Assignment EE6401 2019

Submission Deadline: 14 Oct. 2019

- The hardcopy of assignment report should be submitted to S1-B1a-27 by the above date.
- Part-time student can submit the assignment report via email to egbi@ntu.edu.sg
- Your submission will not be returned.
- **Heavy PENALTY will be enforced for any form of PLAGARISM and CHEATING in your assignment report.**

Question:

We have learns the concept of sampling rate conversion and a few techniques of designing computationally efficient multi-rate signal processing systems. These systems mainly including up/down samplers and a filter, which is generally considered to processing the system in the time domain.

This assignment considers another sampling rate conversion technique performed in the frequency domain. The input signal is firstly divided into many small segments. Each segment is transformed into the frequency domain and its spectrum is manipulated by magnitude scaling and spectrum shifting. Then the modified spectrum is transformed into the time domain as the output with a new sampling frequency.

Please read the paper a few times for a better understanding and to compare the techniques you have learnt from the lectures and the paper.

1. Briefly compare the basic principles to perform the sampling rate conversion in the time domain and in the frequency domain.
2. Both methods have problems of distortions in the output signals. Discuss the sources of the distortions in these methods
3. Compare the required computational complexities required by these methods in terms of the number of multiplications/per output sample.

Guoan Bi and Sanjit K. Mitra

Sampling Rate Conversion in the Frequency Domain

"DSP Tips and Tricks" introduces practical design and implementation signal processing algorithms that you may wish to incorporate into your designs. We welcome readers to submit their contributions. Contact Associate Editors Rick Lyons (R.Lyons@ieee.org) or Clay Turner (clay@claysturner.com).

Sampling rate conversion (SRC) is usually performed in the time domain by using the operations of up-sampling, filtering, and down-sampling. However, it is also possible to perform the SRC in the frequency domain by formulating the desired spectrum from the spectrum of an input signal. This article shows how to perform SRC for both integer- and fractional-rate conversion by manipulating the discrete Fourier transform (DFT), implemented using the fast Fourier transform (FFT), of a time-domain signal. The analysis on error performance and the required computational complexities show that by using the FFT, for both short and long input sequences, improvements in conversion accuracy is achieved at reduced computational costs.

SRC BASICS

The basic time-domain operations used to perform SRC are up-sampling by an integer-valued interpolation factor I, low-pass filtering, and down-sampling by an integer-valued decimation factor D, as shown in Figure 1(a) [1], [2]. The corresponding effects of SRC in the frequency domain are also well explained in the literature. However, although some attempts have been made in the past, SRC based on frequency-domain processing has not received sufficient attention. Here we remedy that situation.

A DFT-based method for interpolation was first reported in [3]. Later, it was shown in [4] that decimation was also achieved in the DFT domain. Interpolation by an integer factor was further investigated with a focus on the issues of computational complexity and the method of formulating the spectrum of the interpolated signals [5], [6]. Finally, an experimental study on the error performance of interpolation was reported in [7]. These reported works did not consider SRC for fractional-rate conversion, nor did they consider how to accommodate long input signal sequences.

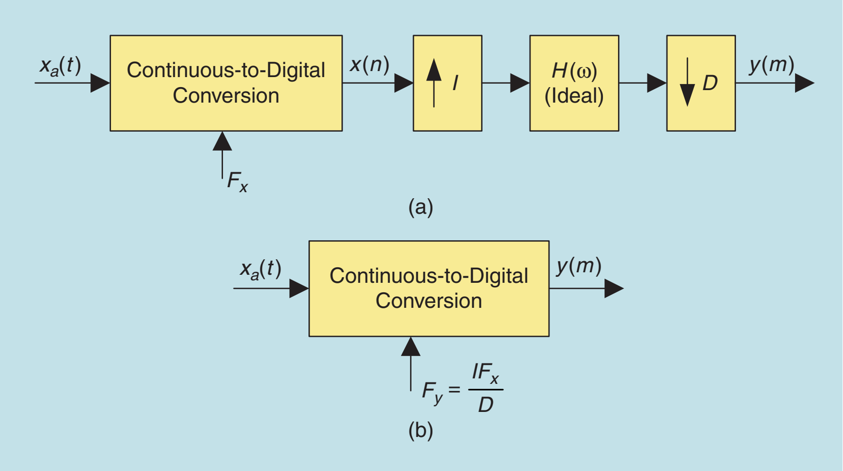
REQUIREMENTS IN THE FREQUENCY DOMAIN

Let $x_a(t)$ denote an analog signal and $x(n)$ the corresponding discrete-time sequence obtained with a sampling frequency F_x as shown in Figure 1(a). The objective of the SRC is to convert the sequence $x(n)$ into another sequence $y(m)$, which is ideally the one obtained by sampling $x_a(t)$ with another sampling frequency F_y, where $F_y/F_x = I/D$, as shown in Figure 1(b). Factors D and I are the decimation and interpolation factors, respectively. For this discussion we assume the low-pass filter $H(\omega)$ in Figure 1(a) is ideal in its operation and that the two $y(m)$ sequences in Figure 1 are identical.

The magnitude response of the digital filter is specified by

$$|H(\omega_v)| = \begin{cases} I, & |\omega_v| \le \min\left(\dfrac{\pi}{I}, \dfrac{\pi}{D}\right), \\ 0, & \text{otherwise}, \end{cases} \tag{1}$$

where ω_v is the post-interpolation sample rate $\omega_v = 2\pi f/IF_x$ radians/sample. Various design techniques to reduce the computational complexity for the filtering operation have been investigated to achieve the desired performance [1], [2]. The effects of the filter operation in the frequency domain have also been well explained.



[FIG1] Time domain SRC: (a) computing $y(m)$ from $x(n)$ and (b) generating $y(m)$ directly.

Digital Object Identifier 10.1109/MSP.2011.940413
Date of publication: 19 April 2011

1053-5888/11/$26.00©2011IEEE

The spectrum of the output signal $y(m)$ after the SRC is expressed as

$$Y(\omega_y) = \begin{cases} \frac{I}{D}\cdot X\left(\frac{\omega_y}{D}\right), \\ \quad 0 \le \omega_y \le \min\left(\pi, \frac{D\pi}{I}\right), \\ 0, \text{ otherwise}, \end{cases} \tag{2}$$

where $\omega_y = D\omega_x/I$ radians/sample, and $X(\omega_x)$ is the spectrum of the input signal $x(n)$. Equation (2) shows that $Y(\omega_y)$, the spectrum of the desired $y(m)$, can be obtained by scaling the magnitude of the spectrum $X(\omega_x)$ and the frequency in the horizontal direction with constant factors. Because $x(n)$ is assumed to be real, the spectrum of $X(\omega_x)$ in the frequency range from π to 2π is the conjugate mirror image of the spectrum in the range from 0 to π.

SAMPLE RATE DECREASE ($D > I$)

Let us define $X(k)$ to be the N-point DFT of the sequence $x(n)$ and $Y(k)$ to be the N_1-point DFT of the sequence $y(m)$, where $x(n)$ and $y(m)$ are obtained by sampling $x_a(t)$ at sampling frequencies F_x and F_y, respectively. For our SRC, $\hat{Y}(k)$, which is the N_1-point DFT of the output sequence $\hat{y}(m)$ where $N_1 = IN/D$, can be formulated by manipulating $X(k)$. Based on (2) $\hat{Y}(k)$, for $D > I$, is obtained by

$$\hat{Y}(k) = \begin{cases} \frac{I}{D}X(k), & 0 \le k < \frac{N_1}{2}, \\ X\left(\frac{N}{2}\right), & k = \frac{N_1}{2}, \\ \frac{I}{D}X(k+N-N_1), & \frac{N_1}{2} < k < N_1. \end{cases} \tag{3}$$

According to (3), the values of $X(k)$, where $(N_1/2) \le k \le (N/2)-1$ and $(N/2)+1 \le k \le N-(N_1/2)$, are ignored. The other values of $X(k)$ are rearranged as shown in Figure 2(a) and (b) to obtain the $\hat{Y}(k)$ spectrum. The desired $\hat{y}(m)$ time-domain sequence, ideally equal to $y(m)$, is then obtained by performing an inverse-DFT on $\hat{Y}(k)$.

Figure 3 shows the computed time-domain signal $\hat{y}(m)$ and its deviation from $y(m)$ whose DFT is given in Figure 2(c) when $D = 4$, and $I = 3$. We see in Figure 3 that in the time domain, $\hat{y}(m)$, has small differences from the desired signal $y(m)$. However, these errors become larger towards both ends of the signal segments.

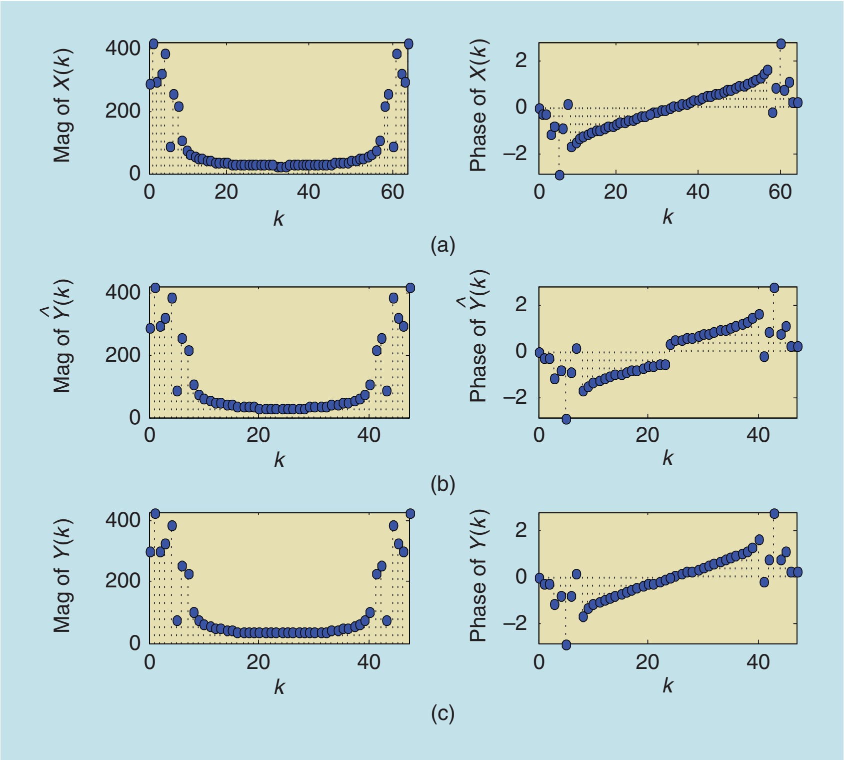
SAMPLE RATE INCREASE ($D < I$)

When $D < I$, the spectrum $X(k)$ is again used to form $\hat{Y}(k)$ by inserting selected numerical values into $\hat{Y}(k)$'s frequency region between $D\pi/I$ and $2\pi - D\pi/I$. In terms of the DFT $X(k)$, $\hat{Y}(k)$, the DFT of output signal $\hat{y}(m)$, is expressed by

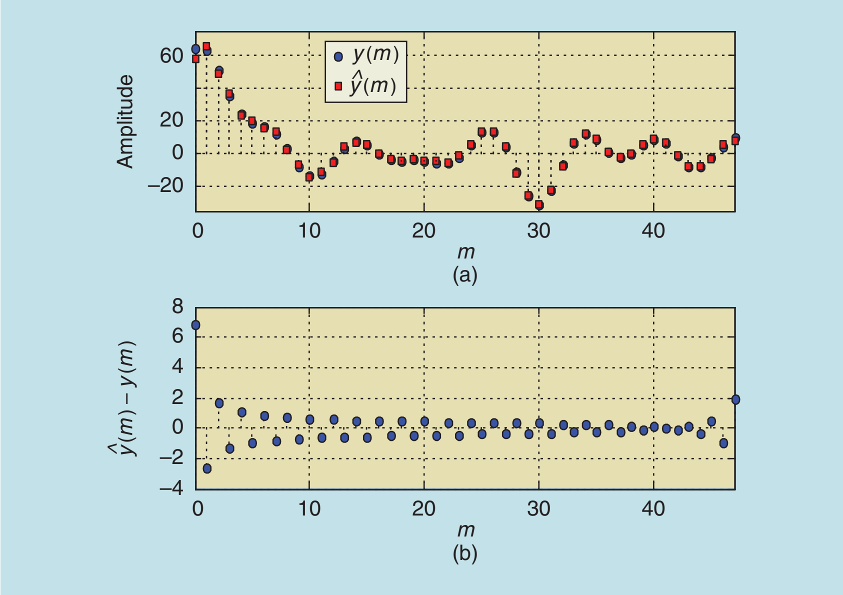
$$\hat{Y}(k) = \begin{cases} \frac{I}{D}X(k), & 0 \le k < \frac{N}{2}, \\ C_I, & \frac{N}{2} \le k \le N_1 - \frac{N}{2}, \\ \frac{I}{D}X(k-N_1+N), & N_1 - \frac{N}{2} < k < N_1. \end{cases} \tag{4}$$

In (4), the value of C_I should be carefully selected to minimize errors that cause $\hat{y}(m)$ to be different from the desired $y(m)$. Figure 4 shows the generation of $\hat{Y}(k)$ from $X(k)$ where $C_I = X(N/2)$. The value of C_I has a direct impact on the deviation of $\hat{Y}(k)$ from $Y(k)$. In general, we seek suitable values of C_I that result in as small errors as possible. To minimize any additional computation associated with C_I, we generally assume C_I to be a constant. For example, $C_I = 0$ is used in [5]. To minimize the errors due to the insertion of zeros, $\hat{Y}(N_1/2)$ was set equal to $X(N/2)$ in [6]. Other C_I values have been studied and the impact of their errors has been investigated by simulation [7].

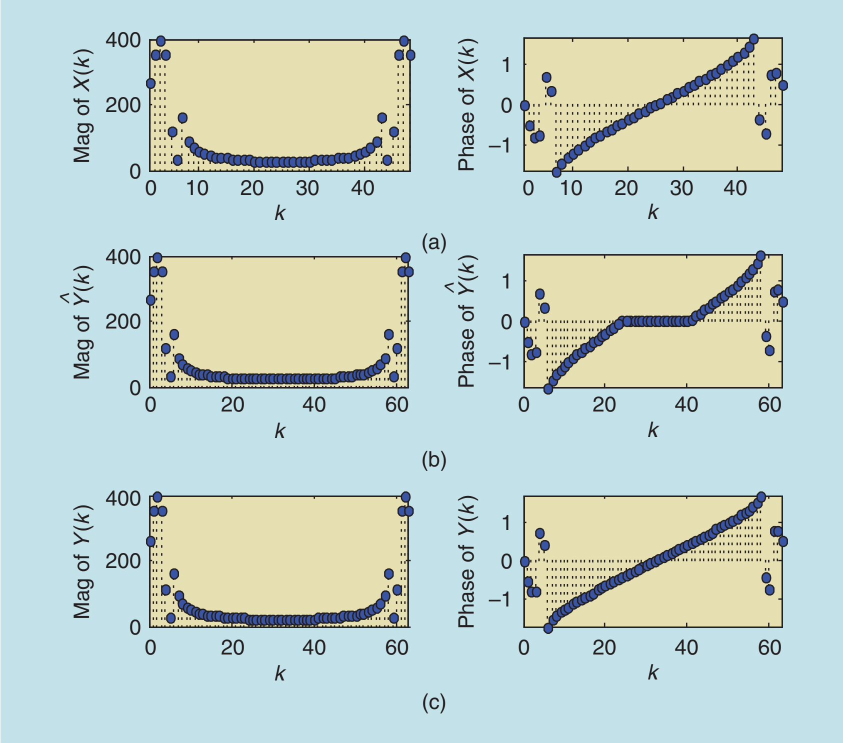
For both the cases illustrated in Figures 2 and 4, the final output sequence $\hat{y}(m)$ is obtained from the inverse DFT of $\hat{Y}(k)$. For comparison purposes, Figures 2(c) and 4(c) show the ideal $Y(k)$ spectra, obtained from a Figure 1(a) sampling scenario, which generally are different from the formulated $\hat{Y}(k)$ spectra. Therefore, the



[FIG2] Formulating the $\hat{Y}(k)$ spectrum of the $\hat{y}(m)$ output signal when $D = 4$, $I = 3$, and $N = 64$.



[FIG3] Figure 2 SRC errors: (a) $\hat{y}(m)$ and $y(m)$ sequences and (b) error sequence $\hat{y}(m) - y(m)$.



[FIG4] Formulating the $\hat{Y}(k)$ spectrum of the $\hat{y}(m)$ output signal when $D = 3$, $I = 4$, and $N = 48$.

computed $\hat{y}(m)$ has errors compared to the true $y(m)$ sequence in Figure 1(a).

OVERLAP APPROACH FOR LONG SEQUENCES

The method described above is only suitable for short input sequences due to the limitation of the maximum DFT length for practical applications. For long input sequences, a widely used approach is to divide the long input sequence into many shorter segments, which are processed individually. The outputs from these processed segments are then combined to form the required output. Such an arrangement is widely used in short-time Fourier transforms and other processing techniques.

If each $x(n)$ input time-domain segment has N points, Figure 5(a) shows that the adjacent input segments are overlapped by $2L$ points, and from each segment an N-point $X(k)$ DFT is computed. The previously described methods are then used to obtain an N_1-point $\hat{Y}(k)$ DFT segment for each $X(k)$ segment, where $N_1/N = I/D$. The value of L should be sufficiently large to effectively minimize the errors that are distributed near both ends of each output time-domain segment.

After performing an N_1-point inverse DFT on each $\hat{Y}(k)$ segment, the corresponding N_1-point time-domain segments are obtained. Finally, the cascading operation with L_1-point overlapping, where $L_1/L = I/D$ for each pair of adjacent time segments, is performed to recover the final $\hat{y}(m)$ time-domain sequence with the desired F_y sampling frequency, as shown in Figure 5(b), where the shaded time samples are discarded.

MEAN SQUARED ERRORS

In general, it is desirable to increase the length of the $2L$ overlap, and the length N of the DFT, so that the mean squared error (MSE) of $\hat{y}(m)$ can be effectively reduced. Let us now consider an example for the error performance affected by these factors. Figure 6 presents the MSEs, defined by $20\log\{[y(m) - \hat{y}(m)]^2/N_1\}$, where N_1 is the length of an overlapped output time segment. This experiment shows that increasing the length of the overlap definitely helps to minimize the errors. However, the effectiveness of error minimization is decreased when the length of overlap is too large because the errors located far away from the ends of each segment are relatively smaller than those located near the ends. We found that for $I > D$, the reduction of the MSEs is effective only when the length of the $X(k)$ DFT is $N_1 > 192$. In addition, a better selection of the C_I value has substantial influence on error minimization. For example, the errors obtained using $C_I = 0$ are much larger than those measured by using $C_I = X(N/2)$.

COMPUTATIONAL COMPLEXITY

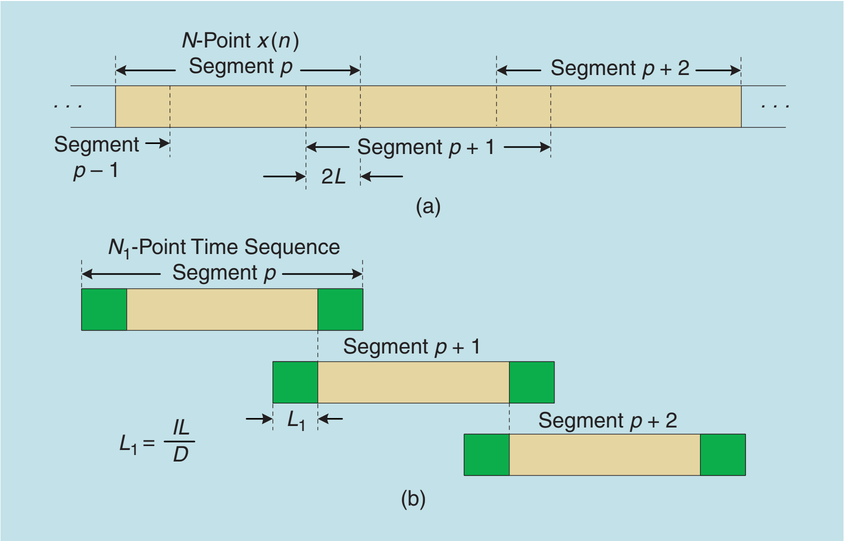
Because factors I and D can be any integer value, we require flexible FFT algorithms that accommodate various input sequence lengths. For example, fast $q{\cdot}2^n$-length FFT algorithms, where q is a small odd integer, were reported in [8]–[10]. By using computationally efficient FFT algorithms for real input sequences, we have estimated the required computational complexity in terms of the number of real additions and real multiplications per data point.

Figure 7 shows the computational complexity for two SRC cases with zero overlap. The analysis shows that the required computational complexity increases with the selected FFT size. This means that to achieve better conversion accuracy, FFT sizes should be increased, which accordingly requires more computation. When the length of overlap is increased, the computational complexity is proportionally increased by a factor of $N/(N-2L)$.

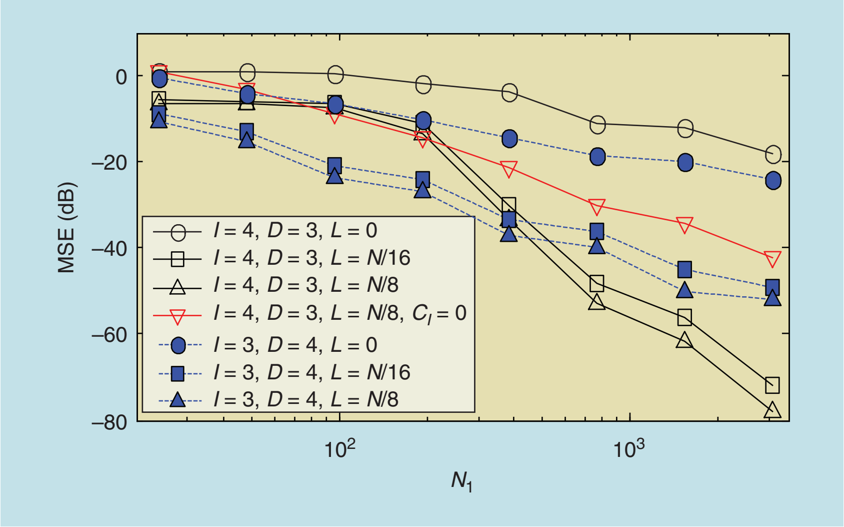
We compared the proposed SRC method with one using a polyphase finite impulse response (FIR) filter in the time domain. In this comparison, $I = 4$, $D = 3$, and the MSE $= -36$ dB. The $x(n)$ input signal had seven sinusoidal terms whose frequencies ranged from 200 to 1,930 Hz and an input sampling frequency of $F_x = 8{,}000$ Hz. To achieve the desired level of MSE, the FIR filter has a transition bandwidth of 100 Hz (from 1,950 and 2,050 Hz), a pass-band ripple = 0.1 dB, and a stop-band ripple = 60 dB. It was estimated that the FIR filter required at least 35 multiplications per data sample. For the proposed frequency-domain SRC method, we chose $N = 768$ and $L = N/16$ to achieve an MSE level of -45 dB. Our proposed approach required only eight multiplications per data sample.

CONCLUSIONS

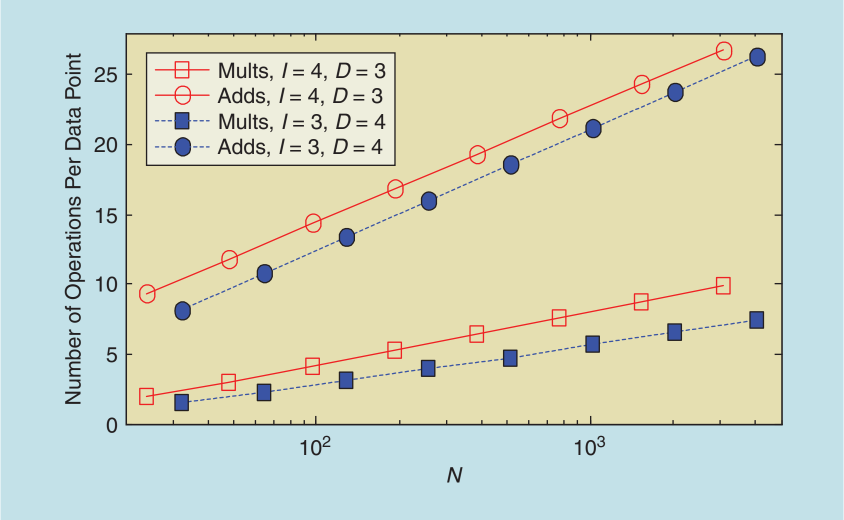
This article describes a frequency-domain method for SRC. We showed



[FIG5] Processing long sequences: (a) $x(n)$ input segmentation and overlapping and (b) $\hat{y}(m)$ output overlap and cascading.



[FIG6] SRC errors versus overlapped time segment length, N_1.



[FIG7] Number of frequency-domain SRC computations per data point versus $X(k)$ FFT size.

that a computationally simple SRC can be implemented by combining an efficient FFT procedure with a trivial deletion [sample rate decrease as in (3)], or insertion [sample rate increase as in (4)], of selected high-frequency spectral sample values. Using efficient FFTs, the computational complexity is substantially reduced compared to conventional time-domain SRC methods. Furthermore, with the overlapping technique, the conversion accuracy can also be increased for long sequences.

AUTHORS

Guoan Bi (egbi@ntu.edu.sg) is with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore.

Sanjit K. Mitra (skmitra@usc.edu) is with the Ming Hsieh Department of Electrical Engineering, University of Southern California, Los Angeles.

REFERENCES

[1] P. Vaidyanathan, *Multirate Systems and Filter Banks*. Englewood Cliffs, NJ, Prentice-Hall, 1993.

[2] S. K. Mitra, *Digital Signal Processing: A Computer-Based Approach*, 4th ed. New York: McGraw-Hill, 2011.

[3] B. Gold and C. Rader, *Digital Processing of Signals*. New York: McGraw-Hill, 1969.

[4] M. Yeh, J. Melsa, and D. Cohn, "A direct FFT scheme for interpolation, decimation and amplitude modulation," in *Proc. 16th Asilomar Conf. Circuits, Syst., Computers*, 1982, pp. 437–441.

[5] K. Prasad and P. Satyanarayana, "Fast interpolation algorithm using FFT," *Electron. Lett.*, vol. 22, no. 4, pp. 185–187, Feb. 1986.

[6] J. Adams, "A subsequence approach to interpolation using the FFT," *IEEE Trans. Circuits Syst.*, vol. CAS-34, no. 5, pp. 568–570, May 1987.

[7] D. Fraser, "Interpolation by the FFT revisited—An experimental investigation," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 37, no. 5, pp. 665–675, May 1989.

[8] G. Bi, Y. Chen, and Y. Zheng, "Fast algorithms for generalized discrete Hartley transform of composite sequence lengths," *IEEE Trans. Circuits Syst. II,* vol. 47, no. 9, pp. 893–901, Sept. 2000.

[9] G. Bi and Y. Chen, "Fast DFT algorithms for length N = q*2m," *IEEE Trans. Circuits Syst. II,* vol. 45, no. 6, pp. 685–690, June 1998.

[10] G. Bi and Y. Chen, "Fast generalized DFT and DHT algorithms," *Signal Processing*, vol. 65, no. 3, pp. 383–390, 1998.

[exploratory DSP] continued from page 131

[7] A. Milutinovic, A. M. Molnos, K. G. W. Goossens, and G. J. M. Smit, "Dynamic voltage and frequency scaling and adaptive body biasing for active and leakage power reduction in MPSoC: A literature overview," in *Proc. Program for Research on Integrated Systems and Circuits*, 2009, pp. 488–495.

[8] D. Ma and R. Bondade, "Enabling power-efficient DVFS operations on silicon," *IEEE Circuits Syst. Mag.*, vol. 10, no. 1, pp. 14–30, 2010.

[9] W. Kim, M. S. Gupta, G.-Y. Wei, and D. Brooks, "System level analysis of fast, per-core DVFS using on-chip switching regulators," in *Proc. IEEE Int. Symp. High Performance Computer Architecture*, 2008, pp. 123–134.

[10] D. N. Truong, W. H. Cheng, T. Mohsenin, Z. Yu, A. T. Jacobson, G. Landge, M. J. Meeuwsen, C. Watnik, A. T. Tran, Z. Xiao, E. W. Work, J. W. Webb, P. V. Mejia, and B. M. Baas, "A 167-processor computational platform in 65 nm CMOS," *IEEE J. Solid-State Circuits*, vol. 44, pp. 1130–1144, Apr. 2009.

[11] E. G. Larsson, "MIMO detection methods: How they work," *IEEE Signal Processing Mag.*, vol. 26, pp. 91–95, May 2009.

[12] M. Čirkić, D. Persson, and E. G. Larsson, "New results on adaptive computational resource allocation in soft MIMO detection," in *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing (ICASSP)*, May 2011.

[13] W. Wang, G. Choi, and K. Gunnam, "Low-power VLSI design of LDPC decoder using DVFS for AWGN Channels," in *Proc. Int. Conf. VLSI Design*, 2009, pp. 51–56.

[14] E. Amador, R. Knopp, V. Rezard, and R. Pacalet, "Dynamic power management on LDPC decoders," in *Proc. IEEE Symp. VLSI,* 2010, pp. 416–421.

[15] J. T. Ludwig, S. H. Nawab, and A. P. Chandrakasan, "Low-power digital filtering using approximate processing," *IEEE J. Solid-State Circuits*, vol. 31, no. 3, pp. 395–400, Mar. 1996.