UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE INGENIERÍA ELÉCTRICA

DESIGN, IMPLEMENTATION AND TESTING OF A REAL-TIME ELECTROMAGNETIC INTERFERENCE DETECTOR AND CLASSIFIER FOR THE FIVE-HUNDRED-METER APERTURE SPHERICAL TELESCOPE (FAST)

MEMORIA PARA OPTAR AL TÍTULO DE INGENIERO CIVIL ELÉCTRICO

DANIEL BRAVO LIBERONA

PROFESOR GUÍA:
RICARDO FINGER CAMUS

MIEMBROS DE LA COMISIÓN:
ANDRÉS CABA RUTTE
SEBASTIÁN JORQUERA TAPIA

Este trabajo ha sido financiado por ANID a través de sus fondos BASAL FB210003
y ALMA 31180005

SANTIAGO DE CHILE
2021

RESUMEN DE LA MEMORIA PARA OPTAR
AL TÍTULO DE: Ingeniero Civil Eléctrico
POR: Daniel Bravo Liberona
FECHA: 10/12/2021
PROF. GUÍA: Ricardo Finger Camus

DESIGN, IMPLEMENTATION AND TESTING OF A REAL-TIME ELECTROMAGNETIC INTERFERENCE DETECTOR AND CLASSIFIER FOR THE FIVE-HUNDRED-METER APERTURE SPHERICAL TELESCOPE (FAST)

One source of astronomical interest to study are Fast Radio Bursts (FRBs), high-power emissions produced in astrophysical processes of an unknown nature. Their power densities in the frequency domain look like Gaussians moving at lower frequencies on millisecond (ms) time scales.

With technological development, the amount of devices that radiate radio signals increases, which is received by a radio telescope as Radio Frequency Interference (RFI). Since the power of these signals are several orders of magnitude greater than those coming from astronomical sources, the effect of RFI is a problem for which detection and mitigation techniques are necessary.

This work presents a review of the detection and mitigation techniques for the design, implementation and testing of a real-time RFI detector in the detection of FRBs. The detection score is mathematically derived to be used with a threshold dependent on the type of RFI present.

The detector is tested in conjunction with a FRB detector, where from a total of 1801 FRB detections, 1791 were identified as coming from RFI, eliminating 99.4448 % of the data corresponding to interference.

The characterization of the received spectra is also studied, generating new characteristics using the statistical moments, the maximum and the minimum of the spectra. This is done in order to cluster the data to identify concentrations in space that depend on the type of interference, such as broadband or narrowband. No relationship is found between the classification of the signals and the chosen features, so the exploration of new features that better characterize the spectra is proposed as future work.

RESUMEN DE LA MEMORIA PARA OPTAR
AL TÍTULO DE: Ingeniero Civil Eléctrico
POR: Daniel Bravo Liberona
FECHA: 10/12/2021
PROF. GUÍA: Ricardo Finger Camus

DISEÑO, IMPLEMENTACIÓN Y PRUEBAS DE UN DETECTOR Y CLASIFICADOR DE INTERFERENCIA ELECTROMAGNÉTICA EN TIEMPO REAL PARA EL TELESCOPIO ESFÉRICO DE QUINIENTOS METROS DE APERTURA (FAST)

Una fuente de interés astronómico son las ráfagas rápidas de radio (FRBs, del inglés Fast Radio Bursts), emisiones de gran potencia producidas en procesos astrofísicos de naturaleza desconocida. Al ver sus densidades de potencia en el dominio de la frecuencia, se ven como gaussianas que se desplazan a frecuencias menores en escalas de tiempo de milisegundos (ms).

Con el desarrollo tecnológico, aumenta la cantidad de dispositivos que emiten señales de radio, las que son recibidas por un radiotelescopio como interferencia de radiofrecuencia (RFI, del inglés Radio Frequency Interference). Dado que las potencias de estas señales son varios órdenes de magnitud mayores que las que provienen de las emisiones astronómicas, el efecto de RFI es un problema por lo que son necesarias técnicas de detección y mitigación.

Este trabajo presenta una revisión de las técnicas de detección y mitigación para el diseño, implementación y pruebas de un detector de interferencia de radiofrecuencia en tiempo real, para la reducción de falsos positivos en la detección de FRBs. Se deriva matemáticamente un puntaje de detección de interferencia para ser utilizado con un umbral dependiente del tipo de RFI presente.

El detector fue probado en conjunto con un detector de FRBs, donde de un total de 1801 detecciones de FRBs, identificó a 1791 como proveniente de RFI, eliminando un 99,4448 % de la data correspondiente a interferencia.

También se estudió la caracterización de los espectros recibidos, generando nuevas características utilizando los momentos estadísticos, el máximo y el mínimo de los espectros. Esto se realiza con el fin de agrupar los datos para identificar concentraciones en el espacio que dependan del tipo de interferencia, como si es de banda ancha o banda angosta. No se encuentra una relación entre la clasificación de las señales y las características escogidas, por lo que se propone como trabajo futuro la exploración de nuevas características.

Acknowledgments

I want to thank my teacher Ricardo Finger for allowing me to work on this work with such exciting and challenging topics, especially the fact that he showed me the wonderful things that can be done as an electrical engineer, when I had not decided my professional career yet.

I want to thank my parents and sisters for supporting me, especially in the last moments of this work.

I also want to thank my friend Aaron Armijo who accompanied me during the last years of the career. I can't imagine the last years without him and I keep a lot of good memories studying together to be able to achieve our goals.

Finally, I want to especially thank Sebastian Jorquera for his great support in the development of this work, he was always willing to help me by accompanying me to take measurements, and explaining to me when I did not know something.

Contents

List of Figures

Chapter 1

Introduction

An astronomical source will emit radiation in a certain range of frequencies, given by the nature of the source's astrophysical processes. In particular, fast radio bursts (FRBs) are very powerful radio emissions of unknown origin and are therefore objects of astronomical interest. Furthermore, as the atmosphere in this frequency range is transparent, the FRBs pass through it, allowing their reception at the Earth's surface using radio telescopes. This atmosphere property led to the development of wireless communication with devices that broadcast on the radio. When measuring the source of interest with a radio telescope, it will also receive other radio emissions coming through the antenna in the form of Radio Frequency Interference (RFI).

The number of devices that emit radio signals has increased over time, thus producing an increase in the amount of radio-frequency interference (RFI) signals that, when produced on the earth's surface, have a power of up to eleven orders of magnitude higher than the weak signals of interest. From this arises the need for methods that are capable of detecting RFI signals to carry out some action. Usually the detection is used to mitigate the effects that RFI has on the observation.

1.1. Thesis Scope

This work focuses on the implementation of a real-time electromagnetic interference detector, modeling it to be compiled and loaded into a field-programmable-gate array (FPGA) that will execute the model. A script must also be made to communicate with the FPGA to read the important data of the model, and perform tests, emulating RFI signals with laboratory equipment and performing tests with antennas in astronomical measurements.

A classification method will be explored, generating characteristics in the antenna test dataset, and then reducing its dimensionality by having the features that have more information about the dataset.

1.2. Objectives

1.2.1. General Objective

Design, implement, and test a RFI detector and classifier, that uses the signals from the primary (sky) and a reference antenna, for the Five-hundred-meter Aperture Spherical Telescope (FAST). The detector will be implemented in a field-programmable-gate array (FPGA). Due to its high throughput rate, a necessary condition given that the detector has to work in real time.

1.2.2. Specific Objectives

- Study the RFI detection and mitigation methods used in astronomy, understanding their advantages, limitations and effectiveness depending on the type of interference.
- Study the polyphasic filter bank theory for the realization of a spectrometer.
- Design and implement the detector in an FPGA.
- Test the designed detector in a controlled laboratory environment and in a real case of astronomical measurements.

1.3. Structure

In chapter one, the document studies RFI detection and mitigation methodologies, as well as feature generation and dimensionality reduction methods to classify it using clusters in chapter 2. Then the work environment is presented in chapter 3 since it is necessary to work with an FPGA. In chapter 4 the methodologies used to achieve detection and classification are presented, showing the implementation of the model in an FPGA, chapter 5 shows the limitations of the detector. Tests are carried out and their results are presented and analyzed in Chapter 6. Chapter 7 presents the main conclusions and in Chapter 8 with suggested future work.

Chapter 2

Theoretical Background

2.1. Detection techniques

2.1.1. Time-domain

These methods try to detect RFI sources in a time series, or a stream of samples spaced in time. For instance consider a pulsed radar, the detectors used in this type of RFI are called pulse detectors, which compare the power of the signal received with a threshold, as shown in figure 2.1. Although this method is theoretically simple to understand, in practice is important that the threshold considers the changes in the system temperature, adding additional complexity to the implementation process. For example in [1] a time-domain threshold was demonstrated by Fridman et al. in 1996.

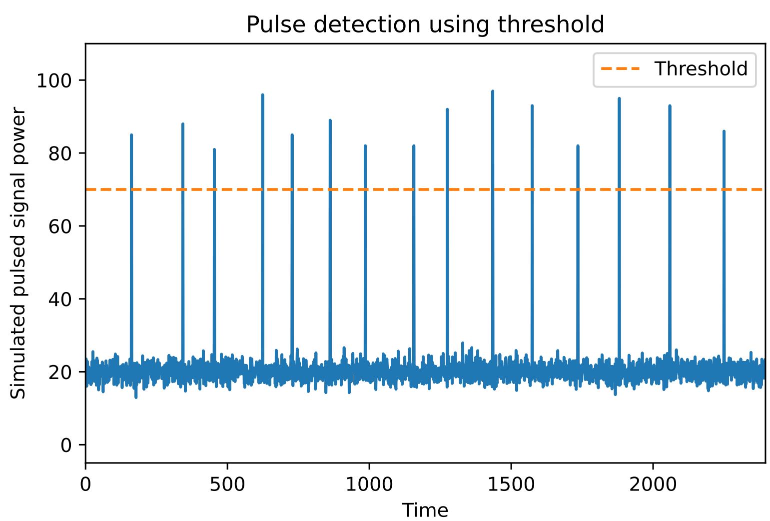


Figure 2.1: Example of pulse detection using a threshold.

2.1.2. Polarimetry

Typically thermal noise produced by a natural source is weakly or not polarized, on the other hand RFI are commonly linearly or circularly polarized, so measuring the Stokes parameters, which are a set of four values that relate with the polarity of the signal, leads a method to detect RFI

2.1.3. Gaussianity tests

Thermal emission by a natural source as well as thermal noise have a Gaussian distribution as opposed to RFI which has a non-Gaussian behaviour, so a difference with this statical function indicates the presence of RFI. Even though numerous methods exist to test Gaussianity in real time, the kurtosis detection algorithm is the most widely used, in which a deviation of the Kurtosis value $\kappa = 3$ according to

$$\kappa = \frac{N \cdot \sum_N (x - \mu_x)^4}{\left(\sum_N (x - \mu_x)^2\right)^2}, \tag{2.1}$$

suggest a non-Gaussian component, where N is the number of data samples, x is the value of the data sample and μ_x is the mean value of the signal. The kurtosis method works with a wide variety of RFI types, but for pulsed sinusoidal interference the detection is poor, in which case it's possible to improve detection performance by subsampling in time and frequency. In [2] the author studied the Shapiro-Wilk test of Gaussianity as an alternative detection method.

Another example of this method is to measure the spectral kurtosis which tests the Gaussianity in the frequency domain. The receiver bandwidth is divided into sub-channels by means of the Fast Fourier Transform (FFT) to calculate de kurtosis of each sub-channel separately. If the data is divided into M sub-vectors $x_i(k)$ of length N_{SK}, the spectral kurtosis is

$$\kappa_s(m) = \frac{M}{M-1}\left[\frac{(M+1) \cdot \sum_{i=1}^{M} |X_i(m)|^4}{(\sum_{i=1}^{M} |X_i(m)|^2)^2} - 2\right], \tag{2.2}$$

where $X_i(m)$ is the Discrete Fourier Transform (DFT) of $x_i(k)$ and m is the frequency bin of each sub-channel. For both kurtosis methods the detection is independent of the level and changes of brightness temperature scene. On the other hand it is necessary to accumulate data in vectors of size N or N_{SK} in order to determines the kurtosis value decreasing the time resolution. However the spectral kurtosis method has a better spectral resolution, also it's important to consider that the effectiveness of the detection depends on the type of RFI, the number of data used to calculate the spectral kurtosis of a channel is N_{SK} times smaller than the kurtosis method, implying that the first one has a lower sensitivity. However, the fact that the bandwidth of the sub-channels of the spectral kurtosis method is M times smaller than the other method, indicates that the interference-to-noise ratio (INR) is better, it's important to mention that an RFI spreading over two or more sub-channels will reduce the INR so may need some signal processing to keep same performance, so an increase of the number of bins will improve the detection probability to a maximum [3].

2.1.4. Spectral Density Estimation

The Spectral Density Estimation (SDE) has many different implementations, one of them is Barlett method which isn't the best method in performance but is one of the simple SDE algorithms and if it's implemented in a system that already has a kurtosis detection method, the Barlett method first calculations steps are the same than the kurtosis algorithm, reducing the costs associated with the hardware involved in these process.

The Barlett method determines the SDE dividing the data into M sub-vectors $X_i(k)$ of size N_{SDE}, then a DFT is applied to each vector, calculating its squared magnitude averaged over i, and divided by πN with $i \in \{1, 2, ..., M\}$.

2.1.5. Cyclostationary RFI

An interference signal is called cyclostationary if its autocorrelation function is constant over time with periodicity T, Rodolphe Weber [4] et al. demonstrate a real time mitigation method based on the detection of this kind of RFI signals, the hardware implementation was made on a field-programmable-gate array (FPGA) due their speed, high data throughput and reconfigurability. The detector only works with interference signals with known T values, although it can be improved to deal with unknown periodicities cyclostationary interference signals.

2.2. Mitigation Techniques

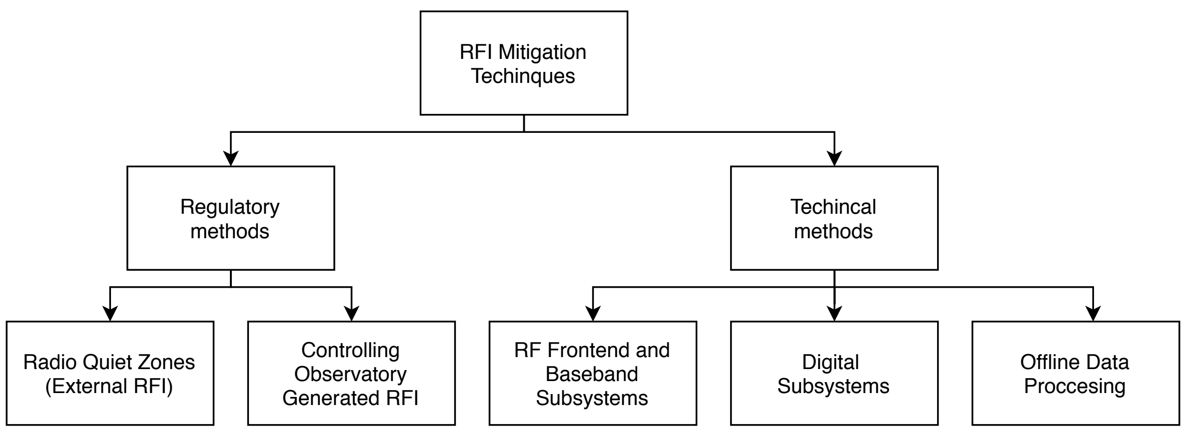


Figure 2.2: Basic scheme of mitigation techniques [5].

2.2.1. Regulatory methods

These methods act directly on the RFI sources, taking actions over the interference generated on the observatory devices or establishing quiet zones around the telescope with different rules depending on the distance to the source.

2.2.1.1. Radio Quiet Zones

Radio Quiet Zones (RQZ) are areas in which radio transmission are heavily restricted in order to facilitate scientific research. For instance the RQZ established surrounding FAST has a surface of 2827 km^2 and a radius R of 30 km, this area is divided into three subsections; the restriction zone is the central subsection with $R < 5$ km, the central area is where 5 km $< R <$ 10 km and the remote area is defined for 10 km $< R <$ 30 km, these last two regions are called as coordination zone. The figure 2.3 shows the restriction zone established for FAST telescope.

On the restriction zone any kind of external transmission is forbidden, meanwhile in the coordination zone operators can transmit as long as they coordinate with the FAST telescope operators, unless the emissions exceed 100 W of power in the same frequency bands that FAST operates, in which case any transmission is prohibited. [6].

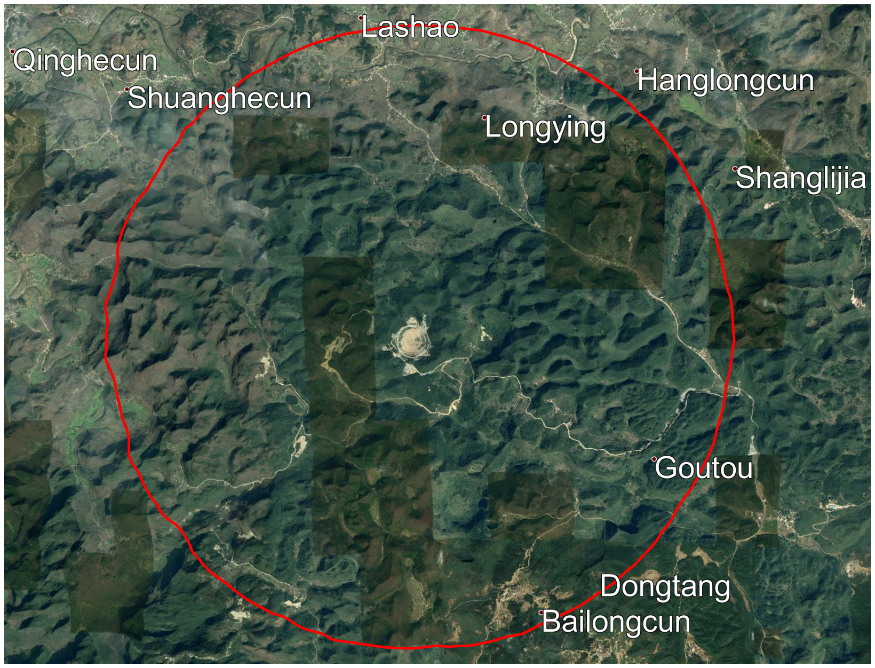


Figure 2.3: Restriction zone of the RQZ established around FAST telescope, the circumference has a radius of 5 km.

2.2.1.2. Controlling Observatory Generated RFI

It is important to control RFI emissions that may occur in facilities, so shielding should be considered in certain areas or equipment. Common devices such as computers, high-power devices, network devices have broadband and characteristic spectral line emissions, so it is also necessary to shield these equipments.

2.2.2. Technical methods

These methods deal with RFI that is already present in the environment. There are a large number of mitigation techniques, which can be classified into three categories.

2.2.2.1. RF Frontend and Baseband Subsystems

The RFI is mitigated at the beginning of the reception chain so that it does not pass to the backend of the system, one way to do this is to design antennas with a highly directive radiation pattern, in practice there is a main lobe that points to the astronomical object and unwanted side lobes by which other emissions are received. A radiation pattern with these conditions is shown in the figure 2.4.

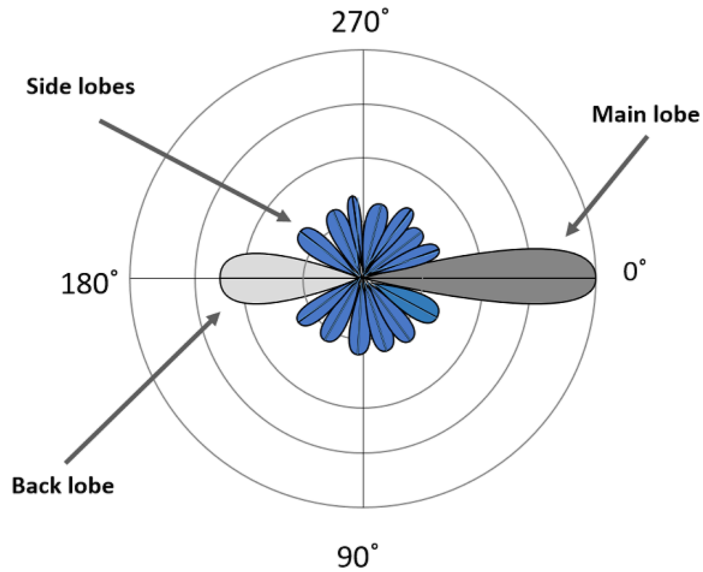


Figure 2.4: Radiation pattern of a directional antenna.

If the emission spectrum of the astronomical object is known, other frequencies that are outside the range of interest can be discarded through frequency filters such as band-pass or notch filters.

2.2.2.2. Digital Subsystem

Digital subsystem techniques take place after signal digitization and before permanent storage, so digital processing must be fast enough to work in real time. To meet this condition FPGAs are used, which are devices with logic blocks whose interconnections are determined by a description language.

The spectral kurtosis is used as an indicator because, as in the case of the time domain, most of the astronomical signals have a Gaussian behaviour while most of the RFI have a non-Gaussian one. In [7] an estimator based on spectral kurtosis is implemented in a FPGA, considering as detection when the estimator exceeds a certain threshold. These detection methods, in which an estimator is compared with a certain value, are known as thresholding techniques.

In addition to mentioned techniques there is another group based on the subtraction of two signals, the signal from the main antenna having information of the object of study and the RFI signal and a reference antenna only it has information of interference. This subtraction cannot be calculated directly, to do this, adaptive filters are commonly used, as shown in figure 2.5, the RFI signals that enter through both antennas are correlated as they are the product of the same sources. The adaptive filter determines this correlation, updating a set of values in which the components that correlate survive, that is the RFI. By subtracting this output of the adaptive filter from the signal of the main antenna, the interference will be mitigated. [8].

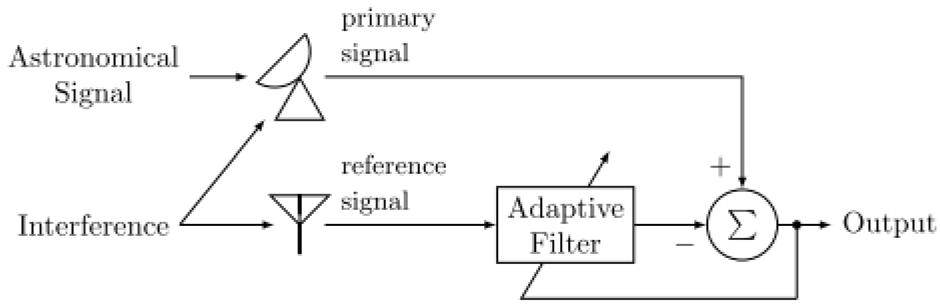


Figure 2.5: Adaptive filter cancelling the interference signal. Extracted from http://www.das.uchile.cl/lab_mwl/publicaciones/Tesis/tesis_franco_curotto.pdf

2.2.2.3. Offline Data Processing

As the data that these techniques use has already been stored, the processing time is not a limitation as in the case of digital subsystems in real time. Machine learning techniques can be used or thresholding techniques using different statistics.

2.3. Cross-correlation

Digital Signal Processing (DSP) uses mathematical tools on discrete signals to achieve some purpose, as in this case, the implementation of a detector. In particular, the cross-correlation between two discrete signals $x[n]$ and $y[n]$ measure the similarity between them when they are displaced from each other by k, it's defined as

$$(x[n] \star y[n])\,[m] \triangleq \sum_{n=-\infty}^{\infty} x^*[n-m] \cdot y[n], \tag{2.3}$$

Autocorrelation measures the similarity between a signal and a delayed version of it with a delay m, this function reaches its maximum when $m = 0$ comparing the signal with itself. It can be expressed as

$$r_{xx}[m] = \sum_{n=-\infty}^{\infty} x^*[n-m] \cdot x[n] \tag{2.4}$$

The Discrete Fourier Transform (DFT) is a discrete transform that converts a temporary discrete sequence $x[n]$ into an equivalent representation in the frequency domain $X[k]$. The DFT is an approximation of the continuous-time Fourier Transform and is given by

$$X[k] = \sum_{n=0}^{N-1} x[n] \cdot e^{-\frac{j2\pi}{N}kn} \tag{2.5}$$

2.4. Spectrometers

A spectrometer is a device that measures the power spectral density (PSD) of a digitized signal, this density can be computed mainly in two ways, taking the autocorrelation of the signal and calculating the DFT using equation 2.4 in 2.6, or converting the signal to the frequency domain and then computing the autocorrelation (which is different from the temporal case), this last method will be studied in depth below.

The Wiener–Khinchin theorem states that the power spectral density ($S[k]$) is related with the autocorrelation by

$$S[k] = \sum_{k=-\infty}^{\infty} r_{xx}[m]e^{\frac{-j2\pi}{N}kn}, \tag{2.6}$$

which can be interpreted as the DFT of the autocorrelation of the signal, which is equivalent to

$$S[k] = X^*[k] \cdot X[k] = \|X[k]\|^2, \tag{2.7}$$

where $X^*[k]$ denotes the conjugate of the DFT of $x[n]$, the derivation of this result is developed in Annex A.1, a diagram of these two methods is shown in the figure 2.6.

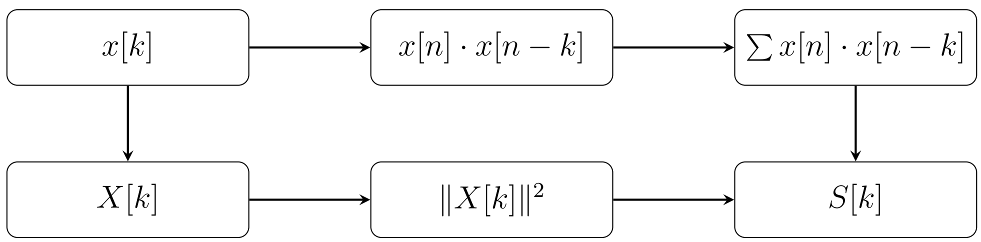


Figure 2.6: Two methods to calculate the PSD of a digitized signal $x[k]$ [9].

The DFTs, depending on the sampling frequency and the number of channels, can generate spectral leakage, this occurs when the energy corresponding to a certain frequency f_0, spreads in the spectral channels close to this frequency, because the DFT assumes that the signal is periodic, and the time series to transform is equivalent to one period of it. To reduce the impact of this phenomenon, Polyphase Filterbanks (PFBs) are used before calculating the DFTs , mitigating the leakage in nearby spectral channels as shown in figure 2.7. Due to their effect on DFT leakage, PFBs are commonly used in the implementation of spectrometers and correlators.

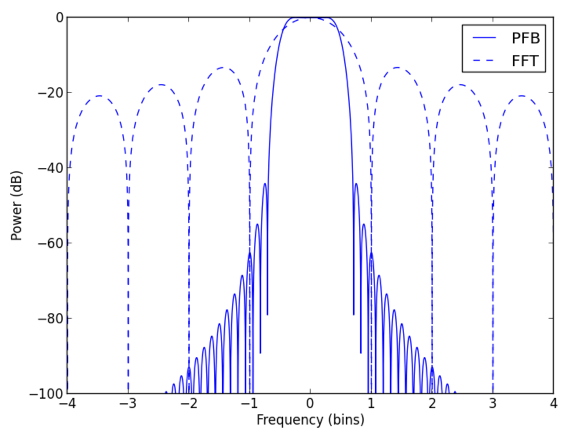


Figure 2.7: Comparison of DFT leakage, when using PFBs before FFT.

2.5. Feature generation

The generation of characteristics is the process of adding new characteristics to a data set, this can be done for example through the calculation of various statistics adding new information. The objective of generating characteristics is to find some that have important information to describe the process that generates it, but since there are also dimensionality reduction methods, characteristics can be added without knowing their impact on the model, to later be reduced by algorithms eliminating the characteristics that do not contribute.

In [10] an extension to a commonly used power spectrum parameterization is proposed, which consists of the truncation of the Taylor series defined by $\ln \mathrm{PSD}(k) = \ln \mathrm{PSD}_* + (n_* - 1)\ln(k/k_*) + \frac{1}{2}n_*' \ln^2(k/k_*)$, this assumes that the values from the fourth term of the series are negligible. This method is used when $\left|n_*' \ln(k/k_*)\right| \ll |n_* - 1|$, but it is shown that for current observations, the method also works when $\left|n_*' \ln(k/k_*)\right| \sim |n_* - 1|$.

Another approach is used in [11], where an RFI mitigation method is proposed from the instantaneous spectra and the probability distributions that dominate them, measuring the Gaussianity to define a detection. The first four statistical moments corresponding to the mean, variance, skewness and kurtosis are calculated, the last three correspond to measures of how the data are spread, a measure of the lack of symmetry and a measure of Gaussianity, being the kurtosis of a gaussian distribution equivalent to three times the squared variance. Also in [12] a way of classifying broadband and narrowband signals is presented.

Broadband signals are defined as signals with a bandwidth greater than the bandwidth of an ADC channel determined by the Nyquist Sampling Theorem, on the contrary, if the signal is totally contained in its bandwidth, it is classified as narrowband. Thus measurements of the average and the maximum of a spectrum have a relationship between broadband and narrowband classification.

2.6. Dimensionality reduction

Two dimensionality reduction methods are presented, which transform a dataset to a smaller one, containing the n characteristics that best characterize the data according to their own methodologies.

2.6.1. PCA

Principal Component Analysis (PCA) is a dimensionality reduction method eliminating features that do not provide information. This method calculates the normalized mean of the dataset, calculates its Covariance Matrix and the Eigen Vector and Eigen Value Matrix. The new characteristics are a linear combination of the previous ones, having the components that provide the most information arranged according to the Eigen Values in decreasing order.

2.6.2. t-SNE

T-SNE is a technique to visualize datasets in 2D and 3D plots converting Euclidean distances between dataset points into conditional probabilities that represent them, to make possible data structures visible, the probability p of x_j given x_i is

$$p_{j|i} = \frac{e^{-\|x_i - x_j\|^2/2\sigma_i^2}}{\sum_{k\neq i} e^{-\|x_i - x_k\|^2/2\sigma_i^2}}, \tag{2.8}$$

where σ_i is the variance of a Gaussian centered on x_i. In the same way for a low-dimensional counterpart y_i and y_j, $q_{j|i}$ is defined as

$$q_{j|i} = \frac{e^{-\|y_i - y_j\|^2/2\sigma_i^2}}{\sum_{k\neq i} e^{-\|y_i - y_k\|^2/2\sigma_i^2}}. \tag{2.9}$$

This method works by minimizing the cost function

$$C = \sum_i KL(P_i||Q_i) = \sum_i \sum_j p_{j|i} \log \frac{p_{j|i}}{q_{j|i}}, \tag{2.10}$$

where P_i is the conditional probability distribution for all dataset points with respect to x_i, Q_i represents the same distribution for y_i and KL is the Kullback-Leibler divergence [13].

2.7. Clustering

Clustering methods are automated algorithms that find concentration structures in the dataset, this section shows the K-Means method to introduce this type of techniques.

2.7.1. K-Means

K-means is an unsupervised clustering algorithm that, given a dataset, assigns K random centroids where K corresponds to the number of clusters to be formed, these centroids are updated in the algorithm. Considering the Euclidean distance between two samples of the dataset $x = (x_1, x_2, \cdots, x_n)^T$ and $y = (y_1, y_2, \cdots, y_n)^T$ defined as

$$\|x - y\| = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}, \tag{2.11}$$

K centroids are randomly assigned, the euclidean distance to the nearest centroid is calculated for each sample of data and is associated with it. After this, the average of the data associated with each cluster is calculated to update the centroid value to this mean, it is iterated until the value of the centroids stops changing.

Chapter 3

Work environment

The detector design must be implemented in a ROACH-2-rev2 board, an FPGA developed by CASPER (Collaboration for Astronomy Signal Processing and Electronics Research), who also developed a library with useful blocks for processing astronomical signals for MATLAB. The models are implemented in Simulink and compiled using Xilinx ISE Design Suite.

A virtual python 2.7 environment is needed in order to process data and establish communication with the FPGA. The following sections will delve into the hardware, compilation and programming tools used for the development of this work.

3.1. ROACH-2

A field-programmable gate array (FPGA) is a semiconductor device which has a large number of Configurable Logic Blocks (CLBs), which can do basic digital operations through small components such as flip-flops, look-up tables (LUTs) and multiplexers. These blocks are surrounded by routing channels, which can be thought of as cables that carry the digital signals between blocks, the interconnection of these routes is done through the configuration of the interconnection switches. The Input/Output Block (I/O Block) is to interface with the board, the figure 3.1 shows a basic diagram of a FPGA with these components. More advanced FPGAs implement more complex blocks such as RAM blocks, multipliers, and DSP blocks to simplify the design and reduce the use of resources.

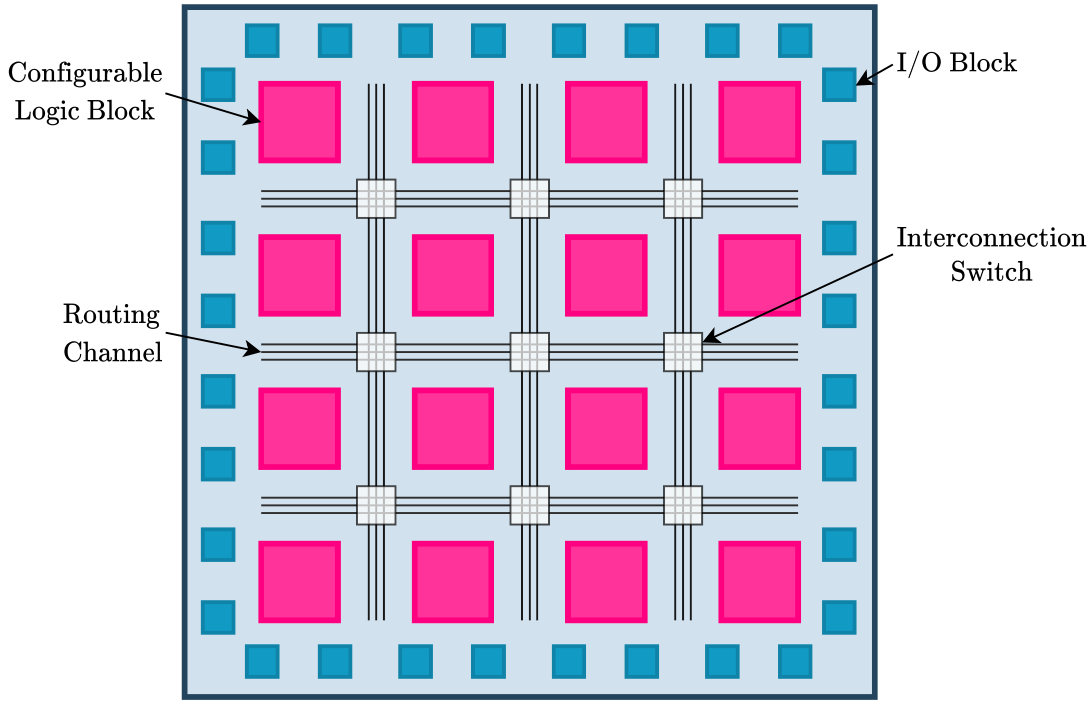


Figure 3.1: FPGA basic diagram [8].

ROACH-2 (Reconfigurable Open Architecture Computing Hardware) rev 2 is the successor of ROACH board, its main component is the Virtex-6 XC6VSX475T FPGA, which has 2016 DSP48E1 blocks and 74400 CLBs. Some functions that can be performed on the DSP48E1 are multiply, three-input add, barrel shift, magnitude comparator, bit-wise logic functions and pattern detect [14], a slice of this block is shown in figure 3.2.

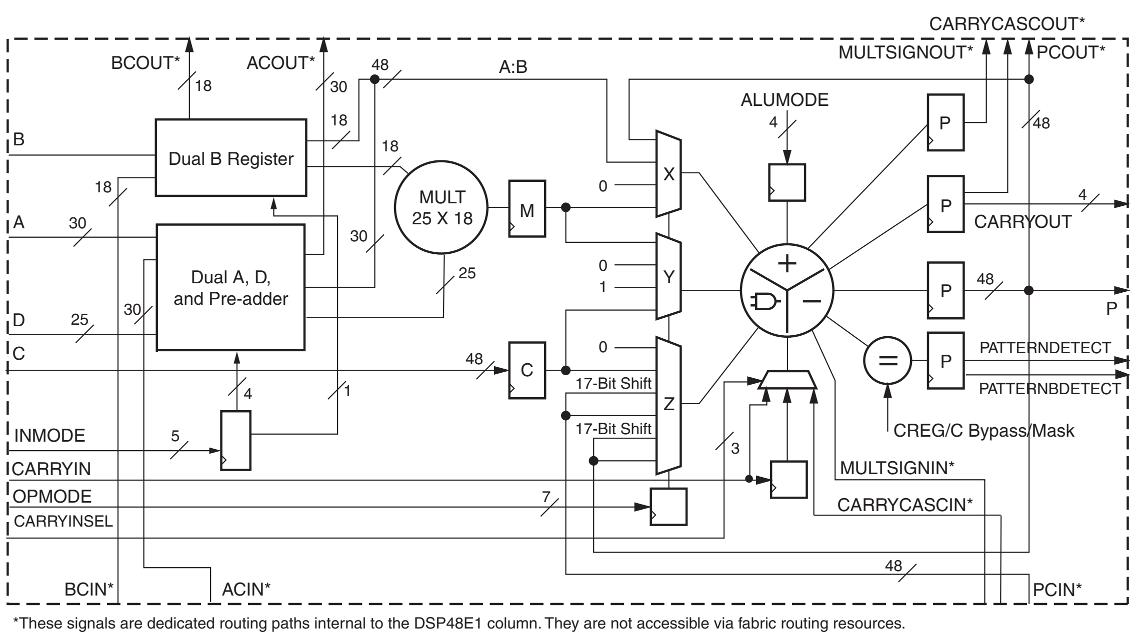


Figure 3.2: Virtex-6 FPGA DSP48E1 Slice. Extracted from https://www.xilinx.com/support/documentation/user_guides/ug369.pdf

Other important components of the ROACH-2 dev 2 are the PowerPC 440EPx stand-alone processor that runs Linux to provide control functions, which allow to reconfigure and interface the FPGA with other devices through Ethernet. It also has two ZDOKs docking connectors for the connection of two Analog-Digital Converters (ADCs), particularly the ADC boards used are the ADC1x5000-8, which can operate as one-channel mode with a resolution of 8 bits and 5 GSPS for one input or in a two-channel mode with the same resolution but 2.5 GSPS for two inputs. Also a Valon 5007 synthesizer provides clock reference for both ADCs. Figure 3.3 shows the ROACH-2 dev 2 block diagram and the figure 3.4 shows a photo of the board with these devices and a Power Supply Unit (PSU) that powers the system.

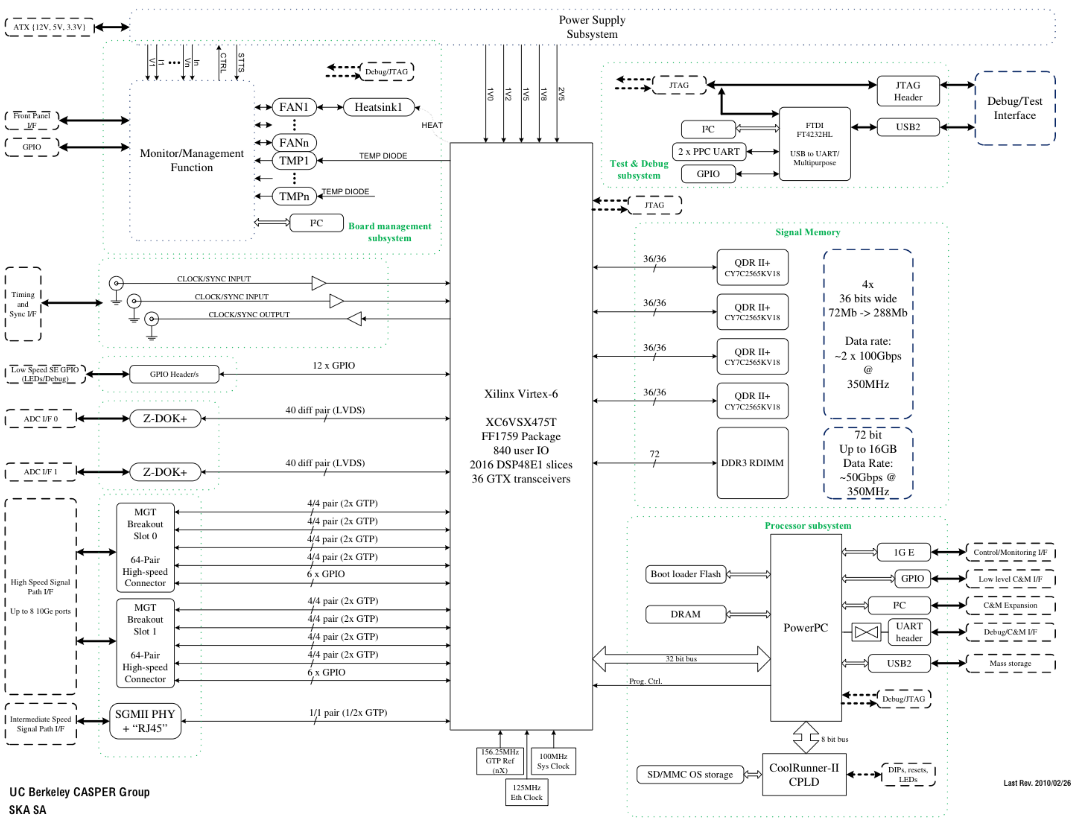


Figure 3.3: ROACH-2 rev 2 block diagram. Exctracted from https://casper.ssl.berkeley.edu/wiki/ROACH-2_Revision_2

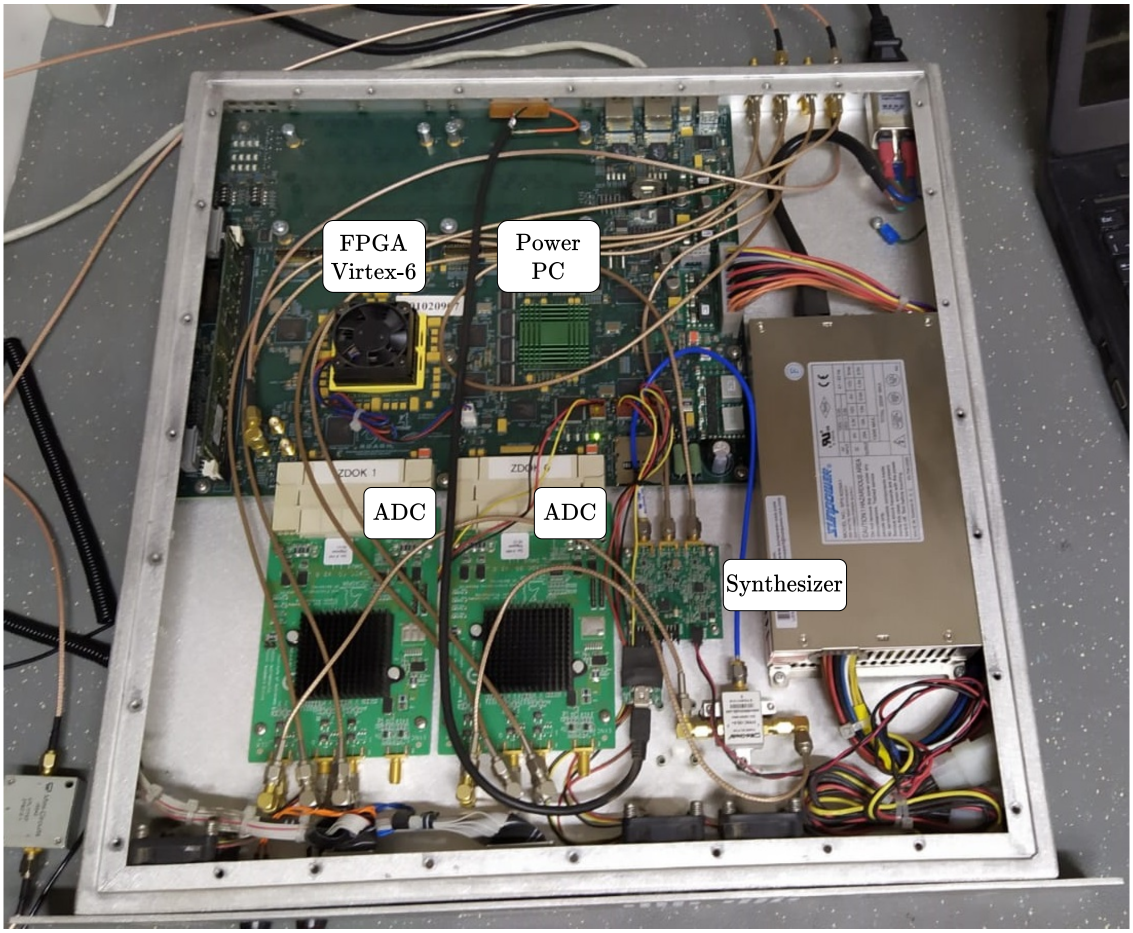


Figure 3.4: ROACH-2 dev 2 board, using two ADCs and a synthesizer to provide clock for both ADCs

3.2. Hardware description

To describe the circuit that will be implemented in the FPGA, Hardware Description Language (HDL) is used, which is a specialized programming language to describe the behaviour or structure of the circuit, a graphic design software can also be used. In both cases, the methodology is designed to be interpreted by humans, so a compilation tool is necessary that generates a bitstream file, this file is interpretable by the FPGA and contains the configurations of the resources of the board, to implement the circuit.

Xilinx, the developer of the Virtex FPGAs used in the ROACH-2 board, also provides tools for the design and compilation of a circuit, through its ISE Design Suite software. For the design, Xilinx makes available a library of basic blocks for Simulink, a graphical programming environment present in MATLAB. CASPER used these blocks to make its own library with more advanced capabilities, such as complex math operations, accumulators, FFT blocks and PFB blocks.

3.3. Compiler

The ISE Design Suite allows the generation of the bitstream file for the FPGA, working in parallel with Simulink, from where it extracts the model of the circuit to compile.

3.4. Communication

In order to read the data of interest from the model and change its parameters through registers, it is necessary to generate a script to communicate with the ROACH-2 board. Casper and the digital group of the Milimeter-Wave Laboratory (MWL) of Universidad de Chile made the corr and calandigital libraries respectively, which facilitate communication and data analysis. Since these libraries are made in the Python 2.7 programming language, it is the language used for the detector and classifier script.

Chapter 4

Methodology

4.1. Formalization of the problem

FRB detections made by FAST and other telescopes have a significant number of false positives due to RFI, leading to increased use of hardware for storage and processing, so it is necessary to implement a real-time RFI detector to avoid these negative effects. For its realization, a main antenna that receives an astronomical signal contaminated with RFI and a reference antenna that only receives RFI must be considered, in addition to the need to work in real time, it must be implemented in a FPGA. It is also important to characterize the present RFI for future considerations, so an RFI classifier must be designed without the need for it to work in real time.

Bibliography related to hardware limitations, RFI detection, mitigation and classification methods is studied to define detector and classifier methodologies. Once these are determined, the work environment is prepared and the knowledge to use it has been acquired, the detector is modeled and compiled to configure the resources of a FPGA. Laboratory tests with signal generators and antennas under real conditions are performed to verify and analyze the detector's operation. Then the classifier is implemented and tested with measurements previously stored in the detection, using the detector's output. A flow chart summarizing the work carried out is shown in figure 4.1.

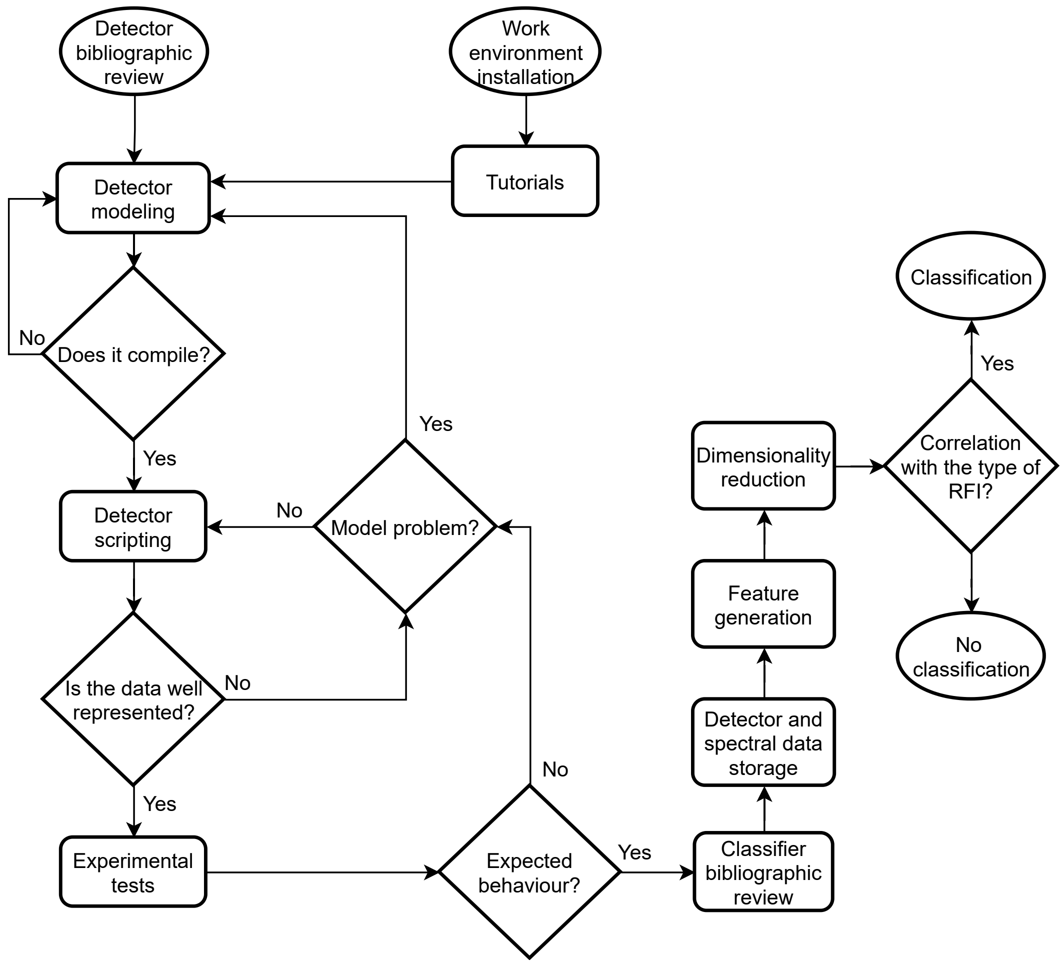


Figure 4.1: General flow chart.

4.2. Tutorials

To get acquainted with the tools used to model, compile, and communicate with the FPGA, calandigital and CASPER provide tutorials to teach how to use blocks such as FFT, PFB, accumulators, and Block Random Access Memory (BRAM), in addition to showing how to compile a model and communicate with the board to upload the bitstream file, configure and establish connection for data transfer. The main activities were the implementation of a snapshot to see a plot of the ADC output, and of a spectrometer plotting the PSD.

4.2.1. Snapshot

This activity[1] shows how to read the ADC output data stored in BRAMs in the ROACH-2 trough Ethernet on a computer, and plotting it in real time using python. The model is shown in figure 4.2, as the ADC sample in rising and falling edge of the clock because it is in one-channel mode and as it has 16 parallel outputs, the frequency value to be set in the Valon 5007 must be 8 times less than the desired sample rate. Figure 4.3 shows a plot of the ADC data, which is stored in a BRAM to be read by the computer.

[1] Available on https://sites.google.com/site/calandigital/tutorials/snapshot-tutorial.

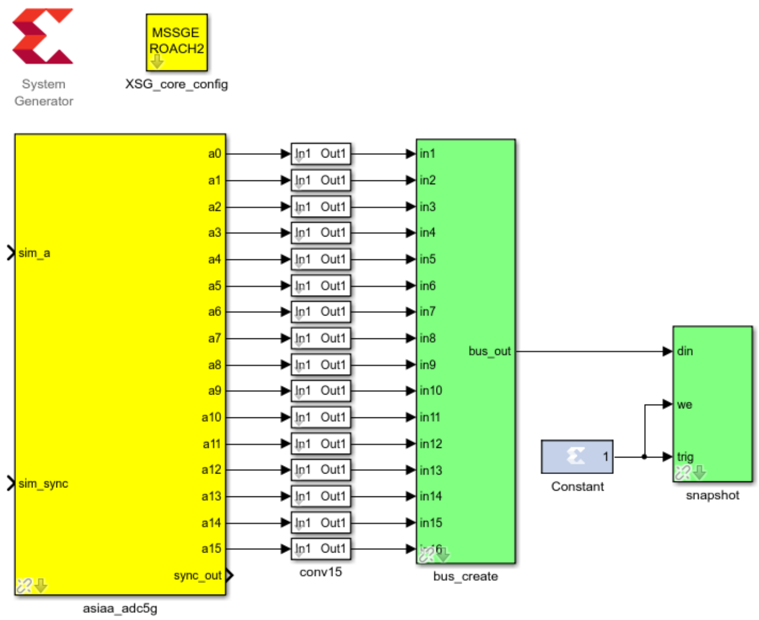


Figure 4.2: Snapshot model in Simulink.

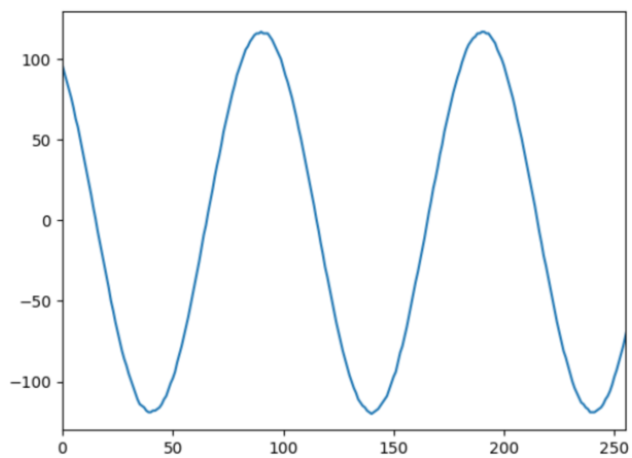


Figure 4.3: 10 Mhz sinusoidal signal sampled at 2GSPS. Extracted from https://sites.google.com/site/calandigital/tutorials/snapshot-tutorial.

4.2.2. Spectrometer

A spectrometer[2] is modeled in Simulink, which has registers that can be read and overwritten through a script, to reset the circuit or change the amount of accumulations. The model calculates the FFT of the ADC output using PFBs, then the power of each channel is calculated to be accumulated, once this process ends, its value is saved in BRAMs and another accumulation cycle begins, which will overwrite the BRAMs values of the respective addresses. Figure 4.4 shows the model implemented in Simulink with sixateen parallel outputs.

[2] Avilable on https://casper.ssl.berkeley.edu/wiki/Wideband_Spectrometer.

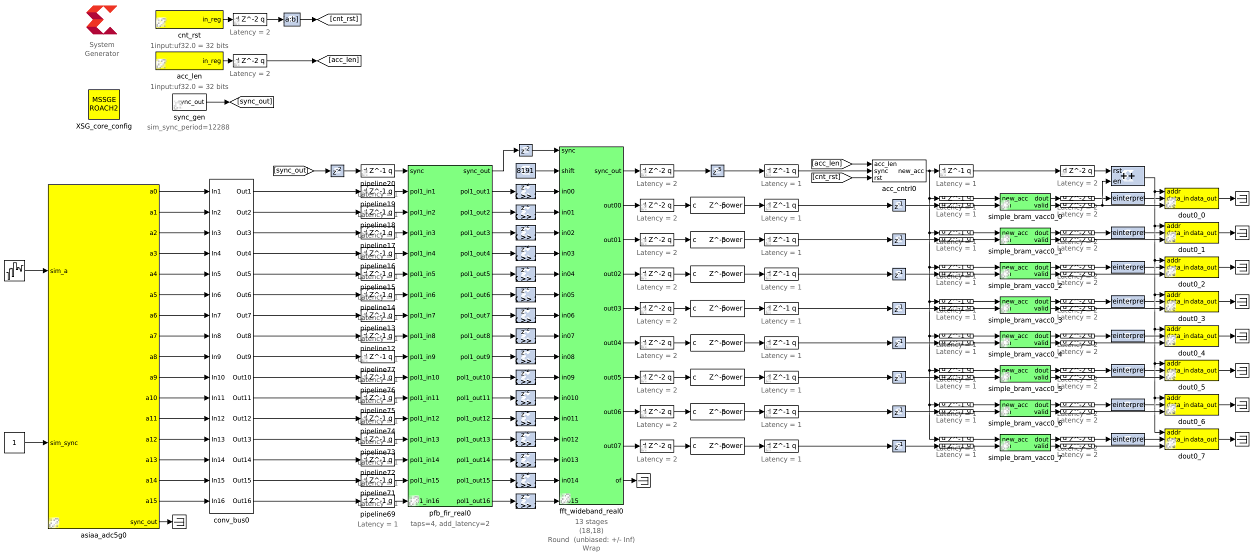


Figure 4.4: Spectrometer model in Simulink.

4.3. Detector

4.3.1. Design

Figure 4.5 shows the general methodology to be used for detection, by quantitatively measuring the similarity between the primary signal $a(t) + i(t)$ and the reference signal $r(t)$, where $r(t)$ it only measures RFI because its radiation pattern points toward the horizon. If both signals are correlated it is because the antennas are measuring mainly RFI, on the contrary, if the value is low it means that there is an astronomical component and/or there is no RFI presence. When using the normalized correlation as a measure of similarity, the score $\mu(f)$, or coherence in [15] for the analog case can be written as

$$\mu = \frac{\|\mathrm{CPSD}\|^2}{\mathrm{PSD_{main}} \cdot \mathrm{PSD_{ref}}}, \tag{4.1}$$

where CPSD is the cross power spectral density between the main and reference signals.

Considering the implementation diagram of figure 4.6, where N samples were accumulated to reduce the variance and eliminate the uncorrelated components, more details in the section 4.3.2. The score is calculated for each bin of the FFT, let $X_{j,k}$ and $Y_{j,k}$ with $j \in [0, N-1]$ and accumulation N be the k -th outputs of the FFTs corresponding to the main and reference signal respectively, (4.1) becomes

$$\mu_k = \frac{\left\| \sum_{j=0}^{N-1} X_{j,k} \cdot Y_{j,k}^* \right\|^2}{\sum_{j=0}^{N-1} \|X_{j,k}\|^2 \cdot \sum_{j=0}^{N-1} \|Y_{j,k}\|^2}. \tag{4.2}$$

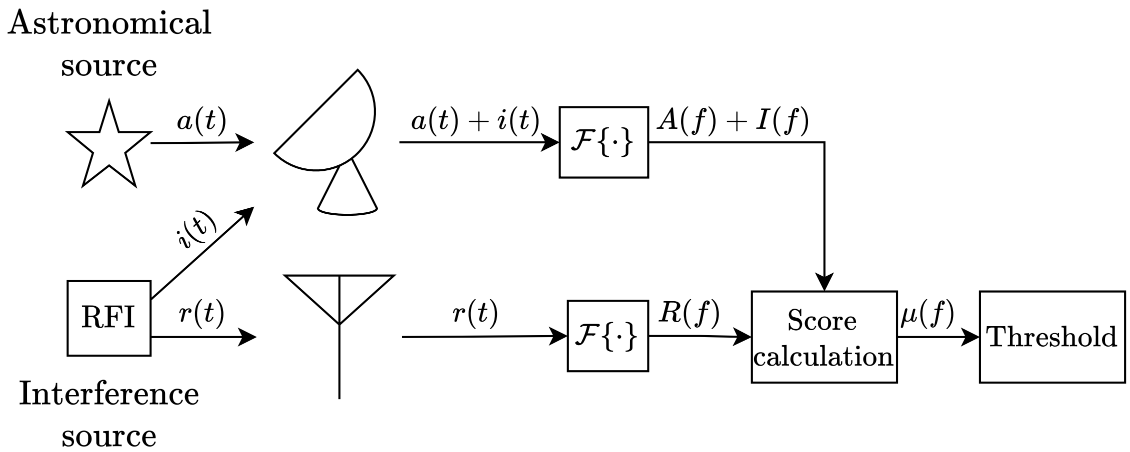


Figure 4.5: Score calculation for threshold detection.

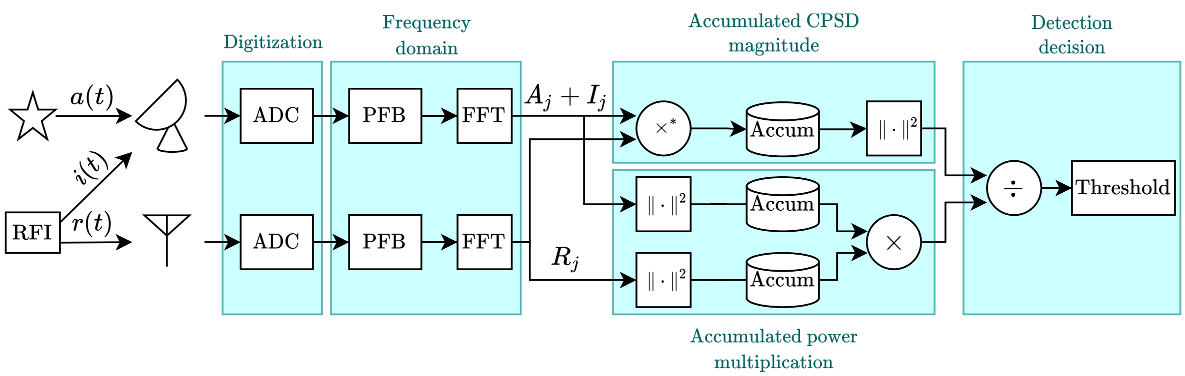


Figure 4.6: Diagram of the detector implementation. The FFTs outputs show just one spectral channel for readability. $\times^*$ represents conjugated multiplication and Accum the accumulation.

4.3.2. Power interpretation

Considering the k-th channel of the FFT of each digitized input signal as $X_{j,k} = A_{j,k} + I_{j,k}$ and $Y_{j,k} = R_{j,k}$ as shown in figure 4.6, for an accumulation N (4.2) remains as:

$$\begin{aligned}\mu_k &= \frac{\left\|\sum_{j=0}^{N-1}(A_{j,k}+I_{j,k})\cdot R_{j,k}^*\right\|^2}{\sum_{j=0}^{N-1}\|A_{j,k}+I_{j,k}\|^2\cdot\sum_{j=0}^{N-1}\|R_{j,k}\|^2}\\ &= \frac{\left\|\sum_{j=0}^{N-1}A_{j,k}\cdot R_{j,k}^* + \sum_{j=0}^{N-1}I_{j,k}\cdot R_{j,k}^*\right\|^2}{\sum_{j=0}^{N-1}\|A_{j,k}+I_{j,k}\|^2\cdot\sum_{j=0}^{N-1}\|R_{j,k}\|^2},\end{aligned} \tag{4.3}$$

$A_{j,k}$ and $R^*_{j,k}$ are not correlated, the phase of the multiplication between them is random, on the other hand, $I_{j,k}$ and $R^*_{j,k}$ are correlated, so the phase of their multiplication is constant. For a sufficiently large value N of accumulations, the contribution of the uncorrelated terms will be negligible compared to those that do correlate, then (4.3) becomes

$$\mu_k = \frac{\left\| \sum_{j=0}^{N-1} I_{j,k} \cdot R^*_{j,k} \right\|^2}{\sum_{j=0}^{N-1} \|A_{j,k} + I_{j,k}\|^2 \cdot \sum_{j=0}^{N-1} \|R_{j,k}\|^2}. \tag{4.4}$$

Since $I_{j,k}$ and $R_{j,k}$ correspond to the RFI signals measured by each antenna, they are correlated, so the complex number $I_{j,k} \cdot R^*_{j,k}$ has constant phase ϕ_0, this and the fact that $\|a \cdot b\| = \|a\| \cdot \|b\|$ allows the substitution

$$\begin{aligned} \left\| \sum_{j=0}^{N-1} I_{j,k} \cdot R^*_{j,k} \right\|^2 &= \left\| \sum_{j=0}^{N-1} \left\|I_{j,k} \cdot R^*_{j,k}\right\| \cdot e^{j\phi_0} \right\|^2 \\ &= \left\| \sum_{j=0}^{N-1} \left\|I_{j,k} \cdot R^*_{j,k}\right\| \right\|^2 \cdot \left\|e^{j\phi_0}\right\|^2 \\ &= \left(\sum_{j=0}^{N-1} \left\|I_{j,k} \cdot R^*_{j,k}\right\| \right)^2 \\ &= \left(\sum_{j=0}^{N-1} \|I_{j,k}\| \cdot \left\|R^*_{j,k}\right\| \right)^2 \end{aligned} \tag{4.5}$$

also if it is assumed that $\|R_{j,k}\|^2 = \alpha \cdot \|I_{j,k}\|^2$, where alpha depends on parameters such as frequency, spectral channel, distance between antennas, angles of incidence and antennas gain, so it is a complex value to compute, using this condition, the result obtained in (4.5) and the fact that $\|a\|^2 = a \cdot a^*$ where $*$ represents the conjugate, the equation (4.4) can be rewritten as

$$\begin{aligned} \frac{\left(\sum_{j=0}^{N-1} \|I_{j,k}\| \cdot \left\|R^*_{j,k}\right\| \right)^2}{\sum_{j=0}^{N-1} \|A_{j,k} + I_{j,k}\|^2 \cdot \sum_{j=0}^{N-1} \|R_{j,k}\|^2} &= \frac{\left(\sum_{j=0}^{N-1} \|I_{j,k}\| \cdot \sqrt{\alpha}\left\|I^*_{j,k}\right\| \right)^2}{\sum_{j=0}^{N-1} \|A_{j,k} + I_{j,k}\|^2 \cdot \sum_{j=0}^{N-1} \alpha\|I_{j,k}\|^2} \\ &= \frac{\alpha \cdot \left(\sum_{j=0}^{N-1} \left\|I_{j,k} \cdot I^*_{j,k}\right\| \right)^2}{\alpha \cdot \sum_{j=0}^{N-1} \|A_{j,k} + I_{j,k}\|^2 \cdot \sum_{j=0}^{N-1} \|I_{j,k}\|^2} \end{aligned}$$

$$= \frac{\left(\sum_{j=0}^{N-1} \|I_{j,k}\|^2\right)^2}{\sum_{j=0}^{N-1} \|A_{j,k} + I_{j,k}\|^2 \cdot \sum_{j=0}^{N-1} \|I_{j,k}\|^2}$$

$$= \frac{\sum_{j=0}^{N-1} \|I_{j,k}\|^2}{\sum_{j=0}^{N-1} \|A_{j,k} + I_{j,k}\|^2} \tag{4.6}$$

Without the presence of interference, the score has a value of $\mu_k = 0$ according to (4.6), when the power of the RFI increase so does the score up to a maximum value of $\mu_k = 1$ that occurs when $\|A_{j,k}\|$ is negligible compared to $\|I_{j,k}\|$ or when $\|A_{j,k}\| = 0$. This result allows the detector to be interpreted as the power ratio between the RFI in the main signal and its total power.

4.3.3. Model implementation

The FRBs of interest are detected between 1.2 and 1.8 Ghz, so the sampling rate must be 1.2 GHz according to the Sampling Theorem, also undersampling is used in the third Nyquist zone to martch the desired frequency range, these concepts are presented in chapter 5.1 . Since the frequency of the Valon 5007 must be 8 times less than the desired sample rate, the model will be implemented to operate at 150 mhz.

The implementation of the diagram of figure 4.6 in Simulink is shown in figure 4.7, where a main and a reference signal are digitized with 8 parallel outputs of 8 bits per input in the orange block, in the purple blocks, a FFT of 2048 channels is calculated for the main and reference parallel outputs using PFBs to decrease the leakage between spectral channels. The output of each FFT has 4 parallel channels, but each value is complex, so they have an imaginary component, these values are received by the blue blocks that represent mathematical operations, calculating the squared module of each spectral channel of the FFT, and their conjugate multiplication, in other words, the main PSD, reference PSD and the CPSD between them are calculated. These values are accumulated in the green blocks with an accumulation N that depends on the value of a register of the FPGA, so that its value can be changed through a script. After accumulation, the outputs are re-quantized to 18 bits in the light blue blocks, because it is very expensive in terms of FPGA resources to use certain Simulink blocks with inputs with a higher number of bits, considering a bandwidth of 600 Mhz and 2048 spectral channels. The accumulated and requantized values of the PSDs and CPSD are operated according to 4.2 and stored in BRAMs, along with other values of interest to be read and processed by a script. In the requantization process, a window with the 18 most significant bits is selected, before this process the value is shifted to the left m bits, where m is the value stored in a register.

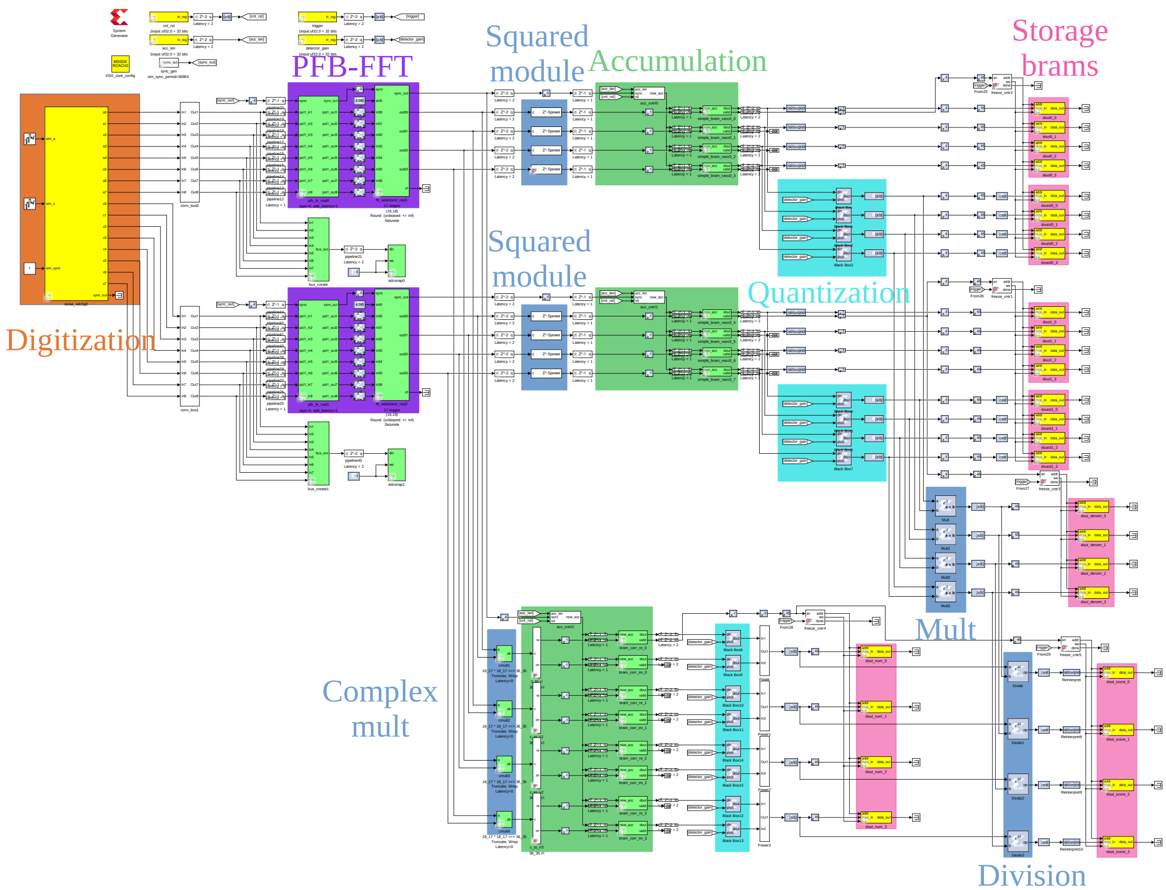


Figure 4.7: Simulink model of the detector.

4.3.4. Detector script

Using Python 2.7 to communicate with the FPGA over Ethernet, the BRAMs correspon-ding to the 2048-channel spectrometers of the main and reference signal are read to analyze the results of the score and debug the script and/or model. Data without requantizing is also read to visualize information lost by this process. Additionally, the values corresponding to the score μ_k of each channel are read, including the numerator and denominator data of the division corresponding to (4.2).

In addition to reading the data from the board, three registers of the FPGA are written by the script to interact with the model, there is a register to control the resetting of the model blocks, and another two registers with the values of the accumulation and detector gain, the last is used to move the 18-bit window in the requantization process.

Figure 4.8 shows the Graphical User Interface (GUI) of the implemented script, where the data stored in the BRAMs is plotted in real time, adding a rectangular area corresponding to the 18-bit window, which can be modified as well as the accumulation through the graphical interface that plots the data. Annex B.1 and Annex B.2 show the scripts corresponding to the GUI detector and its parameters, B.3 the scripts of the detector simulation.

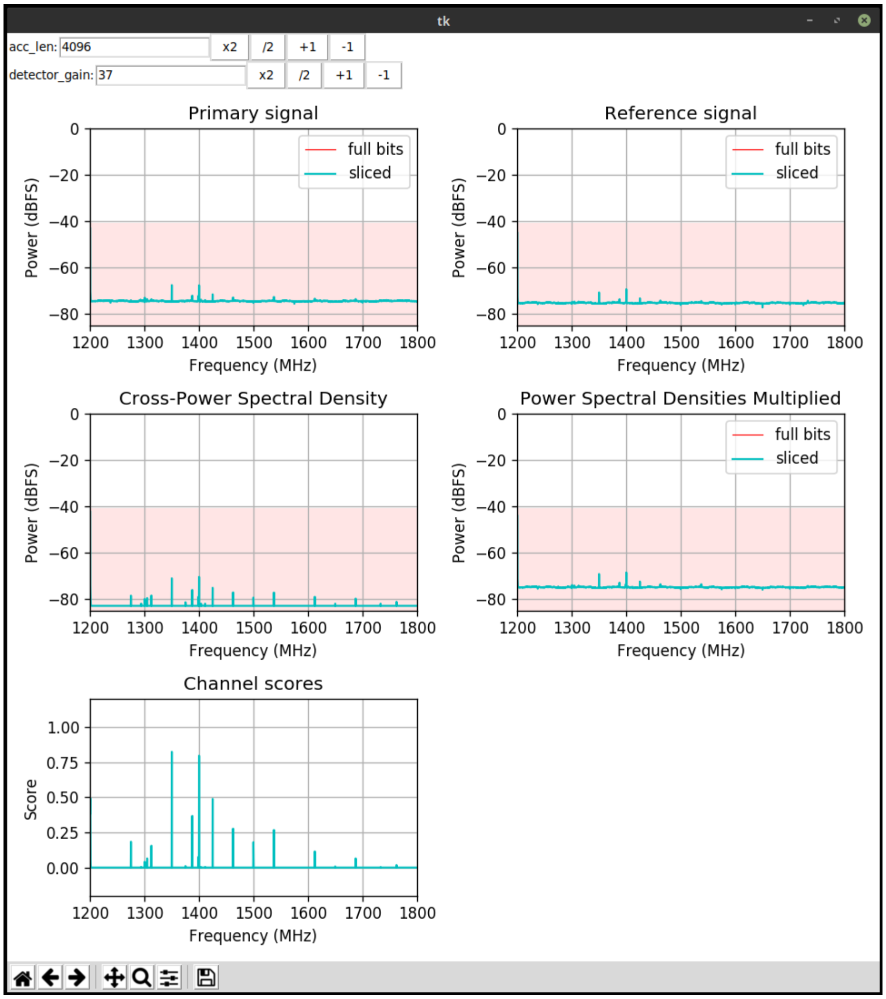


Figure 4.8: Detector script graphical interface.

A script was also made to simulate the input and reference signals using a Gaussian tone and noise with an SNR of 15 dB, in order to analyze the behaviour of the detector and compare it with measurements of injected tones in the laboratory. After this experience, the model is tested using two 1.4 GHz omnidirectional antennas as input. The diagrams corresponding to the circuits with the necessary components for these measurements are shown in figures 4.9 and 4.10.

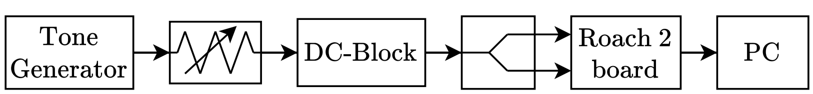


Figure 4.9: Component connection diagram to inject tones as main and reference inputs.

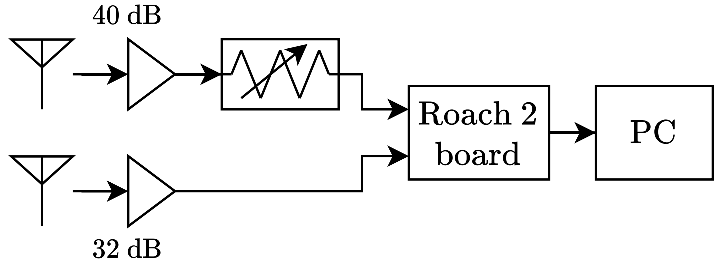


Figure 4.10: Component connection diagram using omnidirectional 1.4 GHz antennas as main and reference inputs.

4.4. Classifier

4.4.1. Design

New features are generated by calculating the first statistical moments, minimum and maximum over the data spectrum. As these features contain information about the type of distribution as shown in section 2.5, and as the maximum and average allow classifying between broadband and narrowband signals, these statistics will be used as features.

Subsequently, the dimensionality of the data is reduced using PCA and t-SNE to cluster using K-means and the score, so that for any value at 0.5, it is assumed as RFI detection.

Finally, the components that contain the most information are plotted using the two dimensional reduction methods indicated and clustered with K-means and the score μ_k. Figure 4.11 shows a diagram of the classifier methdology.

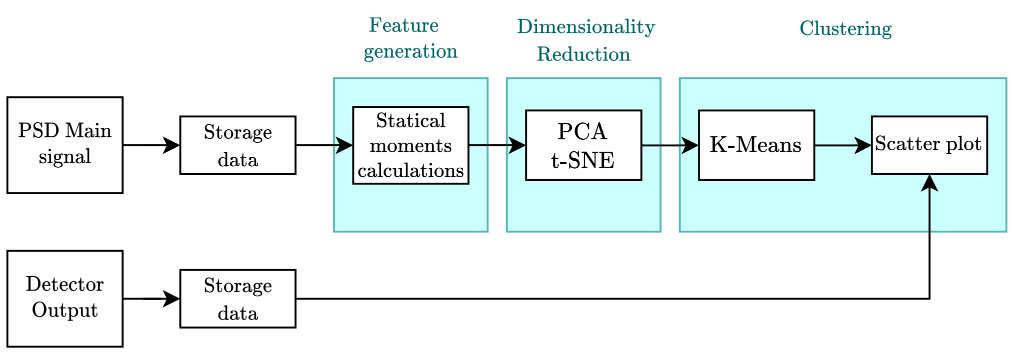


Figure 4.11: Classifier block diagram.

4.4.2. Script

A script is performed in Python to plot the data, when its dimensionality is reduced using PCA and t-SNE, and when it is clustered using K-means and the score. The code of this implementation is shown in Annex B.4.

The script also allows clicking on each point of the plot, to visualize the PSDs of the main and reference signals, in order to be able to visually analyze the relationship between the labels and the spectra.

Chapter 5

Detector and classifier limitations

5.1. Time Quantization

The digitization of a continuous signal in an ADC is carried out by measuring the signal every certain time interval defined by the sampling frequency fs. The Sampling Nyquist Theorem states that when sampling an analog signal, the sample rate must be at least twice the value of the signal's bandwidth. If the signal has a bandwidth BW, mathematically this inequality can be expressed as

$$f_s \geq 2 \cdot BW \tag{5.1}$$

When calculating the DFT according to 2.5, there is a multiplication by an exponential imaginary argument, so its values will wrap around $2 \cdot BW$. In this way, any spectrum that is contained between $i \cdot BW$ and $(i+1) \cdot BW$ will be seen in the first Nyquist zone when i is even as shown in figures 5.1 and 5.2, but for odd values, the spectrum will also be horizontally inverted. The power densities of the signals that coincide in the same frequency ranges in the first Nyquist zone will be added together.

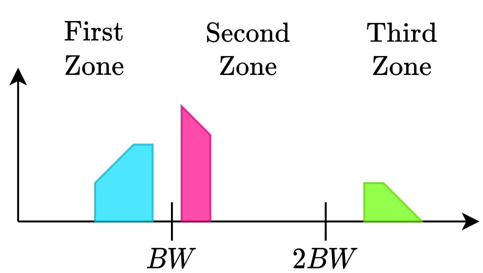


Figure 5.1: Example of the spectrum of a signal with frequency components greater than the ADC Nyquist BW, showing the first three Nyquist zones.

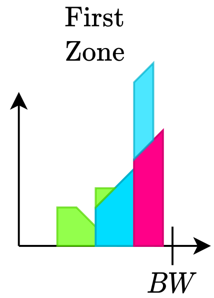


Figure 5.2: Digitization of a signal with a bandwidth greater than BW.

This also allows a sampling technique called undersampling, when sampling with a bandwidth BW, and using a bandpass filter at the nyquist area of interest, the information will be fully contained in the first Nyquist area after sampling, if the filter is used in an even-numbered zone, the frequency axis must be reversed.

5.2. Amplitude Quantization

Digitization also approximates the analog values of the signal, if the ADC has a resolution of N bits, there will be a quantity of 2^N values to approximate the signal, adding a quantization noise corresponding to the difference between the original and quantized signals as shown in figure 5.3. The signal to noise ratio produced by this phenomenon is

$$SNR = N \cdot 6.02\,\mathrm{dB} + 1.76\,\mathrm{dB}, \tag{5.2}$$

when injecting a sinusoidal signal whose RMS value coincides with the maximum digital value of the ADC, if this relationship is considered for a single channel of a FFT with size M [8], it becomes

$$SNR = N \cdot 6.02\,\mathrm{dB} + 1.76\,\mathrm{dB} + 10 \cdot \log_{10}\left(\frac{M}{2}\right)\mathrm{dB}. \tag{5.3}$$

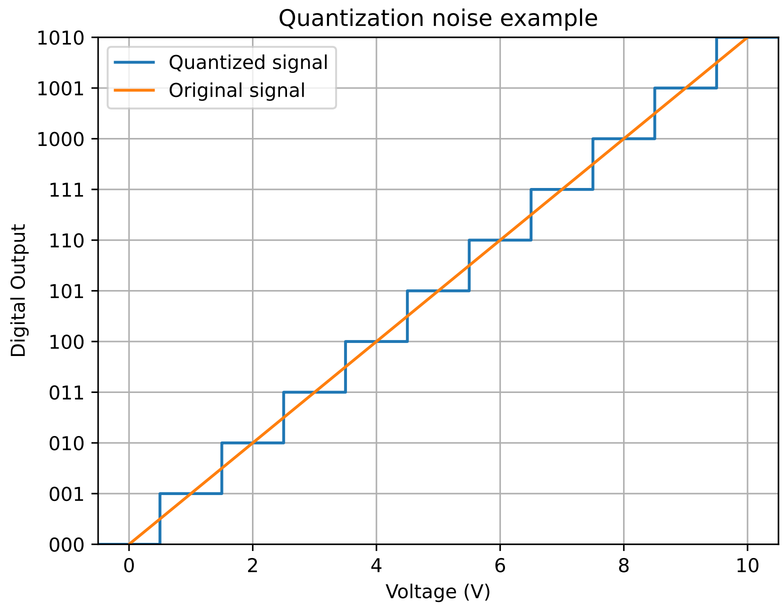


Figure 5.3: Example of a signal and its quantization, the difference between them correspond to the quantization error.

5.3. Frequency Quantization

The bandwidth of a spectral channel of the FFT, corresponds to its frequency resolution BW_m, which is defined as

$$BW_m = \frac{BW}{M} \tag{5.4}$$

If two components from different sources are contained within the same spectral channel, after calculating the FFT there will be a unique value for this channel, so the information from both sources is mixed making them indistinguishable.

5.4. Detector Quantization

It is necessary to re-quantize the representation of the numbers to decrease the use of FPGA resources, detector output is 32 bits with a representation corresponding to the square module of the CPSD and PSDs respectively. Thus, the calculation of decibels is done on the module without being squared, half of the bits are considered, obtaining a representation with a precision of 48 dB, when calculating

$$10 \cdot log_{10}\left(2^{16}\right) = 48 \text{ dB}. \tag{5.5}$$

Chapter 6

Results and analysis

6.1. Detector

6.1.1. Simulations

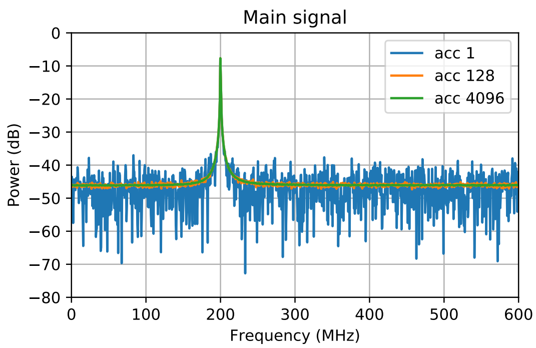
The detector is simulated to analyze the impact of the accumulation and how it behaves depending on it. For this purpose, figures 2.3 and 2.4 show score plots and PSDs in two cases of interest, when both the main signal and the reference signal have a tone of 200 MHz, or when it is only present in the main signal. In addition to this, a Gaussian noise is added to each input to have an SNR of 15 dB.

The effect of the accumulation on the PSDs and their multiplication, is the elimination of Gaussian noise as shown in Figures 6.1a, 6.1b and 6.1d, due the contributions of the noise to the spectral channels power tend to the same value, the spectrum signal shifts upward. On the other hand the integration of the CPSD does not show the same behaviour, it does not eliminate the gaussian noise, but as the accumulation increases the noise floor decreases as seen in figures 6.1a and 6.2a, in the last figure, the tone regardless of the accumulation, has the same amplitude, which is expected according to (4.4), where it was used that for a sufficiently large N, the contribution of $A_{j,k} \cdot R^*_{j,k}$ is negligible as they are not correlated, but $I_{j,k} \cdot R^*_{j,k}$ survives by having constant phase. In this case, the tone of figure 6.1c and the gaussian noise can be considered as $A_{j,k}$ as they are not correlated with the reference signal, having a score of 0 for every channel in 6.4e when accumulating, in contrast, the tones of figure 6.2c correspond to $I_{j,k}$ and $R_{j,k}$ since they are correlated reaching a score of 1 for 200 Mhz in 6.2e when accumulating.

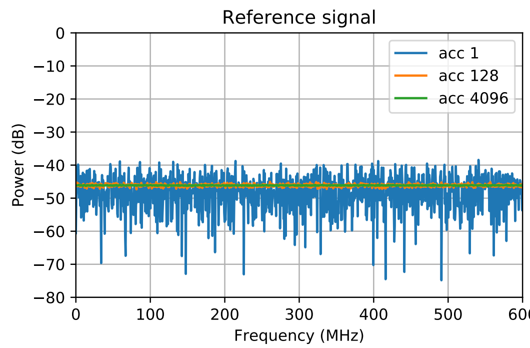
By not using PFBs before calculating the DFT, the effect of the leakage will be greater in the simulation than the real implementation, this behaviour can be seen in the PSDs of figures 6.1a, 6.2a and 6.2b. The effect of leakage on the score can be seen in figure 6.2e, where in the vicinity of the tone frequency the score has a non zero value.

When there is no accumulation, the score is equal to one for all the frequencies in figures 6.1e and 6.2e, in both cases the CPSD is equal to the multiplication of the PSDs of the main and reference signals. This behaviour is explained mathematically according to 4.2, where by not accumulating it becomes

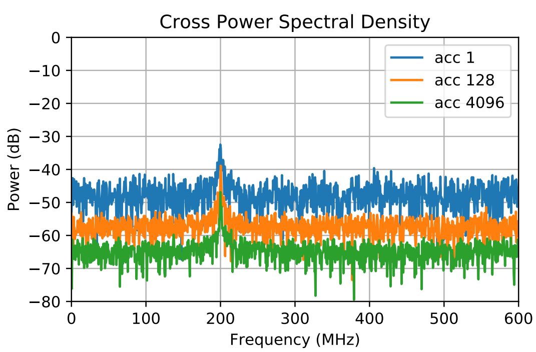
$$
\begin{aligned}
\mu_k &= \frac{\|X_k \cdot Y_k^*\|^2}{\|X_k\|^2 \cdot \|Y_k\|^2} && (6.1)\\
&= \frac{\|X_k\|^2 \cdot \|Y_k\|^2}{\|X_k\|^2 \cdot \|Y_k\|^2} && (6.2)\\
&= 1. && (6.3)
\end{aligned}
$$



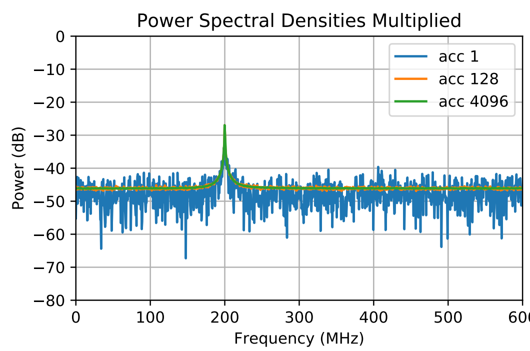
(a) Main integrated PSD.



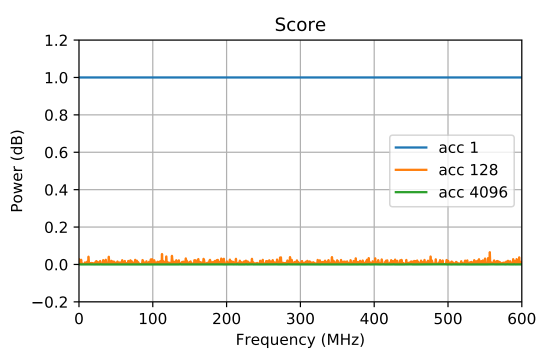
(b) Reference integrated PSD.



(c) Integrated CPSD between main and reference signal.

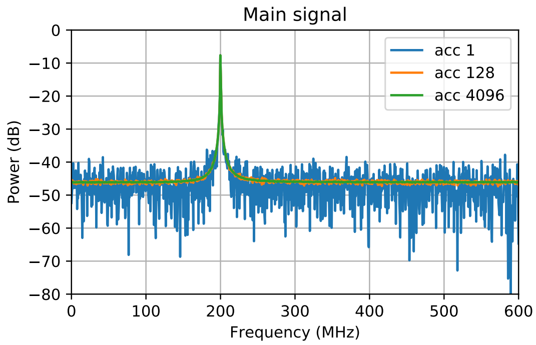


(d) Integrated PSDs multiplied between main and reference signal.

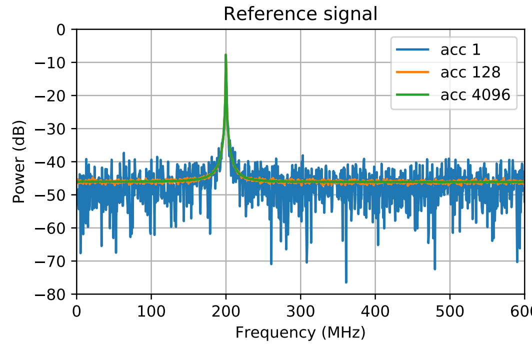


(e) Score between main and reference signal.

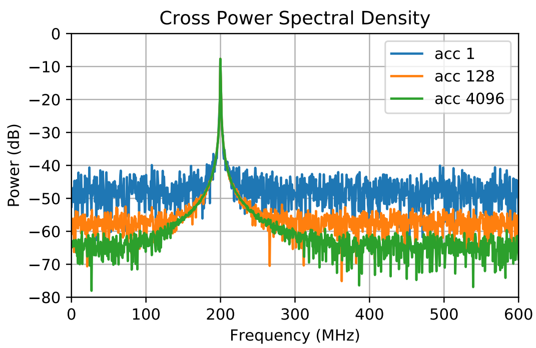
Figure 6.1: Simulation of a 200Mhz tone in one input with $SNR = 15$ dB, and equivalent gaussian noise in the reference input, acc denotes accumulation.



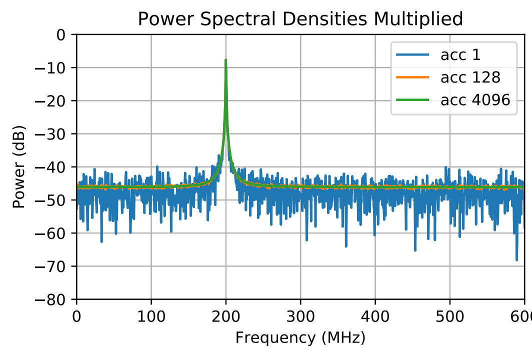
(a) Main integrated PSD.



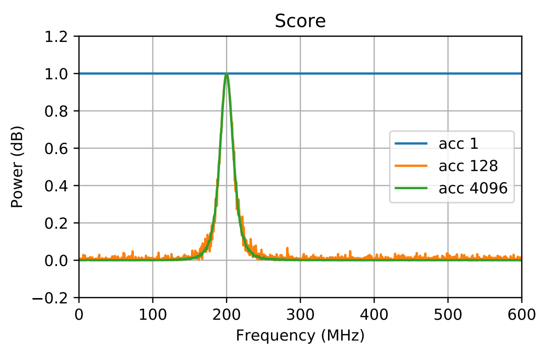
(b) Reference integrated PSD.



(c) Integrated CPSD between main and reference signal.



(d) Integrated PSDs multiplied between main and reference signal.



(e) Score between main and reference signal.

Figure 6.2: Simulation of a 200Mhz tone in two inputs with $SNR =$ 15 dB, acc denotes accumulation.

6.1.2. Experimental Tests

To compare with the simulation, the same experiment is carried out implementing it in the FPGA, injecting a 1300 Mhz tone. Figures 1 and 2 show the detector GUI for when tone is injected into one or both FPGA input. As the bandwidth is 600 MHz, the 1300 MHz tone is seen as a 100 MHz tone in the first Nyquist zone. Because of this, when plotting the third Nyquist zone, the harmonics of the 1300 MHz tone are every 100 MHz.

Figures 6.4a, 6.5a and 6.5b show tone peaks and their harmonics. There is also the presence of other peaks in the plots that correspond to ADC calibration errors, which, being present in the digitization of the primary and reference signal, are correlated so they will have a value in the score.

Unlike the simulation, the use of PFB is done before calculating the DFT, in graphs 6.4e and 6.5e the score corresponding to the tones has no leakage effect. The score value in figure 6.5e shows high score values when the tone is in both inputs, compared to figure 6.4e, where the values are lower, considering an accumulation of 4096.

The implementation of this test is shown in figure 6.3, according to 4.9, where a signal generator emits a tone of 1300 MHz with -4 dbm that passes a DC-block to eliminate any continuous component that can leak and damage a component. The signal enters a variable attenuator to decrease the peak power so that it remains within the red window shown in figures 6.4 and 6.5, which corresponds to where the data is well represented in the detector. Then the signal goes into a splitter to connect to the main and reference signal, the photo shows the case when it is only connected to the main one.

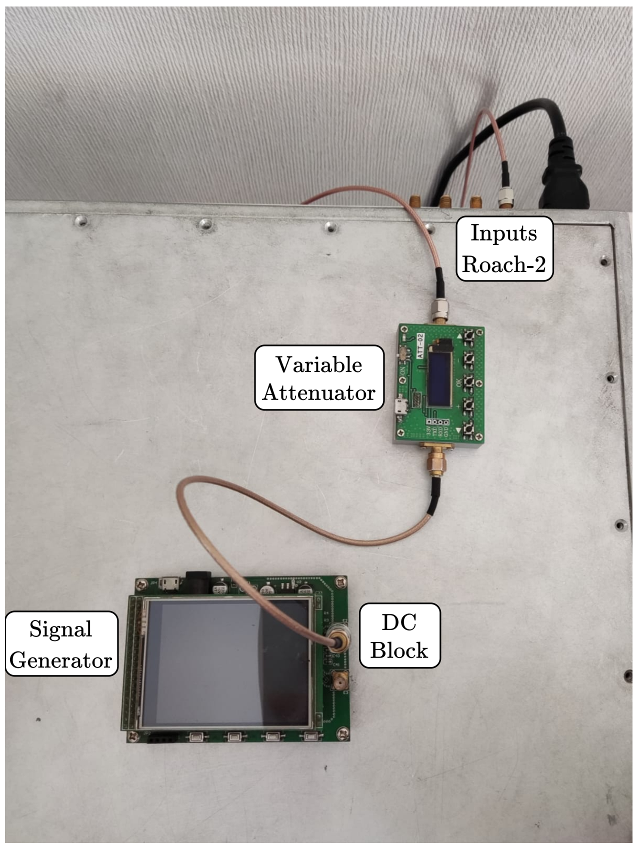
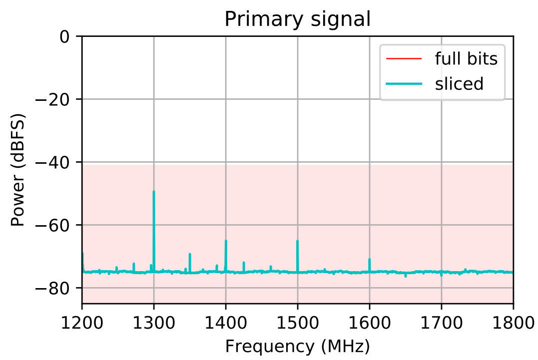
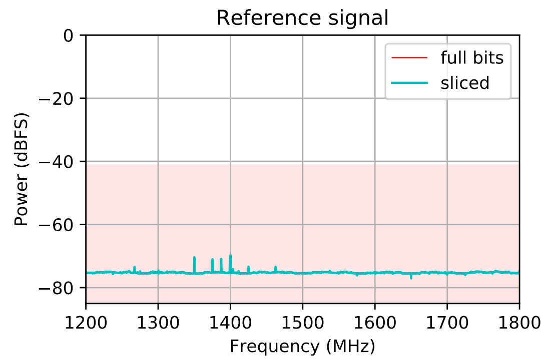


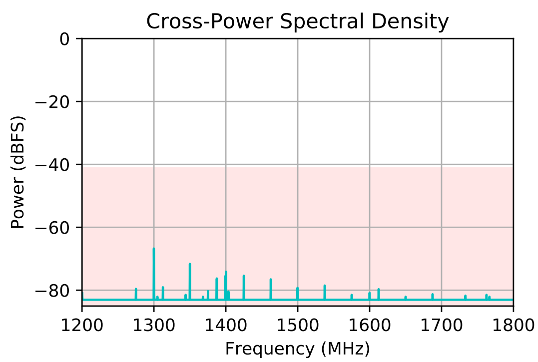
Figure 6.3: Circuit corresponding to the injection of a tone as the main input of the board. Reference channel is not connected.



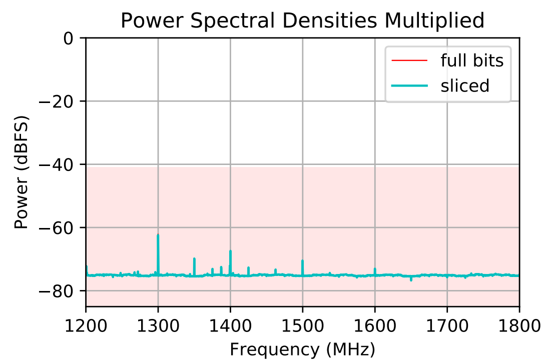
(a) Main integrated PSD.



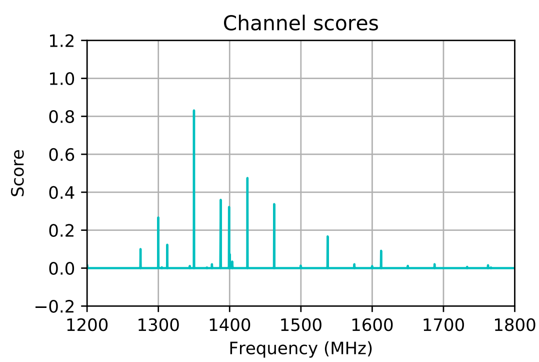
(b) Reference integrated PSD.



(c) Integrated CPSD between main and reference signal.

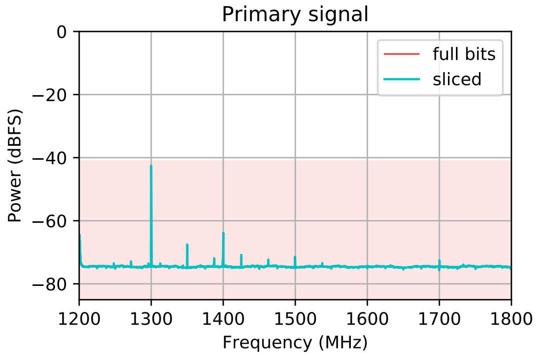


(d) Integrated PSDs multiplied between main and reference signal.

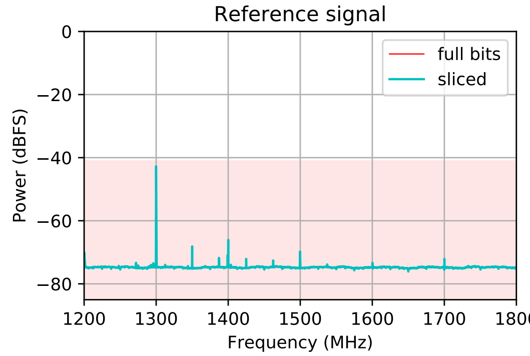


(e) Score between main and reference signal.

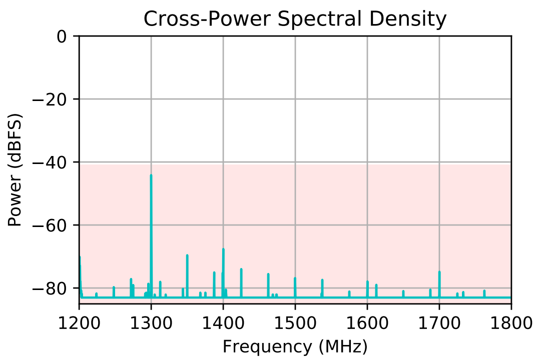
Figure 6.4: Detector GUI with a tone of 1300 MHz as the main signal and no reference signal injected.



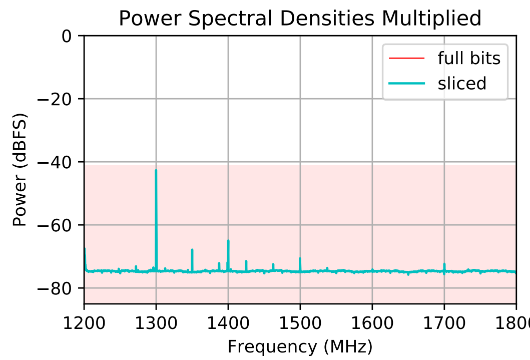
(a) Main integrated PSD.



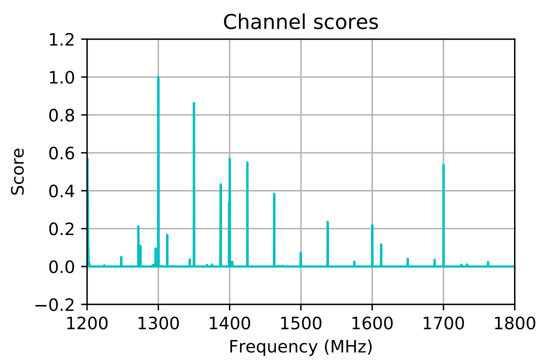
(b) Reference integrated PSD.



(c) Integrated CPSD between main and reference signal.



(d) Integrated PSDs multiplied between main and reference signal.



(e) Score between main and reference signal.

Figure 6.5: Detector GUI with a tone of 1300 MHz as main and reference signals.

To test the detector in a more realistic environment, measurements are made using two 1.4 GHz omnidirectional antennas as the main and reference signals. Both signals are amplified by 32 and 40 dB, where the last one corresponds to the main one and a variable attenuator is used to control signal power, figure 6.6 shows the circuit corresponding to these measurements, according to 4.10.

Since this test were carried out in a building in the center of the city so there is a lot of radio emissions. Figure 6.7 shows the GUI of the circuit of figure 6.6. As both antennas are so close and have the same radiation pattern, they should have a very similar PSD, implying a large number of score values close to one. On the contrary, the frequencies where there are no emissions have values close to zero because the noise floors are not correlated. Figure 6.7c shows what happens when a value smaller than that of the window is quantized, equating it to the noise floor of the ADC, on the other hand, if a value exceeds the maximum of the window, it will be wrapped around the floor of noise.

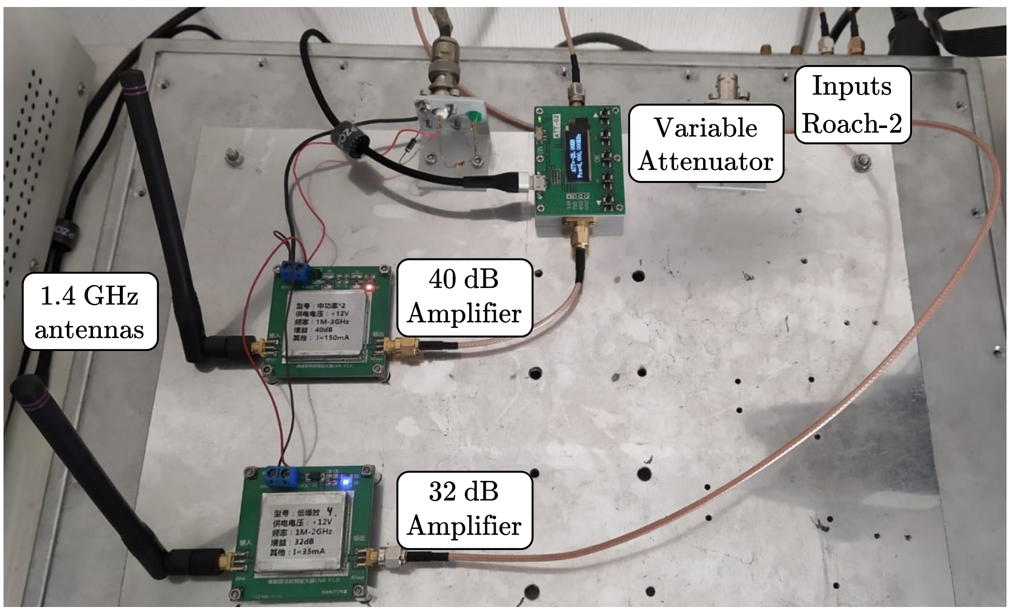
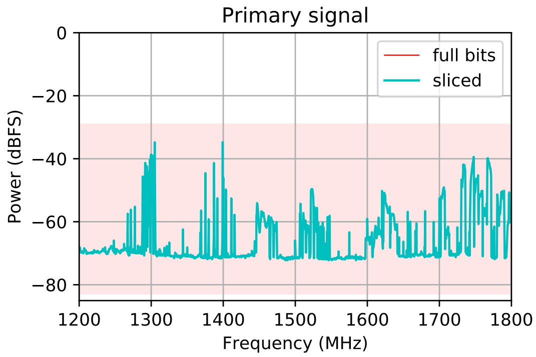
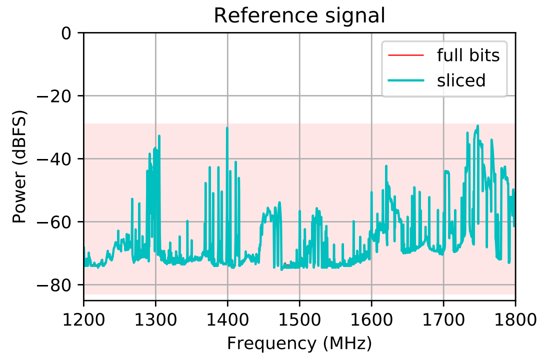


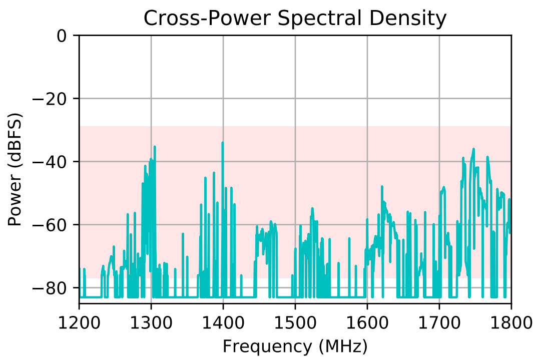
Figure 6.6: Circuit corresponding to two 1.4 Ghz omnidirectional antennas as the main input of the board.



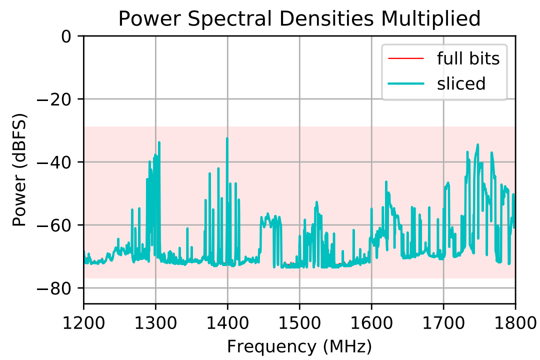
(a) Main integrated PSD.



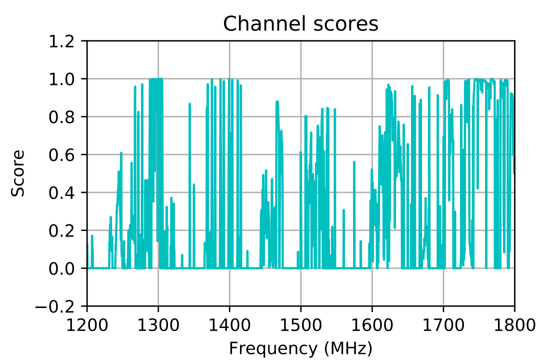
(b) Reference integrated PSD.



(c) Integrated CPSD between main and reference signal.



(d) Integrated PSDs multiplied between main and reference signal.



(e) Score between main and reference signal.

Figure 6.7: Detector GUI with two 1.4 Ghz omnidirectional antennas as inputs.

The detector is used in the implementation of the Astronomical Radio Transients Experiment (ARTE) project developed by the MWL of the Universidad de Chile, which is an array of antennas with a radiation pattern made for the angular distribution of the Milky Way in order to study FRBs in it, with a 600 MHz bandwidth. Also a digital Direction of Arrival (DoA) is used to locate the sources in the sky. The detector is used to reduce the amount of false positives in the detection of FRBs, and not to save unnecessary data such as the location of the source. ARTE implementation is used to take measurements in a real environment of astronomical observations to see the behavior of the detector. This is done at the facilities of the Department of Astronomy located in Cerro Calán in Las Condes.

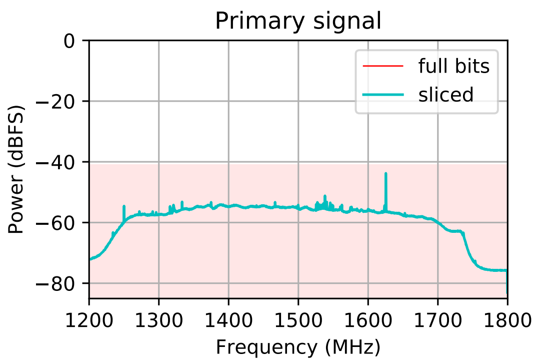
Figure 1.2 shows the arrangement of the antennas during the test, where the radiation received by two of the atenna arrays are combined, synthesizing them into a main signa. The reference antenna whose radiation pattern points to the horizon as the reference, a band pass filter is used between 1200 and 1800 Mhz before digitizing the signals.

for all FRBs detections, it is verified if any of the spectral channels of the corresponding score exceeds a value of 0.5, of the 1801 FRB detections produced, 1791 meeting the threshold condition corresponding to 99.4448 % of the detections. In this case, given that the RFI appears sporadically and in few spectral channels simultaneously, this threshold can be used, but depending on the characteristics of the RFI, the detection condition must change, the score values being still valid.

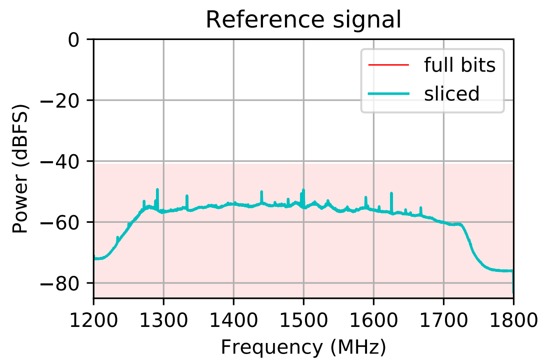
In figure 6.8 GUI of the detector is shown with the measurements made with the antennas of figure 6.8 for a spectrum contaminated with RFI, specifically between 1600 and 1700 MHz, the presence of a tone corresponding to RFI that appears in 6.9a and 6.9b, having a score value for that spectral channel according to 6.9e greater than 0.5, the rest of the spectrum has values close to zero except for some components that correlate.

Unlike figure 6.7, the window is sufficient to represent the data well and the plot of 6.9c shows a clear decrease in the power of the CPSD, compared to the multiplication of the main and reference PSDs 6.9d, indicating low values for the score.

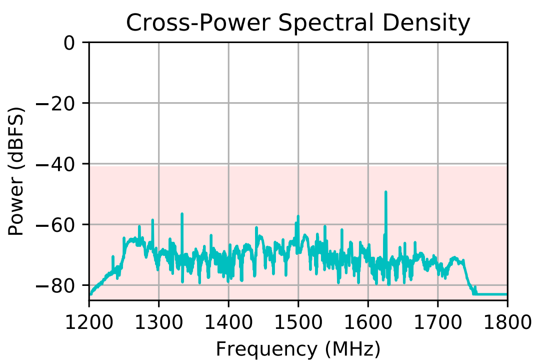
Figure 6.8: Antennas configuration to take measurements.



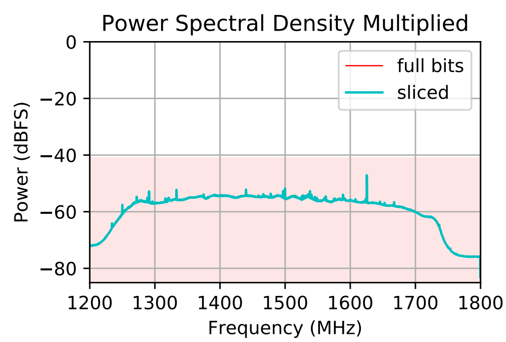
(a) Main integrated PSD.



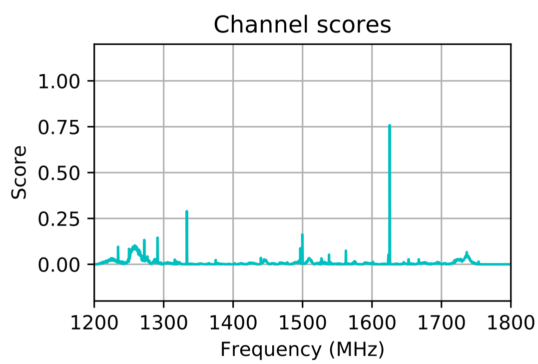
(b) Reference integrated PSD.



(c) Integrated CPSD between main and reference signal.



(d) Integrated PSDs multiplied between main and reference signal.



(e) Score between main and reference signal.

Figure 6.9: Detector GUI with two directional antennas as inputs.

To characterize the measured RFI, the main emission sources are identified, associated with the frequency bands in which the devices that generate them work [16]. Figure 6.10 shows a spectrum of the main signal, with the frequency bands corresponding to different RFI sources.

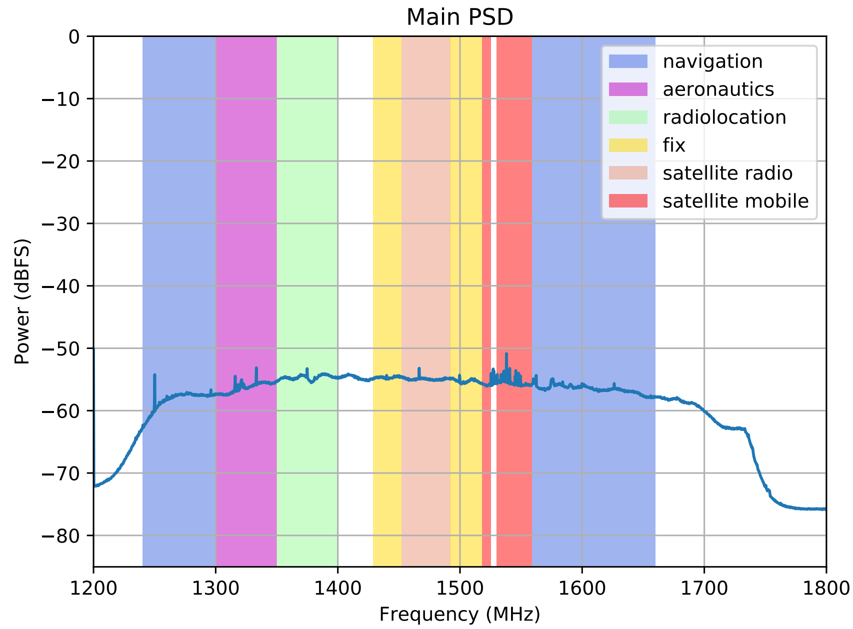
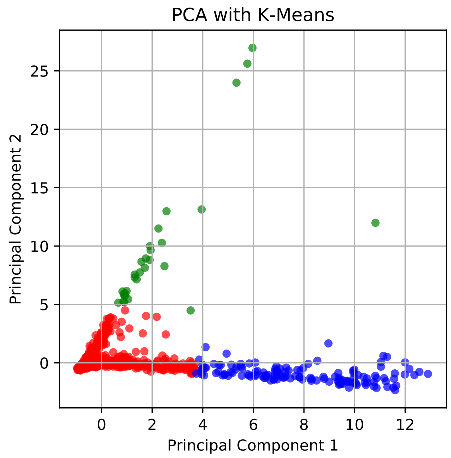


Figure 6.10: RFI frequency bands, for a main signal spectrum.

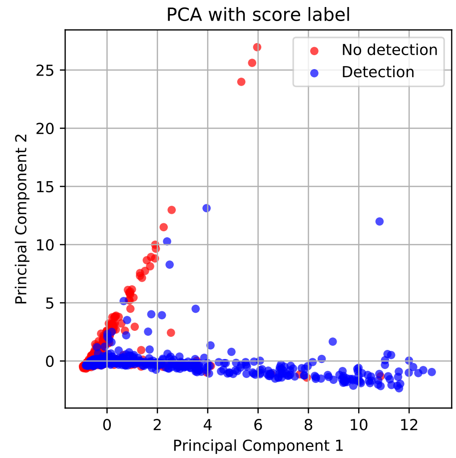
6.2. Classifier

Figures 6.11a and 6.11c show the use of PCA to decrease the dimensionality of the dataset obtained from ARTE measurements, with a size of 2765 by 6, corresponding to the number of samples and features. This features are the first four statistical moments and the maximum and minimum for each spectrum of the main signal dataset. Classifying according to the score and K-means, different numbers of clusters are tested and analyzed using the GUI of the script to determine if the clusters are representative, allowing access to the spectrum of any point in figure 6.11. Similarly in Figures 6.11b and 6.11d the same procedure is carried out, using t-SNE as a tool for decreasing dimensionality.

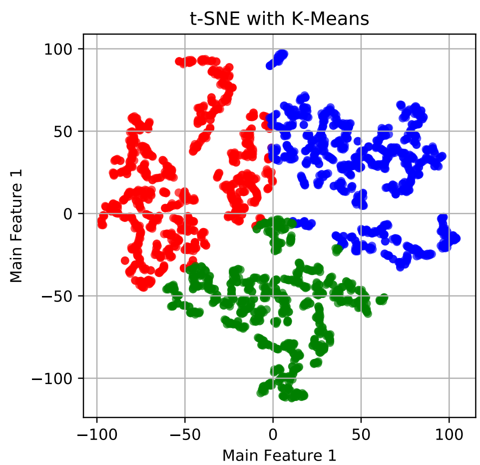
When analyzing the clusters generated by K-Means, no relationship is seen between the points and the spectra of the main and reference signal corresponding to it, regardless of whether PCA or t-SNE is used. When using the score to classify RFI detections, data with the same classification are concentrated in some places of the space in Figures 6.11b and 6.11d. Because the components that contribute the most to the decrease in dimensionality are Skewness and Kurtosis, these clustering zones are mainly due to a measure of Gaussianity, since the RFI has a non-Gaussian behaviour.



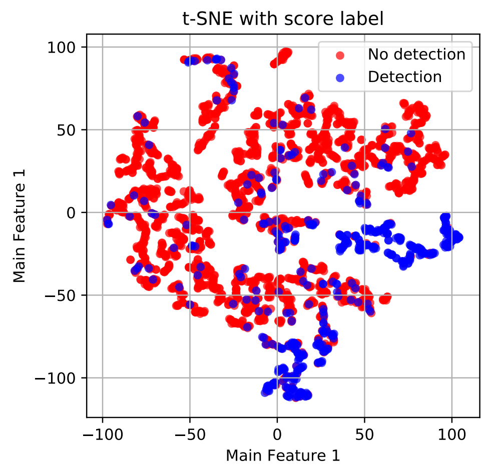
(a) Clustering using PCA to reduce dimensionality and K-means to label three clusters.



(b) Clustering using PCA to reduce dimensionality and score threshold to label.



(c) Clustering using t-SNE to reduce dimensionality and K-means to label three clusters.



(d) Clustering using t-SNE to reduce dimensionality and score threshold to label.

Figure 6.11: Classifier GUI with two directional antennas as inputs.

Figure 6.12 shows a zoom to where the largest amount of data from 6.11b is concentrated where there is no obvious clustering, but when moving away from this area the detections are concentrated in the lower part of the plot, while the non-detections on top. However, when analyzing the spectra, there is no clustering of characteristics such as whether the source of the interference is broadband or narrowband. For figure 6.13 a concentration of data classified as detections of figure 6.11d is shown, as for PCA, there are areas where there is a clear clustering, and areas where a predominant class cannot be distinguished.

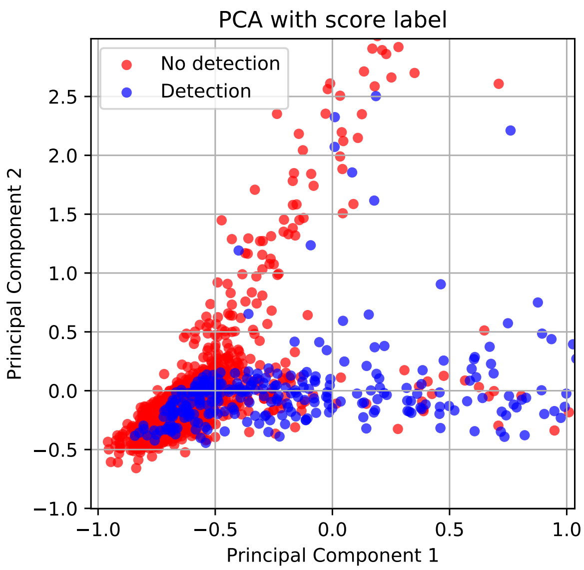


Figure 6.12: Zoom in of the figure 6.11b.

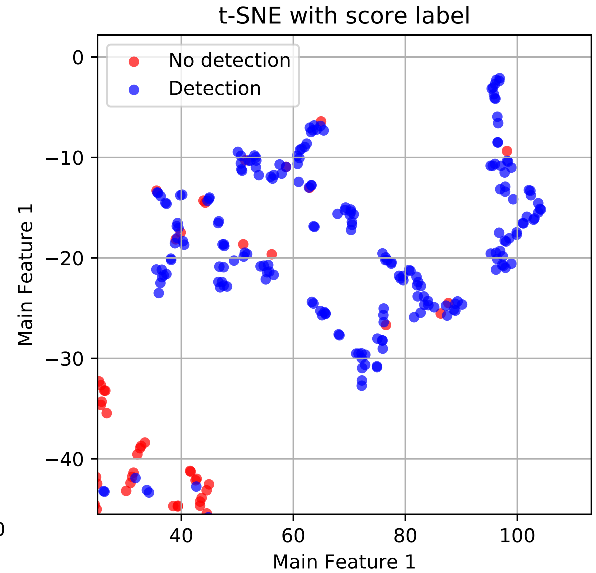


Figure 6.13: Zoom in on a cluster of figure 6.11b.

The GUI allows the visualization of the spectra and the score of each data plotted in figure 6.11, the spectra and score of two data in figure 6.11b are shown in figures 6.14 and 6.15. Visually it is verified that if the maximum score value of a spectrum exceeds the value of 0.5, the spectrum will be considered as contaminated, otherwise it will be considered clean.

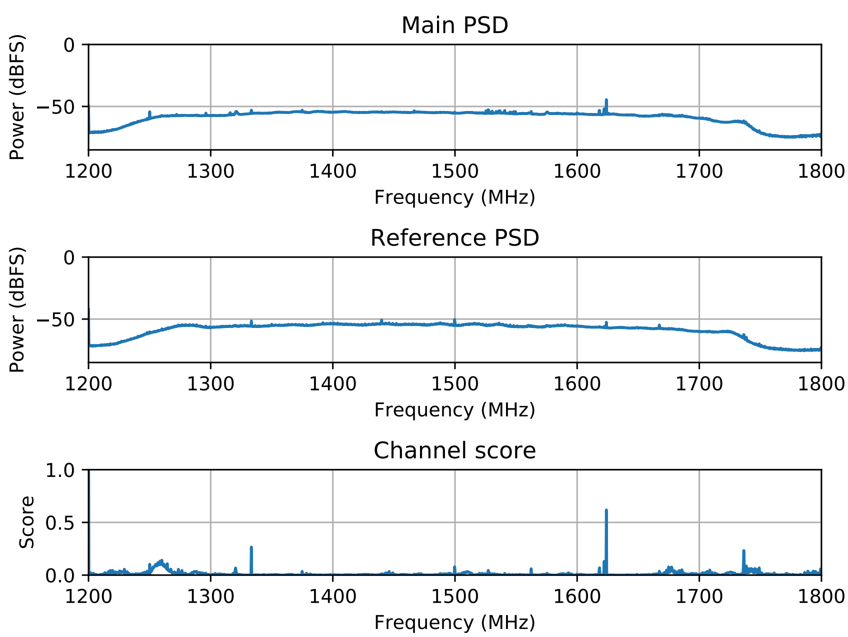


Figure 6.14: GUI example of a contaminated spectrum, using a value of 0.5 as detection threshold.

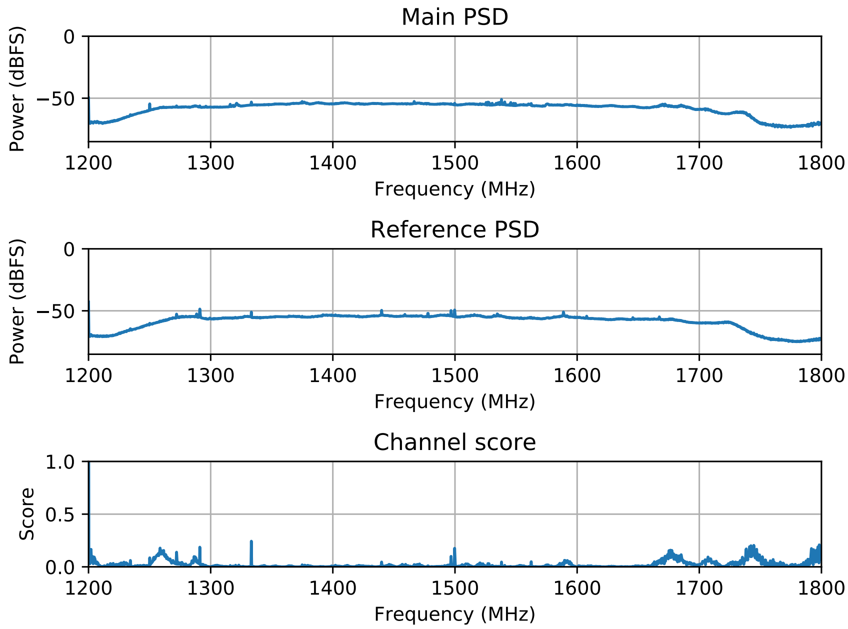


Figure 6.15: GUI example of a clean spectrum, using a value of 0.5 as detection threshold.

Chapter 7

Conclusions

This work presents the design, implementation and tests of a real-time RFI detector, with a 600 Mhz bandwidth, an 8-bit ADC resolution, a spectral resolution of 293 kHz per spectral channel, and a dynamic range of 48 dB, which can be shifted using the gain of the detector. In addition to the exploration of a classification method using the statistical moments, maximum and minimum of the signal.

The implementation of the filter is done in real time and has the necessary parameters in terms of resources and speed, to operate in real astronomical observations. Depending on the type of RFI present in the observation, it is necessary to set the detector's gain to represent the signals, and choose a threshold accordingly, when making measurements in an astronomical measurement environment where the RFI is sporadic, it is enough to use that any spectral channel exceeds a certain value. Furthermore, this can be combined with other mitigation techniques, such as flagging spectral channels with permanent RFI.

The derivation of the score was studied from the measure of normalized similarity between two signals, showing that the detector can be interpreted as the ratio between the interference power captured by the main signal and its total power, also considering the astronomical signal when accumulating enough so that the power contributions of the uncorrelated elements are negligible.

The detector is tested with the array of antennas of the ARTE project of MWL, which made 1801 detections in a time of 105 minutes, of which 1791 were identified as RFI, allowing to eliminate a total of 99.4448 % detections corresponding to RFI and not a FRB.

Power spectrum characterization methods were reviewed to classify RFI. In particular, since the RFI is not Gaussian, and the mean and maximum are used as indicators to classify broadband and narrowband signals, the first four statistical moments, the maximum and minimum of a spectrum, are used as characteristics. There is a relationship between the detections, given by the Gaussianity measure of the spectra, but clustering does not occur in broadband or narrowband when using PCA and t-SNE to dimensionality reduction.

Chapter 8

Future Work

It is proposed to study other methods of spectrum characterization, such as the Taylor series expansion presented in [10]. In addition to this, other characteristics can be calculated without knowing their relationship with the signals, because the dimensionality reduction methods eliminate the characteristics that less provide information about the dataset. On the other hand, when classifying only instantaneous spectra were considered, when the RFI is a temporary event, so it can be approached as an image problem where the magnitude of the spectra is represented with colors, as a function of frequency and time.

The detector works when the RFI signals are seen by both antennas, as the radiation pattern of the reference one points towards the horizon, it is not capable of measuring RFI present in the sky, such as that produced by satellite communications. In this way the detector is blind for these cases, and other mitigation and / or detection methods must be explored.

Also depending on the application, the detector can be used without the need for re-quantizing to increase the dynamic range. This can be done by reducing the bandwidth and the size of the FFT. In addition, in the final implementation, an ADC was used to measure two signals, so changing the configuration so that it only measures one signal, allows to have double the bandwidth and greater slack in the use of resources.

Bibliography

[1] P. A. Fridman and W. A. Baan, "RFI mitigation methods in radio astronomy," *Astronomy and Astrophysics -Berlin-*, vol. 413, no. 3, pp. 1087–1094, 2004.

[2] G. Barıs, S. M., and S. B., "Studies of Radio Frequency Interference Detection Methods in Microwave Radiometry," *International Journal of Automotive Technology*, vol. 16, no. 5, pp. 721–731, 2015.

[3] J. Lahtinen, J. Uusitalo, T. Ruokokoski, and J. Ruoskanen, "Evaluation and comparison of RFI detection algorithms," *14th Specialist Meeting on Microwave Radiometry and Remote Sensing of the Environment, MicroRad 2016 - Proceedings*, pp. 62–67, 2016.

[4] R. Weber and C. Faye, "Real time detector for cyclostationary RFI in radio astronomy," *European Signal Processing Conference*, vol. 1998-Janua, 1998.

[5] J. M. Ford and K. D. Buch, "RFI mitigation techniques in radio astronomy," *International Geoscience and Remote Sensing Symposium (IGARSS)*, no. July, pp. 231–234, 2014.

[6] H. Zhang, H. Gan, Y. Yue, H. Hu, S. Huang, J. Song, and H. Liu, "RFI mitigation on mobile base stations around FAST," *2017 32nd General Assembly and Scientific Symposium of the International Union of Radio Science, URSI GASS 2017*, vol. 2017-Janua, no. August, pp. 1–3, 2017.

[7] G. M. Nita, D. E. Gary, Z. Liu, G. J. Hurford, and S. M. White, "Radio Frequency Interference Excision Using Spectral-Domain Statistics," *Publications of the Astronomical Society of the Pacific*, vol. 119, no. 857, pp. 805–827, 2007.

[8] F. A. Curotto, "Design, Implementation and Characterization of a Radio Frequency Interference Digital Adaptive Filter using a Field-Programmable Gate Array," Universidad de Chile, 2019.

[9] D. C. Price, "Spectrometers and Polyphase Filterbanks in Radio Astronomy," 2021, no. 1, pp. 159–179. [Online]. Available: http://arxiv.org/abs/1607.03579

[10] K. Abazajian, K. Kadota, and E. D. Stewart, "Parametrizing the power spectrum: Beyond the truncated Taylor expansion," *Journal of Cosmology and Astroparticle Physics*, no. 8, pp. 141–151, 2005.

[11] P. A. Fridman, "RFI excision using a higher order statistics analysis of the power spectrum," *Astronomy and Astrophysics -Berlin-*, vol. 413, no. 3, pp. 1087–1094, 2004.

[12] W. Schaefer, “Measurement of impulsive signals with a spectrum analyzer or EMI receiver,” in *IEEE International Symposium on Electromagnetic Compatibility*, vol. 1, 2005, pp. 267–271.

[13] C. R. García-Alonso, L. M. Pérez-Naranjo, and J. C. Fernández-Caballero, “Multiobjective evolutionary algorithms to identify highly autocorrelated areas: The case of spatial distribution in financially compromised farms,” *Annals of Operations Research*, vol. 219, no. 1, pp. 187–202, 2014.

[14] Xilinx Inc., “Virtex-6 FPGA DSP48E1 Slice: User Guide,” *Xilinx Technical Documentation*, vol. 369, no. ug369, pp. 1–52, 2011. [Online]. Available: https://www.xilinx.com/support/documentation/user_guides/ug369.pdf

[15] M. A. Kramer, “An introduction to field analysis techniques: the power spectrum and coherence,” *Society for Neuroscience*, pp. 18–25, 2013.

[16] “Estudio relativo a la actualización del Plan General de Uso del Espectro Radioeléctrico Actualización Marco Regulatorio y Evolución Sector de Telecomunicaciones,” Tech. Rep., 2005.

Appendix A

Mathematical derivations

A.1. Power Spectral Density in terms of correlation

Let $x[n]$ and $y[n]$ be two discrete functions wich DFTs are respectively $X[k]$ and $Y[k]$, the convolution of two discrete signals is defined as

$$(x[n] * y[n])\,[m] = \sum_{n=-\infty}^{\infty} x[m-n] \cdot y[n] \tag{A.1}$$

and the Convolution Theorem states that

$$x[n] * y[n] = X[k] \cdot Y[k]. \tag{A.2}$$

The relationship between the correlation and the convolution is given by

$$(x[n] \star y[n])\,[k] = (x^*[-n] * y[n])\,[k], \tag{A.3}$$

so the DFT of the correlation using (A.3), the Time-Reversal Property which states that $\mathrm{DFT}\{x[-n]\} = X[-k]$ and assuming x[n] and y[n] $\in \mathbb{R}$, can be expressed as

$$\begin{aligned} \mathrm{DFT}\{x^*[-n] \star y[n]\} &= \mathrm{DFT}\{x^*[-n]\} \cdot \mathrm{DFT}\{y[n]\} \\ &= \mathrm{DFT}\{x[-n]\} \cdot \mathrm{DFT}\{y[n]\} \\ &= X[-k] \cdot Y[k], \end{aligned} \tag{A.4}$$

where $X[-k]$ can be conveniently becomes

$$\begin{aligned} X[-k] &= \sum_{n=0}^{N-1} x[n] \cdot e^{\frac{j2\pi}{N}kn} \\ &= \left(\sum_{n=0}^{N-1} x[n] \cdot e^{\frac{-j2\pi}{N}kn} \right)^* \\ &= X^*[k]. \end{aligned} \tag{A.5}$$

Using this result, (A.4) takes de form of

$$\mathrm{DFT}\{x^*[-n] \star y[n]\} = X^*[k] \cdot Y[k]. \tag{A.6}$$

If equation (2.7) is interpreted as the DFT of the autocorrelation, and the result obtained in (A.6) is used. The power spectral density $S[k]$ of a signal $x[n]$ can be written as

$$\begin{aligned} S[k] &= \mathrm{DFT}\{x^*[-n]\} \cdot \mathrm{DFT}\{x[n]\} \\ &= X^*[k] \cdot X[k] \\ &= \|X[k]\|^2 \end{aligned} \tag{A.7}$$

Appendix B

Scripts

B.1. Detector scripts

Código B.1: Detector main script.

```
import matplotlib
import numexpr
import math
import time
from matplotlib import patches as pat
from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg, NavigationToolbar2Tk
from matplotlib.figure import Figure
import Tkinter as tk
from matplotlib import animation
from detector_parameters import *
import calandigital as cd
import matplotlib.pyplot as plt
import matplotlib as mpl
import numpy as np
import os
from matplotlib.transforms import Bbox

matplotlib.use("TkAgg")

roach = cd.initialize_roach(roach_ip, boffile=boffile, upload=True)
roach.write_int(acc_len_reg, acc_len)
roach.write_int(detector_gain_reg, detector_gain)
roach.write_int(cnt_rst_reg, 1)
roach.write_int(cnt_rst_reg, 0)
roach.write_int(adq_trigger_reg, 1)
roach.write_int(adq_trigger_reg, 0)

root = tk.Tk()
root.configure(bg='white')

fig = Figure(figsize=(16, 8), dpi=120)
```

```
fig.set_tight_layout('True')
ax1 = fig.add_subplot(321)
ax2 = fig.add_subplot(322)
ax3 = fig.add_subplot(323)
ax4 = fig.add_subplot(324)
ax5 = fig.add_subplot(325)
ax6 = fig.add_subplot(326)
# ax7 = fig.add_subplot(427)
axes = [ax1, ax2, ax3, ax4, ax5, ax6]
titles = ["Primary signal",
          "Reference signal",
          "Cross-Power Spectral Density",
          "Power Spectral Densities Multiplied",
          "Channel scores",
          "Channel scores sum"]
          # "Score derivative"]
lines = []
lines_full = []
t = []
scoresum = []
score_der_last = np.zeros(nchannels - 1)


def add_reg_entry(roach, root, reg):
    frame = tk.Frame(master=root, bg="white")
    frame.pack(side=tk.TOP, anchor="w")
    label = tk.Label(frame, text=reg + ":", bg="white")
    label.pack(side=tk.LEFT)
    entry = tk.Entry(frame, bg="white")
    entry.insert(tk.END, roach.read_uint(reg))
    entry.pack(side=tk.LEFT)
    button_double = tk.Button(frame, text='x2', command=lambda: reg_double(), bg="
    ↪ white")
    button_double.pack(side=tk.LEFT)
    button_half = tk.Button(frame, text='/2', command=lambda: reg_half(), bg="white")
    button_half.pack(side=tk.LEFT)
    button_add = tk.Button(frame, text='+1', command=lambda: reg_add(), bg="white")
    button_add.pack(side=tk.LEFT)
    button_sub = tk.Button(frame, text='-1', command=lambda: reg_subtract(), bg="
    ↪ white")
    button_sub.pack(side=tk.LEFT)

    def reg_double():
        val = int(numexpr.evaluate(entry.get())) * 2
        entry.delete(0, "end")
        entry.insert(0, val)
        roach.write_int(reg, val)
        roach.write_int(cnt_rst_reg, 1)
        roach.write_int(cnt_rst_reg, 0)

```

```
    def reg_half():
        val = int(numexpr.evaluate(entry.get())) / 2
        entry.delete(0, "end")
        entry.insert(0, val)
        roach.write_int(reg, val)
        roach.write_int(cnt_rst_reg, 1)
        roach.write_int(cnt_rst_reg, 0)

    def reg_add():
        val = int(numexpr.evaluate(entry.get())) + 1
        entry.delete(0, "end")
        entry.insert(0, val)
        roach.write_int(reg, val)
        roach.write_int(cnt_rst_reg, 1)
        roach.write_int(cnt_rst_reg, 0)

    def reg_subtract():
        val = int(numexpr.evaluate(entry.get())) - 1
        entry.delete(0, "end")
        entry.insert(0, val)
        roach.write_int(reg, val)
        roach.write_int(cnt_rst_reg, 1)
        roach.write_int(cnt_rst_reg, 0)


add_reg_entry(roach, root, acc_len_reg)
add_reg_entry(roach, root, detector_gain_reg)

# Define plots patches
patches = []
for i in range(0, 4):
    patches.append(pat.Rectangle((1200, 0), 600, 0, alpha=0.1, facecolor='red'))
    axes[i].add_patch(patches[i])

# Define plots lines
for ax in axes[:4]:
    line, = ax.plot([], [], 'r', lw=0.7, label='full bits')
    lines_full.append(line)
for ax in axes:
    line, = ax.plot([], [], 'c', lw=1.3, label='sliced')
    lines.append(line)
    # if ax != ax5 and ax != ax6 and ax != ax3 and ax != ax7:
    if ax != ax5 and ax != ax6 and ax != ax3:
        ax.legend()

# Place canvas of plots and toolbar
canvas = FigureCanvasTkAgg(fig, master=root)
canvas.draw()
canvas.get_tk_widget().pack(side=tk.TOP, fill=tk.BOTH, expand=1)
toolbar = NavigationToolbar2Tk(canvas, root)
```

```
toolbar.update()
canvas.get_tk_widget().pack(side=tk.TOP, fill=tk.BOTH, expand=1)


def init():
    # Initialize plots
    for ax, title in zip(axes, titles):
        ax.set_xlim(1200, bandwidth + 1200)
        ax.set_ylim(-dBFS - 2, 0)
        ax.set_xlabel('Frequency (MHz)')
        ax.set_ylabel('Power (dBFS)')
        ax.set_title(title)
        ax.grid()
    ax5.set_ylim(-0.2, 1.2)
    ax5.set_ylabel('Score')
    ax6.set_xlim(0, 30)
    ax6.set_xlabel('Time (s)')
    ax6.set_ylim(-100, nchannels + 100)
    ax6.set_ylabel('Sum score')
    # ax7.set_ylim(-1.2, 1.2)
    # ax7.set_ylabel('Score derivative')
    return lines


def run(i):
    # Update registers
    acc_len = roach.read_uint(acc_len_reg)
    detector_gain = roach.read_uint(detector_gain_reg)

    # Get spectrometers data
    specdata1 = cd.read_interleave_data(roach, specs_names[0], spec_addr_width, spec_word_width, spec_data_type)
    specdata2 = cd.read_interleave_data(roach, specs_names[1], spec_addr_width, spec_word_width, spec_data_type)
    specdata1 = np.delete(specdata1, len(specdata1) / 2)
    specdata2 = np.delete(specdata2, len(specdata2) / 2)

    # Get spectrometer sliced data
    pow_factor = pwr_sliced_bits - detector_gain
    specdata_sl1 = cd.read_interleave_data(roach, specs_sl_names[0], score_addr_width, score_word_width,
                                            score_data_type) * (2 ** (pow_factor))
    specdata_sl2 = cd.read_interleave_data(roach, specs_sl_names[1], score_addr_width, score_word_width,
                                            score_data_type) * (2 ** (pow_factor))
    specdata_sl1 = np.delete(specdata_sl1, len(specdata1) / 2)
    specdata_sl2 = np.delete(specdata_sl2, len(specdata2) / 2)

    # Get numerator and denominator of RFI score
    numdata = cd.read_interleave_data(roach, score_names[0], score_addr_width,
```

```
    score_word_width,
                                            score_data_type) * (2 ** (pow_factor * 2 + 4))
    denomdata = cd.read_interleave_data(roach, score_names[1], score_addr_width, score_word_width,
                                            score_data_type) * (2 ** (pow_factor * 2 + 4))
    numdata = [math.sqrt(numdata[i]) for i in range(0, len(numdata))]
    numdata = np.asarray(numdata)
    numdata = np.delete(numdata, len(specdata1) / 2)
    denomdata = [math.sqrt(denomdata[i]) for i in range(0, len(denomdata))]
    denomdata = np.asarray(denomdata)
    denomdata = np.delete(denomdata, len(specdata2) / 2)

    # Get score data
    scoredata = cd.read_interleave_data(roach, score_names[2], score_addr_width, score_word_width,
                                            score_data_type) * 2 ** -30
    scoredata = np.delete(scoredata, len(specdata1) / 2)

    # Score derivative
    global score_der_last
    score_der = scoredata - score_der_last
    score_der_last = scoredata

    ## Save data
    # config = 'data/cfg1_'
    # filenames = ['specdata1.txt', 'specdata2.txt', 'specdata_sl1.txt', 'specdata_sl2.txt', 'numdata.txt',
    #               'denomdata.txt', 'scoredata.txt', 'timedata.txt']
    # data_array = [specdata1, specdata2, specdata_sl1, specdata_sl2, numdata, denomdata, scoredata, time.time()]
    # for filename, data in zip(filenames, data_array):
    #     f = open(config + filename, 'ab')
    #     np.savetxt(f, [data])
    #     f.close()

    # Normalize data by acc_len and convert to dBFS
    specdata1db = cd.scale_and_dBFS_specdata(specdata1, acc_len, dBFS)
    specdata2db = cd.scale_and_dBFS_specdata(specdata2, acc_len, dBFS)
    specdata_sl1db = cd.scale_and_dBFS_specdata(specdata_sl1, acc_len, dBFS)
    specdata_sl2db = cd.scale_and_dBFS_specdata(specdata_sl2, acc_len, dBFS)
    numdatadb = cd.scale_and_dBFS_specdata(numdata, acc_len, dBFS)
    denomdatadb = cd.scale_and_dBFS_specdata(denomdata, acc_len, dBFS)

    # Power Spectral Density full bits, the product and squared root are calculated in python
    multdatadb = [(specdata1db[j] + specdata2db[j]) / 2 for j in range(len(specdata1db))]

    # Add last score sum and time data
    t.append(time.time() - time_start)
    scoresum.append(np.sum(scoredata))
```

```python

    # Acquisition trigger of brams
    roach.write_int(adq_trigger_reg, 1)
    roach.write_int(adq_trigger_reg, 0)

    # Update fig lines
    lines [0]. set_data(freqs, specdata_sl1db)
    lines [1]. set_data(freqs, specdata_sl2db)
    lines [2]. set_data(freqs, numdatadb)
    lines [3]. set_data(freqs, denomdatadb)
    lines [4]. set_data(freqs, scoredata)
    lines [5]. set_data(t, scoresum)
    # lines [6]. set_data(freqs, score_der)
    lines_full [0]. set_data(freqs, specdata1db)
    lines_full [1]. set_data(freqs, specdata2db)
    lines_full [3]. set_data(freqs, multdatadb)


    # Update x-limits of plots  with time to see the last 30 seconds
    # if t [-1] > 30:
    #     ax6.set_xlim(t [-1] - 30, t [-1])

    # Update rectangle patches
    for i in range(0, len(patches)):
        if i < 2:
            y0 = 10 * np.log10(2 ** (pow_factor - np.log2(acc_len))) - dBFS
            height = 10 * np.log10(2 ** 18)
        else:
            y0 = 10 * np.log10(2 ** (pow_factor + 2 - np.log2(acc_len))) - dBFS
            height = 10 * np.log10(2 ** 16)

        patches[i ]. set_y(y0)
        patches[i ]. set_height(height)
    return lines


time_start = time.time()
ani = animation.FuncAnimation(fig, run, interval=10, init_func=init)
root.mainloop()
```

Código B.2: Detector parameters.

```python
# imports
import numpy as np

# communication parameters
roach_ip = '192.168.1.12'
boffile = 'rfidet_div.bof.gz'

# model parameters
```

```
adc_bits = 8
bandwidth = 600  # MHz
acc_len_reg = 'acc_len'
cnt_rst_reg = 'cnt_rst'
detector_gain_reg = 'detector_gain'
adq_trigger_reg = 'trigger'
spec_addr_width = 9  # bits
spec_word_width = 64  # bits
spec_data_type = '>u8'
score_addr_width = 9  # bits
score_word_width = 32  # bits
score_data_type = '>u4'

specs_names = [['dout0_0', 'dout0_1', 'dout0_2', 'dout0_3'],                    # Primary signal
               ['dout1_0', 'dout1_1', 'dout1_2', 'dout1_3']]                    # Reference signal

specs_sl_names = [['doutsl0_0', 'doutsl0_1',  'doutsl0_2',  'doutsl0_3'],      # Primary signal sliced
                  ['doutsl1_0',  'doutsl1_1',  'doutsl1_2',  'doutsl1_3']]      # Reference signal  sliced

score_names = [['dout_num_0', 'dout_num_1', 'dout_num_2', 'dout_num_3'],           # Power Spectral Density multiplied
               ['dout_denom_0', 'dout_denom_1', 'dout_denom_2', 'dout_denom_3'], # Cross-Power Spectral Density
                ['dout_score_0', 'dout_score_1', 'dout_score_2', 'dout_score_3']]   # Score

# experiment parameters
acc_len = 2 ** 12
detector_gain = 37
pwr_sliced_bits = 45

# derivative  parameters
nchannels = 2 ** spec_addr_width * len(specs_names[0])
freqs  = np.linspace(0,  bandwidth, nchannels, endpoint=False)  # MHz
freqs  = np.delete(freqs ,  len(freqs) / 2)
freqs  = [x+1200 for x in freqs]
dBFS = 6.02 * adc_bits + 1.76 + 10 * np.log10(nchannels)
```

Código B.3: Detector simulation.

```
import numpy as np
import matplotlib.pyplot as  plt

# Parameters
A = 10  # Amplitude
freq  = 200  # Signal frequency (Mhz)
```

```python
phi = np.pi / 3  # Offset angle
dataLen = 2 ** 12  # Size data
accLen = 2 ** 7  # Integration length
snr = 20  # SNR (dB)
fm = 1080.0  # Sampling rate (Mhz)

# Define time series and initialize
t = np.flip(np.linspace(dataLen / fm, 0, dataLen, endpoint=False), 0)
PSD1 = []
PSD2 = []
CPSD = []

# Add noise to inputs
for i in range(0, accLen):
    x1 = A * np.cos(2 * np.pi * t * freq)
    x2 = A * np.cos(2 * np.pi * t * freq + phi)
    # x2 = 0
    p1 = np.mean(np.abs(x1) ** 2)
    sigma_noise = np.sqrt(10 ** (np.log10(p1) - snr / 10))
    noise1 = np.random.normal(0, sigma_noise, dataLen)
    noise2 = np.random.normal(0, sigma_noise, dataLen)
    x1 = x1 + noise1
    x2 = x2 + noise2

    # FFT and frequency arrays
    f = np.fft.rfftfreq(dataLen, d=1 / fm)
    X1 = np.fft.rfft(x1)
    X2 = np.fft.rfft(x2)

    # Power and cross-correlation
    P1 = np.real(X1 * np.conj(X1))
    P2 = np.real(X2 * np.conj(X2))
    crosscor = X1 * np.conj(X2)
    PSD1 = np.append(PSD1, P1 / dataLen ** 2)
    PSD2 = np.append(PSD2, P2 / dataLen ** 2)
    CPSD = np.append(CPSD, crosscor / dataLen ** 2)

PSD1 = np.reshape(PSD1, (accLen, len(f)))
PSD2 = np.reshape(PSD2, (accLen, len(f)))
CPSD = np.reshape(CPSD, (accLen, len(f)))
PSD1mean = np.mean(PSD1, 0)
PSD2mean = np.mean(PSD2, 0)
CPSDmean = np.mean(CPSD, 0)

ylim = ((-80, 20))

# Plot signal 1 PSD
c = 10 * np.log10(PSD1[-1])
plt.subplot(4, 2, 1)
plt.plot(f, c)
```

```
plt.title("Main signal PSD")
plt.xlabel("Frequency (Mhz)")
plt.ylabel("Power (dB)")
plt.ylim(ylim)
plt.grid()
plt.xlim((0, fm / 2))

# Plot signal 2 PSD
d = 10 * np.log10(PSD2[-1])
plt.subplot(4, 2, 2)
plt.plot(f, d)
plt.title("Reference signal PSD")
plt.xlabel("Frequency (Mhz)")
plt.ylabel("Power (dB)")
plt.ylim(ylim)
plt.grid()
plt.xlim((0, fm / 2))

# Plot instantaneous CPSD
a = 10 * np.log10(np.abs(CPSD[-1]))
plt.subplot(4, 1, 2)
plt.plot(f, a)
plt.title("CPSD without integration")
plt.xlabel("Frequency (Mhz)")
plt.ylabel("Power (dB)")
plt.ylim(ylim)
plt.grid()
plt.xlim((0, fm / 2))

# Plot integrated CPSD module
b = 10 * np.log10(np.abs(CPSDmean))
plt.subplot(4, 2, 5)
plt.plot(f, b)
plt.title("CPSD module after integration")
plt.xlabel("Frequency (Mhz)")
plt.ylabel("Power (dB)")
plt.ylim(ylim)
plt.grid()
plt.xlim((0, fm / 2))

# Plot CPSD integrated power
e = 10 * np.log10(np.mean(np.abs(CPSD), 0))
plt.subplot(4, 2, 6)
plt.plot(f, e)
plt.title("CPSD module before integration")
plt.xlabel("Frequency (Mhz)")
plt.ylabel("Power (dB)")
plt.ylim(ylim)
plt.grid()
plt.xlim((0, fm / 2))
```

```

# Plot CPSD integrated power
e = np.abs(CPSDmean) ** 2 / (PSD1mean * PSD2mean)
plt.subplot(4, 2, 7)
plt.plot(f, e)
plt.title("CPSD Score")
plt.xlabel("Frequency (Mhz)")
plt.ylabel("Score")
plt.ylim((-0.2,1.2))
plt.grid()
plt.xlim((0, fm / 2))

print("Signal 1 Power")
print("     -Theoretical value: " + str(A ** 2 / 2 + sigma_noise ** 2))
print("     -Time density integration: " + str(np.mean(x1 ** 2)))
print("     -Frequency density integration: " + str(2 * np.sum(np.abs(X1) ** 2) / dataLen
    ↪ ** 2))

plt.gcf().suptitle("Integration size: " + str(accLen))
plt.gcf().set_size_inches(14.5, 7.5)
plt.tight_layout()
plt.show()
```

Código B.4: Classificator main script.

```
import matplotlib
import math
from sklearn.decomposition import PCA
from sklearn import preprocessing
from sklearn.manifold import TSNE
from sklearn.cluster import KMeans
from mpldatacursor import HighlightingDataCursor, DataCursor
import calandigital as cd
from sklearn import preprocessing


import scipy.stats
from scipy import stats
import numpy as np
import pandas as pd
import time
from matplotlib import patches as pat
from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg, NavigationToolbar2Tk
from matplotlib.figure import Figure
import Tkinter as tk
from matplotlib import animation
from detector_parameters import *
import calandigital as cd
import matplotlib.pyplot as plt

```

```python
matplotlib.use("TkAgg")
# score_der_last = np.zeros(nchannels - 1)

config = 'data/cfg3_'
filenames = ['specdata_sl1.txt', 'specdata_sl2.txt', 'numdata.txt', 'scoredata.txt']
cdict = {0: 'red', 1: 'blue', 2: 'green', 3: 'cyan', 4: 'black'}
ldict = {0: 'No detection', 1: 'Detection narrowband', 2: 'Detection broadband'}
xlabels = ['Principal Component 1', 'Principal Component 1', 'Main Feature 1', 'Main Feature 1']
ylabels = ['Principal Component 2', 'Principal Component 2', 'Main Feature 1', 'Main Feature 1']
titles = ['PCA with K-Means', 'PCA with score, NB & WB decision', 't-SNE with K-Means ', 't-SNE with score, NB & WB decision']
colors = []
files = []
score = []
mean = []
var = []
skew = []
kurt = []

scoredata = pd.read_csv(config+'scoredata.txt', delimiter=' ', header=None)
specdata = pd.read_csv(config+'specdata_sl1.txt', delimiter=' ', header=None) / 2**10

for i in scoredata.T:
    row = scoredata.T[i][1:]
    score.append(row.max())

temp = np.mean(scoredata.T[0][1:])
for i in range(0, len(scoredata)):
    if score[i] > 0.5:
        colors.append(1)
        # if np.mean(scoredata.T[i][1:]) >= temp * 1.3:
        #     colors.append(2)
        # else:
        #     colors.append(1)
    else:
        colors.append(0)
    temp = np.mean(scoredata.T[i][1:])

# ind = 2764
# specdata = np.loadtxt(config+filenames[3], skiprows=ind, max_rows=1)
# freqs = np.linspace(0, bandwidth, nchannels, endpoint=False)  # MHz
# freqs = np.delete(freqs, len(freqs) / 2)
# freqs = [x + 1200 for x in freqs]
# # specdata = cd.scale_and_dBFS_specdata(specdata, acc_len, dBFS)
#
# plt.plot(freqs, specdata, c=cdict[colors[ind]])
# plt.ylim([0,1])
# plt.show()
```

```
stats = stats.describe(specdata, axis=1)
stats = np.stack([stats[1][0], stats[1][1], stats[2], stats[3], stats[4], stats[5]], axis=1)
scaled_stats = preprocessing.scale(stats, axis=0)

#PCA
pca = PCA()
pca.fit(scaled_stats)
pca_stats = pca.transform(scaled_stats)

# # #t-SNE
tsne = TSNE(learning_rate=50)
tsne_stats = tsne.fit_transform(scaled_stats)

#K-Means
km = KMeans(n_clusters=3, max_iter=3000)
km.fit(scaled_stats)
km_stats = km.predict(scaled_stats)

fig, ((ax1, ax2), (ax3, ax4))= plt.subplots(2,2)
fig.set_size_inches(18.5, 10.5, forward=True)
fig.set_tight_layout('True')
axes = [ax1, ax2, ax3, ax4]

for ax, xlabel, ylabel, title in zip(axes,xlabels,ylabels, titles):
    ax.set_xlabel(xlabel)
    ax.set_ylabel(ylabel)
    ax.set_title(title)

for i in np.unique(km.labels_):
    ix = np.where(km.labels_ == i)
    ax1.scatter(pca_stats[ix,0], pca_stats[ix,1], c=cdict[i], s=20, picker=True)
    ax3.scatter(tsne_stats[ix, 0], tsne_stats[ix, 1], c=cdict[i], s=20, picker=True)

classindex_color = []
for i in np.unique(colors):
    ix = np.where(colors == i)
    classindex_color.append(ix)
    ax2.scatter(pca_stats[ix,0], pca_stats[ix,1], c=cdict[i], label=ldict[i], s=20, picker=True)
    ax4.scatter(tsne_stats[ix, 0], tsne_stats[ix, 1], c=cdict[i], label=ldict[i], s=20, picker=True)


def onpick1(event):
    ind = event.ind
    xdata = event.artist.get_label()
    print(ind, xdata)
```

```
    for i in range(len(ldict)):
        if xdata == ldict[i]:
            ind = classindex_color[i][0][ind]
    if len(ind) == 1:
        specdata = np.loadtxt(config+filenames[0], skiprows=ind, max_rows=1)
        specdata2 = np.loadtxt(config + filenames[1], skiprows=ind, max_rows=1)
        scoredata2 = np.loadtxt(config + filenames[3], skiprows=ind, max_rows=1)
        freqs = np.linspace(0, bandwidth, nchannels, endpoint=False) # MHz
        freqs = np.delete(freqs, len(freqs) / 2)
        freqs = [x + 1200 for x in freqs]
        specdata = cd.scale_and_dBFS_specdata(specdata, acc_len, dBFS)
        specdata2 = cd.scale_and_dBFS_specdata(specdata2, acc_len, dBFS)
        fig2, axs = plt.subplots(3,1)
        fig2.set_tight_layout('True')
        axs[0].plot(freqs, specdata)
        axs[1].plot(freqs, specdata2)
        axs[2].plot(freqs, scoredata2)
        titles = ['Main PSD', 'Reference PSD', 'Channel score']
        for ax, title in zip(axs, titles):
            ax.set_xlim(1200, 1800)
            ax.set_ylim(-dBFS - 2, 0)
            ax.set_xlabel('Frequency (MHz)')
            ax.set_ylabel('Power (dBFS)')
            ax.set_title(title)
            ax.grid()
        axs[2].set_ylim(0,1)
        ax.set_ylabel('Score')
        plt.show()
fig.canvas.mpl_connect('pick_event', onpick1)
plt.grid()
plt.legend()
plt.show()
```

Appendix C

ISE Design Suite Report

```
Release 14.7 par P.20131013 (lin64)
Copyright (c) 1995-2013 Xilinx, Inc.  All rights reserved.

dondani-ub::  Fri Jun 25 04:36:01 2021

par -w -mt 4 system_map.ncd system.ncd system.pcf


Constraints file: system.pcf.
Loading device for application Rf_Device from file '6vsx475t.nph' in environment
/opt/Xilinx/14.7/ISE_DS/ISE/:/opt/Xilinx/14.7/ISE_DS/EDK.
   "system" is an NCD, version 3.2, device xc6vsx475t, package ff1759, speed -1
vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv
INFO:Security:56 - Part 'xc6vsx475t' is not a WebPack part.
WARNING:Security:42 - Your software subscription period has lapsed. Your current version of Xilinx tools will continue
to function, but you no longer qualify for Xilinx software updates or new releases.

----------------------------------------------------------------------

Initializing temperature to 85.000 Celsius. (default - Range: 0.000 to 85.000 Celsius)
Initializing voltage to 0.950 Volts. (default - Range: 0.950 to 1.050 Volts)


Device speed data version:  "PRODUCTION 1.17 2013-10-13".



Device Utilization Summary:

Slice Logic Utilization:
  Number of Slice Registers:                84,515 out of 595,200   14%
    Number used as Flip Flops:              84,509
    Number used as Latches:                      2
    Number used as Latch-thrus:                  0
    Number used as AND/OR logics:                4
  Number of Slice LUTs:                     70,745 out of 297,600   23%
    Number used as logic:                   37,821 out of 297,600   12%
      Number using O6 output only:          26,935
      Number using O5 output only:           1,469
      Number using O5 and O6:                9,417
      Number used as ROM:                        0
    Number used as Memory:                  16,096 out of 122,240   13%
      Number used as Dual Port RAM:            712
        Number using O6 output only:            24
        Number using O5 output only:             2
        Number using O5 and O6:                686
      Number used as Single Port RAM:            0
      Number used as Shift Register:        15,384
        Number using O6 output only:        12,519
        Number using O5 output only:           433
        Number using O5 and O6:              2,432
    Number used exclusively as route-thrus: 16,828
      Number with same-slice register load:  5,664
      Number with same-slice carry load:    11,164
      Number with other load:                    0

Slice Logic Distribution:
  Number of occupied Slices:                23,973 out of  74,400   32%
  Number of LUT Flip Flop pairs used:       85,262
    Number with an unused Flip Flop:        18,141 out of  85,262   21%
    Number with an unused LUT:              14,517 out of  85,262   17%
    Number of fully used LUT-FF pairs:      52,604 out of  85,262   61%
    Number of slice register sites lost
      to control set restrictions:               0 out of 595,200    0%

  A LUT Flip Flop pair for this architecture represents one LUT paired with
  one Flip Flop within a slice.  A control set is a unique combination of
  clock, reset, set, and enable signals for a registered element.
  The Slice Logic Distribution report is not meaningful if the design is
  over-mapped for a non-slice resource or if Placement fails.
```

```
  OVERMAPPING of BRAM resources should be ignored if the design is
  over-mapped for a non-BRAM resource or if placement fails.

IO Utilization:
  Number of bonded IOBs:                   146 out of     840   17%
    Number of LOCed IOBs:                  146 out of     146  100%
    IOB Flip Flops:                         98

Specific Feature Utilization:
  Number of RAMB36E1/FIFO36E1s:             62 out of   1,064    5%
    Number using RAMB36E1 only:             62
    Number using FIFO36E1 only:              0
  Number of RAMB18E1/FIFO18E1s:            174 out of   2,128    8%
    Number using RAMB18E1 only:            174
    Number using FIFO18E1 only:              0
  Number of BUFG/BUFGCTRLs:                  6 out of      32   18%
    Number used as BUFGs:                    6
    Number used as BUFGCTRLs:                0
  Number of ILOGICE1/ISERDESE1s:            96 out of   1,080    8%
    Number used as ILOGICE1s:               64
    Number used as ISERDESE1s:              32
  Number of OLOGICE1/OSERDESE1s:            34 out of   1,080    3%
    Number used as OLOGICE1s:               34
    Number used as OSERDESE1s:               0
  Number of BSCANs:                          0 out of       4    0%
  Number of BUFHCEs:                         0 out of     216    0%
  Number of BUFIODQSs:                       0 out of     108    0%
  Number of BUFRs:                           1 out of      54    1%
    Number of LOCed BUFRs:                   1 out of       1  100%
  Number of CAPTUREs:                        0 out of       1    0%
  Number of DSP48E1s:                      388 out of   2,016   19%
  Number of EFUSE_USRs:                      0 out of       1    0%
  Number of FRAME_ECCs:                      0 out of       1    0%
  Number of GTXE1s:                          0 out of      36    0%
  Number of IBUFDS_GTXE1s:                   0 out of      18    0%
  Number of ICAPs:                           0 out of       2    0%
  Number of IDELAYCTRLs:                     2 out of      27    7%
  Number of IODELAYE1s:                     32 out of   1,080    2%
  Number of MMCM_ADVs:                       2 out of      18   11%
  Number of PCIE_2_0s:                       0 out of       2    0%
  Number of STARTUPs:                        1 out of       1  100%
  Number of SYSMONs:                         0 out of       1    0%
  Number of TEMAC_SINGLEs:                   0 out of       4    0%


Overall effort level (-ol):   Standard
Router effort level (-rl):    High

PAR will use up to 4 processors
WARNING:Par:288 - The signal rfidet_div_asiaa_adc5g0/rfidet_div_asiaa_adc5g0/FIFO/BU2/U0/grf.rf/mem/gdm.dm/Mram_RAM14_RAMD_D1_O has no load.
   PAR will not attempt to route this signal.
WARNING:Par:288 - The signal rfidet_div_asiaa_adc5g0/rfidet_div_asiaa_adc5g0/FIFO/BU2/U0/grf.rf/mem/gdm.dm/Mram_RAM15_RAMD_D1_O has no load.
   PAR will not attempt to route this signal.
WARNING:Par:288 - The signal rfidet_div_asiaa_adc5g0/rfidet_div_asiaa_adc5g0/FIFO/BU2/U0/grf.rf/mem/gdm.dm/Mram_RAM16_RAMD_D1_O has no load.
   PAR will not attempt to route this signal.
WARNING:Par:288 - The signal rfidet_div_asiaa_adc5g0/rfidet_div_asiaa_adc5g0/FIFO/BU2/U0/grf.rf/mem/gdm.dm/Mram_RAM13_RAMD_D1_O has no load.
   PAR will not attempt to route this signal.
WARNING:Par:288 - The signal rfidet_div_asiaa_adc5g0/rfidet_div_asiaa_adc5g0/FIFO/BU2/U0/grf.rf/mem/gdm.dm/Mram_RAM8_RAMD_D1_O has no load.
   PAR will not attempt to route this signal.
WARNING:Par:288 - The signal rfidet_div_asiaa_adc5g0/rfidet_div_asiaa_adc5g0/FIFO/BU2/U0/grf.rf/mem/gdm.dm/Mram_RAM17_RAMD_D1_O has no load.
   PAR will not attempt to route this signal.
WARNING:Par:288 - The signal rfidet_div_asiaa_adc5g0/rfidet_div_asiaa_adc5g0/FIFO/BU2/U0/grf.rf/mem/gdm.dm/Mram_RAM18_RAMD_D1_O has no load.
   PAR will not attempt to route this signal.
WARNING:Par:288 - The signal rfidet_div_asiaa_adc5g0/rfidet_div_asiaa_adc5g0/FIFO/BU2/U0/grf.rf/mem/gdm.dm/Mram_RAM19_RAMD_D1_O has no load.
   PAR will not attempt to route this signal.
WARNING:Par:288 - The signal rfidet_div_asiaa_adc5g0/rfidet_div_asiaa_adc5g0/FIFO/BU2/U0/grf.rf/mem/gdm.dm/Mram_RAM5_RAMD_D1_O has no load.
   PAR will not attempt to route this signal.
WARNING:Par:288 - The signal rfidet_div_asiaa_adc5g0/rfidet_div_asiaa_adc5g0/FIFO/BU2/U0/grf.rf/mem/gdm.dm/Mram_RAM20_RAMD_D1_O has no load.
   PAR will not attempt to route this signal.
WARNING:Par:288 - The signal rfidet_div_asiaa_adc5g0/rfidet_div_asiaa_adc5g0/FIFO/BU2/U0/grf.rf/mem/gdm.dm/Mram_RAM6_RAMD_D1_O has no load.
   PAR will not attempt to route this signal.
WARNING:Par:288 - The signal rfidet_div_asiaa_adc5g0/rfidet_div_asiaa_adc5g0/FIFO/BU2/U0/grf.rf/mem/gdm.dm/Mram_RAM7_RAMD_D1_O has no load.
   PAR will not attempt to route this signal.
WARNING:Par:288 - The signal rfidet_div_asiaa_adc5g0/rfidet_div_asiaa_adc5g0/FIFO/BU2/U0/grf.rf/mem/gdm.dm/Mram_RAM4_RAMD_D1_O has no load.
   PAR will not attempt to route this signal.
WARNING:Par:288 - The signal rfidet_div_asiaa_adc5g0/rfidet_div_asiaa_adc5g0/FIFO/BU2/U0/grf.rf/mem/gdm.dm/Mram_RAM9_RAMD_D1_O has no load.
   PAR will not attempt to route this signal.
WARNING:Par:288 - The signal rfidet_div_asiaa_adc5g0/rfidet_div_asiaa_adc5g0/FIFO/BU2/U0/grf.rf/mem/gdm.dm/Mram_RAM22_RAMB_D1_DPO has no
   load.  PAR will not attempt to route this signal.
WARNING:Par:288 - The signal rfidet_div_asiaa_adc5g0/rfidet_div_asiaa_adc5g0/FIFO/BU2/U0/grf.rf/mem/gdm.dm/Mram_RAM22_RAMC_D1_DPO has no
   load.  PAR will not attempt to route this signal.
WARNING:Par:288 - The signal rfidet_div_asiaa_adc5g0/rfidet_div_asiaa_adc5g0/FIFO/BU2/U0/grf.rf/mem/gdm.dm/Mram_RAM22_RAMD_D1_O has no load.
   PAR will not attempt to route this signal.
WARNING:Par:288 - The signal rfidet_div_asiaa_adc5g0/rfidet_div_asiaa_adc5g0/FIFO/BU2/U0/grf.rf/mem/gdm.dm/Mram_RAM3_RAMD_D1_O has no load.
   PAR will not attempt to route this signal.
WARNING:Par:288 - The signal rfidet_div_asiaa_adc5g0/rfidet_div_asiaa_adc5g0/FIFO/BU2/U0/grf.rf/mem/gdm.dm/Mram_RAM21_RAMD_D1_O has no load.
   PAR will not attempt to route this signal.
WARNING:Par:288 - The signal rfidet_div_asiaa_adc5g0/rfidet_div_asiaa_adc5g0/FIFO/BU2/U0/grf.rf/mem/gdm.dm/Mram_RAM12_RAMD_D1_O has no load.
   PAR will not attempt to route this signal.
WARNING:Par:288 - The signal rfidet_div_asiaa_adc5g0/rfidet_div_asiaa_adc5g0/FIFO/BU2/U0/grf.rf/mem/gdm.dm/Mram_RAM10_RAMD_D1_O has no load.
   PAR will not attempt to route this signal.
WARNING:Par:288 - The signal rfidet_div_asiaa_adc5g0/rfidet_div_asiaa_adc5g0/FIFO/BU2/U0/grf.rf/mem/gdm.dm/Mram_RAM11_RAMD_D1_O has no load.
   PAR will not attempt to route this signal.
WARNING:Par:288 - The signal rfidet_div_asiaa_adc5g0/rfidet_div_asiaa_adc5g0/FIFO/BU2/U0/grf.rf/mem/gdm.dm/Mram_RAM1_RAMD_D1_O has no load.
   PAR will not attempt to route this signal.
WARNING:Par:288 - The signal rfidet_div_asiaa_adc5g0/rfidet_div_asiaa_adc5g0/FIFO/BU2/U0/grf.rf/mem/gdm.dm/Mram_RAM2_RAMD_D1_O has no load.
```

```
   PAR will not attempt to route this signal.
Starting Multi-threaded Router


Phase  1  : 444628 unrouted;       REAL time: 1 mins 24 secs 

Phase  2  : 264401 unrouted;       REAL time: 1 mins 39 secs 

Phase  3  : 79786 unrouted;       REAL time: 2 mins 22 secs 

Phase  4  : 79831 unrouted; (Setup:0, Hold:37850, Component Switching Limit:0)     REAL time: 2 mins 43 secs 

Updating file: system.ncd with current fully routed design.

Phase  5  : 0 unrouted; (Setup:0, Hold:32484, Component Switching Limit:0)     REAL time: 3 mins 41 secs 

Phase  6  : 0 unrouted; (Setup:0, Hold:32484, Component Switching Limit:0)     REAL time: 3 mins 41 secs 

Phase  7  : 0 unrouted; (Setup:0, Hold:32484, Component Switching Limit:0)     REAL time: 3 mins 41 secs 

Phase  8  : 0 unrouted; (Setup:0, Hold:32484, Component Switching Limit:0)     REAL time: 3 mins 41 secs 

Phase  9  : 0 unrouted; (Setup:0, Hold:0, Component Switching Limit:0)     REAL time: 3 mins 45 secs 

Phase 10  : 0 unrouted; (Setup:0, Hold:0, Component Switching Limit:0)     REAL time: 4 mins 5 secs 
Total REAL time to Router completion: 4 mins 5 secs 
Total CPU time to Router completion (all processors): 5 mins 57 secs 

Generating "PAR" statistics.

**************************
Generating Clock Report
**************************

+---------------------+--------------+------+------+------------+-------------+
|        Clock Net    |   Resource   |Locked|Fanout|Net Skew(ns)|Max Delay(ns)|
+---------------------+--------------+------+------+------------+-------------+
|            adc0_clk | BUFGCTRL_X0Y3| No   |20970 |  0.769     |  2.729      |
+---------------------+--------------+------+------+------------+-------------+
|          adc0_psclk |BUFGCTRL_X0Y31| No   | 1062 |  0.751     |  2.728      |
+---------------------+--------------+------+------+------------+-------------+
|             sys_clk | BUFGCTRL_X0Y2| No   |    4 |  0.005     |  2.264      |
+---------------------+--------------+------+------+------------+-------------+
|          adc0_clk90 | BUFGCTRL_X0Y4| No   |   32 |  0.148     |  2.350      |
+---------------------+--------------+------+------+------------+-------------+
|infrastructure_inst/ |              |      |      |            |             |
|             clk_200 | BUFGCTRL_X0Y0| No   |    2 |  0.127     |  2.180      |
+---------------------+--------------+------+------+------------+-------------+
|MMCM_PHASE_CALIBRATI |              |      |      |            |             |
|ON_ML_LUT2_444_ML_NE |              |      |      |            |             |
|               W_CLK |         Local|      |    2 |  0.000     |  0.472      |
+---------------------+--------------+------+------+------------+-------------+
|rfidet_div_asiaa_adc |              |      |      |            |             |
|5g0/rfidet_div_asiaa |              |      |      |            |             |
|_adc5g0/MMCM0_ML_NEW |              |      |      |            |             |
|                 _I1 |         Local|      |    3 |  0.000     |  1.159      |
+---------------------+--------------+------+------+------------+-------------+
|MMCM_PHASE_CALIBRATI |              |      |      |            |             |
|ON_ML_LUT2_436_ML_NE |              |      |      |            |             |
|               W_CLK |         Local|      |    3 |  0.409     |  0.755      |
+---------------------+--------------+------+------+------------+-------------+
|infrastructure_inst/ |              |      |      |            |             |
|infrastructure_inst/ |              |      |      |            |             |
|MMCM_BASE_sys_clk_ML |              |      |      |            |             |
|             _NEW_I1 |         Local|      |    3 |  0.000     |  2.551      |
+---------------------+--------------+------+------+------------+-------------+
|rfidet_div_asiaa_adc |              |      |      |            |             |
|5g0/rfidet_div_asiaa |              |      |      |            |             |
|_adc5g0/MMCM0_ML_NEW |              |      |      |            |             |
|                _OUT |         Local|      |    2 |  0.000     |  0.470      |
+---------------------+--------------+------+------+------------+-------------+
|infrastructure_inst/ |              |      |      |            |             |
|infrastructure_inst/ |              |      |      |            |             |
|MMCM_BASE_sys_clk_ML |              |      |      |            |             |
|            _NEW_OUT |         Local|      |    2 |  0.000     |  0.358      |
+---------------------+--------------+------+------+------------+-------------+

* Net Skew is the difference between the minimum and maximum routing
only delays for the net. Note this is different from Clock Skew which
is reported in TRCE timing report. Clock Skew is the difference between
the minimum and maximum path delays which includes logic delays.

* The fanout is the number of component pins not the individual BEL loads,
for example SLICE loads not FF loads.

Timing Score: 0 (Setup: 0, Hold: 0, Component Switching Limit: 0)

Number of Timing Constraints that were not applied: 6

Asterisk (*) preceding a constraint indicates it was not met.
   This may be due to a setup or hold violation.

----------------------------------------------------------------------------------------------------------
  Constraint                                |    Check    | Worst Case |  Best Case | Timing |   Timing   
                                            |             |     Slack  | Achievable | Errors |    Score   
----------------------------------------------------------------------------------------------------------
```

```
  PERIOD analysis for net "rfidet_div_asiaa | SETUP       |     0.007ns|     6.659ns|       0|           0
  _adc5g0/rfidet_div_asiaa_adc5g0/mmcm_clko | HOLD        |     0.002ns|            |       0|           0
  ut1" derived from  PERIOD analysis for ne |             |            |            |        |
  t "rfidet_div_asiaa_adc5g0/rfidet_div_asi |             |            |            |        |
  aa_adc5g0/adc_clk_div" derived from NET " |             |            |            |        |
  rfidet_div_asiaa_adc5g0/rfidet_div_asiaa_ |             |            |            |        |
  adc5g0/adc_clk" PERIOD = 3.3333        ns |             |            |            |        |
   HIGH 50%                                 |             |            |            |        |
----------------------------------------------------------------------------------------------------------
  TS_epb_clk_in = PERIOD TIMEGRP "epb_clk_i | SETUP       |     0.036ns|    14.889ns|       0|           0
  n" 67 MHz HIGH 50%                        | HOLD        |     0.058ns|            |       0|           0
----------------------------------------------------------------------------------------------------------
  TS_infrastructure_inst_infrastructure_ins | MINPERIOD   |     0.239ns|     4.761ns|       0|           0
  t_clk_200_mmcm = PERIOD TIMEGRP         " |             |            |            |        |
  infrastructure_inst_infrastructure_inst_c |             |            |            |        |
  lk_200_mmcm" TS_sys_clk_n *         2 HIG |             |            |            |        |
  H 50%                                     |             |            |            |        |
----------------------------------------------------------------------------------------------------------
  NET "rfidet_div_asiaa_adc5g0/rfidet_div_a | MINPERIOD   |     1.667ns|     1.666ns|       0|           0
  siaa_adc5g0/adc_clk" PERIOD = 3.3333      |             |            |            |        |
     ns HIGH 50%                            |             |            |            |        |
----------------------------------------------------------------------------------------------------------
  TS_sys_clk_n = PERIOD TIMEGRP "sys_clk_n" | MINLOWPULSE |     6.000ns|     4.000ns|       0|           0
   100 MHz HIGH 50%                         |             |            |            |        |
----------------------------------------------------------------------------------------------------------
  PERIOD analysis for net "rfidet_div_asiaa | MINLOWPULSE |     3.666ns|     3.000ns|       0|           0
  _adc5g0/rfidet_div_asiaa_adc5g0/adc_clk_d |             |            |            |        |
  iv" derived from  NET "rfidet_div_asiaa_a |             |            |            |        |
  dc5g0/rfidet_div_asiaa_adc5g0/adc_clk" PE |             |            |            |        |
  RIOD = 3.3333        ns HIGH 50%          |             |            |            |        |
----------------------------------------------------------------------------------------------------------
  PERIOD analysis for net "rfidet_div_asiaa | MINPERIOD   |     5.237ns|     1.429ns|       0|           0
  _adc5g0/rfidet_div_asiaa_adc5g0/mmcm_clko |             |            |            |        |
  ut2" derived from  PERIOD analysis for ne |             |            |            |        |
  t "rfidet_div_asiaa_adc5g0/rfidet_div_asi |             |            |            |        |
  aa_adc5g0/adc_clk_div" derived from NET " |             |            |            |        |
  rfidet_div_asiaa_adc5g0/rfidet_div_asiaa_ |             |            |            |        |
  adc5g0/adc_clk" PERIOD = 3.3333        ns |             |            |            |        |
   HIGH 50%                                 |             |            |            |        |
----------------------------------------------------------------------------------------------------------
  TS_infrastructure_inst_infrastructure_ins | SETUP       |     7.225ns|     2.775ns|       0|           0
  t_sys_clk_mmcm = PERIOD TIMEGRP         " | HOLD        |     0.108ns|            |       0|           0
  infrastructure_inst_infrastructure_inst_s |             |            |            |        |
  ys_clk_mmcm" TS_sys_clk_n         HIGH 50 |             |            |            |        |
  %                                         |             |            |            |        |
----------------------------------------------------------------------------------------------------------
  TS_sys_clk = PERIOD TIMEGRP "TNM_sys_clk" | MINHIGHPULSE|     9.168ns|     0.832ns|       0|           0
   100 MHz HIGH 50%                         |             |            |            |        |
----------------------------------------------------------------------------------------------------------


Derived Constraint Report
Review Timing Report for more details on the following derived constraints.
To create a Timing Report, run "trce -v 12 -fastpaths -o design_timing_report design.ncd design.pcf"
or "Run Timing Analysis" from Timing Analyzer (timingan).
Derived Constraints for rfidet_div_asiaa_adc5g0/rfidet_div_asiaa_adc5g0/adc_clk
+-------------------------------+-------------+-------------+-------------+-------------+-------------+-------------+-------------+
|                               |   Period    |       Actual Period       |      Timing Errors        |      Paths Analyzed       |
|           Constraint          | Requirement |-------------+-------------|-------------+-------------|-------------+-------------|
|                               |             |   Direct    | Derivative  |   Direct    | Derivative  |   Direct    | Derivative  |
+-------------------------------+-------------+-------------+-------------+-------------+-------------+-------------+-------------+
|rfidet_div_asiaa_adc5g0/rfidet_|      3.333ns|      1.666ns|      3.329ns|            0|            0|            0|      2020847|
|div_asiaa_adc5g0/adc_clk       |             |             |             |             |             |             |             |
| rfidet_div_asiaa_adc5g0/rfidet|      6.667ns|      3.000ns|      6.659ns|            0|            0|            0|      2020847|
| _div_asiaa_adc5g0/adc_clk_div |             |             |             |             |             |             |             |
|  rfidet_div_asiaa_adc5g0/rfide|      6.667ns|      1.429ns|          N/A|            0|            0|            0|            0|
|  t_div_asiaa_adc5g0/mmcm_clkou|             |             |             |             |             |             |             |
|  t2                           |             |             |             |             |             |             |             |
|  rfidet_div_asiaa_adc5g0/rfide|      6.667ns|      6.659ns|          N/A|            0|            0|      2020847|            0|
|  t_div_asiaa_adc5g0/mmcm_clkou|             |             |             |             |             |             |             |
|  t1                           |             |             |             |             |             |             |             |
+-------------------------------+-------------+-------------+-------------+-------------+-------------+-------------+-------------+

Derived Constraints for TS_sys_clk_n
+-------------------------------+-------------+-------------+-------------+-------------+-------------+-------------+-------------+
|                               |   Period    |       Actual Period       |      Timing Errors        |      Paths Analyzed       |
|           Constraint          | Requirement |-------------+-------------|-------------+-------------|-------------+-------------|
|                               |             |   Direct    | Derivative  |   Direct    | Derivative  |   Direct    | Derivative  |
+-------------------------------+-------------+-------------+-------------+-------------+-------------+-------------+-------------+
|TS_sys_clk_n                   |     10.000ns|      4.000ns|      9.522ns|            0|            0|            0|          186|
| TS_infrastructure_inst_infrast|     10.000ns|      2.775ns|          N/A|            0|            0|          186|            0|
| ructure_inst_sys_clk_mmcm     |             |             |             |             |             |             |             |
| TS_infrastructure_inst_infrast|      5.000ns|      4.761ns|          N/A|            0|            0|            0|            0|
| ructure_inst_clk_200_mmcm     |             |             |             |             |             |             |             |
+-------------------------------+-------------+-------------+-------------+-------------+-------------+-------------+-------------+

All constraints were met.


Generating Pad Report.

All signals are completely routed.

WARNING:Par:283 - There are 24 loadless signals in this design. This design will cause Bitgen to issue DRC warnings.

Total REAL time to PAR completion: 4 mins 29 secs
```

Total CPU time to PAR completion (all processors): 6 mins 22 secs

Peak Memory Usage: 3575 MB

Placer: Placement generated during map.
Routing: Completed - No errors found.
Timing: Completed - No errors found.

Number of error messages: 0
Number of warning messages: 26
Number of info messages: 0

Writing design to file system.ncd

PAR done!