre-processing block was misaligned, the entire demodulation process could break due to misalignment of the OFDM symbols, thus causing havoc in the DQPSK demodulation. Fortunately, the inclusion of a guard interval with cyclic prefixing provided some leeway, but errors could easily arise if one was not careful. It will later be shown how a result can be negatively affected by the incorrect alignment of a DAB frame.

There are two mechanisms built into the DAB signal for synchronisation, namely the NS and the PRS. As specified in the DAB standard document [1], the former symbol is to be used for so-called 'coarse' synchronisation—detecting an approximate location of the frame's beginning; whereas the latter symbol is used for 'fine' synchronisation—detecting the exact sample at which the frame starts.

[1] Frequency synchronisation would also be a vital part of a robust DAB processing chain, but fell outside of this project's scope. Future work should certainly explore this topic.

Naturally, for each of these detection processes, one cannot consider the entirety of an incoming signal. More than just practically impossible, it would be detrimental to do this, for each symbol in a DAB signal is carrying independent information. Instead, only a chunk of the data can be considered at once, as a subset of the longer signal. This process is termed 'windowing,' where the signal is truncated to a finite domain over which it will be examined. Let the length of this window be notated[2] as T_w, and let the interval between successive windows be called the 'advancement interval', T_a. Graphically, these values are depicted in Figure 4.4.

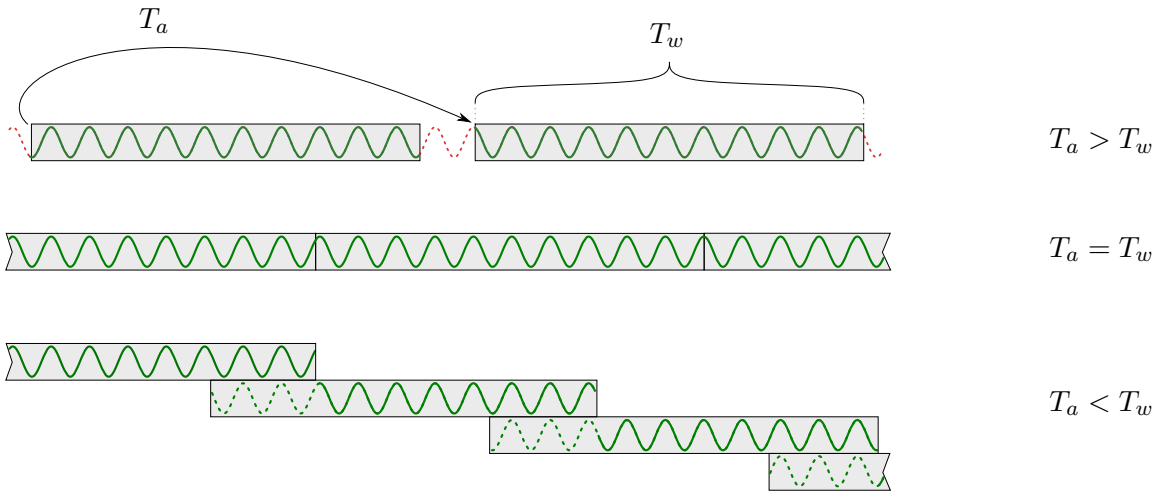


Figure 4.4: Three possible scenarios for the values of T_w and T_a.

In order not to skip any sections of the signal, it is clear that the following must hold:

$$T_a \leq T_w \tag{4.2}$$

Consider firstly the detection of the NS. Recall that this symbol is a period in the signal during which all OFDM carriers are turned off and no power is transmitted. On the receiver's end, there will still be noise recorded from the environment and elsewhere, but the received power over the length of the null symbol, T_{null}, will be significantly less than in other parts of the signal. Hence, in order to detect this symbol, one could calculate the average power starting at index n_0 over a window of $T_w = T_{\text{null}}$. Let this be notated as P_{n_0}, and calculated as:

$$P_{n_0} = \frac{1}{T_w} \cdot \sum_{n=n_0}^{n_0+T_w-1} |x[n]|^2 \tag{4.3}$$

where $x[n]$ is the sampled signal, windowed on the domain $[n_0, n_0 + T_w)$. This window could be moved along the original signal, in increments of T_a. Since the NS only provides a coarse result, and further synchronisation steps would always be required, the specific value of T_a was not too important, provided it was less than or equal to T_w. When the average power of a window decreased below a particular threshold—for example, it could be compared to the average power of a *frame*-lengthed

[2]The is not necessarily standardised notation.

chunk of the signal—the NS had been approximately detected. If things were working correctly, the symbol detection should have occurred within the first T_f samples—if not, there was a problem.

An outline of the coarse detection process is given in Algorithm 1.

input : A sampled DAB signal, $x[n]$
output : The approximate index of the first Null Symbol
parameter : Threshold constant, $\gamma > 1$

begin
  $P_{\text{frame}} \leftarrow \frac{1}{T_f} \cdot \sum_{n=0}^{T_f-1} |x[n]|^2$
  $n_0 \leftarrow 0$
  while $(n_0 < T_f)$ **do**
    $P_{n_0} \leftarrow \frac{1}{T_w} \cdot \sum_{n=n_0}^{n_0+T_w-1} |x[n]|^2$
    if $(P_{\text{frame}} > \gamma P_{n_0})$ **then**
      return n_0
    else
      $n_0 \leftarrow n_0 + T_a$
    end
  end
end

Algorithm 1: Coarse frame synchronization via NS detection.

Consider next the detection of the PRS. Because this symbol is precisely defined by the DAB standard, it was known *a priori* and thus could be detected via a matched filtering process, which is essentially a time-domain correlation. Conceptually, this entailed designing a filter, $H(\omega)$, based on the PRS:

$$H(\omega) = R^*(\omega) \tag{4.4}$$

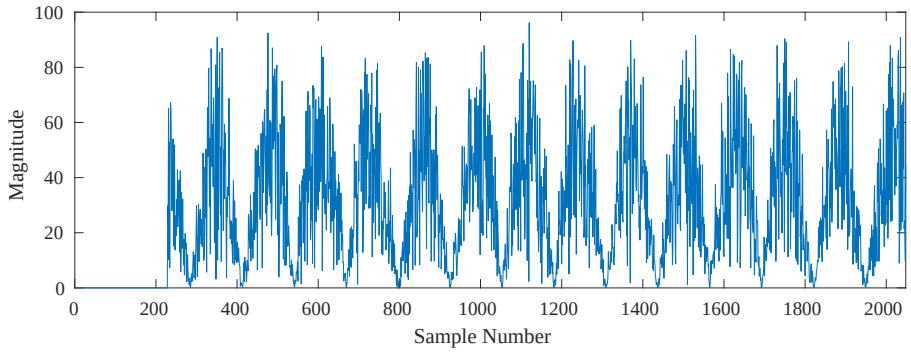
where $R^*(\omega)$ was the conjugate of the continuous-time Fourier transform of the PRS. Of course, since the system was implemented digitally, the filter had to be defined discretely:

$$H[k] = R^*[k] = \text{DFT}\{r[n]\}^* \tag{4.5}$$

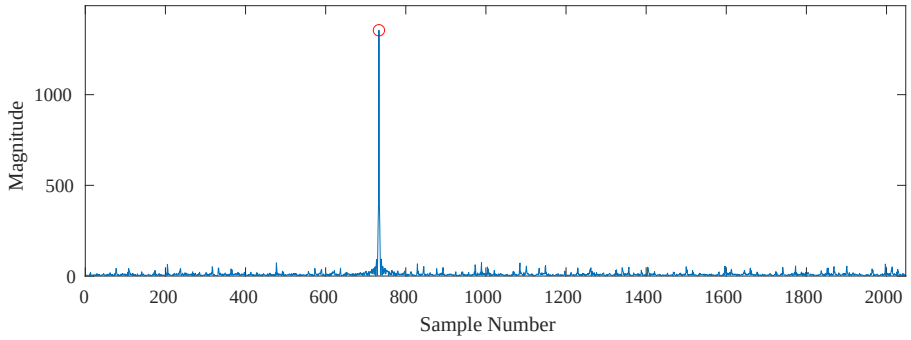
The output of the matched filter in the Fourier domain, $Y[k]$, for a given input signal, $x[n]$, can be calculated via a simple Hadamard (pointwise) product:

$$Y[k] = H[k] \odot X[k] = H[k] \odot \text{DFT}\{x[n]\} \tag{4.6}$$

The output of this filter in time, $y[n]$, will contain a peak value at the location within $x[n]$ where $r[n]$ is most highly correlated. In an arbitrary section of the DAB signal, the output will simply be noise, as the signals are not correlated; however, in a section containing the PRS, the peak value will be noticeably larger than surrounding values, even when using a non-perfect recording. One could detect this peak by comparing it to the average value of the matched filter output, for example. Figure 4.5 shows an example of the filtering process when the PRS falls within the window of $x[n]$.



(a) Input to matched filter, $x[n]$



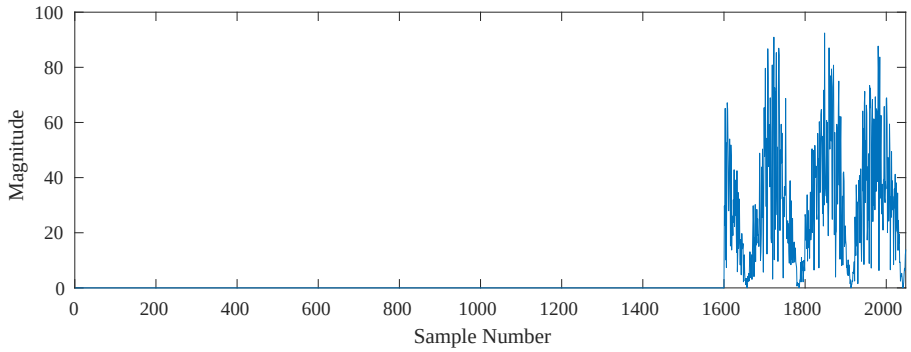
(b) Output from matched filter, $y[n]$, with peak detected

Figure 4.5: Plots showing the input to and output from the matched filter, when the PRS is detected.

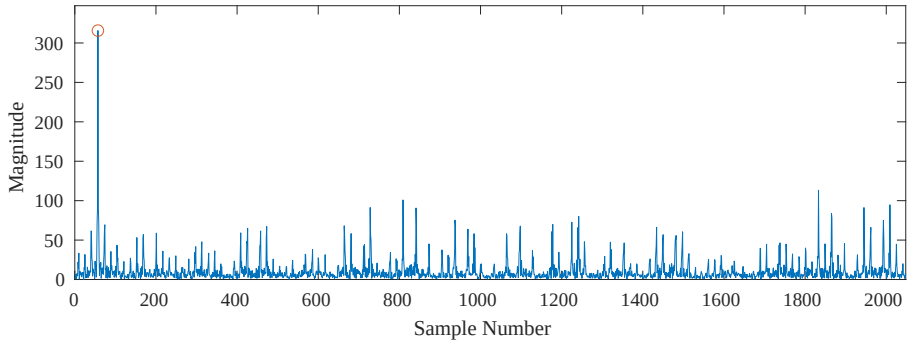
By moving a window of length $T_w = T_u$, for example, across the input DAB signal until a peak was detected in the matched filter output, one could locate the PRS—and thus the beginning of the DAB frame—precisely. This detection should occur within the first T_f samples. There are two important points to make here. Firstly, note that the magnitude of the matched filter's output, $|y[n]|$, decreases as the offset of the recorded PRS within the window increases. That is, suppose the PRS begins at the sample index n_p. If the windowed DAB signal, $x[n]$, covers the domain $n \in [n_p, n_p + T_u]$, then $x[n]$ is perfectly correlated with the PRS (ignoring noise in the recording); thus, the output of the matched filter will be large. However, as n_p increases, less of the PRS signal will be included in $x[n]$, and the correlation peak will decrease. The smaller the correlation peak, the less distinct it will be from other peaks, and the less likely it will be successfully found.

Secondly, notice that the peak of the matched filter output, in Figure 4.5b, does not occur at the onset of the input signal, $x[n]$; rather, it occurs around 500 samples later. This was expected, due to the inclusion of a guard interval—since the matched filter, $H[k]$, is defined around the original PRS signal of length T_u, without the guard interval. Unfortunately, this causes an unwanted effect in the detection process. Consider a situation in which $x[n]$ only contains the guard interval of the PRS, before the actual 'useful' symbol begins, as shown in Figure 4.6a.

Notice that the peak identified in the matched filter output, seen in Figure 4.6b, is erroneous—the actual PRS has, in fact, not yet begun. Recall that the guard interval is duplicated from the tail-end of the original symbol; thus, the end of the PRS in the matched filter has correlated with the guard interval of the input signal, $x[n]$. Though this peak is not massive, it can nevertheless be incorrectly identified as the beginning of the DAB frame.



(a) Input to matched filter, $x[n]$



(b) Output from matched filter, $y[n]$, with peak erroneously detected

Figure 4.6: Plots showing the input to and output from the matched filter, when the PRS is erroneously detected due to the guard interval.

Both of these points—the decreasing correlation peak, and the guard interval problem—motivated the choice of the advancement interval to be less than the window period. Of course, there was a trade-off, as a shorter advancement interval increased the computation required to detect the PRS. For this project's implementation of the DAB processing chain, the values were chosen as $T_a = \frac{1}{2}T_w$. This ensured that any small peak occurring in the second half of a window interval was guaranteed to occur again, as a much larger peak, in the first half of the *next* interval. Therefore, the threshold for a 'true peak' value could be stricter, which also solved the guard interval problem.

An outline of the PRS detection process is given in Algorithm 2.

Recall that detection of the NS was designed for coarse synchronisation, while the PRS was intended for fine synchronisation. Essentially, then, there were two broad algorithms for frame alignment: to locate the approximate beginning of a frame with the NS and thereafter finely align the frame with the PRS, or to use only the PRS without first coarsely detecting the NS. For a robust design decision between these two approaches, one should consider the various computational complexities involved in each of the algorithms, as well as possible optimisations that could be made. Nonetheless, for the sake of simplicity, the latter choice was taken in this project: to use only the PRS for detection. Further work could be done analysing the best decision to make, but such work was outside of the scope of this report.

Another important question when integrating the DAB processing chain into a larger PR system is the regularity of frame synchronisation. Ideally, one would only need to synchronise *once*, if the clock of the transmitter and receiver were perfectly accurate. Unfortunately, in real scenarios, there would be a

inputs : A sampled DAB signal, $x[n]$; The PRS, $R[k]$
output : The exact index of the first PRS in $x[n]$
parameter : Threshold constant, $\gamma > 1$

begin
  $n_0 \leftarrow 0$
  while $(n_0 < T_f)$ **do**
    $X[k] \leftarrow \text{FFT}\{x[n_0 : n_0 + T_u]\}$
    $Y[k] \leftarrow R^*[k] \odot X[k]$
    $y[n] \leftarrow \text{iFFT}\{Y[k]\}$
    if $(\max|y[n]| > \gamma \cdot \text{mean}|y[n]|)$ **then**
      return $\text{argmax}|y[n]|$
    else
      $n_0 \leftarrow n_0 + T_a$
    end
  end
end

Algorithm 2: Fine frame synchronization via PRS detection.

host of unwanted effects, including clock drift, that may cause synchronisation errors over time. The opposite approach would be to synchronise on *every* frame—which is possible because every frame is preceded by both a NS and a PRS. Of course, this would increase the computation required for synchronisation.[3] Alternatively, some intermediary approach could be taken. Naturally, fixed rules cannot be set for this decision, and are largely context-specific. Further details are thus omitted here.

4.2.5 Frame Extraction

Once the index of the PRS has been detected, extracting a frame was trivial, as it simply entailed subsetting the original data array. This subsection is included for the sake of completeness.

4.3 Demodulation

4.3.1 Overview

The purpose of the demodulation block was to extract DAB 'information' from a given frame. This was not a *decoder*, as the intention was not to extract an actual audio signal or similar—as would be the case in a conventional DAB receiver. Rather, it returned the bitstream with which the OFDM carrier waves were modulated at the transmitter. Importantly, this demodulator block was not intended to stand alone, and its output was frankly useless in isolation. Instead, the demodulation and remodulation blocks were designed to operate in tandem, where the output of the demodulate block was the input to

[3] The computation time would not necessarily increase by a drastic amount, if one designs a good algorithm. The location of the first PRS provides an *estimate* of the subsequent PRS locations, even if there is bad clock drift; therefore, one can restrict the domain in which one searches for the later PRSs, and thus synchronise on every frame in a relatively efficient way.

the remodulate block. Ultimately, this was for the higher purpose of perfectly reconstructing a given DAB frame, to be used in a PR chain.

Figure 4.7 shows a block diagram for the demodulation processing chain, with the various sub-blocks, the nomenclature for their respective inputs and outputs, and their associated dimensions.

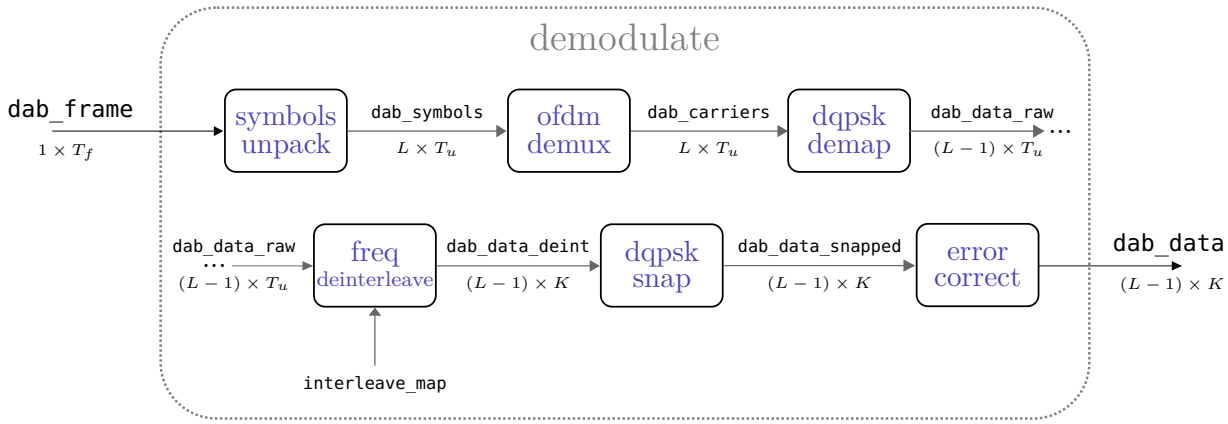


Figure 4.7: Block diagram showing demodulation section of processing chain.

The demodulation pipeline had six key stages: the DAB frame was first unpacked into a collection of symbols, which was then demultiplexed into OFDM carriers. These carriers could then be 'demapped' into DAB data via DQPSK demodulation. Thereafter, these DAB data points were deinterleaved in frequency, then snapped to their closest DQPSK values, and finally, put through an error-correction algorithm. Note that the final step of error correction fell outside of the scope of the project, and is included here for the sake of completeness. Ideally, a real-world processing chain would include this stage for increased robustness. The details for each sub-block are given in the following sections.

4.3.2 Symbols Unpacking

The purpose of this sub-block was simply to split a given DAB frame into its symbols, which could then be demultiplexed and further processed by subsequent sub-blocks. In doing so, this function also removed the symbols' guard intervals, thus extracting only the 'useful symbol' components of length T_u. Figure 4.8 illustrates how the symbols were extracted into the `dab_symbols` variable, while the guard intervals, indicated as Δ, were skipped.

In total, there are L symbols in a single DAB frame. As a result, the function receives a vector of size $1 \times T_f$, and outputs a matrix of size $L \times T_u$. Note that:

$$T_f \neq L \times T_u \tag{4.7}$$

since the original frame length, T_f, includes the guard intervals of length T_g, as well as the NS of length T_{null}. Instead, the following relationship holds:

$$T_f = L \times (T_u + T_g) + T_{\text{null}} \tag{4.8}$$

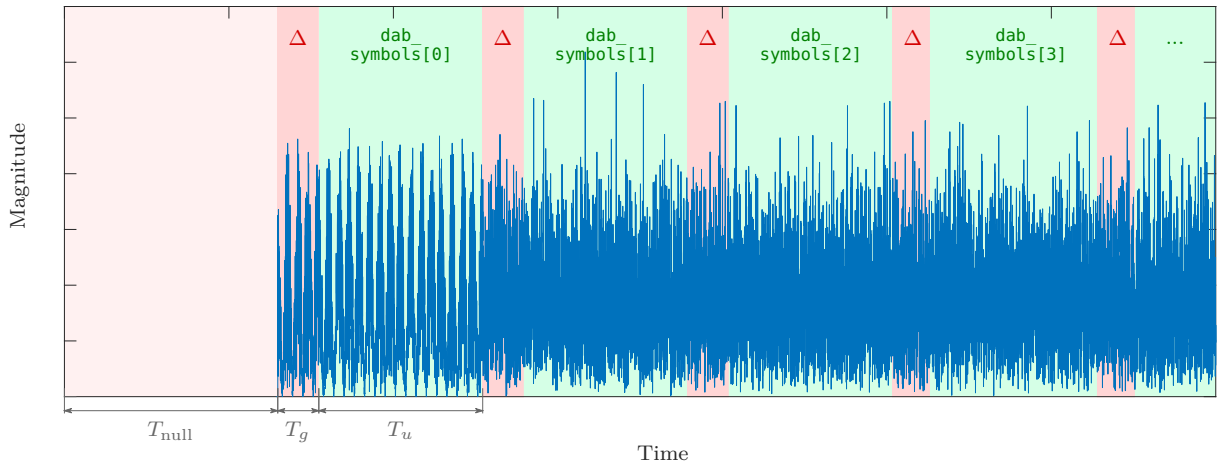


Figure 4.8: Illustration of the `symbol_unpack` functionality.

A graphical summary of the `symbol_unpack` sub-block is given in Figure 4.9.

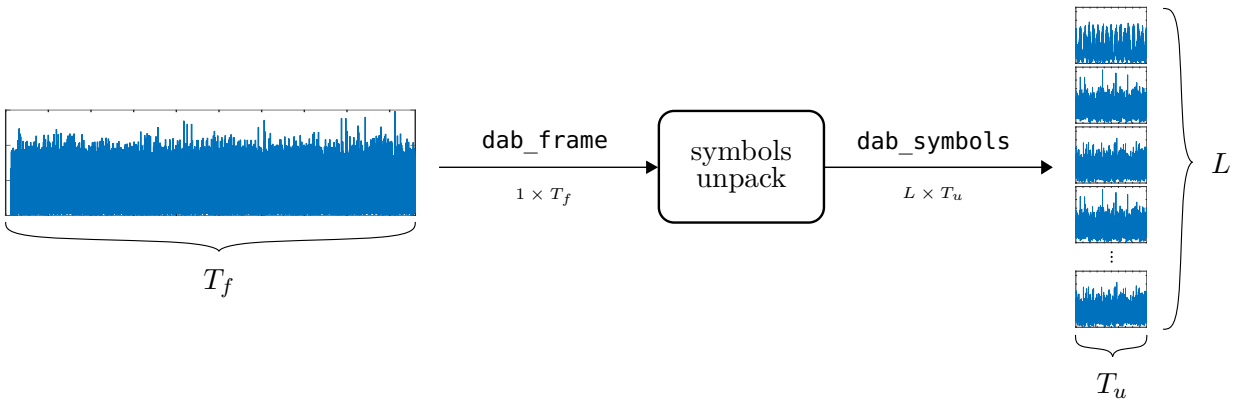


Figure 4.9: Graphical summary for `symbol_unpack` function.

4.3.3 OFDM Demultiplexing

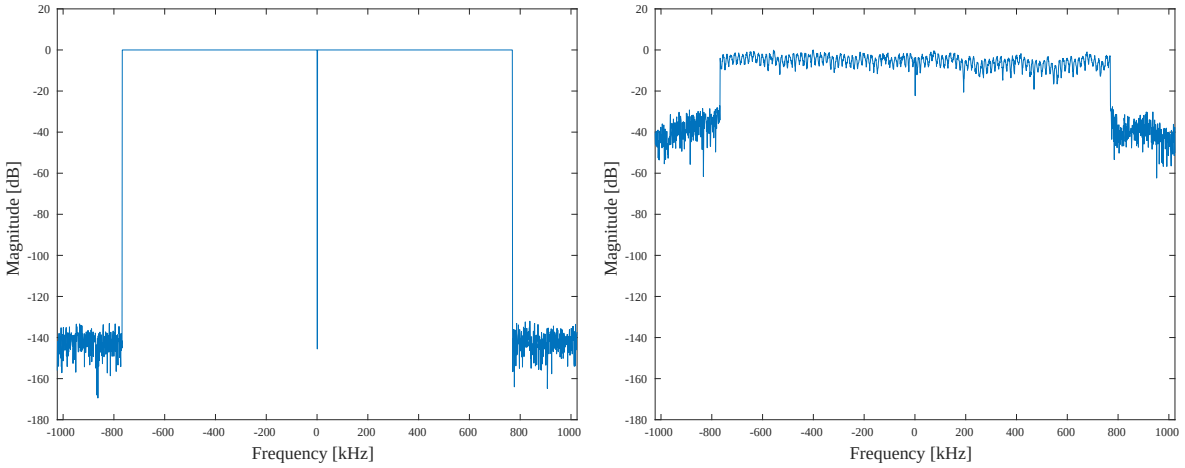
This sub-block was designed to 'demultiplex' the OFDM signal—i.e. to extract the DAB carriers from the provided time-domain symbol data. Recall from the previous chapter that this extraction can be done via a DFT/FFT operation. Consider firstly the case of a single symbol from `dab_symbols`, which is just a row in the $L \times T_u$ matrix. To demultiplex this OFDM symbol, one can use the simple code given in Listing 4.2, where the FFT of the symbol is divided by its length for correct scaling, and the `fftshift` command is used to center the frequency axis on 0 Hz.

```
1 dab_carriers(1,:) = fftshift(fft(dab_symbols(1,:))) ./ length(dab_symbols(1,:));
```

Listing 4.2: MATLAB code for demultiplexing an OFDM symbol

Figure 4.10a shows the plot of the magnitude of the DAB carriers for the perfect data signal. Notice

the distinct rectangular shape of the spectrum, and the virtually non-existent noise-floor, at $-140\,\text{dB}$. As was expected, only the K central carriers were modulated, barring the middle carrier itself, and this can be seen in the plot. Figure 4.10b shows the same plot for some real-world data. Notice that the noise floor here is considerably higher, at around $-40\,\text{dB}$. Moreover, the region containing the K central carriers is not perfectly flat. Nonetheless, the rectangular shape remains.



(a) Perfect Data (b) Real-world Data

Figure 4.10: Plots showing the magnitude of the OFDM carriers for a DAB symbol.

Recall that the `symbols_unpack` sub-block from earlier transformed the DAB frame into L symbols of length T_u. Thus, the `ofdm_demux` sub-block was designed to receive these symbols, and output L corresponding arrays of the T_u carriers for each symbol. One can thus view the output of the function as a three-dimensional surface plot, with L successive layers of plots like those shown in Figure 4.10. This is depicted in Figure 4.11 and Figure 4.12 for the perfect and real-world data respectively.

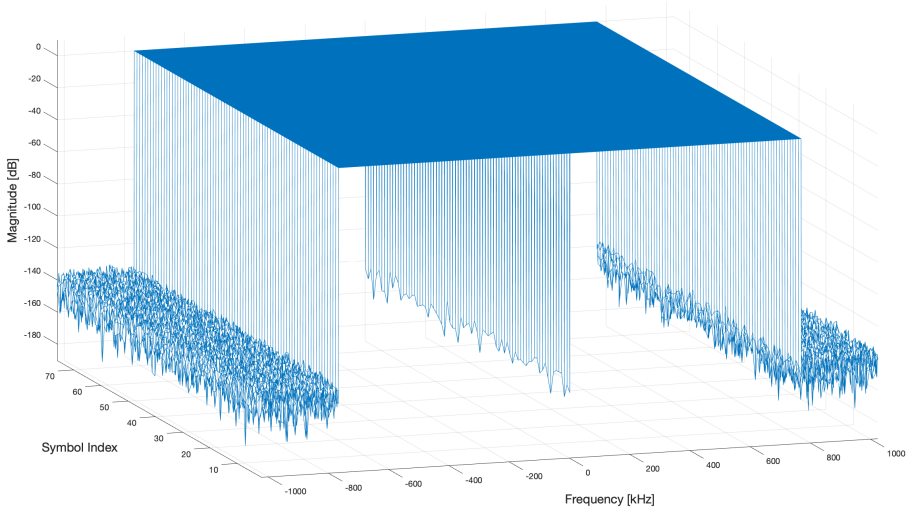


Figure 4.11: Surface plot of the magnitude of the OFDM carriers for perfect data

A graphical summary of the `ofdm_demux` sub-block is given in Figure 4.13, with the illustrations showing an example of real-world data.

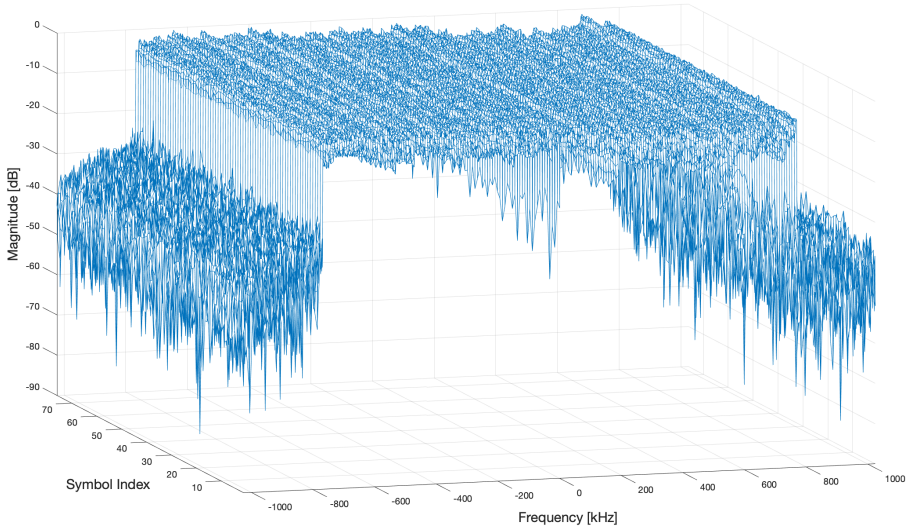


Figure 4.12: Surface plot of the magnitude of the OFDM carriers for real-world data.

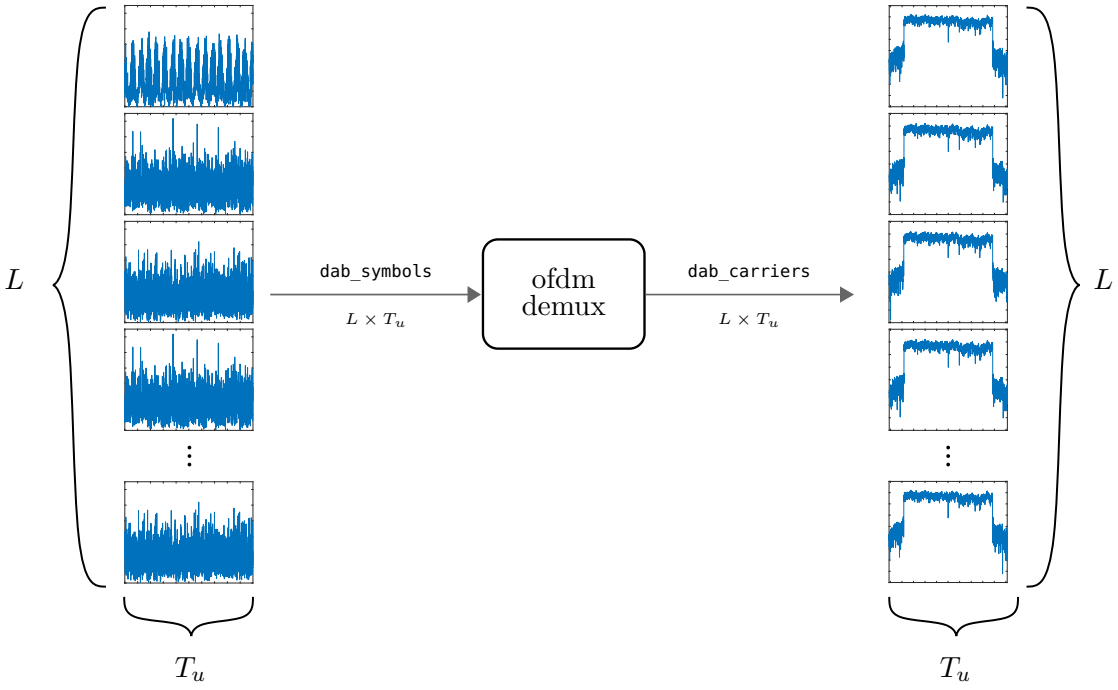
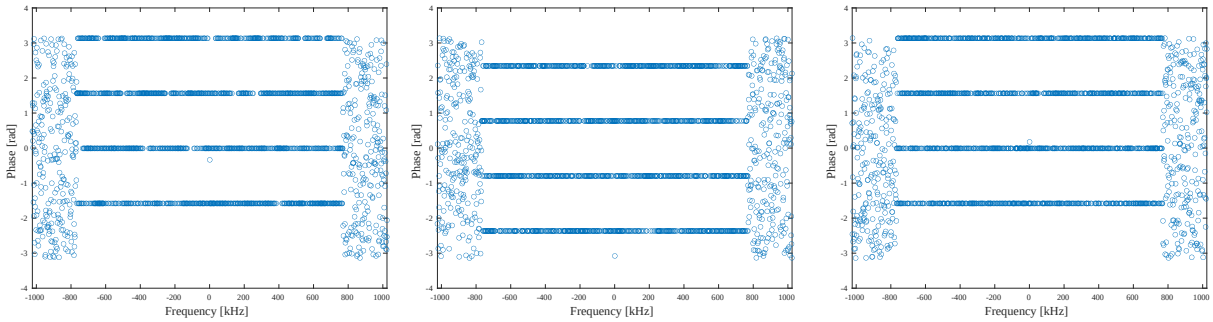


Figure 4.13: Graphical summary of the `ofdm_demux` function.

4.3.4 DQPSK Demapping

This sub-block was designed to extract the modulating signal (the original DAB 'data') from the DAB carriers. Notice that the plots of the carriers shown in the previous subsection were essentially identical from symbol to symbol, especially for the perfect data case. This is because they were showing the *magnitude* of the carriers' spectra, which remains constant over time—recall that the OFDM carriers for a DAB signal are modulated using a $\pi/4$-DQPSK modulation scheme. The rectangular shape seen in the magnitude plots thus provides a good indication that the OFDM symbols were correctly demultiplexed. From that perspective, these plots were an important graphic to consider. However, one must not forget that it is the *phases* of the carriers' spectra that are modulated with actual information.

Consider the carriers from three consecutive symbols—call them S_{l-1}, S_l, and S_{l+1}—of the perfect data. The phase plots for these carriers are shown in Figure 4.14.



(a) $\angle S_{l-1}$ (b) $\angle S_l$ (c) $\angle S_{l+1}$

Figure 4.14: Plots showing the phase of OFDM carriers for a three consecutive DAB symbols of perfect data.

One can see that the phases of the carriers indeed changed from S_{l-1} to S_l, and again from S_l to S_{l+1}. The angles of the relevant carriers (the central K carriers, excluding the middle carrier) clearly correspond to one of four values, which is logical due to the *quadrature* modulation scheme. Notice, though, that the possible values of these angles were different in Figure 4.14a and Figure 4.14c, consisting of the angles $\{-\frac{\pi}{2}, 0, \frac{\pi}{2}, \pi\}$, compared to Figure 4.14b, consisting of $\{-\frac{3\pi}{4}, -\frac{\pi}{4}, \frac{\pi}{4}, \frac{3\pi}{4}\}$. As explained in the previous chapter, this is because the data was *differentially* modulated, with a $\pi/4$ offset added after each symbol.

In line with this modulation scheme, the process of extracting the DAB data from the carriers involved considering the phase difference between consecutive symbols, with the first symbol, the PRS, used as a reference for subsequent symbols (hence its name). This process was termed 'demapping' the DQPSK data. Since the data was demapped using the difference of two symbols, a given set of L symbols mapped to $(L-1)$ streams of DAB data.

Consider the extraction of the data for an arbitrary index, $(l-1)$—call this data D_{l-1}—from the

symbols S_{l-1} and S_l. Clearly,

$$D_{l-1} = \angle S_l - \angle S_{l-1} \tag{4.9}$$

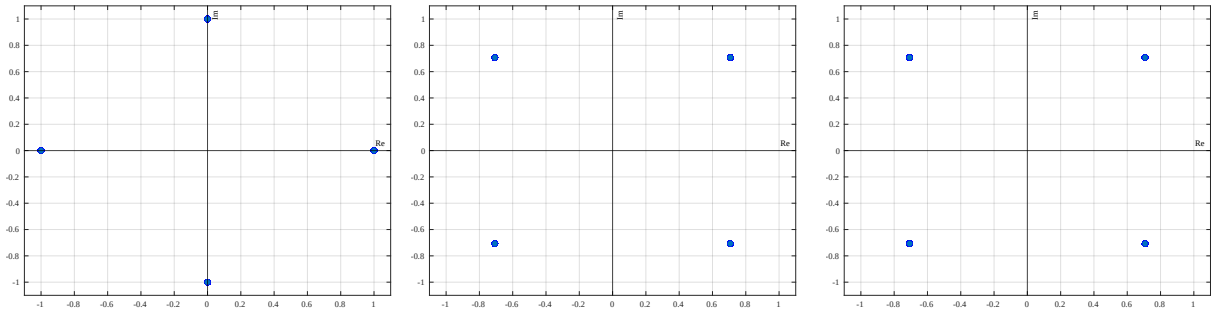
$$= \angle \left(\frac{S_l}{S_{l-1}} \right) \tag{4.10}$$

This process can be easily be achieved in software—for example, using the simple code given in Listing 4.3. To avoid unnecessary division, this calculation was only done for the relevant carriers, indexed in MATLAB using the variable called `mask`. Here, `dab_carriers(n,:)` refers to S_n, and `dab_data_raw(n,:)` refers to D_n.

```
dab_data_raw(l-1,mask) = dab_carriers(l,mask) ./ dab_carriers(l-1,mask);
```

Listing 4.3: MATLAB code for demapping DAB carriers into DQPSK data

It was also useful to view the angles of the carrier waves on a complex plane. Though this perspective does not show the progression of the angles over time, nor the relationships between the angles of the carriers in nearby frequency regions, it does show the spread of the data. Consider once again the carriers from two symbols, S_{l-1} and S_l, of the perfect data. These are shown plotted on the complex plane in Figure 4.15a and Figure 4.15b. These plots essentially show the same information as Figure 4.14a and Figure 4.14b. Figure 4.15c then shows D_{l-1}, which is the quotient of S_l by S_{l-1}. Because this data is perfect, the angles are perfectly assigned to one of the four differential values.

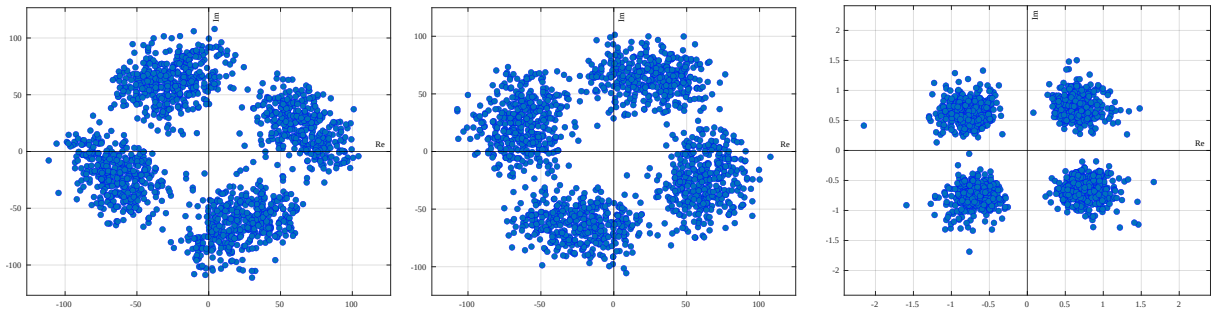


(a) S_{l-1} (b) S_l (c) $D_{l-1} = S_l/S_{l-1}$

Figure 4.15: Plots shown on the complex plane of various data derived from a perfect DAB signal.

Though the plots in Figure 4.15 are correct, they do not demonstrate the power of DQPSK clearly. One may wonder whether simple QPSK modulation would be good enough, given how neat the angles were shown to be. Therefore, it is important to consider some real-world signals. Firstly, consider Figure 4.16, which shows the same plots as before, but for an actual, recorded DAB signal.

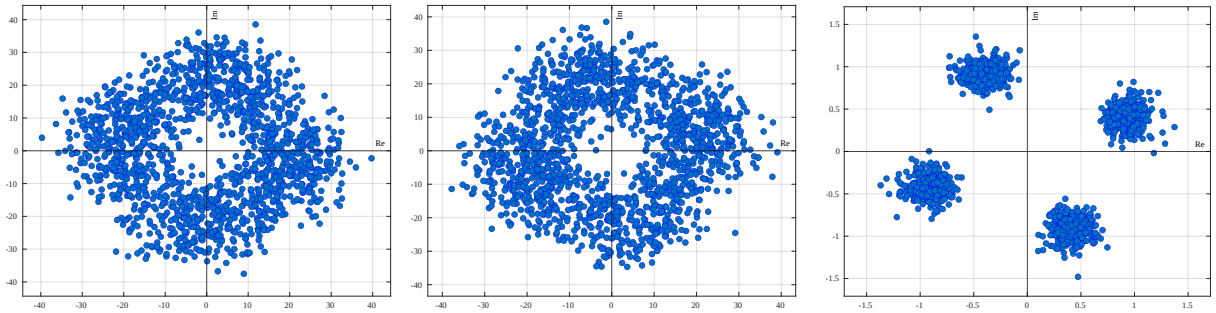
Notice how the spread of the angles in Figure 4.16a and Figure 4.16b is far worse than seen previously with the perfect data. Though one can see four broad groups of data, the precise decision boundary between these groups is uncertain. In contrast, see how much more distinct these groups become in Figure 4.16c, thanks to the differential modulation. Indeed, there is still spreading, but the decision boundary is far clearer.



(a) S_{l-1} (b) S_l (c) $D_{l-1} = S_l/S_{l-1}$

Figure 4.16: Plots shown on the complex plane of various data derived from a real-world DAB signal.

Consider an even more extreme example, from some poor-quality DAB data. Figure 4.17a and Figure 4.17b show the angles of the carriers for the two symbols—notice how there are no distinct groupings whatsoever. In fact, the data seems to be randomly spread. Nonetheless, by looking at the phase-difference between the two symbols, the four groups become far more coherent, as shown in Figure 4.17c. Granted, there seems to be a phase offset in the differences of around $\frac{\pi}{8}$ radians, but the DQPSK values are nonetheless distinct.



(a) S_{l-1} (b) S_l (c) $D_{l-1} = S_l/S_{l-1}$

Figure 4.17: Plots shown on the complex plane of various data derived from a poor-quality DAB signal.

In summary, this block 'demapped' the DQPSK-modulated data from the DAB carriers by calculating the phase-difference between the carriers from consecutive symbols. This was done for all of the L sets of carriers provided, thus producing $(L-1)$ sets of data. This functionality is shown graphically in Figure 4.18.

4.3.5 Frequency Deinterleaving

This sub-block, termed `freq_deinterleave`, was created to receive $(L-1)$ interleaved sets of DQPSK-demodulated data, and return the same data but deinterleaved according to a given map. In the previous chapter, the problem of frequency-selective fading was discussed, along with the DAB standard's mitigation strategy of interleaving the OFDM carriers in a particular way. The precise motivation and details of the specific map used for interleaving are given in ETSI's document in [1]. Note that the input data, the output of the `dqpsk_demap` block, included all T_u carriers, including those which were outside the band of the central K carriers. In performing the frequency deinterleaving process, this function also removed the out-of-band carriers, leaving only the K relevant carriers.

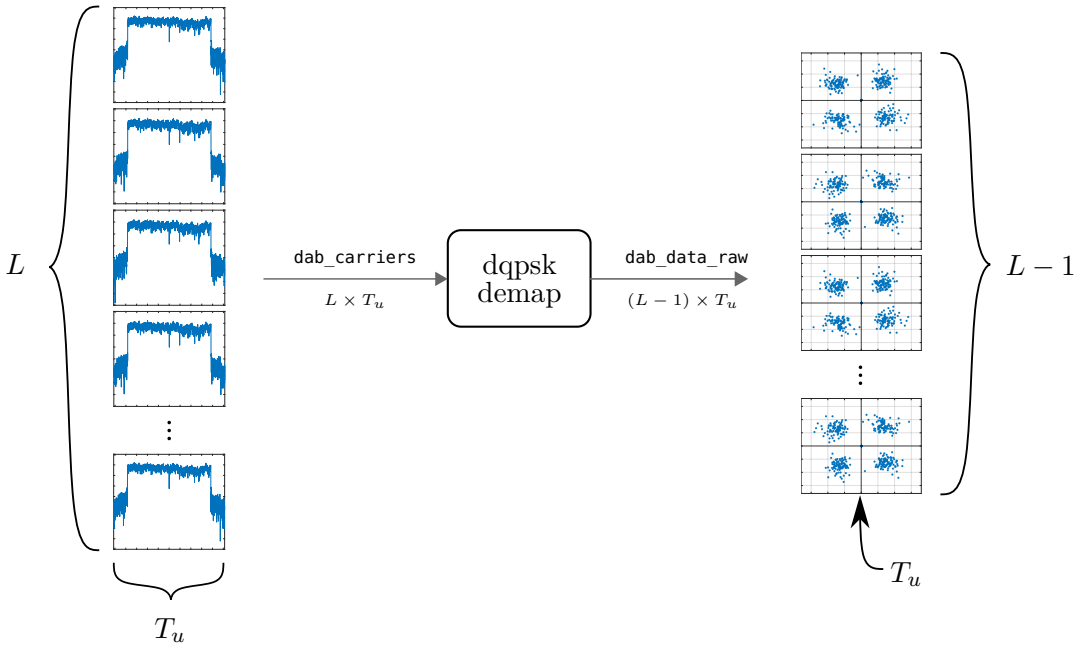


Figure 4.18: Graphical summary of the `dqpsk_demap` function.

For convenience, a helper function was written in MATLAB, called `build_interleave_map()`, which returned an array mapping the K indices of an array to K corresponding interleaved indices. Importantly, note that the mapping used differs for each of the four DAB transmission modes. Since this project was designed using Mode I signals, the code was implemented with Mode I's interleaving map, and with $K = 1536$. Future work should be done to enable this functionality for all four DAB modes.

Once the interleave map was created, deinterleaving the frequency components was trivial, as it simply entailed indexing the MATLAB matrices appropriately. The `freq_deinterleave` function is given in Listing 4.4. Notice that this code is independent of the DAB mode used, relying only on a generic interleaving map.

```
function dab_data_deinterleaved = freq_deinterleave(dab_data_raw, interleave_map)
  dab_data_deinterleaved = dab_data_raw(:,interleave_map); % Use interleave map for indexing
end
```

Listing 4.4: MATLAB code for the frequency deinterleaving functionality.

When visualized, the deinterleaving of the frequency components is not particularly insightful, as the large number of sub-carriers involved can make the plots messy. Consider, for example, the plot of the carriers' phases for a real-world DAB symbol in Figure 4.19.

Even the plot of the interleaving map for Mode I looks somewhat arbitrary, as seen in Figure 4.20.

Nonetheless, the graphical summary for the `freq_deinterleave` function is given in Figure 4.21.

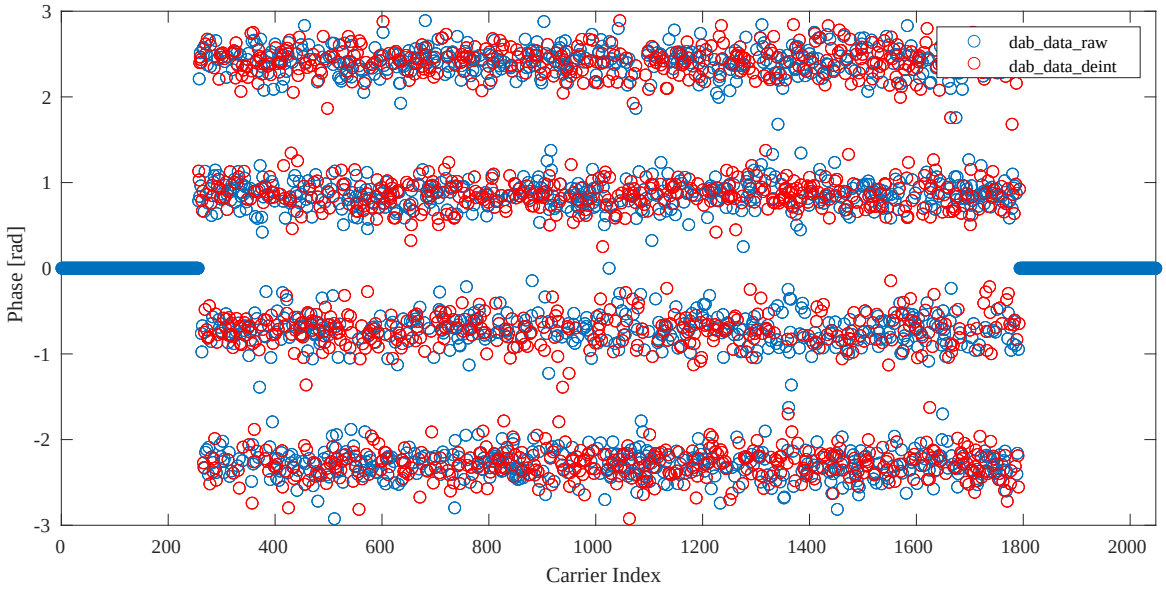


Figure 4.19: Phase plot for the carriers of a real-world DAB symbol, before and after frequency deinterleaving.

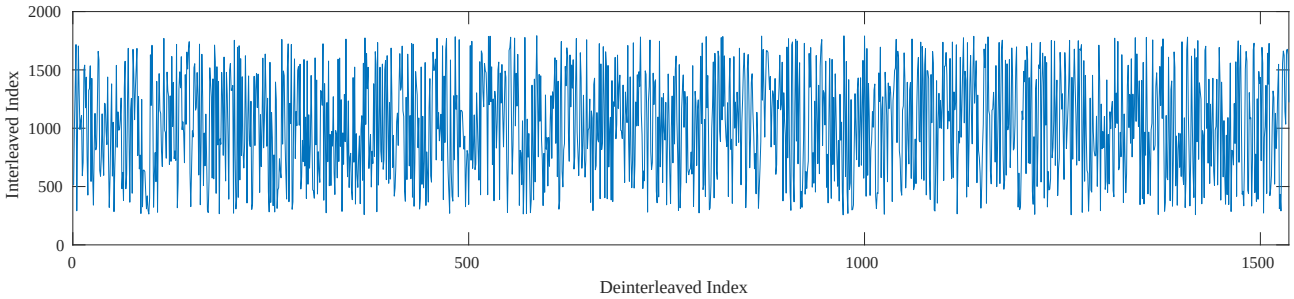


Figure 4.20: Interleave map for DAB Transmission Mode I.

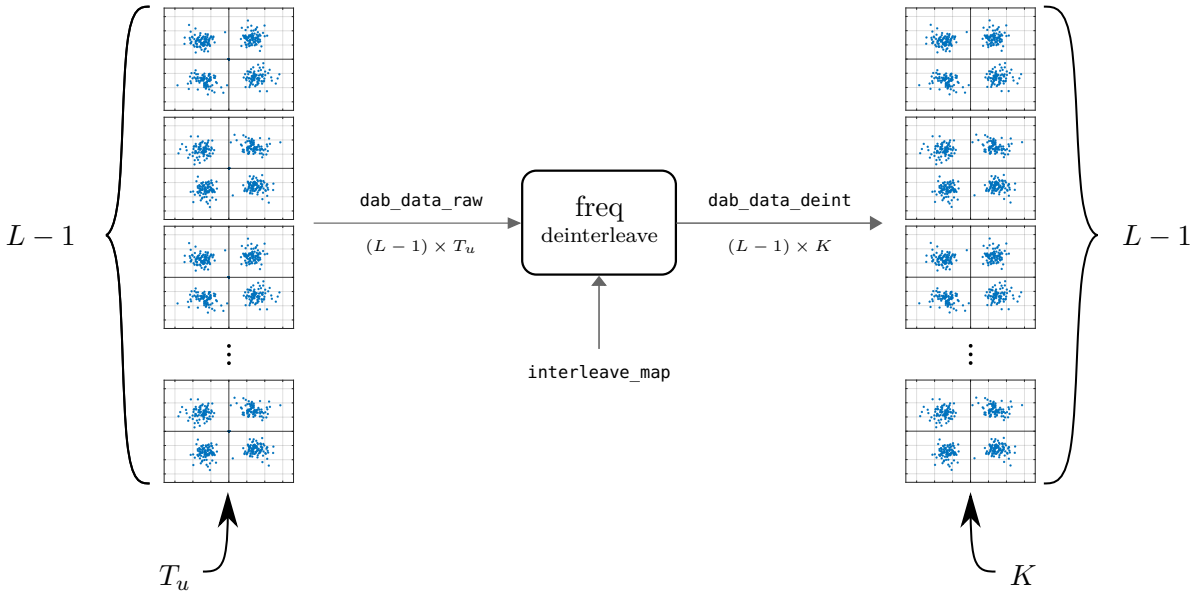


Figure 4.21: Graphical summary of the `freq_deinterleave` function.

4.3.6 DQPSK Snapping

As seen in Figure 4.16c and Figure 4.17c, the `dqpsk_demap` sub-block did well to distinguish four groups of differential phase values from the OFDM carriers, despite being provided with input data that was fairly spread-out. Nonetheless, the spread of the data within these groups was still a problem—the groups each needed to be assigned to one of four values. Fortunately, provided one can assume that the original, transmitted signal was modulated correctly according the DAB standard in [1], the four possible *correct* phase-difference angles were known *a priori*. Those being, $\{-\frac{3\pi}{4}, -\frac{\pi}{4}, \frac{\pi}{4}, \frac{3\pi}{4}\}$. Since only the phase is modulated, the magnitude of the carriers should remain constant from symbol to symbol, and thus the magnitude of the complex ratio between any two symbols should be 1. In rectangular co-ordinates, then, the four possible points correspond to the values, $\pm\frac{1}{\sqrt{2}} \pm j\frac{1}{\sqrt{2}}$.

The easiest way to correct the spread of the DQPSK symbols was to re-assign them with the value that was closest to them out of the four possible points. Due to their angles, the four points fall exactly in the centre of each of the four quadrants respectively. Thus, if a given point was in quadrant X, it was guaranteed to be closest to the correct DQPSK point in that quadrant. Of course, this method of correcting real-world DAB data was not perfect, in that it did not apply any robust error-correction techniques to the data, and it likely assigned some edge cases incorrectly. Nonetheless, it was a quick and easy way to extract meaningful DQPSK data. This process was termed 'snapping', as it snapped given points to their closest possible values. Figure 4.22 shows an example symbol from real-world data (perfect data need not be snapped, as it is perfect already), where the four groups from the DQPSK-demapping fell within the four quadrants on a complex plane. Any given point, then, was simply snapped to the value for that quadrant. The groupings are shown by different colours, and the snapped values are shown by the large blue crosses.

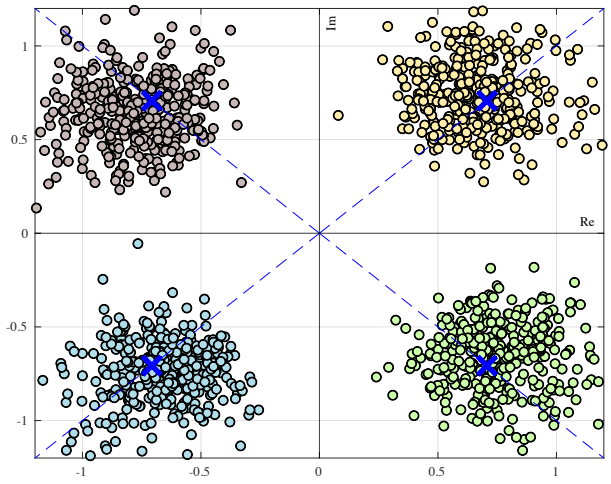


Figure 4.22: Graphic showing the assignment of DQPSK data to one of four known values, based on the quadrant in which the data lies.

The same data points are plotted against their corresponding carrier indices in Figure 4.23, with the snapped value lines and decision boundaries both shown. This is simply an alternative perspective of the previous plot.

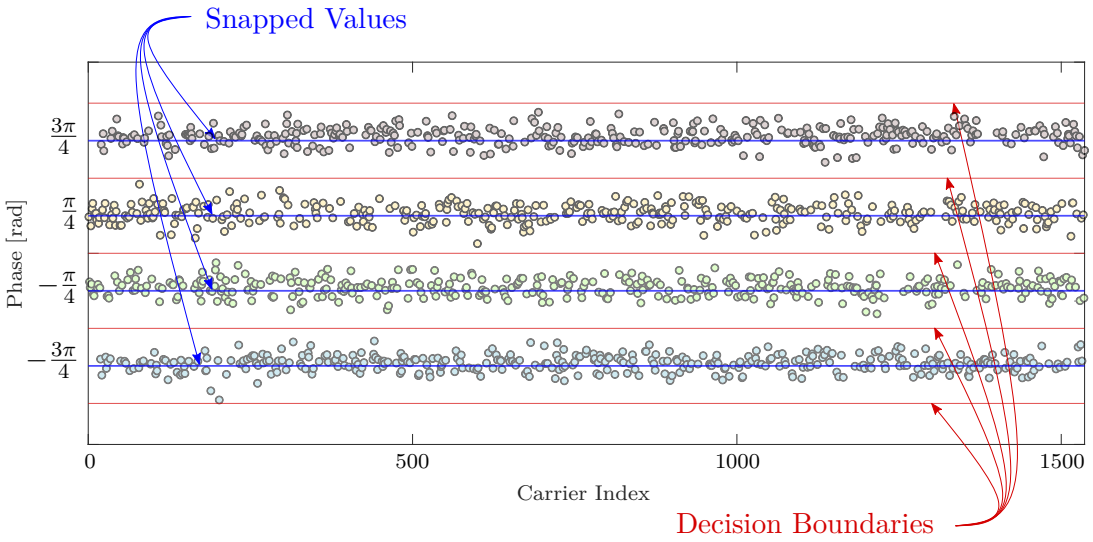


Figure 4.23: Alternative plot of the DQPSK snapping situation, viewed across the K carriers, showing the snapped values and decision boundaries.

In MATLAB, this snapping action was done by simply checking in which quadrant a point was, and assigning it to its correct value accordingly. For the function, this was performed for all points in the input matrix.

A graphical summary of the `dqpsk_snap` function is shown in Figure 4.24.

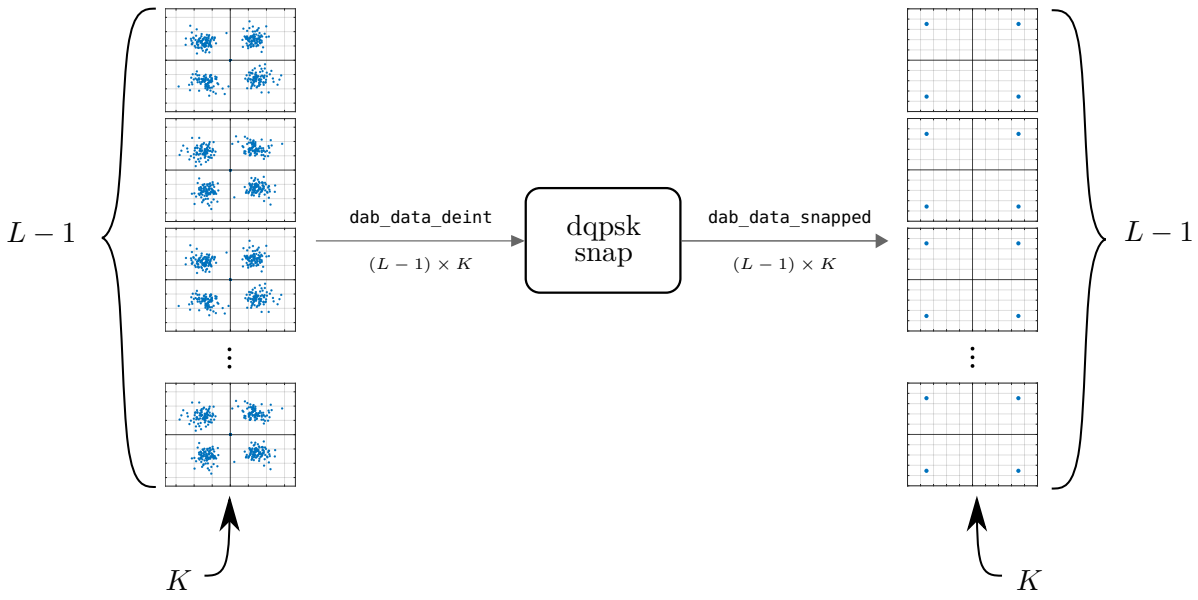


Figure 4.24: Graphical summary of the `dqpsk_snap` function.

4.3.7 Error Correction

Though the snapping of the DQPSK values was a straightforward way to eliminate the spreading in the phase data, it could not account for any errors that may have arisen in the signal. These errors could have come from various sources, including environmental noise and multipath effects. Moreover, some of the values in real-world data fell close to the decision boundary, and may have been assigned incorrectly. The DAB standard provides for this by integrating an error-correction method into any DAB signal, as mentioned briefly in the previous chapter during the discussion around Forward Error Correction (FEC). Essentially, the transmitted data is convolutionally encoded, and can be decoded on the receiver's end with a Viterbi decoder.

Having said that, the original scope of this project did not include any error-correction implementation, and the sub-block is included here for completeness only. Ideally, in a fully-functional chain that is integrated into a PR system, such functionality would indeed be included; however, its necessity depends on the quality of the DAB signals used in the system.

Conceptually, this sub-block would receive snapped DQPSK data with the dimensions $(L-1) \times K$, and then output data with the same dimensions, after adjusting the possibly-erroneous values if necessary.

4.4 Remodulation

4.4.1 Overview

The purpose of this block was to take the data output from the demodulation block, and modulate it anew, back into a DAB frame. Since the demodulated DQPSK data was snapped, the output of the remodulation block was a perfectly reconstructed DAB signal[4]. Once again, this block was somewhat nugatory on its own, as it was specifically designed to be used in conjunction with the demodulator block.

Figure 4.25 shows a block diagram for the remodulation block, with the various sub-blocks, nomenclature, and dimensions shown.

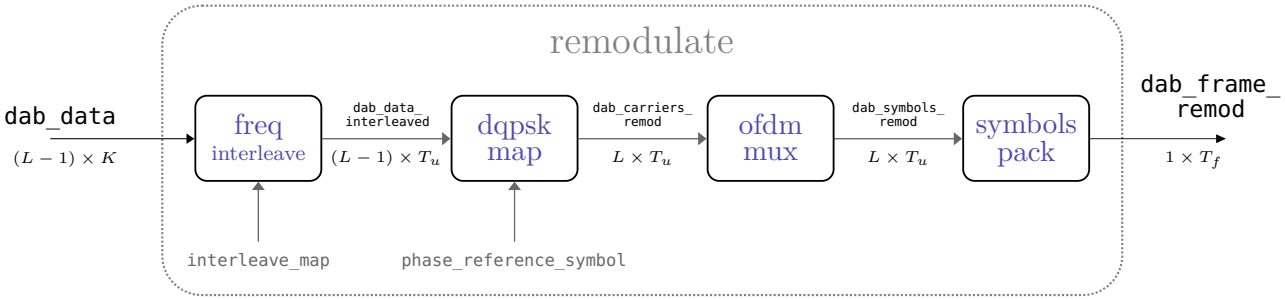


Figure 4.25: Block diagram showing remodulation section of processing chain.

[4]Owing to possible errors in the received signal, the reconstructed signal might not match the original, transmitted signal perfectly; instead, it is 'perfectly reconstructed' in the sense of being a perfect DAB signal.

The remodulation process had four key stages: firstly, the data points from the demodulation chain were interleaved in frequency, and then were mapped to a set of DAB carriers via DQPSK modulation. Thereafter, the carriers were multiplexed into OFDM symbols, and were finally packed into a single DAB frame.

Notice that these four sub-blocks in the remodulation chain were designed as inverses of the first four sub-blocks in the demodulation chain, shown previously in Figure 4.7. For example, the first step in the demodulation block was `symbols_unpack`, while the last step in the remodulation block was `symbols_pack`, and so on. In fact, if perfect data was put through the chain—such that the `dqpsk_snap` and `error_correction` sub-blocks had no effect—the `demodulate` and `remodulate` functions would be perfectly symmetrical. This design choice was intentional, as it enabled a logical flow of operations, forwards and backwards through the chain, and also allowed individual blocks to be tested with their inverses—which will be discussed further in the validation chapter.

The details for each sub-block are given in the following sections.

4.4.2 Frequency Interleaving

The purpose of this sub-block was to re-interleave the data from the `demodulate` block, essentially to undo the effects of the `freq_deinterleave` function. This process was termed frequency interleaving.

Interestingly, the same interleaving map as that created for the `freq_deinterleave` functionality could be used here, only with a slightly different method of assigning the values. Whereas the previous function used the map to index elements on the right-hand side of the assignment, this function used the map to index elements on the left-hand side—see Listing 4.5 for the code showing this. In this way, the same `build_interleave_map` tool could be used for both the interleaving and deinterleaving functionality. Note, however, that in the frequency interleaving function required knowledge the DAB mode, in order to pre-allocate the arrays correctly.

```
function dab_data_interleaved = freq_interleave(dab_data, interleave_map, dab_mode)
  dab_data_interleaved = zeros(dab_mode.L-1,dab_mode.Tu);   % Pre-allocate space
  dab_data_interleaved(:,interleave_map) = dab_data;        % Use interleave map
end
```

Listing 4.5: MATLAB code for the frequency interleaving functionality.

In terms of dimensions, the sub-block received $(L-1)$ sets of K carriers, which it then transformed to $(L-1)$ sets of T_u carriers, with the middlemost and out-of-band carriers reintroduced with a value of 0. As was the case for frequency deinterleaving, visualizing the interleaving process was not particularly helpful—see Figure 4.20 for a reminder of the messy interleaving map for Transmission Mode I. Nonetheless, the graphical summary of the functionality is given in Figure 4.21.

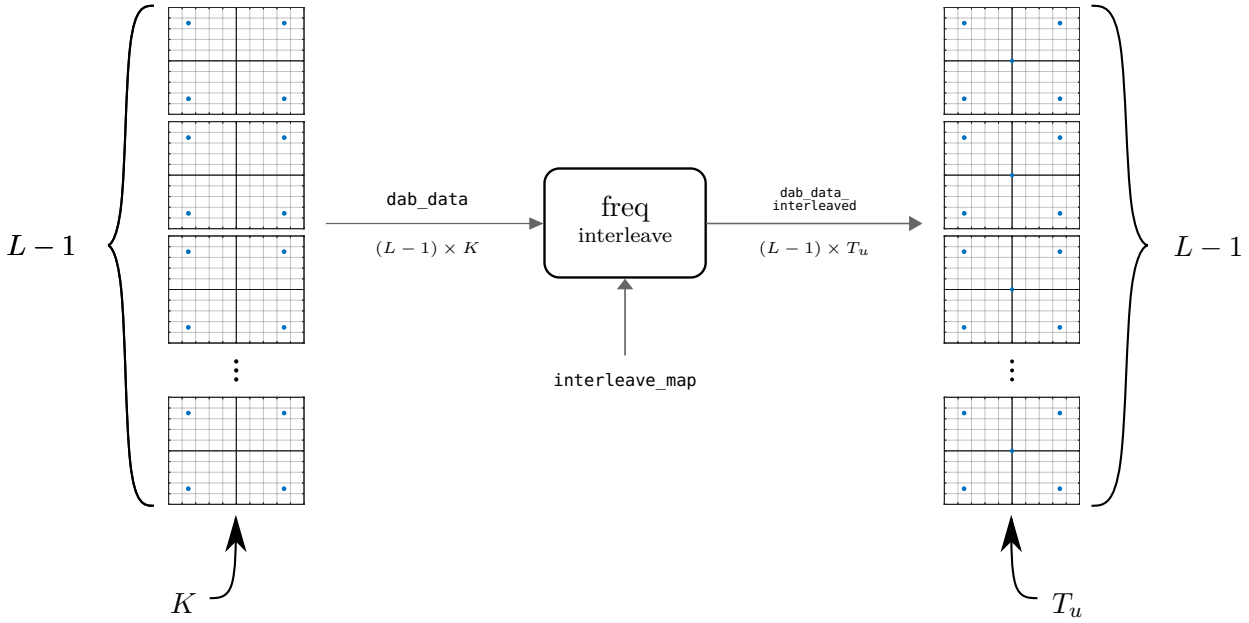


Figure 4.26: Graphical summary of the `frequency_interleave` function.

4.4.3 DQPSK Mapping

With the DAB data values re-interleaved from the previous block, they were then used to modulate the T_u OFDM carriers—a process termed DQPSK mapping, in contrast to the demapping process discussed previously. Since differential modulation was used, the $(L-1)$ sets of DAB data at the input of the sub-block mapped to L sets of DAB carriers. To provide a reference for the differential phase values, the PRS was assigned to be the first output symbol—recall that this was the other purpose of the PRS, in addition to the frame synchronisation discussed previously. Consider Listing 4.6, which shows the function set-up, with memory allocation and the assignment of the first symbol.

```
dab_carriers_remod = zeros(dab_mode.L, dab_mode.Tu); % Pre-allocate space
dab_carriers_remod(1,:) = prs; % First carrier is the phase reference symbol
```

Listing 4.6: MATLAB code for setting up the `dqpsk_map` function.

With the first symbol then set, subsequent symbol values were calculated by multiplying the previous symbol by the interleaved DAB data values for that index. Programmatically, this process was simple, and is shown in Listing 4.7, where `dab_data_interleaved` was the output of the `freq_interleave` sub-block.

```
for l = 2:dab_mode.L
  dab_carriers_remod(l,:) = dab_data_interleaved(l-1,:) .* dab_carriers_remod(l-1,:);
end
```

Listing 4.7: MATLAB code for the actual `dqpsk_map` functionality of differential modulation.

Recall that the DAB data used in this sub-block ultimately came from the output of the demodulation

block, which had snapped the DQPSK values to one of four possible phase values, all with a magnitude of 1. As a consequence of this, the output of this `dqpsk_map` sub-block was now 'perfect,' much like the data plotted in Figure 4.10a—that is, the resulting OFDM symbols were now flat and rectangular in magnitude, and containing only four possible values in phase. Importantly, note that this was the case, even when using real-world data. Consider, for example, the illustration provided in Figure 4.27, where the input OFDM symbol was imperfect and had a high noise-floor; then, after being processed by the intervening blocks in the chain, the output OFDM symbol was cleaned-up, with a low noise-floor and a perfectly rectangular magnitude.

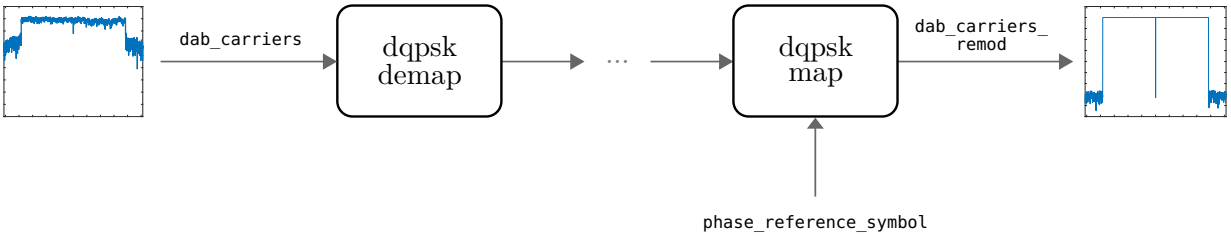


Figure 4.27: Demonstration of a section of the processing chain converting a noisy OFDM symbol into a clean one.

Graphically, the functionality of the DQPSK mapping is shown in Figure 4.28. For clarity, the `phase_reference_symbol` input is coloured green, which corresponds to the first output of the DAB carriers.

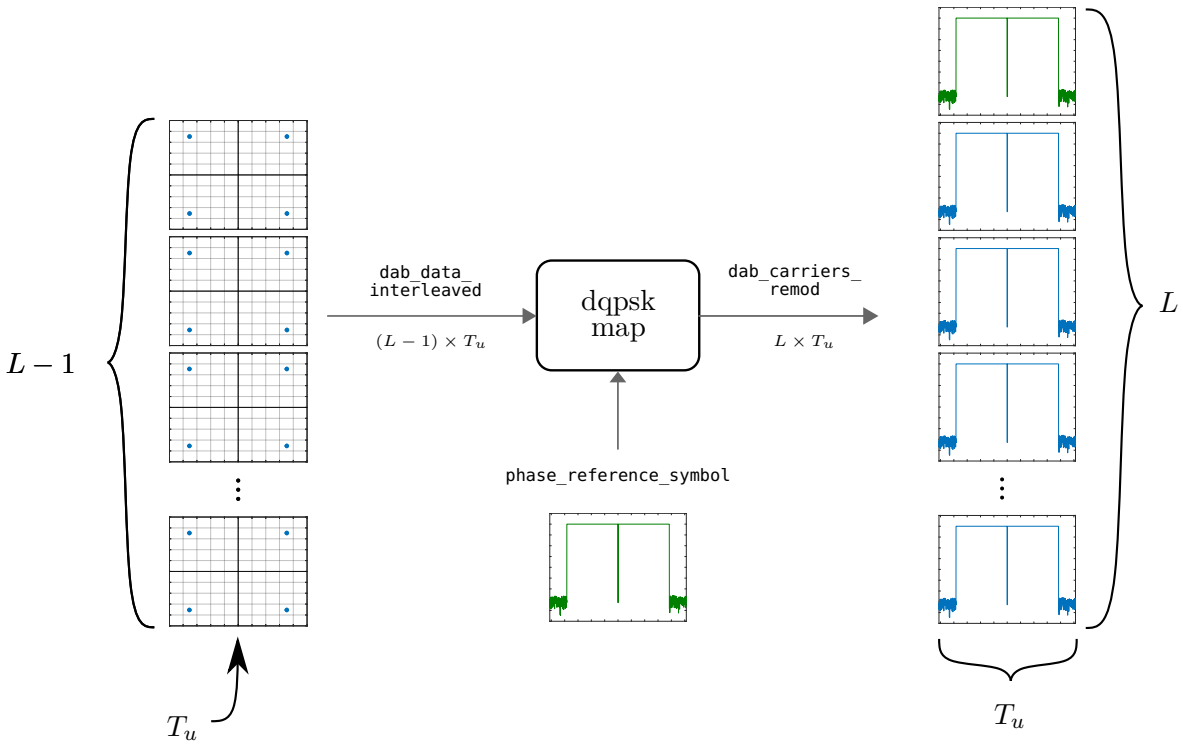


Figure 4.28: Graphical summary of the `dqpsk_map` function.

4.4.4 OFDM Multiplexing

The purpose of this sub-block was to convert the modulated OFDM carriers into time-domain signals, now reconstructed as perfect DAB signals. As explained in the OFDM *de*-multiplexing subsection, these two perspectives of the signal were, in fact, the same. However, since the data was being transformed from multiple signals from which data could easily be extracted (the OFDM carriers), to single signals containing the sum of many carriers, this process was broadly termed 'multiplexing.'

As was the case for demultiplexing, the DFT/FFT could be used for easy extraction and reconstruction of the OFDM symbols. Whereas the previous function used a normal, forward FFT, this function used the Inverse Fast Fourier Transform (IFFT)—which is logical, considering the inverse relationship of the two blocks: one multiplexes, the other demultiplexes.

In code, this functionality looked mostly identical to that for the demultiplexing, shown previously in Listing 4.2. Here, in Listing 4.8, the only differences were that the IFFT was used, and the result was *multiplied* by the length of the array, rather than being divided by it. Once again, this is logical, considering the inverse relationship.

```
1 dab_symbols_remod = ifft(fftshift(dab_carriers_remod,2),[],2) .* length(dab_carriers_remod);
```

Listing 4.8: MATLAB code for multiplexing an OFDM symbol

Since the IFFT is a $\mathbb{C}^N \to \mathbb{C}^N$ transformation, the `ofdm_mux` function took an input of $L \times T_u$ carriers, and output L symbols, each of length T_u. A graphical summary of this functionality is provided in Figure 4.29.

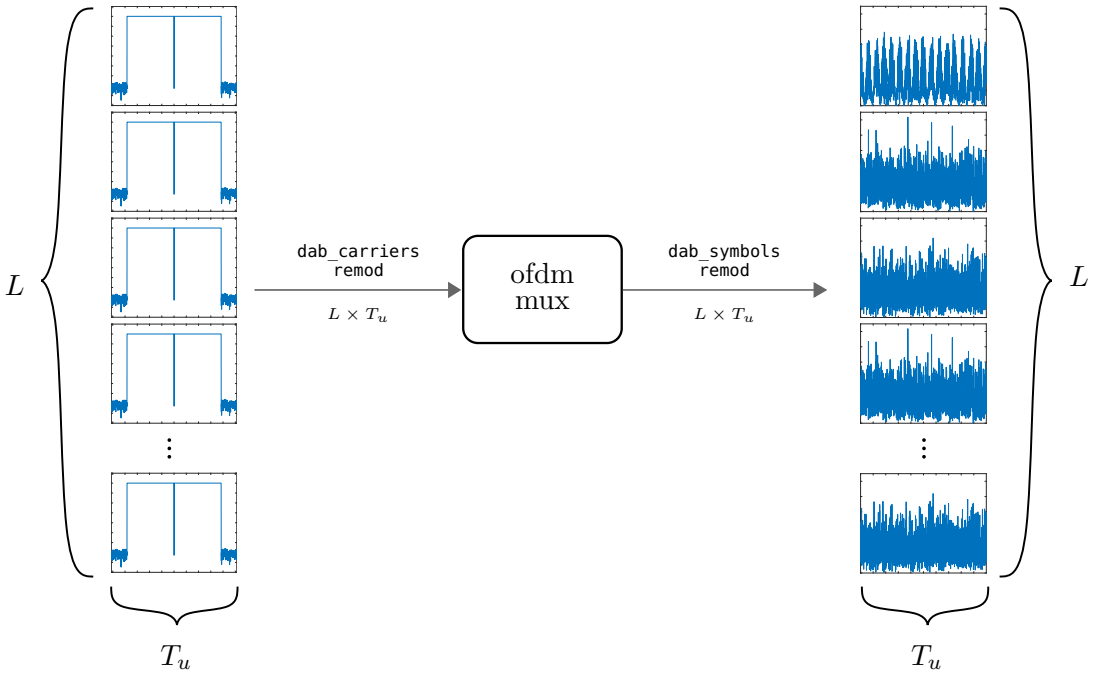


Figure 4.29: Graphical summary of the `ofdm_mux` function.

4.4.5 Symbols Packing

Finally, the purpose of this function was to take the matrix of remodulated DAB symbols, of size $L \times T_u$, and transform it into a single array of the reconstructed DAB frame, with a total length of T_f. In doing this, the sub-block was also responsible for adding the guard intervals and the null symbol.

Combining the guard interval for a particular symbol with the symbol itself was straightforward, as is shown in Listing 4.9, where `l` is the symbol index, and `dab_symbols_remod` is the data coming from the `ofdm_mux` sub-block

```
lth_guard = dab_symbols_remod(l, end - dab_mode.Tg + 1 : end); % Last Tg values
lth_symbol = [lth_guard, dab_symbols_remod(l,:)]; % Full Symbol = [Guard , Original Symbol]
```

Listing 4.9: MATLAB code for prepending a guard interval to a symbol.

Packing the symbols then was a trivial process via a simple for-loop. Including the null symbol entailed simply padding the front of the array with T_{null} zeros.

Graphically, this function is summarised in Figure 4.30.

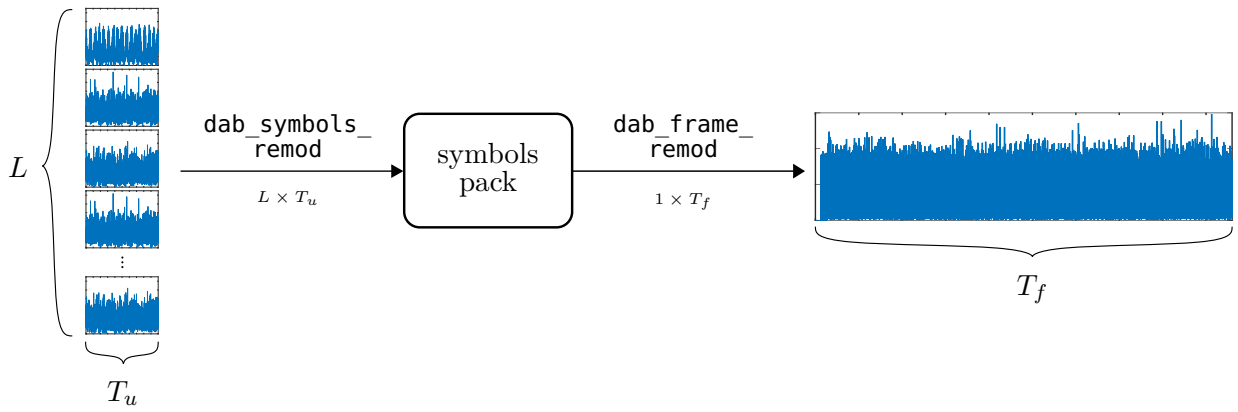


Figure 4.30: Graphical summary of the `symbols_pack` function.

4.5 Summary

This chapter thoroughly explored the design of a DAB processing chain, starting with a high-level overview of the system via block diagrams, and later moving into the details of each block and sub-block. Plots were provided where appropriate, both to demonstrate the functionality of the code, and to provide a summarised graphical reference of each block, to be used when working with the chain. Short code snippets were given where necessary, in order to convey the essence of certain required steps. Longer form algorithms were provided for the frame synchronisation methods, since these were non-trivial.

Chapter 5

Digital Audio Broadcasting: Processing Chain Validation

> For my part, whatever anguish of spirit it may cost, I am willing to know the whole truth, to know the worst, and to provide for it.
>
> —*Patrick Henry*

5.1 Overview

This chapter aims to report on the validation steps that were undertaken when designing the DAB processing chain. These steps formed an integral part of the design task, by both assisting with the debugging of errors while coding, and further ensuring that the final result achieved the project goals. At the end of the chapter, one should be confident that the created DAB chain indeed worked correctly.

The structure of the chapter follows logically with how the design process itself unfolded. It starts with a validation method that was fairly non-rigorous, but still useful for early-stage understanding and debugging: that being, graphical validation—simply looking at the features of the outputs at various points in the chain. Thereafter, since most the demodulation and remodulation sub-blocks were designed as inverses of each other, corresponding pairs of functions were tested together with random input data, in the so-called 'Inverse Relationship' validation tests. Thirdly, the entire chain was tested by running various datasets through the demodulation-remodulation process *twice*, and comparing the results from the first and second iterations—a correct chain would yield the same results after each. Ultimately, then, the entire chain was tested using provided reference data—real-world DAB signals and the perfect reconstructions thereof. Descriptions and results of these validation steps follow.

5.2 Graphical Validation

This initial validation step consisted of looking at the output values at various points in the processing chain, and considering them graphically. There were two important caveats to this approach. Firstly, it was not an infallible method of testing, as certain correct graphical features could have arisen even with an incorrect underlying algorithm. Secondly, it could not be done for all sub-blocks—only a handful of the chain's functionality lent itself to such analysis. Nonetheless, it was an important tool in the early

stages of the design process. The benefit of doing this validation technique was that problems would manifest close their source: for example, a problem with the demultiplexing of the OFDM carriers would be immediately visible in that sub-block's output, and could thus be fixed more rapidly.

5.2.1 Methodology

For at least three points in the chain, one could know *a priori* the correct shape or features of the output data: these points being, after the frame synchronisation step (involving the `prs_detect` function), after the demultiplexing of the OFDM symbols (involving both the `symbols_unpacking` and `ofdm_demux` functions), and after the demapping of the DQPSK data (involving the `dqpsk_demap` functions). The inverses of the latter functions were also considered, but are not covered here, as the verification procedures were almost identical.

Validating these sub-blocks' functionality, then, simply involved looking at their outputs and comparing them to what was expected. Some of these plots were already provided in the previous chapter, and shown to be correct, but a handful more will be provided here for clarity.

5.2.2 Frame Synchronisation

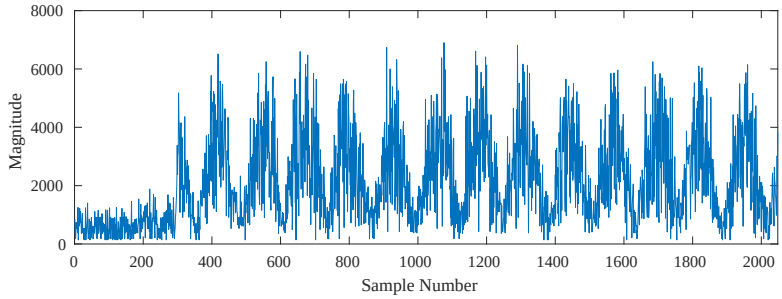
Recall that the frame synchronisation process involved using a matched filter to detect the location of a pre-defined PRS within a given DAB signal. Since the PRS is usually fairly distinct from the data-carrying symbols, and because each frame is always preceded by a NS, it was easy to recognize the beginning of a frame visually. Thus, by comparing the output index from the `prs_detect` function to a plot of the original data, one could quickly see if the algorithm was working correctly. As a reminder, this was not a flawless mode of validation, but it provided good initial insight.

The plot in Figure 4.5 from the previous chapter has already shown the matched filter working on perfect data. Graphically, one can easily see that the PRS was detected, after the guard interval. It was also via graphical intuition that the problem shown in Figure 4.6 was discovered.

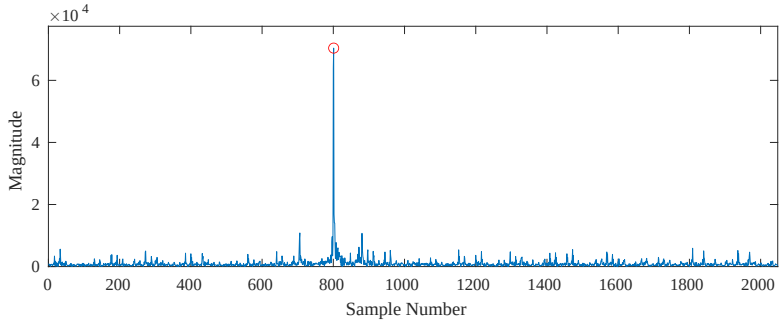
For two further examples, Figure 5.1 and Figure 5.2, given here, show the detection process working with real-world data. Importantly, the latter of the two figures displays a situation where the input data was not sampled at $F_s = 2.048\,\text{MHz}$, but rather at $F_s = 2.5\,\text{MHz}$. Thus, inadvertently, this plot additionally demonstrated the correct functioning of the `iq_resample` sub-block, since the matched filter was designed around the PRS, which was sampled at $F_s = 2.048\,\text{MHz}$. If the resampling was not working, neither would the PRS detection process.

5.2.3 Symbols Unpacking & OFDM Demultiplexing

The next two sub-blocks considered were `symbols_unpack` and `ofdm_demux`. Both were fairly simple blocks, but the validity of the latter's output relied heavily on the validity of the former's correct functionality. Recall that the OFDM demultiplexing stage received the DAB data, after it was split into L symbols, and the guard intervals were removed. The sub-block then performed a FFT on the

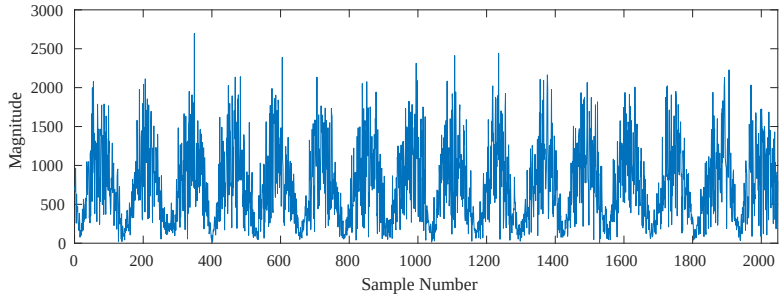


(a) Input to matched filter

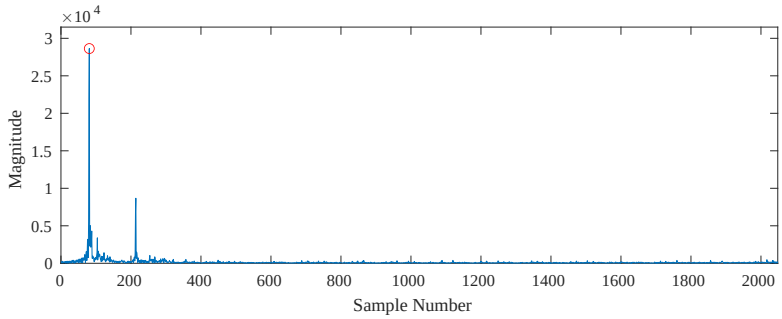


(b) Output from matched filter with peak detected

Figure 5.1: Plots demonstrating correct functioning of the `prs_detect` sub-block using real-world data.



(a) Input to matched filter

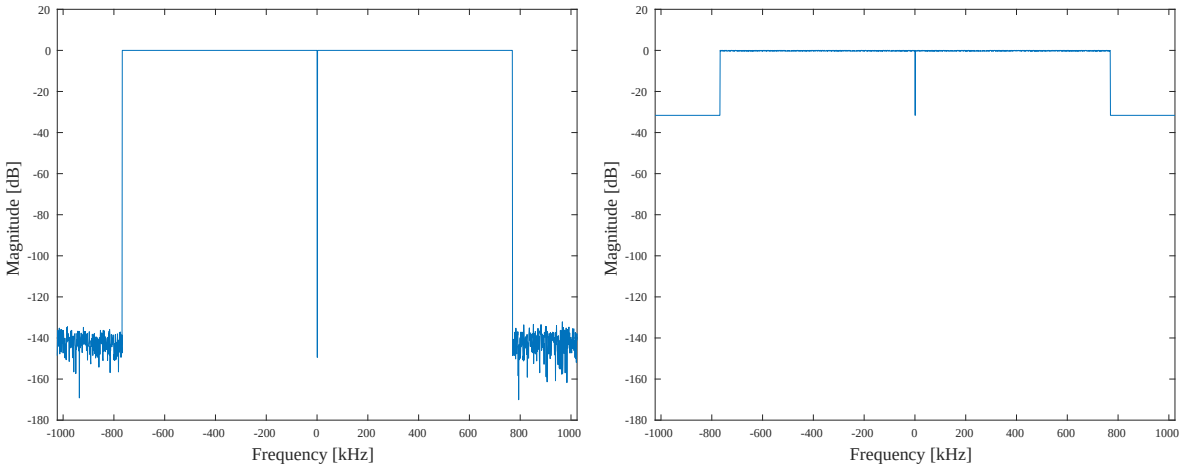


(b) Output from matched filter with peak detected

Figure 5.2: More plots demonstrating correct functioning of the `prs_detect` sub-block, but using real-world data that was sampled at $F_s \neq 2.048\,\text{MHz}$.

data, thus extracting the DAB carriers. The plot given in Figure 4.10a, from the previous chapter, shows an example magnitude plot of these carriers for perfect data, where the spectrum is clearly rectangular and flat in the passband. This implies that the symbols were correctly unpacked.

However, it was worth understanding how such plots would change were the DAB symbols unpacked *incorrectly*. Suppose the given DAB frame began at sample n_0, as detected via the PRS, but the `symbols_unpack` function erroneously misaligned the symbols during its unpacking process. The resulting magnitude plots of the DAB carriers of a single symbol are shown in Figure 5.3 for two situations—one where the symbols have been aligned at $(n_0 - 1)$ samples, the other at $(n_0 + 1)$ samples.



(a) Symbol aligned at $(n_0 - 1)$ samples

(b) Symbol aligned at $(n_0 + 1)$ samples

Figure 5.3: Plots showing the magnitude of the OFDM carriers for a misaligned DAB symbol.

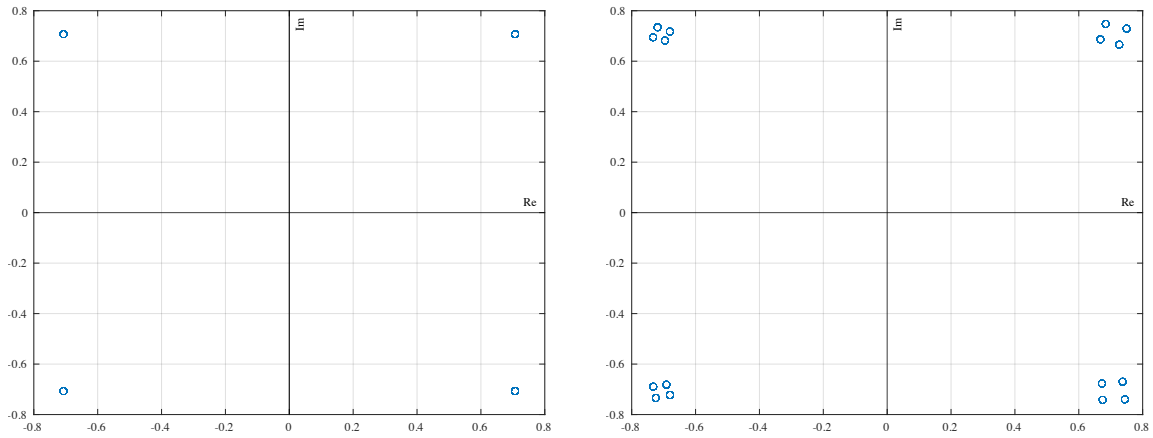
Notice how aligning the symbol prematurely at $(n_0 - 1)$, as shown in Figure 5.3a, produced an identical output to that shown before, in Figure 4.10a, where the passband was flat and the out-of-band noise-floor was virtually zero. However, when the symbol was aligned incorrectly at $(n_0 + 1)$, as shown in Figure 5.3b, the noise-floor increased by over 100 dB. As the symbol became more misaligned in the 'forward' direction, the resulting OFDM carriers became increasingly imperfect. On the other hand, there was leeway for the symbol to be aligned prematurely, without any effect on the resulting functionality.

This effect was due to the guard interval. Remember that a guard interval *precedes* each DAB symbol. Thus, quite simply, reading a symbol prematurely involved reading the original symbol and some of its guard interval. Reading the symbol too late, however, involved reading the original symbol, as well as the guard interval of the following symbol—which contained completely different data.

It is important to note that these effects were easily visualised when using perfect data, but not so much when using real-world data. The real-world OFDM carriers indeed looked somewhat rectangular in magnitude, as seen in Figure 4.10b, but the impact of symbol misalignment was far less pronounced. It is thus recommended that these sub-blocks be designed and initially tested using perfect data.

5.2.4 DQPSK Demapping

The demapping of the DQPSK data was already discussed extensively in the previous chapter, and Figures 4.15, 4.16 and 4.17 all demonstrated the success of DQPSK demodulation, which was able to split the data into four groups of angles, albeit spread out at times. Once again, though, it was worth considering the impact of symbol misalignment on the `dqpsk_demap` functionality—using perfect data for clarity. Building on the same two situations from the previous step, Figure 5.4 demonstrates the impact that misalignment had on the DQPSK values.



(a) Symbol aligned at $(n_0 - 1)$ samples

(b) Symbol aligned at $(n_0 + 1)$ samples

Figure 5.4: Plots of the complex plane with the values from the DQPSK demapping of a misaligned DAB symbol.

As was expected, aligning the symbol prematurely, as in Figure 5.4a, had no impact on the extracted DQPSK data; whereas, aligning the symbol belatedly, as in Figure 5.4b, spread out the resulting angles. This was, of course, the same effect of the guard interval as before.

5.3 Inverse Relationship Validation

An important part of the design rationale for the DAB processing chain was the idea of 'inverse relationships.' Essentially, for as many parts of the pipeline as possible, the sub-blocks were designed to have an inverse counterpart. For example, the `symbols_pack` function had an inverse in the `symbols_unpack` function. Zooming out, the `demodulation` and `remodulation` blocks were also somewhat inverted, with one taking a recorded DAB signal and producing DAB data, and the other doing the opposite. In fact, this demodulation-remodulation process was the essence of the entire project, with the additional intention that real-world data could be put in on one end, and a perfect reconstruction of this data would be returned. These inverse relationships provided a logical flow to the processing chain, but also provided a good way to test functionality.

5.3.1 Methodology

The essence of this validation process is shown in Figure 5.5, where f and f^{-1} are two sub-blocks that are being tested together.

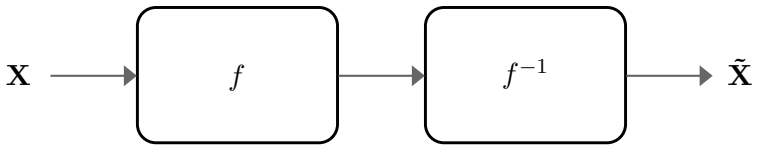


Figure 5.5: Depiction of the inverse relationship validation process for a generic pair of blocks.

The sub-blocks would be considered valid inverses if there was no net change on the input data; thus, an error signal was defined:

$$\mathbf{E} = \tilde{\mathbf{X}} - \mathbf{X} \tag{5.1}$$

Since this relationship should hold for any input $\mathbf{X}$, sets of pseudorandom data were provided, and even then, the data was to remain the same at the output. For a valid inverse relationship, then, the maximum value of $\mathbf{E}$ had to be zero (or extremely close to zero, accounting for numerical inaccuracies) for multiple iterations of arbitrary complex data, $\mathbf{X}$.

There are two important notes to make here. Firstly, as was the case for graphical validation, this method was not infallible. Showing that two given sub-blocks demonstrated the above relationship proved that their functionality was inverted, but it said nothing about the correctness of that functionality itself. The block performing $f^{-1}(\mathbf{X})$ could have been built around the same, incorrect underlying algorithm as $f(\mathbf{X})$, and the validation step would not reveal this truth.

Secondly, in some cases within the DAB chain, a function $f(\mathbf{X})$ discarded certain unnecessary information from $\mathbf{X}$. For example, the `freq_deinterleave` sub-block transformed $(L-1)$ rows of DAB data of length T_u, to $(L-1)$ rows of DAB data of length K, with $T_u > K$. Thus, there was a discarding of $(T_u - K)$ values in each of the $(L-1)$ rows. The inverse function, `freq_interleave`, performed the opposite transformation, and thus had to reintroduce these $(L-1) \times (T_u - K)$ data points. These values were initialized to zero. None of this mattered for the DAB functionality itself, since only the K values contained useful information. Nonetheless, testing the two blocks as inverses with pseudorandom data would fail, since the $(T_u - K)$ values in each row would be lost in the deinterleaving step. The solution to this caveat was either to look only at the relevant data in an output, or to test the inverses only in a particular direction—for example, this problem did not arise when putting random data in `freq_interleave` first, and thereafter into `freq_deinterleave`.

Nonetheless, testing these inverse relationships was a good step in validating the code written for the DAB chain. The following subsections show results for the four sub-block pairs, and thereafter, show an inverse test for the remodulation and demodulation blocks, with certain restrictions on the data used.

5.3.2 Symbols Packing & Unpacking

The inverse validation test for the `symbols_pack` and `symbols_unpack` functions is depicted in Figure 5.6.

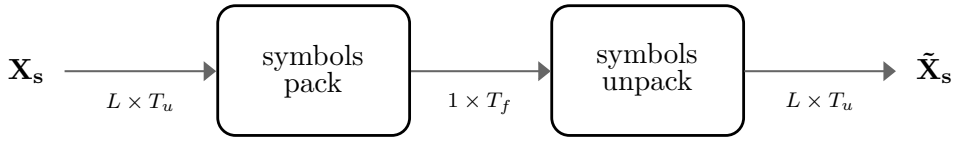


Figure 5.6: Depiction of the inverse relationship validation process for the `symbols_pack` and `symbols_unpack` functions.

Recall that the `symbols_unpack` function did discard some data by removing the guard intervals and the null symbol. Therefore, if this function was placed first in the testing pipeline, the input data used for $\mathbf{X_s}$ would have to be structured in a certain way for a correct inverse relationship. Instead, the order was chosen as shown in the figure: after a matrix of pseudorandom symbols was generated, they were first packed by the `symbols_pack` function (which included the insertion of guard intervals and the null symbol), and thereafter unpacked by the `symbols_unpack` function.

Over multiple iterations, the results of this test were:

$$|\max\{\mathbf{E_s}\}| = 0 \tag{5.2}$$

That is, the functions were *exact* inverses of each other. This makes sense, since the data was simply rearranged in the two sub-blocks, without any changes to the actual data values.

5.3.3 OFDM Multiplexing & Demultiplexing

The inverse validation test for the `ofdm_mux` and `ofdm_demux` functions is depicted in Figure 5.7.

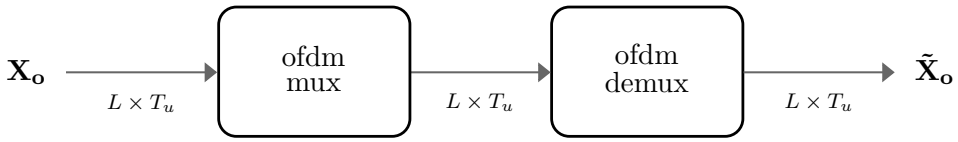


Figure 5.7: Depiction of the inverse relationship validation process for the `ofdm_mux` and `ofdm_demux` functions.

Due to the straightforward nature of these functions—simply a IFFT and FFT, respectively—they could also be tested in the opposite direction.

Over multiple iterations, the order of magnitude of these test results was:

$$|\max\{\mathbf{E_o}\}| \approx 10^{-13} \tag{5.3}$$

Notice that the difference between the input and output signals was not exactly zero, but was nonetheless very small. This was due to the numerical inaccuracies involving the use of double-precision floating-point values, together with multiplication, division, FFT and IFFT operations. Such an error was sufficiently small to be regarded as zero, implying that the functions were indeed inverses of each other.

5.3.4 DQPSK Mapping & Demapping

The inverse validation test for the `dqpsk_map` and `dqpsk_demap` functions is depicted in Figure 5.8.

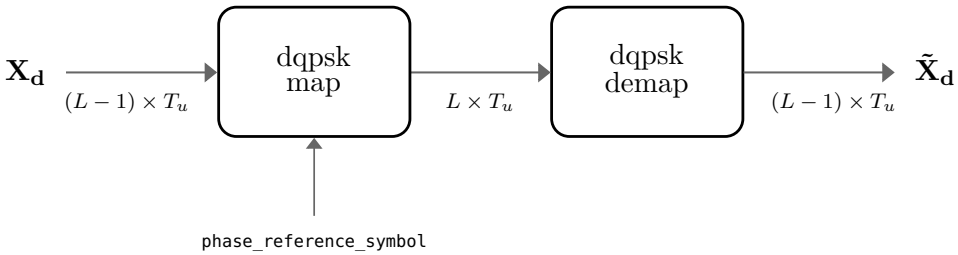


Figure 5.8: Depiction of the inverse relationship validation process for the `dqpsk_map` and `dqpsk_demap` functions.

In order to save computation time, the `dqpsk_demap` function was designed to demap only the central K carriers (barring the centremost carrier) of the provided OFDM symbols, despite being given all T_u carriers. As a consequence of this, the inverse relationship test could only be considered for these K carriers, rather than for the entire set of values. As mentioned previously, this was not a problem, since the ignored data was unimportant for the DAB chain.

Note that for additional testing, the pipeline from Figure 5.8 was also arranged such that $\mathbf{X_d}$ was first fed into the `dqpsk_demap` function, and then into the `dqpsk_map` function. However, in that case, the `phase_reference_symbol` provided to the latter function had to be set to the first row of the original input data, $\mathbf{X_d}$.

Across multiple iterations of both of these tests, considering only the relevant K values in each row, the order of magnitude of the maximum error was:

$$|\max\{\mathbf{E_d}\}| \approx 10^{-15} \tag{5.4}$$

Once again, there was a tiny error that existed between the input and output data. This was caused by a numerical inaccuracy involving the multiplication and division operations on the double-precision floating-point values. With such a negligible error, there was no doubt that these functions were indeed true inverses of each other.

5.3.5 Frequency Interleaving & Deinterleaving

The inverse validation test for the `freq_interleave` and `freq_deinterleave` functions is depicted in Figure 5.9.

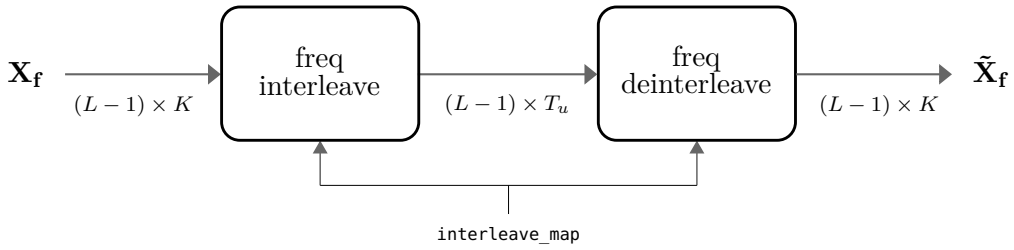


Figure 5.9: Depiction of the inverse relationship validation process for the `freq_interleave` and `freq_deinterleave` functions.

As discussed at the beginning of this section, the `freq_deinterleave` sub-block was designed to discard the out-of-band carriers, and thus the inverse relationship validation test had to be run in the direction shown, with `freq_interleave` placed first. Notice that the same `interleave_map` was used for both sub-blocks—recall that this ability was an intentional design decision for simplicity: both of the sub-blocks can use the `build_interleave_map()` utility function.

This validation test was run several times, and across all of these tests, the results were:

$$|\max\{\mathbf{E_f}\}| = 0 \tag{5.5}$$

In other words, the two functions were exact inverses. As with the `symbols_pack` and `symbols_unpack` test, this result makes sense, since the data was only being shuffled, not operated on.

5.3.6 Remodulation & Demodulation

The final inverse relationship test was between the system's main two blocks: `remodulate` and `demodulate`. There were a couple of important caveats when testing this relationship, though. Firstly, in order for the demodulation process to have worked correctly, the provided `dab_frame` must have had a specific structure—with the NS, PRS, and guard intervals all placed appropriately. Thus, an arbitrary, pseudorandom input could not be used as the `dab_frame`, for it would not have met such criteria. Moreover, recall that a crucial part of the demodulation functionality was the `dqpsk_snap` sub-block (together with the `error_correct` sub-block, which was not implemented in this project). By running arbitrary, imperfect data through the chain (even if it met the structural requirements of a DAB signal), the output data values would be snapped to one of the four correct DQPSK angles. Remodulation of the data, then, would produce a completely different result—rendering the inverse test pointless.

As a consequence, the test was set-up in a specific way: the data was first remodulated, and thereafter demodulated, rather than the other way around. Furthermore, the generated `dab_data` used as an input

was constrained to have four possible DQPSK angles—essentially making it a perfect DAB bitstream, and thus eliminating any action required by the `dqpsk_snap` and `error_correct blocks`.

A graphical depiction of the inverse validation test for the `remodulate` and `demodulate` functions is given in Figure 5.10, where $\mathbf{X_p}$ is pseudorandomly-generated DAB data meeting the constraints discussed above.

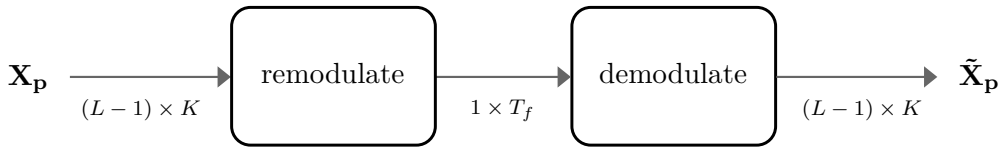


Figure 5.10: Depiction of the inverse relationship validation process for the `remodulate` and `demodulate` blocks, using pseudorandomly-generated DAB data.

Multiple iterations of the test were run, and the order of magnitude of the error was:

$$|\max\{\mathbf{E_p}\}| \approx 10^{-16} \tag{5.6}$$

This negligible value, again due to numerical inaccuracies, therefore confirmed that the demodulation and remodulation processes were indeed inverses, when working with perfect DAB data.

5.4 Repeated Processing Validation

Recall that the overarching purpose of the DAB processing chain was to reconstruct a provided DAB signal perfectly—not that the reconstruction was necessarily identical to the original transmitted DAB signal, but such that its OFDM carriers had a flat magnitude spectrum and its DQPSK values were limited to one of the four allowable angle values. Logically, then, if this 'perfect' signal was to be re-input into the processing chain, its output should be identical to its input, since no changes would need to be made to the signal. Ensuring this fact to be true was the next step of validation, termed here as 'Repeated Processing.'

5.4.1 Methodology

The essence of this process is depicted in Figure 5.11, with two iterations[1] of the demodulation-remodulation process shown.

As is clear from the diagram, this validation step was extremely straightforward: it simply involved passing a recorded DAB frame, notated as $\mathbf{X}$, through the processing chain—specifically the demodulation and remodulation blocks—twice. Then, the outputs after each iteration were compared, with

[1] As a side note, if one can demonstrate this validation process successfully for two iterations, then by mathematical induction, one can prove it to be true for n iterations, $\forall\, n \in \mathbb{N}$.

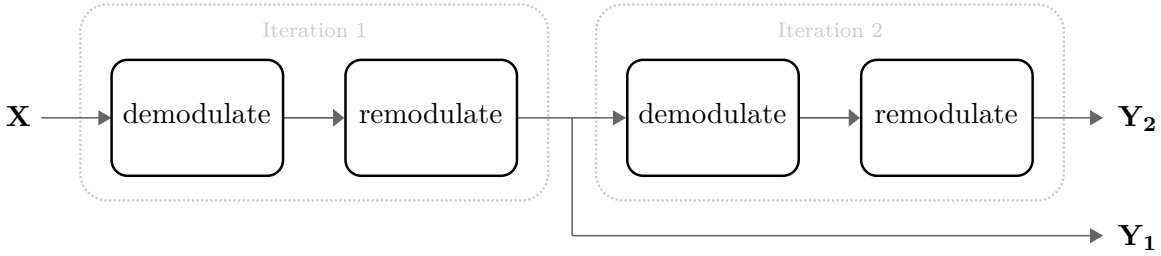


Figure 5.11: Depiction of the Repeated Processing validation method.

the intention that they be identical. The error value for this test was defined as:

$$\mathbf{E_Y} = \mathbf{Y_2} - \mathbf{Y_1} \tag{5.7}$$

A valid chain required all values of $\mathbf{E_Y}$ to be zero.

Note, once again, that this test was not infallible. Rather than proving that the system necessarily worked correctly as a DAB processing chain, it instead showed that the chain was *consistent*: a perfect signal generated once would be regenerated if processed again. This was indeed an important step to validate.

5.4.2 Results

This validation test was run on a variety of data, from both perfect and real-world signals, across several frames for each. For every single one of these cases, the result was as follows:

$$|\max\{\mathbf{E_Y}\}| = 0 \tag{5.8}$$

That is, the output signals from the first and second iterations of the chain were identical for every test. On the surface, it may be peculiar that no numerical inaccuracies crept into the results, since such errors did arise in previous validation tests. Nonetheless, this makes sense: the same rounding-off errors that arose via the first iteration's calculations arose identically in those of the second iteration. Either way, these results demonstrate that the designed DAB processing chain was indeed consistent.

5.5 Reference Data Validation

Some of the data provided for this project was termed 'perfect' data. Essentially, this data had already been processed by another DAB chain, and was thus known to be a perfect DAB reconstruction of some signal. The ultimate test of this designed processing chain, then, was comparing the output of the chain when given the perfect input, to the perfect input data itself. This validation step was the most robust of all the tests done, as it compared the chain's output directly to a known DAB processing chain's output. It was also the least helpful in the case of debugging an error, as there would be so many places for an error to arise; thus, it was saved for last. This step was simply termed 'Reference Data' validation.

Note that this test was subtly different to the Repeated Processing test, done previously. In the latter, the demodulation-remodulation chain was used to generate perfect data, and then used again, to check if the same perfect data was regenerated. On the other hand, this test compared the perfect reference data created by an independent system that was known to be correct, to the perfect reference data created by this chain.

5.5.1 Methodology

A depiction of this validation step is given in Figure 5.12, where $\mathbf{X}$ is the original DAB frame, $\tilde{\mathbf{X}}$ is the output of this project's processing chain for the given frame, and $\hat{\mathbf{X}}$ is the provided reference data for that frame—a reconstruction of the DAB signal that was known to be correct. Since perfect data is being used for this test, $\mathbf{X} = \hat{\mathbf{X}}$. Non-perfect data could be used as well, provided its reference data was available.

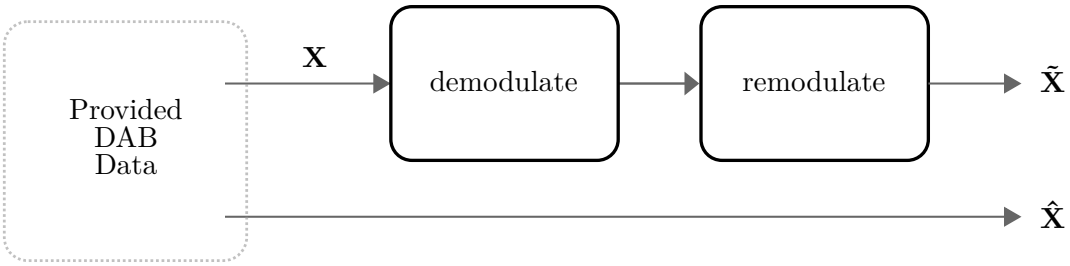


Figure 5.12: Depiction of the Reference Data validation method.

As done before, an error signal was defined as the difference between this chain's output and the reference data:

$$\mathbf{E_R} = \tilde{\mathbf{X}} - \hat{\mathbf{X}} \tag{5.9}$$

A valid chain would have the values of $\mathbf{E_R}$ equal to zero, or close thereto. It is worth noting that the reference data provided was calculated using single-precision floating-arithmetic, in contrast to the double-precision alternative used for this project's chain. Therefore, naturally, some small errors were bound to arise.

Unlike the previous validation situations, a valid result from this test would indicate the correct functioning of the DAB processing chain as a whole, not only in some limited way. Hence, it was a crucial test.

5.5.2 Results

This test was run across several frames of the signal, all of which had a similar result. Figure 5.13 shows the magnitude of the error signal for one of the provided DAB frames. Importantly, notice the scale on the vertical axis.

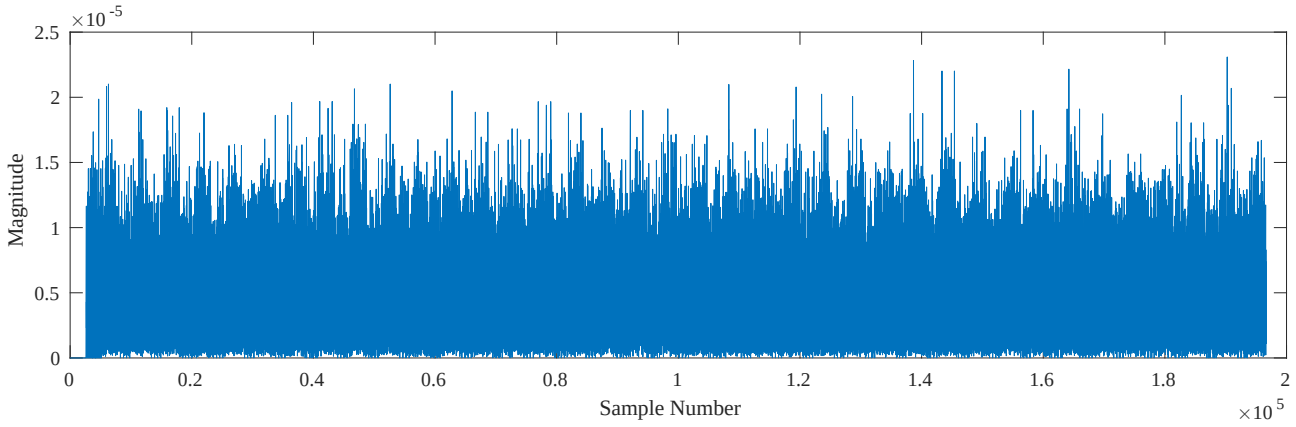


Figure 5.13: Plot of $|\mathbf{E_R}|$ for a provided perfect DAB frame.

The order of magnitude of the error signal across the various frames was consistently as follows:

$$|\max\{\mathbf{E_R}\}| \approx 10^{-5} \tag{5.10}$$

The magnitude of this error was consistent with what was expected, considering that the provided data was calculated using single-precision floating-point values. It was sufficiently small to be regarded as negligible.

In light of this test, it was concluded that the designed DAB processing chain worked correctly when given perfect data.

5.6 Summary

This chapter presented a series of four validation tests that were performed on the designed DAB processing chain. Firstly, certain sub-blocks were considered graphically, using the approximate idea of what a correct output would look like for them. Secondly, many of the sub-blocks were tested using their 'inverse relationships', where opposite functions in the demodulation and remodulation blocks were considered together, to check if they were truly inverses of each other. Thirdly, the entire chain was then tested by running data through it *twice*, and comparing the outputs of the first and second iterations. Finally, perfect data was provided to the chain, and the output was compared to the input. All of these tests were passed, and there is thus convincing evidence that the designed DAB chain indeed functions correctly.

Chapter 6

Passive Radar Integration

Oh, to see without my eyes.

—*Sufjan Stevens*

6.1 Overview

This brief chapter serves to provide a rudimentary insight into the context for which the DAB processing chain was designed: integration into a PR system. The scope of this project did not facilitate a thorough analysis into such integration, and the details surrounding its performance, accuracy, and efficiency were not considered in any way. Instead, the work done for this chapter was purely illustrative, included to demonstrate the bigger picture to the reader, in a way that omits dense mathematical analysis.

6.2 System Description

A conventional, bistatic PR system requires two signals for its operation: a reference signal, and a surveillance signal. The former is the signal that arrives via the direct-path, from the uncooperative transmitter to system's receiver. The latter is the signal that comes from the scene, as a recording of the scatterings and reflections of the EM energy. However, when a *digitally*-modulated signal is used—one which has an open and accessible standard, such as the ETSI's DAB standard—the reference data can be generated using the surveillance data. This trick enables the system to use only one antenna for operations, which is advantageous in various ways. Several authors have discussed such systems in further detail, such as in [51], [50], and [66].

The aim of this project was to design the functionality for converting surveillance data into reference data for a given DAB signal. This was done via the demodulation of the received signal, and then the remodulation thereof, thereby producing a 'perfect' copy of the received signal. This perfect copy (used as reference data), together with the original recorded signal (used as surveillance data), could be provided to a standard PR algorithm for further processing and analysis. A block diagram depicting the larger PR system's set-up is provided in Figure 6.1.

One can see in the figure how the `pre-process`, `demodulate`, and `remodulate` blocks fit into the context of a PR system. Note, however, that the `pre-process` block would likely be markedly different in such

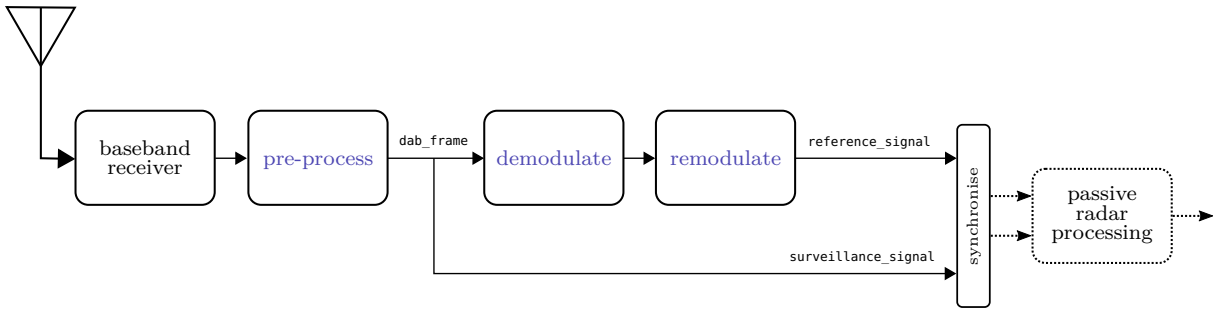


Figure 6.1: Block diagram showing how the `demodulate` and `remodulate` blocks fit into a larger PR system.

a system, compared to the one designed in this project. For example, the data from the baseband receiver would likely be streamed, rather than read from a file, as was done here. The other two blocks may also be slightly different when implemented in a real system, but their fundamental routines would remain the same. As shown, the reference and surveillance data from the DAB processing chain must always be synchronised before being used in the PR processing chain.

For the sake of brevity, details of the `baseband receiver` and `synchronise` blocks are not given here. Moreover, specifics of the `passive radar processing` chain are similarly omitted.

6.3 Illustration

A common output that is plotted for a particular PR situation is the Amplitude-Range-Doppler (ARD) map, which shows the two-dimensional cross-correlation of the reference and surveillance data over a finite integration period. This plot can be used to identify targets from a scene. To illustrate the use of a DAB signal in a PR chain, and what an ARD map might look like in such a situation, a simple simulation was performed.

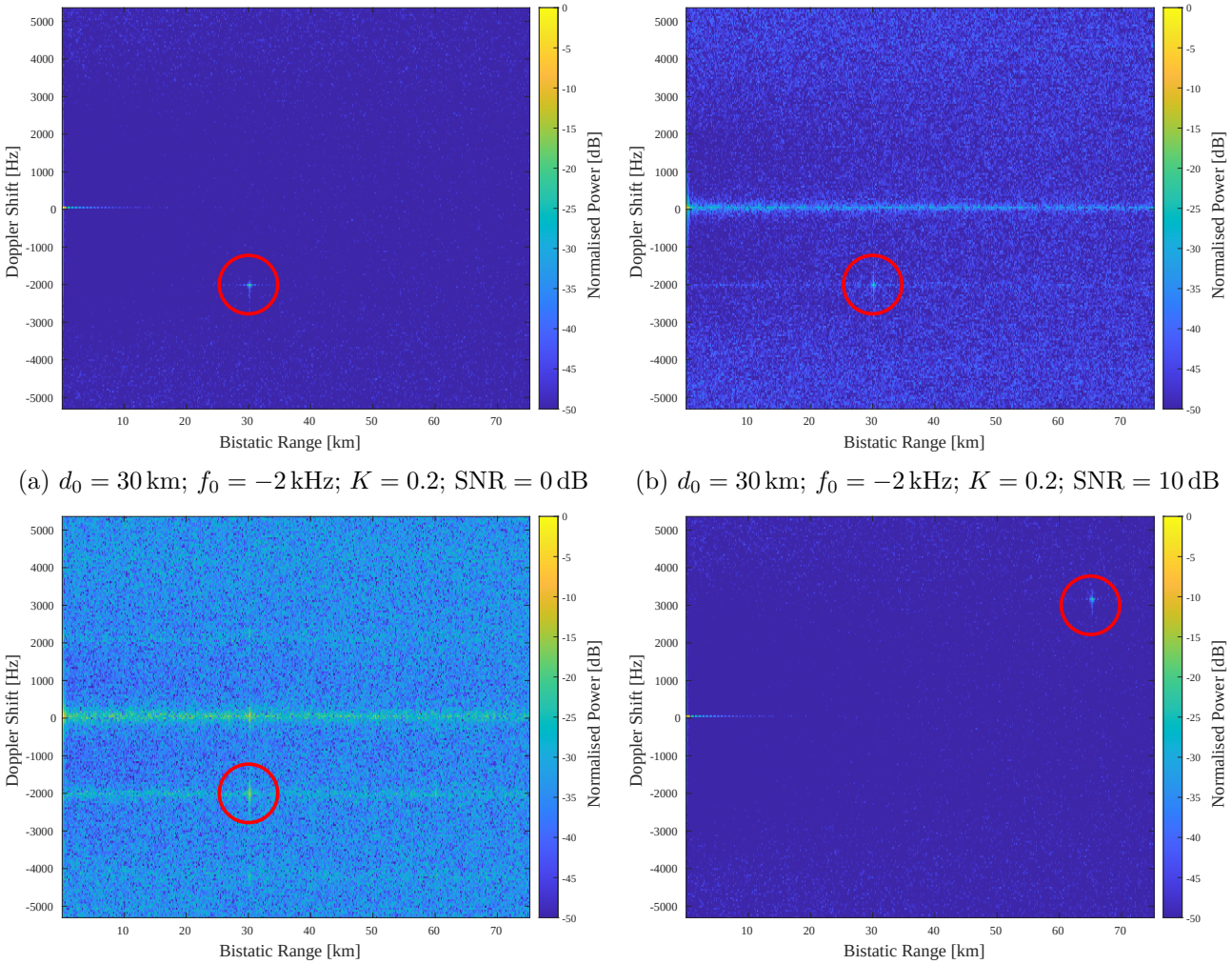
The simulation was based off an arbitrary, perfect DAB signal, $x[n]$. To simulate an object in a scene, a delayed and doppler-shifted version of this signal was created, called $y[n]$, with a bistatic range of d_0 and doppler shift of f_0. A normally-distributed random noise signal was also defined, $\eta[n]$. The surveillance signal was then defined as the sum of these signals, with the latter two scaled by constants K and σ respectively, to control their relative weightings in the simulation:

$$s[n] = x[n] + K \cdot y[n] + \sigma \cdot \eta[n] \tag{6.1}$$

Ideally, by passing the surveillance signal through the demodulation-remodulation processing chain, the original DAB signal, $x[n]$, would have been recovered. However, depending on the echo and noise signals added, some bit-errors may have arisen in the processed signal. Nonetheless, this reconstructed signal was used as the reference signal:

$$r[n] = \text{Remod}\Big\{ \text{Demod}\Big\{ \; s[n] \; \Big\} \Big\} \tag{6.2}$$

These two signals, the surveillance and reference, were passed into a provided PR processing chain, with an ARD map as the output. A variety of situations were simulated, with different values chosen for each parameter. Note that the parameter σ controlled the Signal-to-Noise Ratio (SNR). Four of the simulation results are shown in Figure 6.2, with red circles superimposed to highlight the simulated object's location in each.



(a) $d_0 = 30\,\text{km}$; $f_0 = -2\,\text{kHz}$; $K = 0.2$; SNR $= 0\,\text{dB}$

(b) $d_0 = 30\,\text{km}$; $f_0 = -2\,\text{kHz}$; $K = 0.2$; SNR $= 10\,\text{dB}$

(c) $d_0 = 30\,\text{km}$; $f_0 = -2\,\text{kHz}$; $K = 0.8$; SNR $= 0\,\text{dB}$

(d) $d_0 = 65\,\text{km}$; $f_0 = 3\,\text{kHz}$; $K = 0.2$; SNR $= 0\,\text{dB}$

Figure 6.2: Results from four of the simulations of a DAB signal in a PR processing chain.

This short chapter has simply illustrated how the DAB processing blocks fit into a larger PR chain, and how an output thereof might appear. There is clearly scope for further investigation, both in simulation and in real-world testing.

Chapter 7

Conclusions

> The same rule holds for us now, of course: we choose our next world through what we learn in this one. Learn nothing, and the next world is the same as this one.
>
> —*Richard Bach, Jonathan Livingston Seagull*

The purpose of this project was to design and implement a DAB processing chain for the context of a PR system. The use of digital broadcasting signals—such as DAB—in PR has become an increasingly attractive topic for researchers. PR itself offers a host of advantages over its active radar counterpart, and *digital* broadcasting-based PR promises to mitigate some of the problems encountered with *analogue* broadcasting-based PR. The exciting field continues to grow.

This report began with a general surveillance of the relevant literature in Chapter 2, in order to paint a backdrop of the project for the reader. This perusal was not about finding assistance to design the DAB processing chain itself; instead, it was about understanding the *context* for which the chain was being designed, through the lens of the literature. Accordingly, PR and digital broadcasting were covered as independent topics, followed by a consideration of the use of the former in the latter: digital broadcasting-based PR. It was found that the literature lacked a clear and coherent exposition of the use of DAB signals in PR contexts—focusing too much on either the intricacies of the DAB standard, or the intricacies of the PR implementation. Therefore, the potential value of this project was noted.

The literature review was followed in Chapter 3 by a richer unpacking of the DAB standard itself, beginning with the two main concepts involved in DAB: COFDM and DQPSK. Building upon these foundations, a short description of the DAB transmission frame's structure was then provided. Because of the larger PR context in which this project fits, only the relevant aspects of the frame were covered. This minimal approach led to many useful simplifications. The explanations used in this chapter attempted to be as graphical as possible, omitting the often-obfuscating mathematical equations underlying the DAB signals.

The bulk of the work for this project followed next, in Chapter 4. Here, the DAB processing chain that was designed in MATLAB was documented. Three main blocks of the chain were identified: pre-processing, demodulation, and remodulation. Each of these was constructed from a variety of sub-blocks, which mapped directly to underlying MATLAB functions. The pre-processing block was designed to extract a single DAB frame from a binary file of a DAB recording; then, with that single frame extracted, the demodulation and remodulation blocks were designed to work in tandem to reconstruct the frame perfectly, according to the DAB specifications. Of the three blocks, the

pre-processing block would likely be the most different in a real-world PR system, and this was noted in the chapter. Nonetheless, for all three of the blocks, a thorough account was provided, with detailed block diagrams and sample plots shown. In some places, relevant algorithms were given and discussed; in others, short code-segments were listed. Ultimately, this chapter was intended to act much like a handbook for the MATLAB code that was written for the project.

In Chapter 5, the designed DAB processing chain was then validated. Four different validation approaches were proposed, in increasing degrees of rigour. For each of these, an explanation of the philosophy and methodology behind the approach was outlined, followed by its results. Starting at the sub-block level, and moving to the full-chain considerations, all of these validation tests passed. This included the tests done with known reference data. These results thus provided cogent evidence that the DAB processing chain was indeed designed successfully.

Finally, Chapter 6 attempted to provide a short illustrative example of how the DAB processing chain would fit into a larger PR system. A block diagram for a possible integration approach was shown, followed by a series of quick simulations and associated plots. This segment was included solely to provide the reader with a bigger-picture understanding of the project and its motivations. No performance analysis was provided.

In summary, the project achieved the goals that were set out, by designing and demonstrating a valid DAB processing chain. All of the required aspects of the chain were implemented, and omitted functionality was clearly indicated as such. Moving forwards, this report can indeed act as a guide for the use of DAB signals in PR contexts. Therefore, the project is considered to be a success.

Chapter 8

Recommendations

> It is for us the living, rather, to be dedicated here to the unfinished work which they who fought here have thus far so nobly advanced.
>
> —*Abraham Lincoln*

The nature of this short project has left many interesting and important aspects of the topic unaddressed. It is hoped that the cursory work completed here will act as a trampoline for future investigations. The recommendations for further study are summarised briefly below.

8.1 DAB Processing Chain

- The use of the other three DAB transmission modes should be investigated and tested.
- Frequency synchronisation methods should be considered and implemented into the existing DAB processing blocks.
- The built-in error correction techniques of the DAB signal should be further explored and applied to real-world data.
- The designed MATLAB code should be ported into a language appropriate for real-time implementation, such as C++, and this code should be optimised where possible. Much more work can be done in considering the efficiency of the algorithms and indivudal sub-blocks too.
- The entire processing chain should be adapted to allow for live streaming of DAB data, rather than simply reading from a pre-recorded binary file.

8.2 Real-world Passive Radar Integration

- The DAB processing blocks should be integrated into an actual PR processing system.
- The PR system should then be tested with regards to effiency, accuracy, and similar metrics, with and without the various DAB processing techniques and blocks.

Bibliography

[1] EN ETSI, "300 401 Ver. 1.4.1: Radio Broadcasting Systems," *Digital Audio Broadcasting (DAB) to mobile, portable and fixed receivers*, Jan. 2006.

[2] L. Brown, *Technical and military imperatives: a radar history of World War 2*. Crc Press, 1999.

[3] J. C. Maxwell, "VIII. A dynamical theory of the electromagnetic field," *Philosophical transactions of the Royal Society of London*, no. 155, pp. 459–512, 1865.

[4] D. Sengupta and T. Sarkar, "Maxwell, Hertz, the Maxwellians, and the early history of electromagnetic waves," *IEEE Antennas and Propagation Magazine*, vol. 45, no. 2, pp. 13–19, Apr. 2003.

[5] H. Hertz, "On electromagnetic waves in air and their reflection," in *Electric Waves, Being Researches on the Propagation of Electric Action with Finite Velocity Through Space*. Macmillan, 1893, pp. 519–528.

[6] D. Cichon, "The Heinrich Hertz wireless experiments at Karlsruhe in the view of modern communication," in *International Conference on 100 Years of Radio*. IEE, 1995.

[7] G. Galati and P. van Genderen, "History of radar: The need for further analysis and disclosure," in *2014 11th European Radar Conference*. IEEE, 2014.

[8] S. S. Swords, *Technical History of the Beginnings of Radar*. Institution of Engineering & Technology, 1986. [Online]. Available: https://www.ebook.de/de/product/32341325/s_s_swords_technical_history_of_the_beginnings_of_radar.html

[9] D. Pritchard, *The radar war: Germany's pioneering achievement 1904-45*. Harpercollins, 1989.

[10] H. Griffiths, P. Knott, and W. Koch, "Christian Hülsmeyer: Invention and Demonstration of Radar, 1904," *IEEE Aerospace and Electronic Systems Magazine*, vol. 34, no. 9, pp. 56–60, 2019.

[11] A. O. Bauer, "Christian Hülsmeyer and about the early days of radar inventions," 2005.

[12] H. Kuschel and D. O'Hagan, "Passive radar from history to future," in *11th International Radar Symposium*, 2010, pp. 1–4.

[13] H. Griffiths and N. Long, "Television-based bistatic radar," *IEE Proceedings F (Communications, Radar and Signal Processing)*, vol. 133, no. 7, p. 649, 1986.

[14] M. Malanowski, *Signal Processing for Passive Bistatic Radar*. Artech House Publishers, 2019. [Online]. Available: https://www.ebook.de/de/product/37899587/mateusz_malanowski_signal_processing_for_passive_bistatic_radar.html

[15] D. O'Hagan, "Passive bistatic radar performance characterisation using FM radio illuminators of opportunity," Ph.D. dissertation, University College London, 2009.

[16] S. Paine, F. Schonken, M. Malape, D. W. O'Hagan, J. Swart, F. Louw, and M. Setsubi, "Multi Band FM and DVB-T2 Passive Radar Demonstrator," in *2018 19th International Radar Symposium (IRS)*. IEEE, 2018.

[17] E. Giusti, M. Martorella, A. Capria, M. Conti, C. Moscardini, and F. Berizzi, "Electronic Countermeasure for OFDM-Based Imaging Passive Radars," in *2018 International Conference on Radar (RADAR)*. IEEE, 2018.

[18] C. Schüpbach and U. Böniger, "Jamming of DAB-based Passive Radar systems," in *2016 European Radar Conference (EuRAD)*, 2016, pp. 157–160.

[19] D. W. O'Hagan, S. Paine, and C. Schüpbach, "Overview of Electronic Attacks Against Passive Radar," in *2019 International Applied Computational Electromagnetics Society Symposium (ACES)*, 2019, pp. 1–2.

[20] S. Paine, "Electronic Countermeasures Applied to Passive Radar," Ph.D. dissertation, University of Cape Town, 2019.

[21] J. S. Belrose, "A Fessenden Christmas Eve Broadcast Retrospective [Historical Corner]," *IEEE Antennas and Propagation Magazine*, vol. 54, no. 4, pp. 284–287, 2012.

[22] R. W. Chang, "Synthesis of Band-Limited Orthogonal Signals for Multichannel Data Transmission," *Bell System Technical Journal*, vol. 45, no. 10, pp. 1775–1796, 1966.

[23] S. Weinstein and P. Ebert, "Data Transmission by Frequency-Division Multiplexing Using the Discrete Fourier Transform," *IEEE Transactions on Communication Technology*, vol. 19, no. 5, pp. 628–634, 1971.

[24] G. McNally, "Digital audio in broadcasting," *IEEE ASSP Magazine*, vol. 2, no. 4, pp. 26–44, Oct. 1985.

[25] R. Lassalle and M. Alard, "Principles of modulation and channel coding for digital broadcasting for mobile receivers," *EBU Tech. Rev*, vol. 224, no. 1, pp. 68–190, 1987.

[26] M. Alard, R. Halbert, B. L. Floch, and D. Pommier, "A new system of sound broadcasting to mobile receivers," in *8th European Conference on Electrotechnics, Conference Proceedings on Area Communication*. IEEE, 1988.

[27] C. Weck and G. Theile, "Digital audio broadcasting: optimizing of a combined concept of source and channel coding with respect to subjective criteria," in *1988 International Broadcasting Convention, IBC 1988*, 1988, pp. 360–363.

[28] J. Rault, D. Castelain, and B. L. Floch, "The coded orthogonal frequency division multiplexing (COFDM) technique, and its application to digital radio broadcasting towards mobile receivers," in *IEEE Global Telecommunications Conference, 1989, and Exhibition. Communications Technology for the 1990s and Beyond*. IEEE, 1989.

[29] P. Bernard, "'STERNE': the CCETT proposal for digital television broadcasting," in *1992 IBC International Broadcasting Convention*, 1992, pp. 372–374.

[30] E. Stare, "Development of a prototype system for digital terrestrial HDTV," in *IEE Colloquium on Applications of Video Compression in Broadcasting*, 1992, pp. 12/1–12/6.

[31] P. Shelswell, C. Gandy, J. L. Riley, and M. Maddocks, "Digital audio broadcasting," in *IEE Colloquium on Vehicle Audio Systems*, 1991, pp. 8/1–8/3.

[32] H. Price, "CD by radio: digital audio broadcasting," *IEE Review*, vol. 38, no. 4, p. 131, 1992.

[33] M. C. D. Maddocks, "Digital audio broadcasting experimental results of the coverage obtained using the cofdm system," in *1992 IBC International Broadcasting Convention*, 1992, pp. 260–264.

[34] J. Hallier, T. Lauterbach, and M. Unbehaun, "Multimedia broadcasting to mobile, portable and fixed receivers using the Eureka 147 Digital Audio Broadcasting system," in *5th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, Wireless Networks - Catching the Mobile Future.* IOS Press, 1994.

[35] B. O'Neill, "DAB Eureka-147: a European vision for digital radio," *New Media & Society*, vol. 11, no. 1-2, pp. 261–278, Feb. 2009.

[36] Y. Wu, S. Hirakawa, U. Reimers, and J. Whitaker, "Overview of Digital Television Development Worldwide," *Proceedings of the IEEE*, vol. 94, no. 1, pp. 8–21, 2006.

[37] C. Gandy, "DAB: an introduction to the Eureka DAB System and a guide to how it works," *BBC R&D White Paper*, 2003.

[38] D. Witherow and P. Laven, "Digital audio broadcasting - the future of radio," in *International Broadcasting Conference IBC 95.* IEE, 1995.

[39] I. Matsepe-Casaburri, "Broadcasting Digital Migration Policy for South Africa," Government Gazette No. 21408, Aug. 2008. [Online]. Available: https://www.gov.za/sites/default/files/gcis_document/201409/31408958.pdf

[40] H. D. Griffiths, A. J. Garnett, C. J. Baker, and S. Keaveney, "Bistatic radar using satellite-borne illuminators of opportunity," in *92 International Conference on Radar*, 1992, pp. 276–279.

[41] D. O'Hagan, F. Colone, C. Baker, and H. Griffiths, "Passive bistatic radar (PBR) demonstrator," in *IET International Conference on Radar Systems 2007.* IEE, 2007.

[42] J. W. Brown, "FM airborne passive radar," Ph.D. dissertation, University College London, 2013.

[43] C. A. Tong, "A scalable real-time processing chain for radar exploiting illuminators of opportunity," Ph.D. dissertation, University of Cape Town, 2014.

[44] C. Tong and J. Coetser, "A minimal architecture for real-time, medium range aircraft detection using FM-band illuminators of opportunity," in *2015 IEEE Radar Conference (RadarCon).* IEEE, 2015.

[45] D. Moser, G. Tresoldi, C. Schupbach, and V. Lenders, "Design and evaluation of a low-cost passive radar receiver based on IoT hardware," in *2019 IEEE Radar Conference (RadarConf)*. IEEE, 2019.

[46] D. W. O'Hagan, H. Kuschel, J. Heckenbach, M. Ummenhofer, and J. Schell, "Signal reconstruction as an effective means of detecting targets in a DAB-based PBR," in *11th International Radar Symposium*, 2010, pp. 1–4.

[47] H. Griffiths and C. Baker, "Measurement and analysis of ambiguity functions of passive radar transmissions," in *IEEE International Radar Conference*. IEEE, 2005.

[48] S. Searle, J. Palmer, L. Davis, D. W. O'Hagan, and M. Ummenhofer, "Evaluation of the ambiguity function for passive radar with OFDM transmissions," in *2014 IEEE Radar Conference*. IEEE, May 2014.

[49] C. R. Berger, B. Demissie, J. Heckenbach, P. Willett, and S. Zhou, "Signal Processing for Passive Radar Using OFDM Waveforms," *IEEE Journal of Selected Topics in Signal Processing*, vol. 4, no. 1, pp. 226–238, Feb. 2010.

[50] S. Searle, L. Davis, and J. Palmer, "Signal processing considerations for passive radar with a single receiver," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, Apr. 2015.

[51] W. C. Barott and J. Engle, "Single-antenna ATSC passive radar observations with remodulation and keystone formatting," in *2014 IEEE Radar Conference*. IEEE, May 2014.

[52] D. Poullin, "Passive detection using digital broadcasters (DAB, DVB) with COFDM modulation," *IEE Proceedings - Radar, Sonar and Navigation*, vol. 152, no. 3, p. 143, 2005.

[53] M. Radmard, M. Bastani, F. Behnia, and M. Nayebi, "Feasibility analysis of utilizing the '8k mode' DVB-T signal in passive radar applications," *Scientia Iranica*, vol. 19, no. 6, pp. 1763–1770, 2012.

[54] J. Palmer, S. Palumbo, A. Summers, D. Merrett, S. Searle, and S. Howard, "An overview of an illuminator of opportunity passive radar research project and its signal processing research directions," *Digital Signal Processing*, vol. 21, no. 5, pp. 593–599, 2011.

[55] T. Peto, L. Dudas, R. Seller, and P. Renner, "Digital television broadcast -based passive radar research and development," in *2014 20th International Conference on Microwaves, Radar and Wireless Communications (MIKON)*. IEEE, 2014.

[56] D. W. O'Hagan, M. Setsubi, and S. Paine, "Signal reconstruction of DVB-T2 signals in passive radar," in *2018 IEEE Radar Conference (RadarConf18)*. IEEE, Apr. 2018.

[57] I. Low, X. Chia, and S. Y. Huang, "A Low-Cost DVB-T2-based Passive Bistatic Phased Array Radar for Target Detection and Localisation," in *2019 IEEE Asia-Pacific Microwave Conference (APMC)*. IEEE, 2019.

[58] H. J. Yardley, "Bistatic Radar Based on DAB Illuminators: The Evolution of a Practical System," in *2007 IEEE Radar Conference*. IEEE, Apr. 2007.

[59] C. Coleman and H. Yardley, "DAB based passive radar: Performance calculations and trials," in *2008 International Conference on Radar*. IEEE, Sep. 2008.

[60] C. Schupbach, C. Patry, F. Maasdorp, U. Boniger, and P. Wellig, "Micro-UAV detection using DAB-based passive radar," in *2017 IEEE Radar Conference (RadarConf)*. IEEE, May 2017.

[61] M. Daun and W. Koch, "Multistatic Target Tracking for Non-Cooperative Illuminating by DAB/DVB-T," in *OCEANS 2007 - Europe*. IEEE, Jun. 2007.

[62] M. Edrich, F. Meyer, and A. Schroeder, "Design and performance evaluation of a mature FM/DAB/DVB-T multi-illuminator passive radar system," *IET Radar, Sonar & Navigation*, vol. 8, no. 2, pp. 114–122, 2014.

[63] P. Shelswell, "The COFDM modulation system: the heart of digital audio broadcasting," *Electronics & Communication Engineering Journal*, vol. 7, no. 3, pp. 127–136, Jun. 1995.

[64] J. Bingham, "Multicarrier modulation for data transmission: an idea whose time has come," *IEEE Communications Magazine*, vol. 28, no. 5, pp. 5–14, May 1990.

[65] A. Grami, "Synchronization," in *Introduction to Digital Communications*. Elsevier, 2016, pp. 357–375.

[66] G. Fang, J. Yi, X. Wan, Y. Liu, and H. Ke, "Experimental Research of Multistatic Passive Radar With a Single Antenna for Drone Detection," *IEEE Access*, vol. 6, pp. 33 542–33 551, 2018.

Appendix A

Repository Locations

All MATLAB code designed for this project can be found at the following link:

https://github.com/callumtilbury/EEE4022S-Code

Furthermore, the LaTeX source code for this report, as well as the SVG and PDF diagrams that were designed and included, can be found at:

https://github.com/callumtilbury/EEE4022S-Report

Appendix B

Proof of OFDM Carrier Orthogonality

Statement: Given two complex exponential signals with frequencies that differ by $\Delta\omega = \frac{2\pi k}{T_u}$, the signals are orthogonal over an integration period of T_u for any integer value of k.

Proof: Consider the two complex exponential signals that differ by a frequency of $\Delta\omega$,

$$\psi_1 = \rho_1 \cdot e^{j(\omega_0 t + \theta_1)} \tag{B.1}$$

$$\psi_2 = \rho_2 \cdot e^{j((\omega_0 + \Delta\omega)t + \theta_2)} \tag{B.2}$$

where ρ_1 and ρ_2 are arbitrary real values, and θ_1 and θ_2 are arbitrary phase-shifts.

The inner product of the two signals over an integration period of T_u is defined as,

$$\langle \psi_1, \psi_2 \rangle := \int_0^{T_u} \psi_1(t)\psi_2^*(t)\, dt \tag{B.3}$$

where $\psi_2^*(t)$ is the complex conjugate of $\psi_2(t)$. Substituting ψ_1 and ψ_2, and then simplifying,

$$\langle \psi_1, \psi_2 \rangle = \int_0^{T_u} \rho_1 e^{j(\omega_0 t + \theta_1)} \cdot \rho_2 e^{-j((\omega_0 + \Delta\omega)t + \theta_2)}\, dt \tag{B.4}$$

$$= \rho_1\rho_2 \int_0^{T_u} e^{j(\omega_0 t + \theta_1 - \omega_0 t - \Delta\omega t - \theta_2)}\, dt \tag{B.5}$$

$$= \rho_1\rho_2 e^{j(\theta_1 - \theta_2)} \int_0^{T_u} e^{-j\Delta\omega t}\, dt \tag{B.6}$$

Letting $C = \rho_1\rho_2 e^{j(\theta_1 - \theta_2)}$, substituting $\Delta\omega = \frac{2\pi k}{T_u}$, and calculating the definite integral,

$$\langle \psi_1, \psi_2 \rangle = C \cdot \frac{1}{-j2\pi\frac{k}{T_u}} \left[e^{-j2\pi\frac{k}{T_u}t} \right]_{t=0}^{t=T_u} \tag{B.7}$$

$$= C \cdot \frac{1}{-j2\pi\frac{k}{T_u}} (e^{-j2\pi k} - 1) \tag{B.8}$$

Since $e^{-j2\pi k} = 1 \,\forall\, k \in \mathbb{Z}$,

$$\langle \psi_1, \psi_2 \rangle = 0 \tag{B.9}$$

Therefore, ψ_1 and ψ_2 are orthogonal over the integration period T_u.

Appendix C

DAB Parameter Values

The DAB standard—as specified by ETSI in [1]—defines four transmission modes, with each mode designed for a particular use-case. Table C.1 shows the various parameter values for each of the modes. Note that all of the period values are defined as multiples of the sampling period, with $F_s = 2.048\,\text{MHz}$, and therefore $T = \frac{1}{F_s} = \frac{1}{2.048 \times 10^6}$.

	DAB Transmission Mode			
Parameter	**I**	**II**	**III**	**IV**
L	76	76	153	76
K	1536	384	192	768
T_f	196 608 T	49 152 T	49 152 T	98 304 T
T_{null}	2 656 T	664 T	345 T	1 328 T
T_s	2 552 T	638 T	319 T	1 276 T
T_u	2 048 T	512 T	256 T	1 024 T
Δ	504 T	126 T	63 T	252 T

Table C.1: Parameters for the four DAB transmission modes, as defined by ETSI in [1]

with:

- L → Number of DAB symbols in each DAB transmission frame
- K → Number of sub-carriers in an OFDM symbol
- T_f → Transmission frame period
- T_{null} → NS period
- T_s → DAB symbol period (i.e. including guard interval)
- T_u → Useful symbol period (i.e. excluding guard interval)
- Δ → Guard interval period

Appendix D

Ethics Form

Not included in this copy of the report.