DAB TRANSMISSION SYSTEM SIMULATION

Master thesis performed in Electronic Systems
by

Héctor Uhalte Bilbao

LiTH-ISY-EX-3629-2004
2004-08-24

DAB TRANSMISSION SYSTEM SIMULATION

Master thesis in Electronic Systems for Telecommunication Engineering
at Linköping Institute of Technology
by

Héctor Uhalte Bilbao

LiTH-ISY-EX-3629-2004

Supervisor: Kent Palmkvist

Examiner: Kent Palmkvist
Linköping 2004-08-24

Avdelning, institution
Division, Department

Institutionen för systemteknik
581 83 Linköping

Datum
Date
2004-08-24

Språk
Language

☐ Svenska/Swedish
X Engelska/English
☐ _____

Rapporttyp
Report category

☐ Licentiatavhandling
X Examensarbete
☐ C-uppsats
☐ D-uppsats
☐ Övrig rapport
☐

ISBN
_____
–

ISRN LiTH-ISY-EX-3629-2004

Serietitel och serienummer
Title of series, numbering

ISSN
_____

URL för elektronisk version

Titel **Simulering av ett dataöverföringssystem baserat på DAB standarden**

Title
DAB Transmission System Simulation

Författare: Héctor Uhalte Bilbao
Author

Sammanfattning
Abstract

DAB (Digital Audio Broadcasting) is the radio digital system developed as an european standard by the ETSI, EN 300 400, based on the Eureka-147 group works, to improve the performance of the analogue radio systems (AM and FM). The system is based on the OFDM technology which allows DAB to exploit the spectrum frequencies in a better way with a higher quality of sound for mobile receivers specially. The main part of the OFDM system is based on the FFT algorithms to spread the data flow over different orthogonal carriers. The simulation has been developed in Simulink™ and Matlab™ and the layout designed follows faithfully the standard for the transmission system. The simulation can be reloaded by the user with the information presented in this thesis. Thus, this work can be continued to complete the DAB whole system simulation. The results obtained running this simulation show the main DAB system characteristics.

Nyckelord:

Keywords:

DAB, Terrestrial digital broadcasting, OFDM, COFDM, IFFT, FFT, SFN, Simulink™, Differential modulator.

FOREWORD

This project has been developed with the support of the Electronic Systems department and the department of Electrical Engineering (ISY) staff.

I would like to thank my family and friends in Spain for their constant support all through these years. My sincere thanks also goes to my partners and teachers at the University of Cantabria, I would not have made it without their help. I would also like thank my supervisor Kent Palmkvist, the international coordinator Olle Seger and the international erasmus network group, for helping me so much here in Sweden.

Finally I would like to dedicate this thesis to the people who has shared with me this wonderful time here in Linköping, I will never forget you.

ABSTRACT

DAB (Digital Audio Broadcasting) is the radio digital system developed as an european standard by the ETSI, EN 300 400, based on the Eureka-147 group works, to improve the performance of the analogue radio systems (AM and FM). The system is based on the OFDM technology which allows DAB to exploit the spectrum frequencies in a better way with a higher quality of sound for mobile receivers specially. The main part of the OFDM system is based on the FFT algorithms to spread the data flow over different orthogonal carriers. The simulation has been developed in Simulink™ and Matlab™ and the layout designed follows faithfully the standard for the transmission system. The simulation can be reloaded by the user with the information presented in this thesis. Thus, this work can be continued to complete the DAB whole system simulation. The results obtained running this simulation show the main DAB system characteristics.

CONTENTS

1. INTRODUCTION TO DAB

1.1 What is DAB?

DAB (Digital Audio Broadcasting) is a whole digital radio broadcasting system that not only is intended for audio services as usual, but also for low data rate, graphs, HTML pages, ancillary data, just to name a few. It is available mainly in a terrestrial propagation system but it is also thought to be supported by satellite coverage and cable networks.

The DAB standard comes up from the necessity to exploit the benefits of the digital technology in terrestrial broadcasting in order to improve the characteristics and the services provided by the analogue radio networks. The radio is one of the most widespread communication media, but in the analogue versions, both AM (Amplitude Modulation) and FM (Frequency Modulation), do not provide high quality for mobile receivers. The analogue systems were thought and designed for static users, but the average radio listener has changed since the arising of the FM technology in the 1950's and even more since the early 1920's with the appearance of first AM broadcasting networks. With the development of new smaller and cheaper electronic devices, new portable and car radio receiver were also possible. Analogue networks are able, of course, to provide radio services for the most of the mobile and portable users, but they do not offer any kind of protection against the multi path interference which is impossible to be avoided when we are talking about mobile communication systems or even the AM signal often suffers fading and interference problems. With just a simple non-directional whip antenna DAB eliminates interference and the problem of multi path, wide geographical areas are covered with no signal interruption, and the benefits of the SFN (Single Frequency Networks), which forward will be explained, allows the transmission to be developed with low power transmission, which it becomes in a cheaper and easier transmitter networks.

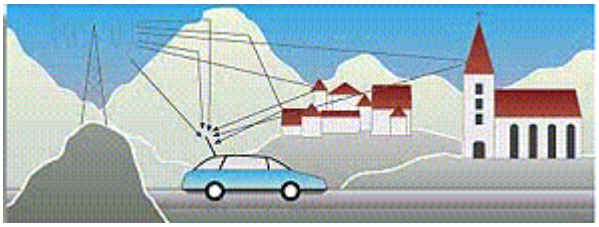

Figure 1.1 Multipath transmission example.

The standard [5] has been developed within an european project called Eureka 147. DAB is a very strong system designed for both home and portable receivers, but specially for mobile. It is also thought to be use with satellite and cable technologies. To avoid the multipath interference all the incoming signals in the reception side should be added together, which means that all of them would be contributing to increase the signal level in a positive way.

It is already well known that the digital technology allows a better exploitation of the frequency spectrum; as such many more channels are available using the same amount of spectrum band frequencies, so many more programs and networks are also available. Different quality of service can be provided by the broadcasting networks, using the digital facilities in the same way as are used in the 3rd mobile standard generation for example. Thus several low-quality can be multiplexed together or instead the multiplex can be made up with a few high quality programs.

The digital technology brings data service facilities, therefore the transmission characteristics that are useful information for a better reception quality, can be delivered together with program associated data, such as the the titles of the songs, the names of the singers or the lyrics in several languages for example.

Therefore the main benefits of DAB in comparison with analogue radio, are a better sound quality, value added services (like the ones mentioned above), more flexibility inside the DAB multiplex and a more efficient transmission specially with the idea of SFN's.

1.2 Brief history of DAB

The first attempt to develop a digital sound broadcasting system was in the early 1980's. Satellite technology was used to it with very low data compression and, of course, it was not intended for mobile reception and no local services could be provided by the satellite networks. Thus the research for a terrestrial digital solution was essential.

To develop this new digital solution an international research project was necessary. So, in 1986 some organizations from France, Germany, United Kingdom and The Netherlands signed an agreement to cooperate in the development of a new standard. And then in 1987 the Eureka-147 project was born. Since the beginning most of the european official groups, like the EBU (European Broadcasting Union) which had already supported the satellite services for mobiles before, joined the project. Later relations with ITU-R (International Telecommunications Union – Radio) and ETSI (European Telecommunications Standard Institute) were formed in order to begin the standardization process.

The goals were set up from the beginning in the next terms:

- Digital technology should be used to obtain higher sound quality than the one from FM and AM analogue systems and that quality should be similar to the CD sound quality. The coverage area for the new service should be, at least, as high as the one with FM and AM systems.

- The mobile reception should be also available with the same high quality as fixed-receivers even at high speeds.
- Ancillary data transmission should be available.
- An efficient frequency spectrum use, and low transmitting power should be both required.
- Broadcasting services should be delivered by terrestrial, cable or satellite options.
- Smooth transition to digital technology from FM and AM would be desired in order to allow the final users and receiver designers, to begin to be in touch with the new system in a soft and gradual way.
- The project should become a standard in an european level or better in a world-wide standardization.

Different transmission method were proposed to achieve the previous goals:

- One narrow-band system.
- One single carrier spread-spectrum system.
- One multicarrier OFDM (Orthogonal Frequency Division Multiplex) system.
- One frequency-hopping system.

There were tests for all of them that they showed that the latest three, with broad band systems, worked better than the narrow one option. So it was rejected first. The frequency hopping choice seemed to be very demanding considering the usual radio broadcasting networks. To select between the spread spectrum option and the OFDM one, it was taken into account that by that time the latter was already available with the COFDM (Coded Orthogonal Frequency Division Multiplex), so the final decision was to use the OFDM technology. In forward chapters a detailed explanation of COFDM will be exposed.

The selection of the appropriate method to establish a solution for audio coding was the other main decision in the Eureka-147 project. By that time the MPEG (Moving Pictures Expert Group) had already been worked on standardization for data compression, both video and audio coding. The solutions proposed by the Eureka-147 were sent to the MPEG Audio group to be evaluated with other several options from other countries. The performance offered by the methods submitted by the Eureka consortium were clearly superior so they were standardized by the MPEG as MPEG Audio Layers I, II and III. It took a long time until the final decision to which standard should be used for DAB was taken. Layer II was finally chosen, it is also known as MUSICAM.

The DAB bandwidth was established in 1.5 MHz. In the beginning two different options were examined. From a network and service planning point of view and thinking about the frequency spectrum as an ensemble of 16 programs and one TV-channel bandwidth (7 MHz.) transmitter were thought. The test results offered very good performances in multipath environment, but there was no flexibility in

working with such big bandwidth transmission in order to obtain local services for example, so a low bandwidth was required. Experiments in Canada showed that the sound quality was not in the required level if COFDM technology worked with lower than 1.3 MHz bandwidth transmission. Thus the solution for the bandwidth was 1.5 MHz, which means that four DAB blocks, (each one with 1.5 MHz bandwidth) can make up a TV channel with each of them carrying between five up to seven programs. So now local services could be also planned.

DAB was thought to work in frequency range between 30 MHz and 3GHz. Taking into account the changes of the propagation environment characteristics with the frequency, four different transmission modes are defined. They support different distances between transmitters in order to control the delays of all the incoming signals in the receiver to secure that they will be added with no so large phase difference between them and also, in order not to have ISI (InterSymbol Interference). The doppler shift effect degradation, which depends on both the receiver speed and the transmitter frequency, can be also bounded by selecting one of the transmission modes. The selection of the appropriate transmission mode is made automatically by the receiver so the final user does not need to change the receiver settings by himself.

- Transmission Mode I was intended for SFN (Single Frequency Networks) which forward will also be explained, for frequencies below 300 MHz.
- Transmission Mode II was designed for local services and frequencies below 1.5 GHz.
- Transmission Mode III is available for satellite transmission for frequencies below 3 GHz. However was also thought to deliver terrestrial broadcasting in zones where it could be available.
- Transmission Mode IV was intended for large coverage areas with frequencies placed in the L-Band. It is an intermediate mode between Mode I and Mode II.

So, with the main technology characteristics of the system decided, the whole DAB system tests were ready to begin, in 1988 the official presentation of DAB was showed in World Administrative Radio Conference in Geneva. Since then, and in the next successive six years, other demonstrations took place around the world, specially in Canada and Europe. Finally the first DAB standard was achieved in 1993, with the very successfully audio coding standards MPEG/Audio Layers I, II and III. In that year the first ETSI standard for the whole system became a reality and in 1995 the ETSI adopted DAB as the only european standard for digital radio and the last goal were reached finally by the Eureka-147 consortium.

Nowadays the DAB service is becoming a world-wide standard, and only in the USA and Japan is not in consideration. Japan has decided to develop its own national solution so they have opted for the ISDB-T (Terrestrial Integrated Services Digital Broadcasting) solution which allows the transmission both in narrow and wide bandwidth. This technology seems to be available in 2005-2007. In the USA,

the National Association of Broadcasters refuses to adopt the Eureka-147 standard even if the laboratory simulations are clearly satisfactory. This opposition is based on the lack of new frequencies in the spectrum, so a non so demanding bandwidth solution is being searched.

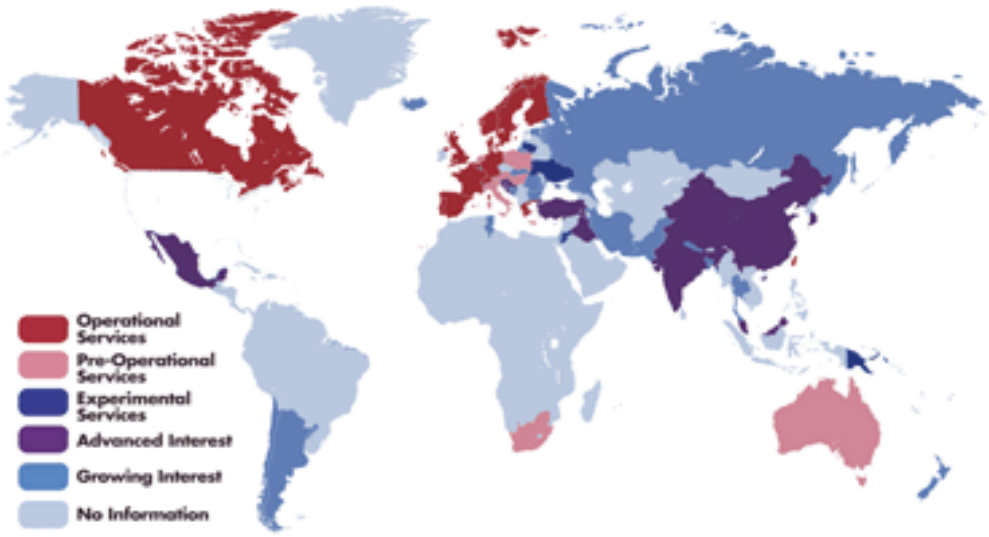

Figure 1.2. DAB World Coverage Map (January 2003)

2. HOW DOES DAB WORK?

2.1 Introduction to FDM

The benefits of the DAB have already been presented in the previous chapter. But, how does the “magic” DAB really work? The digital communications techniques have got the answer. The main DAB subsystem, at least in the transmission and reception sides, is the COFDM.

Digital communications techniques are not new at all. They are very well known since the telegraph was started. One of the most simple and most important use and benefits of the digital techniques, is that no matter how much the incoming signal has been degraded by the communication channel, if the receiver is able to rebuild the signal, that is if the receiver is able to decide if the incoming signal (bit) is “one”, or instead is “zero” with a bounded error rate, then the receiver is able to send again exactly the same signal along the path. This is not possible with analogue techniques, in which the receiver has to manage the signal with amplifiers that add more noise to the signal, in order to recover it, but there is no way to get again the signal which had been sent from the transmitter. The other main and simple advantage is that, “one” and “zero” can be translated by the proper transductor devices in any kind of “analogue” information, such as images, sound, text, etc. Thus several kinds of information can be treated with the same techniques and so the digital systems work with numbers and the analogue systems work with signals.

The use of FDM, was already used for the telegraph system. It was carried over a wide bandwidth channel using a separate carriers for the transmitting path and for the reception one. That means, in the reception side, if the receiver detected the presence of the reception path frequency carrier it decided that a “one” had been sent, otherwise the decision adopted was “zero”. So the system needed two different frequency carries to work. These carriers were selected to be very far away in the spectrum from each other, in order to be able to make realizable filters to single the appropriate channel out.

However, not only different carriers for different information channels can be used, the different frequencies can carry bits of a single message, if we are able to transmit the information in a parallel form, this idea can be realizable. So anyone of the “n” frequencies, would be part of a higher binary rate system. The resulting binary rate would not be the sum of the individual binary rate frequencies, because an unused interval guard, that means, the separation between the carriers, must be respected, in order to avoid the ISI, just like in the telegraph example.

2.2 Orthogonality

Maybe there is a fixed frequency separation between the carriers to improve the performance of the FDM systems. If we are able to position one carrier just in one of the zero energy frequency point for all of the rest carries they could be even overlapped and the system would be even free of intercarrier interference and ISI, because the receiver we will keep on being able to separate the desired carrier from the rest to check it out.

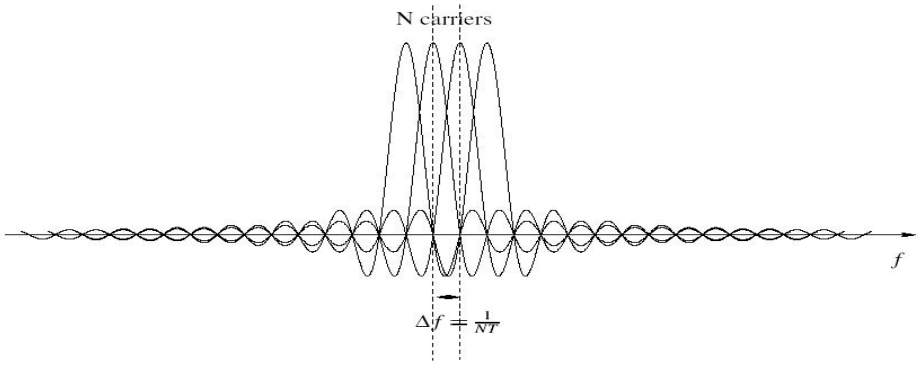

Figure 2.1. OFDM carriers.

That is exactly what Orthogonality means. A very easy example to understand orthogonality is the relation between the sine and cosine math functions(used for quadrature modulators). They are orthogonal between them and so when the sine absolute value is maximum the cosine absolute value is minimum, in fact zero. The sin(x)/x function has got also this property with a shifted version of itself and it is used in the OFDM transmission. In the Figure 2.1 a four-carrier model is shown. To ensure the orthogonality condition, the frequency spacing of the carriers is fixed to be the inverse of the active symbol periods of the, now, OFDM carriers. So during the active symbol period every single carrier is orthogonal to any other one.

So the receiver, in the frequency domain, could handle just the desired frequency by multiplying the incoming whole signal with that only frequency, the non-desired frequencies would result to zero. In the time domain this multiplying operation is translated by the Inverse Fourier Transform into a convolution operation between the desired tone and the signal. This operation is called cross correlation (R_{xy}) and is the result of calculate the product of the both functions integrated over the active symbol period. That is, let x(t) be the selected frequency to be evaluated, and y(t) the OFDM signal, if the frequency carriers amplitude is normalized to one, then if the cross correlation results one, the decision should be “one”, otherwise it should be “zero”.

$$R_{xy} = \int_0^T x(t)y(t)dt$$

2.3 Use of the FFT (Fast Fourier Transformation)

As seen in the previous chapter, where the orthogonality concept has been exposed, one of its immediate applications was the facility to split the bits of the signal into two orthogonal components, called the Phase (I) and Quadrature components(Q). So now the bits can be handled like a complex number where the real part would be the phase component and the imaginary part would be the quadrature component. So in fact, now the whole signal could be transmitted in a parallel way with the two shifted versions of the same carrier(i.e., sin and cosine). If the amplitude of the carriers is divided into several levels the digital QAM (Quadrature Amplitude Modulation) constellations are introduced.

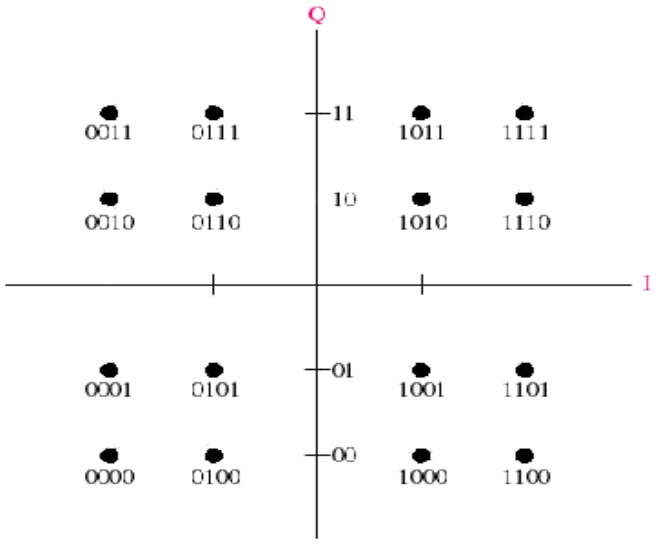

Figure 2.2. 16 QAM constellation map.

Using OFDM techniques, the spectrum of the signal on line is the same to N, where N would be the number of the carriers in OFDM, separate QAM signals. Each QAM signal carries one of the original input complex numbers. If we make N=4 the frequency spectrum of the OFDM signal would be exactly the spectrum shown in the figure 2.1.

So we can consider the whole complex data flow divided in N blocks each one modulating one of the N carriers. A hardware solution to create the carriers could be using banks of oscillators in parallel but it is not a realizable solution for real devices. However there is a solution. The signal, as it has been mentioned can be considered in N different frequency carriers blocks, so it seems to be reasonable if the signal is considered to belong to the frequency domain in the next term:

$$s(t) = \sum_{n=-N/2}^{N/2} Z_n \bullet e^{j2\pi n(t/T)}$$

Where z_n are the complex symbols, so the signal can be expressed as it was said as the sum of all the carriers being multiplied, and this is the formal definition for finite Fourier Series. Thus is the IDFT (Inverse Discrete Fourier Transform) that translates the signal from the frequency domain to the time domain and it is defined on an interval, which is the Fourier period, of length T. The DC component is not used so when n=0, z_n=0. Now the signal should be converted to an RF (Radio Frequency) signal by a simple quadrature modulator.

The IDFT can be easily implemented by software, so this is the optimum solution to make the OFDM transmitter and the receiver systems.

At the receiver side, Fourier analysis will produce the complex symbols again using, this time, the DFT:

$$Z_n = \frac{1}{T} \cdot \int_0^T e^{-j2\pi n(t/T)} \cdot s(t)dt$$

A N-to-N point transformation requires N^2 multiplications and additions. In order to work with real time systems, it would be useful to handle the, now complex signals as quick as possible. The method to work faster with the DFT's is the FFT algorithm, first time developed by J. W. Cooley and J. W. Tukey in 1965 [1], which is the origin of the main part of the DAB transmission system. If the outputs of the transformations are calculated at time, taking into account the cyclic properties of the multipliers complex carriers, FFT reduces the number of computations to the order of N*log(N). The FFT is most efficient when N is a power of two, so the number of the frequency carriers should be always a power of two. The FFT functions can be performed by a general purpose DSP (Digital Signal Processor) chips.

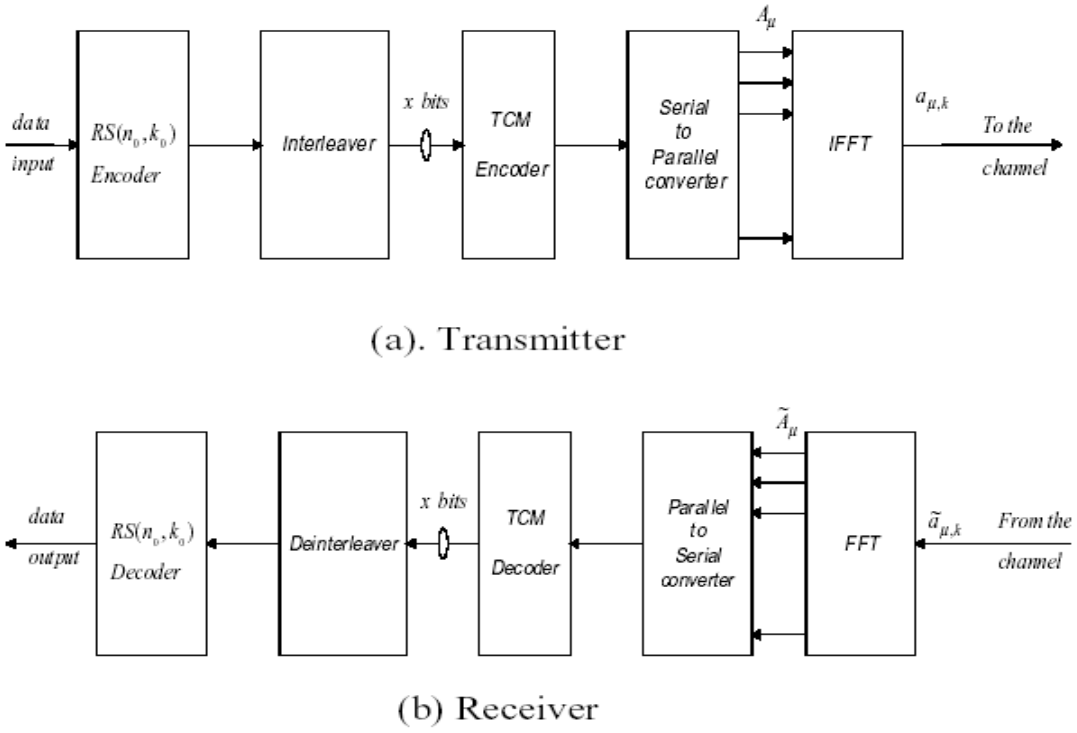

Figure 2.3. FFT Transmission OFDM chain.

2.4 Coded OFDM

The term “coded”, refers to the techniques that are used in order to combat the selective frequency fading. The idea is to distribute the signal over all the carriers and so to spread the information symbols. As a result if a selective carrier fades away it will cause some error bits in several block symbols and not many error bits in only one symbol. So the channel codification will be able to correct the wrong data by using the correct information that is present in the rest of the symbol thanks to all of the rest frequency carriers which were not fading. COFDM uses a Viterbi convolutional encoding as a channel codification and in the transmission block, the frequency interleaver is present to spread the data.

2.5 Guard Interval, Cyclic prefix and SFN's

In the 2.3 point, it was said that the FFT's techniques has got optimum features when the number of carriers is set to be a power of two, but in order to make the system strong enough against the ISI and cross talk interference, there is the necessity to have a guard interval between the OFDM symbols, as it was exposed in section 2.1. The solution to create the guard interval is to decrease the data rate, that is to use less frequency carriers to support the data, so some of the

carriers will be now part of the guard interval but they will not be carrying some additional information, so the system will allow them to stand ISI.

The guard interval is formed by a cyclic continuation of the signal so the information in the guard interval is actually present in the OFDM symbol. The next figure shows clearly the meaning of guard interval.

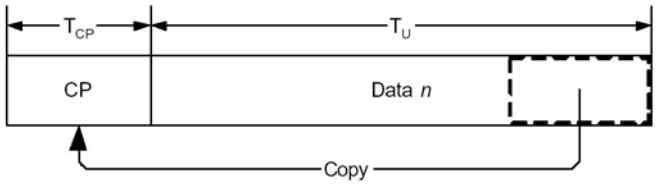

Figure 2.4 Guard Interval and cyclic prefix scheme.

Where T_u is the OFDM symbol time without guard interval and T_{cp} is the duration of the copied information in the guard interval using cyclic prefix. So looking at this figure is going to be easy to understand the significance of the guard interval concept for DAB. If two OFDM symbols interfere one with each other, it is just because the guard interval and part of the data field of one of them has been overlapped by the other one. This can happen when the transmission delay of the symbols is bigger than the guard interval duration. So to avoid the ISI the system has to guarantee that no symbol echoes longer than the duration of the guard interval are going to be present in the system. The way to secure that without increasing the cyclic prefix is by establishing a maximum delay path for the signal. That is that the signal can not travel more time than the interval guard duration, so the transmitter can not be too much far away from the receiver. In the table 3.1 typical values of the DAB system parameters for the four transmission modes are shown. With this idea the SFN's were born. If the transmitters are well synchronized they could support the services with the same frequency without interference between the signals. In fact the receiver would not distinguish if the incoming signals are from reflected rays from only one transmitter or from many transmitters at time.

Transmission Mode	***Number of carriers***	***Sub carrier spacing***	***FFT length***	***Symbol duration***	***Guard interval duration***	***Max RF frequency***
TM I	1536	1 KHz	2048	≈1246 μs	≈246 μs	≈375 MHz
TM II	384	4 KHz	512	≈312 μs	≈62 μs	≈1.5 GHz
TM III	192	8 KHz	256	≈156 μs	≈31 μs	≈3 GHz
TM IV	768	2 KHz	1024	≈623 μs	≈123 μs	≈750 MHz

Table 2.1 DAB system parameters.

There are many advantages for using SFN's with the first one being the power economy. The receiver can handle all signals received in a constructive way,

so if there is no ISI all of them will be added. Moreover the SFN's are shown to have got a diversity effect, that means that maybe one of the carriers from one of the transmitters is fading but maybe the same carriers is being transmitted as usual by another one so this symbol would be received with no errors. The presence of shadows in several signals is much lower than the probability for shadowing in just one signal, this effect is called the network gain, and it will be shown in the simulation results. Thus, the DAB networks are very economical, because lower power transmitters are allowed in comparison to the FM with the same coverage area. Power savings can reach 10 dB. [2]. The frequency economy is quickly perceived and the fact that large areas are allowed to be covered by just one radio frequency becomes in a great planning possibilities for complex broadcast landscapes. The future improvement of the system, then will not need frequency allocations, if some additional areas need coverage, just new transmitters must be added.

The main SFN's disadvantage is that extremely precision in the synchronization of the network must be achieved in order to avoid long delays between the symbols, but the technology to do that is very experimented and the data rate in the DAB transportation network does not need to be very high. Another disadvantage is that no local services can be provided if the SFN is intended to have a national or even international coverage area. The solution used in these cases is disconnecting some of the transmitters from the main network to create with them the desired local SFN with another frequency carrier.

3. THE SIMULATION MODEL

3.1 About the simulation

This chapter describes the process for modeling the DAB transmission system, and one of the possible receivers, as is depicted in the ETSI standard [5], which has been the main source to develop this work. The simulation has been intended to be as faithful as possible to the standard layout and its parameters. Not too many programmes have been designed in order to emphasize the work over the layout, so the standard understanding is made in a easy way because the blocks represented in the standard diagrams are depicted as well in the simulation model. In fact the devices parameters represented, develop exactly the DAB transmission signal along the model simulation in all of the four transmission modes, which can be selected by the user. The figures described in the next two pages show the block diagram for the whole DAB system and the DAB transmission system[5]. The high amount of function blocks forced to focus the work of this thesis in the last part of the transmission chain (Transmission system), from the Transmission frame multiplexer, so neither the channel coding blocks nor the time interleaving are present in the simulation. The simulation layout is described in standard terms in the Figure 3.2. All the work has been developed in baseband transmission, so the last block which defines the information about the RF transmitter has not been included. By working in baseband enables us to see the DAB characteristics, like the network gain showed in the analysis of the DAB transmitting and receiving signals. The data error rate is also available in order to realize how much data error protection should be included in the channel coding part [5]. Finally the digital mapping signal is obtained to see how much phase interference is produced by the multipath channel, so a very complete knowledge can be achieved running the proposal simulation.

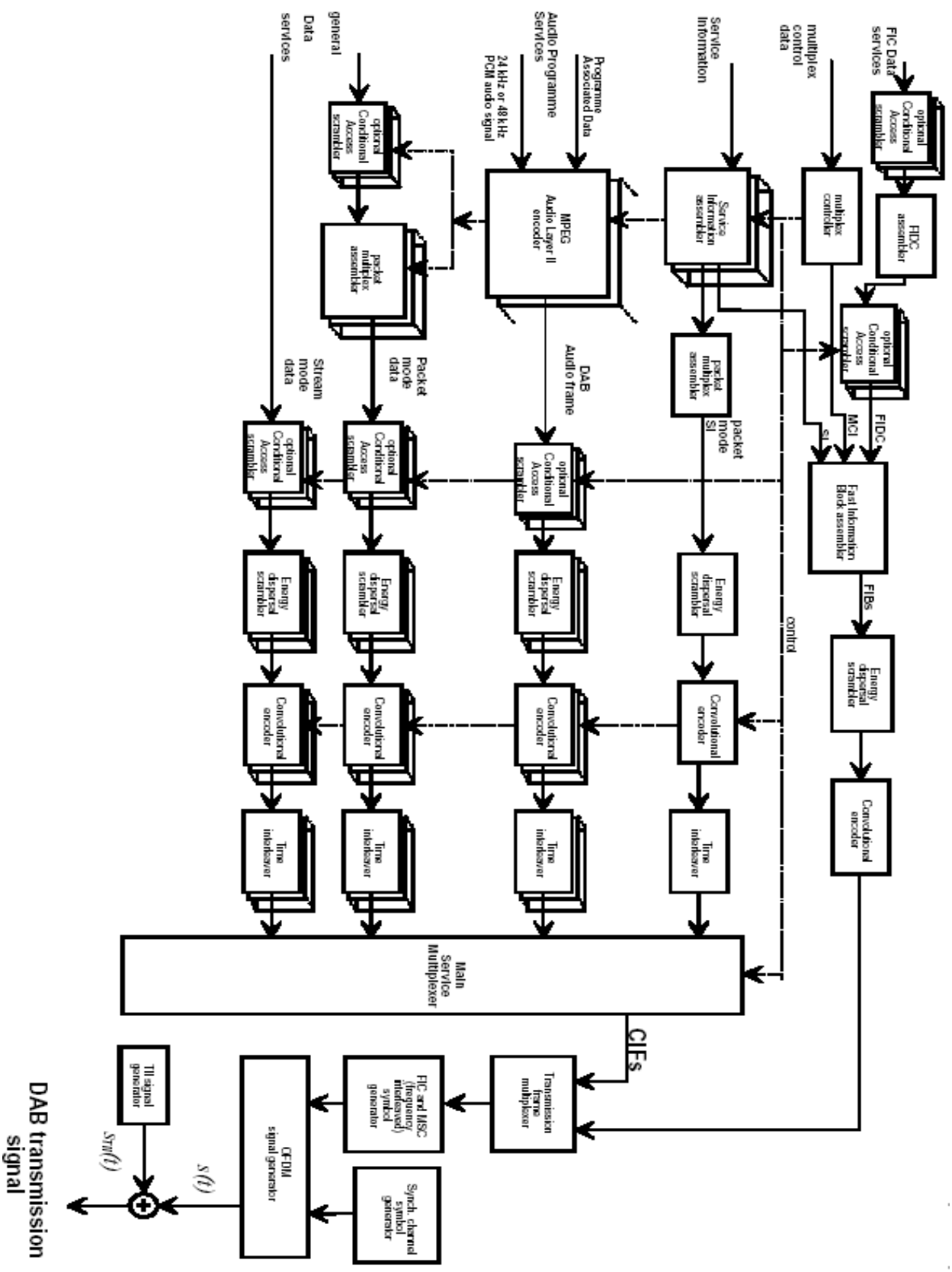

Figure 3.1 DAB scheme.

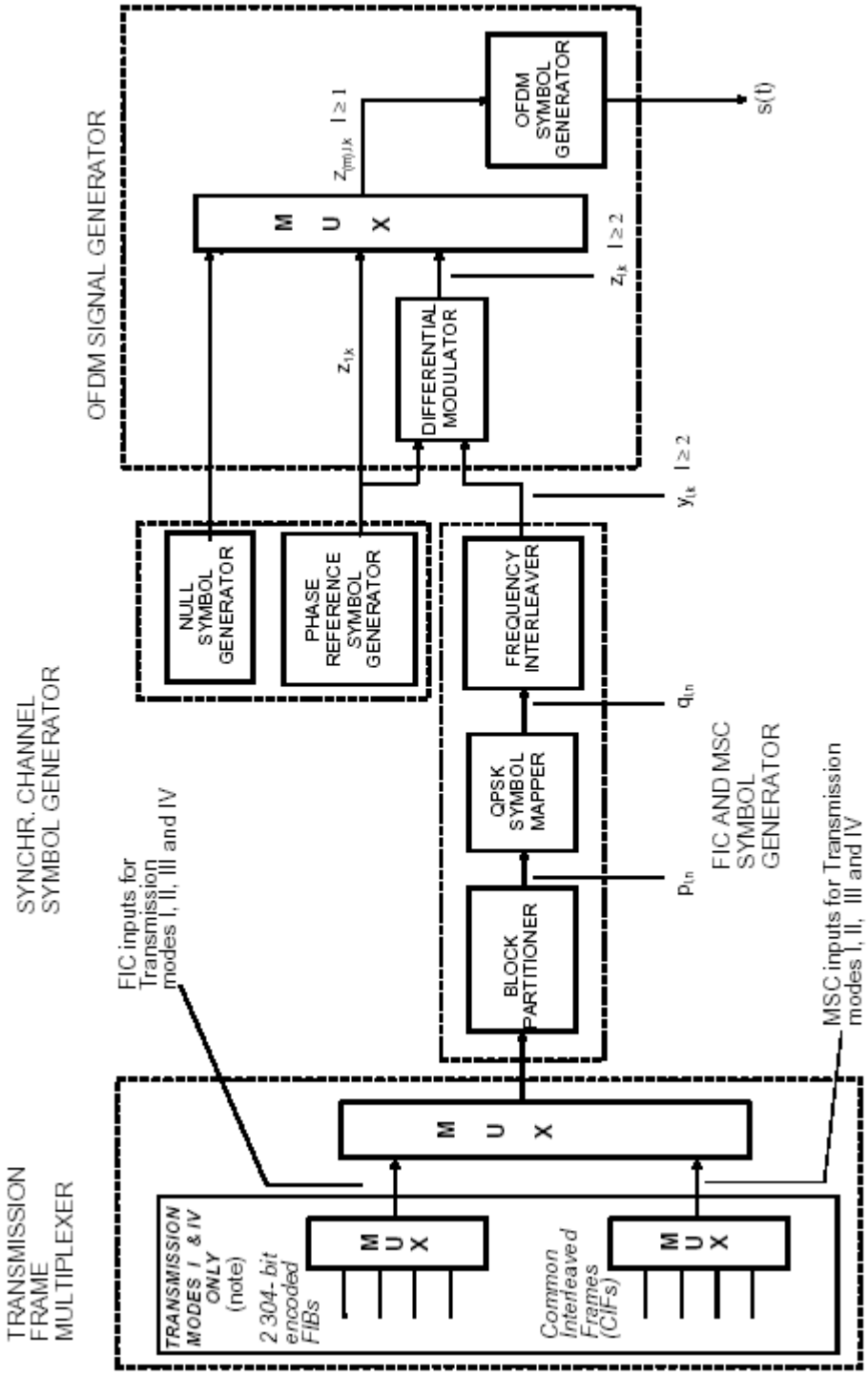

Figure 3.2 DAB Transmission scheme.

The software selected to develop the model, has been Matlab™ and its simulation toolbox Simulink™ . The reason to select them is because Matlab™ is a very common software tool for engineering so other simulations like this one had

already been designed with it. Even more, a lot of libraries and toolboxes in communications and digital signal processing mainly, are offered by Simulink™. As it has been said, not too many M-Matlab™ functions have been used, just in a few blocks and in an absolute necessary way, otherwise blocks would not work in the appropriate way. The complete model layout is going to be exposed in this chapter, and the programs used to make it work has been added in the Appendix section, so the simulation can be set by the reader in order to work with it or even to continue with it, by adding other function blocks to complete the whole DAB system.

The way to represent the blocks is going to follow the data stream. So it is started with the sources and it is ended with the error rate calculation block used to get some of the results achieved with the DAB standard showed in chapter 1 and 2.

First of all, as it was already said, all of the four transmission modes can be simulated, so all of their own characteristics must be set up before the simulation begins. To do that in a transparent way for the user, Simulink™ allows to configure, within the File menu, the model properties by pressing the "Callbacks" tab and by using a group of user functions. Two of the fields are used, the "Model pre-load function" and the "Model initialization function". The first one defines all of the parameters used in the simulation so no "parameter or function unknown" errors are generated when the DAB simulation is called. Like the rest of the functions it is showed in the Appendix section, it loads the Transmission Mode I parameters just because it was the first mode to be designed. The second function, is made in a "tab menu" way, so the user can loads the parameters of any of the transmission modes just clicking with the mouse in the desired transmission mode tab, so no parameter knowledge is needed to run this DAB simulation.

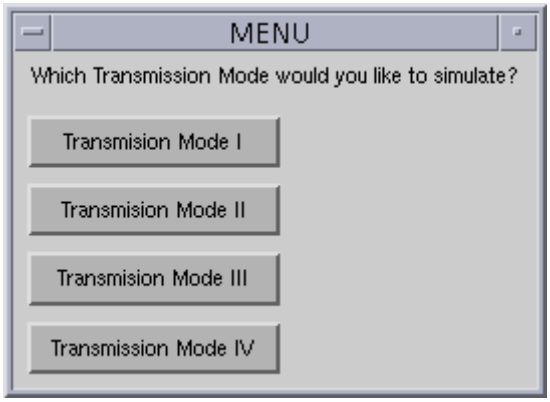

Figure 3.3 Transmission Mode selection menu.

The main parameters loaded by the menu in the model are showed in the next table [2][5]. There are some other parameters loaded by the menu but they will be better exposed in their own model sections because they result from a M function call or because they have to be used in order to work in the way Matlab™ works, just to adapt the standard to it so a deeper explanation is necessary. Some of them are time parameters, related to the only global parameter of the

transmission system and transport DAB network, which is T the period of the system clock which works at 2.048 MHz, and this value, T (sample time) is loaded in the "Model pre-load function". The result of multiplying T by the frame length results in the duration of the transmission frame for each mode. So the simulation time parameter is forced to be frame length by T in order to get a simulation with one complete frame.

Mode	*Frame length*	*FIC length (bits)*	*MSC length (bits)*	*OFDM symbol length*	*Block length (bits)*	*Null symbol duration*	*FFT length*	*Guard Interval*
TM I	196608*T	2304	55296	2552*T	3072	2656*T	2048	504*T
TM II	49152*T	576	13824	638*T	768	664*T	512	126*T
TM III	49152*T	768	13824	319*T	384	345*T	256	63*T
TM IV	98304*T	1152	27648	1276*T	1536	1328*T	1024	252*T

Table 3.1 Simulation parameters.

The complete simulation block diagram is shown in the figure in the next page. In the figure, the transmission mode ran was the transmission mode I, so the signal sizes belong to its parameters.

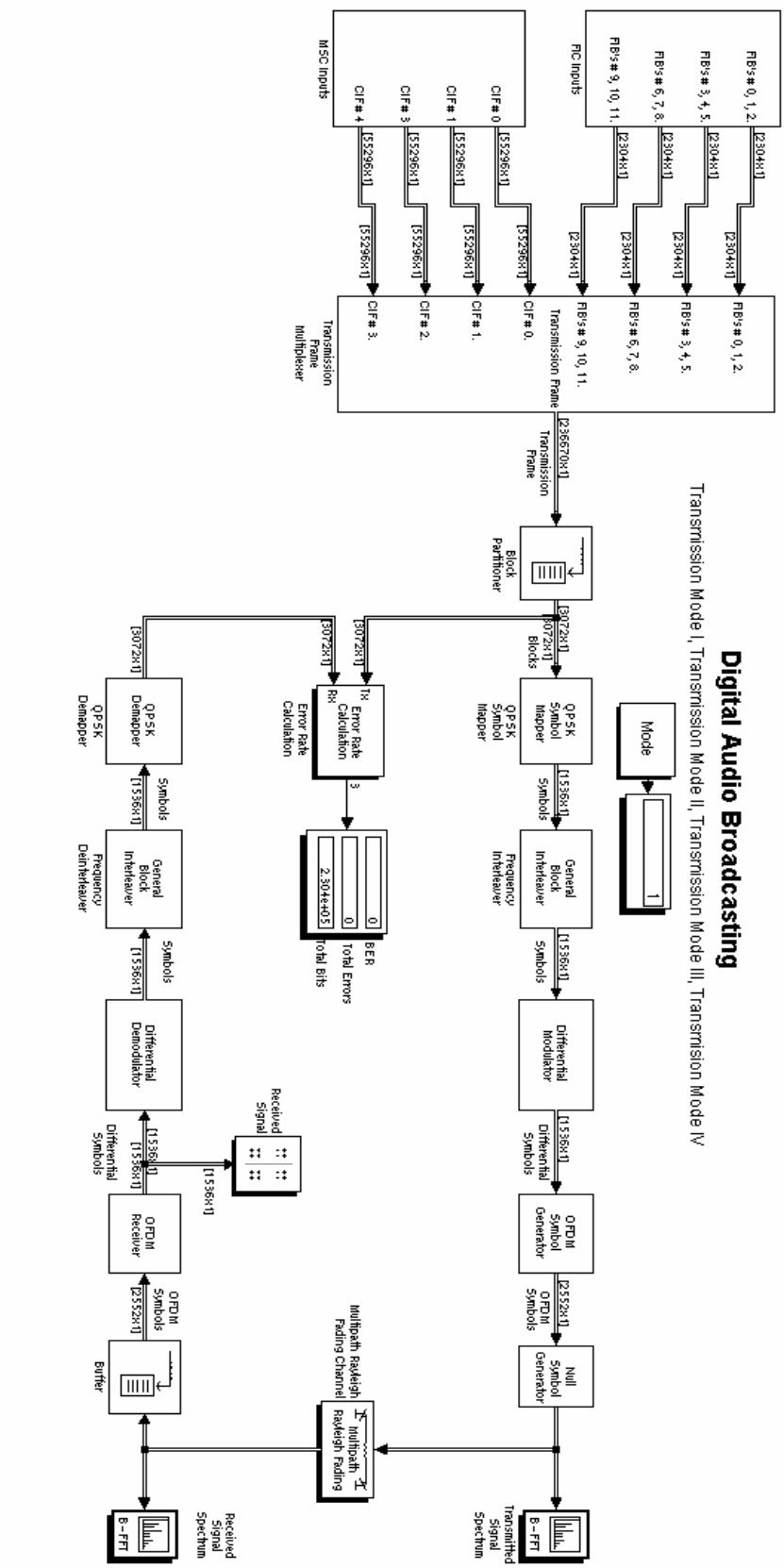

Figure 3.4 Simulation block diagram.

3.2 FIC (Fast Information Channel) and MSC (Main Service Channel) blocks

The data sources of the transmission system, the Fast Information Channel and the Main Service Channel, have been chosen by two different reasons. The first one is that both of them are multiplexed to conform the transmission signal, so the user is able to understand where the information comes from. The second reason is, according to the fact that the Transport Network, which delivers the DAB signal to all of the DAB network transmitters [2], is working with packets made up of group of CIF's (Common Interleaved Frames), that is the MSC, and with packets made up of FIB's (Fast Information Blocks), that is the FIC, as well. Both of them are received in the network transmitter side, so they are multiplexed to create the transmission signal as is shown in the Figure 3.2.

The information delivered by these two logical channels has to be exposed. The FIC is used as a support data to control the configuration of the DAB multiplex. There are several fields where the receiver can recognize patterns in order to decode the sub-channels presented in the signal. Therefore, we are talking about information which has to be delivered in a fast way so no time interleaving process is implemented, as is shown in the Figure 3.1. The FIC is made up of a number of FIB's. Each FIB is formed by 32 bytes: 2 bytes for error correction that is the CRC, and the 30 remaining are filled with data, so 256 bits is the length of one FIB. Depending on the transmission mode used, different number of FIB's are multiplexed in one transmission frame to form the FIC, these values are shown in the next page in the table 3.2.

The MSC is the logical channel where the information of the programs is carried. It is not usual that one of the services consumes all the MSC capacity so it is divided in sub channels as well and they are formed by an integral number of CU's (Capacity Unit) which is the smallest part of the MSC, 864 CU's shape one CIF, which contains 55296 bits. The data are convolutionally encoded and time interleaved also. Then they are sent across the transport network to the transmitters. Depending on the transmission mode different combination of CIF's are multiplexed in the transmitter to form the MSC in the transmission frame. More information can be consulted in the standard [5].

Mode	***Duration of the transmission frame (ms)***	***CIF's per transmission frame***	***FIB's per transmission frame***
TM I	96	4	12
TM II	24	1	4
TM III	24	1	3
TM IV	48	2	6

Table 3.2 Composition of the DAB frame.

In the simulation layout the design shown belongs to the transmission mode I, in which there are 12 FIB's divided into 4 different blocks forming the FIC, so the first block is made with FIB's number 0, 1 and 2, and so on. MSC is made up of 4 independent CIF's. The layout for the rest of the modes does not need to be changed, because the rest of the modes can be expressed in terms of 4 FIB's and CIF's blocks, just decreasing the length of both fields, these values are shown in table 3.1.

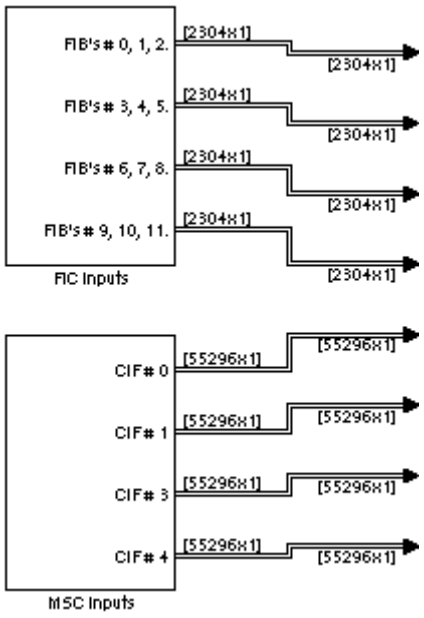

Figure 3.5 Transmission Mode I data sources.

Both subsystems are thought in the same way. By working with Matlab™ and Simulink™ there is the option to work with real DAB data taking the data stream from files for example, however taking into account that the simulation is not focused on the whole system it has got no sense to complicate the model. So inside the source subsystems, the “Bernoulli binary generator” which belongs to the communication blocksets sources library in the Communications blockset, generates the random data flow. The “frame-based output” option is enable in order to work with vectors along the path. It is an important aspect of the simulation because it is easier to set the successive block parameters and it also makes work with the M functions easier since the data size is known in every single point of the simulation layout. To fit the length of the frame with the standard value is necessary to change the “sample time” parameter of the sources, otherwise a longer or shorter data flow would be generated. So in the duration of the frame just one CIF and three FIB's should be generated in every single “Bernoulli” data source, so the sample time for the FIC source should be:

$$\frac{T \cdot (\text{Frame length} + 1)}{\text{FIC length}}$$

Where the FIC length indicates the length of every one of the four groups with three FIB's blocks each. A similar solution is taken with the CIF source.

So the “frame length” parameter has to be set to the MSC length and the FIC length values shown in the Table 3.1.

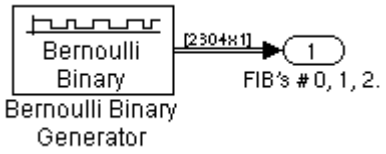

Figure 3.6 FIC source block.

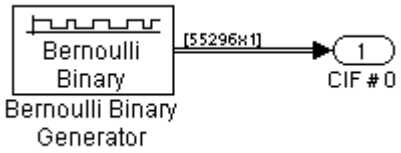

Figure 3.7 MSC source block.

3.3 Transmission Frame Multiplexer

The block designed to multiplex the DAB information channels is the Transmission Frame Multiplexer, as is shown in the Figure 3.2 its inputs are the FIC and MSC. Later by the Phase Reference Symbol Generator and the Null Symbol Generator the Synchronization Channel is added forming this way, finally the DAB transmission signal. The next figure shows the proposal layout for this subsystem.

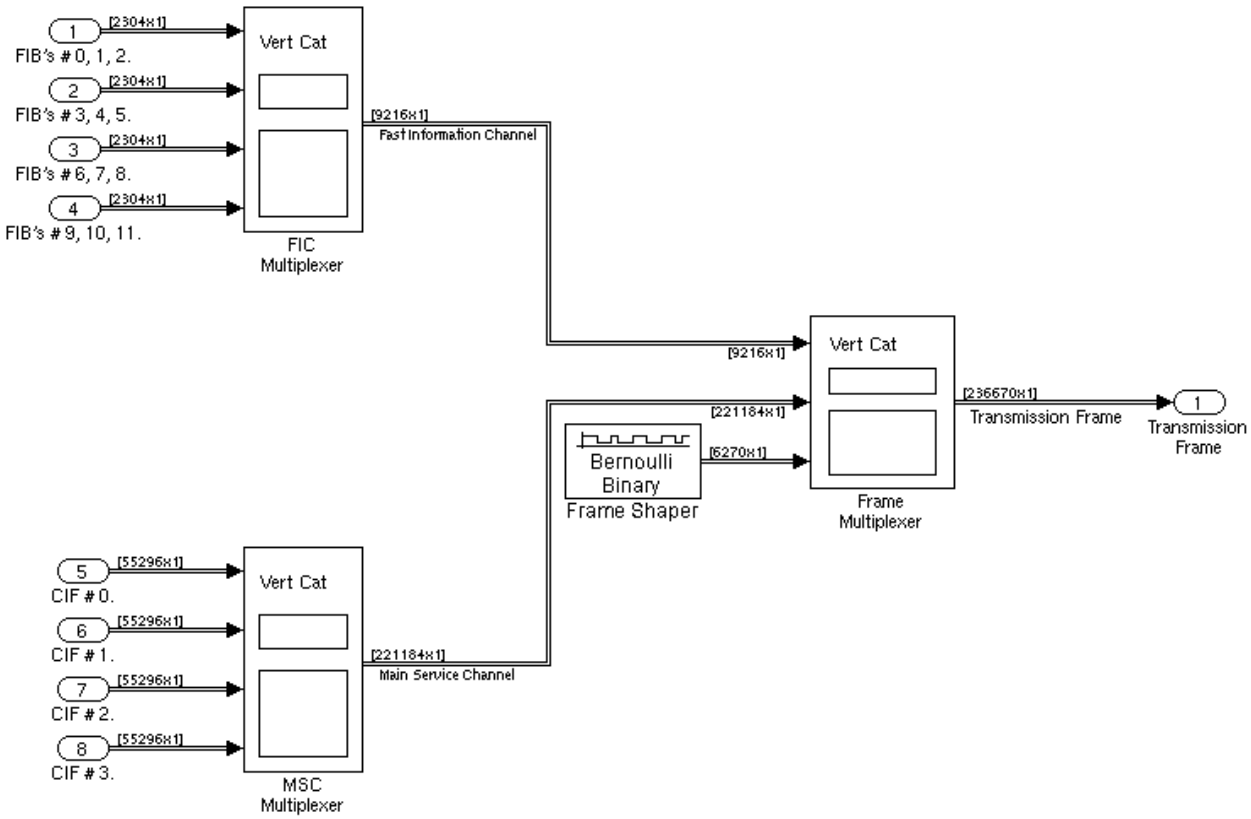

Figure 3.8 Transmission Frame Multiplexer.

As multiplexers, the matrix concatenation blocks from the vector operations in the math operations blockset in Simulink™ are used. They work in the same way as a normal multiplexer device, so all the FIB's are joint in the FIC by the FIC matrix concatenation and all the CIF's in the MSC by the MSC matrix concatenation. The subsystem shows the length of both channel, for every single transmission mode, here the transmission mode I is shown again. The frame shaper is one of the systems added to the simulation in order to be able to work with Simulink™, and its use has to be deep exposed. For a better understanding the next figure shows the whole DAB transmission frame, including the synchronization channel.

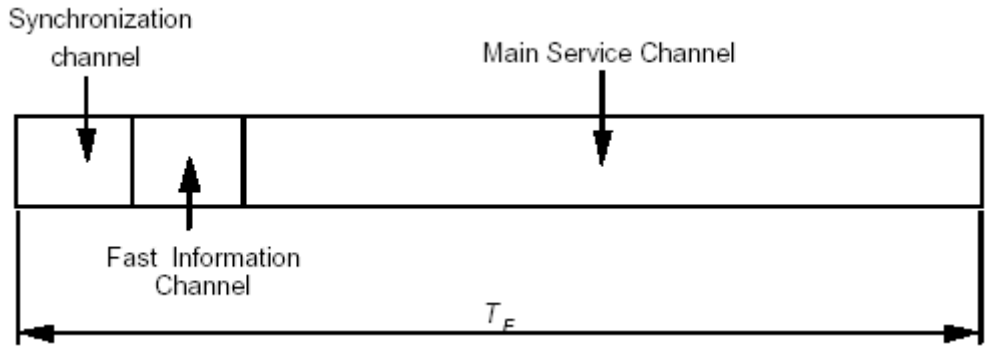

Figure 3.9 DAB Transmission Frame.

So we can see that the first input to the frame multiplexer, is the FIC, and the second one is the MSC just as it is shown in the scheme. It is no easy to add or remove vectors from the frame or even it is not possible to work with different vector lengths (the Null Symbol is larger than the rest of the OFDM symbols, see Table 3.1). If a vector work mode is used as it was explained before. So, it is necessary to create in this point of the simulation system, the length of the whole frame, just in the same way as a zero-padding system works. When the synchronization channel is created in the last subsystems of the transmission chain, the empty part will be fill up with the Phase Reference Symbol and zeros, that is the Null Symbol. It is set to be the third input to the frame multiplexer, but the Figure 3.9 shows that it should be the first one to follow the scheme. In the section 3.7, the way the Differential Modulator works in the simulation is exposed, and there it will be easier to understand the decision adopted. The Differential Modulator works with a delay block, in order to simulate the behavior of the difference equation which rules the DAB Differential Modulator block. This delay block shifts the whole frame by adding blocks filled with zeros at the beginning of the frame, so the last blocks will disappear when the whole frame will be shifted in order to keep the frame length. The next figure shows the frame transformation after the Differential Modulator.

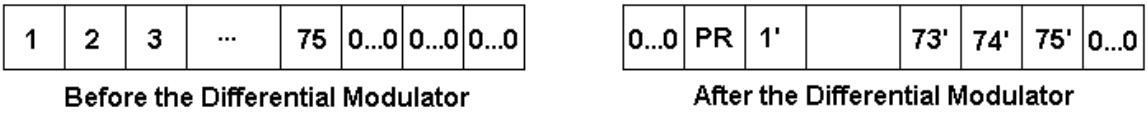

Figure 3.10 Frame shift in the Differential Modulator.

Now the reason to add in this point of the system the length of the Synchronization Channel is already explained. The amount of the "space" added by the "Frame Shaper" block must be exposed. As it was described the Null Symbol is larger than the rest. The Phase Reference Symbol has got the same length than the OFDM symbols. Is not possible to work with different vector lengths in Matlab™ so three blocks more are needed, two for the Null Symbol and one for the Phase Reference Symbol. But in this point of the layout we can not talk about blocks yet since the "Block Partitioner" has not been used yet. In fact we have to add bits in this moment and the number of bits to add results from the next relation

$$\text{Number of bits} = 2 \cdot \text{Block length} + \left\lceil \frac{\text{Block length} \cdot (\text{Null Symbol length} - \text{OFDM length})}{\text{OFDM length}} \right\rceil$$

The second term of the expression is defined in order to get enough bits for two blocks more, and a number depending on the transmission mode for the extra length needed for the Null Symbol defined as an integral number of bits. This extra

length depends on the relation between three transmission mode parameters. First of all the difference between the length of the Null Symbol and the OFDM symbols must be calculated, (e.g.,. for transmission mode I that difference will be 104, see Table 3.1), but it has to be expressed in terms of the relation between the transmission mode block and OFDM symbols length, in order to maintain the frame length after the Null Symbol generator which works in a non frame base output. So it takes into account only for bits generated before the "Block Partitioner". The number of bits added in the transmission mode I, as is shown in the Figure 3.8, is 6270, (2*3072 + 126), but after the Block Partitioner we can only work with blocks, so three whole blocks are added. The point is that in Null Symbol Generator the system does not care about blocks, but about bits, so they must be defined in this exact way.

3.4 Block Partitioner

Following the example for the transmission mode I, 236,670 bits are generated after the "Transmission Frame Multiplexer" block, which would result in 77. 041..... blocks with 3072 bits long, but as it has been said, an integral number of blocks is obligatory in a frame base work mode in Matlab™ so 78 blocks are generated by the block partitioner as it will be shown in the results. The next table shows the number of blocks for the different transmission modes.

Transmission Mode	*Symbols in the simulation*	*OFDM symbols per frame (without Null Symbol)*
TM I	78	76
TM II	78	76
TM III	155	153
TM IV	78	76

Table 3.3 Number of Symbols in the simulation.

So this block works as a FIFO queue. It receives the incoming bits, and it sends out the block with the "block length" long appropriate, joining this way the bits into the corresponding blocks.

3.5 QPSK Symbol Mapper

The next block of the chain is the digital symbol mapper. The incoming signal is formed by a serial group of bits, these bits are mapped in parallel into a digital constellation according to the QPSK (Quadrature Phase Shift Keying) modulation

scheme. Two data streams are generated (called I and Q, see section 2.3). In the RF interface (not described in this thesis) these two parallel streams are low pass filtered to band limit each symbol, and then both of them are double sideband suppressed carrier modulated with a LO (local oscillator) to modulate the data symbols onto two carrier signals. The carrier signal is the same for both data streams, but with a 90º phase shift applied to one of the streams as it was said in section 2.3. So the resulting signals are orthogonal to each other. [3]

The next figure shows the proposal layout for the QPSK symbol mapper.

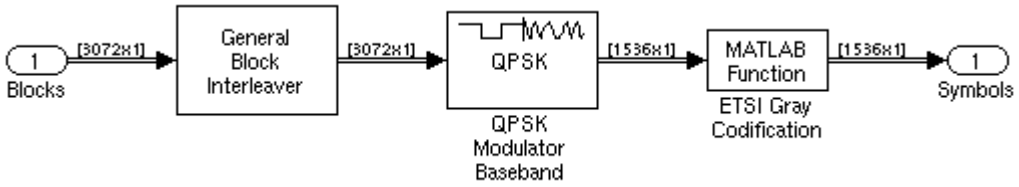

Figure 3.11 QPSK Symbol mapper.

This model follows in an exact way the definition to create the QPSK symbols according to the standard pattern, which is exposed in the next equation.

$$q = \frac{1}{\sqrt{2}} \cdot [(1 - (2 \cdot p_n)) + j \cdot (1 - (2 \cdot p_{n+k}))]$$

Where q is the symbol generated with two input bits, p_n and p_{n+k}, (the value for p can be 1 or 0) with k=(Block length)/2 and n=1,2,...,k. It means that there is a interleaving process in the block of bits before mapping them onto symbols. That is the reason to use the "General Block Interleaver" present in the Interleaving library from the Communication blockset. So in the transmission mode I, (see figure 3.11), the first symbol is generated with the bit number one (n=1) and the bit number 1537 (n+k=1537) and so on, so the last symbol of the block will be formed by the bit number 1536 (n=1536) and the last bit, 3072 (n+k=3072). The mathematical expression in Matlab™ to get this interleaving process is the following one.

```
[((1 + [1:Block_length]) / 2) .* mod([1:Block_length],2) + ((Block_length / 2) +
                ([1:Block_length] / 2)) .* (1 – mod([1:Block_length],2))]'
```

This expression must be the "elements" parameter in the "General Block Interleaver" so the main block in the QPSK mapper, the modulator, can be feed in the appropriate way. The next block used is the "QPSK Modulator Baseband". Setting its parameters in "bit" input mode and "Gray" codification mode. The Gray code is more efficient with noise-induced errors since just only one bit changes going from level to level, so if an error decision symbol is adopted, it is harder this way to have got both wrong bits in the symbol, and the channel codification scheme can work with less error data rate. So the mapping constellation resulting is the next

one.

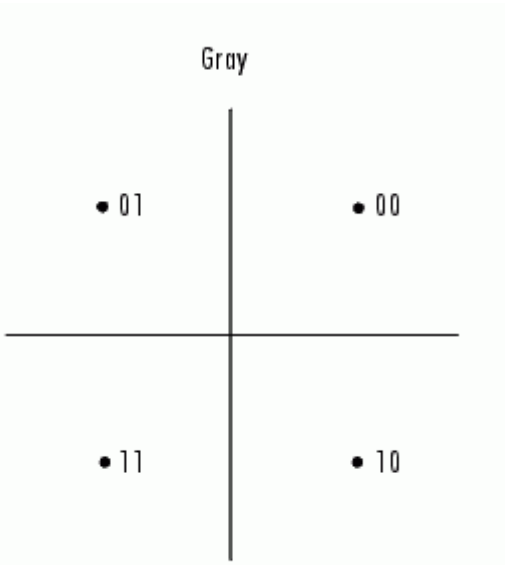

Figure 3.12 Gray QPSK constellation mapping in Matlab™

But according to the standard equation, the symbol (01) should have got a positive real part and a negative imaginary part, and the symbol (10) should have got negative real part but a positive imaginary part. That means that the mapping of these two symbols must be rearranged, and that is the mission of the next block, the "ETSI Gray Codification" block. The expression, this time is simple:

u .* sign(real(u)) .* sign(imag(u))

All the symbols are multiplied by 1 if its both real part and imaginary signs are equal to each other, so the mapping of the symbols (00) and (11) do not change, but if the signs are different the symbol is going to be multiplied by -1, so its place in the constellation is going to be the opposite, just to follow the standard constellation mapping.

3.6 Frequency Interleaver

In order to eliminate the effects of selective fading the frequency interleaving is present . It consists in a rearrangement of the symbol data stream over the carriers. The block to implement this function is the same block used in the QPSK Symbol Mapper, the "General Block Interleaver", but this time the interleaving process is different for the four transmission modes, and the table loaded in the "elements", which name is *[dab_freq_interleaver_table]*', parameter is the result of four different M-functions called from the menu to select the transmission mode to simulate. The M-code for these four interleaving functions are shown in the

Appendix section. To explain how the frequency interleaving works in DAB, we are going to keep on studying the example for the transmission mode I according to the standard [5].

Let $y_k = q_n$ be the relationship between the input and the output of the frequency interleaver, where "y" denotes the output from the frequency interleaver and "q" is the symbol generated by the mapper, with k=F(n). That is, the index k is obtained from the incoming index n using a mathematical expression F(n). This function is defined in the next terms.

Let Π(i) be a permutation in the set of integers i=0,1,...,2047 from the next relation:

$$\Pi(i) = [13*\Pi(i-1) + 511] \ (\text{mod } 2048) \text{ with } \Pi(0)=0$$

But there are only 1536 carriers in the transmission mode I, so taking account that F(n) is a one-to-one mapping relation between n and k so, let D be the set D= $\{d_0, d_1,...,d_{1535}\}$ containing 1536 elements in the same order than Π(i) but excluding the elements of Π which are not in the range [256, 1792] and excluding 1024, so $d_n = \Pi(i)$.Therefore the correspondence between n and k is:

$$k = F(n) = d_n - 1024.$$

So k Є {-768, -767,...,768} \ {0}.

The table generated by the interleaving function is exposed in the next page.

i	Π(i)	d_n	n	k
0	0			
1	511	511	0	-513
2	1010	1010	1	-14
3	1353	1353	2	329
4	1716	1716	3	692
5	291	291	4	-733
6	198			
7	1037	1037	5	13
8	1704	1704	6	680
9	135			
10	218			
11	1297	1297	7	273
12	988	988	8	-36
13	1076	1067	9	43
14	46			
15	1109	1109	10	85
16	592	592	11	-432
17	15			
18	706	706	12	-318
:	:	:		
2044	1676	1676	1533	652
2045	1819			
2046	1630	1630	1534	606
2047	1221	1221	1535	197

Table 3.4 Frequency Interleaving table for transmission mode I.

As it can be seen in the previous table, no k values are adopted when Π(i) is out of range. So the one-to-one mapping relationship between n and k is established. For n=0, k=-513; n=1, k=-14; etc.

This has been the interleaving process just as is shown in the standard, but Matlab™ and Simulink™ can not work with negative or zero index in the matrices, so a new rearrangement is necessary, 769 is added to the negative indexes and 768 to the positive indexes. So now k Є {1,2,...,1536}. And the relationship is transformed in Matlab™ and Simulink™, so now if n=1, k=256; n=2, k=755; n=3, k=1097; etc.

Similar functions are set for the rest of the transmission modes, the changes are the length of the permutation Π(i), and the length of the incoming data blocks, so other expression for Π(i) and other data valid ranges are defined, but the interleaving works just in the same explained in this section.

3.7 Differential Modulator

At this moment, the symbol blocks in the system belong to the QPSK constellation map, that is the information is carried by the phase of the signal. So if the phase is 45º, the symbol will be (00), so the bits in the data stream will be 00, etc. But there is a problem with this kind of transmission, the receiver has to be able to recognize the phase of the incoming carrier. This kind of transmission-reception

systems are called, coherent systems, and it has been exposed in the previous sections. In mobile communications the multipath spread of the signal can degrade the phase of the carriers, so the solution is sending the information not through the phase but through the difference between the phase of two successive symbols. The main advantage of this kind of technology is that the synchronization demands are lower than the coherent systems, in fact in the simulation layout there is no synchronization at all, and the system keeps on working.

The usual way in systems which work with differential digital modulations, to make possible this techniques, is creating a well known phase reference. This symbol uses to be the first symbol of the frame, like in the DAB standard, (without taking account the Null Symbol), so to get the first differential data symbol, the first incoming data block from the QPSK stream has to be multiplied by the Phase Reference Symbol. So this phase reference can be considered like the seed of the differential signal. It is very useful in this kind of multipath mobile communications, because it is generated by a fixed pattern, (in DAB it depends on the transmission mode used). The receiver also knows how is supposed to be the incoming phase reference, therefore, just comparing the pattern with the received Phase Reference Symbol, the receiver can get the behavior of the channel every single frame, (in DAB the Phase Reference Symbol appears once in the frame). This is a technique called “receiver training”, and it is very common working with Viterbi decoders and soft decision since the receiver has got information about the most probably changes in the data flow looking at the changes in the Phase Reference Symbol.

Following the standard, the differential modulation process is defined by the next difference equation:

$$z_n = z_{n-1} * y_n$$

Where “z” represents the differential symbols that is, the output of the differential modulator block (DQPSK symbols), and “y” represents the incoming QPSK data stream. The equation has to be translated into a layout block in the simulation. In order to be able to transform the output “z” into an input “z_{n-1}” a feedback loop is necessary, and the way to do that in Simulink™ is shown in the layout of the system.

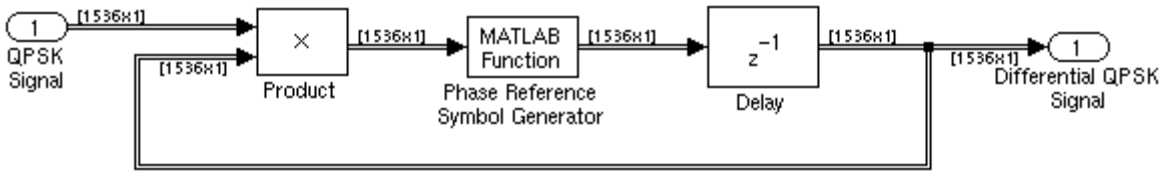

Figure 3.13 Differential modulator layout.

So the upper input would be “y_n” and the lower one would be “z_{n-1}”. The delay system as it was briefly exposed in section 3.3, allow us to work with the feedback

loop, just generating the initial condition values for the first "z_{n-1}" block, otherwise the differential modulator would generate an error. The value of this delay is set to -1, which means that only one "full zero" block will be added in the beginning of the frame (see Figure 3.10) so the parameter of the delay block shall be set to work with frames. This block would be considered like the main and big part of the Null Symbol, so there is no problem to put these zeros like the first symbol, in fact it is an advantage, because just changing the data rate (see section 3.9) the full Null Symbol will be generated.

So, when the first symbol, full up with zeros is obtained, the Matlab™ Function block has to generate the Phase Reference Symbol. This is the only function called in all the simulation, but it is essential. The M-functions created to this purpose are shown in the Appendix section. All of them works following the standard idea which is exposed next.

The definition of the DAB Phase Reference Symbol is:

$$z_k = e^{j\varphi_k}$$

Where k belongs to the range of integral numbers between 1 and the length of the DQPSK symbol blocks, which is the number of carriers for each transmission mode. The values of φ_k shall be obtained from the next expression:

$$\varphi_k = \frac{\pi}{2} \cdot \left(h_{i,k-k'} + n\right)$$

The indices i, k', and n are specified for the four transmission modes (see Appendix section). The parameter $h_{i,j}$ is obtained in common for the four transmission modes and the table to generate it is shown in the Appendix section as well.

When the differential modulation is applied the odd symbol constellation shall be the normal π/4 shifted QPSK constellation and the even symbol constellation shall be π/4 more shifted from the odd symbols, this is shown in the next figures.

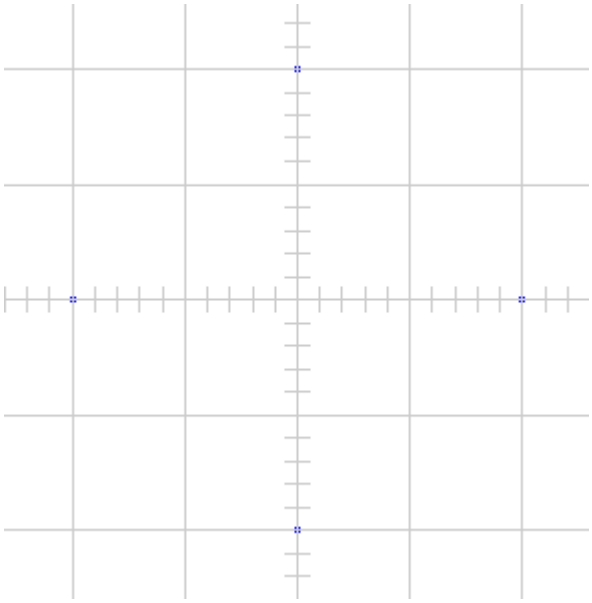

Figure 3.14 Constellation mapping for odd symbols.

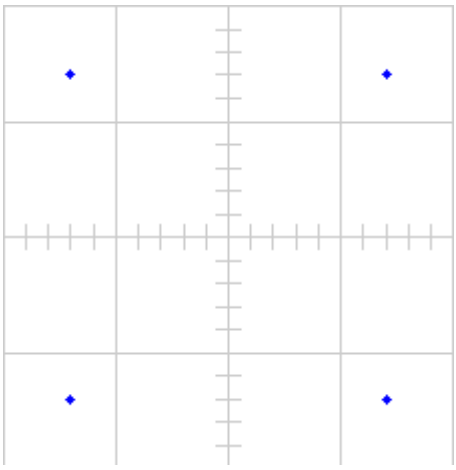

Figure 3.15 Constellation mapping for even symbols.

So finally the purpose to transmit the information in the phase difference between the successive symbols is achieved this way.

3.8 OFDM Symbol Generator

As it was said in the second chapter, the main transmission block in DAB is the OFDM Symbol Generator. The layout designed for this block is shown in the next figure.

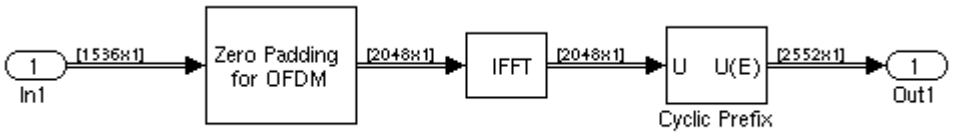

Figure 3.16 OFDM Symbol Generator design.

The OFDM technology has been already explained in the second chapter in this thesis, so this section is only intended to expose how the OFDM symbols are generated in the simulation.

3.8.1 Zero Padding for OFDM

The main part of this subsystem is the IFFT block, the FFT/IFFT algorithms have got a better performance if the number of carriers is a power of two (see section 2.3) [1]. Since the length of the DQPSK blocks is not a power of two, a zero padding is needed in order to fit the length of the blocks to the next power of two, (in the Transmission Mode I, from 1536 to 2048 for example, see Figure 3.16). The "Zero Padding subsystem is made up of two blocks.

Figure 3.17 Zero Padding for OFDM subsystem.

The "Zero Pad" block belongs to the Digital Signal Processing (DSP) toolbox in the Signal Operations library. There are just two parameters to be set, the length of the output, which is expressed by the FFT_length parameter (see Table 3.1 page #19), and the position of the padding, in the end of every single block. Now the length of the blocks is a power of two. The selector, (signal routing library) is necessary to rearrange the data because the IFFT changes the position of the carriers, if the "Selector" would be no present the carriers would follow the next allocation.

[1024, 1025, ..., 2048; 1, 2, ..., 1023]

The next figure exposes clearly the rearranged necessary, to work with the lower frequencies in the first part of each OFDM symbol and with the upper frequencies in the last part of the OFDM symbol.

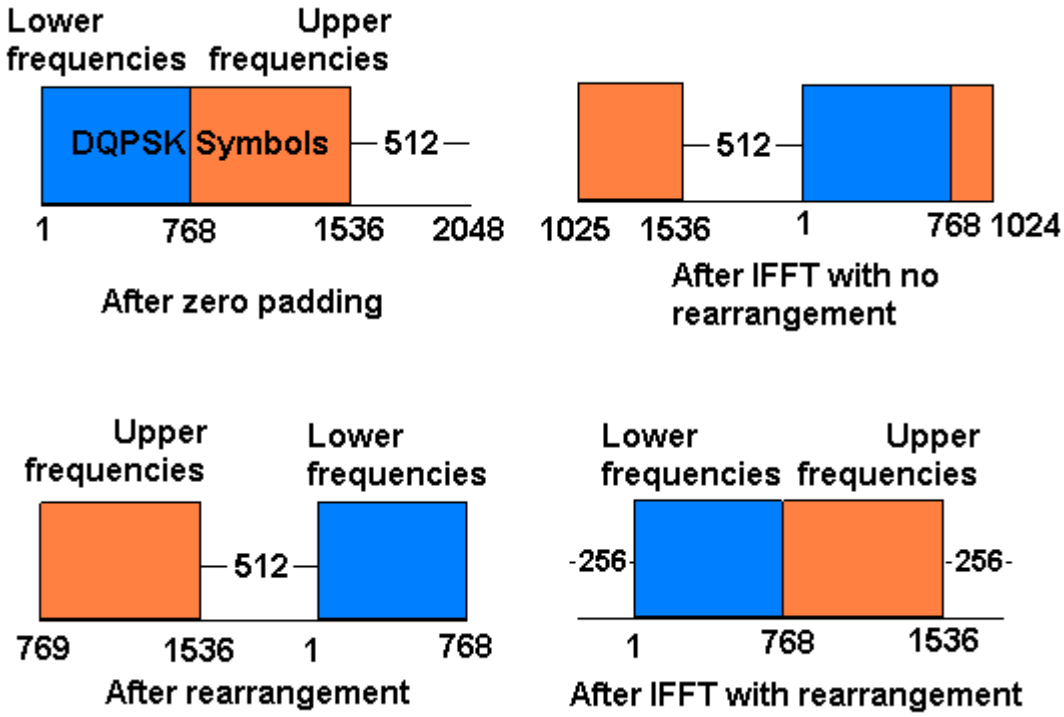

Figure 3.18 Rearrangement in the Zero Padding for OFDM (TM I example).

The expression to set in the "elements" parameter in the Selector is the next one:

[(Block_length/4) + 1:FFT_length 1:(Block_length/4)]

As the input port with, the parameter FFT_length shall be set.

Is easier now to see how this expression transforms the output from the zero pad block into the desired OFDM symbol form.

3.8.2 IFFT block

Now, with the block with the desired shape, the IFFT algorithm is implemented by the IFFT block, the main part of the whole simulation is the simplest block as well since there is no parameter necessary to set in the subsystem.

3.8.3 Cyclic Prefix

This is the system selected to create the cyclic prefix which was explained in the second chapter. The "Selector" block is used for this purpose again, with the

same value for the input port width than the previous selector block. The cyclic prefix is actually a copy of the one part of the OFDM symbol, so a new rearrangement is thought to develop the prefix, which is the Guard Interval. The expression to create it is the next one, and it shall be set as the "elements" parameter in the "Selector" block.

[FFT_length-Guard_Interval+1:FFT_length 1:FFT_length]

So at the beginning of each OFDM symbol a copy of the last part of the symbol is actually present, the length of this copy is equal to the length of the Guard Interval, (see Table 3.1 page #19). And finally the length of the OFDM symbols is the value of the parameter OFDM length (see Table 3.1 page #19). The next figure is similar to Figure 2.4 and it shows the final shape of the OFDM symbols.

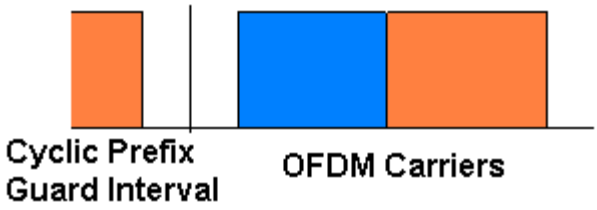

Figure 3.19 Complete OFDM Symbol.

3.9 Null Symbol Generator

This is the final block of the transmission chain. Its mission is to transform the data rate to allow the simulation system to work with a longer Null Symbol than the rest of the OFDM Symbols, just like the standard works. It has been thought to be as simple as possible and the layout is shown in the next figure.

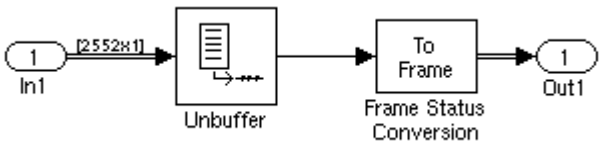

Figure 3.20 Null Symbol Generator Subsystem.

The "Unbuffer" system is present in the same library as the buffer block (see section 3.4). As it has been already exposed, the extra bits introduced in the Transmission Frame Multiplexer, are added to the beginning of the frame. So the transmitting signal is considered like one simple unit by the communication channel. This only unit is shaped in a frame by the "Frame Status Conversion" block which belong to the DSP toolbox in the Signal Management library. Simulating frames is faster than simulating non framed samples so the meaning of this last block is to improve the performance of the simulation.

The output of this subsystem is the signal s(t) as is shown in both Figure 3.1 and Figure 3.2, the DAB transmission signal defined in the ETSI standard [5].

3.10 Spectrum Scope

Following with the simulation chain, this block allow to the user to see the Transmitting and the Receiving DAB signal in the frequency domain. In the next chapter, where the results obtained in the different simulations will be exposed, the spectrum scope shall show the amplitude and the bandwidth of both signals, before and after the communication channel. So in these scopes the network gain can be seen just comparing the level of both amplitudes.

There are various parameters to be set, all of them belong to the "scope properties". The "Buffer input" is essential to be set with the OFDM_length value for every single Transmission Mode. Therefore, the Null Symbol will appear divided into two different blocks but it will be actually as long as the standard version, the thing is that is no possible to have got different lengths for the blocks to divide the incoming frame, so in order to be able to see every single OFDM symbol, this parameter shall be set as the length of these OFDM symbols, OFDM_length. The "FFT length" as its own name shows shall be set as the FFT_length. Finally the number of spectral averages has been set to be 2, like other of the similar simulation OFDM systems in Simulink™.

3.11 Multipath Rayleigh Fading Channel

The communication channel selected from the "Channels" library in the communication blockset in Simulink™, has been the Rayleigh Fading Channel, in order to simulate a multipath environment to work with mobile conditions to show the most eminent DAB properties. If fixed receiver communications were simulated, the most appropriate channel would be the AWGN (Average White Gaussian Noise).

There are four main parameters to be set. The first of them is the "Maximum Doppler shift" expressed in Hz, because of the relative speed between the mobile receiver and the fixed transmitter there is a frequency shift in the incoming signal. The next expression is used to get the value of this frequency shift.

$$f_{D\,max} = \frac{v}{c} \cdot f_0 \approx \frac{1}{1080} \cdot \frac{f_0}{\text{Mhz}} \cdot \frac{v}{\text{Km/}h}$$

As it will be exposed in the next chapter this parameter is the most important in the system. The faster the speed of the mobile receiver is the higher the binary

error rate is detected. The transmitter radio frequency is f_0, while v is the speed of the mobile.

The next parameter is the "Sample time" defined by Simulink™ as the time between two successive samples of the signal. So its value shall be the result of the following expression: T*Frame_length / (Frame_length – 1).

The last two parameters allow to the user to define the amount of paths to receive the transmitted signal. That is, with the "Delay vector" the user can establish the time difference between the main path and the rest, which number can be also established. The "Gain vector" is useful to decide the receiving signal power of the different paths, so the bigger delay, the bigger loss of signal power. In all the simulations two paths have been defined according with the different parameters from each Transmission mode.

Different propagation conditions can be simulated just changing these exposed parameters, so the communication channel is the only block allowed to me modified by the user, the rest of the systems shall not be modified.

3.12 The reception side

Nothing is said in the ETSI standard [5] about how to design the DAB receiver since all the receivers must be able to work with the DAB signal, there are several ways to develop a DAB receiver. In order to complete the transmission system to achieve some results about the DAB technology a receiver system was necessary. The easiest way to do that is by designing the symmetrical system to the transmitter. So in the next few subsections the receiver shall be exposed, and its understanding will be easier since all of the subsystems are reflected in the transmitter subsystems.

3.12.1 Buffer block

This block divides the frame into same long blocks, each of them containing one ODFM symbol, so the parameter "output buffer size" shall be OFDM_length in order to be able to work with data blocks again.

3.12.2 OFDM receiver

The most important system in the reception side is the OFDM receiver.

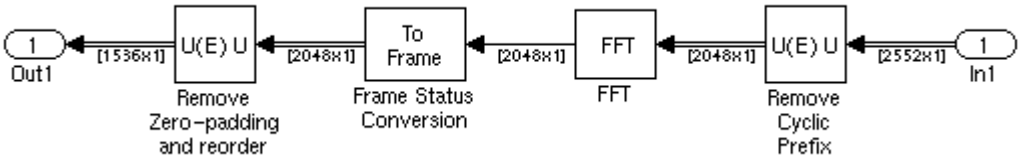

Figure 3.21 OFDM receiver subsystem.

The first block is used to remove the cyclic prefix, it works similar to the block exposed in section 3.8.3. The “elements” parameter shall be the next one.

[Guard_Interval + 1:OFDM_length]

Which means that only the OFDM information carriers (see Figure 3.19) shall be take to the next block of the reception chain. The input port width must be set to the OFDM_length parameter.

Now the length of the vector is set to be a power of two again, then the FFT block is used to transform the OFDM symbols again to the frequency domain. No parameter is necessary to set in this block.

The “Frame Status Conversion” block is used to work with vectors after the FFT algorithm since the output of the FFT block is formed by 1- dimension data units, so in order to keep on working with the same data type this block is added to the layout.

The last block in the OFDM receiver removes the zero padding and rearranges the data again to feed the Differential Demodulator with DQPSK symbol blocks. The “elements” parameter in this block is defined in the next terms.

[FFT_length – (Block_length/4) + 1:FFT_length 1:(Block_length/4)]

In the next figure the removal and rearrangement carried out in this block is shown.

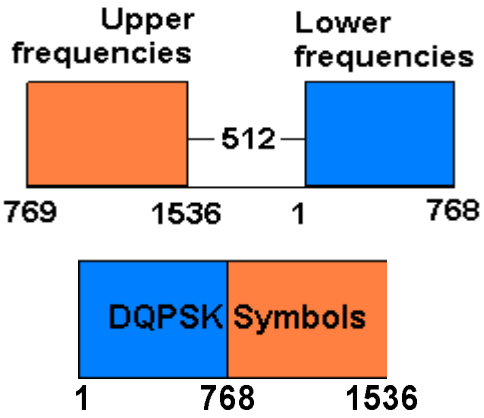

Figure 3.22 Zero padding removal and rearrangement in OFMD receiver.(TM I example).

So the output of the OFDM receiver is made up of DQPSK symbols placed in the same order than the output signal from the Differential Modulator. Lower symbol indices in the beginning of each block and upper indices behind them.

3.12.3 Discrete-Time scatter plot scope

This block, placed in the communication blockset in the communication sinks library, is used to show the DQPSK symbols received to the user in order to see the offset produced by the multipath effect over the different signal components defined in the communication channel block and the variation of the amplitude of the signal, so the gain network characteristic can be seen not only in the Spectrum scope block but in this time scope as well.

There are two parameters to be set in this block, the number of points displayed and the number of new points per display, both of them shall be set with the same value, Block_length/2, so in every single display one DQPSK symbols block
will be shown.

3.12.4 Differential demodulator

The difference equation which rules the behavior of the differential demodulator is symmetrical to the equation which manages the differential modulator. The expression is the next one.

$$y_n = z_n / z_{n-1}$$

Since there is no “divider” block available in Simulink™ the division has to be expressed as a multiplication. Z_{n-1} represents complex symbols so $1/Z_{n-1}$ can be

expressed as the complex conjugate of Z_{n-1}, so the difference equation for the Differential demodulator can be defined in the next terms.

$$y_n = z_n * \text{conj}(z_{n-1})$$

The layout for this subsystem is shown in the next figure.

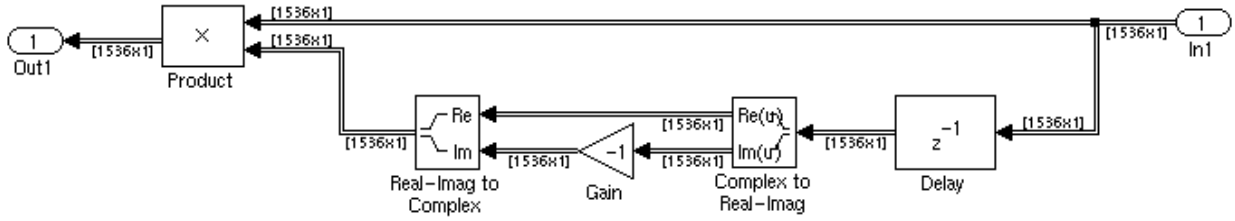

Figure 3.23 Differential Demodulator design.

As it can be seen, the differential demodulator is more simple than the modulator. The delay works with whole blocks, so its parameter must be set to work with frames as the delay in the modulator.

3.12.5 Frequency deinterleaver

This system works with a symmetrical table in comparing with the interleaving table present in the Frequency interleaver block. In fact the table to deinterleave the receiving signal is loaded with the interleaving table at time when the Transmission mode is selected in the beginning of the simulation time. (see the frequency interleaving functions for the four transmission modes in the Appendix section). The name of the table to put in the parameter box in this block is *[dab_freq_interleaver_table_rx]'*.

3.12.6 QPSK demapper

This is the last block belongs to the reception side. It transforms the the complex incoming QPSK symbols into bits again, so the BER (Binary Error Rate) can be established with the output of this system. Once again the layout of this block is symmetrical in relation to the QPSK mapper in the transmission chain.

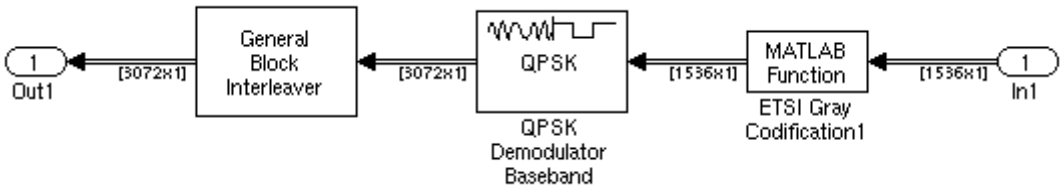

Figure 3.24 QPSK demapper system.

The function to translate the standard QPSK constellation into Gray Matlab™ QPSK constellation is exactly the same function used in the mapper. So the QPSK demodulator works bits as data type output and with Gray option on as block parameters. Finally the General Block Interleaver to rearrange the bits is expressed in the next terms.

[1:2:Block_length-1 2:2:Block_length]

So the indices of the bits are in order again.

3.12.7 Error rate calculation block

This block allows to the use to calculate the BER in a comfortable way, there is no need to get the both transmitting and receiving signal from the Matlab™ workspace using this block, which belongs to the sinks library in the Communication blockset.

There are just two parameters to set, the “receiver delay” which takes the value 3*Block_length for all of the Transmission modes. As it was shown in Figure 3.10 in page number 24, there are three symbols with no information in them, they are just added to create the space for the Synchronization channel, so they are not taken account when the BER is being calculated. The output port of this block shall be set as “port” in order to show the BER value and the number of errors in the simulation through a display.

4. RESULTS AND CONCLUSIONS

The aim of this thesis has been achieved since the transmission signal is exactly the same as the signal generated in the standard. Therefore some of the main characteristics of the DAB system can be tested with this simulation.

The first one is the so called Network Gain. In the next figure, both the transmitting and receiving spectrum signals shall be shown. This can be seen when the received symbols have got a higher power level than the transmitted ones. Even if the speed of the mobile is very fast some of the symbols are improved by this effect, (already explained in section 2.5)

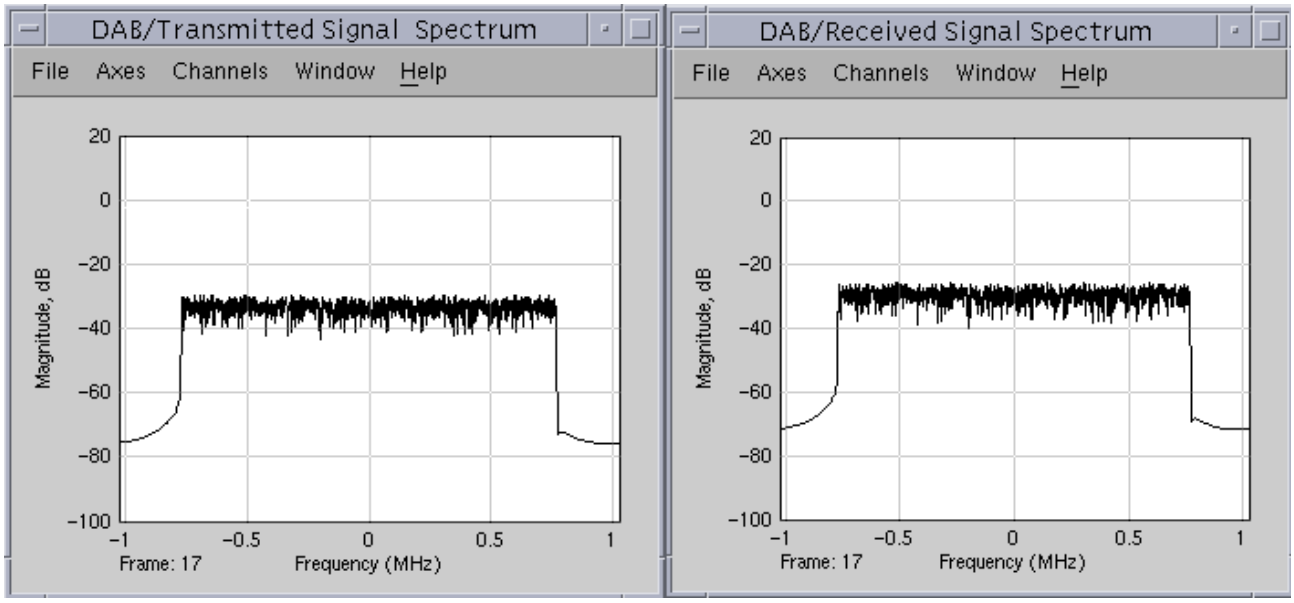

Figure 4.1 Network Gain characteristic.

So in this simulation (Transmission Mode I. 60 Km/h. f_0=240 MHz), in the symbol number 17 the level of the received signal is even higher than the transmitted. This figure also shows the bandwidth of the signal represented in baseband mode. All the symbols in the four Transmission modes have got the same bandwidth, 1.536 MHz, as it can be evaluated.

The offset added to the DAB signal by the multipath channel can be seen easily in the discrete-time scatter plot scope. The next figure shows that the symbols transmitted have not got the same phase as the symbols generated by the Differential Modulator (see Figures 3.14 and 3.15 in page number 31).

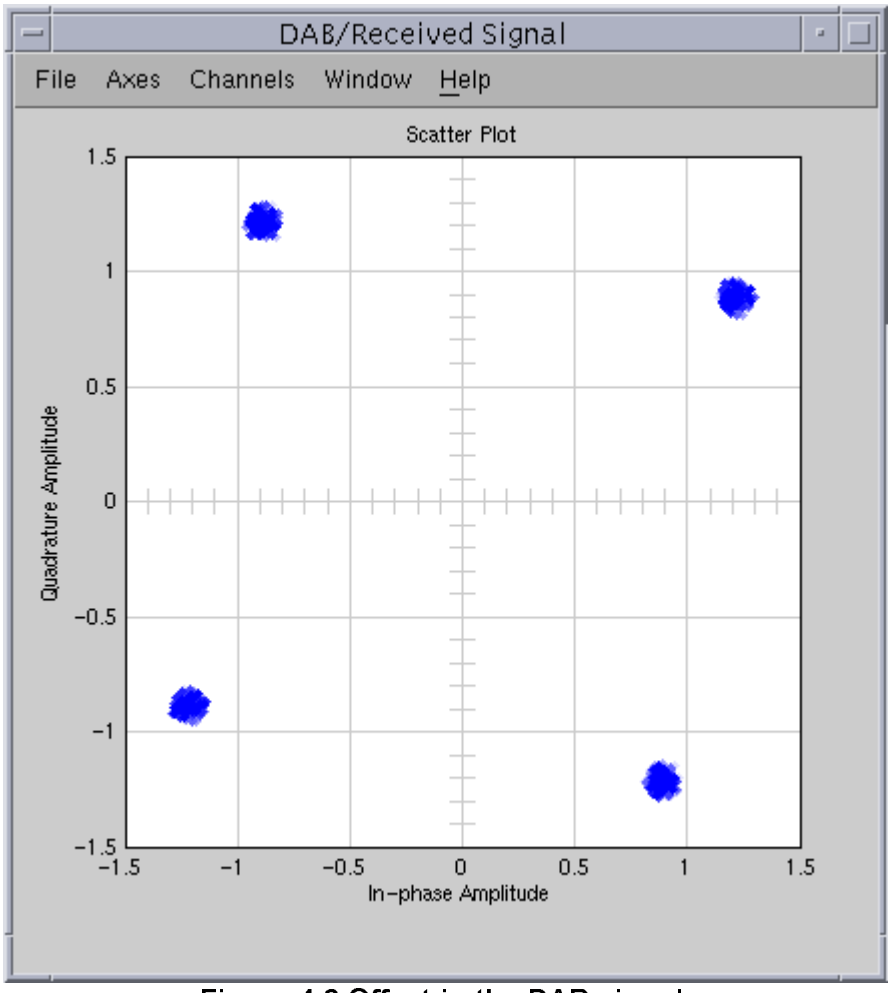

Figure 4.2 Offset in the DAB signal.

This offset is absolutely normal since the different signals across different paths are added then with different phase in the reception side. (Transmission Mode I. 200 Km/h. f_0=240 MHz). In all the simulation results shown in this chapter, two only paths have been included in the delay vector parameter in the channel block, to run the simulations as quick as possible.

It is easy to realize when error data are being generated just by looking at the time scope. If the points in the DQPSK constellation get very close to each other, then decision errors shall be produced. They can also be seen in the spectrum scope, if the level of the received signal is much lower than the transmitted one. The next figure shows both options.

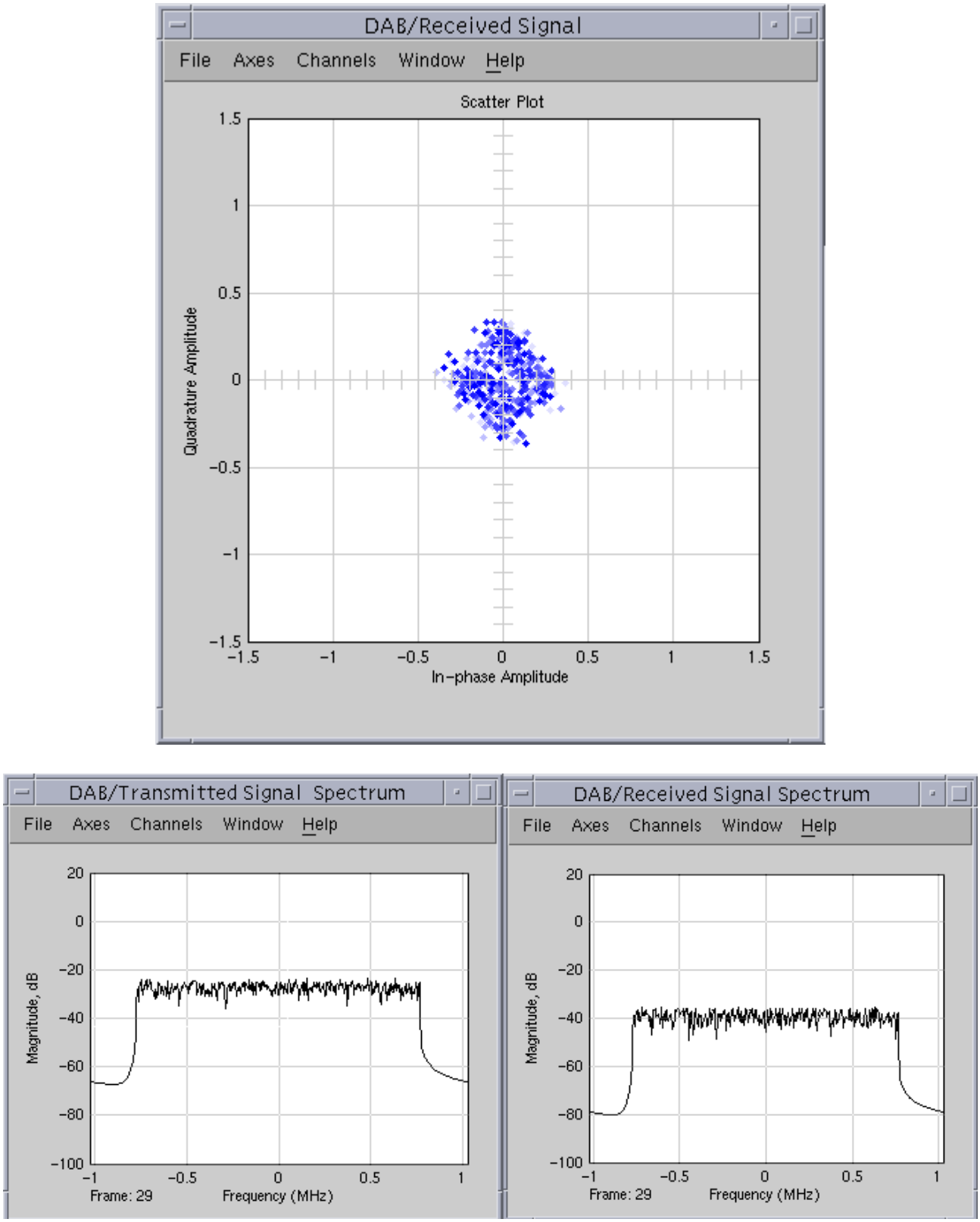

Figure 4.3 Error patterns in the scopes. (TM II. 200 Km/h. f_0=1.492 GHz)

The relationship between the receiver speed and the BER is shown in the table 4.1 where the results of the simulations are shown. It can be seen there, that the doppler effect is the most important since changes in the delay vector and delay gain vector do not influence as much as speed does. The results have been obtained using Monte-Carlo analysis in Simulink™.

Transmission Mode.	***60 Km/h***	***90 Km/h***	***120 Km/h***	***160 Km/h***	***200 Km/h***
TM I (240 MHz)	$3*10^{-3}$	$6*10^{-3}$	$8*10^{-3}$	$1*10^{-2}$	$3*10^{-2}$
TM II (1.492 GHz)	$9*10^{-3}$	$1*10^{-2}$	$3*10^{-2}$	$4*10^{-2}$	$6*10^{-2}$
TM III (3 GHz)	$6*10^{-3}$	$1*10^{-2}$	$3*10^{-2}$	$4*10^{-2}$	$6*10^{-2}$
TM IV (1.5 GHz)	$2*10^{-2}$	$4*10^{-2}$	$8*10^{-2}$	10^{-1}	$2*10^{-1}$

Table 4.1 BER results.

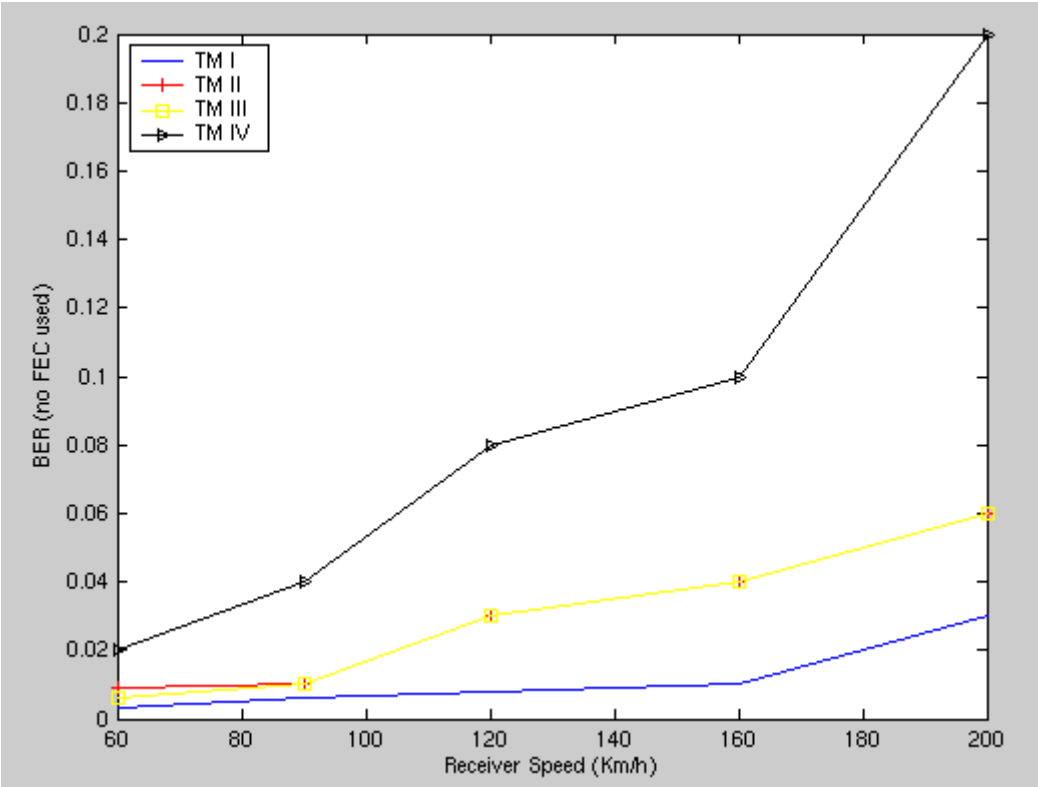

Figure 4.4 Graphical BER results.

A good BER for audio is considered to be 10^{-4}, so a channel encoder is necessary in all of the transmission modes even with low receiver speed. But these results show another DAB characteristic. There are several protection grades for the data depending on the transmission mode used, so a SFN working in mode I could have got a lower data protection and the throughput in the data stream would be increased. The Transmission mode I is the most spread one, so most of the radio networks are working according to it.

REFERENCES

[1] A. Bruce Carlson (1986). *Communication Systems. An introduction to Signals and Noise in Electrical Communication. McGraw Hill. Singapore.*

[2] Wolfang Hoeg, Thomas Lauterbauch (2001). *Digital Audio Broadcasting. Principles and Applications. Wiley. England*

[3] Seamus O'Leary (2000). *Understanding Digital Terrestrial Broadcasting. Artech House Inc. USA.*

[4] Ahmad R. S. Bahai, Burton R. Saltzberg (1999). *Multi-Carrier Digital Communications. Theory and applications of OFDM. Kluwer Academic / Plenum Publishers. USA.*

[5] ETSI EN 300 401 v1.3.3 (2001). *Radio Broadcasting Systems; Digital Audio Broadcasting (DAB) to mobile, portable or fixed receivers.*

[6] (1997). *Using Simulink™. The MathWorks, Inc. USA.*

APPENDIX

In this section all of the necessary M-functions are included to allow to the user to rebuild the simulation model without the Matlab™ files.

Model pre-load function (Model properties menu).

```
% Function to set DAB's parameters
% default transmission mode I
T=488.28125e-9;
Frame_length=196608;
FIC_length=2304;
MSC_length=55296;
OFDM_length=2552;
Block_length=3072;
Null_Symbol_Duration=2656;
Padding_length=2*Block_length+ceil((Null_Symbol_Duration-OFDM_length)
                                          *Block_length/OFDM_length);
freq_interleaver1;
FFT_length=2048;
Guard_Interval=504;
```

Model initialization function (Model properties menu)

```
% Function to load the transmission mode desired parameters
Mode=menu('Which Transmission Mode would you like to
simulate?','Transmision Mode I','Transmision Mode II','Transmision
Mode  III','Transmission Mode IV');
switch Mode
      case 1
            Frame_length=196608;
            FIC_length=2304;
            MSC_length=55296;
            OFDM_length=2552;
            Block_length=3072;
            Null_Symbol_Duration=2656;
            Padding_length=2*Block_length+ceil((Null_Symbol_Duration-
                              OFDM_length)*Block_length/OFDM_length);
            freq_interleaver1;
            FFT_length=2048;
            Guard_Interval=504;
      case 2
            Frame_length=49152;
            FIC_length=576;
            MSC_length=13824;
            OFDM_length=638;
            Block_length=768;
            Null_Symbol_Duration=664;
            Padding_length=2*Block_length+ceil((Null_Symbol_Duration-
                              OFDM_length)*Block_length/OFDM_length);
            freq_interleaver2;
            FFT_length=512;
            Guard_Interval=126;
      case 3
            Frame_length=49152;
```

```
            FIC_length=768;
            MSC_length=13824;
            OFDM_length=319;
            Block_length=384;
            Null_Symbol_Duration=345;
            Padding_length=2*Block_length+ceil((Null_Symbol_Duration-
                              OFDM_length)*Block_length/OFDM_length);
            freq_interleaver3;
            FFT_length=256;
            Guard_Interval=63;
      case 4
            Frame_length=98304;
            FIC_length=1152;
            MSC_length=27648;
            OFDM_length=1276;
            Block_length=1536;
            Null_Symbol_Duration=1328;
            Padding_length=2*Block_length+ceil((Null_Symbol_Duration-
                              OFDM_length)*Block_length/OFDM_length);
            freq_interleaver4;
            FFT_length=1024;
            Guard_Interval=252;
end
```

Frequency Interleaving functions.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Program to manage the Frequency Interleaver (Mode I)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% First af all, P sequence which length is
% 2048 (included 0 index) and k array which
% length is 1536 are generated
k(1)=-513;
P(1)=511;
m=2;
for i=2:2047
      P(i)=mod(13*P(i-1)+511,2048);
      if ((P(i)>=256)&(P(i)<=1792)&(P(i)~=1024))
            k(m)=P(i)-1024;
            m=m+1;
      end
end
% Possitive index are needed in Matlab, so:
for m=1:1536
      if k(m)<0
            k(m)=k(m)+769;
      else
            k(m)=k(m)+768;
      end
end
% Finally the table is rearranged to work
% according to the way that the General
% block Interleaver works in Simulink
for i=1:1536
      aux=k(i);
      dab_freq_interleaver_table(aux)=i;
      dab_freq_interleaver_table_rx(i)=aux;
end


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Program to manage the Frequency Interleaver (Mode II)
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% First af all, P sequence which length is
% 512 (included 0 index) and k array which
% length is 384 are generated
k(1)=-129;
P(1)=127;
m=2;
for i=2:511
      P(i)=mod(13*P(i-1)+127,512);
      if ((P(i)>=64)&(P(i)<=448)&(P(i)~=256))
            k(m)=P(i)-256;
            m=m+1;
      end
end
% Possitive index are needed in Matlab, so:
for m=1:384
      if k(m)<0
            k(m)=k(m)+193;
      else
            k(m)=k(m)+192;
      end
end
% Finally the table is rearranged to work
% according to the way that the General
% block Interleaver works in Simulink
clear dab_freq_interleaver_table
clear dab_freq_interleaver_table_rx
for i=1:384
      aux=k(i);
      dab_freq_interleaver_table(aux)=i;
      dab_freq_interleaver_table_rx(i)=aux;
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Program to manage the Frequency Interleaver (Mode III)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% First af all, P sequence which length is
% 256 (included 0 index) and k array which
% length is 192 are generated
k(1)=-65;
P(1)=63;
m=2;
for i=2:255
      P(i)=mod(13*P(i-1)+63,256);
      if ((P(i)>=32)&(P(i)<=224)&(P(i)~=128))
            k(m)=P(i)-128;
            m=m+1;
      end
end
% Possitive index are needed in Matlab, so:
for m=1:192
      if k(m)<0
            k(m)=k(m)+97;
      else
            k(m)=k(m)+96;
      end
end
% Finally the table is rearranged to work
% according to the way that the General
% block Interleaver works in Simulink
clear dab_freq_interleaver_table
clear dab_freq_interleaver_table_rx
for i=1:192
```

```
      aux=k(i);
      dab_freq_interleaver_table(aux)=i;
      dab_freq_interleaver_table_rx(i)=aux;
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Program to manage the Frequency Interleaver (Mode IV)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% First af all, P sequence which length is
% 1024 (included 0 index) and k array which
% length is 768 are generated
k(1)=-257;
P(1)=255;
m=2;
for i=2:1023
      P(i)=mod(13*P(i-1)+255,1024);
      if ((P(i)>=128)&(P(i)<=896)&(P(i)~=512))
            k(m)=P(i)-512;
            m=m+1;
      end
end
% Possitive index are needed in Matlab, so:
for m=1:768
      if k(m)<0
            k(m)=k(m)+385;
      else
            k(m)=k(m)+384;
      end
end
% Finally the table is rearranged to work
% according to the way that the General
% block Interleaver works in Simulink
clear dab_freq_interleaver_table
clear dab_freq_interleaver_table_rx
for i=1:768
      aux=k(i);
      dab_freq_interleaver_table(aux)=i;
      dab_freq_interleaver_table_rx(i)=aux;
end
```

Parameter h function.

This funciton is used to generate the values for the h parameter according to next table obtained from the standard:

j	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$h_{0,\,j}$	0	2	0	0	0	0	1	1	2	0	0	0	2	2	1	1
$h_{1,\,j}$	0	3	2	3	0	1	3	0	2	1	2	3	2	3	3	0
$h_{2,\,j}$	0	0	0	2	0	2	1	3	2	2	0	2	2	0	1	3
$h_{3,\,j}$	0	1	2	1	0	3	3	2	2	3	2	1	2	1	3	2

j	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
$h_{0,\,j}$	0	2	0	0	0	0	1	1	2	0	0	0	2	2	1	1
$h_{1,\,j}$	0	3	2	3	0	1	3	0	2	1	2	3	2	3	3	0
$h_{2,\,j}$	0	0	0	2	0	2	1	3	2	2	0	2	2	0	1	3
$h_{3,\,j}$	0	1	2	1	0	3	3	2	2	3	2	1	2	1	3	2

```
function h = parameter_h
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Program to generate the parameter h values
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
h=zeros(4,32);
h(1,2)=2;
h(1,7:8)=1;
h(1,9)=2;
h(1,13:14)=2;
h(1,15:16)=1;
h(2,2)=3;
h(2,3)=2;
h(2,4)=3;
h(2,6)=1;
h(2,7)=3;
h(2,9)=2;
h(2,10)=1;
h(2,11)=2;
h(2,12)=3;
h(2,13)=2;
h(2,14:15)=3;
h(3,4)=2;
h(3,6)=2;
h(3,7)=1;
h(3,8)=3;
h(3,9:10)=2;
h(3,12:13)=2;
h(3,15)=1;
h(3,16)=3;
h(4,2)=1;
h(4,3)=2;
h(4,4)=1;
h(4,6:7)=3;
h(4,8:9)=2;
h(4,10)=3;
h(4,11)=2;
h(4,12)=1;
h(4,13)=2;
h(4,14)=1;
h(4,15)=3;
h(4,16)=2;
h(1:4,17:32)=h(1:4,1:16);
```

Phase Reference generating function.

These functions generate the Phase Reference Symbol using k', i and n parameters obtained from the different tables in the standard.

Mode I

k in the range of		*k'*	*i*	*n*
min	max			
-768	-737	-768	0	1
-736	-705	-736	1	2
-704	-673	-704	2	0
-672	-641	-672	3	1
-640	-609	-640	0	3
-608	-577	-608	1	2
-576	-545	-576	2	2
-544	-513	-544	3	3
-512	-481	-512	0	2
-480	-449	-480	1	1
-448	-417	-448	2	2
-416	-385	-416	3	3
-384	-353	-384	0	1
-352	-321	-352	1	2
-320	-289	-320	2	3
-288	-257	-288	3	3
-256	-225	-256	0	2
-224	-193	-224	1	2
-192	-161	-192	2	2
-160	-129	-160	3	1
-128	-97	-128	0	1
-96	-65	-96	1	3
-64	-33	-64	2	1
-32	-1	-32	3	2

k in the range of		*k'*	*i*	*n*
min	max			
1	32	1	0	3
33	64	33	3	1
65	96	65	2	1
97	128	97	1	1
129	160	129	0	2
161	192	161	3	2
193	224	193	2	1
225	256	225	1	0
257	288	257	0	2
289	320	289	3	2
321	352	321	2	3
353	384	353	1	3
385	416	385	0	0
417	448	417	3	2
449	480	449	2	1
481	512	481	1	3
513	544	513	0	3
545	576	545	3	3
577	608	577	2	3
609	640	609	1	0
641	672	641	0	3
673	704	673	3	0
705	736	705	2	1
737	768	737	1	1

Matlab™ can not work with non positive indices so the in the left table 769 is added to all of the k and k' values. In the right table 768 is added to all of the k and k' values.

```
function Phase_Reference = phase_reference_symbol1
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Program to generate the Phase Reference Symbol
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% First af all generate the k', i and n arrays
% is needed
for k=1:1536
      switch k
            case 1
                  k_bis=1;
                  i=0;
                  n=1;
            case 33
                  k_bis=33;
                  i=1;
                  n=2;
            case 65
                  k_bis=65;
                  i=2;
                  n=0;
```

```
case 97
      k_bis=97;
      i=3;
      n=1;
case 129
      k_bis=129;
      i=0;
      n=3;
case 161
      k_bis=161;
      i=1;
      n=2;
case 193
      k_bis=193;
      i=2;
      n=2;
case 225
      k_bis=225;
      i=3;
      n=3;
case 257
      k_bis=257;
      i=0;
      n=2;
case 289
      k_bis=289;
      i=1;
      n=1;
case 321
      k_bis=321;
      i=2;
      n=2;
case 353
      k_bis=353;
      i=3;
      n=3;
case 385
      k_bis=385;
      i=0;
      n=1;
case 417
      k_bis=417;
      i=1;
      n=2;
case 449
      k_bis=449;
      i=2;
      n=3;
case 481
      k_bis=481;
      i=3;
      n=3;
case 513
      k_bis=513;
      i=0;
      n=2;
case 545
      k_bis=545;
      i=1;
      n=2;
case 577
      k_bis=577;
      i=2;
```

```
      n=2;
case 609
      k_bis=609;
      i=3;
      n=1;
case 641
      k_bis=641;
      i=0;
      n=1;
case 673
      k_bis=673;
      i=1;
      n=3;
case 705
      k_bis=705;
      i=2;
      n=1;
case 737
      k_bis=737;
      i=3;
      n=2;
case 769
      k_bis=769;
      i=0;
      n=3;
case 801
      k_bis=801;
      i=3;
      n=1;
case 833
      k_bis=833;
      i=2;
      n=1;
case 865
      k_bis=865;
      i=1;
      n=1;
case 897
      k_bis=897;
      i=0;
      n=2;
case 929
      k_bis=929;
      i=3;
      n=2;
case 961
      k_bis=961;
      i=2;
      n=1;
case 993
      k_bis=993;
      i=1;
      n=0;
case 1025
      k_bis=1025;
      i=0;
      n=2;
case 1057
      k_bis=1057;
      i=3;
      n=2;
case 1089
      k_bis=1089;
```

```
            i=2;
            n=3;
      case 1121
            k_bis=1121;
            i=1;
            n=3;
      case 1153
            k_bis=1153;
            i=0;
            n=0;
      case 1185
            k_bis=1185;
            i=3;
            n=2;
      case 1217
            k_bis=1217;
            i=2;
            n=1;
      case 1249
            k_bis=1249;
            i=1;
            n=3;
      case 1281
            k_bis=1281;
            i=0;
            n=3;
      case 1313
            k_bis=1313;
            i=3;
            n=3;
      case 1345
            k_bis=1345;
            i=2;
            n=3;
      case 1377
            k_bis=1377;
            i=1;
            n=0;
      case 1409
            k_bis=1409;
            i=0;
            n=3;
      case 1441
            k_bis=1441;
            i=3;
            n=0;
      case 1473
            k_bis=1473;
            i=2;
            n=1;
      case 1505
            k_bis=1505;
            i=1;
            n=1;
end
% Now it's time to calculate the phase reference symbol!
% First of all the phi(k) phase is needed, so the "h" parameter
% has to be determinated.
h=parameter_h;
phi(k)=(pi/2)*(h(i+1,k-k_bis+1)+n);
z(k)=exp(j*phi(k));
if abs(real(z(k)))<1
      Phase_Reference(k)=j*imag(z(k));
```

```
        else
            Phase_Reference(k)=real(z(k));
        end
end
```

Mode II

k in the range of		*k'*	*i*	*n*
min	max			
-192	-161	-192	0	2
-160	-129	-160	1	3
-128	-97	-128	2	2
-96	-65	-96	3	2
-64	-33	-64	0	1
-32	-1	-32	1	2

k in the range of		*k'*	*i*	*n*
min	max			
1	32	1	2	0
33	64	33	1	2
65	96	65	0	2
97	128	97	3	1
129	160	129	2	0
161	192	161	1	3

Matlab™ can not work with non positive indices so the in the left table 193 is added to all of the k and k' values. In the right table 192 is added to all of the k and k' values.

```
function Phase_Reference = phase_reference_symbol2
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Program to generate the Phase Reference Symbol
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% First af all generate the k', i and n arrays
% is needed
for k=1:384
      switch k
            case 1
                  k_bis=1;
                  i=0;
                  n=2;
            case 33
                  k_bis=33;
                  i=1;
                  n=3;
            case 65
                  k_bis=65;
                  i=2;
                  n=2;
            case 97
                  k_bis=97;
                  i=3;
                  n=2;
            case 129
                  k_bis=129;
                  i=0;
                  n=1;
            case 161
                  k_bis=161;
                  i=1;
                  n=2;
            case 193
                  k_bis=193;
                  i=2;
                  n=0;
            case 225
                  k_bis=225;
                  i=1;
                  n=2;
            case 257
```

```
                    k_bis=257;
                    i=0;
                    n=2;
              case 289
                    k_bis=289;
                    i=3;
                    n=1;
              case 321
                    k_bis=321;
                    i=2;
                    n=0;
              case 353
                    k_bis=353;
                    i=1;
                    n=3;
        end
        % Now it's time to calculate the phase reference symbol!
        % First of all the phi(k) phase is needed, so the "h" parameter
        % has to be determinated.
        h=parameter_h;
        phi(k)=(pi/2)*(h(i+1,k-k_bis+1)+n);
        z(k)=exp(j*phi(k));
        if abs(real(z(k)))<1
              Phase_Reference(k)=j*imag(z(k));
        else
              Phase_Reference(k)=real(z(k));
        end
end
```

Mode III

k in the range of		*k'*	*i*	*n*
min	max			
-96	-65	-96	0	2
-64	-33	-64	1	3
-32	-1	-32	2	0

k in the range of		*k'*	*i*	*n*
min	max			
1	32	1	3	2
33	64	33	2	2
65	96	65	1	2

Matlab™ can not work with non positive indices so the in the left table 97 is added to all of the k and k' values. In the right table 96 is added to all of the k and k' values.

```
function Phase_Reference = phase_reference_symbol3
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Program to generate the Phase Reference Symbol
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% First af all generate the k', i and n arrays
% is needed
for k=1:192
      switch k
            case 1
                  k_bis=1;
                  i=0;
                  n=2;
            case 33
                  k_bis=33;
                  i=1;
                  n=3;
            case 65
                  k_bis=65;
```

```
                i=2;
                n=0;
            case 97
                k_bis=97;
                i=3;
                n=2;
            case 129
                k_bis=129;
                i=2;
                n=2;
            case 161
                k_bis=161;
                i=1;
                n=2;
        end
        % Now it's time to calculate the phase reference symbol!
        % First of all the phi(k) phase is needed, so the "h" parameter
        % has to be determinated.
        h=parameter_h;
        phi(k)=(pi/2)*(h(i+1,k-k_bis+1)+n);
        z(k)=exp(j*phi(k));
        if abs(real(z(k)))<1
            Phase_Reference(k)=j*imag(z(k));
        else
            Phase_Reference(k)=real(z(k));
        end
    end
```

Mode IV

k in the range of		*k'*	*i*	*n*
min	max			
-384	-353	-384	0	0
-352	-321	-352	1	1
-320	-289	-320	2	1
-288	-257	-288	3	2
-256	-225	-256	0	2
-224	-193	-224	1	2
-192	-161	-192	2	0
-160	-129	-160	3	3
-128	-97	-128	0	3
-96	-65	-96	1	1
-64	-33	-64	2	3
-32	-1	-32	3	2

k in the range of		*k'*	*i*	*n*
min	max			
1	32	1	0	0
33	64	33	3	1
65	96	65	2	0
97	128	97	1	2
129	160	129	0	0
161	192	161	3	1
193	224	193	2	2
225	256	225	1	2
257	288	257	0	2
289	320	289	3	1
321	352	321	2	3
353	384	353	1	0

Matlab™ can not work with non positive indices so the in the left table 385 is added to all of the k and k' values. In the right table 384 is added to all of the k and k' values.

```
function Phase_Reference = phase_reference_symbol4
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Program to generate the Phase Reference Symbol
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% First af all generate the k', i and n arrays
% is needed
for k=1:768
    switch k
        case 1
            k_bis=1;
            i=0;
            n=0;
```

```
case 33
      k_bis=33;
      i=1;
      n=1;
case 65
      k_bis=65;
      i=2;
      n=1;
case 97
      k_bis=97;
      i=3;
      n=2;
case 129
      k_bis=129;
      i=0;
      n=2;
case 161
      k_bis=161;
      i=1;
      n=2;
case 193
      k_bis=193;
      i=2;
      n=0;
case 225
      k_bis=225;
      i=3;
      n=3;
case 257
      k_bis=257;
      i=0;
      n=3;
case 289
      k_bis=289;
      i=1;
      n=1;
case 321
      k_bis=321;
      i=2;
      n=3;
case 353
      k_bis=353;
      i=3;
      n=2;
case 385
      k_bis=385;
      i=0;
      n=0;
case 417
      k_bis=417;
      i=3;
      n=1;
case 449
      k_bis=449;
      i=2;
      n=0;
case 481
      k_bis=481;
      i=1;
      n=2;
case 513
      k_bis=513;
      i=0;
```

```
                        n=0;
                case 545
                        k_bis=545;
                        i=3;
                        n=1;
                case 577
                        k_bis=577;
                        i=2;
                        n=2;
                case 609
                        k_bis=609;
                        i=1;
                        n=2;
                case 641
                        k_bis=641;
                        i=0;
                        n=2;
                case 673
                        k_bis=673;
                        i=3;
                        n=1;
                case 705
                        k_bis=705;
                        i=2;
                        n=3;
                case 737
                        k_bis=737;
                        i=1;
                        n=0;
        end
        % Now it's time to calculate the phase reference symbol!
        % First of all the phi(k) phase is needed, so the "h" parameter
        % has to be determinated.
        h=parameter_h;
        phi(k)=(pi/2)*(h(i+1,k-k_bis+1)+n);
        z(k)=exp(j*phi(k));
        if abs(real(z(k)))<1
                Phase_Reference(k)=j*imag(z(k));
        else
                Phase_Reference(k)=real(z(k));
        end
end
```