KEY CONSIDERATIONS *when* DESIGNING MULTI GIGAHERTZ SOFTWARE DEFINED RADIOS

Benjamin Egg and Todd Thornton, Aethercomm Inc, begg@Aethercomm.com; fred harris, San Diego State University, fred.harris@sdsu.edu; Chris Dick, Xilinx Inc, chris.dick@Xilinx.com; Mike Rupar & Ivan Corretjer, US Naval Research Laboratory, rupar@nrl.navy.mil

Abstract

Higher data rate communication system demands and multi-function, multi-band, devices have driven the theoretical software defined radio (SDR) toward a maturing reality. Although a plethora of SDR prototypes have been and are being developed, one of the most common characteristics of each is high R&D costs. Common proprietary data confidentiality keeps many advanced architectures and discoveries unpublished, leading to 'reinventing of the wheel'. We have chosen to discuss three areas we consider key building blocks to a successful and timely SDR prototype. These three key points may be obvious to some and surprising to others, or as the reader delves deeper, possibly both—surprisingly obvious.

1. Introduction

Instantaneous availability of ultra-wide bandwidths and near teraflop digital signal processing (DSP) capacity combine to produce the versatility of a tier-1 SDR system. The ever-increasing bandwidths necessitate clean transceivers—amplifiers, data converters, and digital processing architectures. To satisfy the need of increasing bandwidths, data conversion rates (or sample rates), must increase; consequently, digital signal processing computational demands increase. Furthermore, the wider bandwidths often result in undesired spectral artifacts—distortions, co-channel interference, and analog filter non-linearities. These artifacts place added demands on the DSP engine to mitigate their impact. This paper focuses on satisfying the wide bandwidth requirements via multi giga-sample per second (GSPS) data converters. Also, we identify clock jitter as a significant issue and present DSP techniques for digitally reducing bandwidths or processing full-bandwidth signals.

Our discussion of ultra-wide bandwidth software defined radio architecture is broken into three topics, followed by a hardware example and the conclusion. First we discuss some spectral consideration relative to hardware digitization in section three, pointing out that clock jitter (noise) is one of the most significant spectral offenders in UWB systems, while remaining largely undetected as such. Secondly, in section four, we discuss utilization of DSP techniques to reduce bandwidths and channelize signals of interest to select specific spectra, while removing undesired signals or artifacts. This is an extremely versatile solution for a variety of very high speed processing filter functions. Next, in section five, we discuss a newer and sure to be important topic of Hyper-process systems where ultra high data rates require massively parallel DSP. This is essentially the inverse of resource reuse. In other words, resource sharing is Hypo-processing, and hyper-processing resource distribution. To clarify this subtle, but significant difference, hyper-processing is a dependable, efficient, yet 'brute-force' method of performing the common sub-function of any DSP operation—the Multiply and Accumulate (MAC). Whereas, the elegant filters of section 4 have substantial versatility and potential at the cost of more development time.

Section five briefly reviews our third generation hardware design—a multi giga-hertz SDR prototyping platform. Among several uses, it was developed and used to verify our DSP architectures and transceiver expectations. We share a few illustrations and review architectural highlights.

In conclusion we summarize each section and connect the hardware platform of section five with the theoretical or empirical topics of the previous section. Hopefully, the relation to actual hardware implementation will add substance to the mildly related topics of clock jitter, parallel processing and hyper processed filter structures.

2. Clock jitter, bandwidth, and SNR

The quality of a wideband SDR front end digitizer is heavily dependent on the internal and external characteristics of its data converters. Internally (inside the data converter), we have very little control once the data converter is selected; therefore, we focus on the external interface signals that significantly impact the quality of the analog front end performance.

The signal to noise ratio (SNR) of an analog-to-digital converter can be closely defined by the number of effective quantization bits specified. One limiting factor for UWB data converters is the track and hold bandwidth of the frontend buffer, which is the converters ability to track the changing input signal and instantly freeze the signal level when the clock strobe indicates, as depicted in figure 1(Maxim, Inc. [1]). Limiting a converter's effective number of bits (ENOB), the conversion speed must be matched to a satisfactory track and hold front-end circuit.

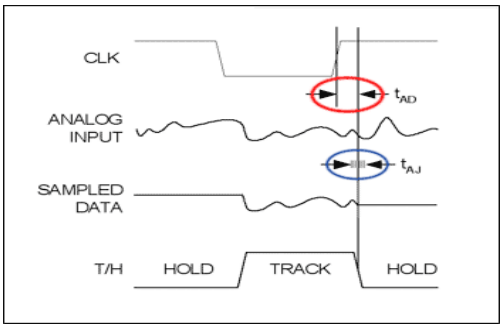

Figure 1. Track and hold amplifier.

Track and hold amplifiers are only as good as the strobe, or clock signal, provided to the data converter; thus, if a high performance track and hold amplifier is driven by a lower performance sample clock, the SNR floor is dominated by the noisy sample clock. Intuitively, if the analog input signal is lower frequency, i.e. it is changing very slowly with respect to time, the track and hold accuracy is less significant. On the other hand, as the signal level changes very rapidly, any variations from the ideal clock rising edge instant inserts noise into the quantization process. Figure 2 illustrates this by contrasting a slowly moving signal in figure 2a, and a faster moving (higher frequency) signal in figure 2b (Analog Devices [2]).

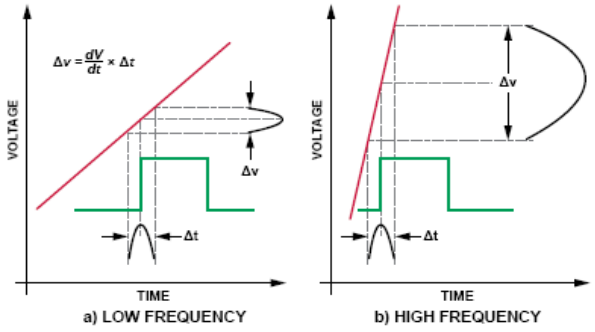

figure 2. Sample clock jitter effects for (a) slowly varying signal, and (b) rapidly varying signal.

However obvious this brief analysis may be, the impact of equation (1), which defines the SNR ratio due to clock jitter, at a given input frequency has derailed some very advanced SDR designs, while going all but undetected as the critical link causing the link failure. Noise inserted as a function of clock jitter is rarely identified via analysis unless one is familiar with equation 1.

$$SNR = -20\log_{10}\left(2\pi f \cdot t_{jitter}\right), \qquad (1)$$

Illustrating the significant impact of clock jitter, two authors assisted on a failing SDR design where months of debug and testing failed to identify a devastating noise source—their clock. The failing system was a high dynamic range design, using a 100 MSPS ADC at 14 bits, used to IF sub-sample a 700 MHz analog input signal. The clock source was a high quality 20 MHz reference clock fed into an FPGA and digitally multiplied to a higher frequency (inside the FGPA logic) to output a 400 MHZ sample clock. It is important to note that even high performance FPGA's can scarcely provide a clock source with less than $t_{jitter} < 100$ ps. Equation two reveals their true effective signal to noise ratio and effective number of bits (ENOB) for this design.

$$7.13dB = -20\log_{10}\left(2\pi \cdot 700 \cdot 10^{6} \cdot 100 \cdot 10^{-12}\right) \qquad (2)$$

As seen above, the effective dynamic range of this system, due to the 100 ps clock jitter is merely 7.13 dB. Using equation three, 7.13 dB is converted to an effective number of bits:

$$ENOB = \left[SNR - 1.76\right]/6.02 \qquad (3)$$

Thus, applying the 7.13 dB value from (2) the designers of said high performance system where actually dealing with an effective ADC resolution of just over one bit. A crying shame when the end user was depending on almost 14 bits of resolution since they had purchased a 14 bit converter!

The correct clocking structure for a high speed system provides the 'clean' clock to the data converters first, or only passes it through ultra-low clock jitter digital logic (illustrated in figure 3, grey box). The SDR platform presented in section 5 uses this clocking structure, illustrated below in figure 3.

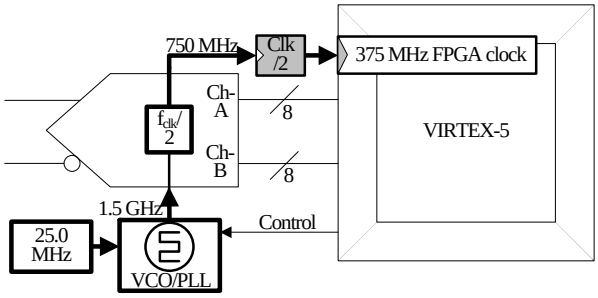

figure 3. Low jitter clock sourced directly from the VCO/PLL, then externally fed to the FPGA.

It is worth noting that all the equation presented above are not dependant on the sample rate, only the error in the samples' period to period variations—jitter, and the input signals' rate of change relative to time—frequency. Besides the illustration (3) above, several low jitter and inexpensive solutions have been published, with easy to implement architectures. The reader is encouraged to see [3] analog, [4] Maxim, [5] National, and reassured that those solutions

have been independently tried and work to specifications therein.

3. Bandwidth and Sample Rate—I

Reduction –We invoke signal processing architectures with embedded parallelism when the input sample rate delivered to the signal processor approaches the multiply-accumulate rate of the processor. The first response to parallelism is the use of numerous multipliers and adders as we relinquish the option of sequential time sharing a few resources in favor of numerous resources operating simultaneously.

As an example, let us assume a multiply accumulate rate of 400 MHz, or 2.5 nsec per multiply, applied to a 20 tap filter operating at an input sample rate of 1 MHz. A single multiplier can be applied sequentially to all 20 filter coefficients being able to perform the 20 multiplies in 50 nsec, or in 5% of the input sample clock. Now let us increase the input data sample rate from 1 MHz to 10 MHz and operate the same 20 tap filter. The same, single multiply, sequential structure can be used for this sample rate. The 50 nsec required to perform all 20 multiplies with a single multiplier is only 50% of the sample clock interval.

Continuing in our progression, we raise the sample rate from 10 MHz to 100 MHz. The time allowed to perform the filtering task is 10 nsec, an interval which permits only 4-multiply-accumulate cycles of a single multiplier. To perform the 20 multiplies of the filter, we require 5 multipliers operating simultaneously. Still, assuming the multiply-accumulator resources are available this is not a problem.

We hit our first wall now as we raise the input sample rate from 100 MHz to 1000 MHz. The problem here is that the input clock interval is now shorter than the multiply-accumulate time of the multiplier. We can invoke a brute force option to operate the 20-tap filter as three parallel time offset versions of the filter with time interleaved outputs supporting the high input and output sample rate. We consider this option in the next section! In this section we invoke finesse and convert the input data stream operating at the 1 GHz sample rate into multiple parallel data streams operating at reduced sample rates into a set of parallel time interleaved filters.

In one of many options, we partition the 1 GHz input time series into 4-parallel paths operating at 250 MHz via a maximally decimated perfect reconstruction filter bank as shown in figure 4. This process is identical to the channelization process used in the MP3 player. The 4-subchannels centered on the 4-cardinal direction on the unit circle are down sampled and aliased to baseband. In this process, the adjacent channel in the filter band edges of the channelizing filter are aliased into their respective channels during the down sampling. The spectral phasing of the perfect reconstruction filter banks cancel the aliased band edges. In this example the 4-channels have a bandwidth and sample rate equal to 250 MHz.

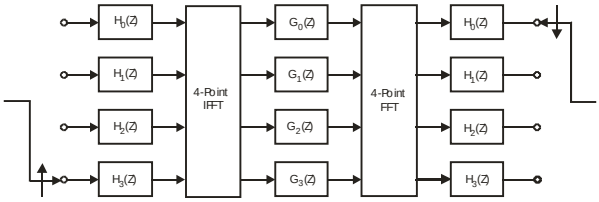

figure 4. Maximally Decimated, Perfect Reconstruction Filter Bank with desired Spectral Modifications performed between the banks at reduced sample rate.

In a second option, we partition the 1-GHz input time series into 8-parallel paths with 125 MHz bandwidth each sampled at 250 MHz via a non-maximally decimated perfect reconstruction filter bank as shown in figure 5. Here 8-subchannels, centered on the 8-octant directions on the unit circle are down sampled and aliased to baseband. By operating the 8-channels at a sample rate twice the channel bandwidth there is no aliasing of the transition band edges during the down sample operation. Operating the polyphase filters without aliasing the filter transition bands offers a wider range of filter options for the spectral modifications applied between the 8-path input and output channelizers [6].

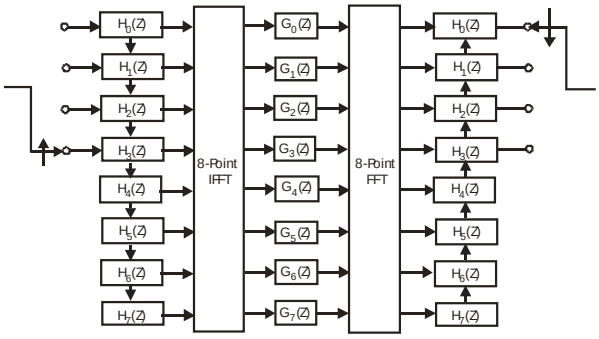

figure 5. Non-Maximally Decimated, Perfect Reconstruction Filter Bank with desired Spectral Modifications performed between the banks at reduced sample rate without Aliased Transition Bands.

4. Bandwidth and Sample Rate—II

Hyper-process filters –The basic operation of a digital signal processing system is the MAC—multiply and accumulate operation. One step lower is simply the multiple operation, where no accumulation function is required, as would be the case when normalizing a digital signal by multiplying by a constant, as in figure 6. For each input sample x(n), an output sample y(n) is

delivered on the subsequent clock cycle, which is a linearly scaled version of the input signal.

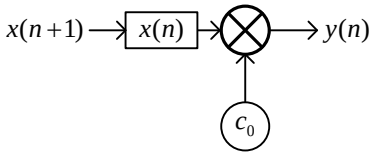

figure 6. Single Multiple by a constant coefficient c_0.

Taking the single product function a bit further, we have two forms of our 4-tap filter in figure 7. The first is the tapped delay line model that stores delayed samples in four registers. The second is the partial sum accumulator, which is the dual graph of the first model. This model stores delayed products and partial sums and has the advantage of separating the multiple adds by registers and thus avoids the ripple carry of the adder tree. Without consideration of sample delays, for simplicity and focus on the topic, for each sample input x(n), a corresponding result y(n) is delivered. In other words on each sample clock a result is delivered and a new sample is latched into a register.

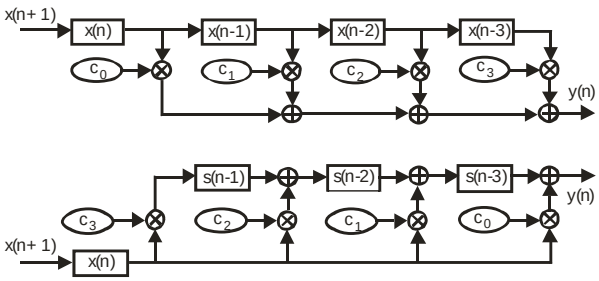

figure 7. Two forms of a Four tap filter requiring four multiplies and three additions per clock cycle.

When the systems processing capability exceeds the input data rate, it is common to optimize the design by sharing resources to perform several operation between input sample events. Figure 8 illustrates a MAC filter that performs the four multiply and accumulate operation illustrated in figure 7, but using only one multiplier and adder.

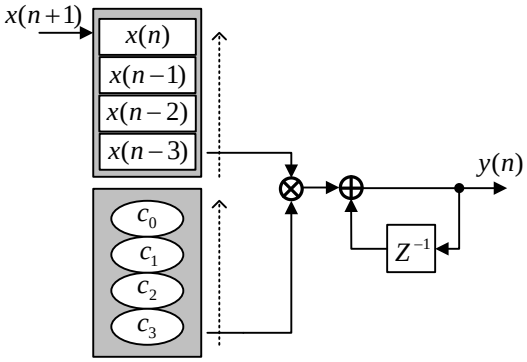

figure 8. Resources reuse forming a 4-tap FIR function.

The equations (4, 5) defining the operations illustrated in figures 7 and 8 are identical in result, yet the equations are left in their intuitive expansions for clarity and contrast.

$$y(n) = x(n)\cdot c_0 + x(n-1)\cdot c_1 + x(n-2)\cdot c_2 + x(n-3)\cdot c_3 \quad (4)$$

$$y(n) = \sum_{k=0}^{3} x(n-k)\cdot c_k \quad (5)$$

The equation in (5) and illustrated by figure 8 requires the internal processing rate to be 4 times faster than the incoming data; whereas, equation 4 (and figure 7) is processed at the system sample rate, equal to the digital processer's clock rate.

Now that we have refamiliarized ourselves with the MAC operation at system sample rate, equal to the processing rate, and resource reuse when the input sample rate is ¼ the processing clock rate, we are ready to investigate hyper-processed filters.

Hyper processed filters are inversely related to the filter illustrated in figure 8 and defined in equation (5). A hyper-processed system has a data input rate that exceeds high speed digital signal processing clock rates. For example if we would like to perform the simple 1-tap multiple of figure 6 when the data rate is 4 time faster than the processing rate, when have a 4-by hyper-process system (4x HPS), illustrated in figure 9.

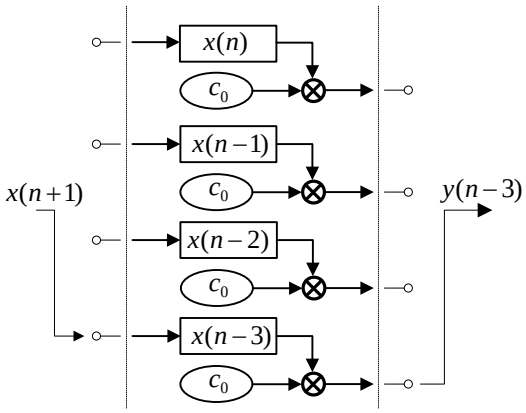

Figure 9. Single tap Hyper-Processed system, at 4xHPS.

As busy as figure 9 appears, relative to figure 6, they perform the exact same function, at different rates. The commutator delivering sample into the port of figure 9 operates at 4 times the core clock rate. Similarly the samples are re-multiplexed at 4 times the DSP clock rate, which is the DAC update rate.

A real world example of such a system can by illustrated by imagining a 2.0 GSPS ADC delivering samples to a cutting edge Field Programmable Gate Array (FPGA), capable of registering samples (via 2

ports) at 1 GSPS per port. The internal multipliers of the Xilinx flagship signal processing FPGA can operate comfortably at 500 MHZ [7]. Thus, if the functional goal was simply to normalize the incoming data via some constant (i.e. calculated in an offline AGC subsystem), the Virtex-5 built in serial to parallel IO blocks would reduce the data rate by 4, then block-deliver the four samples to the hyper-process subsystem. There, each sample x(n) to x(n-3) would be multiplied by the same constant c_0, then delivered as y(n) to y(n-3) for subsequent digital processing within the FPGA.

While this operation may seem trivial, it is worth considering that the fastest serial processed DSP available cannot perform that simple operation. The IO rate is simply too fast for any know bus available on a DSP. Furthermore, this single coefficient multiply would have consumed the most to all of the processing capability available, if the sample where somehow latched into the DSP. Consequently, this is a unique fit to the FPGA parallel processing fabric.

We take this thought process one step further by considering a 4 tap 4xHPS system, wherein the 4 coefficient FIR filter must be processed with data sampled at 4 times the FPGA fabric clock. Figure 10 illustrates this interesting operation for a 'simple' 4-tap FIR filter. Not shown for space considerations is the summation of the products in each row; however, the equations described in (6) clarify this necessary summation. Furthermore, the summation of a single row is identical to that illustrated in figure 4. Additionally, the input and output commutators are omitted, yet implied.

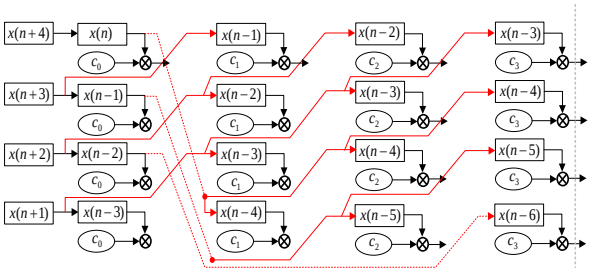

Figure 10. Four tap Hyper-process FIR filter, 4xHPS.

$$y(n) = x(n)c_0 + x(n-1)c_1 + x(n-2)c_2 + x(n-3)c_3$$
$$y(n-1) = x(n-1)c_0 + x(n-2)c_1 + x(n-3)c_2 + x(n-4)c_3$$
$$y(n-2) = x(n-2)c_0 + x(n-3)c_1 + x(n-4)c_2 + x(n-5)c_3$$
$$y(n-3) = x(n-3)c_0 + x(n-4)c_1 + x(n-5)c_2 + x(n-6)c_3$$

Equations (6. a, b, c, d, descending respectively)

…or compactly described, relative to each row of figure 10 and equation 6, a new equation (7) uses M to represent the row numbers of equation 6, rows 0 to 3.

$$y(n-M) = x(n-M)c_0 + x(n-M-1)c_1 + x(n-M-2)c_2 + x(n-M-3)c_3 \quad (7)$$

The point to consider here is that when bandwidth cannot be reduced, or when all the sampled data is to be processed with the requirement that 'no Hertz be lost', and the data rate is greater than the logic processing speed, hyper-process filters distribute the workload over multiple processing elements. As figure 7 indicates, processing resources increase inversely with the data-rate fan-out (or decrease). Thanks to Moore's law [7] regarding semiconductor prices and dimensions, this is a very natural and desirable tradeoff as semiconductor prices continue to drop and dimensions decrease!

5. Hardware realization

The authors present the BDR development system, a third generation SDR [2, 11], from which waveforms are both generated and captured to illustrate and provide test data for the processes and functions discussed herein [8]. Figure 12 is the BDR system with 1.5+ GSPS ADC [9] and 2.3+ GSPS DAC [10] and two virtex-5 SX50 FPGAs [6]. This system has excellent signal conditioning and figure 12 illustrates the SNR floor for a 1.4 GSPS input single. The clocking structure of figure 3 produced the captured data of the spectral plot figure 11, which indicates a 43 dB SNR and SFDR for an input RF tone of 870 MHz at a sample rate of 1.4 GSPS. One can calculate backward (using equation 1) while examining the FFT of the captured data (seen in figure 11), to verify that this system has less than 1.5ps RMS jitter

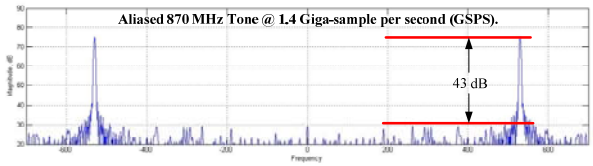

figure 11. Spectral plot of an 870 MHz signal IF sub-sampled (and aliased) via a 1.4 GSPS analog to digital converter.

The DSP processing architectures discussed in the Hyper-process filter section were used to generate the tone of figure 11, implemented in the BDR platform. A tone is a very basic operation in DSP; however, a single tone that digitally spans a 1 GHz bandwidth with only a lowpass analog filter (no mixers, etc.) requires a more involved effort. However, once this design is achieved, it can be repeated to produce multiple tones. The design and architecture for 8 independent tones, each covering 1 GHz bandwidth is accommodated in only 44% of (one of the two) the Virtex-5 SX50T FPGA (lower of the two FPGAs) in figure 12.

The hardware platform of figure 12 has the analog filters, variable gain, ADC and FPGA for the receive section on the top half of figure 12, and similar transmit functions on the bottom half. Furthermore, this system has independent Tx and Rx clocking, each at ultra low RMS jitter. Last, but not insignificant, 2 fiber optic and two Copper Ethernet ports are provided, all performing standard Ethernet protocol at 1 Gbps, for an aggregate 4+ Gbps user IO—all Ethernet compliant.

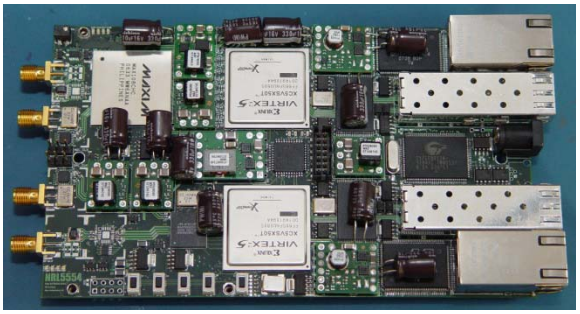

Figure 12. BDR prototypingsystem produced for Naval Research Laboroatories, with support from Xilinx, Aethercomm, and fred harris and associates.

6. Conclusion:

We discussed the idea that many SDR prototypes make leaps forward technologically, but have remained unpublished or obscure due to proprietary restrictions on data sharing. We discussed three points we have deemed important to a successful, ultra-high speed SDR prototype—clock jitter, versatile parallel processing, and hyper-process filtering architectures. While variable bandwidth filtering provides great versatility and ease of use once designed, they require considerable design expertise and effort for initial success of implementation. On the other hand, hyper-process filters are straight forward and very efficient when bandwidth must not be decimated or reduced. These sort of ultra-fast MAC filters are the essential building block to full-bandwidth DSP in the giga-sample range.

Finally, we discussed the Basic Digital Radio SDR platform design by the authors (in a variety of contributions). This SDR platform provides independent low jitter clocks, and very clean front-end analog signal conditioning. Furthermore, at ultra high speed, massively parallel processing capability is demanded. This platform sports 576 multipliers at over 500 MHz processing speed. That alone, ignoring the plethora of other logic resources, provides over a quarter trillion multiply and accumulate function per second. *This is a truly versatile SDR platform—with bandwidth, processing, and software reconfigurability.*

Bibliography:

[1] Maxim IC, www.maximic.com, http://pdfserv.maxim-ic.com/en/an/AN641.pdf

[2] Benjamin Egg, Chris Dick, fred harris, Implementation of a Scalable Software Defined Radio Prototyping Platform with Giga-Sample AFE , GSPx Conference October 2006, San Jose, CA.

[3] Analog Devices, www.analog.com, http://www.analog.com/static/imported-files/application_notes/5975649406491234250544717599125702454693706225592151118385418068775 5AN501_a.pdf .

[4]Maxim IC, www.maximic.com http://pdfserv.maxim-ic.com/en/an/AN800.pdf.

[5] National Semiconductor, www.national.com, http://webench.national.com/appinfo/interface/files/National_Clock_Design_Tool_Reference_Guide.pdf

[6] Chris Dick, fred harris, "FPGA Multi-rate Filters: A Case Study Using Virtex", ICSPAT'99, Orlando, FL, October 1999.

[7] Virtex-5 Data Sheets, Xilinx Inc, www.xilinx.com, http://www.xilinx.com/support/documentation/data_sheets/ds202.pdf

[8] Gordon E. Moore, "Cramming More Components onto Integrated Circuits", ftp://download.intel.com/museum/Moores_Law/Articles-Press_Releases/Gordon_Moore_1965_Article.pdf

[9] MAX108 data sheet, www.maximic.com, http://datasheets.maxim-ic.com/en/ds/MAX108.pdf

[10] MAX19692, www.maximic.com, http://www.maxim-ic.com/quick_view2.cfm/qv_pk/5172

[11] Benjamin Egg, fred harris, Chris Dick, "Ultra-Wideband 1.6 GHz Channelizer", Software Defined Radio Conference, Anaheim, CA, October, 2005.