VARIABLE BANDWIDTH POLYPHASE FILTER BANKS

A Thesis

Presented to the

Faculty of

San Diego State University

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

in

Electrical Engineering

---

by

Behrokh Farzad

Spring 2014

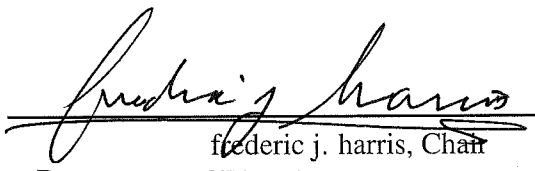
ProQuest LLC.
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106 - 1346

SAN DIEGO STATE UNIVERSITY

The Undersigned Faculty Committee Approves the

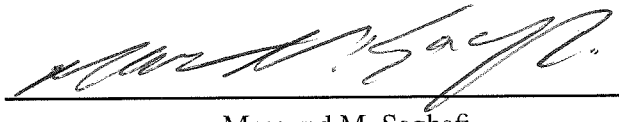
Thesis of Behrokh Farzad:

Variable Bandwidth Polyphase Filter Banks



frederic j. harris, Chair
Department of Electrical and Computer Engineering

Santosh Nagaraj
Department of Electrical and Computer Engineering



Massoud M. Saghafi
College of Business Administration

10/4/13

Approval Date

DEDICATION

Dedicated to my parents, *Farideh and Bahram*.

We must know, we shall know.

– David Hilbert

ABSTRACT OF THE THESIS

Variable Bandwidth Polyphase Filter Banks
by
Behrokh Farzad
Master of Science in Electrical Engineering
San Diego State University, 2014

This Thesis is a proposal for a variable bandwidth filter banks using analysis and synthesis filters that is based on polyphase channelizer architecture. Details regarding polyphase configuration and architecture of the up and down converter channelizer will be presented. First, a polyphase channelizer was used to design a variable bandwidth filter banks. Such approach is required for designing a Software Defined Radio receiver and transmitter. Second, the structure of the variable bandwidth was modified to enhance the resolution. The finer resolution will help to improve the quality of extracting the channel of interest from a mixture of different channels. Details regarding our proposed cascaded M-path channelizer filter will be presented. The first stage of the filter will recover the coarse channel while the second stage will focus on the recovery of finer bandwidth. Analysis results of a successful recovery for different signals with different bandwidths will be demonstrated. Multiple configurations of the proposed filter will be studied and simulation results will be presented.

TABLE OF CONTENTS

LIST OF TABLES

PAGE

LIST OF FIGURES

PAGE

ACKNOWLEDGMENTS

I would like to specially thank my adviser, Professor fredric j. harris for his guidance and inspiration during the course of my thesis.

CHAPTER 1
INTRODUCTION

A Software Defined Radio (SDR) system is a radio communication system, which is implemented by means of software or embedded computing device. Software Defined Radio have been significantly used in many applications such as military and cell phone services that require a wide variety of changing radio protocols. SDR requires filters of different bandwidths to process many signals with different bandwidth. Designing these types of filters has two main requirements [1]:

- Minimum time delay when filter switches between different bandwidths.
- Minimum power consumption and resource allocation.

Transition between two bandwidths will require computation and uploading new coefficients, which increase the time delay. On the other hand, filters that support multiple bandwidths require extra reservation of the resources such as memory. In this research, we try to propose a suitable filter to tackle these challenges.

1.1 MULTIRATE SIGNAL PROCESSING

Digital Signal Processing (DSP) is a mathematical process to represent and modify a digital signal. Analog signal processing directly modify an analog signal, where the digital signal processing steps are using sample values of an analog signal. One of the most important concepts in DSP is sampling process, that is the step when an analog signal is converted in to its corresponding sampled points. Since each sample contains the information of the original signal, the way that sampling process is conducted is critical. If we sample analog signal properly, it is possible to recover the continuous time version of the original signal from the sampled version without losing any information and with minimum error [2].

Multirate Digital Signal processing is the process of changing rates of samples, and the filter that performs the multirate processing is a Multirate Filter. Two important factors in designing a multirate filter are, first enhancing the performance of changing rates of samples, and second reducing the cost of implementation. An M-path polyphase filter was introduced to meet these two requirements. This type of filter is now adopted not only in many DSP applications but also in multiple communication systems as well.

1.2 SOFTWARE DEFINED RADIO

Software Defined Radio is a radio architecture in which all of its components such as filters, detectors, modulators, de-modulators, etc., are defined via software. The SDR unlike the Hardware Defined Radio (HDR) has a flexible configuration. The HDR structure is fixed, thus any changes in one or more components of HDR architecture results in changing the entire structure of radio. For example, if the bandwidth of the transmitting signal needs to be updated, the entire transmitter of the HDR has to be changed accordingly [3].

On the other hand, the SDR can be configured for different wave forms, bandwidths, and modes of operations dynamically. This implies that the SDR is a multi-functional radio, which can support a variety of services. One of the goals of the SDR is to support all of these services in a low-cost and power-efficient manner. The SDR was first proposed in 1991, by Joe Mitola [4]. In this thesis, we address the design architecture of the transmitter and receiver for the SDR.

1.3 PROPOSED TRANSMITTER FOR SDR

Digital transmitter main task is to conduct the filtering process and conversion of signal. The first step of any transmitter signal processing starts with up sampling the input signal to obtain a wider interval between the sampled data. Next step, we use a complex heterodyne to move the input data to an intermediate frequency. Finally, the digital to analog conversion is performed by transmitter. In a typical transmitter, we need to replicate the up sampling component for different signals. In this thesis, we discuss a technique to avoid this replication [5].

The proposed transmitter, which is a good candidate for SDR, is called Synthesis Channelizer. The architecture of this up converter channelizer is based on the standard M-path polyphase up converter channelizer. The difference between these two is that the standard up converter performs M-to-1 up sampling while the synthesis channelizer performs M-to-2 up sampling [6]. The synthesis channelizer architecture, same as the standard up converter channelizer consists of an IFFT block and an M-path polyphase filter block. This transmitter should be able to detect the total bandwidth available for communication medium dynamically, and constantly send the signals over the available bandwidth. Note that the input signals could have different bandwidths and center frequencies, and also could be generated from different sources. If the bandwidth of the input signal is wider than the synthesis filter, then the input signal will go first through a smaller down converter step. The down converter will divide the input signals to smaller sub-bands. These sub-bands will be de-fragmented without any loss of energy at the receiver. In Chapter 2, we explain the full design of the synthesis channelizer.

1.4 PROPOSED RECEIVER FOR SDR

A typical digital receiver conducts the filtering and signal conversion process similar to the digital transmitter. On top of that, the receiver block needs to estimate some unknown parameters of the input signal such as frequency and amplitude. The receiver first samples the received signal and then moves the sampled data to the base-band by using a down converter. In final step, it will adopt a matched filter to maximize the Signal-to-noise ratio (SNR). These types of digital receivers have the same issue as mentioned in Section 1.3. The problem is for down converting multiple input signals, we need to implement more down converter blocks. The proposed receiver for cognitive and SDR, is called Analysis Channelizer, and avoids this replication requirement. The architecture of the analysis filter is based on the standard M-path down converter channelizer. In a similar procedure, the analysis channelizer performs M-to-2 down sampling and shifts the received data to the base-band with aliasing [7]. It consists of a prototype filter block and an IFFT block. The analysis channelizer should be able to detect the multiple signals independent of their bandwidths and to process them without energy losses. In Chapter 2, we discuss the complete architecture design of the standard analysis channelizer.

1.5 THESIS CONTRIBUTIONS

This thesis studies analysis and synthesis filter banks based on channelizer architecture as a proposal for SDR variable bandwidth filter. Details regarding polyphase configuration and architecture of the channelizer up and down converter will be presented in details. During the course of this thesis, we propose a novel filter architecture for SDR which has both variable bandwidth and fixed configuration. Thus, it will enable integration of a low cost filter that can support multiple standards based on the requirement specifications. Second, to improve the performance of our filter we optimize the architecture such that it can support finer resolution of channel bandwidth extractions on top of the coarse bands. The finer resolution will help to improve the quality of extracting channel of the interest from mixture of different channels. Details regarding our proposed cascaded M-path channelizer filter will be presented. The first stage of our filter will recover the course channel where the second stage of our filter will focus on the recovery of finer channel. Our analysis results will demonstrate the successful performance of our filter for recovering different signals with minimum error. Multiple configurations of the proposed filter will be studied and simulation results will be presented.

The remainder of this thesis is organized as follows. We first devote Chapter 2 to explain how to design the standard M-path polyphase filter. The M-path polyphase filter structure is the starting point to introduce the architecture of the standard polyphase down converter and up converter. Chapter 3 of this research discuss about the design of the standard

M-path polyphase down and up converters, such that with some modifications these filters become suitable for cognitive and software defined radio. In Chapter 4, we introduce the architecture of our proposed variable bandwidth filter which is optimized for finer resolution channel extraction. Our analysis and simulation results will be explained in details that shows successful performance of our filter extraction. Finally, we will discuss the future directions for our research and conclude the summary of our results in Chapter 5.

CHAPTER 2
MULTIRATE FILTERS

The sampling process is one of the most important concepts in digital signal processing which connects the continuous time domain to discrete domain. The main goal in this process is to conduct the sampling process without losing information. Nyquist Sampling Criterion defines the proper sampling frequency as twice the signal frequency.

$$f_s \geqslant 2(BW_{signal}) \tag{2.1}$$

This assures that the periodic spectral replicas will not overlap each other and cause aliasing. If aliasing happens, there is almost no chance to recover the original signal from the samples. As mentioned above, we want to conduct the sampling process in a way that no information lost. One of the keys of avoiding losing information is choosing the correct sampling rate [2]. In many DSP applications we need to change the sampling rate many times during the process. The process of changing the sampling rate is called resampling, and multirate filters performs the process of increasing and decreasing the sample rate. There are two major resampling process, one is down sampling and the other one is up sampling as illustrated in Figure 2.1. The simplest multirate filters can perform up sampling of 1-to-P or down sampling of Q-to-1.

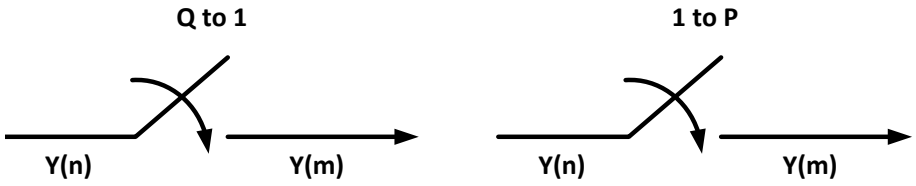


Figure 2.1. Symbols of Q-to-1 down sampling and 1-to-P up sampling.

In the down sampling process a subset of input samples will be set to zero. On the other hand, the process of inserting zeros between existing input sample values is called up sampling. In the down sampling procedure, we process the input signal $x(n)$ by either a low-pass or band-pass filter to obtain a reduced bandwidth output signal. Next step is to adjust

the sample rate with the current bandwidth by using a Q-to-1 down sampler as shown in Figure 2.2.

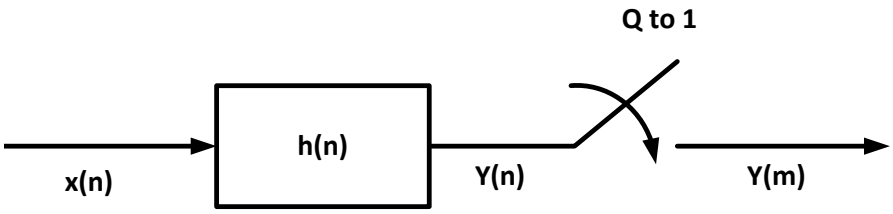


Figure 2.2. Block diagram of Q-to-1 down sampling process before filtering.

Also for the up sampling procedure, we have to follow two steps, as shown in Figure 2.3. One is sending the input signal into a 1-to-p up sampler to obtain a higher sample rate, and then send the output of the resampler into a filter $h(n)$. The filter will process the signal and discard the zero valued samples. The output of the filter has the same spectrum as the input signal but sampled at a higher sampling rate.

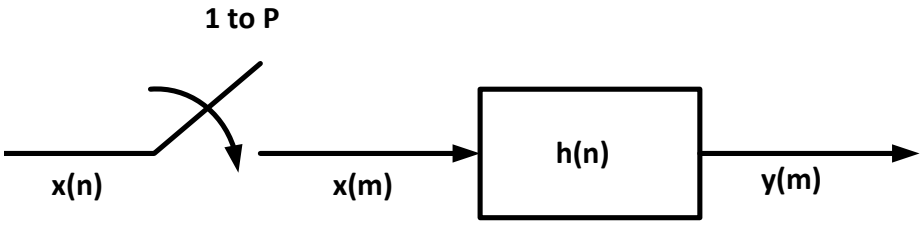


Figure 2.3. Block diagram of 1-to-P up sampling process before filtering.

2.1 STANDARD M-PATH DOWN CONVERTER CHANNELIZER (DEMODULATOR)

The goal of this section is showing how to down convert a single channel located in a multiple channel Frequency Division Multiplexed (FDM) input signal. FDM is a technique in which the total available bandwidth is divided into a series of sub-channels, and each of those channels will be used to send separate signals. The channels as shown in Figure 2.4, are equally spaced with equal bandwidths.

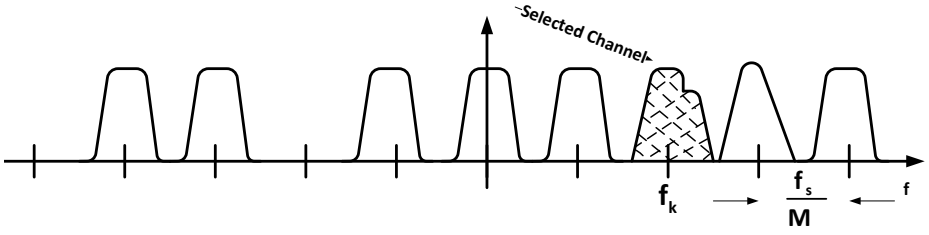


Figure 2.4. Frequency Division Multiplexing (FDM) input signal.

There is a standard process for down converting a selected channel. This standard process performs the down conversion of a selected band with a complex heterodyne, a low-pass filter and a down sampler, shown in Figures 2.5.

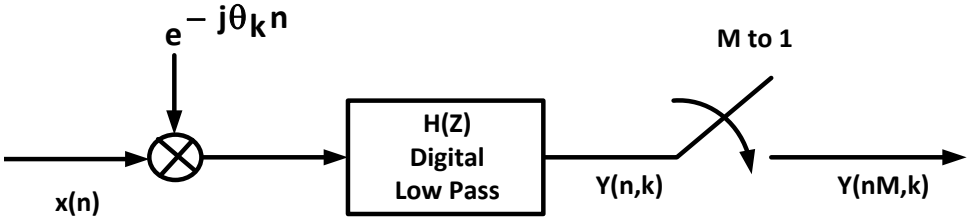


Figure 2.5. The diagram of a standard down converter proposed by Edwin Armstrong.

By rearranging the summation of the time series output from the down converter, we can show the Equivalency Theorem. This theorem allows us to substitute the down converter followed by a low-pass filter with a band-pass filter followed by a down converter [8]. As shown in Figure 2.6, the down sampler performs a sample rate reduction in a way that one samples remains in every M-samples.

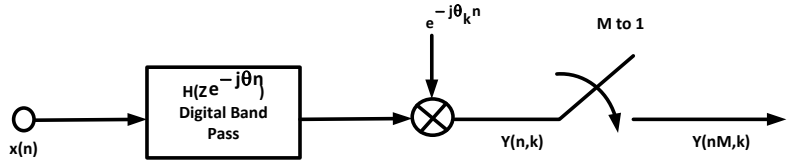


Figure 2.6. Block diagram of a down converter based on Equivalency Theorem.

Since down converting the samples which will be discarded in down sampling is redundant, therefor we can avoid this by bringing the heterodyne to the output of the resampler, as shown in Figure 2.7.

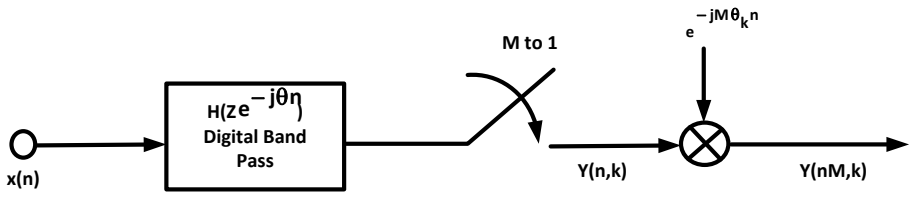


Figure 2.7. The diagram of a down converter using a band-pass filter.

The rotation rate of the input of the resampler is θ_k and rotation rate of the output of the complex heterodyne is $M\theta_k$. The main task of the heterodyne is to shift the center frequency of the desired channel to the DC frequency, and our next step will be finding a relation between these two rotation rates and if it was possible to implement the heterodyne inside the band-pass filter. If we choose $M\theta_k$ as a multiple of 2π, $M\theta_k = 2\pi * k$, or $\theta_k = 2\pi * k/M$, the rotation rate of the output of the band-pass filter will definitely alias to the base-band or zero frequency before sending to M-to-1 down sampler. Figure 2.8 shows this change in the band-pass filter of the down converter.

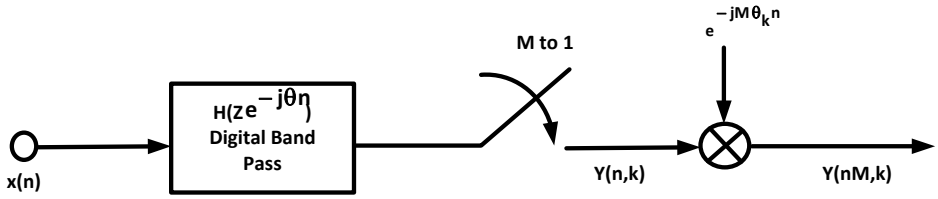


Figure 2.8. Base-band down converter with a down sampler.

There is one more step to take and optimize the structure of a down converter. All the input samples will be processed by a band-pass filter and will go through a down sampler. The down sampling will apply to every M sample. Here we notice that filtering the samples which will be discarded in the resampling process is redundant. The Noble Identity allows us to take the down sampler to the input of the band-pass filter. The output of a band-pass filter followed

by a down sampler is equal to the out put of a down sampler followed by a low-pass filter, As shown in Figure 2.9.

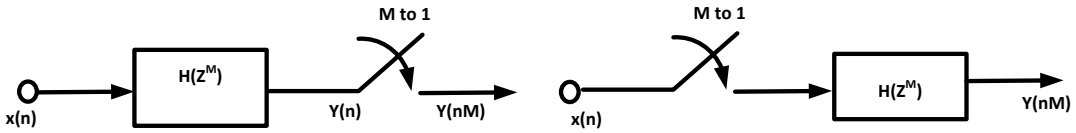


Figure 2.9. Noble identity states that a filter processing of every Mth sample followed by a M to 1 down sampler is the same as a M to 1 down sampler followed by a filter processing every input sample .

The only modification that needs to be done is to modify the filter in a way that only filter every M^{th} sample. We can address this issue by partitioning the filter to M parallel paths as follows in Equation 2.2 [9]:

$$\begin{aligned} H(Z) &= \sum_{N-1}^{n=0} h(n)Z^{-N} \\ &= h(0)+h(1)Z^{-1}+h(2)Z^{-2}+h(3)Z^{-3}+h(4)Z^{-4}+...+h(N-1)Z^{N-1} \\ &= h(0)+h(M+0)Z^{-M}+h(2M+0)Z^{-2M}+... \\ &\quad +h(1)Z^{-1}+h(M+1)Z^{-(M+1)}+h(2M+1)Z^{-(2M+1)}+... \\ &\quad +h(2)Z^{-2}+h(M+2)Z^{-(M+2)}+h(2M+2)Z^{-(2M+2)}+... \\ &\quad +h(3)Z^{-3}+h(M+3)Z^{-(M+3)}+h(2M+3)Z^{-(2M+3)}+... \\ &\quad +h(M-1)Z^{-(M-1)}+h(2M-1)Z^{-(2M-1)}+h(3M-1)Z^{-(3M-1)}+... \end{aligned} \tag{2.2}$$

This partition format converts a one-dimensional array to a two-dimensional array. Each row of this equation can be described as $Z^{-r}H_r(Z^M)$, so we can rewrite it in a compact form as bellow:

$$H(Z)=H_0(Z^M)+Z^{-1}H_1(Z^M)+Z^{-2}H_2(Z^M)+Z^{-3}H_3(Z^M)+...+Z^{M-1}H_{M-1}(Z^M) \tag{2.3}$$

or

$$H(Z)=\sum_{(r=0)}^{(M-1)} Z^{(-r)}H_r(Z^M)=\sum_{(r=0)}^{(M-1)} Z^{(-r)} \sum_{(n=0)}^{((N/M)-1)} h(r+nM)Z^{-nM} \tag{2.4}$$

Figure 2.10 shows the M-path partition of prototype low pass filter:

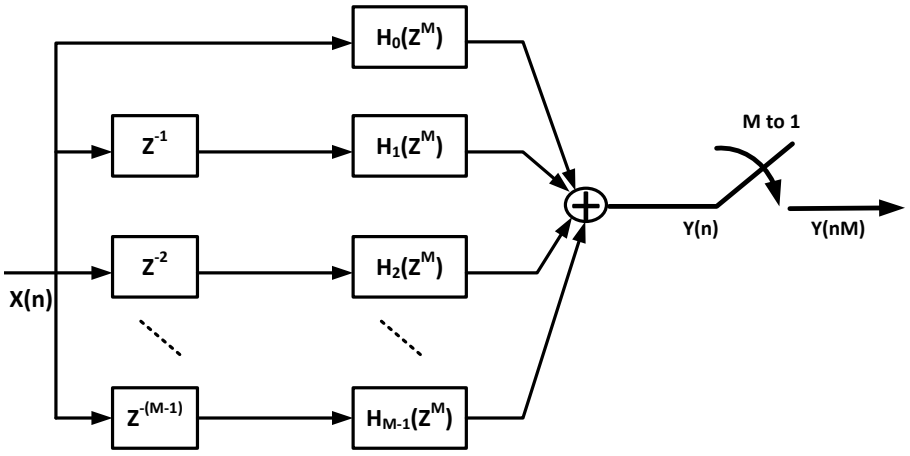


Figure 2.10. M-path partition of a low pass filter with outpt resampler .

We also can apply the Noble Identity here and pull the resampler to the input side of each filter partition. These resamplers all operate at the same time, and when the switches close, the input of the first row is the input signal, and the input of the second row is the same input with a delay, and so on. We can replace the resamplers and the delay lines with an input commutator that delivers M successive inputs to the partitions, as shown in Figure 2.11.

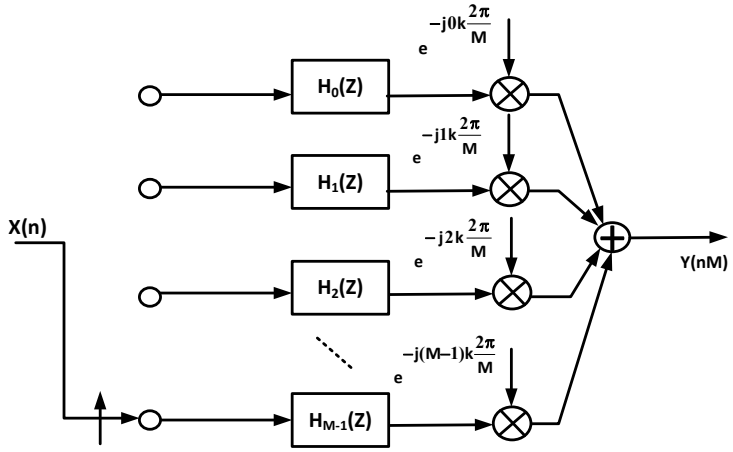


Figure 2.11. M-path partition of a lowpass filter with input commutator.

Next is converting the low pass filter to a band pass filter. If the low pass filter, $h(n)$, has a Z-transform of $H(Z)$, then Z-transform of a band-pass filter with a impulse response of

$h(n)e^{(+j\theta n)}$, will be $H(Ze^{(-j\theta n)})$. When we partition a filter to M stages, the channels at each stage experience a phase rotation due to their delays and center frequencies. When all the channels are summed together, the channels with the same phase values but different polarities will cancel each other. To prevent this to happen and losing data information, we can apply a phase rotation at each path. For a small number of the channel, like less than 10 channels, a set of phase rotators are sufficient to implement at the output of the M-path filter, as shown in Figure 2.12 [10].

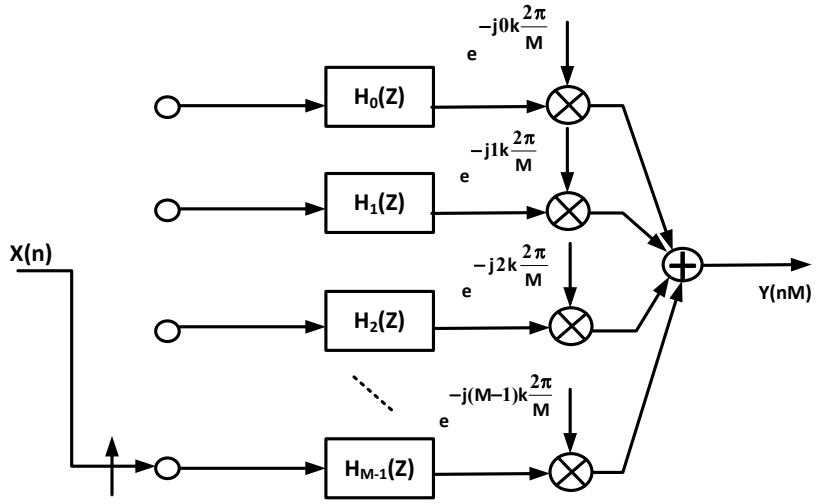


Figure 2.12. Resampling M-path down converter with phase rotator.

For a large number of the channel,$log_2 N$, we use IDFT. For computational efficiency, the IFFT is used instead of IDFT as shown in Figure 2.13.

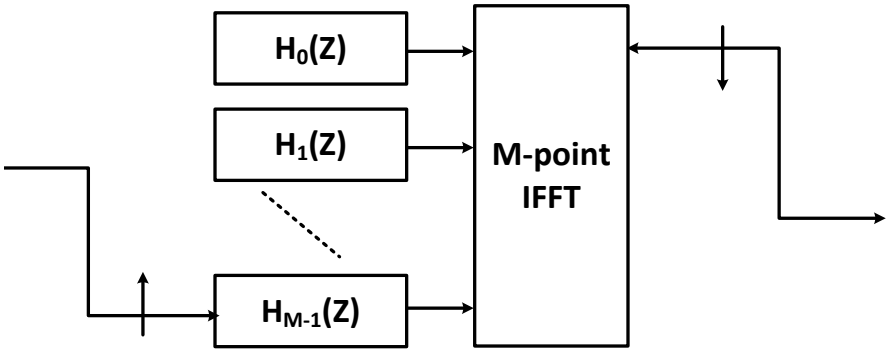


Figure 2.13. M-path down converter channelizer with IFFT.

2.2 STANDARD M-PATH UP CONVERTER CHANNELIZER (MODULATOR)

In this section, we will explain the structure of the standard up converter channelizer which consists of a 1-to-M up sampler followed by a low-pass filter. The 1-to-M up-sampler up samples the input signal by inserting zero values between the samples of the input signal. Inserting zero values not only decreases the interval between the samples of the input signal but also increase the sample rate by a rate of M. Note that the up-sampling process does not change the spectral contents of the input signal. To convert this basic structure to the standard up converter channelizer, we need to apply some modifications. We start with this fact that the zero values inserted by the up sampler contain no information about the original signal so there is no need to send them to low-pass filter. The way we can avoid processing these zero values is to track the location of the non-zero values. The M-path polyphase filter is able to keep track of the non-zero values, and only process them [11].

$$H(Z) = \sum_{(r=0)}^{(M-1)} Z^{(-r)} \sum_{(n=0)}^{((\frac{N}{M})-1)} h(r+nM)Z^{-nM} \tag{2.5}$$

$$H(Z) = \sum_{(r=0)}^{(M-1)} Z^{(-r)} H_r(Z^M) \tag{2.6}$$

Figure 2.14 shows the initial structure of the M-path up converter.

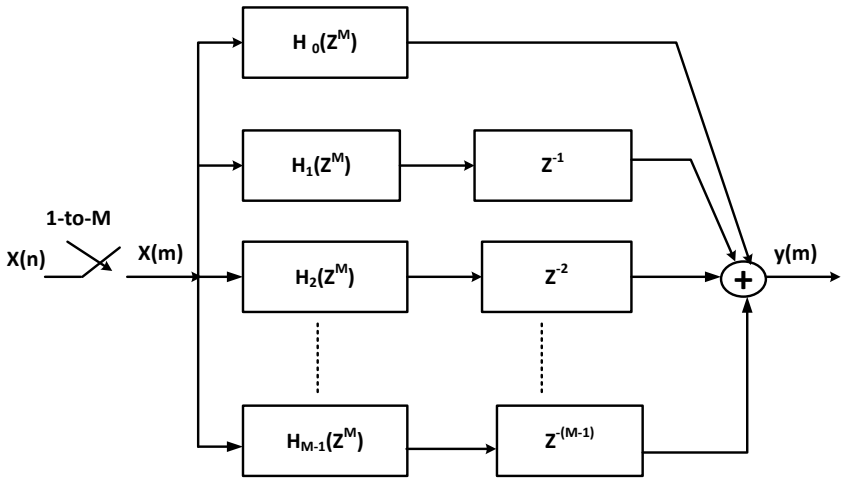


Figure 2.14. Initial diagram of M-path polyphase up converter.

Next modification will be applying the noble Identity theorem, and moving the up-sampler inside each arm of the M-path polyphase filter, and also replace the 1-to-M

up-sampler and the delays at each arm with an output commutator. This commutator will allow us to only sum the non-zero values at the summation junction, as shown in Figure 2.15.

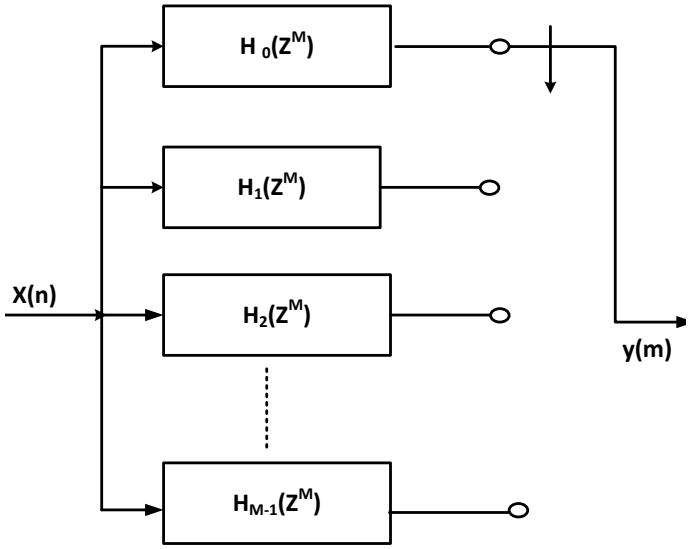


Figure 2.15. Standard structure of M-path polyphase interpolator.

Here we also need to replace the low-pass filter with a band-pass filter. The purpose of using the up sampler is to increase the sample rate by inserting zero values. This process will generate M spectral copies of the input signal. The low-pass filter will reject the spectral copies and only pass the spectral copy which is located at zero frequency to the up-converter. The up-converter will locate the remained spectral copy to the desired center frequency. Instead of doing this, we can use a band-pass filter which will only extract one of the spectral copies and these is no more need to translate this copy to other center frequencies. So far, we up sample and up convert the input signal to a particular center frequency by aliasing. Here, same as section 2.1, we need to apply a set of phase rotators in order to avoid channels with the same phase value but different polarities, to cancel each other. The phase rotators will be located at each arm as demonstrated in Figure 2.16. The input samples are passed through these phase rotators and then are sent to the M-path polyphase filters [9]. It is easier to replace all the phase rotators with a block of IFFT. The IFFT will apply a phase rotation to the samples. At the end, the output will have a new sampling rate but at a reduced bandwidth.

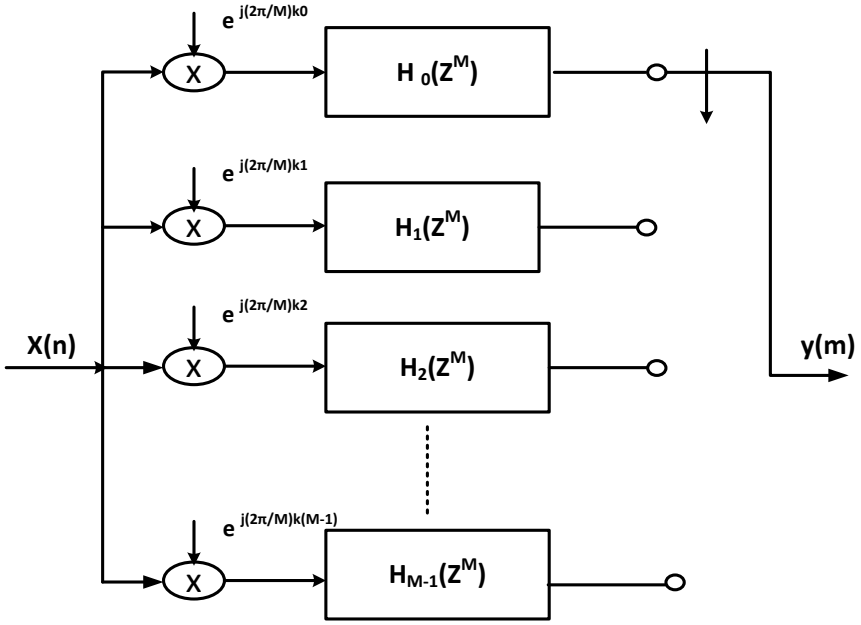


Figure 2.16. M-path polyphase interpolator with phase rotator.

Figure 2.17 represents the final structure of a M-path up converter.

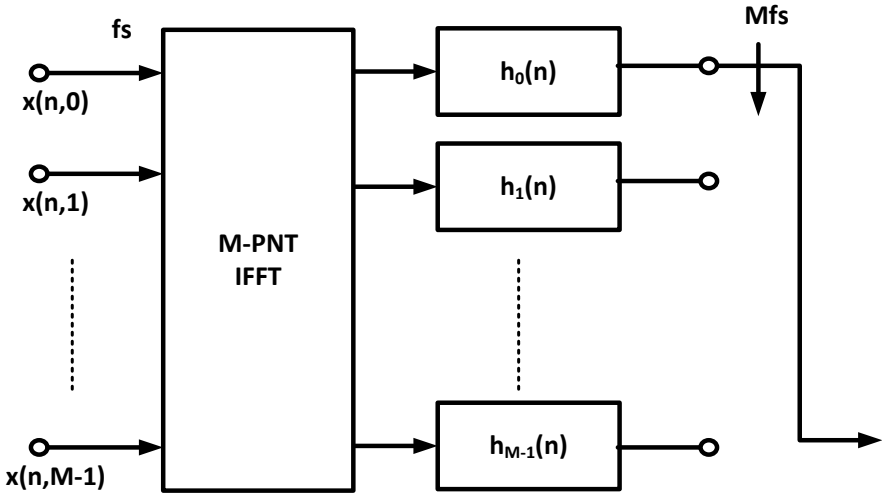


Figure 2.17. Structure of standard M-path up converter polyphase channelizer.

Next chapter will show how to change the sampling rate inside the M-path polyphase. The goal will be to show how we can get the output sample rate as a ratio of the input sample rate. This structure will be used in the proposed receiver for cognitive and software define radio.

CHAPTER 3
ANALYSIS AND SYNTHESIS CHANNELIZER

In Chapter 2, we explain the structure of a standard M-path polyphase down converter channelizer. The sampling frequency, the channel bandwidth and the channel spacing for those types of channelizers are equal and fixed. The standard polyphase channelizer is mainly used for communication systems. In this chapter we explain a version of the standard polyphase channelizer that is able to change the sample rate. Note that these modifications will allow us to obtain an output sample rate that can be any ratios of the input sample rate.

3.1 M-TO-2 DOWN CONVERTER CHANNELIZER

Figure 3.1 shows the standard polyphase down converter in which the channel spacing and the channel bandwidth are equal to input sample rate. The commutator for the standard polyphase down converter delivers M samples to the M-path polyphase starting from port $M-1$ up to port 0. The M-path polyphase will perform the down sampling of M to 1.

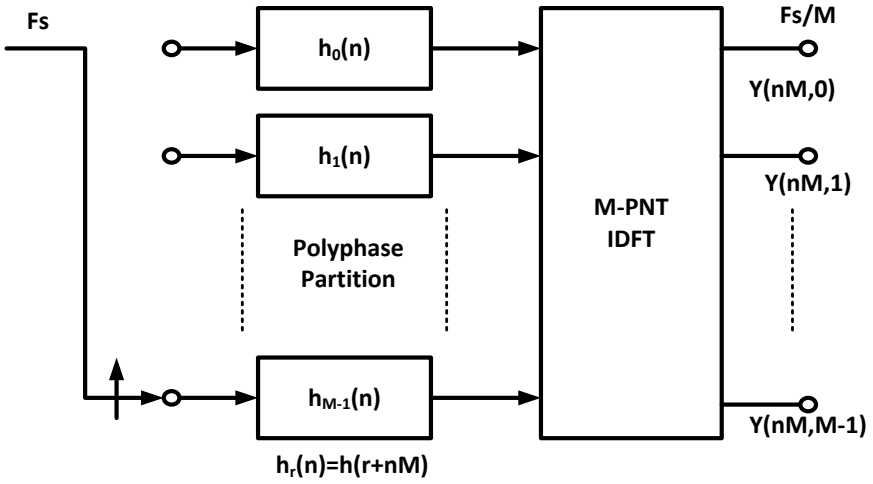


Figure 3.1. Standard M-path polyphase down converter channelizer.

Here we explain how to use a polyphase channelizer to increase the sampling rate by a factor of two, and also show how to change the input frequency f_s to $2f_s$ with maintaining the channel bandwidth and spacing equal to f_s. The first problem with the standard down

converter is that the transition band edges of the channelizer filter alias together, because the channel bandwidth and channel spacing are equal to the output sample rate. With small modifications, the polyphase filter will be able to perform any arbitrary ratios of the input and output sample rates. Here we need to perform a $M/2$ to 1 down sampling instead of M to 1. To do so, we first modify the way that the commutator delivers input to the M-path polyphase filter. Instead of delivering M samples, we modify the commutator to deliver $M/2$ samples starting at port $M/2$ - 1 up to port 0. Figure 3.2 shows this modification in order to obtain the desired sample rate change [12].

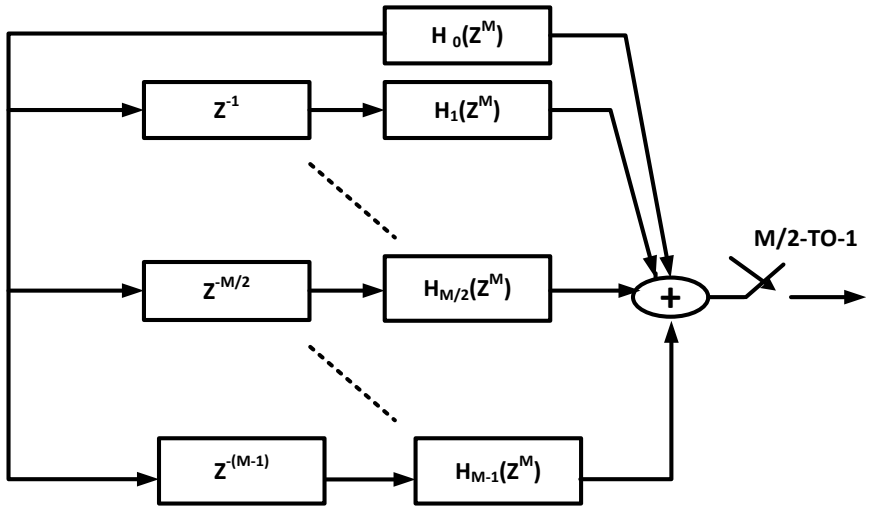


Figure 3.2. Initial structure of M-path filter with M/2 to 1 down sampler.

Next is applying the Nobel Identity and bring the down sampler inside the filter paths. At the end we replace the delays and the down samplers with a commutator. These two steps are shown in Figure 3.3 and Figure 3.4 respectively. After all the samplers are delivered to M-path polyphase, and the outputs are computed, the next step will be IFFT. But there is an extra work to do before sending these outputs to IFFT. We need to apply a time alignment and modify the phase of the outputs back to the original phase. By modifying the commutator to deliver $M/2$ samples instead of M samples, we change the time origin of the sample. This shifted origin needs to be fixed before sending the output of the M-path polyphase to the IFFT. The IFFT origin is aligned to the data origin without the shifting. The phase shifting happens when we change the way that commutator delivers the samples.

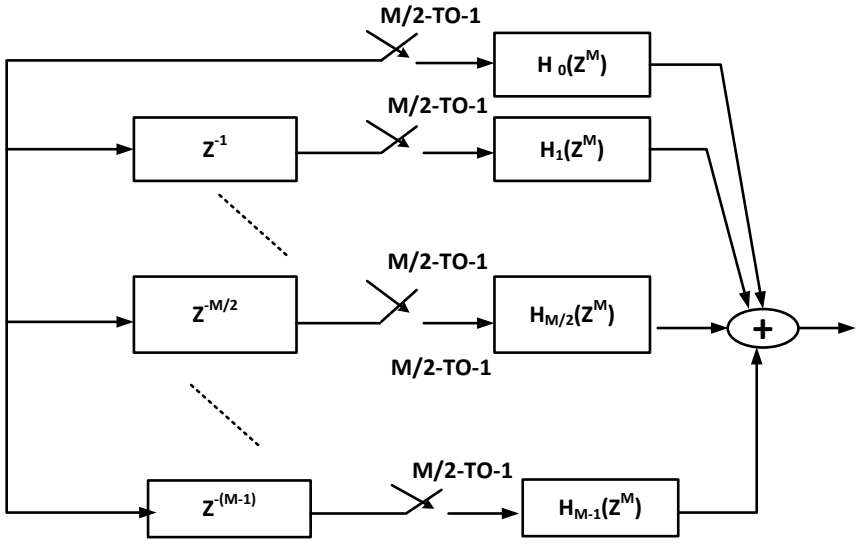


Figure 3.3. M-path down converter after applying nobel identity.

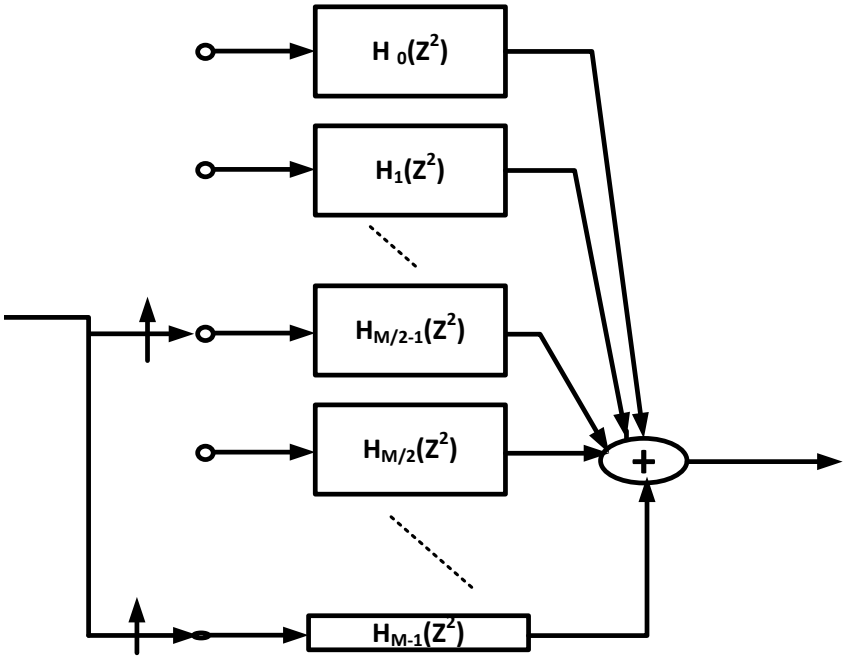


Figure 3.4. M-path filter when path delays and resamplers are replaced by an input commutator.

This issue is shown in Figure 3.5 for better illustrating. The first $M/2$ samples delivered to the data register have the same time origin as the IFFTs origin. But the problem will show up when the second $M/2$ get delivered. The first $M/2$ samples are shifted by a length of $M/2$, to get space opened for the second series of the samples. So now the time origin will be $M/2$ but the time origin of the IFFT is 0.

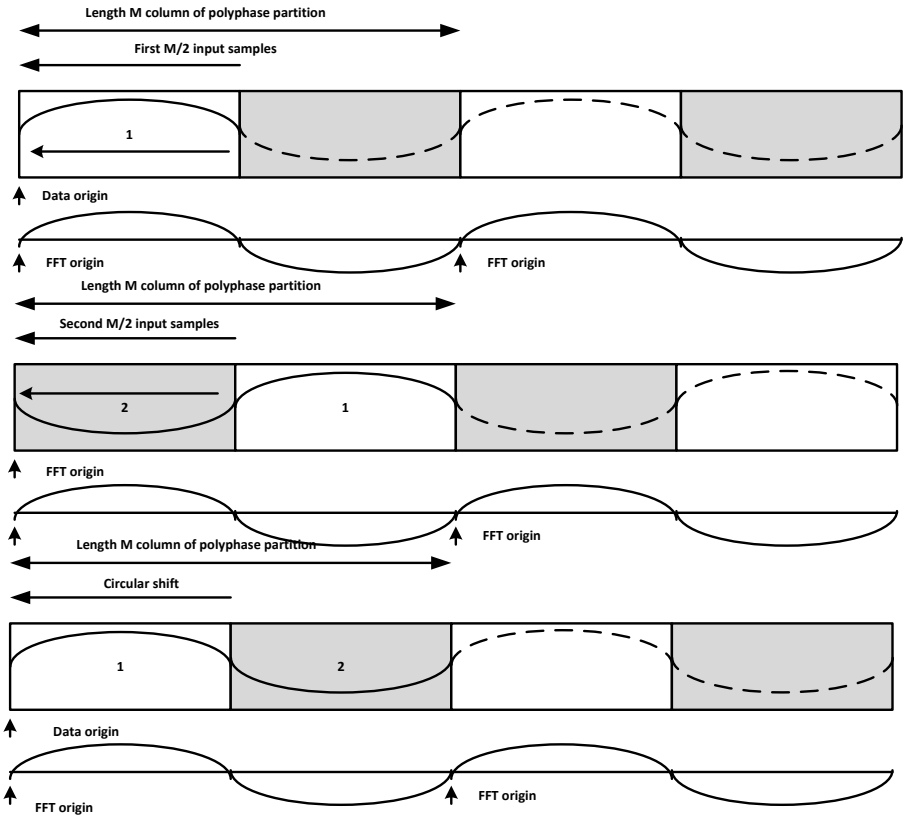


Figure 3.5. Demonstrating of phase reversal of M-point input to M/2 path polyphase filter.

Equations 3.1 to 3.4 show this phase shifting:

$$\Theta(\omega) = \Delta t \cdot \omega \tag{3.1}$$

In equation 3.1 we have: Δt = Time delay, ω = Frequencies of interest, n = Number of samples, and T = Time interval between samples.We know that $\Delta t = nT$, and also $\omega = k\frac{1}{M}\frac{2\pi}{T}$.

$$\Theta(\omega) = nT.k\frac{1}{M}\frac{2\pi}{T} = \frac{nk}{M}2\pi, \tag{3.2}$$

$$\Theta(\omega)_{for(n=M)} = 2\pi.k, \tag{3.3}$$

$$\Theta(\omega)_{for(n=M/2)} = \pi.k, \tag{3.4}$$

As described, we need to perform a phase shift correction before sending the samples to IFFT. This phase shift can be corrected by applying a circular time shift to the samples. The way this circular shift works is to apply a circular time shift to the outputs of the M-path polyphase filter as illustrated in Figure 3.6 [9].

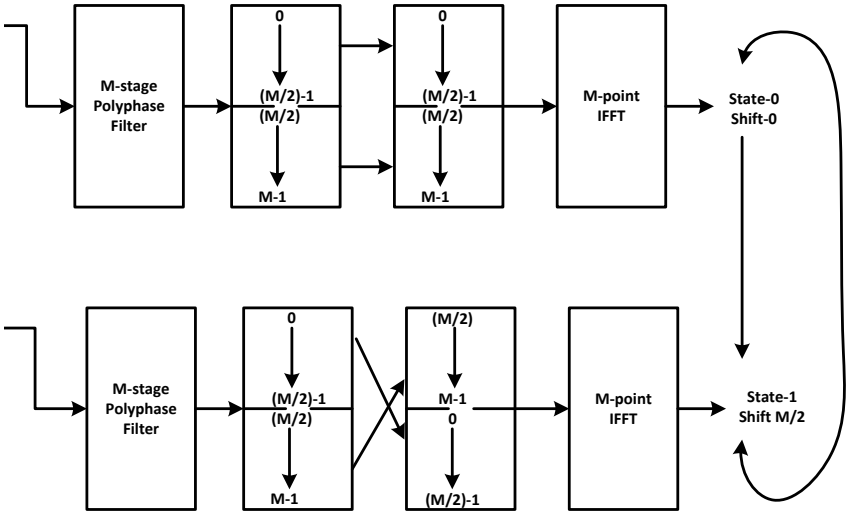


Figure 3.6. Circular shift in M/2 path polyphase channelizer.

Figure 3.7 shows the M-to-2 down converter channelizer with all the modifications that we explained:

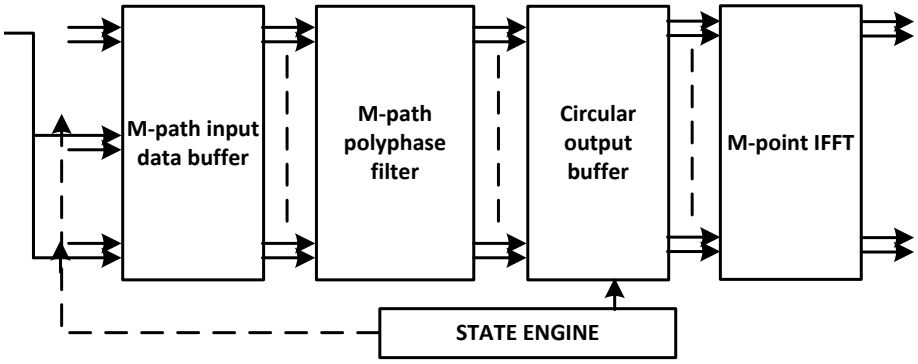


Figure 3.7. Structure of M to 2 down converter channelizer.

3.2 DIGITAL DOWN CONVERTER USED IN SDR

In Section 3.1, we introduce a way to change the sample rate inside the standard M-path down converter channelizer. This change can be considered as one of the modifications needed for using this type of channelizers in cognitive and software defined radio. We show how the choice of using M-to-2 down sampling helps us to avoid the aliasing at the edge of the adjacent channels. This design still needs more modifications in order to be used in cognitive and software defined radio. The signals which are processed in the standard M-path down converter has to be exactly located at the center frequency of the channels, kf_c, which $k = 0, 1, ..., M-1$. When the input signal has a center frequency of $kf_c \pm \Delta\ f_k$, the standard M-path channelizer will move the signal but instead of shifting it to the DC, it will shift the signal by the same offset of $\Delta\ f_k$ from the DC. The standard M-path filter is not able to compensate the frequency offsets, and more processing blocks should be added to the current structure to address this issue. The type of the low-pass filter needs to be considered to avoid the energy losses. Nyquist filters are proposed to be used in M-path channelizer. This filter has a unique property. The band edge gain of this filter is about -6dB and when M of them is used in each path of the channelizer, the band edges of these filters overlap each other. So as a result when the input spectrum occupies more than one channel, the channelizer is able to collect all the signal energy [13]. Figure 3.8 shows the proposed structure for down converter channelizer for cognitive and software defined radio.

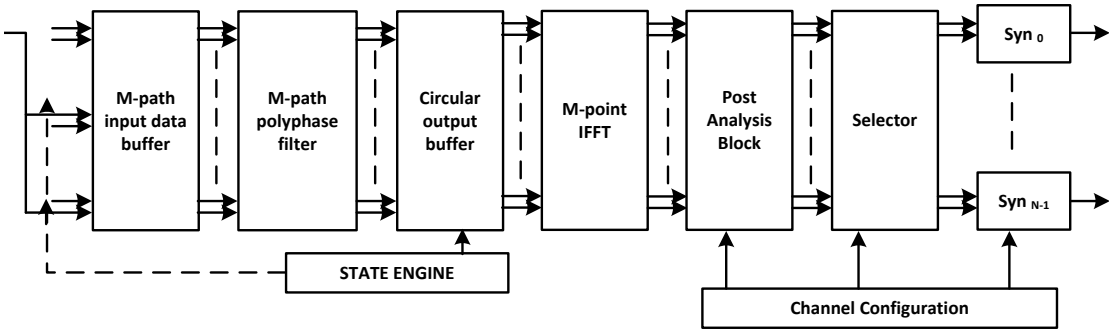


Figure 3.8. Proposed structure of the down converter chanelizer for SDR.

There are other issues with the standard M-path channelizer which need to be considered [14]. Another issue is when the spectra of the input signal is wider than the single channel, we need to break the signal to small fragments and reassemble these fragments at the end of the process. Other sections are added to the proposed M-path channelizer such as

post-analysis and post-synthesis block. When two or more spectra of the different input signals are processed in one channel, further filtering process is done by the post-analysis block. It will separate the bands belonged to different signals and then send them for further processing. In the other cases, The input signal has a wider bandwidth than the channel spacing. The analysis channelizer, breaks the input signal into several fragments. These fragments need to be reassemble in order to achieve the original signal bandwidth. The post-synthesis block is responsible to re-sample the fragmented signal by conducting the up sampling and translating each sample to the proper frequency region.

3.3 M-TO-2 UP CONVERTER CHANNELIZER

In this section, we introduce a version of the standard up converter channelizer which performs a sample rate change of $2\ f_s/M$ to f_s. Note that the channel bandwidth and the channel spacing are both equal to f_s. The task of the up-sampler is to do the zero-packed process. First we start with the initial structure of the M-path up-converter shown in Figure 3.9 which needs some modifications.

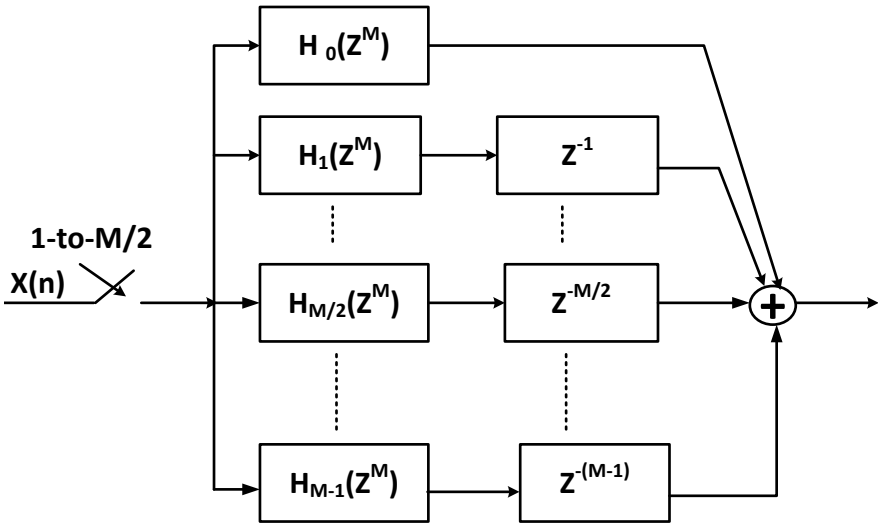


Figure 3.9. Initial structure of M-path up-converter with M/2 to 1 up sampler.

The first modification will be applying the noble identity and move the up-sampler inside the filter paths. Next is to replace the 1-to-$M/2$ up-sampler and the delays on each path of the polyphase filter with a pair of output commutators. This structure still needs two more modifications. First the same issue that was explained in previous section regarding the phase rotations generated by each M-path. The upper and lower parts of the M-path filter have the

same phase values but opposite polarities. To avoid these channels to cancel each other at summation part, we need to apply phase rotators, or easier way, an IFFT block prior to the M-path filters. Next modification will be apply to align the time origin of the output signal of the M-path filter with the original input. We start from the way that the two output commutators to explain the issue. The first commutator receives the output of the upper half of the M-path filter, started at point $M/2$ to zero, and the second commutator delivers outputs of the lower half of M-path filter starting from port M-1 up to M-2. Consider a sine wave as an input of the M-path up-converter. The time origin of this input is zero. But when this input is processed by the upper half of the M-path filter, the first commutators will deliver the data, the time origin of the output will be $M/2$ instead of zero. So there is a π radiant shift between the origins of the original input data and the output data. To address this issue, we implement a circular shift between the IFFT block and the M-path filter to maintain phase alignment of the output of the M-path filter. The final structure of 2-to-M up-converter channelizer has been shown in Figure 3.10.

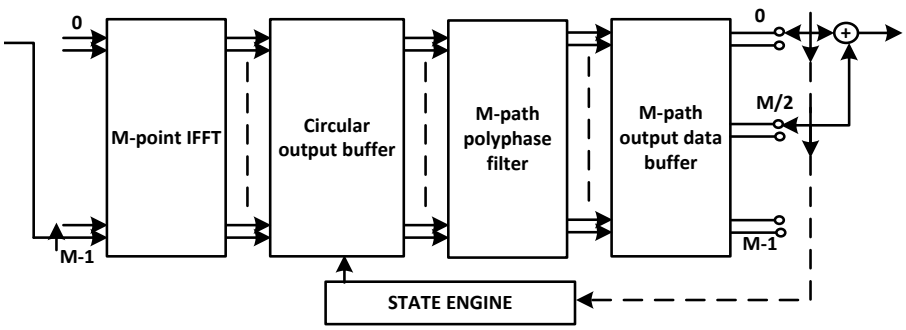


Figure 3.10. Final structure of 2-to-M up-converter channelizer.

3.4 DIGITAL UP CONVERTER USED IN SDR

The structure of proposed transmitter using in SDR contains a 2-to-M up-converter channelizer, N pre-processing analysis channelizer, and a channel selector. If the bandwidth of the input signal is less than or equal to the bandwidth of the channelizer's channel, the 2-to-M up-converter channelizer up-samples the input and shifts it to the base-band. But when the bandwidth of the input signal is wider than the channel's bandwidth of the up-converter channelizer, we first have to break the signal to smaller fragments and send them to the 2-to-M up-converter channelizer. This task is done by the N pre-processing analysis block, which are smaller analysis channelizer. The number of the pre-processing, N, indicates

the number of the signals which their bandwidths are greater than the synthesis channelizer channel bandwidth. This pre-processing analysis block partitions these input signals to smaller fragments, and down converts them to the base-line. The channel selector which is connected to a channel configuration, routes the correct input signals to the synthesis channelizer. If the input signal has a bandwidth wider than the synthesis channelizer channel bandwidth, the channel selector will rout the signal to the N pre-processing analysis channelizer for further processing. Figure 3.11 shows the final structure of the proposed transmitter for SDR.

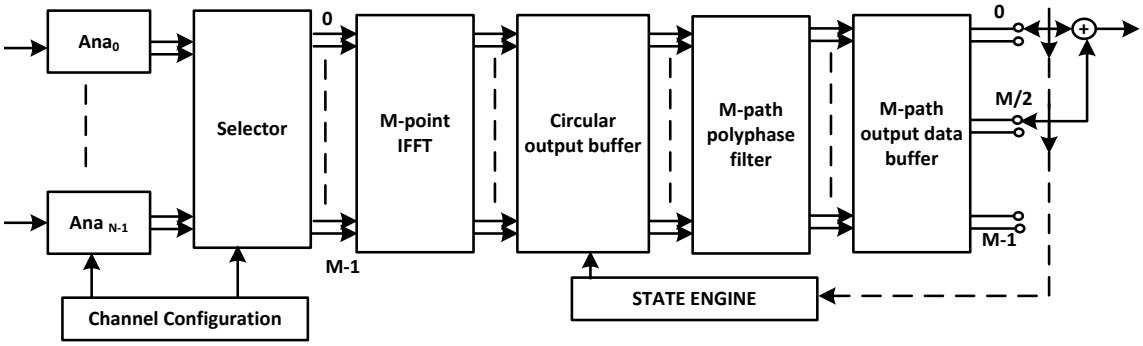


Figure 3.11. Proposed structure of the up converter channelizer for SDR.

CHAPTER 4
VARIABLE BANDWIDTH POLYPHASE FILTER BANKS

4.1 128-PATH POLYPHASE CHANNELIZER

In Chapter 4, we introduce the design of our proposed variable bandwidth filter. We start with designing a cascade of 128-path polyphase analysis-synthesis filter bank, which operates at two samples per channel to avoid aliasing of channel band edges together. The first step is to design the filter for each of the analysis and synthesis block. We use the *sinc* algorithm to design the filter for analysis filter bank and *remez* algorithm for synthesis, as shown in Figure 4.1. The length of the filter is 1791, and the sampling frequency is 128 kHz.

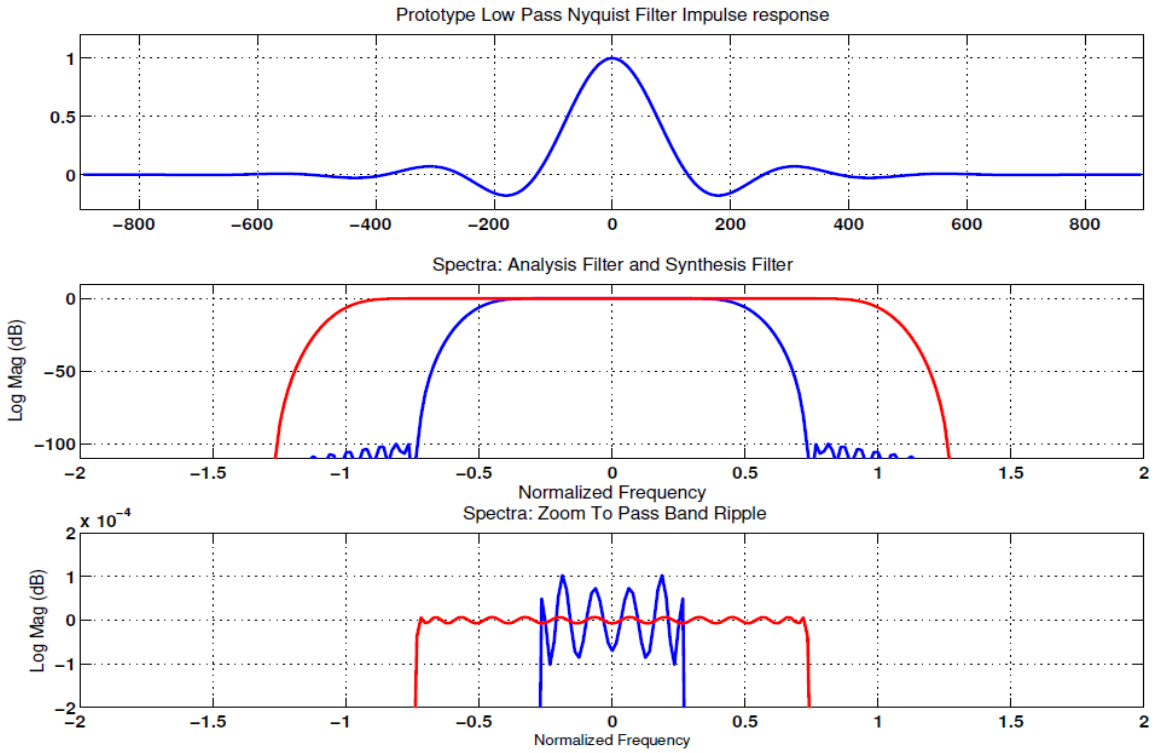


Figure 4.1. Spectra of 128-path analysis and synthesis in red and blue respectively.

The commutator is designed in a way that only 64 samples are delivered to the 128-path analysis filter per time. The first 64 samples are loaded in upper half of the

commutator and the same set of samples are loaded to the lower half of the commutator. The way that commutator works is that it starts inserting the 64 samples to 128-path polyphase starting at port 63 up to port 0. At the same time it inserts the next 64 samples to the 128-path polyphase starting from port 127 up to port 64. The number of the coefficients at each path of the polyphase filter is equal to 14. The number of samples at the output of the 128-path polyphase filter is 128. The next block is the circular shift buffer. When the commutator starts from the middle of the 128-path polyphase filter to send the samples, the time origin of the output of the 128-path will be different than the original input signal. The time origin of the output of the 128-path polyphase is 63, but the time origin of the original input is zero. To align this time origin, we send the outputs of the 128-path filter to a circular shift buffer. This buffer will also make the data origin align to the IFFT time origin which is also zero. The task of this buffer is to apply a shift in time domain in a way that it inverses the order of the received data from the 128-path analysis filter. Here the first 63 samples processed by the upper half of the 128-path filter will be shifted to the lower section of the circular buffer and the outputs of the lower half of the 128-path will sit at the upper half of the buffer. After aligning the time origin, all of the 128 samples will be pushed to IFFT block for phase rotation process. The input and output of 128-path channelizer are shown in Figure 4.2.

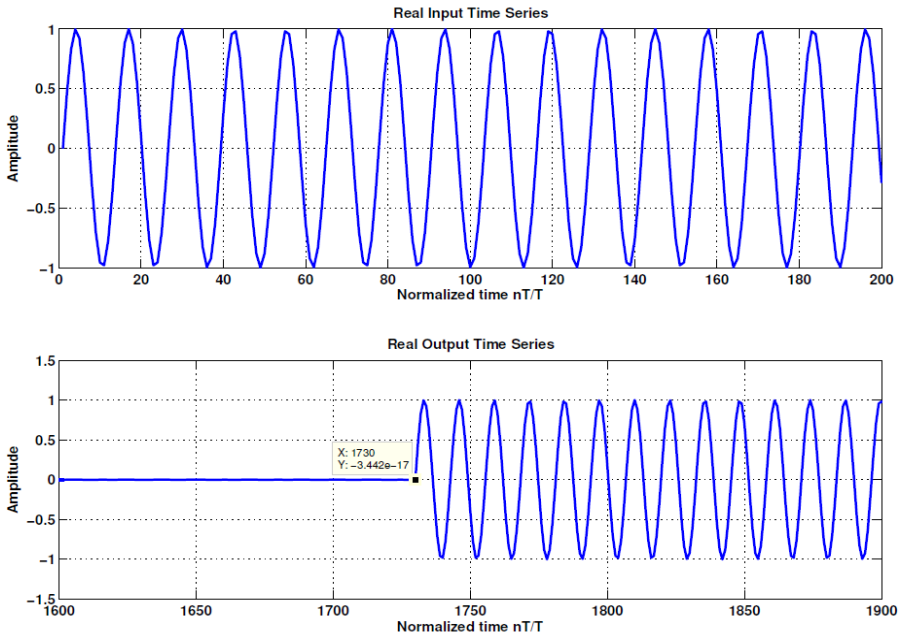


Figure 4.2. Input and output of analysis and synthesis filters.

The procedure for synthesis block is inverse of the analysis. Thus, we first send all of the 128 outputs of the analysis filter to the IFFT block of the synthesis filter, then we apply the

circular shift, and last, filter all of the samples and deliver the filtered samples to two commutators. The first commutator sends the data out from the 128-path synthesis polyphase filter starting from port 63 up to port 0, and the lower commutator outputs the lower samples of the 128-path polyphase starting from port 127 up to port 64. We apply two different set of inputs to this design, one with an impulse and one with a periodic sine wave. For both input types, we observe a delay of the 1731 which corresponds to the processing delay time of analysis-synthesis filter banks.

4.2 Simulation Results for 128-Path Polyphase Channelizer

One configuration of the 128-path channelizer filter is to see how it functions if we turn off all the analysis channels and leave a few of the channels on. First, we turn off all the channels and let only outputs of 25 channels go through the synthesis filter. The stop-band of the filter is equal to ± 25.75 kHz, as shown in blue in Figure 4.3. Next we leave 22 and then 19 channels on and set the rest to zero. The stop-band for these configurations are ± 22.75 kHz and ± 19.75 kHz, respectively. Note that for both cases, the DC is included, and also for proper visualization of filter impulse response, we use impulse as the input signal. We observe that the bandwidth of the filter will shrink when fewer outputs of the analysis filter go through the synthesis channelizer. Thus, by turning the channels on or off, we are able to achieve a variable bandwidth filter function. Here for all cases the delay remains the same as 1731.

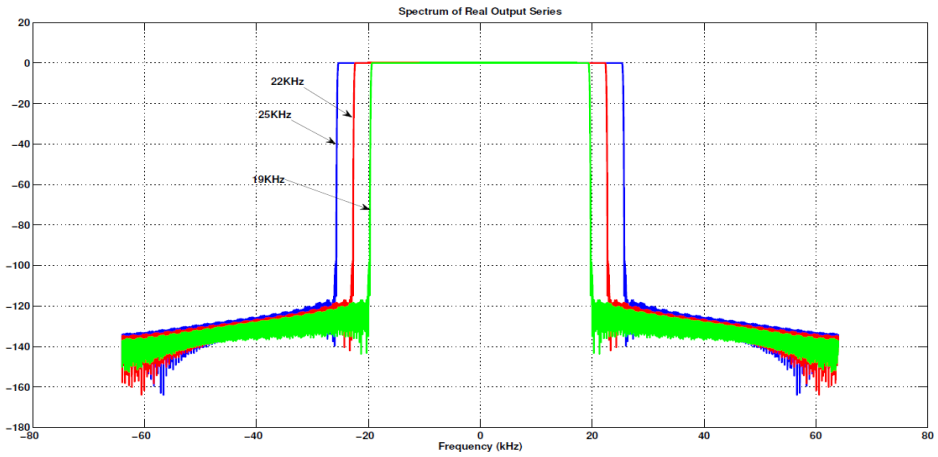


Figure 4.3. Spectra of 128-path channelizer with different bands.

4.3 10-Path Polyphase Channelizer

The same procedure will be followed as explained in the previous section to design a 10-path analysis-synthesis channelizer. The filter length is 139, and the sampling frequency is 10 kHz. The same algorithms explained in Section 4.1 will be used to design the analysis and synthesis filters, and the spectra representation of this channelizer is shown in Figure 4.4.

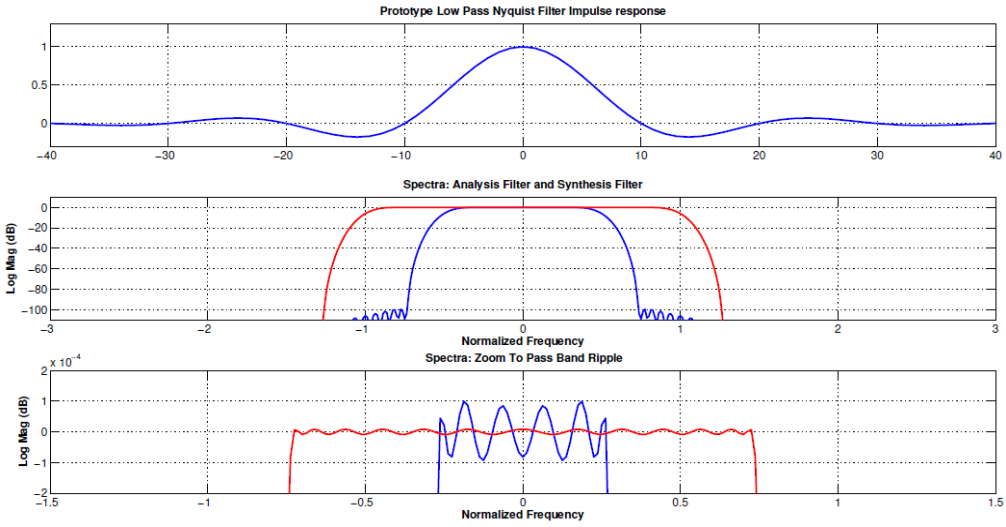


Figure 4.4. Spectra representation a 10-path analysis and synthesis filter.

The input commutator of the analysis is delivering 5 samples to the 10-path polyphase filter. Note that the number of the coefficients of each path of the 10-path polyphase filter is 14. A circular shift will be applied to align the time origin of the outputs of the 10-path polyphase filter with the time origin of the IFFT block output. The analysis filter doubles the sampling frequency, and the output of it will be pushed to synthesis filter for recovery of the original signal. We try to sets, first we send a sine wave and next we send an impulse. For both cases, the delay at the output of the synthesis filter is 136. Figure 4.5 shows the output of the 10-path channelizer with impulse as the input signal. We will show how to use this delay to deliver data to synthesis filter when we have a combination of a cascade of two or more analysis-synthesis channelizers.

4.4 Combination of Two Channelizers with Different Filter Paths Number

Now we want to implement the 10-path channelizer between the analysis and synthesis filter banks of the 128-path channelizer. To do so, we first start with the 128-path channelizer and send the outputs of the analysis to the synthesis with a delay of 136.

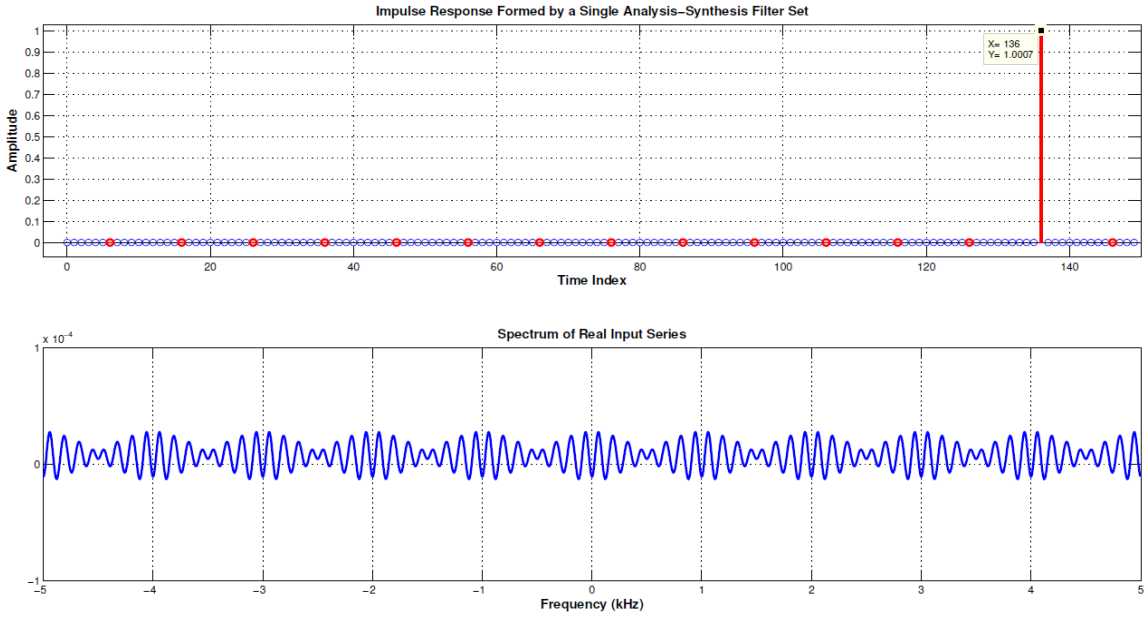


Figure 4.5. Input and output of an impulse signal from the 10-path analysis and synthesis filters.

The delay of 136 is equal to the delay generated by the 10-path filter. To create that delay between the outputs of the analysis and the inputs of the synthesis, we introduced a buffer of size 128x137, shown in Figure 4.6. Each time filter processes data, the outputs of the 128-path IFFT will sit on the first column of the buffer, and the rest of the data in the buffer will be shifted to the right. Note that the input of the 128-path synthesis will be obtained from the last column of the buffer. Since it takes 136 times for the first output of the 128-path filter to get to the last column of the buffer, we need to repeat this transaction for 136 times extra compared with previous step. We have about 375 packets of the 64 samples, so our loop should run for a total of 512, that is 375 times for the actual data packets and 136 for the added delay.

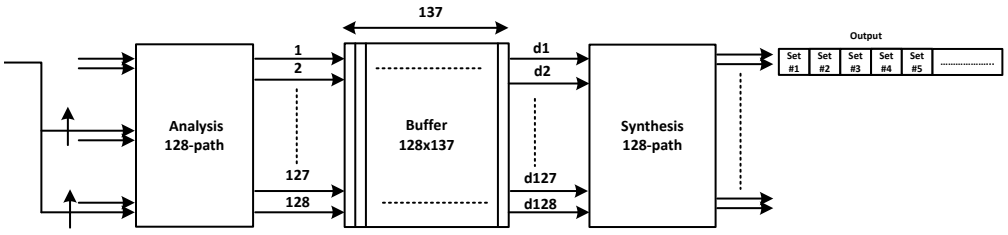


Figure 4.6. Buffer of size 128x137 used in 128-path channelizer.

As shown in Figure 4.7 the total delay of the output of the 128-path channelizer is now 10434. This increase in delay is caused by the buffer. This buffer applies an extra delay of 8704, to the total output of the 128-path synthesis. Since we have about 375 packets of 64 samples, and a delay of 136 was applied by the buffer to each samples, so 64×136 is equal to 8704. The 128-path channelizer by itself applies a delay of 1730 to the signal. So the total of 8704 plus 1730, equals to 10434 is the final delay of the 128-path channelizer with the buffer of 128x136. This step is equivalent to the condition that we apply multiple 10-path channelizer at the output of the 128-path analysis before sending these outputs to the 128-path synthesis.

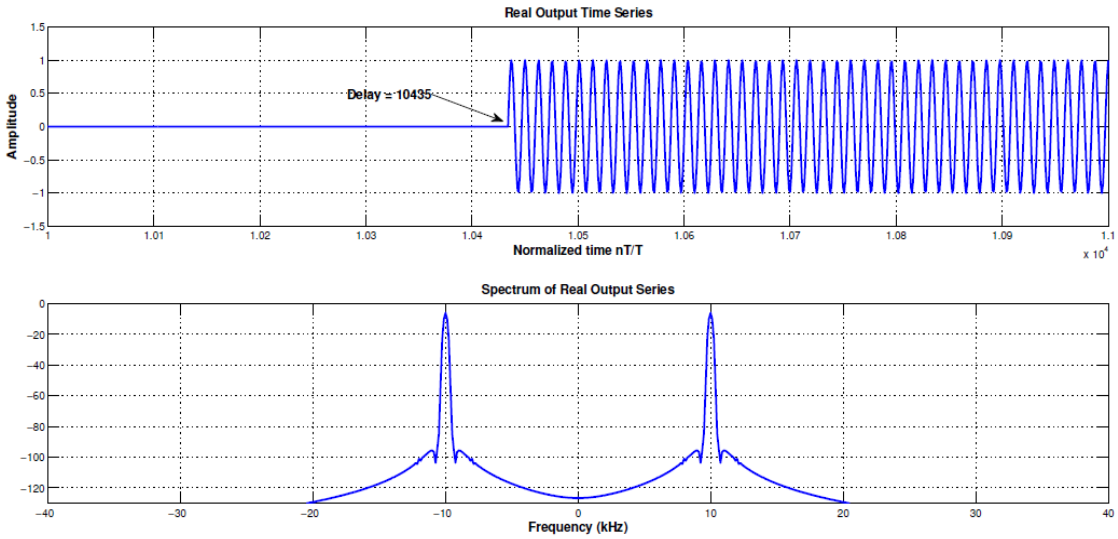


Figure 4.7. Output of the 128-path channelizer after applying 136 delay to each input sample.

4.4.1 128-Path Analysis with a 10-Path Channelizer

Next filter configuration is to send the outputs of the 128-path analysis to the synthesis through a 10-path channelizer. Here we verify two different configurations. One structure contains a 128-path analysis channelizer along with a full 10-path channelizer, and the second structure contains the full 128-path and 10-path channelizer. In both structures, we process the whole input signal in the 128-path analysis, and then extract the outputs of the channel $\pm$ 25 from DC, and send them to the 10-path channelizer. In the first structure, shown in Figure 4.8, we stop the process at the output of the 10-path channelizer, and capture the result.

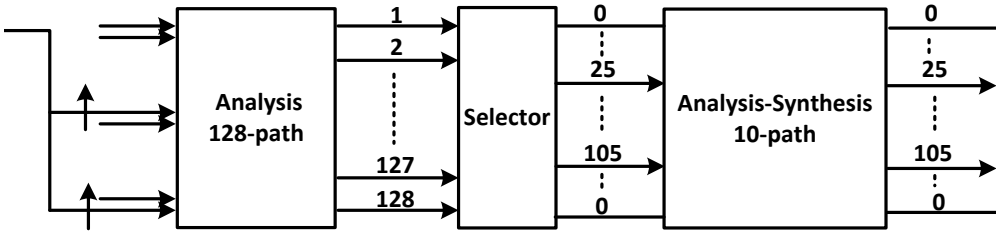


Figure 4.8. Structure of 128-path analysis along with a 10-path channelizer.

In the second one, shown in Figure 4.9, we will send the outputs of the 10-path channelizer to a 128-path synthesis. Note that only the outputs of the 10-path channelizer will be processed in the 128-path synthesis.

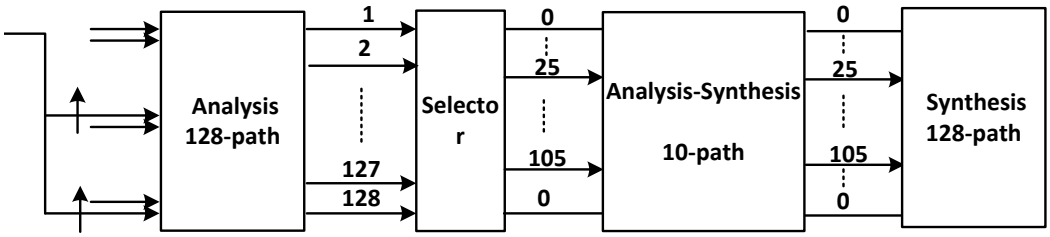


Figure 4.9. Structure of 128-path channelizer along with a 10-path channelizer.

In the first configuration, the input signal will be processed in the 128-path analysis filter. Then we define a weighted vector and zero all the outputs of the 128-path analysis filter except the outputs of channel $\pm$ 25 from DC. Now we need to prepare the output and send them to the 10-path channelizer. The input rate for the 10-path is 5 samples at a time. Thus, we define a channel of 128×5, and when all the 5 data are ready in this buffer, we will push them to the 10-path analysis commutator. Since we are only sending the outputs of the 128-path analysis to the 10-path channelizer, we should expect that all the outputs of the 10-path channelizer will be in base-band. The 128-path channelizer will down sample the input signal, while down converting it to the base-band by aliasing. If the center frequency of the input signal aligns with the center frequency of the channelizer channel's center

frequency, all of the outputs of the 10-path will be on the base-band. The result is shown in Figure 4.10. In this set, the center frequency of the input signal is equal to 25 kHz.

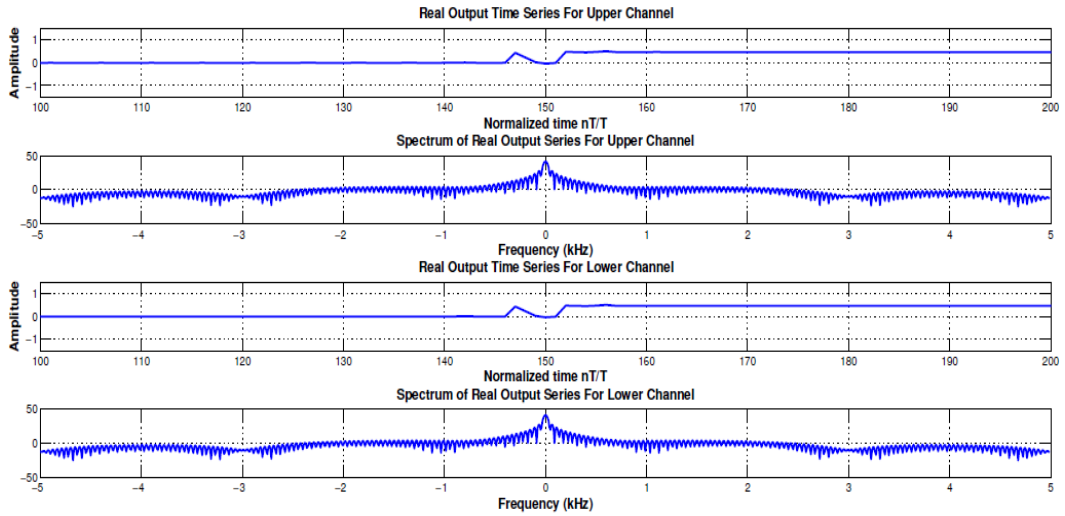


Figure 4.10. Outputs of upper and lower channels from 10-path channelizer.

Next we change the center frequency of the input signal to 25.2 kHz to see what would be be the impact if the center frequency of the input signal is not aligned with the channel's center frequency. We observe a move in the output spectrum. Instead of having all the outputs at the base-band, all of the channels are now with an offset from the DC, as shown in Figure 4.11.

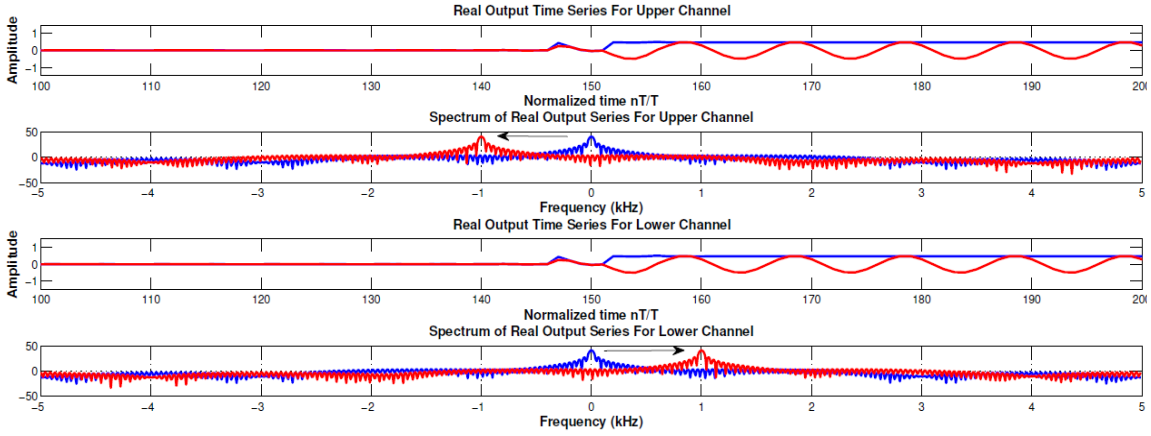


Figure 4.11. The outputs of upper and lower channels from 10-path channelizer with the offset from DC.

Now in the second configuration, we first process all of the input signal in a 128-path analysis, extract the outputs in Channels $\pm$ 25, and send them to the 10-path channelizer. We

add another step and add a 128-path synthesis at the output of the 10-path channelizer. The output is shown in Figure 4.12.

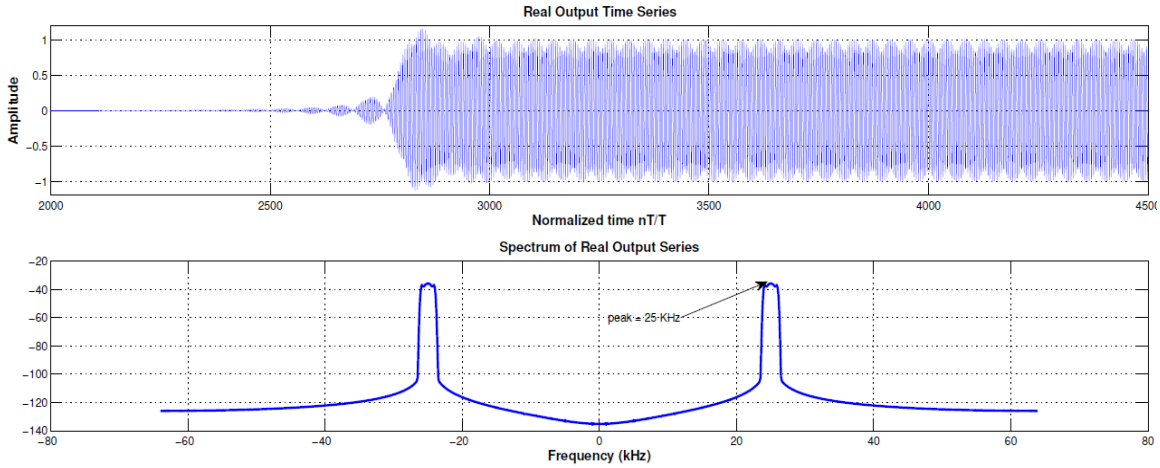


Figure 4.12. Outputs of channels from 10-path channelizer and 128-path synthesis.

We turn off some of the channels at the output of the 10-path analysis and only send part of the data to the 10-path synthesis and from there to the 128-path synthesis. The goal of doing this step is to show how by narrowing the channel we can increase the resolution of the receiver's input. First we turn off the first half of the 10-path upper channel, and turn off the second half of the lower channel of the 10-path analysis output, and then turn off all of the channels except the DC on the both lower and upper channels, and last we compare these two conditions at the output spectrum. Figure 4.13 demonstrates the difference of when only half of the channels are on, compared with when only DC is on.

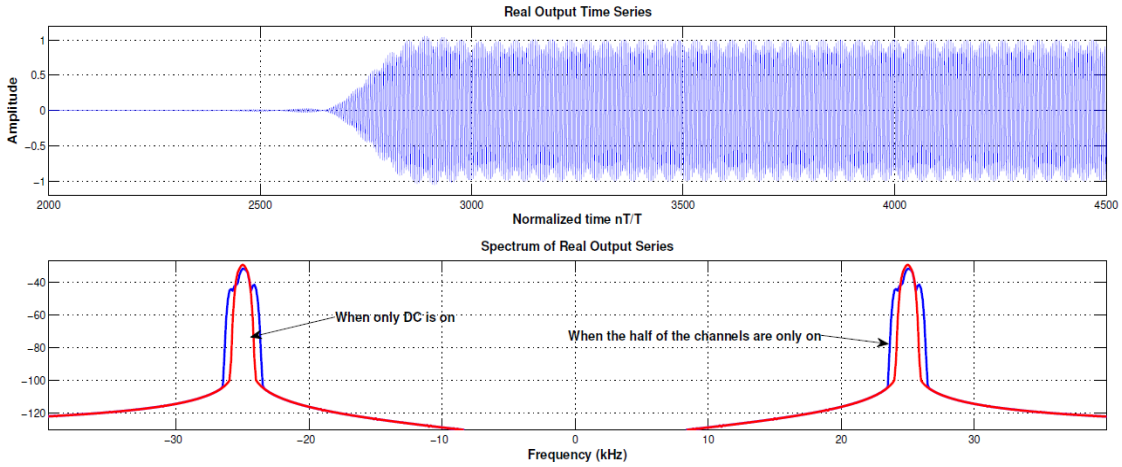


Figure 4.13. Comparison when all channels are on and when a few channels are on.

Therefore, the bandwidth of the channel in which only the DC is on, is narrower than the other one. So these configurations allow us to narrow the bandwidth of the desired channel in order to increase the resolution before sending the data to the receiver. As shown in Figure 4.14, we plot the output spectrum of the 128-path synthesis in the condition that all channels are on, and the condition where only the channel at DC in on.

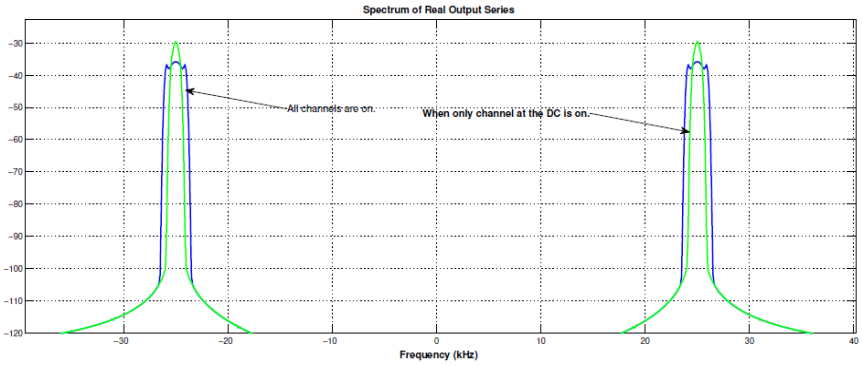


Figure 4.14. Comparison when all the channels are switched on and when all are off except the DC.

4.4.2 128-Path Analysis-Synthesis with a 10-Path Channelizer

In this section we explain how to combine the 128-path and 10-path channelizer in a way that part of the input of the 128-path synthesis will be first processed in a 10-path channelizer. The summery of the procedure is first processing all the input signal in the 128-path analysis, then send the outputs of last step to a selector to extract only a number of the outputs. Here our target channels are DC, and 25 channels upper and lower of DC, which is a total of 51 channels. The selected channels are channel 1 through channel 26, which are called upper channels, and channels 104 to 128, which are called lower channels. The data located in channel 26 and 104 will be processed in a 10-path channelizer, and from there aligned with their original data in the other channels at the output of the 128-path analysis, and will go through the 128-path synthesis to recover the original input signal. The data in other channels should be moved column by column in a buffer while the data in channel 26 and 104 are processing in the 10-path channelizer. The size of this buffer is determined by the delay which the 10-path channelizer cause to its input. As mentioned in Section 4.3, this delay is equal to 136. So each of data which comes out of the selector and have not been sent to 10-path has to have a delay of 136 before being processed in the 128-path synthesis. As the data is loaded in the buffer, it has to have a 136 move to compensate the delay caused by the 10-path channelizer. Thus we define a buffer with size 128×137. Note that the input of the

128-path synthesis will be from the last column of the buffer. Here four different configurations were explored. We explain them one by one. The first design is demonstrated in Figure 4.15.

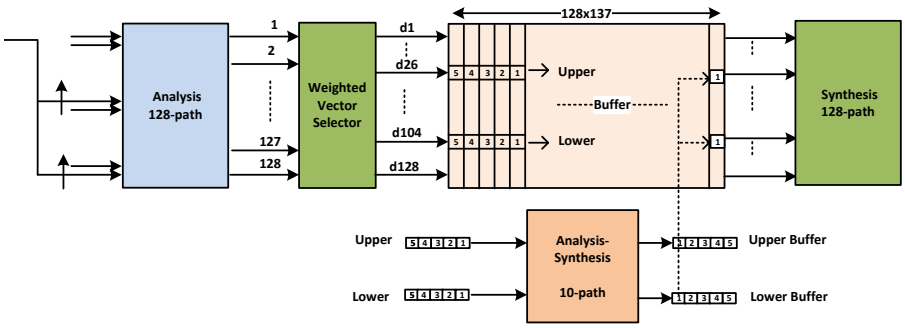


Figure 4.15. Complete variable bandwidth diagram with buffer size 128x137.

We send the outputs of the selector to the buffer 128×137, but we control condiction that when 5 outputs at each channel, from the selector are ready, send them to the 10-path channelizer. Since the 10-path channelizer procedure does not start unless 5 outputs of the selector are ready, we need to add 5 more column to the buffer to compensate this waiting time. The rest of the data in the other channels will remain in the buffer and at each run, they will be moved one column to the right. We design the last column of the buffer to be updated by the output of the 10-path channelizer at channel 26 and 104. The output of the first design has been shown in Figure 4.16.

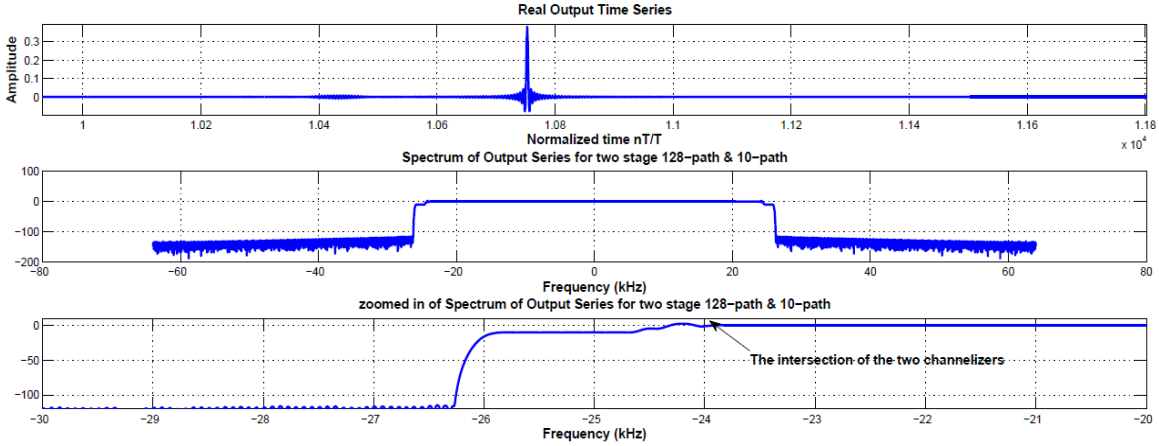


Figure 4.16. Result of updating the buffer one output of 10-path channelzier per time.

The problem, as we observe is a mismatch at the place that channel 24 and channel 25 are connected. Also the attenuation of the output of 10-path channelizer should be 0 dB, but it is down to -10 dB. We can get 5 outputs from the 10-path channelizer, we update the buffer, from channel 137 to 141, and let the outputs of the 10-path moves with the other data in the buffer and get them into the 128-path synthesis. Result of this step is shown in Figure 4.17.

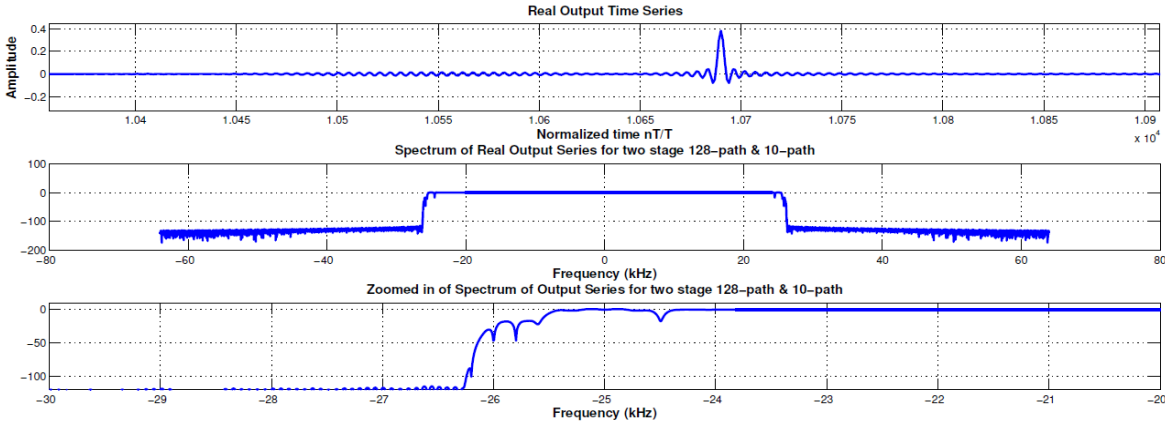


Figure 4.17. Result of updating buffer with one output of 10-path channelzier per time.

The tnext set up has the same structure of the other two designs as explained so far, with this difference that as soon as the first output of the 128-path analysis is ready, we send it to the 10 path. The order of the first 5 input to the 10-path is shown in Table 4.1. Since we do not have the condition of having 5 outputs of 128-path ready before loading the commutator of the 10-path channelizer, we can now reduce the number of column of the buffer back to 137. If we define a matrix of 1×5 for the output of the 10-path channelizer, only the last cell of this matrix will be the data which overwrites the current data in the last column of the buffer. This is applied for both sets of upper and lower channels.

Table 4.1. Order of the Input and Output

	input of the 10-path	output of the 10-path
1^{st} set	1 0 0 0 0	0 0 0 0 1
2^{nd} set	2 1 0 0 0	0 0 0 1 2
3^{rd} set	3 2 1 0 0	0 0 1 2 3
4^{th} set	4 3 2 1 0	0 1 2 3 4
5^{th} set	5 4 3 2 1	1 2 3 4 5

The output of this configuration is shown in Figure 4.18.

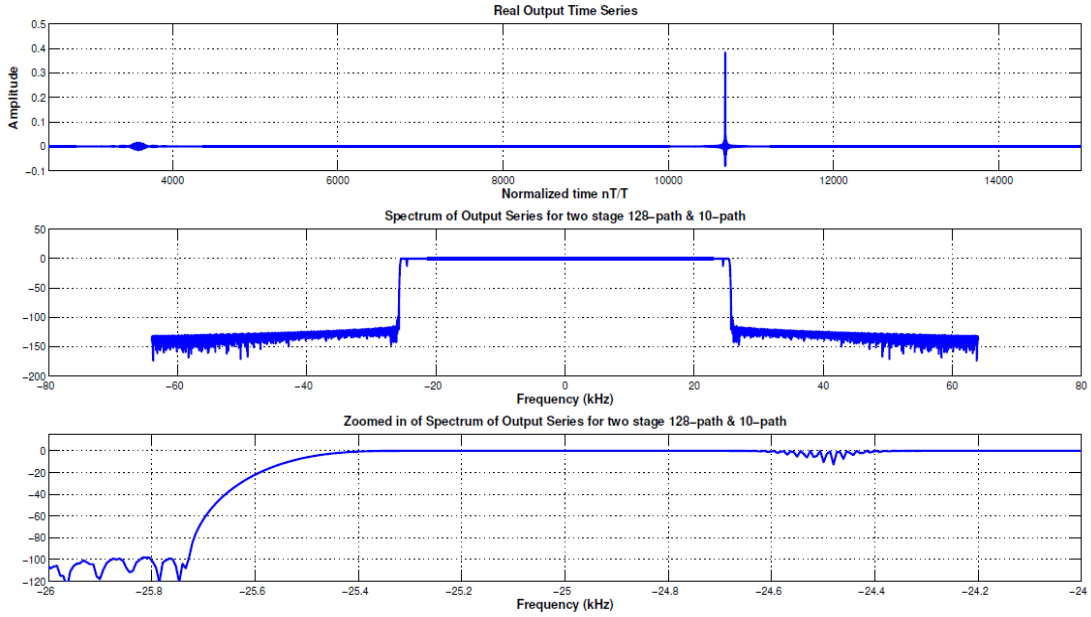


Figure 4.18. The output of the 128-path syntheisis with no proposed conditions.

We send the outputs of this channel to the 128-path synthesis through a buffer of size 128×137, and the second time, through a 10-path channelizer. When the outputs of the 128-path analysis go through a buffer of size 128×137, and then from there to the 128-path synthesis, the output sits at point 3650, as shown in Figure 4.19.

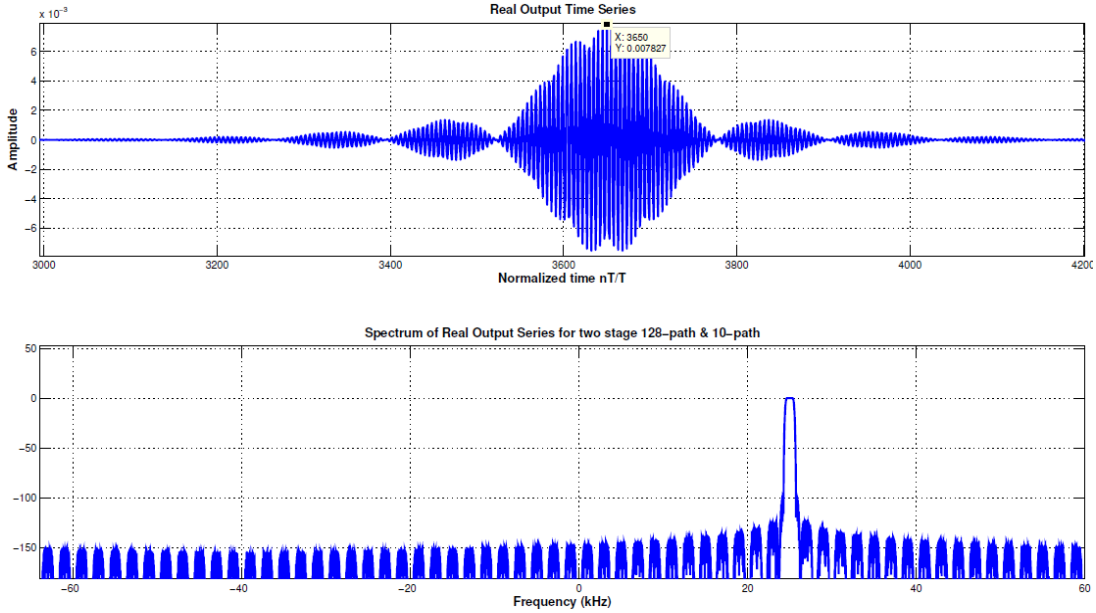


Figure 4.19. The output of sending the data of one channel through the buffer.

The second time, we replace the buffer with a 10-path channelizer, and plot the output of the 128-path synthesis. This time the output sits at point 10434, as pointed in Figure 4.20.

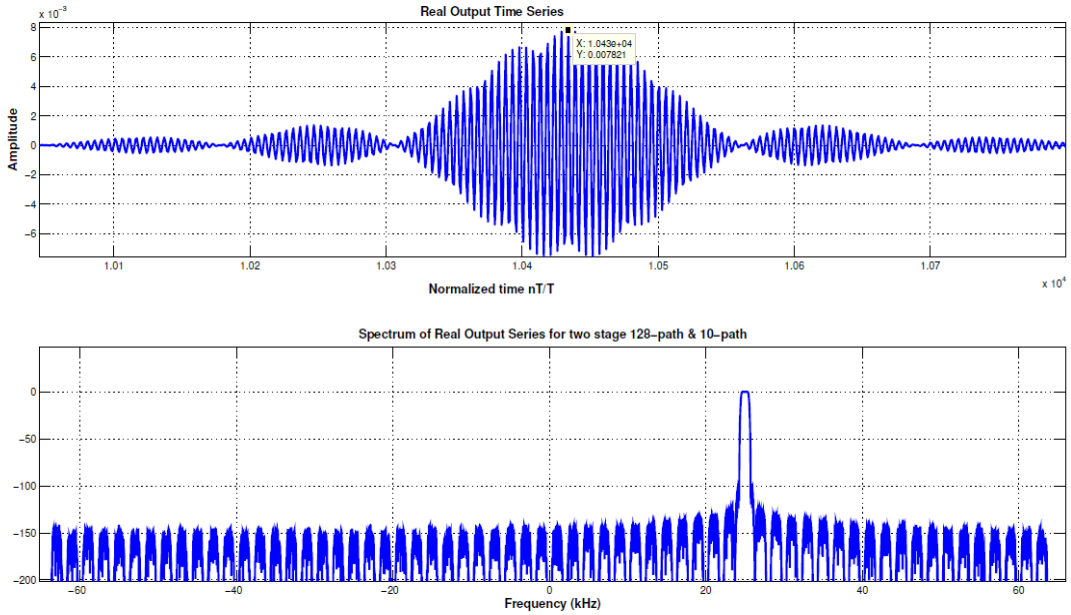


Figure 4.20. The output of sending the data of one channel through the 10-path channelizer.

Figure 4.21 illustrates the comparison of the outputs in time and frequency domain.

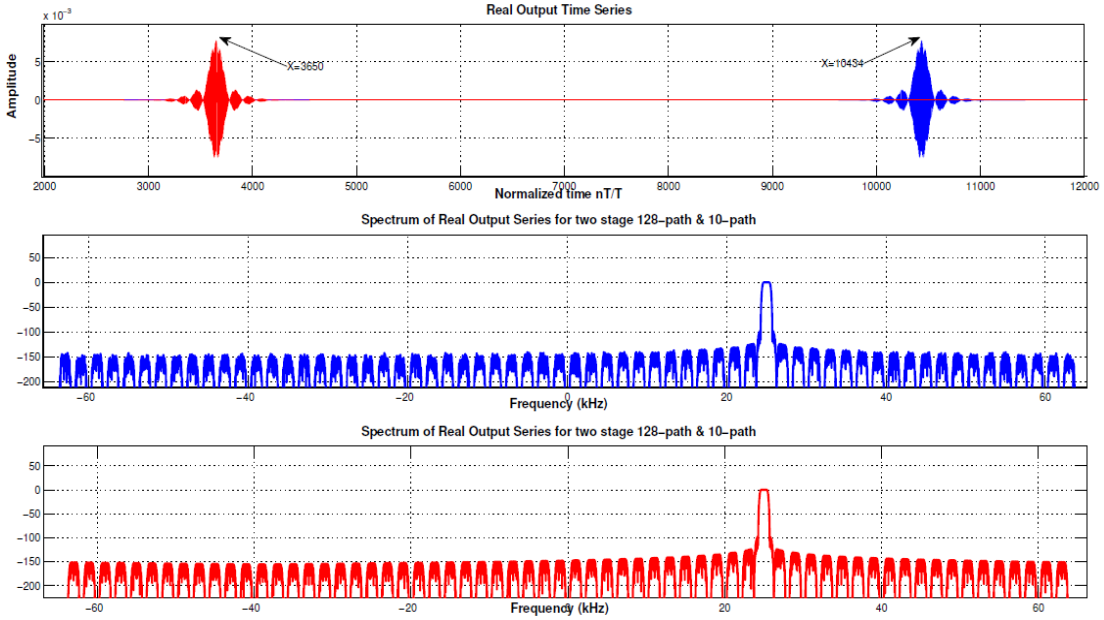


Figure 4.21. Comparing between the outputs of buffer or 10-path channelizer.

We observed that when the outputs of one channel go through the buffer of size 128×137 before going to the 128-path synthesis, the total delay is equal to:

$$1731 + (136 \times 64) = 10434 \tag{4.1}$$

On the other hand when the same outputs go through a 10-path channelizer before going to 128-path synthesis, the total delay is 3650. With a simple calculation the total delay applied to the outputs of the channel by the 10-path comes up as 30. The way we compute it is shown in Equation 4.3.

$$1731 + (m \times 64) = 3650 ==> m = 30 \tag{4.2}$$

Only based on above calculation, we start updating the buffer 128×137 with the outputs of the 10-path channelizer from column 30 for both upper and lower channels. With a slight change, updating the buffer from column 31 gives a better result, as shown in Figure 4.22.

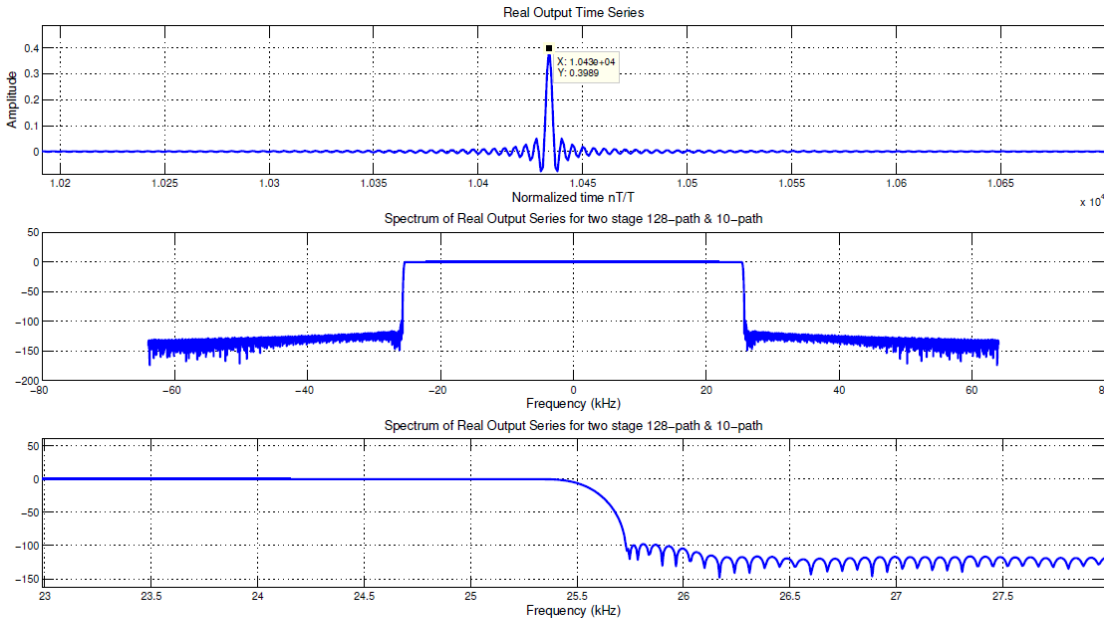


Figure 4.22. Final output of the 128-path channekizer in the presence of a 10-path channelizer.

The problem with the last design is that if we turn off all the channels in 10-path channelizer, the bandwidth of our filter will reduce by one channel from both upper and lower side, as shown in Figure 4.23. But this configuration does not work in the case that only half of the channels needs to be off. Since this design does not satisfy the condition that only half of the two channels at the band edges is needed to be on, we need to revise the architecture to address this requirement.

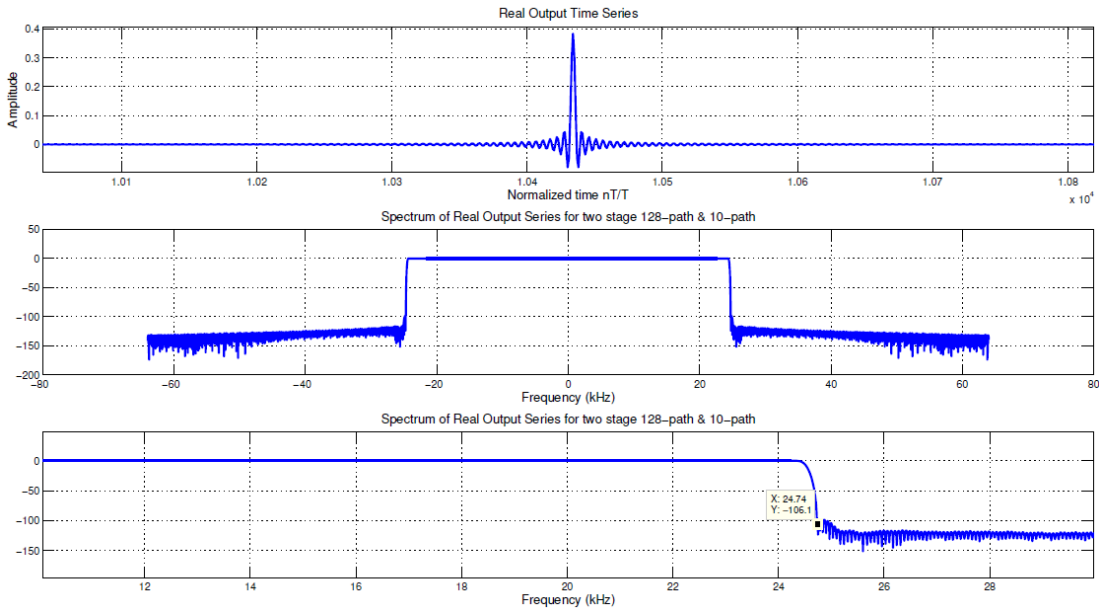


Figure 4.23. Output of the 128-path synthesis with no outputs from 10-path.

The limitation with the previous design is first we need to deliver 5 samples from the input signal, and when all 5 samples are ready then start the step of 10-path channelizer, but If same as the previous design, as soon as we receive data we start the process of sending even single sample of the input signal to the 10-path channelizer, we are not performing the down-converting of the input signal in the 10-path channelizer. Next is to design a structure which resolve the current limitation. The next proposed design is to process 5 sets of input signal, which each set contains 64 samples of the input signal, each time we run the 128-path analysis and synthesis channelizers. This will satisfy the criteria of down-converting process in the 10-path channelizer. We explain this design step by step. The first step is dividing the input signal to 75 packets, since we want to provide the whole system with 5 sets of 64 samples per time, and the total input signal contains 24000 samples, we will have a total of 75 packets. This is shown in Figure 4.24.

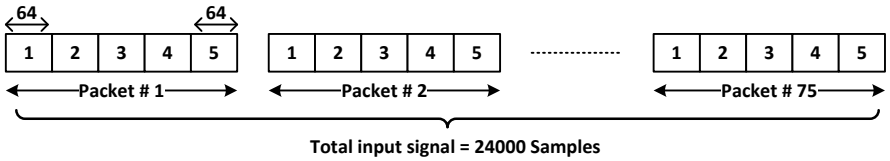


Figure 4.24. Total of 75 packets input data, each consists of 5 sets of 64 samples.

Now we define a buffer of size 128×5, which as reference we call it input buffer, to load the inputs for 128-path analysis filter, as shown in Figure 4.25. When all the 5 sets of the packet are loaded to the input buffer, we send them to the 128-path analysis filter, and let the filter to be performed for 5 times and deliver 5 sets of out puts. Before we send these outputs for further processing, we send them through a selector, which will turn off all the channels except dc and 25 channels upper and lower of it. So the total channels which carry data is 51 channels. Each time that we receive a set of output from the selector, we will send them to another buffer, refereed as output buffer, and let it move to the right each time we have a new sets ready from the selector.

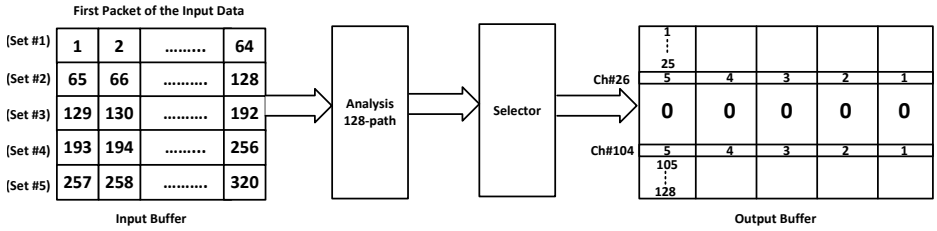


Figure 4.25. The 128-path channelizer input and output buffer.

Now we need to extract the inputs of the 10-path channelizer from this output buffer. We extract the data in channel 26 and 104, and locate them in two buffers of size 1×5, one for channel 26, which we call it upper buffer, and one for channel 104, which we call it lower buffer, as shown in Figure 4.26. Theses two buffers are carrying the input data for the 10-path channelizer. When these two sets of data are processed in the 10-path channelizer, we will have two sets of outputs.

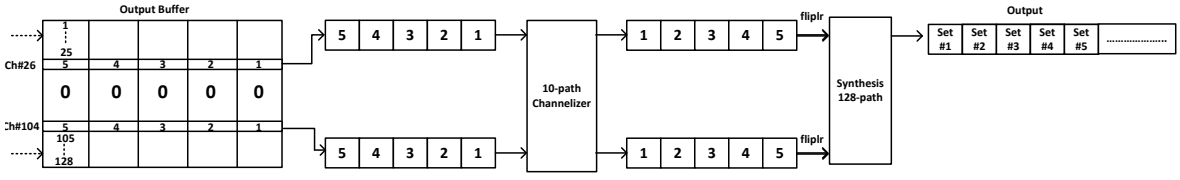


Figure 4.26. Loading 10-path analysis-synthesis channelizer from the 128-path output buffer.

We sit these two sets of outputs back in the output buffer of 128-path channelizer, and will send them along with the rest of the channel which are left on from selector, to the

128-path synthesis filter. The output of this step is shown in Figure 4.27. The first peak appears at point 1730, which is equal to the total processing time of a 128-path channelizer. Also there is a smaller peak at point 10434, which is the total processing time of a 128-path channelizer plus a 10-path channelizer. This difference proves us that the outputs of the 128-path analysis filter are processed to 128-path synthesis exactly 136 units earlier that the samples which are processed in the 10-path channelizer.

$$(10434 - 1730)/64 = 136 \tag{4.3}$$

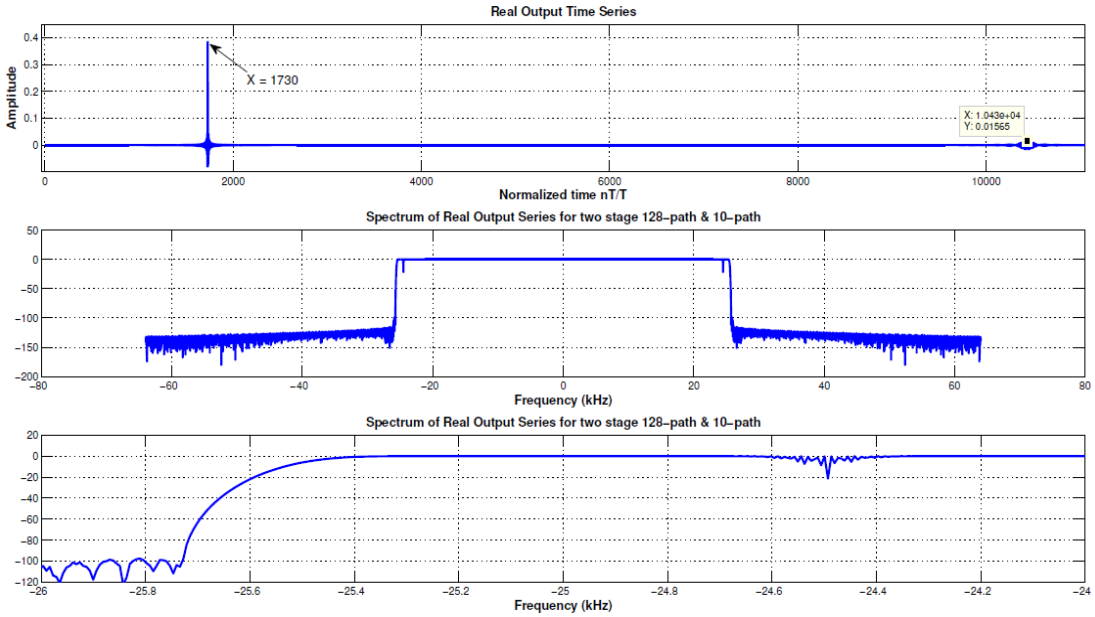


Figure 4.27. The current output of the sending 5 sets of 64 samples to the channelizer.

We conclude that we need to apply a 136 delay to the outputs of the 128-path analysis filter. This buffer is introduced after the output buffer of 128-path analysis. The outputs of the selector will be placed to this buffer and for each new sets, the buffer moves to the right by one column. Also the inputs of the 128-path synthesis are feed from this buffer. So instead of updating the outputs of the 10-path channelizer in the output buffer, we now update the buffer with these outputs. Another point that need attention is the correct size for the data. We know that each set of outputs need to move inside the buffer 136 times, so the correct size for the buffer seems to be 128×137, but since in this design we have 5 sets of data ready each time we perform the process, we need to add an extra 5 column to apply exactly 136 delay to all 5 sets of data. The correct size of this buffer is define as 128×141. The outputs of the 10-path

channelizer are updated inside the buffer from column number 137 to column 141, as shown in Figure 4.28.

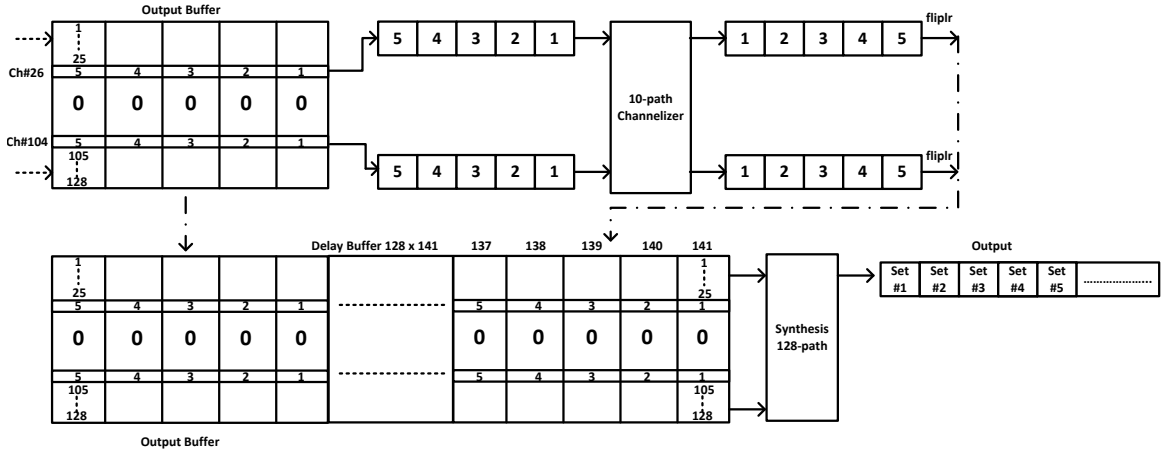


Figure 4.28. Introducing a delay buffer of 128X141 to delay the outputs of the input buffer.

The next step will be processing 5 sets of the data inside the buffer in the 128-path synthesis, and form the output signal. The result after applying these recent modifications is shown in Figure 4.29.

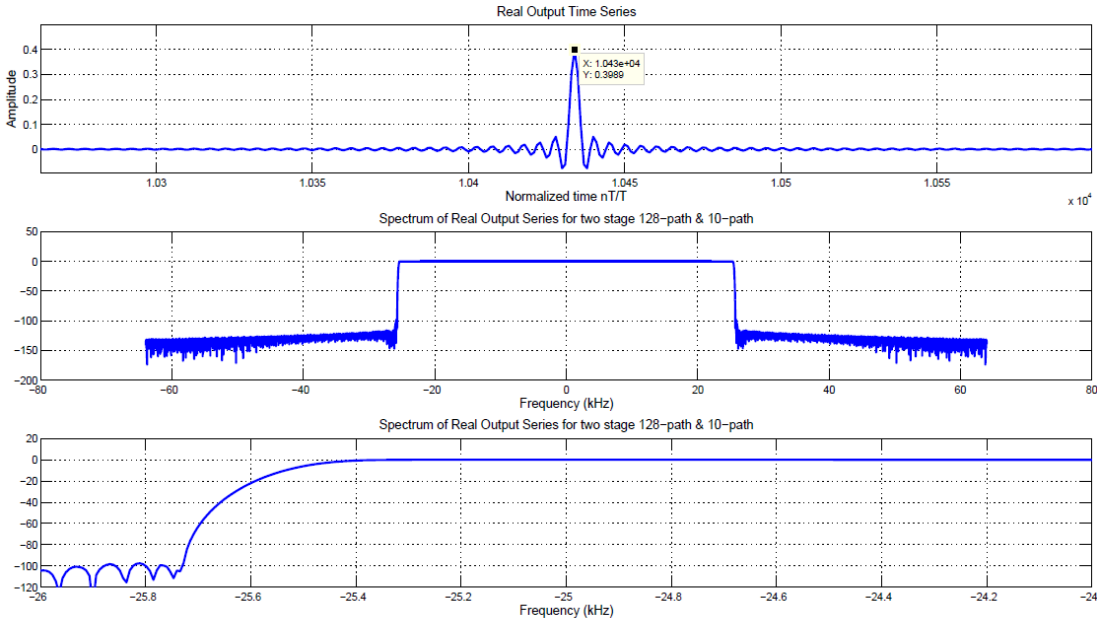


Figure 4.29. Final output of 128-path channelizser and 10-path channelizer.

Now it is time to perform two different configurations. The first configuration is to make all the band-edge channels off, and the second is to only make half of the band-edge channels off. The two different configurations are demonstrated in the Figure 4.30, and 4.31.

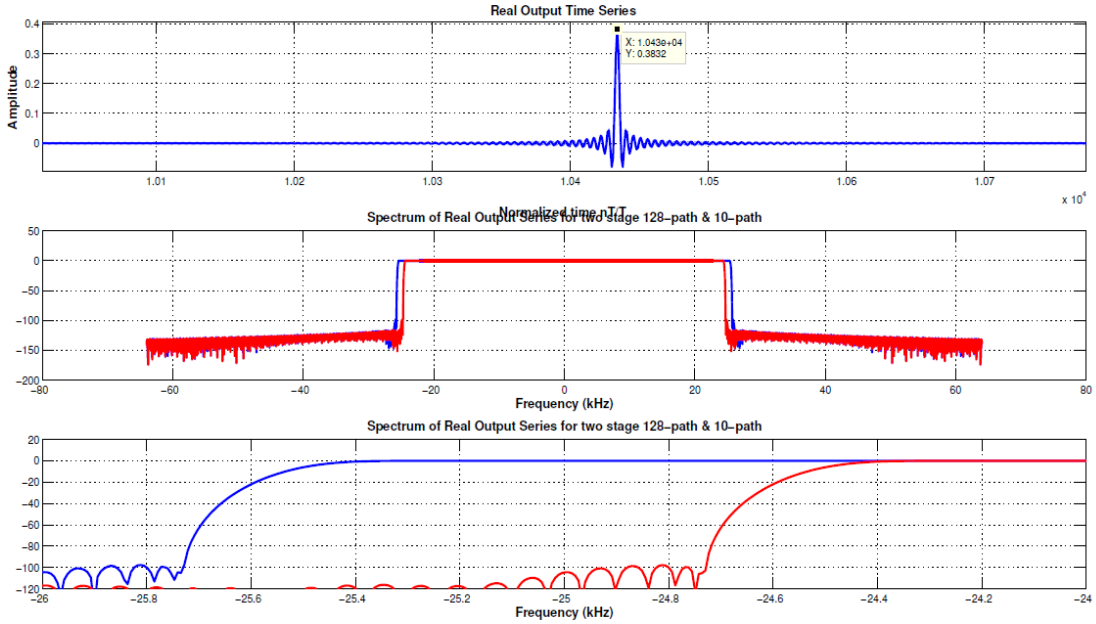


Figure 4.30. The final output of the 128-path and 10-path channelizer when the band-edge channels are off.

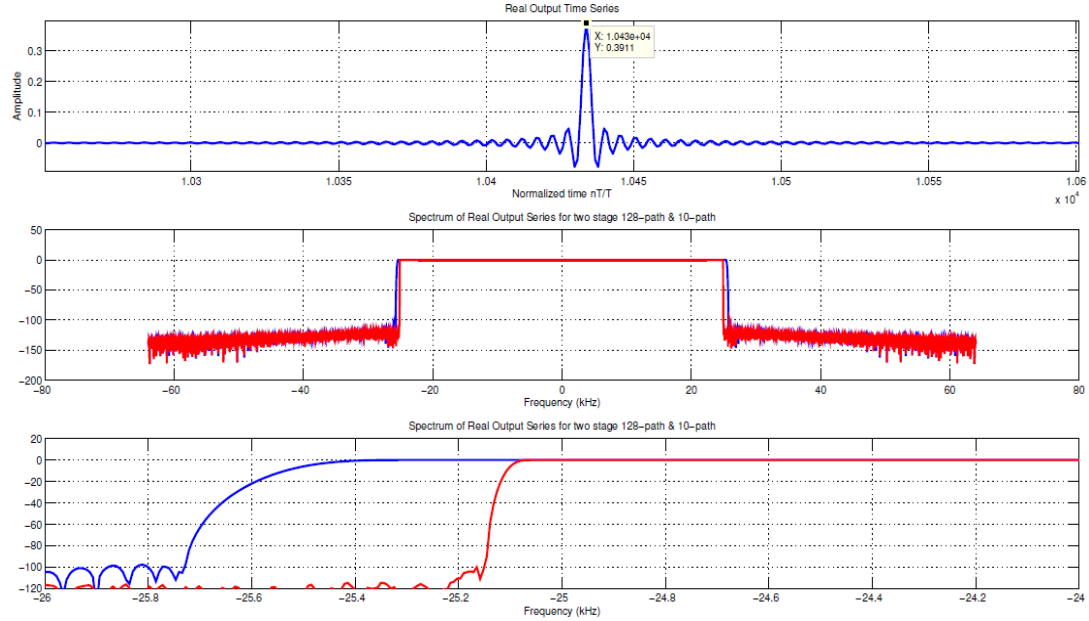


Figure 4.31. The output of the 128-path and 10-path channelizer when half of the band-edge channels are off.

This final design for proposed filter banks also has the capability of performing as a variable bandwidth filter. This design allows us to process different signals with different bandwidth without changing the filter coefficients or filter length. This variability has been shown in Figure 4.32.

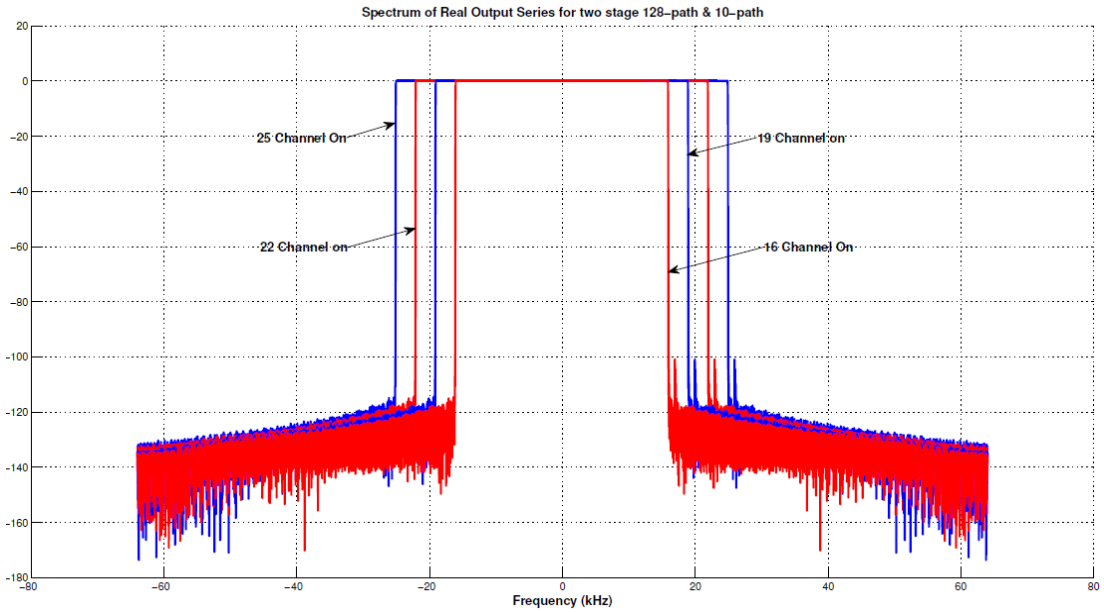


Figure 4.32. A demonstration of the filter banks with different bandwidth but fixed coefficients.

CHAPTER 5
CONCLUSION AND FUTURE WORK

5.1 CONCLUSION

We have proposed a novel architecture for variable bandwidth filter banks in this thesis study. The main contributions of each chapter are summarized as follows. Chapter 2 provides an overview of the multirate filters and standard M-path up convertor channelizer (modulator) and down converter channelizer (demodulator). In Chapter 3 the analysis and synthesis filters were studied. The enhancement of analysis and synthesis filters compared with the standard channelizer discussed. We have introduced the design of our variable bandwidth filter banks in Chapter 4. Our filter is based on cascaded two stage design. The first stage of filter is a 128-path channelizer filter and can extract the main bands from mixture of signals. The second stage of our filter design is a dedicated 10-path filter that can extract finer resolution of signals to enhance the quality of extractions. We design dedicated buffer in our analysis and synthesis filters to synchronize and align the arrival time of different channels at the receiver. The analysis results show that our proposed filter is capable of extracting variable bandwidth with different resolutions and minimum error successfully. We have studied multiple configurations and explained the characteristics of them in Chapter 4. The simulation results for different waveforms were illustrated in Chapter 4. Overall our proposed filter provides a base for extracting variable resolution bandwidths with fixed design which can be adopted in software defined radio architecture.

5.2 FUTURE DIRECTION

For future of our research we would like to pursue few directions based on our proposed variable bandwidth filter banks. First, we would like to optimize the structure of first stage and second stage filter design. Different combination of M-path and N-path parameters in each stage and combination of multiple filters such as sync and remez can be studied to reduce error further. The other aspect would be to optimize the design of the filter to save hardware and memory as much as possible. Finally application of our proposed variable bandwidth filter banks in different domains such as wireless communications would be a studied in future.

BIBLIOGRAPHY

[1] f. j. harris *et al.*, "Variable bandwidth m-path filter with fixed coefficients formed by m-path polyphase filter engines," in *20th European Conf. on Circuit Theory and Design*, Sweden, 2011, pp. 5–8.

[2] A. V. Oppenheim and R. W. Schafer, *Discrete-Time Signal Processing*, 3rd ed. Upper Saddle River, NJ: Prentice Hall Press, 2009.

[3] T. J. Rouphael, *RF and Digital Signal Processing for Software-Defined Radio: A Multi-Standard Multi-Mode Approach*. Newton, MA: Newnes, 2009.

[4] J. Mitola, "Software radios: Survey, critical evaluation and future directions," in *IEEE Aerosp. Electron. Syst. Mag.*, vol. 8, no. 4, 1993, pp. 25–36.

[5] f. j. harris *et al.*, "Design of pr nmdfb based up and down converters for transmit downlink and receive uplink of combined 3gpp lte and umts radios," *J. Analog Integr. Circuits Signal Process.*, vol. 1, no. 1, pp. 1–12, 2013.

[6] f. j. harris *et al.*, "Polyphase synthesis filter bank up-converts unequal channel bandwidths with arbitrary center frequencies," *J. Analog Integr. Circuits Signal Process.*, vol. 1, no. 1, pp. 281–293, 2012.

[7] f. j. harris *et al.*, "Digital receivers and transmitters using polyphase filter banks for wireless communications," *IEEE Trans. Microw. Theory Techn.*, vol. 51, no. 1, pp. 1395–1412, 2003.

[8] f. j. harris, *Multirate Signal Processing for Communication Systems*. Upper Saddle River, NJ: Prentice Hall PTR, 2004.

[9] E. Venosa *et al.*, *Software Radio, Sampling Rate Selection, Design and Synchronization*, ser. Analog Circuits and Signal Processing. New York, NY: Springer, 2012.

[10] f. j. harris, "Selectable bandwidth filter formed from perfect reconstruction polyphase filter bank," in *44th Asilomar Conf. on Signals, Sys. and Computers*, Pacific Grove, CA, 2010, pp. 1292–1296.

[11] f. j. harris *et al.*, "Wideband 160-channel polyphase filter bank cable tv channeliser," *Signal Processing, IET*, vol. 5, no. 3, pp. 325–332, 2011.

[12] f. j. harris *et al.*, "Polyphase analysis filter bank down-converts unequal channel bandwidths with arbitrary center frequencies," *J. Analog Integr. Circuits Signal Process.*, vol. 71, no. 3, pp. 481–494, 2012.

[13] f. j. harris *et al.*, "M-path channelizer with arbitrary center frequency assignments," in *Int.l Symp. on Wireless Personal Meultimedia Communication*, Brazil, 2010, pp. 5–8.

[14] f. j. harris, "Fixed length fir filters with continuously variable bandwidth," in *1st Int. Conf. on Wireless Comm., Vehicular Techno., Info. Theory and Aerosp. Elect. Syst. Techno.*, Denmark, 2009, pp. 931–935.