Jennifer P. Smith

Overlapped Polyphase Filter Bank

Motivation

Many signal processing technologies require separating and monitoring signals of various frequencies. The simplest way to accomplish this channelization process is to apply an N-point Discrete Fourier Transform (DFT) which outputs the amplitude and phase of N different frequency sine waves that could be summed to recreate the original time stream with minimal error. The underlying DFT math involves wrapping the time stream on itself and evaluating the inner product of the timestream with all N frequency sine waves. Problems can arise if the timestream is not periodic on the boundary because the wrapping step introduces a discontinuity which diminishes the signal to noise ratio in the DFT output[1] (see figure 1). This undesirable behavior can be mitigated by applying a filter to the timestream which forces periodicity on the boundary. The polyphase filter bank is simply an architecture which efficiently implements this filter.

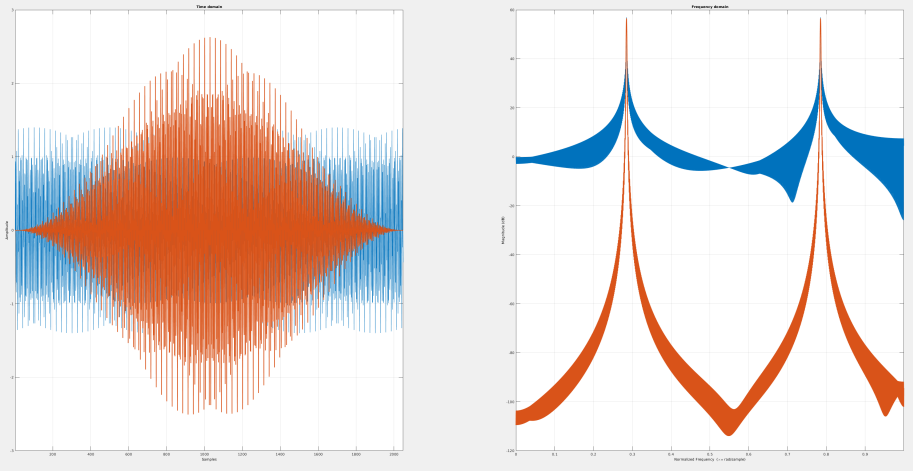

Figure 1: Left: Time-domain representation of a superposition of two frequencies shown with a sine envelope windowing function applied (red) and without (blue). Right: DFT output of the left time stream. The signal to noise is vastly improved in the windowed time stream (red).

Multirate Channelization

Attempting to digitally filter incoming data at the rate it is being acquired by an Analog to Digital Converter (ADC) is both inefficient and unnecessary. This is because in many signal processing and scientific applications, the signal of interest is a modulation of a high frequency signal. For example, illuminating a 1GHz MKID with a laser source yields about 100 counts/second. The Nyquist Sampling Theorem implies the initial timestream must be acquired at 2*1GHz=2GSPS. However, the signal can immediately be translated (in frequency space) to 0Hz, lowpass filtered, and sampled at the reduced rate 2*100Hz=200Hz. This allows all downstream processing to run at the reduced rate which greatly reduces complexity and computing resources. This process of translating, filtering, and down-sampling is ubiquitous in channelization schemes and is shown in figure 2.

[1] This is known as the Gibbs Phenomenon

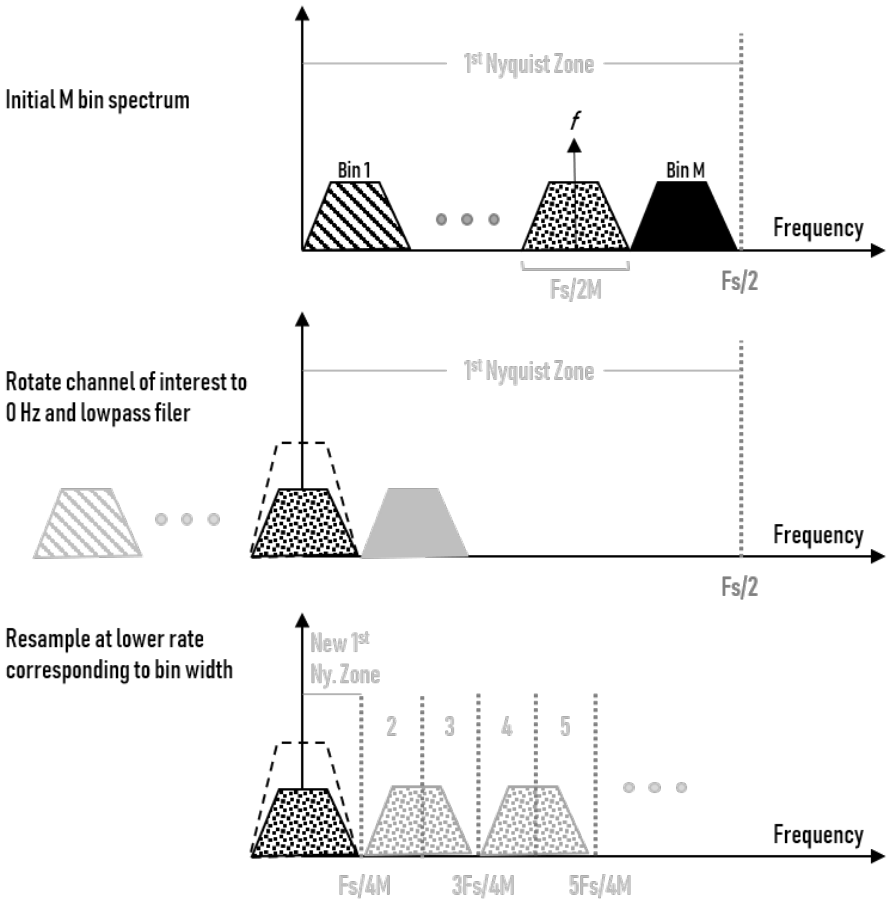

Figure 2: Basic channelization scheme. The bin of interest (centered at frequency f) is shifted to 0 Hz. A Low-pass filter is applied to discard the other bins. The new spectrum is re-sampled at the new Nyquist rate. Images of the bin of interest appear in higher Nyquist zones but can be ignored.

Digital Filtering

The simple cosine envelope windowing function referenced previously is an example of a linear time-invariant (LTI) filter. The relationship between a LTI filter input $x(n)$ and output $y(n)$ is described by the convolution sum with the filter's impulse response $h(n)$,

$$y(n) = (x * h)(n) = \sum_{k=0}^{M} h(k)x(n-k) = h_0x(n) + h_1x(n-1) + ... + h_Mx(n-M). \quad (1)$$

Here M represents the order of the FIR filter and the impulse response is an $(M+1)^{th}$ length one-dimensional array that can be expressed as $\boldsymbol{h} = [h_0, h_1, ..., h_M]$. Figure 3 shows how equation 1 can be implemented in hardware using M delays and an adder. Here Z^{-1} means "delay by one sample" so $x(n)Z^{-1} \rightarrow x(n-1)$.

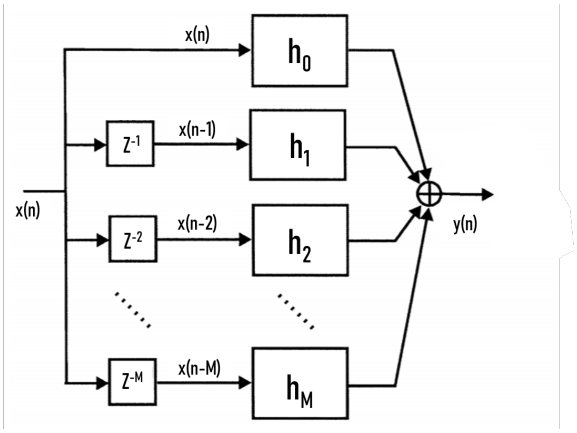

Figure 3: Direct implementation of an M-order FIR filter using a M-path partition. The h0,..,hm filter impulse response coefficients multiply the delayed signal before being added to create the filtered output y(n). Figure modified from [2, 3].

Polyphase Filter Bank

The polyphase filter bank (PFB) efficiently implements the spectral translation, filtering, and re-sampling described in the channelization section (figure 2). The efficacy of the design relies on the "Nobel Identity" of Multirate Digital Signal processing which states: "The output from a filter H(Z^M) followed by an M-to-1 down sampler is identical to an M-to-1 down sampler followed by the filter H(Z)" (figure 4) [2]. In this context, "M-to-1 down sample" means use only every Mth sample.

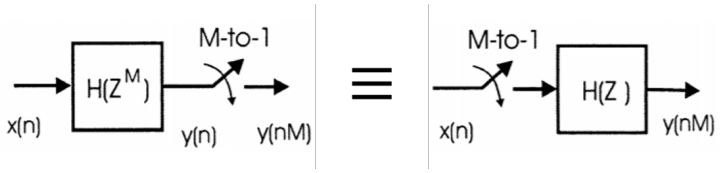

Figure 4: Noble identity: A filter processing every Mth input sample followed by an output M-to-1 down sampler is the same as an input M-to-1 down sampler followed by a filter processing every Mth input sample. Figure reproduced from [2].

Moving the down sample operation ahead of the filter as shown in figure 4 results in a more efficient design. This change requires modifying the filter decomposition so that it operates on every Mth sample. This is known as a polyphase decomposition. To understand polyphase decomposition we will start with the more familiar mathematical description of a polynomial filter H as a sum of delayed polynomials in Z^M.

$$H(Z) = h(0) + h(1)Z^{-1} + h(2)Z^{-2} + ... + h(N-1)Z^{-(N-1)}. \quad (2)$$

Here N represents the length of data samples and must be an integer multiple of the FFT size. This sum can be rearranged into a sum of sums which maps the one dimensional array of weights and index markers Z^{-n} to a two dimensional array as shown in equation 4 [3].

$$H(Z) = \sum_{r=0}^{M-1} Z^{-r} \sum_{n=0}^{(N/M)-1} h(r+nM)Z^{-Mn} = \sum_{r=0}^{M-1} Z^{-r} H_r(Z^M) \quad (3)$$

$$H(Z) = \begin{array}{llllll} h(0) & + & h(M+0)Z^{-M} & + & h(2M+0)Z^{-(2M+0)} & + \ \dots \\ h(1)Z^{-1} & + & h(M+1)Z^{-(M+1)} & + & h(2M+1)Z^{-(2M+1)} & + \ \dots \\ h(2)Z^{-2} & + & h(M+2)Z^{-(M+2)} & + & h(2M+2)Z^{-(2M+2)} & + \ \dots \\ h(3)Z^{-3} & + & h(M+3)Z^{-(M+3)} & + & h(2M+3)Z^{-(2M+3)} & + \ \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \\ h(M-1)Z^{-(M-1)} & + & h(1M-1)Z^{-(2M-1)} & + & h(3M-1)Z^{-(3M-1)} & + \ \dots \end{array} \quad (4)$$

This mapping is also used in the Cooley-Tukey fast Fourier transform (FFT) and allows the array to be loaded by columns but processed by rows [2]. The first row is a polynomial in Z^M and will be denoted $H_0 Z^M$ which means begin at index 0 and increment by M. Factoring Z^{-1} from the second row of the array also yields a polynomial in Z^M meaning the second row of the array corresponds to $Z^{-1}H_1(Z^M)$. In this way, the r^{th} row of the array can be described by $Z^{-r}H_r(Z^M)$. This notation can be used to recast (4) in a way that can be implemented similar to the FIR filter shown in figure 3 [2],

$$H(Z) = H_0(Z^M) + Z^{-1}H_1(Z^M) + Z^{-2}H_2(Z^M) + ... + Z^{-(M-1)}H_{(M-1)}(Z^M). \quad (5)$$

A schematic hardware realization of this mathematical form is shown in figure 5. The nobel identity allows the downsample operation to be moved ahead of the filter. The input path delays and resample operation can be realized by a commutator as shown in figure 5, right [2, 3]. The commutator achieves the desired result by switching the memory addresses of the appropriate data points.

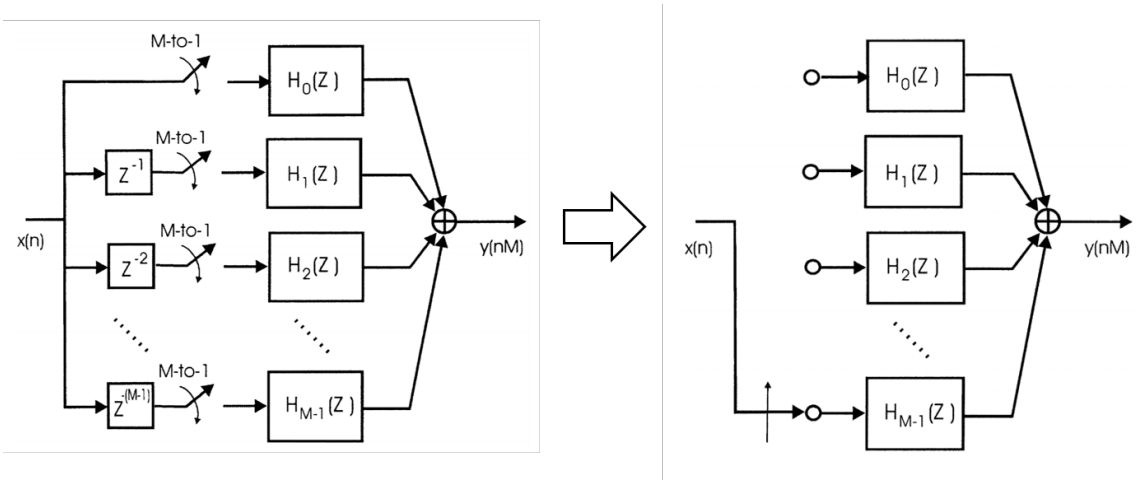

Figure 5: Left: M-path polyphase partition of a filter. Right: Equivalent design with input path delays and re-sampler replaced by a commutator. Figure modified from [2, 3].

The commutator implements an M-to-1 downsample operation which breaks the Nyquist criterion and allows M-multiples of the output sample rate to alias down to 0Hz, effectively translating each of the M bins to 0Hz. As each successive sample comes in, the phase of the center frequency of the kth bin rotates by 2πk/M [1, 2, 3]. The aliased center frequencies incur an additional phase delay relative to one another which is the product of the center frequency and the path delay Z^{-r} shown explicitly in figure 5, left. It is these branch dependent, differential phase effects which allow the bin centers to be separated and monitored after all aliasing to the same spectral zone.

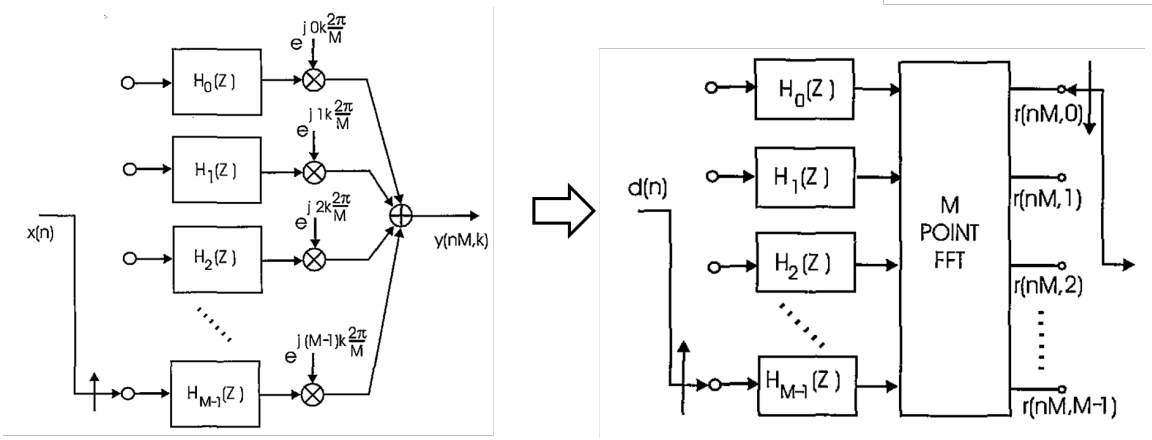

Figure 6: Polyphase channelizer architecture. Figure modified from [2, 3].

The separation of the aliases is accomplished with the addition of phase rotators (figure 6, left) which derotate and align the phase shifts from the kth center frequency present at the successive ports of the input commutator [3, 1, 2]. When the branches are summed as shown in equation 6, the aliased terms which were not explicitly aligned destructively cancel. Specifically, this process of aligning the bin centers and summing the result is an example of a phase coherent summation and is identical to applying an M-point FFT. Note that if the direction of the commutator is reversed, the phase rotators incur a minus sign and the phase coherent summation is identical to applying an M-point IFFT. The process of aligning the bin centers and summing the result can be efficiently implemented by applying an M-point FFT/IFFT to the vector of M time samples produced from the M polyphase paths at time nM [2, 1]

$$y(nM,k) = \sum_{r=0}^{M-1} y_r(nM) e^{\pm j\frac{2\pi}{M}rk}. \tag{6}$$

The polyphase channelizer can be heuristically understood as a window filter which is moved through the full frequency bandwidth by the FFT. There is a relationship between the size of the FFT, the output bin sample rate, and the bandwidth of the window filter. In the above design, the input and output commutators are running at the same rate set by the sampling frequency Fs and the FFT size M. The M-point FFT operating on Fs bandwidth will yield M frequency bins each Fs/M wide which are Nyquist sampled by the output commutator at Fs/M[2]. Typically, the window filter will be designed so frequencies greater than the half FFT bin width (Fs/2M) are highly attenuated (figure 7). This prevents high frequencies from aliasing into the bins. This design is ideal for receiver-type applications where the goal is to minimize bin cross talk and ensure the received signals are as clear as possible. This high quality approach comes at the cost that any signal falling in the transition bandwidth will be highly attenuated. The design engineer will know the approximate transmission frequencies and can ensure they do not fall in a transition zone.

This design is not suitable for spectrum analyzer type applications where the user has minimal control over where the incoming tones will fall. In this situation, the filtering window must be widened so that no matter where a tones falls in the spectrum, it may pass through a bin with minimal attenuation (figure 7). This bin overlapping design introduces a problem where the filter window is larger than the FFT bin, meaning frequencies greater than half the bin sample rate may aliase into the FFT bins. This problem can be solved by increasing the bin sample rate. This modification is the subject of the next section.

[2]In this discussion complex sampling is assumed. In this regime, sampling the real and imaginary parts of the signal at Fs yields Fs bandwidth extending from -Fs/2 to Fs/2. The individual bins are centered on 0 Hz so they will only encompass frequencies $\pm$ Fs/2M, meaning the Nyquist bin sample rate is Fs/M.

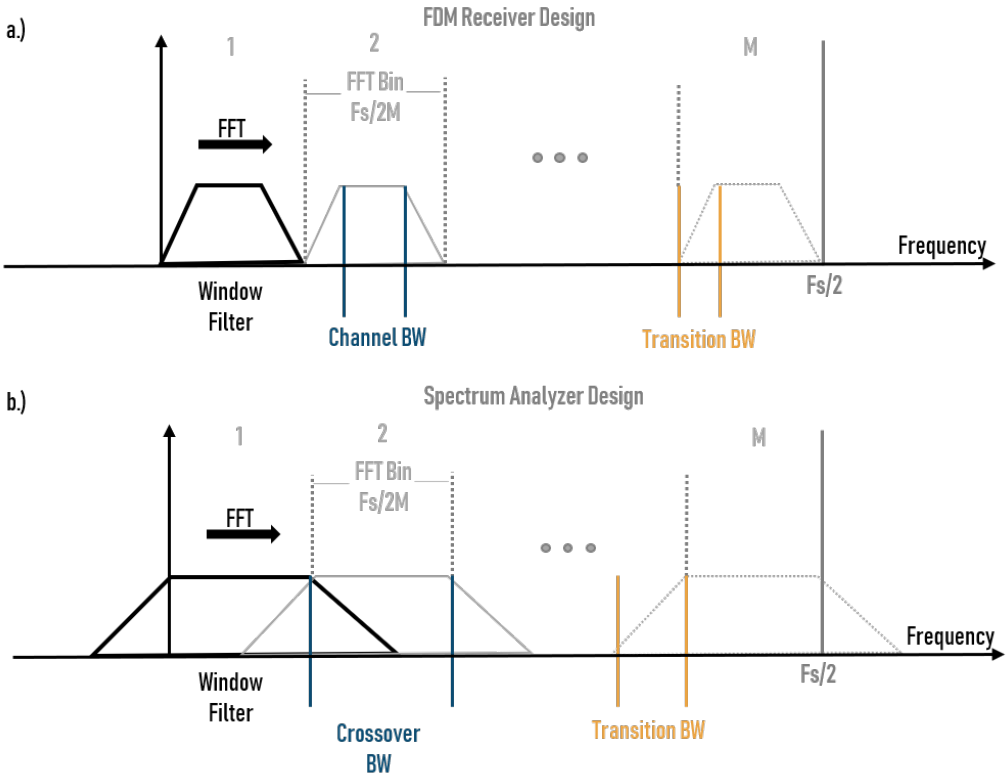

Figure 7: Example filter window size in comparison to the FFT bin size for a FDM receiver design (a.) and spectrum analyzer design (b.)

Overlapped Polyphase Filter Bank

The polyphase channelizer presented in the previous section delivers M input samples along the M branches and computes the output for each channel at a rate Fs/M. In this section, the design is modified to compute M output samples after only D (D<M) inputs have been loaded. This results in the FFT bins being over sampled by a factor of M/D, meaning they can be widened by a factor of M/D without risk of aliasing. The trick is to recycle some of the previously loaded data much in the same way a moving average function can compute an average of many data points after only a few new points have been loaded. The ratio of output to input sample rate of the polyphase channelizer can be made to be any rational number by recirculating the data as next described [3, 6].

In the non-overlapped structure, successive input samples are first delivered to port M-1 and progress upwards to port 0 (figure 6). To double the output sample rate, M/2 successive input samples are delivered to port (M/2)-1 and progress up to port 0. When the the next M/2 samples come in, the previous samples snake through the 2-D polyphase filter array [3, 2, 5]. This is equivalent to applying a linear shift to the 1-D filter expressed in equation 2 prior to the polyphase decomposition. Figure 8 (top) shows a schematic representation of the 2-D polyphase decomposition from equation 4. The bottom of figure 8 explicitly shows the serpentine movement for the case of M/D=6/4. This architecture can be implemented in hardware by removing the unused input ports from the commutator schedule and adding a buffer which swaps the appropriate memory addresses [3].

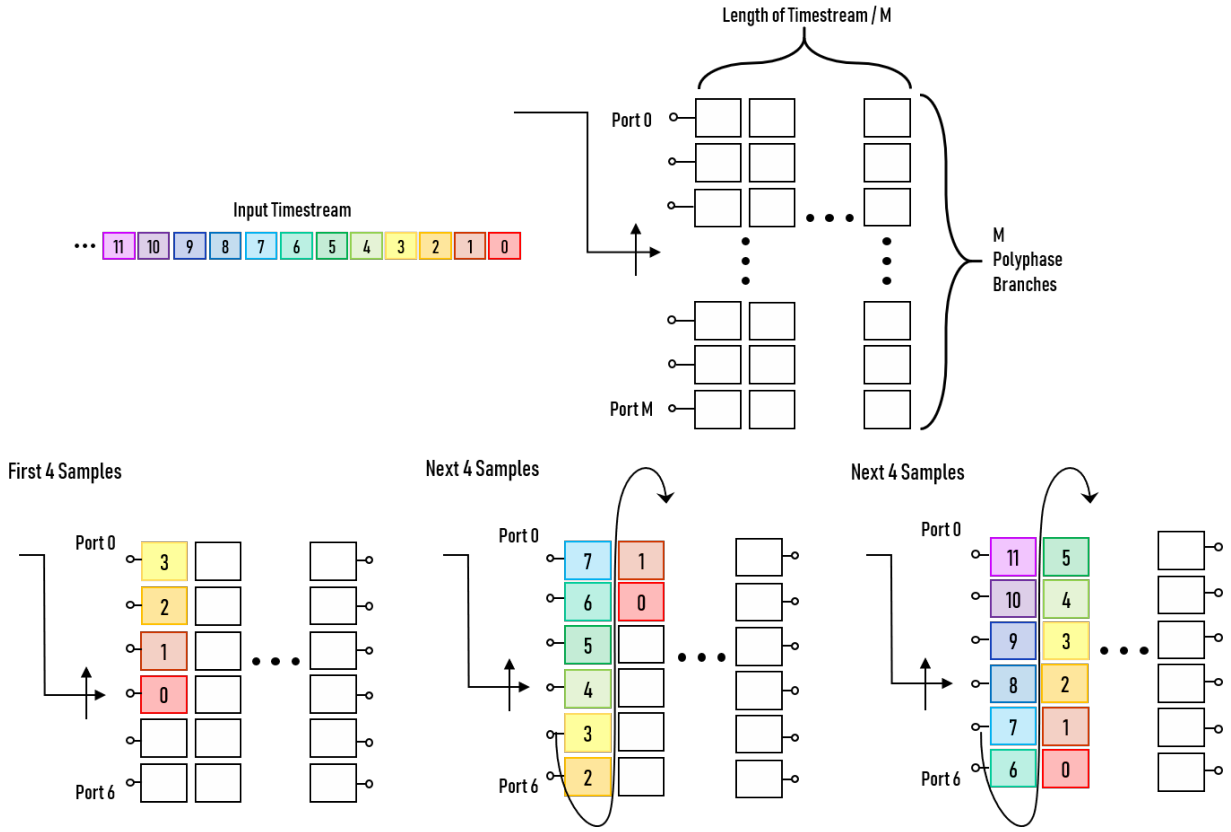

Figure 8: Successive loading of data samples and serpentine movement of data through the 2D polyphase filter for M/D=6/4. The input commutator runs only from port 4 up to port 0, then resets to port 4.

So far, two changes have been made to the original polyphase filter bank. First, the commutator was modified to start at port D-1 and progress upwards and second, a shift buffer was added. A final, third modification is needed to resync the data going into the IFFT. Because the data windows length D are smaller than M, the origin of the IFFT is offset from the origin of the data by N different offsets where N is the numerator of the reduced fraction D/M. A circular shift buffer with N different states reorders the data before the IFFT. The state changes each time a new frame of D data points is loaded into the polyphase filter. This can be realized in hardware with a simple state machine. The amount the data going into the IFFT needs to be circulated by is given by [5]

$$R(n) = (nD) \mod M,\ n = 0, 1, ..., N-1. \tag{7}$$

For the example shown in figure 8 with M/D=6/4, there are 3 shift states: [0, 4, 2]. Data movement from the polyphase filter through the circular shift buffer and into the FFT is shown in figure 9.

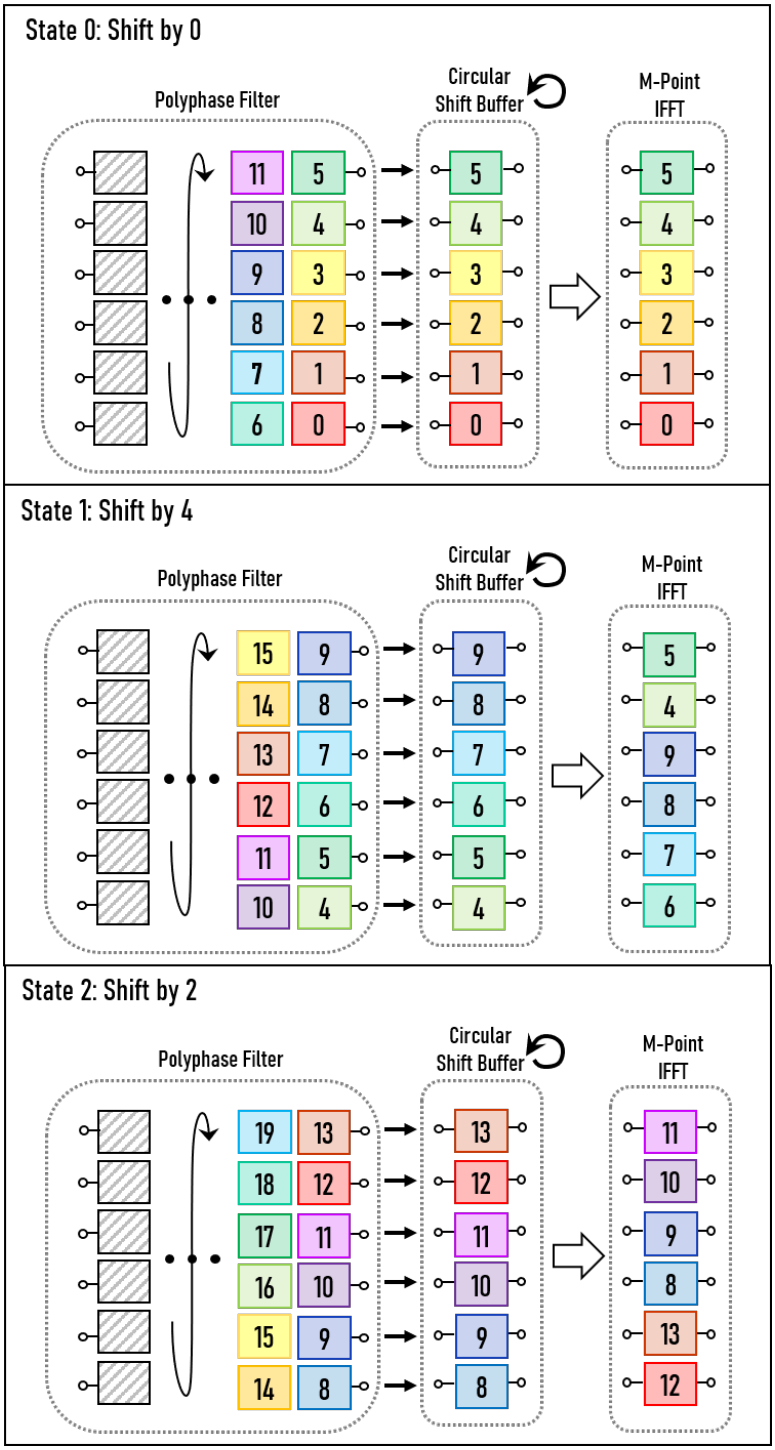

Figure 9: Data movement from the polyphase stage to the IFFT through the circulating shift buffer for each possible shift state. Notice the shift corrects the frame-offset between states so that the two recycled samples are input to the same IFFT bins. The shift state increments by one after a new 4-long data frame enters the polyphase stage. The state wraps back to state 0 after state 2.

Combining all of the modifications discussed above, a sample block diagram for a standard PFB and PFB with 50% bin overlap is shown below.

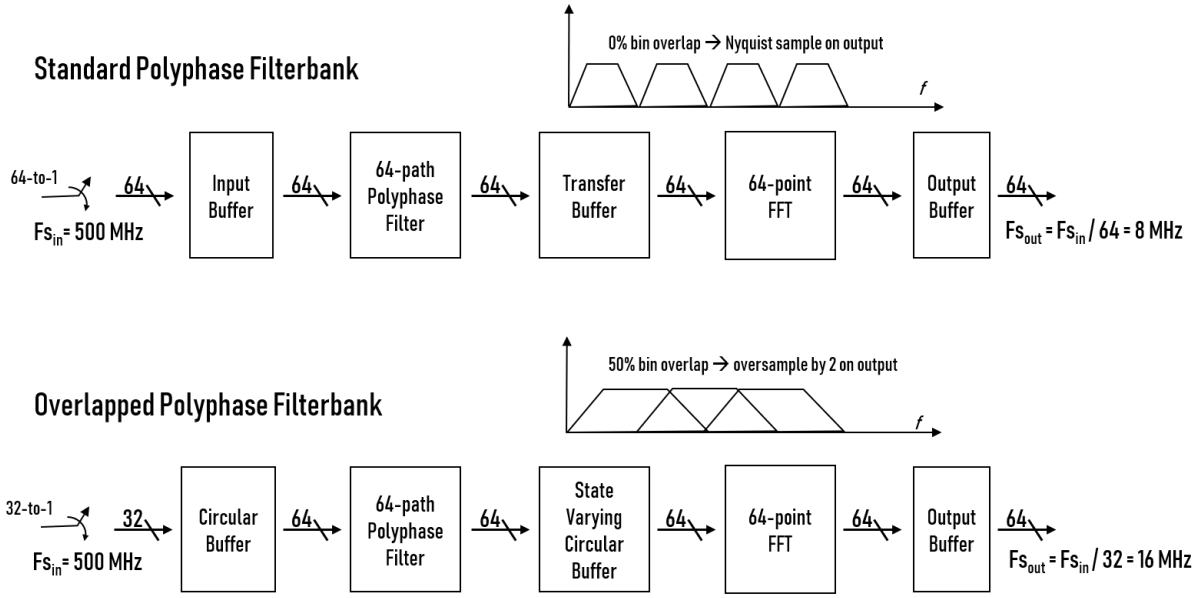

Figure 10: Block diagrams showing a standard polyphase design (top) and an overlapped polyphase design with 50% bin overlap (bottom).

Summary

This document summarizes how the polyphase filterbank is able to efficiently channelize an incoming spectrum. The key simplification involves using the Nobel identity of multirate digital signal processing to pull the downsample operation ahead of the filter. Consequently, the filter must be modified from a 1D array to a 2D array in what is known as a polyphase decomposition. Other channelization schemes include a per-channel approach, which applies a different filter to each channel, and a tree structure, which uses a binary tree of down conversions to continually halve the spectrum into individual channels. These methods were not discussed in this document for brevity's sake but a comparison of their resource utilization when implemented on an FPGA was characterized by [4] and is reproduced here. The table confirms the polyphase filterbank channelizer is more efficient than other channelization schemes.

Channelization Method	Number of Channels	LUTs	Memory (bits)
Per-Channel	256	317,498	436,224
	512	650,114	876,544
	1024	1,336,754	1,761,280
Tree Structure	256	27,930	3,840
	512	32,270	6,529
	1024	36,610	10,625
Polyphase Filterbank	256	4,608	4,608
	512	4,793	4,793
	1024	5,345	5,345

Table 1: FPGA resource utilzation comparison between the per-channel, tree structure, and polyphase filterbank approaches for varying numbers of channels. The polyphase filterbank is the most resource-efficient design.

References

[1] B. Farzad. Variable bandwidth polyphase filter banks. Master's thesis, San Diego State University, San Diego, CA., 2014.

[2] F. J. Harris. Digital receivers and transmitters using polyphase filter banks for wireless communications. *IEEE Transactions on Microwave Theory and Techniques*, 51(4), 2003.

[3] F. J. Harris. *Multirate Signal Processing For Communication Systems*. Prentice Hall, Upper Saddle River, NJ, 2004.

[4] J. Lillington. Comparison of wideband channelization architectures. Master's thesis, CTO RF Engines Limited, Newport, Isle of Wight, UK.

[5] J. Tuthill, G. A. Hampson, J. D. Bunton, and F. J. Harris. Compensating for oversampling effects in polyphase channelizers: A radio astronomy appplication. pages 255–259, 2015. doi: http://dx.doi.org/10.1109.

[6] F. Wu. Fpga based uniform channelizer. Master's thesis, National University of Ireland Maynooth, Maynooth, Ireland, 2016.