Sample Rate Conversion for Software Radio

Tim Hentschel and Gerhard Fettweis, Dresden University of Technology

ABSTRACT

Software radio terminals must be able to process many various communications standards. These standards are generally based on different master clock rates and thus employ different bit/chip rates. The most obvious solution to cope with the diversity of master clock rates in one terminal is to provide a dedicated master clock for each standard of operation. Not only too costly, this kind of solution limits the applicability of a realized terminal. Hence, it is much more elegant to run the terminal on a fixed clock rate, and perform digital sample rate conversion controlled by software.

INTRODUCTION

Software radio is a notion as ambiguous as any object of research can be. Depending on the point of view, different facets can be emphasized (e.g., the network or terminal aspect), where either the hardware or software aspects may be of more interest. From a signal processing point of view, software radio is a means to realize as many as possible signal processing tasks of a communications transceiver by means of software. Hence, there must be a hardware platform on which this software can run. As technology advances, this hardware platform will more and more appear as a kind of compute engine driven by a digital signal processor (DSP) and equipped with certain accelerators. Together with the necessary interfaces — the analog-to digital converter (ADC) and digital-to-analog converter (DAC) — this compute engine is clocked by a master crystal. The interfaces especially require high-quality clocks with very low jitter (Fig. 1).

Since different communications standards are based on different master clock rates, it is mainly necessary to provide these different clock rates. However, due to the strong requirements for clock quality, it is reasonable to assume that only one fixed master clock will be provided in practical software radio applications. A solution to this is to provide the different clock rates virtually by means of digital sample rate conversion (SRC). Hence, with the advent of software radio a new functionality has to be introduced to the signal processing of digital communications transceivers: SRC. Apart from being just a lukewarm rehash of the well-known interpolation problem, SRC is the task of converting the sample rate of a digital signal to another sample rate while a certain amount of information, usually in a limited frequency band, must be preserved. As will be seen in the course of the article, the main concern is not interpolation but anti-aliasing [1].

SRC has to take place at each of the analog-to-digital or digital-to-analog interfaces. Depending on the characteristics of a signal and the amount of information to be preserved, different solutions for SRC exist. Among those, interpolation as suggested in [2] is a solution for a certain application scenario only. The reason interpolation is often favored is a “historical” one. Interpolation is a well-known solution to the mathematical problem of calculating in-between values of tabulated functions. In the context of signal processing the table elements are replaced by signal samples. However, when comparing the requirements for calculating in-between values of tabulated functions and calculating those of discrete-time signals, it becomes apparent that the classical approach of employing interpolation for SRC is not sufficient. The idea of interpolation is to have a smooth curve whose samples are the tabulated values. Thus, interpolation is a time-domain approximation. In signal processing the constraints on calculating in-between values can be different. A smooth course of the signal is generally not the issue; the main concern is the information carried by the signal or parts of it. To understand this it is helpful to shift the point of view from the time to the frequency domain, where usually different frequency bands carry different information. By doing this the requirement for SRC can be formulated as *calculating in-between values of a discrete-time signal such that a certain frequency band of the signal is not distorted*.

Once the signal characteristics (and thus the distribution of the interesting information in the frequency domain) are known, the problem of SRC can be tackled with the idea of *resampling after reconstruction*, where an analog signal is (virtually) reconstructed from the digital signal by means of DAC and filtering. Eventually the reconstructed signal is resampled [3]. From this idea the basic equations can be derived, resulting in an all-digital description of the necessary filtering task. Although this all-digital description enables one to forget the

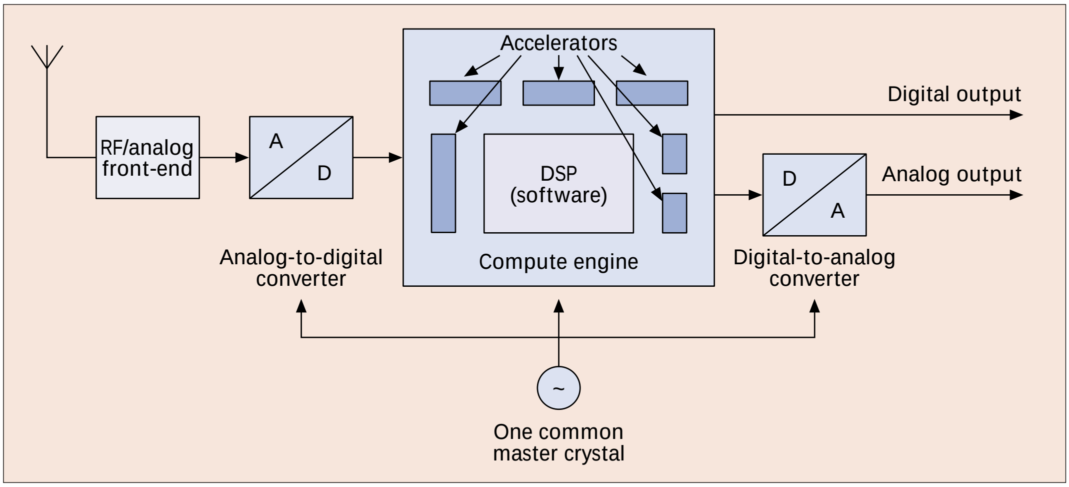


Figure 1. *The hardware platform of a software radio receiver.*

detour via reconstructing an analog signal and resampling it, it is of paramount importance to always keep in mind that *SRC is a process of resampling*. The fundamental effects of sampling — imaging and aliasing — must be expected to appear with SRC. Hence, SRC turns out to be mainly a problem of designing appropriate (anti-imaging and anti-aliasing) filters and hardware structures implementing these filters. In particular, the underlying hardware structures should not be underestimated. Since power consumption is a major issue in mobile communications systems — at least on the mobile terminal side — hardware structures are sought which implement the necessary filtering tasks efficiently. Moreover, these hardware structures should be adaptive in such a way that SRC can be performed independent of the current standard of operation of the software radio terminal. Different rate change factors, pass-bandwidth, and stop-band attenuation are the main characteristics which must be variable, and call for reconfigurability of the hardware platform for SRC in software radio transceivers. In order to show how demanding this reconfigurable hardware platform is, the article starts with a brief discussion of signal processing issues of software radio.

A Signal Processing Perspective on Software Radio

Approaching the software radio concept from a signal processing perspective means that one tries to implement the different functionalities of a transceiver by means of software. Since the signals at the antenna are analog signals, there will always be the need for a radio frequency (RF) part and an analog front-end. Depending on how many functionalities of the transceiver can be realized in the digital domain, the complexity of the RF part and analog front-end varies. It should be noted that realizing a certain functionality in the digital domain does not necessarily mean that it is fully reflected by software running on a DSP. There are certain DSP algorithms which are too demanding for a (conventional) DSP to achieve a cost-effective solution. A typical example is digital down-conversion (DDC) of a received signal, which is performed directly after ADC, and thus at a relatively high sample rate. DDC is multiplication of a bandpass signal with a rotating complex phasor. It can be realized by two separate multiplications with sine waves having a mutual phase difference of /2. One just has to imagine a sample rate of 65 Msamples/s to quickly realize that a conventional DSP with one multiply-accumulate (MAC) unit being clocked at 65 MHz is not capable of handling this task.

Approaching the software radio concept from a signal processing perspective means that one tries to implement the different functionalities of a transceiver by means of software. Since the signals at the antenna are analog signals, there will always be the need for an RF part and an analog front-end.

Therefore, such “number-crunching” functions are realized by dedicated pieces of hardware named *accelerators*. These accelerators are not programmable. Still, in the context of software radio they should be adaptable to the different modes of operation of the transceiver. This can be reached by making them parameterizable. Due to the very limited functionality of a certain accelerator, it is sufficient to just change its parameters in order to adapt it to a certain task; for example, it is sufficient to change the carrier frequency and phase offset of a DDC.

The different accelerators and DSP form the compute engine of a software radio transceiver. In Fig. 1 is sketched how a compute engine, the ADC, the DAC, and the RF part form the hardware platform of a software radio receiver. By reversing the order of the signal flow, the principal structure of a hardware platform of a software radio transmitter can be obtained.

There are a manageable number of different signal processing tasks which might require acceleration: Viterbi equalization, despreading/spreading, digital filtering (at high sample rates), digital down/upconversion, and so on. At any time some of them are part of every communications transceiver. Hence, it is sensible to merge them with the DSP, yielding a new generation of application-tailored DSPs; for further details on this concept see [4, 5].

As will be seen later in this article, SRC should be implemented on a cascaded multirate architecture where some parts have to run at very high clock rates. Therefore, the necessary filtering tasks overstretch the capabilities of conventional DSPs, as DDC does. Hence, SRC is a candidate to get its own accelerator.

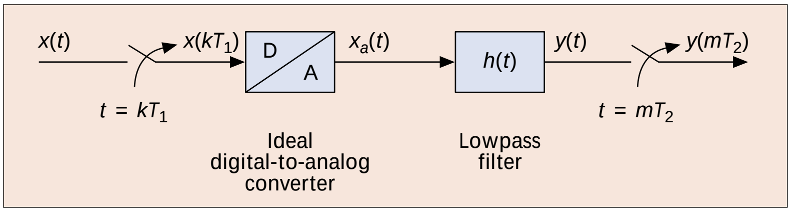


Figure 2. *Resampling after reconstruction.*

Signal Characteristics

Before dealing with SRC something must be said about the characteristics of signals whose sample rate should be converted. Basically, signals with high dynamic range and those with lower dynamic range must be distinguished. In the context of mobile communications the first are mainly a result of sampling a signal covering a bandwidth which comprises several channels of a certain mobile communications standard. Due to the adjacent channel interference characteristics allowing an adjacent channel to have a considerably higher power level than the channel of interest itself, such *multichannel* signals have high dynamic range. Conversely, single-channel signals have lower dynamic range.

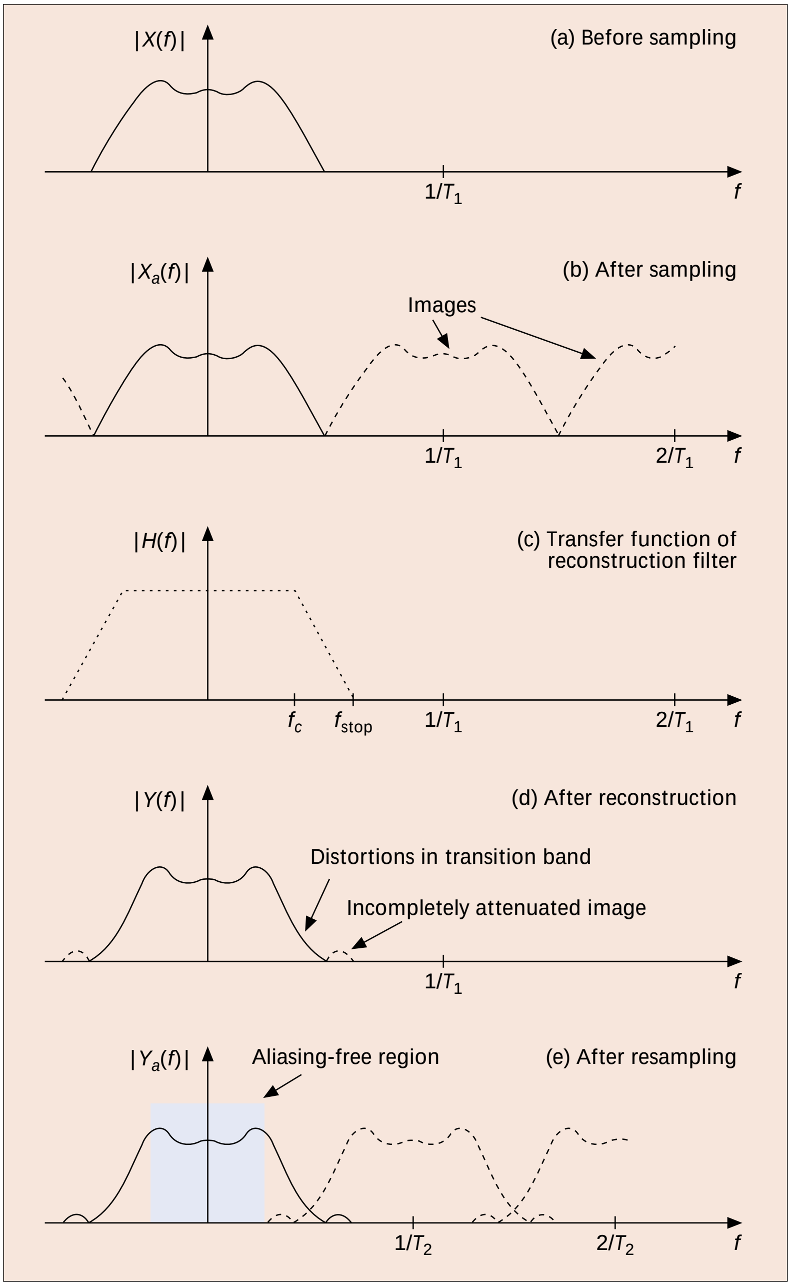


Figure 3. *Spectral interpretation of SRC (given for the case* $T_1 < T_2$*). (*$|Y_a(f)|$ *is the magnitude spectrum of the sampled version of* y(t)*, since* $|X_a(f)|$ *is the magnitude spectrum of the sampled version of* x(t)*.)*

With respect to Fig. 1, the ADC and a possible following SRC have to cope with a multichannel signal, while the DAC and a possible preceding SRC might just have to process a single-channel signal (e.g., voice). In a transmitter the DAC has to cope with a multichannel (transmission) signal, while the ADC just needs to digitize a single-channel signal (e.g., an audio signal).

With respect to the definition of SRC in the introduction to this article, it should be noted that in many cases it is sufficient to preserve the information of one single channel of interest of a multichannel signal when performing SRC on a multichannel signal.

Resampling after Reconstruction

A straightforward solution to SRC is to reconstruct the original analog signal from the discrete-time signal and resample the reconstructed signal with the new clock period T_2. This approach is sketched in Fig. 2.

Assuming ideal DAC (ideal Dirac impulses) and a reconstruction filter with impulse response $h(t)$, the discrete-time signal after SRC can be related to the discrete-time signal before SRC as follows:

$$\begin{aligned} y(mT_2) &= \sum_{k=-\infty}^{\infty} x(kT_1)\cdot h(mT_2 - kT_1) \\ &= \sum_{n=-\infty}^{\infty} x\left(T_1\left(\left\lfloor \frac{mT_2}{T_1} \right\rfloor - n\right)\right)\cdot h(T_1(n+\mu_m)) \end{aligned} \qquad (1)$$

with

$$\mu_m = \frac{mT_2}{T_1} - \left\lfloor \frac{mT_2}{T_1} \right\rfloor$$

and

$$n = \left\lfloor \frac{mT_2}{T_1} \right\rfloor - k.$$

The quantity $_m$ is limited to the interval $_m$ [0,1). It reflects the position of the current sample inside to be calculated inside the sample period T_1, and is commonly called the *intersample position*. Equation 1 is a direct digital representation of SRC. The impulse response $h(t)$ is sampled with the period T_1 of the input signal, still, with different time offsets $_mT_1$ which depend on the position of the output sample with respect to the input samples. This means that for each calculation of an output sample, a different set of samples of the impulse response $h(t)$ is used. Hence, the digital representation of SRC is a time-varying filtering operation. The filter $h(t)$ influences the quality of the SRC process, which itself is determined by the effects *aliasing* and *imaging*. This can be understood from interpreting SRC spectrally, as in Fig. 3.

If the sampling period T_1 is short enough with respect to the bandwidth of the bandlimited signal $x(t)$, the images caused by spectral repetition due to sampling do not overlap and thus do not cause aliasing, as shown in Fig. 3b. Perfect reconstruction would mean that $h(t)$ is an ideal low-pass filter canceling all images. Still, this is neither possible nor necessary in most cases. Depending on the band to be kept free from distortions by SRC, the filter $h(t)$ can employ more or less relaxed constraints, possibly causing passband distortions of the original signal and/or incomplete attenuation in the stop-band (e.g., incomplete attenuation of the images). These effects are sketched in Fig. 3c, d. Since any sampling process causes spectral repetition with respect to the sampling period, resampling also does. According to the characteristics of $h(t)$ the images overlap. Consequently, aliasing distortions arise, as shown in Fig. 3e. Thus, the design of the filter is the first task to be solved when tackling the problem of SRC. The main purpose of this filter is to control aliasing. As stated in [1], *anti-aliasing is the most prominent constraint to be obeyed by any sample rate conversion system*.

The power of the aliasing distortions depends on the signal characteristics. It should be remembered that in the context of SRC the aliasing components are a result of the discrete-time nature of the original signal $x(kT_1)$ and the respective spectral repetitions (Fig. 3). In a single-channel signal the aliasing components cannot be stronger than the channel of interest, while in multichannel signals the aliasing components can have a much higher power level than the channel of interest. This occurs if the channel of interest in a multichannel signal has low power while some adjacent channels potentially causing aliasing to the channel of interest have high power. From this short discussion it can be concluded that it is sensible to stick to the frequency domain approach to SRC rather than the time domain approach.

From Fig. 3 it is evident that the smaller the region which should be kept free from aliasing, the smaller the stop-bands of the filter attenuating the aliasing components can be. This enables the application of comb filters with reduced effort.

Rational Factor SRC

Rational factor SRC is the case if the sample periods of the input and output signals are related by two positive integers L and M (i.e., $T_1/T_2 = L/M$). Now, the second part of Eq. 1 can be rewritten as

$$y(mT_2) = \sum_{n=-\infty}^{\infty} x\left(T_1\left(\left\lfloor \frac{mM}{L} \right\rfloor - n\right)\right)\cdot h(T_1(n+\mu_m)). \qquad (2)$$

The intersample position

$$\mu_m = \frac{(mM)(\mathrm{mod}\, L)}{L}$$

can only take L distinct values. Thus, the number of different sets of samples of $h(t)$ used for the calculation of $y(mT_2)$ is limited to L. Since $_m$ is periodic with period L, the filter $h(\cdot)$ can be implemented as a periodically time-varying filter with period LT_2.

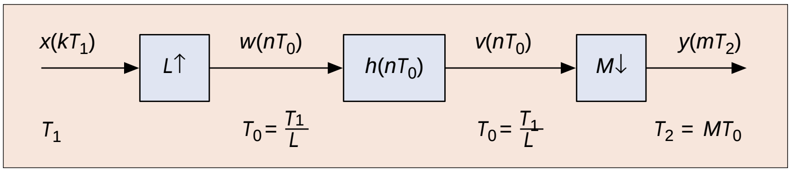


Figure 4. *Rational factor SRC.*

Introducing the period $T_0 = T_1/L = T_2/M$, and the two signals $v(nT_0)$ and $w(nT_0)$ enables us to rewrite Eq. 1 and derive the well-known block diagram of a system for rational factor SRC [3] sketched in Fig. 4. The signal $w(nT_0)$ is the result of upsampling the signal $x(kT_1)$ by L. It is generated by the $L\neq$ block, called an *upsampler* or a *sample-rate expander*. Upsampling by L is realized by inserting $L - 1$ zeros between two consecutive samples of the original signal. In order to obtain the output signal $y(mT_2)$ the signal $v(nT_0)$ is downsampled by M. This is done by the M block, called a *downsampler* or a *sample-rate compressor*. Downsampling by M is realized by deleting all but every Mth sample from the signal.

Although very useful for investigations, the structure of Fig. 4 is not applicable in practice. This is due to the possibly very high intermediate sample rate $1/T_0$ at which the filter would have to be clocked.

Integer Factor SRC

By setting L or M to 1, integer factor SRC as a special case of rational factor SRC can be achieved. All considerations made with respect to rational factor SRC can be applied directly to integer factor SRC.

In integer factor sample rate increase ($L > 1$, $M = 1$), the images caused by the SRC process can be regarded as not overlapping, and thus not causing aliasing. The filter is a pure anti-imaging filter. Often, it is called an *interpolation filter*, which — together with the upsampler — forms an interpolator (i.e., a system for integer factor sample rate increase).

Setting $L = 1$ leads to a reduction of the sample rate by M. Depending on the bandwidth of the signal before SRC the images resulting from resampling might overlap, which means that aliasing occurs. Aliasing can be reduced by proper anti-aliasing filtering with $h(\cdot)$. The combination of the anti-aliasing filter and downsampler is called a *decimator*.

Concepts for SRC

Since the sample rate both before and after SRC can be expressed as an integer number of samples per time unit, the rate change (conversion) factor is a rational number. An exception to this is the case where two asynchronous digital systems are interfaced [3]. Besides the conversion factor, another design parameter is important: the ratio between the sample rate of the signal,

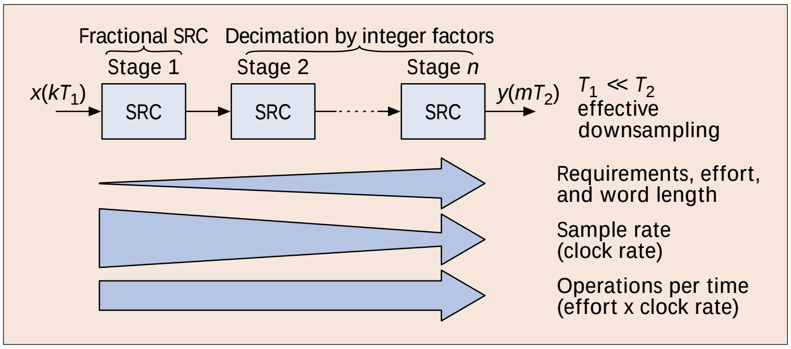


Figure 5. *Effort vs. sample rate in a cascaded SRC system (with the suggestion of placing fractional SRC at a high sample rate).*

and the bandwidth of the channel of interest, that is, the oversampling ratio (OSR) of the channel of interest. The OSR *after* the SRC process directly determines the relative bandwidth (compared to the sample rate) of potential aliasing components that have to be attenuated by the SRC filter. The higher the OSR, the smaller the passband and the stop-bands of this filter can be. Hence, a high OSR (after SRC) relaxes the design constraints, leading to simpler filter structures. A consequence of this is that SRC is advantageously implemented on a cascaded multirate system (i.e., a cascade of filters with relaxed requirements at high sample rates and strong requirements at low sample rates) [3].

Since the number of coefficients of a filter complying with a given tolerance scheme is heavily dependent on the type of filter, it is not possible to give quantitative estimates that are generally valid. Therefore, it should be sufficient to note that decreasing the sample rate while keeping the bandwidth of the channel of interest constant means a decreasing OSR. Again, this leads to an increase in the necessary number of coefficients of the respective filter stage. Hence, in multirate systems a trade-off between sample rate and hardware effort (number of coefficients) can be made.

The ratio of the output and input sample rates is $f_2/f_1 = L/M$, where L and M are relatively prime (i.e., the greatest common divisor of L and M is 1). It can be distinguished between effective reduction ($L < M$) and effective increase ($L > M$) of the sample rate.

Moreover, it can be distinguished between integer factor and fractional SRC. Although having several commonalities, integer factor and fractional SRC also have differences. Therefore it is sensible to separate them. The rate change factor can be factorized to a fractional part $(L/M)_{\text{frac}}$ and an integer part L_{int} (or M_{int}):

$$\frac{L}{M} = \begin{cases} \left(\frac{L}{M}\right)_{\text{frac}} \cdot L_{\text{int}} & \text{effective increase} \\ \left(\frac{L}{M}\right)_{\text{frac}} \cdot \frac{1}{M_{\text{int}}} & \text{effective reduction} \end{cases}.$$

It should be noted that the fractional factor $(L/M)_{\text{frac}}$ is limited to the interval (0.5,2). If the overall conversion is an effective reduction of the sample rate, this factor can be limited further to the interval (0.5,1) or (1,2) in the case of an effective increase of the sample rate. Whether this further limitation is sensible or not depends on the filter algorithms and their implementation (e.g., it might be a problem to temporarily increase the sample rate in a system realizing an overall sample rate reduction).

The order of arranging the fractional and integer parts is an open question. It should be noted that the OSR before fractional SRC is of the same order as after fractional SRC. Therefore, the requirements for integer factor SRC remain the same regardless of its position within the sequence of filtering. Thus, the above mentioned trade-off between sample rate and hardware effort can be made for the fractional part of the SRC system independently.

The sample rate of the filter directly determines the multiplication rate. Still, the multiplication rate and hardware effort are determined by the filter type. For equiripple finite impulse response (FIR) low-pass filters, the number of coefficients N depends on the pass-, transition-, and stop-band width, and thus on the OSR [3]. The relation between N and the OSR is given by the proportionality $N \sim OSR/(OSR - 1)$. Hence, as the OSR approaches 1, the number of coefficients explodes. Although this number cannot be infinite in practice, it can be concluded that placing the fractional part of SRC at a very low sample rate in a cascaded multirate system is not the optimal solution to minimizing the number of coefficients, at least if equiripple FIR low-pass filters are employed.

In Fig. 5 a cascaded multirate structure for SRC is sketched, indicating the increase of hardware effort as the sample rate, and thus the OSR, decreases. This leads to a (theoretically) constant effort in terms of operations per time unit regardless of the position in the cascaded structure.

Different filter types have different advantages and disadvantages. This leads to the fact that placing fractional SRC at a high sample rate in a cascaded multirate system would lead to applying another type of filter compared to the type which would be used in case of placing fractional SRC at a low sample rate in the cascade. For instance, if the OSR is high, comb filters merely suppressing the aliasing components are a very efficient choice. Still, for low OSRs such filters are not sufficient.

Generally, the advantages of placing fractional SRC at a high sample rate are:

- Lower complexity due to relaxed requirements.
- If noise-shaping ADCs with low output word length feed the filter, coefficient multipliers can be realized by simple switches or lookup tables of relatively small size.
- The complete system for SRC has greater flexibility. The fractional part runs on the fixed input sample rate, while the integer part can be made parameterizable with respect to its output rate. If placed at the end of the cascade, the fractional part would have to run on different output sample rates, making it more difficult to find efficient solutions.

The disadvantages are:

- The high clock rate at which the filters have to run
- The required high aliasing attenuation for multichannel signals

From these advantages and disadvantages the pros and cons of placing fractional SRC at a lower sample rate can be derived. Since the requirements on the filters are stronger with lower OSRs, the effort for fractional SRC at a lower sample rate is higher. This is mainly reflected by the number of coefficients of the filter, and thus the number of multipliers. If the sample rate is so low that time-division hardware sharing is possible, several coefficient multipliers can be realized by a small number of MAC units of a DSP. In a field programmable gate array (FPGA) or an application-specific integrated circuit (ASIC) based implementation, each coefficient is usually realized by its own multiplier. Therefore, in this case it is far more advantageous to place fractional SRC at a high sample rate. This enables the application of simple comb filters whose implementation requires only a small number of multipliers.

> *Being a fundamental signal processing task, integer factor SRC is not only a part of a cascade for rational factor SRC but is also a part of any system employing oversampling.*

Systems for Integer Factor SRC

Being a fundamental signal processing task, integer factor SRC is not only a part of a cascade for rational factor SRC but is also a part of any system employing oversampling. In such systems it is necessary to convert between an oversampled rate and a processing rate, which could be the chip or symbol rate of a certain communications standard.

The *direct approach* of realizing integer factor SRC is determined by the fact that downsampling has to be preceded by filtering in order to avoid aliasing, while upsampling can be followed by filtering in order to remove spectral images. Thus, the filter is always placed at the high sample rate. Principally, any kind of filter can be used, insofar as it obeys a given tolerance scheme for magnitude and phase response.

The signal at the input to an interpolation filter comprises $L - 1$ zeros between two consecutive samples of the original signal. At the output of the decimation filter $M - 1$ out of M samples are deleted. Exploiting this, the filters can be combined with the upsampler or downsampler, respectively. This leads to solutions much more efficient than the direct ones. Their main parts are clocked at the lower of the two involved sample rates. These solutions are based on the polyphase representation of both the signal and the impulse response of the filter, and can be derived, for example, from the corresponding block processing structure [6].

Block processing is a very graphic way of analyzing systems for SRC. It is as simple as *constructing a system with* M *inputs and* L *outputs which takes blocks of* M *samples of the input signal and generates blocks of* L *samples of the output signal*. Such systems are generally time-invariant, enabling the use of the large apparatus of time-invariant system theory, which is the great advantage of applying block signal processing in this respect. The relation between a periodically time-varying system (e.g., Fig. 4) and the respective time-invariant block processing system is given by the so-called *lifting isomorphism* or *raising procedure* [1, 7]. It is based on the state-space description of linear systems. For integer factor SRC the well-known structures of the polyphase-decimator and polyphase-interpolator result. The hardware platform of these structures can be made independent from the rate change factor by just foreseeing one polyphase branch and changing the set of coefficients periodically. For details on polyphase structures the reader is referred to [3, 8].

A disadvantage of these solutions is that L sets of coefficients of the impulse response $h(t)$ (sampled with $1/T_1$ and L different offsets ${}_mT_1$) must be stored. Especially for large values of L a considerable size of coefficient memory results. This can be avoided if the coefficients are not stored but calculated on demand. If the effort for calculating the coefficients is less than the effort for storing them, this approach is sensible. In order to keep the effort low for calculating the coefficients, the impulse response $h(t)$ must be a function (or must be approximated by a function) of relatively low complexity. Polynomials are such functions. Thus, $h(t)$ can be described by piecewise polynomials. Combining polynomial filtering and block processing leads to structures which can be realized very efficiently. One of those is the Farrow structure, resulting from implementing a polynomial impulse response on a polyphase interpolator [9, 2]. Although it is very efficient, there are still several multipliers required in the Farrow structure. The actual number depends on the length of the impulse response of the filter and the degree of the polynomials of which the impulse response consists. Therefore, the Farrow structure and similar implementations of polynomial impulse responses should be realized on a DSP at a low sample rate where the few MAC units of the DSP can realize all necessary multipliers in a time-shared manner.

Another class of simplified structures for integer-factor SRC results from sensible factorizations of the transfer function of certain filter types. These are cascaded comb filters [10] and cascaded integrator comb (CIC) filters [11]. Especially, the latter have enjoyed great success and wide application. They are multiplierless comb-filters with low effort. Due to the small width of their stop-bands they should be applied at high OSRs of the channel of interest. Their simple structure supports an implementation at high sample rates.

Finally, it should be mentioned that the application of block processing also leads to novel filter structures employing time-varying feedback loops [12, 13]. These structures exhibit very low complexity with respect to the state-space (and thus, the number of registers).

Systems for Fractional SRC

Implementing Eq. 2 as shown in Fig. 4 can be regarded as the *direct approach* to realizing fractional SRC. A time-invariant realization of the filter on the high intermediate sample rate $1/T_0$ circumvents additional controlling effort, which is necessary in time-varying systems. Still, very quickly the intermediate sample rate can reach values that are not realizable. As mentioned above, the intermediate sample rate is directly

Time-varying polyphase structures for fractional SRC can be obtained from combining a polyphase interpolator with a downsampler, or an upsampler with a polyphase decimator.

determined by the ratio of the two rates at the input and output of the sample rate converter. Thus, realizations according to Fig. 4 are only feasible in certain applications where the fractional rate-change factor $(L/M)_{\text{frac}}$ is given as a ratio of two small positive integer numbers L and M.

If the impulse response of the reconstruction filter can be realized by a comb filter, its implementation as a CIC filter can result in reduced effort, as in the case of integer factor SRC. However, parts of the filter have to be clocked at the high intermediate sample rate, which again limits the application to small numbers L and M.

Directly employing block processing is not a solution either, due to the fact that a system with M inputs and L outputs is hardly realizable if L or M get large. Moreover, the dependence of the number of inputs and outputs on the rate change factor makes such systems unsuitable for arbitrary rate change factors, which is, however, necessary in software radio applications.

The limitations of implementing time-invariant filters (on a high intermediate sample rate, or as block processing filters) for fractional SRC suggest the application of time-varying structures. A straightforward solution would be to realize Eq. 2 directly. This leads to time-varying polyphase structures.

Time-varying polyphase structures for fractional SRC can be obtained from combining a polyphase interpolator with a downsampler, or an upsampler with a polyphase decimator. Hence, it is simply a certain partitioning of Fig. 4 where either the upsampler and filter are realized by a polyphase interpolator, or the filter and downsampler are realized by a polyphase decimator. The great advantage of these structures is that they are exactly the same as the polyphase structures for integer factor SRC (interpolation and decimation). Only the additional downsampling or upsampling processes have to be taken into account. This is done by appropriately selecting the respective polyphase defined by the intersample position $_m$. It is also a simple matter to extend the application of polynomial filters to fractional SRC. Hence, the Farrow structure as an efficient realization of polynomial polyphase filters is also applicable [14], but with the restriction to low sample rates.

Polyphase realizations can also be given for CIC filters. They can be found, for example, by exploiting the block processing idea. The result is time-varying CIC-filters for fractional SRC with arbitrary factors [15, 16] which provide a means to realize fractional SRC at high sample rates.

A Case Study

Although concrete solutions and structures for SRC are beyond the scope of this article, an illustrative example should be given. It is based on references given at the end of the article. They should help the reader fully comprehend the example.

The object of the example is designing a system for fractional SRC which is part of a receiver for the Global System for Mobile Communications (GSM), IS-95, and Universal Mobile Telecommunications System (UMTS) air interfaces. Digitization of the signal at intermediate frequency (IF) with a sample rate of 80 Msamples/s and sufficient dynamic range of the ADC is taken for granted. Furthermore, it is assumed that the analog filtering before digitization is solely for anti-aliasing. Hence, 40 MHz of (real) bandwidth is available, comprising the channel of interest as well as adjacent channel interferers.

Two solutions to SRC should be examined that are principally shown in Fig. 6: time-varying CIC filters [15, 16] placed first in a cascaded structure, and the Farrow structure [14] running at twice the chip or bit rate of the current mode of operation. Both approaches have been subject to investigation in the context of the ACTS project Software Radio Technology supported by the European Commission. Time-varying CIC filters have been chosen for an implementation in a demonstrator of this project.

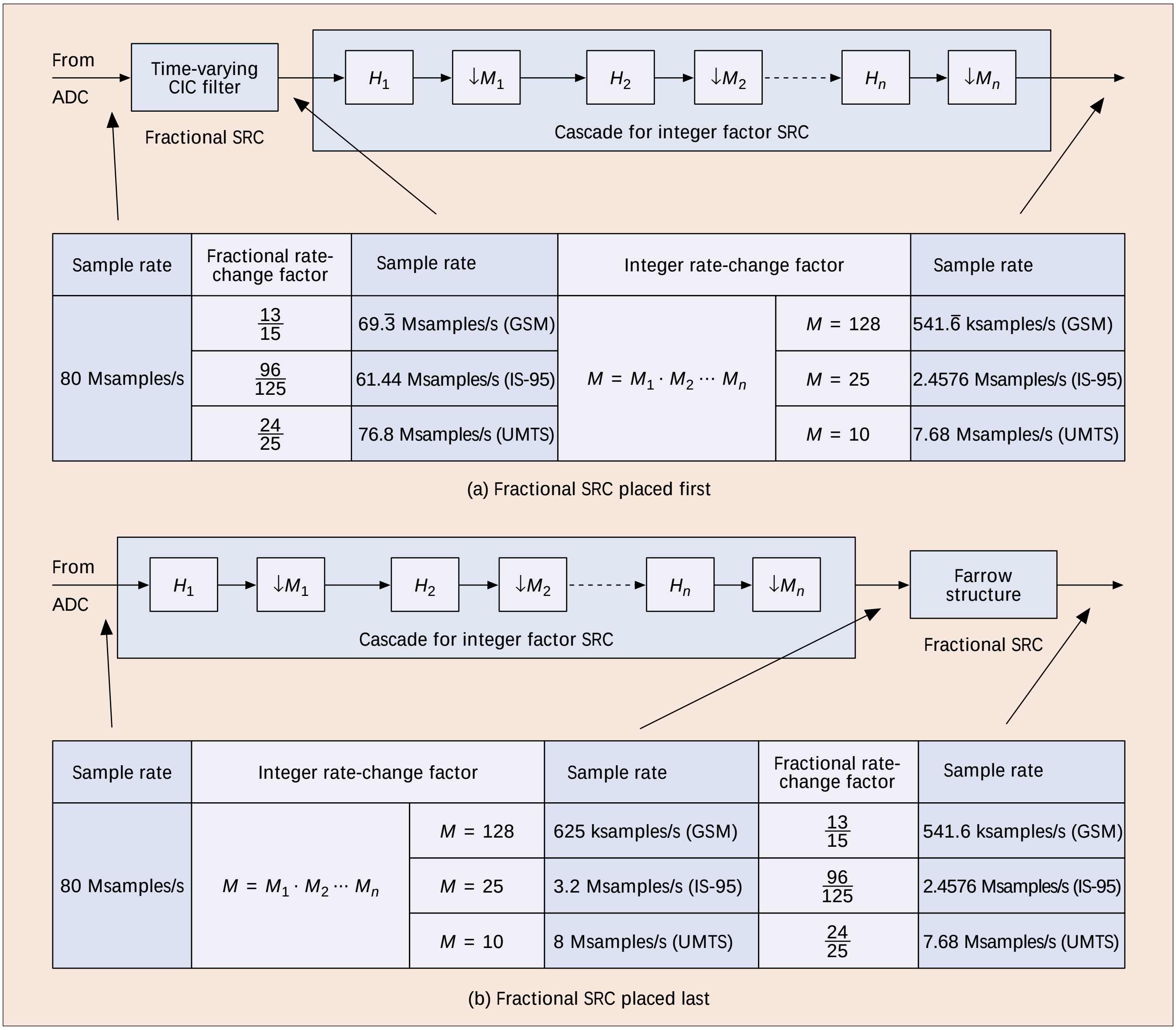
Placing fractional SRC at the end of a cascaded structure for SRC means that all adjacent channel interferers have been attenuated before. Potential aliasing components can only stem from spectral replications of the channel of interest itself, which can distort the signal. Therefore, an aliasing attenuation of 40 dB is claimed to be sufficient for fractional SRC. For an OSR of 10 this attenuation can be achieved by means of linear interpolation (i.e., a linear interpolator suppresses images of a 10x oversampled signal by 40 dB at minimum).

In order to achieve this OSR the signal must be upsampled by 10. The resulting nine spectral images must be attenuated. In order to keep up with the linear interpolator, an attenuation of 40 dB is required for the anti-imaging filter. This can be achieved with an equiripple low-pass filter with approximately 40 taps.

An implementation of the anti-imaging filter in conjunction with the linear interpolator on the Farrow structure circumvents high intermediate sample rates (e.g. a 10x increased input sample rate) at the output of the equiripple low-pass filter. It results in a time-varying system requiring 2 x 40 coefficients to be implemented. The doubling of the number of coefficients comes from the linear interpolation combined with the filtering. The filter implemented on the Farrow structure has a piecewise linear polynomial impulse response (straight lines connecting the 40 coefficients). Hence, to describe each of the linear polynomial pieces two values are needed.

Since the incoming signal to the system is (virtually) 10x oversampled, only four samples of the impulse response are used at a time. This results in a requirement of eight multipliers for calculating the required samples from the linear polynomial pieces. For each incoming signal sample different polynomial pieces are sampled. Therefore the 8 multipliers must be general purpose and cannot be fixed coefficient multipliers. Additionally, one general purpose multiplier is necessary for the calculation of the linear interpolation itself.

Principally any desired fractional rate-change factor can be implemented on this system without the need to change the number of coefficients. This is due to the fact that the actual SRC with arbitrary rational factors takes place in the linear interpolator having a continuous-



Sample rate	Fractional rate-change factor	Sample rate	Integer rate-change factor		Sample rate
80 Msamples/s	$\frac{13}{15}$	69.$\overline{3}$ Msamples/s (GSM)	$M = M_1 \cdot M_2 \cdots M_n$	$M = 128$	541.$\overline{6}$ ksamples/s (GSM)
	$\frac{96}{125}$	61.44 Msamples/s (IS-95)		$M = 25$	2.4576 Msamples/s (IS-95)
	$\frac{24}{25}$	76.8 Msamples/s (UMTS)		$M = 10$	7.68 Msamples/s (UMTS)

(a) Fractional SRC placed first

Sample rate	Integer rate-change factor		Sample rate	Fractional rate-change factor	Sample rate
80 Msamples/s	$M = M_1 \cdot M_2 \cdots M_n$	$M = 128$	625 ksamples/s (GSM)	$\frac{13}{15}$	541.6 ksamples/s (GSM)
		$M = 25$	3.2 Msamples/s (IS-95)	$\frac{96}{125}$	2.4576 Msamples/s (IS-95)
		$M = 10$	8 Msamples/s (UMTS)	$\frac{24}{25}$	7.68 Msamples/s (UMTS)

(b) Fractional SRC placed last

Figure 6. *A cascaded structure for SRC (down to twice the bit/chip rate of the standard of operation).*

time impulse response. Any sample of this impulse response can be calculated. For further details see [14].

When placing fractional SRC first in the cascade (i.e., directly after the ADC) and digital downconversion of the signal to baseband, all adjacent channel interferers are still part of the signal. Hence, the aliasing attenuation of the filter for SRC must be much higher than in the previous case. This is where the anti-aliasing constraint is important. It states that the required high aliasing attenuation is necessary only in the very frequency bands which contribute to aliasing in the channel of interest. Comb filters suppressing these aliasing components result from this idea. For fractional SRC, time-varying CIC filters should be chosen. They achieve high attenuation in a narrow bandwidth (i.e., for high OSRs) and less attenuation as the bandwidth of the aliasing bands increases (i.e., the OSR decreases). Within the context of this example second-order time-varying CIC filters suffice. Concretely, they achieve 110 dB aliasing attenuation for GSM, 70 dB for IS-95, and 50 dB for UMTS. A second-order time-varying CIC filter requires one general-purpose multiplier and one fixed coefficient multiplier. The latter requires one coefficient to be stored in memory. For further details on time-varying CIC filters see [16].

Assuming a digitization rate of 80 Msamples/s, the effort of the two solutions is compared in Table 1. The Farrow structure is implemented on two times the target rate of the standard of operation (i.e., two times the bit rate of GSM, and two times the chip rate of IS-95 or UMTS, respectively). With 270.83 ksamples/s the bit rate of GSM, 1.2288 Mchips/s the chip rate of IS-95, and 3.84 Mchips/s the chip rate of UMTS, the multiplication rates in million multiplications per second can be calculated.

It can be concluded that the hardware effort for the time-varying CIC filter is much lower,

	Farrow structure (first-order interpolation, 40 taps)	Time-varying CIC filter (second-order)
Number of general-purpose multipliers	9	1
Number of fixed coefficient multipliers	0	1
Number of coefficients to store	80	1
Multiplication rate (millions of multiplications per second)		
for GSM	4.9	160
for IS-95	22	160
for UMTS	69	160

Table 1. *Efforts of the Farrow structure vs. time-varying CIC filter for SRC.*

while the multiplication rate is lower for the Farrow structure. Hence, the Farrow structure is advantageously implemented if hardware effort is not an issue, or time-shared hardware is available (e.g., on a DSP), while time-varying CIC filters have the great advantage of a small occupied chip area in an ASIC or FPGA.

Conclusions

Digital communications standards are generally based on different master clock rates. Thus, sample rate conversion is required in systems that process signals of different communications standards. In the context of software-defined radio sample rate conversion must be realized digitally.

Sample rate conversion is a process of resampling. Since any sampling process causes aliasing and imaging, resampling also does. These two fundamental characteristics of sampling require filtering. Thus, the design of systems for SRC is mainly a filter design problem. Since it is the very characteristics which can destroy the signal, *aliasing is the most important effect of SRC* which must be avoided by proper filter design.

Since the constraints on the filters are most relaxed at high oversampling ratios of the signal of interest, it is reasonable to place the most complex part of SRC, namely fractional SRC, at high sample rates.

Feasible implementations for integer factor SRC can be found by realizing block processing structures, or by properly factorizing the transfer functions of the filters. Among the first are the well-known polyphase decimators and interpolators, while CIC filters belong to the second group.

Based on time-varying realizations of the combination of block filters for integer factor SRC with either an up- or a downsampler, systems for fractional SRC can be derived. These structures are parameterizable, and thus can adopt arbitrary rate change factors, which is required for software radio applications.

Acknowledgment

This work has been partially supported by the European Commission ACTS Software Radio Technology project.

References

[1] T. Hentschel, M. Henker, and G. Fettweis, “The Digital Front-End of Software Radio Terminals,” *IEEE Pers. Commun.*, vol. 6, no. 4, Aug. 1999, pp. 40–46.

[2] H. Meyr, M. Moeneclaey, and S. A. Fechtel, *Digital Communication Receivers: Synchronization, Channel Estimation and Signal Processing*, Wiley, 1998.

[3] R. E. Crochiere and L. R. Rabiner, *Multirate Digital Signal Processing*, Prentice Hall, 1983.

[4] G. P. Fettweis *et al.*, “Breaking New Grounds Over 3000 MOPS: A Broadband Mobile Multimedia Modem DSP,” *ICSPAT ’98*, Toronto, Canada, Sept. 13–16, 1998.

[5] M. Weiss, F. Engel, and G. P. Fettweis, “A New Scalable DSP Architecture for System on Chip (soc) Domains,” *ICASSP ’99*, Phoenix, AZ, Mar. 15–19, 1999.

[6] T. Hentschel, *Systems for Sample Rate Conversion in Software Radio*, Ph.D. thesis, in preparation, Dresden Univ. of Tech., 2000.

[7] A. Feuer and G. C. Goodwin, *Sampling in Digital Signal Processing and Control*, Birkhauser, 1996.

[8] N. J. Fliege, *Multirate Digital Signal Processing: Multirate Systems, Filter Banks, Wavelets*, Wiley, 1994.

[9] C. W. Farrow, “A Continuously Variable Digital Delay Element,” *Proc. IEEE Int’l. Symp. Circuits and Sys.*, Espoo, Finland, June 1988, pp. 2641–45.

[10] M. E. Frerking, *Digital Signal Processing in Communication Systems*, Van Nostrand Reinhold, 1994.

[11] E. B. Hogenauer, “An Economical Class of Digital Filters for Decimation and Interpolation,” *IEEE Trans. Acoustics, Speech and Sig. Proc.*, vol. ASSP-29, no. 2, Apr. 1981, pp. 155–62.

[12] T. Hentschel and G. P. Fettweis, “Reduced Complexity Comb-Filters for Decimation and Interpolation in Mobile Communications Terminals,” *Proc. 6th IEEE Int’l. Conf. Elect., Circuits and Sys.*, Papfos, Cyprus, Sept. 5–8, 1999, vol. 1, pp. 81–84.

[13] T. Hentschel and G. P. Fettweis, “Time-Varying Recursive Filters for Decimation and Interpolation,” *Proc. 10th Euro. Sig. Processing Conf.*, Tampere, Finland, Sept. 5–8, 2000.

[14] L. Lundheim and T. A. Ramstad, “An Efficient and Flexible Structure for Decimation and Sample Rate Adaptation in Software Radio Receivers,” *Proc. ACTS Mobile Commun. Summit*, June 1999, pp. 663–68.

[15] T. Hentschel, M. Henker, and G. P. Fettweis, “Sample Rate Conversion in Software Radio Terminals,” *ACTS Mobile Commun. Summit*, Sorrento, Italy, June 1999, pp. 733–38.

[16] M. Henker, T. Hentschel, and G. P. Fettweis, “Time-Variant CIC-Filters for Sample-Rate Conversion with Arbitrary Rational Factors,” *IEEE 6th Int’l. Conf. Elect., Circuits and Sys.*, Paphos, Cyprus, 5.-8. Sept. 1999, pp. 67–70.

Biography

Tim Hentschel [S’93] (hentsch@ifn.et.tu-dresden.de) received his M.Sc./Dipl.-Ing. degree in electrical engineering from King’s College London, University of London, U.K., and Dresden University of Technology, Germany, in 1993 and 1995, respectively. From 1995 to 1996 he was with Philips Communications Industries, Nurnberg, Germany. Since May 1996 he has been with the Mannesmann Mobilfunk Chair for Mobile Communications Systems, working toward his Ph.D. His current research interests include software radio, specifically the design and investigation of digital signal processing algorithms for reconfigureable front-ends.

Gerhard Fettweis [S’82-M’90-SM’98] (fettweis@ifn.et.tu-dresden.de) received his M.Sc./Dipl.-Ing. and Ph.D. degrees in electrical engineering from Aachen University of Technology (RWTH), Germany, in 1986 and 1990, respectively. From 1990 to 1991 he was a visiting scientist at the IBM Almaden Research Center, San Jose, California, working on signal processing for disk drives. From 1991 to 1994 he was a scientist with TCSI, Berkeley, California, responsible for signal processor developments for mobile phones. Since September 1994 he has held the Mannesmann Mobilfunk Chair for Mobile Communications Systems at Dresden University of Technology, Germany. He is an elected member of the SSC Society’s Administrative Committee and the IEEE ComSoc Board of governors since 1999 and 1998, respectively. He has been associate editor for *IEEE Transactions on CAS II*, and is now associate editor for the IEEE J-SAC Wireless Series.