Guillaume-Jean Herbiet
GTID : 902141603
gjh@gatech.edu

ECE 6604: Personal and Mobile Communications
Course project

Design issues and challenges in
sample rate conversion
for Software Defined Radio systems

Abstract

A software-defined radio (SDR) system is a radio communication system which can tune to any frequency band and receive any modulation across a large frequency spectrum by means of a programmable hardware which is controlled by software.

An SDR performs significant amounts of signal processing in a general purpose computer, or a reconfigurable piece of digital electronics. The goal of this design is to produce a radio that can receive and transmit a new form of radio protocol just by running new software.

Software radios have significant utility for the military and cell phone services, both of which must serve a wide variety of changing radio protocols in real time.

After defining the purposes and applications of software defined radio, particularly in the field of tactical communications, the report will describe the general architecture of SDR and key design issues due to the large spectrum and variety of signals that has to be analyzed.

In a last part, computational-optimized methods to perform sample rate conversion so as to adapt the sample rate of the received digital signal to those required for channelizing and synthesizing and their solutions are presented.

Table of Contents

Index of Illustrations

Index of Tables

I. Introduction

The recent decades have seen a large increase in the number of analog or digital communication standards being defined, both in the civil and military context. This multiplication of definitions makes even harder the task to establish common worldwide standards and future mobile systems, due to the competition between Asia, Europe and America, are likely to use different communication protocols.

On the other hand, today's wireless services are more and more ubiquitous and the global communication infrastructure requires them to be more flexible and reconfigurable so as to offer complements or at times completely substitute to wired communications, using the proliferation of services offered over satellites, cellular networks and other wireless WANs or LANs.

This part describes some key features and benefits of the software radio concept along with an example of application in the field of tactical communications.

1) *Software defined radio goals and features*

In this context, the concept of software defined radio appears as a potential pragmatic solution so as to achieve interoperability between standards, while using a software implementation of the user terminal enabling a dynamic adaptation to the radio environment and standards in use at that time and for the current communications.

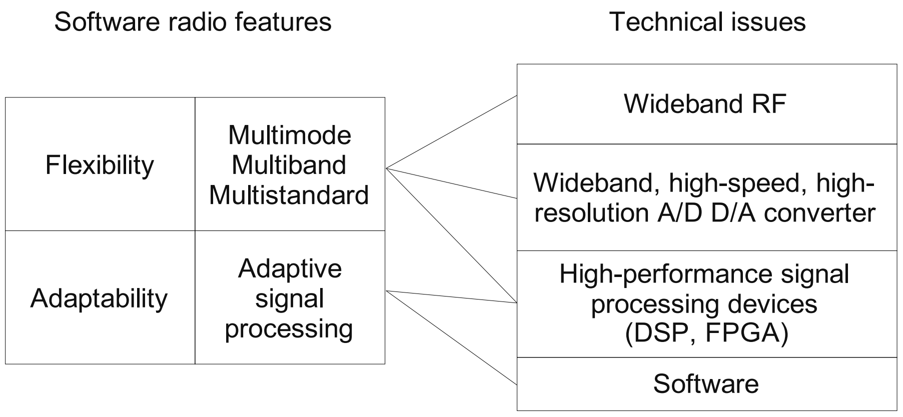

Software radio features
Technical issues
Flexibility
Multimode
Multiband
Multistandard
Adaptability
Adaptive signal processing
Wideband RF
Wideband, high-speed, high-resolution A/D D/A converter
High-performance signal processing devices (DSP, FPGA)
Software

Illustration 1: Software radio key features and related technical issues

Therefore, software radios implications are an increased ability to tolerate and support interoperability across heterogeneous air interface technologies, a better support for network upgrades and a substitution of general-purpose hardware to particular waveform-dedicated components. Besides, as those techniques improve the management of channel congestion and allow the use of a more flexible spectrum usage model.

2) *A favorable context for the development of software radio*

Moving to the concept of software radio allows an increased flexibility (in terms of customization, evolution and even a faster time-to-market) at lower costs . But other factors are pushing for software radios to be realized in commercial markets and are described in this paragraph.

Recent advances in hardware technology, smart antennas, adaptive power management and modulation and signal processing techniques make software designed radios feasible, despite the major design issues that remain (which are described further in this document). Those steps forward are welcome as an answer to the multiplicity of communication standards (due for example to different spectrum allocation in different countries).

These developments are perennized by commercial market opportunities. The military has been the first field of use for software radios but today the development of new wireless devices and the associated multimedia services providers to mix different media for delivering different types of service and create a new market for this concept.

3) An example of application: tactical networks

Software defined radio systems are reckoned as the key component for radio communication in the tactical domain of "network centric warfare". The concept of software radio offers interoperability during joint and combined operations while unities may be using different radio systems from different suppliers, using new radio technologies or legacy systems with a frequency range from HF to UHF.

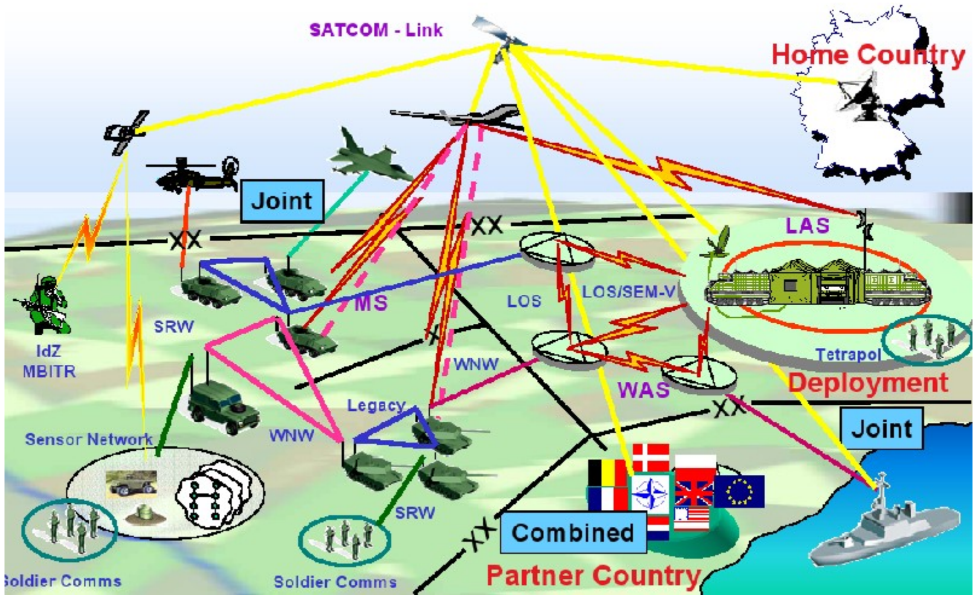


Illustration 2: Info-centered warfare

Besides, software designed radios allow support for high data transmission capabilities by using high data rate waveforms (WNW) and for integrated networking functions of the radio nodes that are key enabler to build tactical mobile ad-hoc radio networks (MANET).

II. The software radio architecture

As stated before, the software radio concept alters traditional radio designs in three distinct and complementary ways:

- It moves the analog/digital conversion process as close to the receiving antenna as possible, making to apply the advances of digital computing sooner in the radio;
- It substitutes software for hardware processing;
- It facilitates a transition from dedicated to general purpose hardware;

These paradigms enable some of the key features of software defined radios, such as :

- Flexible TX/RX architecture, controlled and programmable by software using downloadable upgrades at every layer of the protocol stack;
- Use of signal processing to replace as much radio functionalities as possible;
- Dynamic definition by software of frequency band, radio channel bandwidth, modulation and coding scheme, radio resource and mobility management schemes at the transmitter.

The remainder of this part, after describing the legacy radio architecture, will introduce the ideal and concretely achievable structures of software designed radios.

1) Traditional transceiver architecture

In traditional, or super-heterodyne, transceivers, the RF (radio frequency) and IF (intermediate frequency) stages are totally analog, the only digital operations being located in the BB (base band stage) and performed by ASIC-based technologies.

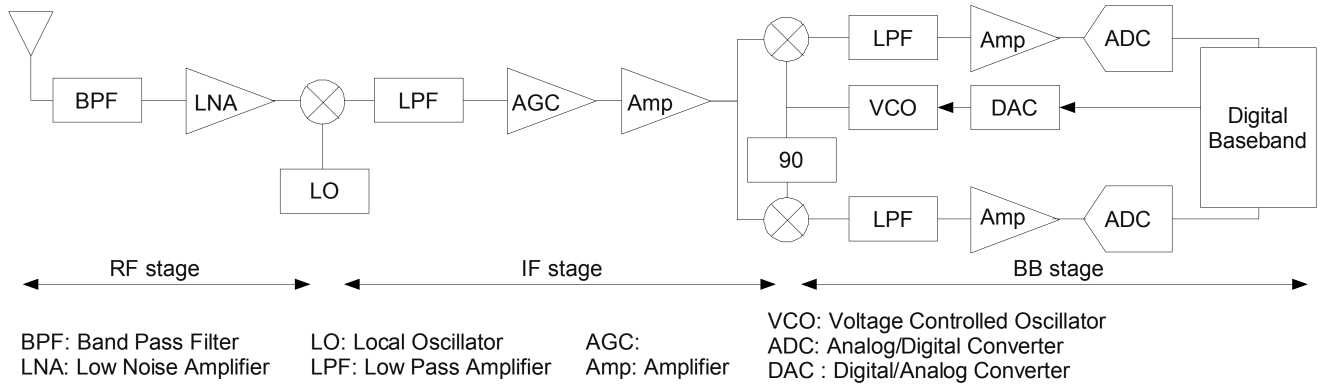


Illustration 3: Legacy super-heterodyne receiver

Those transceivers, often called “hardwired-looking” terminals , even built with high-performance signal processors, have a front-end which parameters such as channel spacing/bandwidth and carrier frequency could not be changed by means of software. Therefore, a digital font-end to replace analog digital processing is required to enable the possibility to softly reconfigure the transceiver and enable the implementation of different air interfaces on a given hardware platform.

An ideal architecture for such a software radio receiver and achievable structure are presented in the next two paragraphs.

2) *The Ideal Software Radio receiver*

As opposed to legacy architectures, the ideal scheme of a software radio transceiver has a very reduced analog stage, the only analog components being the antenna, the bandpass filter and the low noise amplifier. Besides, the analog to digital conversion is operated at the RF level so as to digitally elaborate the signal on a completely reprogrammable board, built with a DSP-based technology.

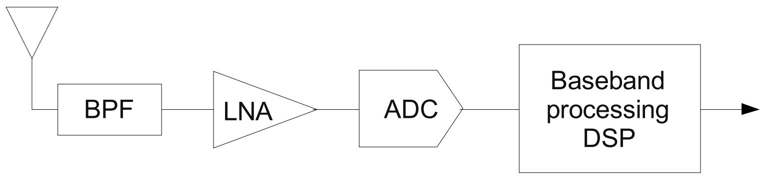


Illustration 4: Ideal Software Radio receiver

In this scheme, channelization operations are performed digitally and the analog to digital converter (ADC) therefore has to process the signal bandwidth for which the terminal has been designed. Considering the characteristics of communication channel, such as fading and shadowing, and the very likely existence of strong interfering signals, the signals received have a very dynamic range.

The analog to digital converter, is the key component of a software radio terminal and its design should be efficient so as to be able to follow dynamic ranges far above that with which conventional terminals have to cope, as it has to process a large number of channels simultaneously, the channel selection function being shifted from the analog to the digital domain. As a example, the GSM system itself has a dynamic range of about 100 dB and the extension to the simultaneous reception of other standards makes this number even higher.

To improve the the dynamic range of ADCs, a possible solution is to increase the number of bits quantized and/or the oversampling ratio, which is defined as the ratio of the sample rate and the channel bandwidth of the currently received signal spectrum. However, oversampling only yields a small gain in dynamics range and even the more efficients ADCs, that perform nearly 100 million samples per second (MSPS) with a 14 bit quantization, would not achieve well enough.

Besides the dynamic range, the sample rate also has to fulfill the Nyquist criterion. For a bandwidth W to be digitized, the sampling rate f must at least be twice W, and in practical systems $f > 2.5W$ is typically achieved. This fact thus limits the bandwidth that can be digitized in practice.

3) *A Feasible Software Radio receiver*

A consequence of the issues described above is that the ideal software radio architecture is today not achievable. However, the bandwidth the ADC has to digitize and the digital front-end has to process can be reduced, using partial band digitization.

To achieve this concept, a limited band is selected out of the full band of the received spectrum, by means of analog conversion and intermediate frequency (IF) filtering. This result in an architecture using intermediate frequency sampling, as in legacy transceivers.

In this scheme, the digital front-end, or Programmable Downcoder (PDC), is responsible for channelization and sample rate conversion (SRC). The latter comes from the fact that it is sensible to sample the analog signal at a fixed rate, whatever the waveform, and simplifies clock generation for the ADC. The first performs all tasks necessary to select the channel of interest, including conversion to baseband, channel filtering and despreading when required.

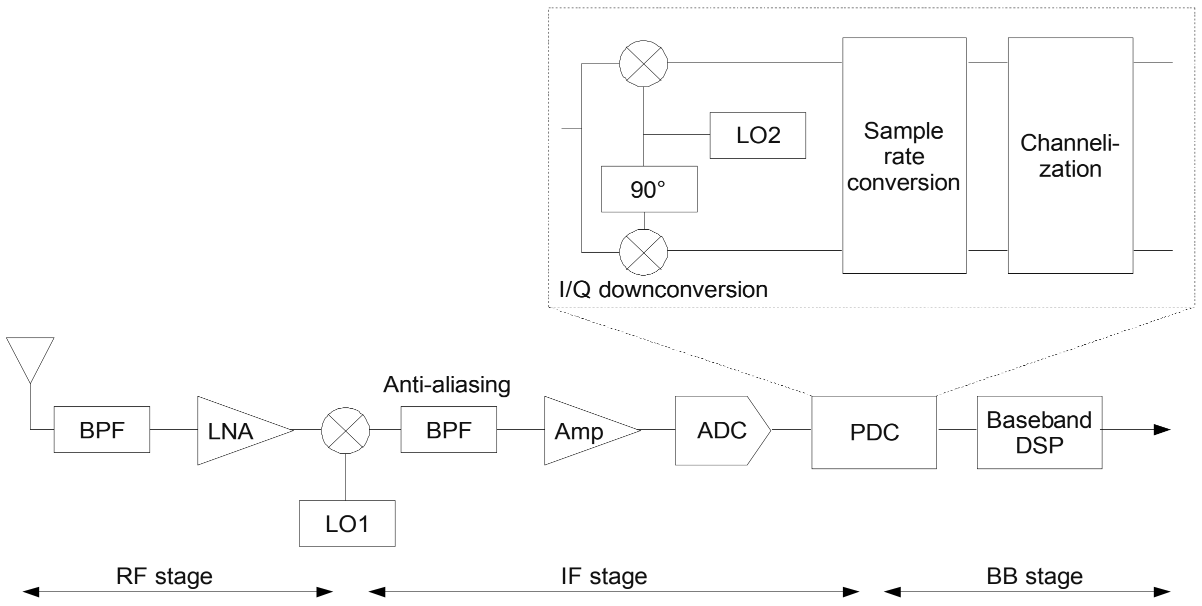


Illustration 5: Practical software radio receiver architecture

Limiting the bandwidth to be digitized is important as, for a fixed digitization bandwidth, i.e. the sample rate and the anti-aliasing filter are fixed whatever the current standard of operation, the dynamic range of a mobile communication signal diminishes as the channel bandwidth increased. This is referred as the bandwidth-dynamic range trade-off in the literature.

By this principle, the number of channels simultaneously digitized depends on the channel bandwidth of the current standard of operation and suggests that a very high dynamic range is only required in relatively narrow bands where the channels of interest lie.

This has important consequences in the components design as, following this principle, filters involved in SRC can be simplified, advanced ADCs applying noise-shaping techniques can be used for digitization and channelization filters can now be implemented as multirate filters with relaxed constraints.

The design of a concrete software radio transceiver should also take part of commonalities existing between waveforms. However feasible with FPGA-based technologies, a signal processing platform running at high clock rate and completely reconfigurable, with a digital front-end designed for each standard on a common platform, may not be the better solution. Despite the maximized degree of freedom, the price to pay in power consumption and chip area is often prohibitive.

A smarter approach consists of using commonalities of different standards of operation by developing special algorithms realizing the front-end functionalities, like time-varying multiplier-free filters or defining reconfigurable logic blocks as candidates for the hardware platform.

III. Implementation issues

After defining the main lines of the practical software radio architecture, this section will detail the implementation issues in two key functionalities, sample rate conversion and channelization, that make the design of such a device challenging.

1) *Issues in sample rate conversion*

The target of the sample rate conversion process is a signal at a new sample rate with the essential information it carries preserved. Besides, severe restrictions regarding bandwidth and the oversampling ratio of the signal to be converted exist.

So, SRC can be defined by the reconstruction of the original signal $x(t)$ from a signal sampled with period T_1, $x(kT_1)$, followed by a resampling process with a new period T_2 and characterized by the factor relating the period of the input signal $x(kT_1)$ and the new (output) signal $y(mT_2)$.

The traditional chain of sample rate conversion is composed by:

- A digital-to-analog converter, recovering the original signal from the sampled input signal;
- A lowpass filter, also called reconstruction filter, that cancels all signal components resulting from spectral repetition (image components), due to the previous operation;
- A sampler with new rate T_2, that gives the output signal.

Ideally the output signal is identical to the original signal, which mean that no aliasing appeared. Thus, anti-aliasing is the most prominent constraint to be obeyed by any sample rate conversion system.

This assumption is generally true if T_1 is small enough (several times smaller than the Nyquist sampling period) and if the lowpass filter is ideal, which is practically unachievable. However, in most cases, a certain amount of aliasing is tolerable as, recalling the bandwidth-dynamic range trade-off, aliasing components have to be highly attenuated only in a relatively narrow frequency band. And all aliasing components not falling in this frequency band are most of the time removed by filtering after the SRC process.

Besides, the lowpass filter can be combined with the matched filter, which is necessary in most receivers and design constraints on the reconstruction filter can be relaxed.

There are various possible realizations of the SRC operation in software radio terminals:

- A simple technique is oversampling and band-limiting the signal, but this requires a high sample rate and anti-imaging filters to attenuate the image components caused by oversampling;
- Another idea is to approximate arbitrary impulse responses by piecewise polynomial interpolations, however, this requires sophisticated controlling;
- The utilization of cascaded integrator Comb (CIC) filters to attenuate potential aliasing is possible, even if it implies a high intermediate sample rate at which the filters have to operate.

2) Issues in channelization

Channelization is defined as the functionality comprising all necessary tasks to extract a single user channel for further processing at the baseband. Those processes involve downconversion, filtering and despreading, for spread-spectrum systems. The implementation of those steps are discussed below.

As downconversion is realized digitally in software radio transceivers, samples of the complex phasor can be stored in memory and online generation of the samples is feasible. The simplest case is when the center frequency of the digitized channel equals a quarter of the sample rate. Then, the sine and cosine signals that represent the rotating complex phasor simplifies to the sequences *{0, 1, 0, -1}* and *{1, 0, -1, 0}* respectively.

Channel filtering is then required to extract frequency-divided channels. The employed filters have to attenuate adjacent channels that interfere and meet blocking characteristics of the current standard of operation. These requirements are very similar to those imposed to ADCs during the SRC phase.

Due to their fixed structure, direct finite response or polyphase filters are most of the time are not used for channel filtering. Depending on whether the reception is single-channel or multiple-channel based, different solutions are used instead.

In the case of single-channel reception, it can be show that, if the channel is somewhat oversampled, cascaded multirate filters, as the CIC filters mentioned in the previous part, can be used for channel filtering. This is really interesting as, under these conditions, the same physical entity can perform both SRC and channelization.

So as to achieve multiple-channel reception, several single-channel units can be used in parallel. But a smarter approach is to combine downconversion and channelization in filter banks. If only signals of the same standard are received at a time, discrete Fourier transform filters can be used (as all channels have the same bandwidth).

The last operation that can be performed during the channelization process is depreading and only applies to spread-spectrum systems, such as wideband code-division multiplexing. Due to multipath propagation being present in mobile communication systems, a solution called rake receiver is used most of the time. Such a receiver consists of several identical and parallel rake fingers which comprise a correlator and a decimator.

IV. Recent advances in sample rate conversion technology

The previous sections have described the general architecture of an achievable software radio transceiver and emphasized on the key role of the sample rate conversion (SRC) process. This part will describe four state of the art techniques that allow to achieve more efficient conversions demanding less computational requirements and therefore being less energy demanding.

1) An enhanced multi-stage architecture

In their paper[7], Wang and Li propose a multistage sample rate converter. The authors reckon that, in the front-end of a software defined radio receiver, the decimator (used in integer factor SRC) is the key point of interest as the over sampling ratio (OSR) is generally high. If this integer-factor decimator is implemented right after the analog-to-digital converter, a cascaded integrate Comb filter is good practical solution, as it only requires a small chip area and can be implemented using an ASIC or a FPGA.

Therefore the idea behind their work is to use a CIC filter for decimation with a large factor, followed by a multistage decimator with a finer factor and a fractional-factor SRC operating in lower sample rate. The cooperation between those components, in the architecture depicted in the figure below, allowing to efficiently achieve the desired sample rate, according to the current standard of operation.

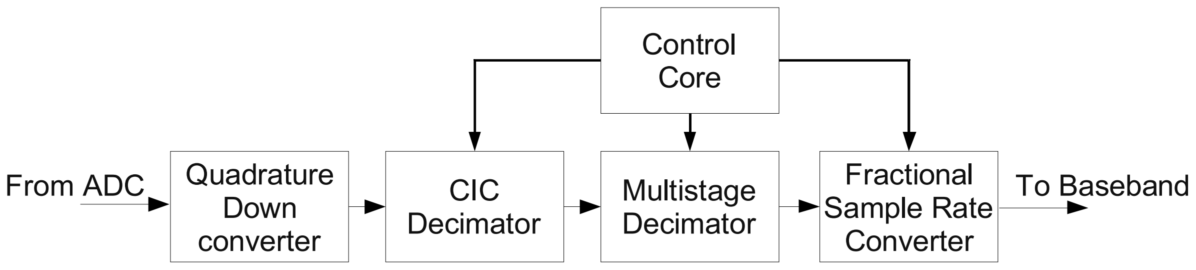


Illustration 6: Proposed multistage sample rate converter architecture

a) Architecture description

The quadrature downconverter, first component of the proposed multi-stage architecture, introduces band pass sub-sampling so as to relax the requirements on the ADC component. As stated previously, this stage can most of the time be expressed as predetermined sequences of phases.

The CIC filter is positioned right after the quadrature downconverter, where the oversampling ration is relatively high. Thus, as the OSR value directly determines the relative bandwidth of the aliasing components that the SRC filter need to attenuate, requirements on the CIC decimator filter are greatly relaxed.

The multistage decimator is the interesting point of the work presented by the authors. As this component is placed after the CIC filter, it has a decreased OSR and therefore must comply with much stricter specifications.

The proposed architecture contains a multistage decimator, consisting of three different parts: a half band filter (HBF), which acts as an anti-aliasing filter, a factor-2 downsampler, and multiplexer which will enable or disable certain stages. As requirements on the HBF filter become more and more stricter as the stage are getting closer to the baseband, the author are using a Nyquist filter for the first two stages (as less implementation effort is the priority) and equiripple low pass filter in later stages (as efficient stopband attenuation is then required).

The Nyquist filter used is realized in the L-band poly-phase form and has the following general form: $H(z)=E_0(z^L)+z^{-1}E_1(z^L)+...+z^{-(L-1)}E_{L-1}(z^L)$. So as to create a half-band filter, L is set to 2 and k to 0, and the implemented filter has the form $H(z)=\alpha+z^{-1}E_1(z^2)$, which is the typical transfer function of a half band filter. The passband and stopband of such a filter are both located at midway in the baseband and such a filter can therefore be used to change the sampling rate by a factor of two.

This filter is also implementation efficient, as half its coefficients are zeroes, which reduces the number of multiplication required. Besides, the author rely on the work from Goodman and Carey that have compiled a list of coefficients for half-band filters with small amplitude distortion. Those results are showed in the table below. In their work, the author chose the *F8* set so as to implement their filter.

Name	*Length*	*Ripple*	*f(0)*	*f(1)*	*f(3)*	*f(5)*	*f(7)*	*f(9)*
F6	11	49dB	346	208	-44	9		
F7	11	77dB	521	302	-53	7		
F8	15	65dB	802	490	-116	33	-6	
F9	19	78dB	8292	5042	-1277	429	-116	18

Table 1: Coefficients for half-band filters with small amplitude distortion

In the last stages of the multi-stage decimator, a linear phase Equiripple FIR filter is adopted, in order to achieve more steep transition band and a larger attenuation in stopband. The filter is designed as a Type II FIR Equiripple filter, which has an interesting symmetry property, allowing to reduce the number of required multipliers, and coefficients are decided using the Parks-McCellan algorithm. The obtained structure is detailed in the figure below.

Finally, the fractional SRC filter is realized with a Farrow Filter structure, which provides an efficient realization of polynomial poly-phase filter.

b) Achieved performances

The described architecture has been tested by the authors with a factor decimation up to 128. Simulation results show that even with such a high factor, the shape of input sequences is preserved even if high frequency noise is filtered out.

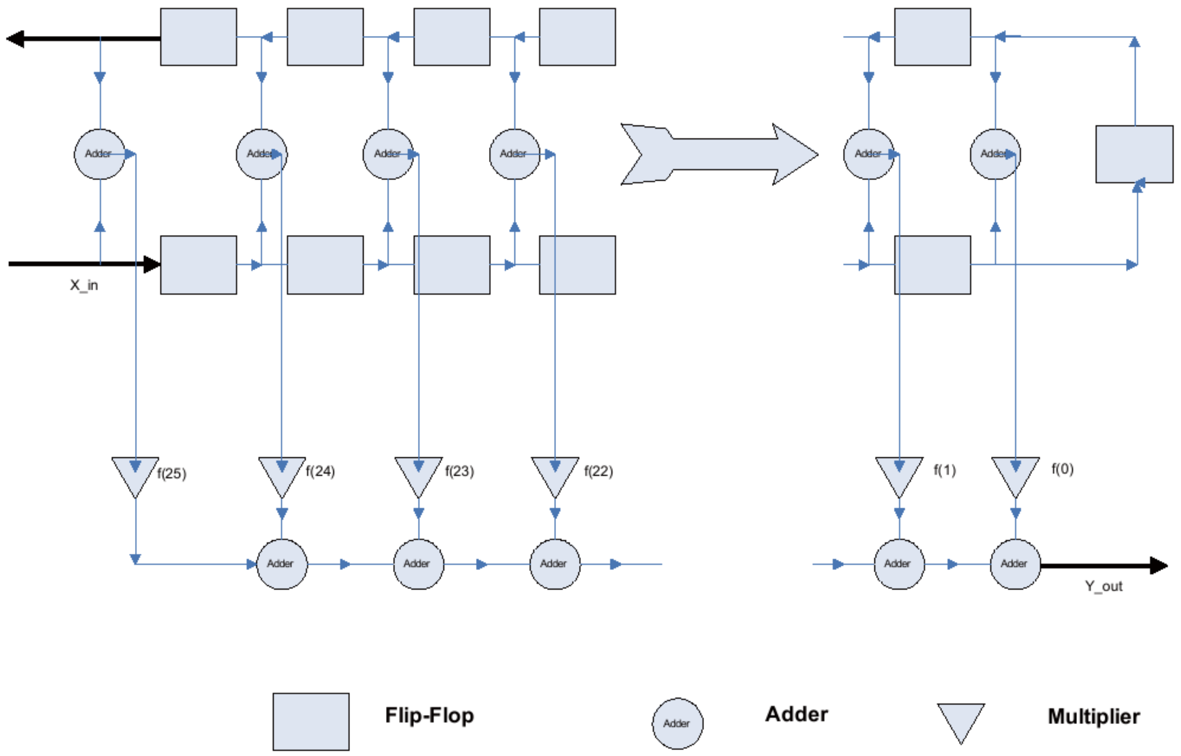


Illustration 7: Structure of the Equiripple filter

2) *Efficient implementation of linear-phase FIR filters*

In [8], the authors, Bregovic, Saramäki, Yu and Lim, introduce an efficient implementation for linear-phase FIR filters in the case where the sampling rate conversion factor can be expressed as a rational number *M/L*, where *L* is the up-sampling and *M* the down-sampling factor. In such an architecture, the sampled input signal in first up-sampled by a factor *L* (using an interpolator block), filtered (so as to suppress image spectra and prevent aliasing effects of down-sampling), then down-sampled by a factor *M* (using a decimator block). This is really similar to the architecture introduced in the paragraph presenting SRC. The position of the filter (between up and down-sampler) make possible to exploit the symmetry of its coefficients so as to achieve and efficient implementation.

a) Architecture description

So as to present the architecture, let's consider the example of a sample rate converter with *L=2, M=3* and a FIR filter of order *N=11*. Under those conditions, the relation of the output samples *y[n]* and the input samples *x[n]*, taking into account the symmetry of the coefficients hi of the filter, can be expressed as :

$$\begin{bmatrix} y[n] \\ y[n+1] \end{bmatrix} = \begin{bmatrix} 0\, h_0\, h_2\, h_4\, h_5\, h_3\, h_1 \\ h_1\, h_3\, h_5\, h_4\, h_2\, h_0\, 0 \end{bmatrix} \boldsymbol{x}_{m+1,m-5} \quad \text{where} \quad m = \frac{M}{L} n \quad \text{and} \quad \boldsymbol{x}_{k,l} = [x[k], x[k+1], \ldots, x[l]]^T$$

Some terms are very simple to express and can be directly implemented. For the other terms, the previous equation can be manipulated so as to achieve the following expression :

$$\begin{bmatrix} h_0 & h_2 & h_5 & h_3 \\ h_3 & h_5 & h_2 & h_0 \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_{m,m-1} \\ \boldsymbol{x}_{m-3,m-4} \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} c_0 & c_1 & 0 & 0 \\ 0 & 0 & d_1 & d_0 \end{bmatrix} \boldsymbol{x}^{(1)}_{m,m-4} \quad \text{where}$$

$$c_0=(h_0+h_3)/2 \quad , \quad c_1=(h_2+h_5)/2 \quad , \quad d_0=(h_0-h_3)/2 \quad , \quad d_1=(h_2-h_5)/2 \quad \text{and}$$

$$\boldsymbol{x}^{(1)}_{m,m-4} = \begin{bmatrix} I_2 & J_2 \\ J_2 & -I_2 \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_{m,m-1} \\ \boldsymbol{x}_{m-3,m-4} \end{bmatrix}$$

, where I_i and J_i are the identity and counter identity matrices.

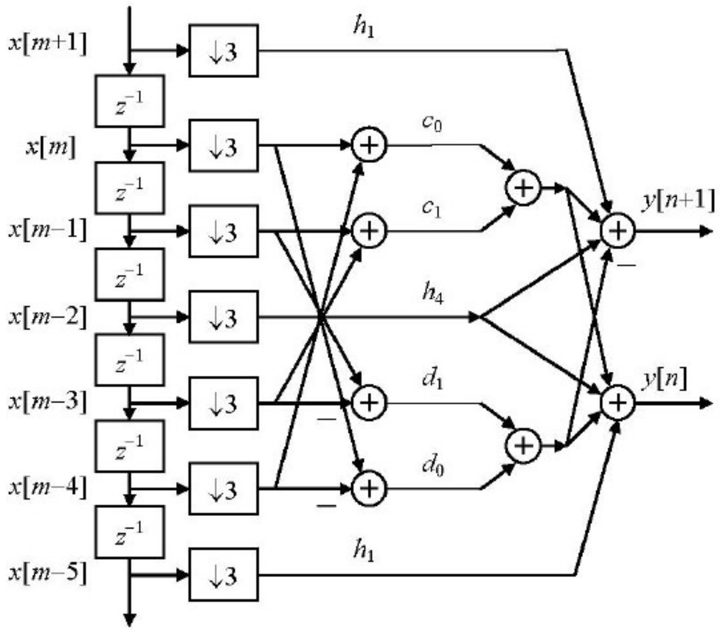


Illustration 8: Implementation structure for rational sampling 3/2

This expression leads to the implementation depicted in the figure above. As can be seen, the number of required multiplications is not exactly halved (as can be expected while using the symmetry property of the coefficients). This is due to the h_1 value being repeated, which could be avoided by adding additional delay to the structure. This number being proportional to the order of the filter, this can be high memory consuming, depending on the filter used. Besides, the central term (h_4 here) is only present if the order of the filter has the form *4k-1*, $(k \in \mathbb{N} \setminus \{0,1\})$.

The presented model can be extended for any filter of the two following forms : $N=M(L-1)+2kL$ or $N=M(L-1)+(2k+1)L$, with $k \in \mathbb{N}$

The general expression of the input *y[n]*, given the input *x[n]* is:

$$\begin{bmatrix} y[n] \\ y[n+1] \\ \vdots \\ y[n+L-1] \end{bmatrix} = \boldsymbol{H}_{L(p+q+1)} \boldsymbol{x}_{m+p,m-q} \quad \text{where} \quad \begin{matrix} q=\text{floor}(N/L) \\ p=\text{floor}((L-1)M/L) \end{matrix} \quad \text{and}$$

$\boldsymbol{H}_{L(p+q+1)}$ is an L by $p+q+1$ matrix containing the coefficients of the filter used.

Again, manipulating this expression leads to the following equation, that can be used to implement the efficient architecture:

$$\boldsymbol{x}_{m,m-4}^{(s-1)} = \begin{bmatrix} I_s & J_s \\ J_s & -I_s \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_{m,m-s+1} \\ \boldsymbol{x}_{m-q+p-s+1,m-q+p} \end{bmatrix} \quad \text{where} \quad \boldsymbol{x}_{k,l}^{(r)} = \begin{bmatrix} x[k]+x[l] \\ x[k-1]+x[l+1] \\ \cdots \\ x[k-r]+x[l+r] \\ x[k-r]-x[l+r] \\ \cdots \\ x[k-1]-x[l+1] \\ x[k]-x[l] \end{bmatrix}$$

b) Performance achievements

The implementation complexity can be measured in terms of number of multiplication (C^*) and additions (C^+) required so as to perform the sample rate conversion process. The comparison with the requirements of a direct implementation for various values of the filter order N is given in the table below.

	Dirrect		Proposed		Comparison	
N	C_d^*	C_d^+	C_p^*	C_p^+	C_p^*/C_d^*	C_p^+/C_d^+
25	8.7	8.3	5.3	7.3	0.615	0.880
37	12.7	12.3	7.3	10.0	0.579	0.865
49	16.7	16.3	9.3	14.0	0.560	0.857
61	20.7	20.3	11.3	17.3	0.548	0.852
73	24.7	24.3	13.3	20.7	0.541	0.849
109	36.7	36.3	19.3	30.7	0.527	0.844
145	48.7	48.3	25.3	40.7	0.521	0.841
205	68.7	68.3	35.3	57.3	0.515	0.839

Table 2: Comparison of computational efficiency for various values of N

3) *Efficient use of trapezoidal interpolation*

In [9], the authors present an interesting approach using trapezoidal interpolation for sample rate conversion in software designed radio transceivers. In this approach, the reconstruction filter is sampled at an intermediate rate and interpolated so as to approximate the desired filter impulse response. The interpolation function, based on trapezoidal interpolation, takes into consideration anti-imaging and anti-aliasing requirements and uses a very simple algorithm for implementation.

As stated in the description of the SRC process and its challenges, the received signal *x(t)* is sampled at period T_1 by the A/D converter, then passed through the reconstruction filter *h(t)*, so as to obtain the output *y(t)*, which is then sampled with period T_2. Therefore, we have the following equation, representing a time-varying digital filtering:

$$y(mT_2) = \sum_{k=-\infty}^{\infty} x(kT_1)\, h(mT_2 - kT_1)$$

To simplify this process to a time-invariant operation, *h(t)* is approximated by a digital filter $\hat{h}(nT_3)$ with intermediate sampling period T_3 and constructed using an interpolation function to reconstruct *h(t)*. Therefore, we can write, in both time and frequency domain:

$$h(t) = \sum_{n=-\infty}^{\infty} \hat{h}(nT_3)\, r(t - nT_3) \quad \text{and} \quad Y(f) = X(e^{j2\pi f T_1})\, H(f) = X(e^{j2\pi f T_1})\, R(f)\, \hat{H}(e^{j2\pi f T_3})$$

The authors have chosen to design the interpolation function as two *sinc* components, the first rejecting images of the signal, the second rejecting images of the approximated digital filter. This yields to the following design:

$$R(f) = T_1 \frac{\sin(\pi f T_1)}{\pi f T_1} \exp^{-j\pi f T_1} T_3 \frac{\sin(\pi f T_3)}{\pi f T_3} \exp^{-j\pi f T_3}$$

The filter is thus the convolution of two rectangular pulses in the time domain, which is a trapezoidal pulse. So as to implemented the filter, T_3 is chosen as an integer fraction of T_2 and the trapezoidal pulse is decomposed as a ramped step function of duration T_1 minus the same ramped delayed by T_3. The ramped step is further decomposed so as to achieve the practical implementation showed in the figure below.

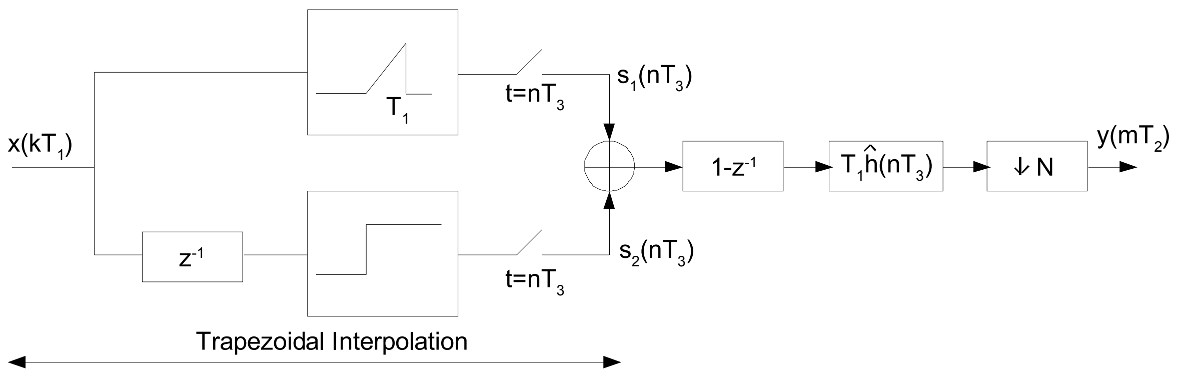


Illustration 9: Implementation of efficient trapezoidal implementation

Since the *x(kT₁)* sequence is spaced at T_1, which is also the width of the ramp, there is no overlapping after the sample has passed through the unit ramp filter. Therefore, $s_1(nT_3)$ and $s_2(nT_3)$ can be computed, with only one multiplication, as follows:

$$s_1(nT_3) = x\left(\left[\frac{nT_3}{T_1}\right] T_1\right)\left(\frac{nT_3}{T_1} - \left[\frac{nT_3}{T_1}\right]\right)$$

$$s_2(nT_3)=\left\{\begin{array}{c} s_2((n-1)T_3)+\sum_{k=\left[(n-1)\frac{T_3}{T_1}\right]}^{\left[\frac{nT_3}{T_1}\right]-1} x((k-1)T_1),\ \mathit{if}\left[\frac{nT_3}{T_1}\right]-1\geqslant\left[(n-1)\frac{T_3}{T_1}\right] \\ s((n-1)\,T_3)\,,\mathit{otherwise} \end{array}\right\}$$

$$\text{with } \left[\frac{nT_3}{T_1}\right] \text{ is the maximum integer less than } \frac{nT_3}{T_1}$$

The digital filter itself is then designed so as to have similar spectrum response as a root raised cosines, which is typically used as match-filter for receivers. The trapezoidal interpolation inducing a delay of *$(T_1+T_3)/2$*, the time response of the filter should be advanced of the same amount so as to have a overall zero phase filter impulse response.

A zero phase filter with correct spectral behavior is first designed using frequency sampling, yielding to the following definition in time-domain:

$$\hat{h}_0(t)=\sum_{k=-\infty}^{\infty}\left|\frac{H(k\Delta f)}{R(k\Delta f)}\right|e^{j2\pi k\Delta f t}$$ where *H(f)* is the frequency response of the root raised cosine filter.

Finally, the desired linear phase FIR filter is obtained by delaying this function and sampling it with a period of *T_3*. The resulting filter, defined by the equation below, can be efficiently implemented using the linear phase property leading to a symmetry of the impulse response.

$$\hat{h}(nT_3)=\hat{h}_0(nT_3-\frac{1}{2\Delta f}+\frac{1}{2}T_3)\quad,\quad n=0,1,\ldots,\frac{1}{\Delta f\,T_3}-1$$

4) *Hierarchical architecture for sample rate conversion*

In [10], the authors state that up-sampling in software defined radio transmitters requires as much computational power as down-sampling in SDR receivers. Therefore, their rationale is that increasing the computational efficiency of up-sampling is as important as increasing the efficiency of down-sampling in receivers.

They propose a computationally efficient method for the sample rate conversion process with rational factors greater than unity, particularly efficient when input signals have low over-sampling factors and high dynamic ranges. The proposed method uses a hierarchical architecture so as to compute the output samples, allowing to reuse previously computed samples, and relies on a recursive combination of FIR filters and polynomial interpolation.

a) Architecture description

As the sampling ratio *M/L* is greater than unity, the continuous-time waveform *x(t)* can be approximated from the samples *$x(kT_1)$* using a continuous-time low-pass filter with the following impulse response, using a symmetric window *w(t)*:

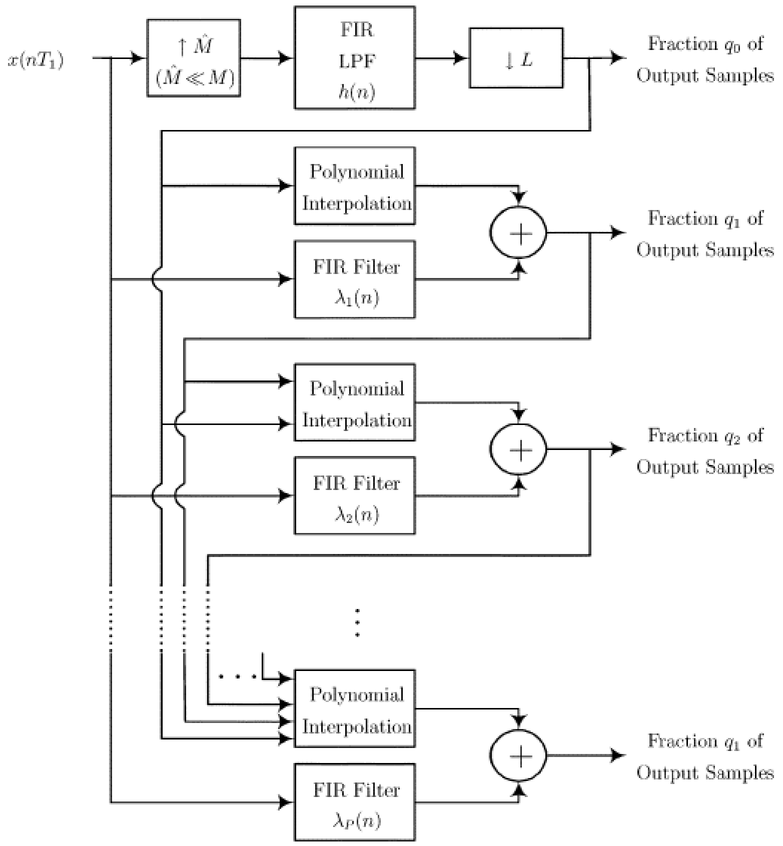
$$h(t) = w\left(\frac{t}{AT_1}\right) sinc\left(\frac{t}{T_1}\right) \quad \text{with} \quad w(0)=1 \quad \text{and} \quad w(t/AT_1)=0, |t| > AT_1$$

This yields to the following approximation of the waveform, which can be resampled with period T_2:

$$\hat{x}(t) = \sum_{k=\left[\frac{t}{T_1}\right]-A+1}^{\left[\frac{t}{T_1}\right]+A} x(kT_1)\, w\left(\frac{t-kT_1}{AT_1}\right) sinc\left(\frac{t-kT_1}{T_1}\right)$$, where $[x]$ is the larger integer less than x

$$\hat{x}(mT_2) = \sum_{k=\left[m\frac{T_2}{T_1}\right]-A+1}^{\left[m\frac{T_2}{T_1}\right]+A} x(kT_1)\, w\left(\frac{mT_2-kT_1}{AT_1}\right) sinc\left(\frac{mT_2-kT_1}{T_1}\right) \quad \text{with} \quad T_2 = \frac{MT_1}{L}$$

This expression shows that each output sample only depends upon a finite number of input samples. Besides, it can be shown that an input sample influences the magnitude of an output sample in proportion to the inverse time difference between the two sample. This is the rationale behind the efficient architecture described in the figure below.



$x(nT_1)$
$\uparrow \hat{M}$ $(\hat{M} \ll M)$
FIR LPF $h(n)$
$\downarrow L$
Fraction q_0 of Output Samples
Polynomial Interpolation
FIR Filter $\lambda_1(n)$
Fraction q_1 of Output Samples
Polynomial Interpolation
FIR Filter $\lambda_2(n)$
Fraction q_2 of Output Samples
Polynomial Interpolation
FIR Filter $\lambda_P(n)$
Fraction q_1 of Output Samples

Illustration 10: Hierarchical computation of output samples

b) Performance achievements

The computational performance of the proposed architecture is compared with traditional single-stage and two-stage implementations for sample rate conversion. Those requirements are determined by several factors like the choice of the conversion factors (*L* and/or *M*), the bandwidth, the oversampling ratio and the sample rate of the input signal.

When a FIR low-pass filter is used to attenuate the undesired SRC images, the order of this filter is proportional to the value of *L* and inversely proportional to the transition bandwidth. It can be therefore shown that for a single-stage SRC process using a filter with an order of *2CL*, the the computational requirement is *MOPS = 2C*, where *C* depends on the oversampling ratio and the maximum passband and stopband ripples.

For a two-stage SRC process, with a conversion factor *L/M = (L1/M1)(L2/M2)* using two filters of orders *2DL1* and *2EL2*, the computational requirement is *2DM2/L2 + 2E*, where *D* and *E* depend on he oversampling ratio and the maximum passband and stopband ripples.

For the proposed architecture, the computational requirement can be expressed as:

$MPOS = 2q_0 A + 1 + \sum_{j=1}^{N} q_j * P_i$, where q_i is the fraction of output samples computed at step *i*, *N* is the total number of steps, P_i is the number of multiplications required at step *i* and *A* is used to set the span of the sampling window.

V. Conclusion

In this document, the main purposes software designed radio and challenges in its development have been exposed. The key part of the sample rate conversion process has also been emphasized.

This step consists in adapting the sample rate of the digital received signal, different across the standards that the SDR is able to receive and process to a unified sample rate used in the next process stages in the base-band.

Due to the large bandwidth to process and the dynamic behavior of the received signals, the design of the sample rate conversion block is really challenging, as it should particularly avoid aliasing on the resampled signal and be computationally efficient so as to limit the number of required components and the energy consumption associated with the process.

However, recent work, depicted in this document, show interesting architectures ans implementations so as to achieve SRC efficiently. Those techniques rely either on an efficient implementation of the filters themselves, or on a clever architecture reducing the required number of operations (additions and multiplications) required to compute the output samples, or on both of those.

As such processes seemed hardly achievable when the software designed radio concept was first introduces more than a decade ago, such techniques make concrete SDR transceiver efficiently realizable.

References

[1] J. Mitola, *The Software Radio Architecture*. IEEE Communications Magazine, 1995.

[2] E. Buracchini, *The Software Radio Concept*. IEEE Communications Magazine, 2000.

[3] T. Hentschel, M. Henker and G. Fettweis, *The Digital Front-End of Software Radio Terminals*. IEEE Personal Communications, 1999.

[4] Mahesh, R.; Vinod, A.P., *Reconfigurable Low Complexity Fir Filters for Software Radio Receivers*. IEEE 17th International Symposium on Personal, Indoor and Mobile Radio Communications, Sept. 2006.

[5] Hassani, M.; Karami-Mollaei, M.R.; Mohammad-Taheri, M., *Computationally efficient algorithm for reducing the complexity of software radio receiver's filter bank*. Canadian Conference on Electrical and Computer Engineering, May 2005.

[6] T. Hentshel, G. Fettweis, *Sample Rate Conversion for Software Radio*. IEEE Communications Magazine, 2000.

[7] T. Wang, C. Li, *Sample Rate Conversion Technology in Software Defined Radio*. IEEE CCECE/CCGEL, May 2006.

[8] R. Bregovic, T. Saramäki, Y.J. Yu, Y.C. Lim, *An Efficient Implementation of Linear-Phase FIR Filters for a Rational Sampling Rate Conversion*, ISCAS 2006.

[9] X. Huang, Y. Li, S. Nguyen, *Sample Rate Conversion by Trapezoidal Interpolation for Software Defined Radio*. Proceedings of the 14th IEEE International Symposium on Personal, Indoor and Mobile Radio Communication, 2003.

[10]W.A. Abu-Al-Saud, G.L. Stüber, *Efficient Sample Rate Conversion for Software Radio Systems*. IEEE Transactions on Signal Processing, Vol. 54, No. 3, March 2006.