Implementing Multirate Filters in FLEX Devices

TECHNICAL BRIEF 8 JUNE 1996

Finite impulse response (FIR) filters are used in many digital signal processing (DSP) systems to perform signal preconditioning, low-pass filtering, and anti-aliasing functions. Modifications of the FIR filter allow it to perform multirate filtering, including decimation and interpolation. Multirate filters are digital filters that change the sampling rate of a digitally represented signal. These filters convert one set of input samples to another set of data that represents the same analog signal sampled at a different rate. A system incorporating multirate filters can process data sampled at various rates. The two types of multirate filtering processes are decimation and interpolation. Decimation reduces the sample rate of a signal to eliminate redundant information and compact the data, allowing more information to be stored, processed, or transmitted. Interpolation increases the sample rate of a signal to fill in missing information between samples of a signal.

Applications for multirate filters include:

- Digital reconstruction/anti-alias filters for digital audio
- Down-sampling a system that employs over-sampling
- Image resizing
- Narrow-band spectra calculation for vibration analysis
- Sample-rate conversion between digital audio systems
- Sub-band coding for vocoder speech processing

One example of sample-rate conversion aligns two digital systems of unequal sample frequency (i.e., DAT at 48 kSps to CD at 44.1 kSps). The signal is first up-sampled (interpolated) to a common multiple, filtered, and then down-sampled (decimated) to the required sampling rate. This results in a sample rate conversion that retains the fidelity of the original signal.

Decimating Filter for Sample Rate Reduction

Decimation is commonly used in digital systems that employ over-sampling techniques. Many DSP systems employ over-sampling (i.e. 8x) for purposes such as noise shaping. The final stages of low-pass filtering and decimation are combined into a decimating filter that converts the oversampled rate back to the original rate.

A decimating filter only computes every *n*th result, where *n* is the decimation factor. Decimating filters are created by simply discarding any unwanted results from a regular FIR filter. For example, if the input data rate is 100 MHz, to decimate the data rate by 2, the output data rate is only 50 MHz. See Figure 1. However, this implementation is inefficient. The low-pass FIR filter must run at the XIN data rate (i.e., 100 MHz), which wastes half of the computations performed by the FIR filter.

Figure 1. Typical Decimating Filter

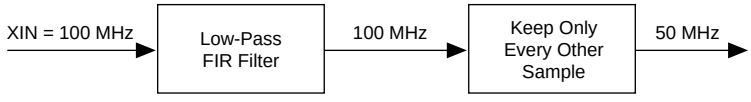
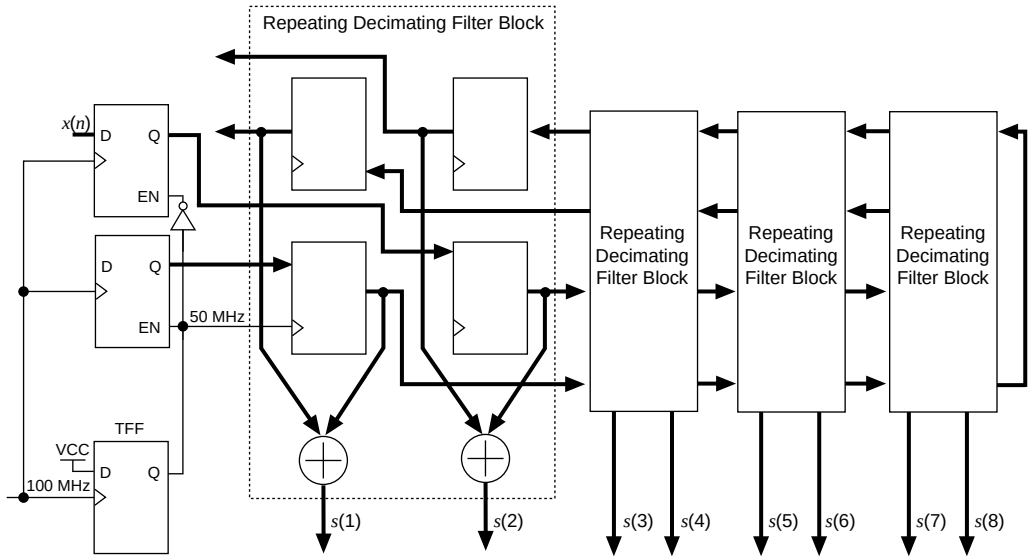


Figure 2 shows the shift register section of a 16-tap, decimate-by-2 FIR filter. This filter only computes every other result (i.e., the data skips every other register). Therefore, most of the filter runs at the output rate rather than the input rate, saving about 50% of the power consumed by the full-speed filter. The filter in Figure 2 also permits a faster input sample rate. For example, if the input data rate is 100 MHz, only the T flipflop must run at the 100-MHz rate—the rest of the FIR filter runs at 50 MHz.

Figure 2. FLEX Decimating Filter



Interpolating Filter

An interpolating filter (i.e., an up-sampling filter) performs the opposite function of a decimating filter: it increases the sample rate by a factor of *n*. One way to perform interpolation is to add extra zero-value samples between each input data sample. See Figure 3. The data stream (with the zeros) is sent through a low-pass filter; this output data rate is at a higher sample rate than the input data rate.

Figure 3. Stuffing Zeros into the Data Stream

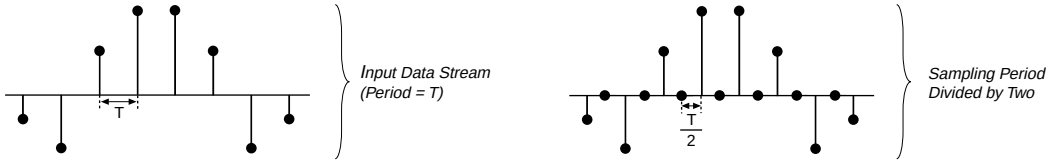
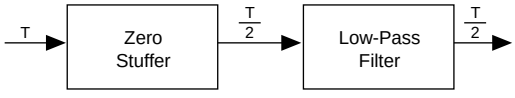


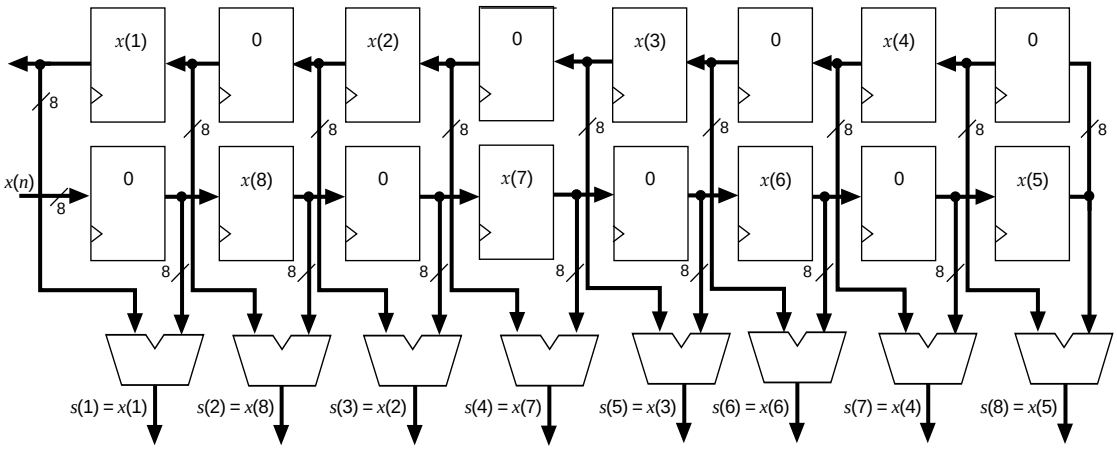
Figure 4 shows the interpolating filter block diagram.

Figure 4. Interpolating Filter



Because much of the input data to the low-pass filter is zero, the shift register is always sparsely populated. Figure 5 shows an example of an interpolating filter where the data is up-sampled by two. One input to each of the symmetric tap adders is always zero. Therefore, the adder is not necessary.

Figure 5. Up-Sampling Data by Two



AN 73 (Implementing FIR Filters in FLEX Devices) provides further details on the optimization of decimation and interpolation functions. This application note illustrates the ease of converting a FLEX FIR filter into a decimation or interpolation filter by simply re-routing the shift register. This methodology provides great flexibility, enabling one FLEX device to be reconfigured for normal, decimation, and interpolation modes.

DSP Design Kit Available from Altera

FLEX devices provide the high performance of custom DSP solutions without sacrificing flexibility. Altera's DSP Design Kit includes customizable building blocks for implementing DSP functions. These building blocks include fully parallel FIR filters with parameterized features. DSP functions are optimized for Altera FLEX architectures to allow customization for fitting specific design parameters.

The documents listed below provide more detailed information. Part numbers are in parentheses.

Product Information Bulletins

PIB 23 *Digital Signal Processing in FLEX Devices (A-PIB-023-01)*

Application Notes

AN 73 *Implementing FIR Filters in FLEX Devices (A-AN-073-01)*

You can request these documents from:

- Altera Express fax-back service at (800) 5-ALTERA
- World-Wide Web at **http://www.altera.com**
- Your local Altera sales representative