Parallel Modem Architectures for High-Data-Rate Space Modems

Edgar Satorius*

ABSTRACT. — Existing software-defined radios (SDRs) for space are limited in data volume by several factors, including bandwidth, space-qualified analog-to-digital converter (ADC) technology, and processor throughput, e.g., the throughput of a space-qualified field-programmable gate array (FPGA). In an attempt to further improve the throughput of space-based SDRs and to fully exploit the newer and more capable space-qualified technology (ADCs, FPGAs), we are evaluating parallel transmitter/receiver architectures for space SDRs. These architectures would improve data volume for both deep-space and particularly proximity (e.g., relay) links. In this article, designs for FPGA implementation of a high-rate parallel modem are presented as well as both fixed- and floating-point simulated performance results based on a functional design that is suitable for FPGA implementation.

I. Introduction

Existing software-defined radios (SDRs) for space are limited in data volume by several factors, including bandwidth, space-qualified analog-to-digital converter (ADC) technology, and processor throughput, e.g., the throughput of a space-qualified field-programmable gate array (FPGA). For example, for ongoing relay operations of the Mars Reconnaissance Orbiter (MRO), the Electra modem [1] currently processes a maximum signal data rate of 4.096 Mbps, occupying a signal bandwidth of approximately 7 MHz in the ultra-high-frequency (UHF) band. This version of Electra employs an older Xilinx Virtex-1 FPGA. Other versions of Electra employ newer, more capable FPGAs, e.g., the Xilinx Virtex-2 FPGA used on both the Mars Atmosphere and Volatile Evolution (MAVEN) mission and the Trace Gas Orbiter (TGO)[2].

Further increase in data rates and thus data volume will be achieved with newer, more capable FPGAs and ADCs. Although the existing Electra radios use 16–20 Msps ADCs for UHF relay operations, space-qualified ADCs exist at much higher rates for use in S-, X- and Ka-band operations. Flight demonstrations of such technology are currently underway utilizing the Space Communications and Navigation (SCaN) Testbed, which operates on a truss

* Flight Communications Systems Section.

The research described in this publication was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

of the International Space Station (ISS). This testbed is ideally suited for demonstrations of high-rate SDRs, as discussed in [3].

In an attempt to further improve the throughput of space-based SDRs and to fully exploit the newer and more capable space-qualified technology (ADCs, FPGAs), we are evaluating parallel transmitter/receiver architectures for space SDRs. These architectures would improve data volume for both deep-space and particularly proximity (e.g., relay) links. Parallel receiver/transmitter architectures have been extensively developed in [4–8] and are already being used in ground receivers [9]. In general, existing digital space modems are based on a serial implementation, e.g., the Electra transceiver [1]. Similarly, the Small Deep Space Transponder (SDST) [10] is based on a serial architecture implemented with nonprogrammable, application-specific integrated circuits (ASICs).

This article summarizes a study of parallel architectures for space applications. An overview of the parallel receiver architectures is presented in Section II and a functional model of the most promising architecture for space applications is presented in Section III. Simulated performance results are presented in Section IV and conclusions in Section V.

II. Parallel Receiver Architectures

There are two general classes of parallel receiver architectures based on (i) polyphase filter bank theory and (ii) frequency domain representations. These architectures are particularly suitable for highly parallel FPGA implementation. Furthermore, they have previously been considered for ground-based receiver applications [4–7], and thus appear to be good candidates for space-based SDR implementation.

The parallel receiver (PRX) architecture was originally presented in [4] and is based on a uniform discrete Fourier transform (DFT) polyphase filter bank structure. A block diagram of the PRX architecture is provided in Figure 1. The PRX comprises analysis, matched filter, and synthesis sections. The basic idea is to first break the input into subbands (L = 2M subbands as depicted in Figure 1) and downsample in the analysis section, followed by matched filtering and then signal reconstruction (upsampling and combining) in the synthesis section. In this way, most of the computationally intensive operations can be done at a lower sampling rate (integer factor of M lower) than is used to sample the high-bandwidth input signal.

The second class of alternate parallel receiver (APRX) architectures is implemented in the frequency domain and is based on the "Overlap and Save" method for performing high-speed filtering via the DFT, which is computed with the fast Fourier transform (FFT) algorithm. By parallelizing into 2M paths, but decimating only by M, each DFT operates on M points from the previous cycle along with M new points. This provides the overlap required for calculating all of the linear convolutions. Following the 2M-point inverse DFT (IDFT), the middle M parallel outputs (which are unaliased) are used for detection, tracking, etc.

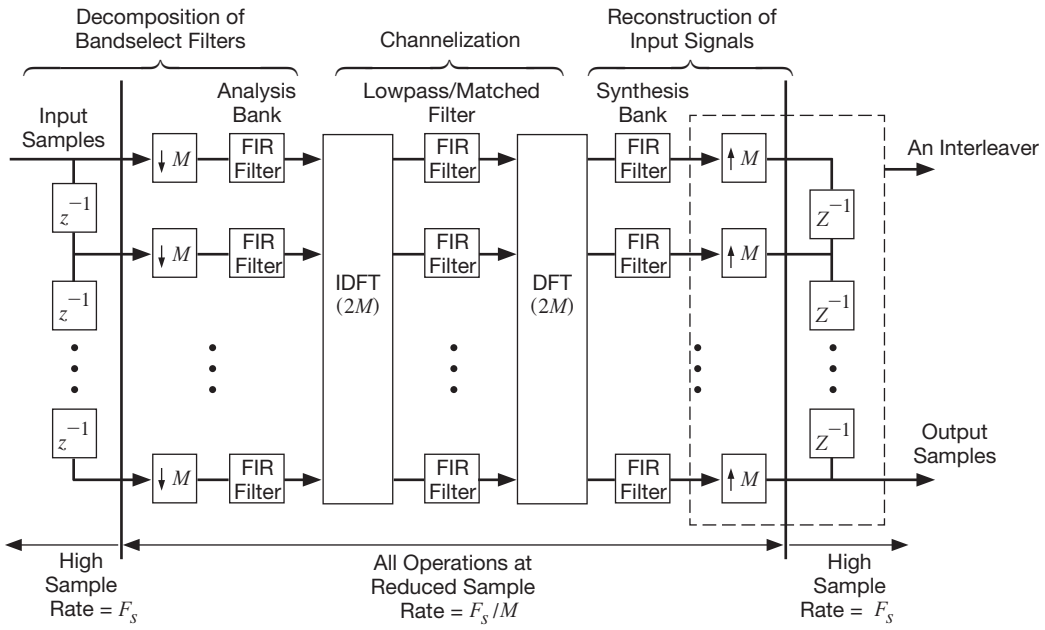

Figure 1. Polyphase filter bank parallel receiver (PRX) architecture.

This class of parallel receiver architectures was originally developed in [5–8]. A block diagram of the APRX architecture is provided in Figure 2 corresponding to L = 2M frequency channels. The multipliers H_k correspond to the L-point DFT of the lowpass/matched filter coefficients. Assuming N_s samples per symbol and rectangular pulse shaping, H_k is given by

$$H_k = e^{-j\omega_k(N_s-1)/2} \cdot \frac{\sin(\omega_k N_s/2)}{\sin(\omega_k/2)}, \omega_k \equiv 2\pi k, /L, 0 \le k \le L-1.$$

A lowpass filter can be incorporated into the H_k simply by zeroing out the middle M components in the frequency domain that correspond to the high-frequency terms, i.e., $H_k = 0, M/2 \le k \le 3M/2 - 1$.

Note that the low-rate filtering operations required in the PRX (Figure 1) are replaced with simpler low-rate multipliers (H_k) in the APRX (Figure 2). As such, the APRX is simpler to implement and thus has been chosen for further development for a space modem application. Nevertheless, it should be pointed out that the PRX does have the flexibility to accommodate multiple input carriers through the subband decomposition. To this end, a simplified PRX analysis section is being included as a digital front-end for a multi-user space modem that is currently under development.

In Section III, a functional MATLAB model for the APRX architecture is presented, including carrier and symbol tracking loops as well as a digital automatic gain control (AGC) loop. All of the loop implementations presented in Section III are currently being used in Electra on MRO, Mars Science Laboratory (MSL), MAVEN, and TGO and are thus well understood and considered low risk from an FPGA development standpoint.

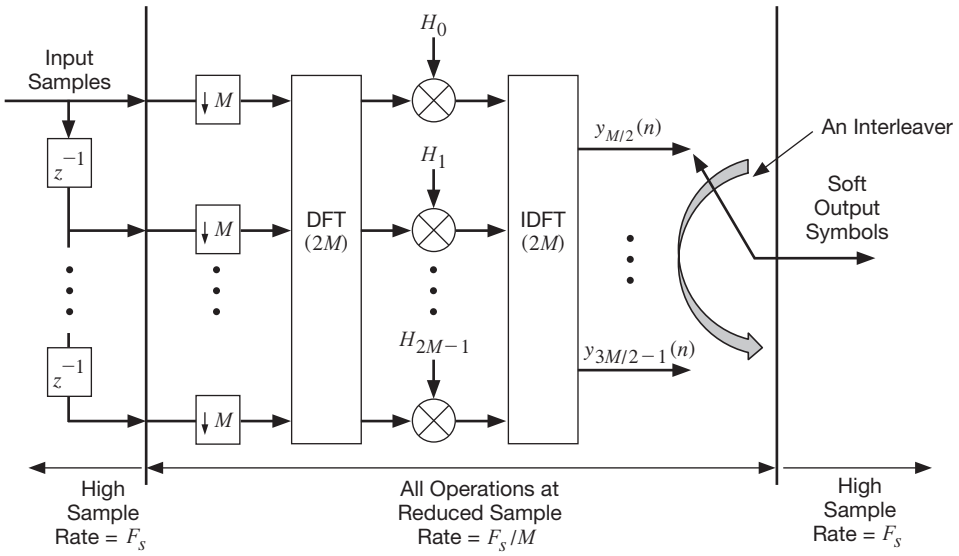

Figure 2. Frequency domain alternate parallel receiver (APRX) architecture.

Section III. APRX Functional Model of the APRX Modem

A functional MATLAB model of the APRX for performance evaluation has been developed and is shown in Figure 3. The vector $\underline{V}$ in Figure 3 is a delay vector that is used to adjust the symbol timing based on the output from the symbol tracking loop (STL), δ, (which is generally not an integer).

Note that $\underline{V}$ is symmetrized to have complex-conjugate symmetry about the Nyquist point, $V_M = e^{2\pi j\delta/2}$, which is usually zeroed out by the lowpass filtering operation noted above. Thus, $\underline{V}$ is not the normal delay vector associated with a circular shift by P integer samples, i.e., $[1e^{2\pi jP/(2M)} \cdots e^{2\pi jP/(2M-1)(2M)}]^T$ (^{T}denotes transpose), except in the special case when δ is an integer, $\delta = P$. Thus, when operating with fractional delays it is crucial to use the symmetrized delay vector as defined in Figure 3.

In implementing the different loops, we have tested various configurations but have found that a single output pin from the 2M IDFT outputs can successfully drive the AGC and carrier tracking loops. This results in a significant reduction in computational complexity. Currently, suppressed-carrier binary phase-shift keying (BPSK) modulation is being tested and thus the carrier tracking loop is implemented as a second-order Costas loop [1]. (Modifications for residual carrier modulation have also been developed.) The AGC loop implementation is shown in Figure 4.

For the APRX, we have chosen an early–late gate STL (as currently implemented in Electra on MAVEN and TGO). The STL architecture is shown in Figures 3 and 5. This loop is driven by a pair of outputs from the IDFT separated by $N_s/2$ pins. For purposes of illustration in Figures 3 and 5, we have chosen $N_s = 8$ and thus the pair of outputs that are fed into the STL are separated by $N_s/2 = 4$ output pins, i.e., y_{M-3} and y_{M+1}.

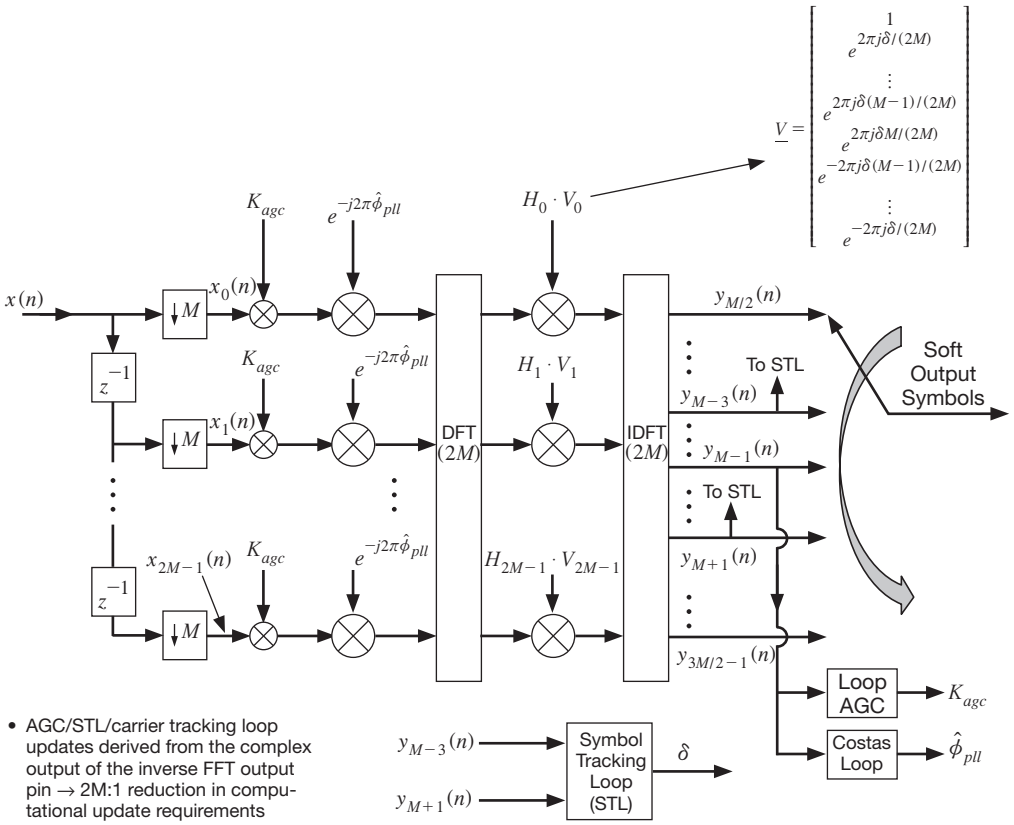

Figure 3. Functional model of the APRX.

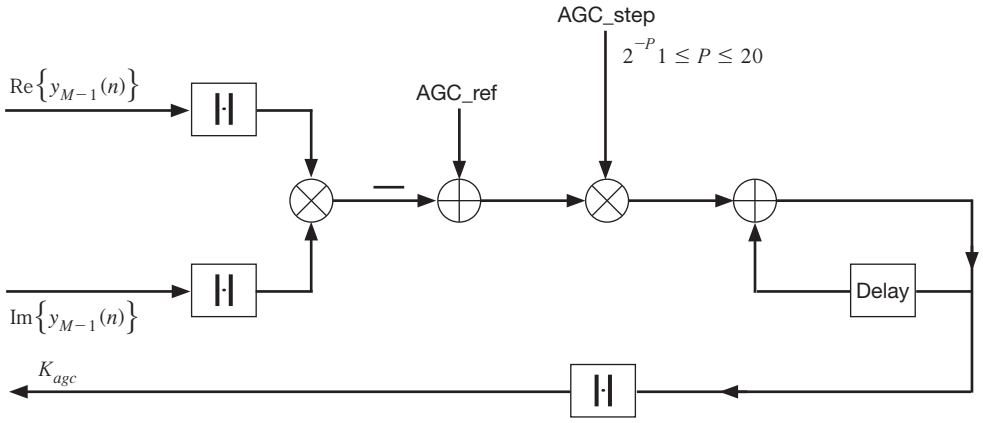

Figure 4. APRX AGC loop.

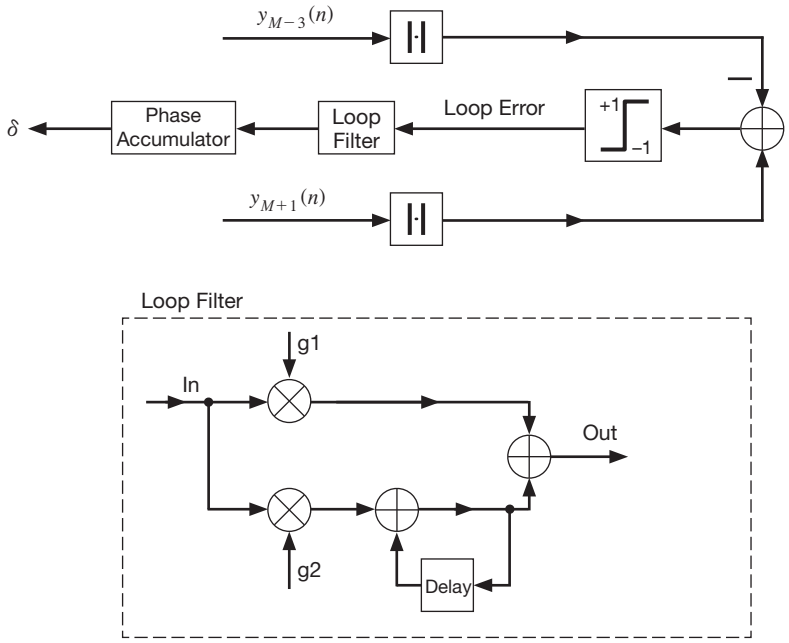

Figure 5. APRX STL architecture.

The STL loop filter coefficients, g1 and g2, include the normalization by the sample frequency:

$$g1 = (8/3) \cdot B_{Ld}/f_s$$
$$g2 = (32/9) \cdot B_{Ld}^2/f_s^2,$$

where B_{Ld} denotes the loop bandwidth (Hz). The accumulator in Figure 5 wraps the output to modulo N_s samples. Thus, δ is between 0 and 127 samples, although typically $0 \leq \delta \leq 7$. The conversion from δ to the phases $e^{\pm 2\pi j \delta k/(2M)}$ (to generate $\underline{V}$ in Figure 3), will be implemented via sin/cosine look-up tables.

Section IV. Simulated Performance Analysis

We have conducted numerous floating- and fixed-point MATLAB simulation experiments in evaluating the APRX model.[1] For the simulation results presented here, L = 32 frequency channels are used with a decimation factor of M = 16 for each channel.[2] Sample bit-error rate (BER) curves are shown in Figure 6 corresponding to 8 samples/bit where a static delay of 3 samples (at the input sample rate, i.e., $x(n)$ in Figure 3) and a static phase offset of 65 deg have been introduced. Both floating- and fixed-point results are shown.

For these simulations, the normalized Costas loop bandwidth, B_L/f_s, for the floating-point model is set at 5.3e–05 and for the fixed-point model B_L/f_s is set at 5.3e–04. The larger value for the fixed-point model is based on the smaller input signal levels used in the fixed-point

[1] Fixed-point word lengths for the APRX are based on computer optimization as well as previous Electra sizing.

[2] In subsequent development, we have used L=16/M=8 to enable a fully parallel, radix-4 FFT implementation.

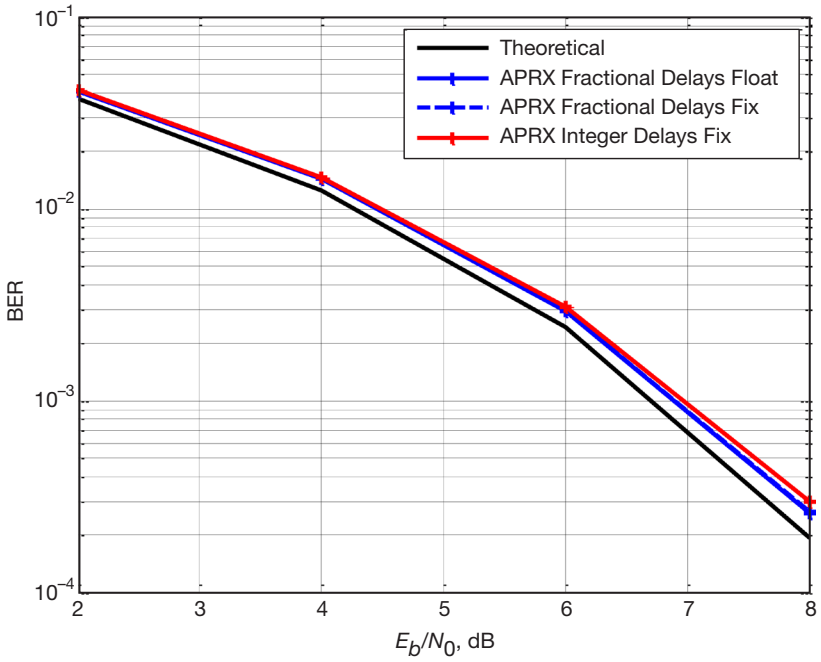

Figure 6. APRX BER simulation results with a static delay offset of 3 samples.

model. In addition, the normalized STL loop bandwidth, B_{Ld}/f_s, is set at approximately 4.5e–04 for both the floating- and fixed-point models, and the AGC stepsize used with the fixed-point model (AGC_step in Figure 4) is 2^6 with a reference level (AGC_ref in Figure 4) of 0.15.

In addition in Figure 6, a curve is shown wherein the delay (δ in Figure 5) is constrained to be an integer. As is seen, there is very little difference in BER performance with integer or fractional delays (about 0.25 dB loss at 8 dB E_bN_0 for either implementation). However, if a non-integer delay is introduced in the input (synthesized by first upsampling, delaying, and then downsampling), there is a marked difference in BER performance. This is clearly seen in Figure 7 where an input delay of 2.5 samples is used (with the same loop parameters as before). In this case, the APRX implemented with fractional delays performs as before (comparing Figures 6 and 7).

However, when δ is constrained to be an integer, there is an additional 0.3 dB loss at 8 dB E_bN_0. This additional loss is due to the dithering of the integer delay between 2 and 3 sample delays to achieve on average the fractional delay of 2.5 samples. It can arise whenever the STL is constrained to have an integral delay, which is the case with the STLs used on the various versions of Electra (MRO, MSL, MAVEN, etc.). Therefore, the use of fractional delays in the APRX yields an added performance advantage for this architecture.

Finally, we show typical loop responses in Figure 8 for the APRX at 8 dB E_bN_0 and with a fractional delay of 1.8 samples introduced in the input as well as a 65-deg static phase offset. For this simulation, an input sample rate of 150 MHz was used, corresponding to the space-qualified Universal Space Transponder (UST) wideband ADC, again with a data rate

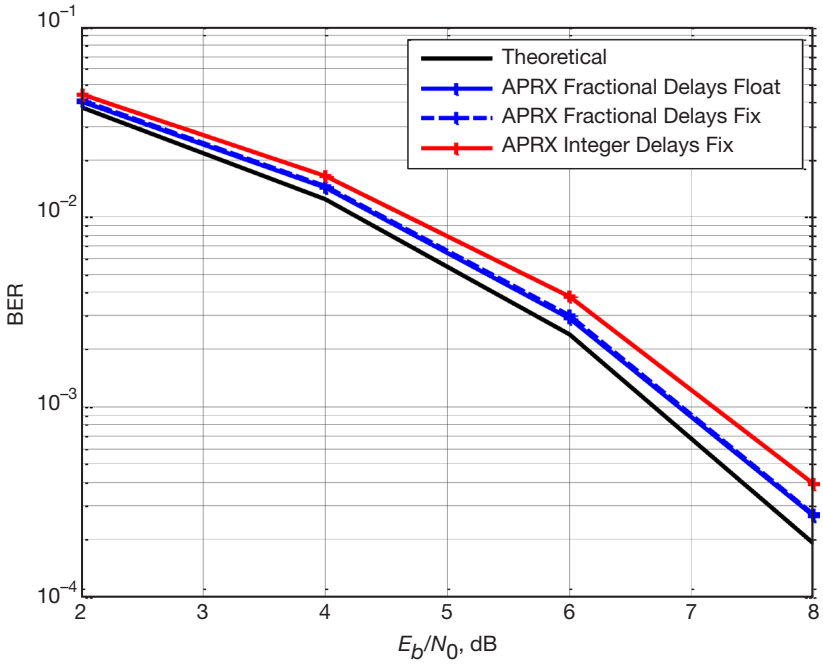

Figure 7. APRX BER simulation results with a static delay offset of 2.5 samples.

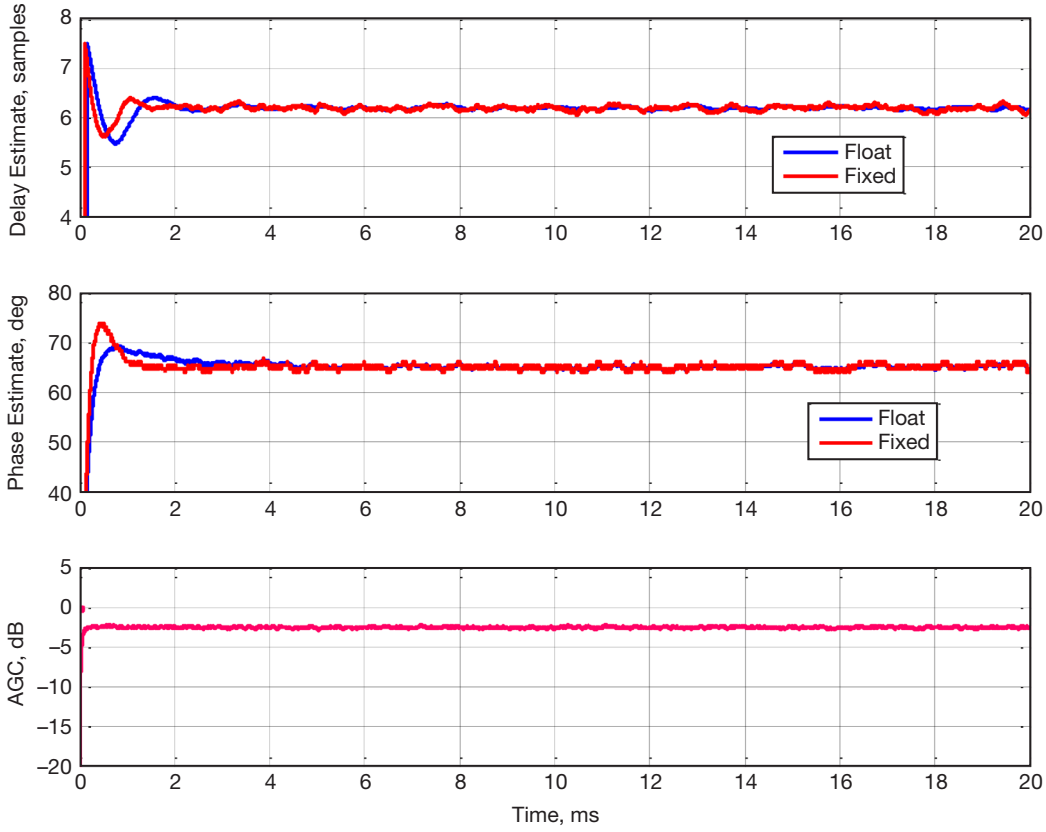

Figure 8. APRX floating- and fixed-point loop responses at E_b/N_0 = 8 dB with a static delay offset of 1.8 samples and a static phase offset of 65 deg.

of 8 samples per symbol, i.e., 18.75 Mbps. Loop parameters for this simulation are 500 Hz (float)/5 kHz (fixed) — Costas loop bandwidth; 4.25 kHz (float/fix) — STL loop bandwidth and the same parameters as stated above for the AGC loop (with reference to Figure 4, AGC_step = 2^6 and AGC_ref = 0.15).

As is seen, all of the loops settle within a couple of milliseconds or less, which is consistent with the loop bandwidth parameters. It is also noted that the fixed- and floating-point models produce about the same Costas and STL loop responses. Similarly, the BER curves for the fixed- and floating-point models (Figures 6 and 7) are about the same.

Section V. Conclusions

Based on the results of this study, we have found that the APRX architecture is best suited for a space modem application. (Similar conclusions were reached in [7] relating to the hardware development of a parallel receiver.) Though the uniform DFT polyphase filter bank structure (Figure 1) has some very nice features relative to a multi-user environment, the frequency domain architecture presents the least risk in terms of a space modem firmware implementation. Another added bonus with the frequency domain implementation is its capability to synthesize fractional symbol delays. As seen in Figures 6 and 7, this reduces modem losses resulting from using integer delays in conventional STL implementations.

Given the results from this study, the next step is to conduct prototype demonstrations using either a SCaN Testbed FPGA development board (ADC clocked at 49.244 MHz) or a UST FPGA development board (ADC clocked at 150 MHz). At 4 ADC clock samples per symbol, demonstrations with either board would show a significant increase over data rates available with current space modems (limited to 4.096 Mbps with the existing versions of Electra, though this is more a function of the ADC clocking rate, which is nominally 16–20 Msps).

As the first step toward the development of an FPGA APRX modem for space, we have carried out a preliminary utilization analysis of the APRX for FPGA implementation assuming L = 2M = 32 frequency channels. The results of this analysis are presented in Figure 9 for two different FPGAs: (1) the Xilinx Virtex-2 FPGA used on MAVEN and TGO, and (2) the Xilinx Virtex-4 used on UST. As is seen, we have partitioned the required processing into (a) a digital downconverter required to convert the real output from the ADC into a complex baseband data stream [$x(n)$ in Figure 3]; (b) the tracking loops (STL, Costas/carrier tracking loop, and AGC); (c) the FFTs (forward and inverse); (d) the complex multiplies, $K_{agc} \cdot e^{-2\pi j \hat{\phi}_{pll}}$, prior to the FFT (labeled Pre-FFT mults in Figure 9), and (e) the complex multiplies, $H_k \cdot V_k$, between the two FFTs (labeled Post-FFT mults in Figure 9 — see Figure 3).

Clearly, the Post-FFT mults require the most slices, though they can be time-shared since they operate at only one-sixteenth the input sampling rate. In fact, the utilization estimates shown in Figure 9 assume that by time-sharing we only need to implement 8 complex multipliers on the FPGA. Similarly, we assume that the Pre-FFT mults can be imple-

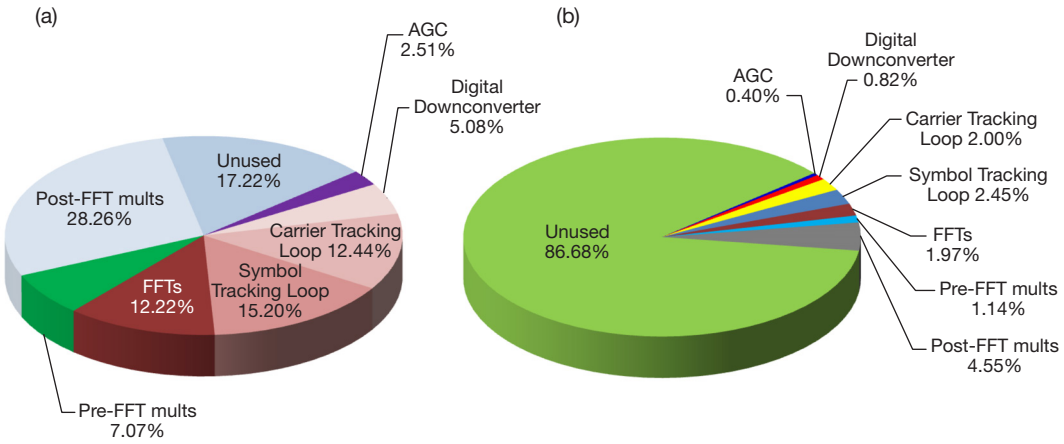

Figure 9. APRX utilization with (a) Xilinx Virtex-2 and (b) Xilinx Virtex-4 in terms of slices.

mented with only 2 complex multipliers by time-sharing. Alternatively, we can use the built-in FPGA hardware multipliers to implement the Pre- and Post-FFT mults, which would further reduce the utilization requirements for these multiplies. In any case, the APRX architecture fits nicely in either FPGA with room to spare (especially with the Virtex-4). The unused space can be allocated to decoders and modulators.

From our studies to date, we find that space modems based on parallel receiver and potentially parallel modulator architectures are feasible with the potential of significantly enhanced data throughputs. Extensions of the APRX to accommodate higher-order modulations as well as residual carrier formats have been carried out. Current effort is directed toward the FPGA implementation and testing of the APRX on the SCaN Testbed.

References

[1] E. Satorius, T. Jedrey, D. Bell, A. Devereaux, T. Ely, E. Grigorian, I. Kuperman, and A. Lee, "The Electra Radio," Chapter 2 in *Autonomous Software-Defined Radio Receivers for Deep Space Applications*, J. Hamkins, M. Simon, J. Yuen, eds., Wiley-Interscience, October 13, 2006. Available electronically: Deep-Space Communications and Navigation Series,Jet Propulsion Laboratory, Pasadena, California, 2006. http://descanso.jpl.nasa.gov/Monograph/series9/Descanso9_02.pdf

[2] C. Edwards, B. Arnold, D. Bell, K. Bruvold, R. Gladden, P. Ilott, and C. Lee, "Relay Support for the Mars Science Laboratory and the Coming Decade of Mars Relay Network Evolution," *Proceedings of the 2012 IEEE Aerospace Conference*, pp. 1–11, Big Sky, Montana, March 3–10, 2012.

[3] S. Johnson, R. Reinhart, and T. Kacpura, "CoNNeCT's Approach for the Development of Three Software-Defined Radios for Space Application," *Proceedings of the 2012 IEEE Aerospace Conference*, pp. 1–13, Big Sky, Montana, March 3–10, 2012.

[4] R. Sadr, P. P. Vaidyanathan, D. Raphaeli, and S. Hinedi, *Parallel Digital Modem Using Multirate Digital Filter Banks*, JPL Publication 94-20, Jet Propulsion Laboratory, Pasadena, California, August 1994.

[5] A. Gray, S. Hoy, and P. Ghuman, "Parallel VLSI Equalizer Architectures for Multi-Gbps Satellite Communications," *Proceedings of Global Telecommunications Conference, (GLOBECOM)*, vol. 1, pp. 315–319, San Antonio, Texas , November 25–29, 2001.

[6] A. Gray, "Very Large Scale Integration Architectures for Nyquist-Rate Digital Communication Receivers," PhD Dissertation, University of Southern California, Los Angeles, California, May 2000.

[7] M. Srinivasan, C.-C. Chen, G. Grebowsky, and A. Gray, "An All-Digital, High Data-Rate Parallel Receiver," *The Telecommunications and Data Acquisition Progress Report*, vol. 42-131, Jet Propulsion Laboratory, Pasadena, California, pp. 1–16, November 15, 1997.
http://ipnpr.jpl.nasa.gov/progress_report/42-131/131K.pdf

[8] G. Grebowsky et al., U.S. patent 6,177,835 B1, January 23, 2001.

[9] K. Andrews, J. Gin, N. Lay, K. Quirk, and M. Srinivasan, "Real-Time Wideband Telemetry Receiver Architecture and Performance," *The Interplanetary Network Progress Report*, vol. 42-166, Jet Propulsion Laboratory, Pasadena, California, pp. 1–23, August 15, 2006.
http://ipnpr.jpl.nasa.gov/progress_report/42-166/166H.pdf

[10] C.-C. Chen, S. Shambayati, A. Makovsky, F. Taylor, M. Herman, S. Zingales, C. Nuckolls, and K. Siemsen, *Small Deep Space Transponder (SDST) DS1 Technology Validation Report*, JPL Publication 00-10, Jet Propulsion Laboratory, Pasadena, California, October 2000.
http://pds-smallbodies.astro.umd.edu/holdings/ds1-c-micas-3-rdr-visccd-borrelly-v1.0/document/doc_Apr04/int_reports/SDST_Integrated_Report.pdf