A Non-Uniform Sampler for Wideband Spectrally-Sparse Environments

Michael Wakin, Stephen Becker, Eric Nakamura, Michael Grant, Emilio Sovero,
Daniel Ching, Juhwan Yoo, Justin Romberg, Azita Emami-Neyestanak, Emmanuel Candès

***Abstract*—We present the first custom integrated circuit implementation of the compressed sensing based non-uniform sampler (NUS). By sampling signals non-uniformly, the average sample rate can be more than a magnitude lower than the Nyquist rate, provided that these signals have a relatively low information content as measured by the sparsity of their spectrum. The hardware design combines a wideband Indium-Phosphide (InP) heterojunction bipolar transistor (HBT) sample-and-hold with a commercial off-the-shelf (COTS) analog-to-digital converter (ADC) to digitize an 800 MHz to 2 GHz band (having 100 MHz of non-contiguous spectral content) at an average sample rate of 236 Msps. Signal reconstruction is performed via a non-linear compressed sensing algorithm, and an efficient GPU implementation is discussed. Measured bit-error-rate (BER) data for a GSM channel is presented, and comparisons to a conventional wideband 4.4 Gsps ADC are made.**

***Index Terms*—Non-uniform sampler, compressed sensing, wideband ADC, indium-phosphide HBT, sample-and-hold.**

I. Introduction

In such far-ranging fields as radio, telephony, radar, image, audio and seismic acquisition, most analysis techniques follow the same pattern: (1) digitize an analog signal, (2) perform DSP, and, optionally, (3) convert back to the analog domain. The common piece of hardware in this chain is the analog-to-digital converter (ADC). The current trend in systems is wider bandwidths and larger dynamic ranges, and designing a single ADC to meet both of these requirements simultaneously is difficult. To get around this, systems typically use time-interleaved ADCs or channelize the band and digitize each channel separately. However, these approaches do nothing to reduce the output data rate and can require prohibitively high power.

M. Wakin is with the Department of Electrical Engineering and Computer Science, Colorado School of Mines, Golden, CO, e-mail: mwakin@mines.edu
D. Ching, E. Nakamura, and E. Sovero are with the Northrop Grumman Corporation, Redondo Beach, CA, e-mail: eric.nakamura@ngc.com
A. Emami-Neyestanak and J. Yoo are with the Electrical Engineering Option at the California Institute of Technology, Pasadena, CA
E. Candès is with the Departments of Mathematics and Statistics at Stanford University, Stanford, CA
J. Romberg is with the School of Electrical and Computer Engineering at the Georgia Institute of Technology, Atlanta, GA
S. Becker and M. Grant are with the Applied and Computational Mathematics Option at the California Institute of Technology, Pasadena, CA; S. Becker is also with the laboratoire Jacques-Louis Lions at Paris 6 University, Paris, France

This work was supported by the DARPA/MTO Analog-to-Information Receiver Development Program under AFRL contract FA8650-08-C-7853.

A. A new paradigm

The *effective instantaneous bandwidth* (EIBW) of an ADC is the total bandwidth of the spectrum that can be unambiguously recovered. Although it is generally desirable to design receivers with high EIBW—say, for applications involving cognitive radio or communications intelligence—it may also often be the case that, at any given time instant, much of the spectrum within this bandwidth is unoccupied. One can define the *information bandwidth* of such signals to be the actual amount of occupied spectrum. In this paper, we present a receiver design intended for signals with high EIBW but low information bandwidth. We do so by adopting concepts from the field of compressed sensing (CS).

The theory of CS [7], [14] suggests that randomized low-rate sampling may provide an efficient alternative to high-rate uniform sampling. For a survey of the modern CS literature, the reader is referred to [8].

To put CS on a concrete footing, we give an explicit (but for the moment, discrete-time) example. Let x be a length-N signal, and suppose the Discrete Fourier Transform (DFT) of x, denoted X, is K-sparse, meaning that it has only $K \ll N$ nonzero entries.

Now, collect only a *subset* Ω of all the entries of x. Suppose the sample locations Ω are chosen uniformly at random, and let M be the size of Ω. Because $M < N$,[1] it is generally not possible to recover x using a linear method. The remarkable fact of CS is that if M is merely proportional to $K \log N$, then with very high probability (which can be made precise [7]), it is possible to *exactly* recovery x by solving the linear program

$$\min_{x'} \|X'\|_1 \quad \text{subject to} \quad \forall k \in \Omega, x'(k) = x(k).$$

Here, $\|X\|_1 = \sum_{k=1}^{N} |X(k)|$. There are related approaches, such as greedy methods, that offer similar guarantees; see [25] for a survey.

This result itself has limited application to signal processing since (1) it is unlikely that a digital signal has an exactly sparse DFT, and (2) the model does not account for noise. Fortunately, there are robust versions of the above statement, which allow signals to be only *approximately sparse*, and which allow noise [6]. In this case, exact recovery is not possible, but the recovered signal agrees with the true signal up to the noise level.

This finite dimensional model does not fully cover the continuous case since an analog signal, unless it is bandlimited

[1]In our implementation, M is approximately $19\times$ smaller than N.

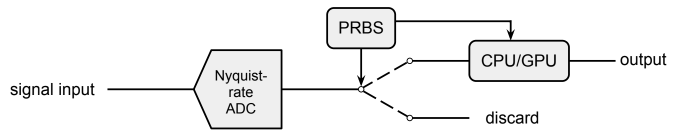


Fig. 1: Conceptually, the NUS takes Nyquist-rate samples of the input signal and then randomly discards most of the samples. The implemented version uses a clock rate of 4.4 GHz and effectively keeps only one of every 19 samples (on average) for a mean output sample rate of 236 MHz.

and periodic, must be treated in an infinite dimensional setting. The infinite dimensional setting may be attacked directly, and there is recent theory [2], [13] that connects the finite and infinite dimensional problems. On a practical side, some non-periodic (also known as multi-coset) sampling results for multi-band signals [17] have recently been extended to multi-band signals when the band locations are unknown [18]. The approach in [18], [19] is a hybrid finite-infinite approach that solves a finite dimensional problem to determine the band locations and then processes the samples directly in analog.

The approach taken in this paper deals with the infinite dimensional problem indirectly. Through extensive numerical simulation, and by using standard signal processing techniques such as windowing, it is shown that the error incurred by using a large but finite number of samples is insignificant compared to circuit non-idealities. Numerical simulations are required regardless since CS theories rely on possibly conservative constants and also on signal-dependent parameters, such as the sparsity of the signal.

B. Approach

1) Overview: The CS example suggests that signals with high EIBW but low information bandwidth can be efficiently captured using non-uniform samples. Our implementation is such a non-uniform sampling (NUS) approach, which we describe here and treat in more detail in Section II. The key ideas are to leverage the high resolution that can be achieved with lower-rate ADCs and to exploit the fact that the electromagnetic spectrum in our bandwidth is typically not full.

There are two sets of signal restrictions for the NUS. The first is a familiar restriction requiring the EIBW to be less than half the equivalent Nyquist sampling rate. The second restriction is an algorithmic one: CS theory dictates that the input signal should have spectral sparsity in order to achieve accurate reconstruction. Roughly speaking, in our implementation the information bandwidth may be up to 10% of the EIBW.

The idea behind the NUS is explained in Figure 1. For our setup, the maximum EIBW is 2.2 GHz because of an underlying "Nyquist rate" clock with a frequency of $f_s = 4.4$ GHz. For the sake of explanation, assume there is a Nyquist rate ADC which samples the input signal—the actual implementation does not use a Nyquist rate ADC, since the point of the NUS is to avoid a high-rate ADC. A pseudo-random bit sequence (PRBS), generated off-chip, controls which of these samples are collected and which samples are ignored. Of every 8192 Nyquist-rate samples, only 440 are collected. Note that our method of "on grid" non-uniform sampling is very different from allowing arbitrarily spaced samples that are not integer multiples of the underlying Nyquist rate, since the latter approach would be nearly impossible to calibrate.

The actual implementation, shown in Figure 2, replaces the theoretical sub-sampled Nyquist-rate ADC with a non-uniformly clocked sample-and-hold (S/H). The sample times of the S/H are controlled by the PRBS sequence, and the same sequence controls a single low-rate ADC which performs the final quantization step. The custom S/H is necessary because the ADC is not designed for 2.2 GHz bandwidth signals.

Reconstruction—that is, interpolation of the omitted Nyquist-rate samples—is performed on a desktop personal computer using a block algorithm described in Section III. In addition to delay due to processing, there is a small latency while the whole block is acquired. Each block is composed of $N = 65536$ Nyquist-rate samples, which corresponds to a time interval of length $T = 14.9\ \mu s$. When calculating the *occupied* bandwidth, the entire time interval must be considered, so the algorithm-based restriction is that at most 10% of the length-N DFT of the sample block should be non-negligible. Extremely short duration signals are automatically excluded from the signal model since they have a broad frequency spectrum which ruins the sparsity. Our block-by-block reconstruction strategy allows for the capture of frequency hopping signals, although for the specific tests reported in Section IV we require some stationarity of the spectrum (over, say, a period of 2 ms) to allow the spectral support to be identified.

2) Hardware specifications: The NUS IC is designed in an InP HBT technology. A TI ADS5474 14-bit 400 Msps ADC (10.9 ENOB at 230 MHz) is used to digitize the samples, and the data is transferred to a computer for processing.

The specifications of the NUS are described in Table I. Power consumption is relatively high because the front-end is designed for wide bandwidth and high dynamic range. However, lower system power is possible for applications that require data transmission, since the NUS produces $19\times$ fewer samples.

Comparing the NUS to a Nyquist ADC is difficult since recovery error depends on spectral sparsity. To measure the resolution of the samples, the NUS can be operated in a uniform sampling mode. This measurement shows that the NUS has 8.8 ENOB performance across the frequency band from 800 MHz to 2 GHz. It is important to note that this measurement does not assess the reconstruction accuracy and does not directly relate to ADC ENOB. Instead of a direct comparison, the results in Section IV show promising GSM bit error rate (BER) performance.

C. Related work in compressed sensing

1) Non-uniform samplers: To the best of our knowledge, there have been no IC implementations of the NUS that fully reconstruct the signal. The interesting work [3] on optical sub-Nyquist sampling is similar in spirit, but it works in the optical domain with COTS components and uses a least-squares fit to reconstruct pure tones rather than a CS-based recovery

	Bandwidth	Occupied spectrum	Power consumption	Resolution of samples
NUS	2.0 GHz	100 MHz	5.8 W	8.8 ENOB, 55 dB SNDR

TABLE I: NUS specifications at a glance. The power includes the commercial ADC, but not the clock or the PRBS pattern generation. The chip allows up to 2.2 GHz of bandwidth though we do not have SNDR measurements for this range. The amount of occupied spectrum is conservatively estimated, and furthermore the amount of occupied spectrum can be increased when lower-fidelity recovery is acceptable.

algorithm to reconstruct an information-carrying modulated signal.

2) Other CS devices: The random modulation pre-integrator (RMPI), a type of random demodulator (RD), is a CS receiver and digitizer which has recently been implemented in [26]; see also [26] for references and related approaches. The RMPI uses a far more powerful CS approach that takes nearly random linear combinations of samples, akin to multiplying the digital vector x by a random matrix. The theory predicts that such an approach is optimal for nearly all types of sparse signals, not just signals that are sparse in the Fourier domain. The cost of this generality is that the RMPI is more difficult to implement, the signal processing is less straightforward, there is nontrivial calibration, and recovery is slower than for the NUS. The NUS also allows simple post-processing windowing techniques.

3) Xampling: The modulated wideband converter (MWC) [20], which follows the principles of xampling [19], works directly in the analog domain when possible. The approach requires signals that have a few dense bands of spectrum, such as three or four bands. The digital step is a continuous-to-finite (CTF) block that finds the location of the bands and must be run every time the band structure changes. However, the MWC does not naturally handle our sparse spectral model since the signals may not be easy to group into contiguous blocks. The hardware prototype in [20] has yet to be extended to an IC implementation.

In summary, these other CS and xampling approaches all have their own merits, but for the sparse spectral sensing model defined in the preceding subsection, we believe that the NUS is the best candidate.

D. Outline

In Section II, the implementation of our approach is described. Because signal recovery is non-standard, Section III covers the recovery process in detail, describing the general CS recovery method as well as the necessary changes and improvements for our specific architecture. Experimental hardware results are presented in Section IV, and the results of the prototype compare with previous state-of-the-art ADCs. The paper concludes in Section V with some learned wisdom and with a discussion of future challenges.

II. Hardware Implementation

A simplified block diagram of the non-uniform sampler (NUS) receiver is shown in Figure 2. The low-jitter 4.4 GHz clock is used to re-clock the non-uniform pattern to accurately set the sampling instances. For flexibility in testing, the NUS pattern is set by a repeating pseudo-random bit sequence (8192 bits in length) provided by an external pattern generator. A

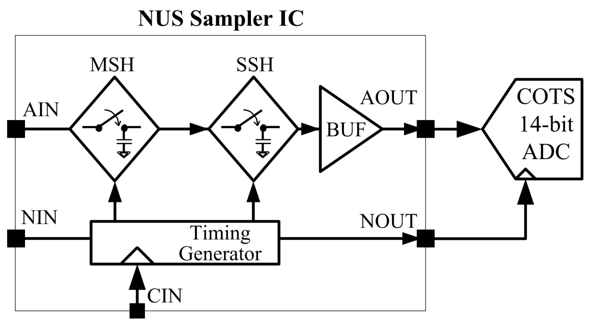


Fig. 2: Simplified block diagram of non-uniform sampler (NUS) receiver. The NUS sampler IC (left block) was implemented with the Northrop Grumman Aerospace Systems (NGAS) InP HBT process.

commercial 400 Msps ADC (TI ADS5474 [23]) is used to digitize the samples which are captured by a logic analyzer. In order to recover the signal, the samples must be aligned to the NUS pattern; this is accomplished with a synchronization pulse from the pattern generator.

The main building blocks of the NUS receiver are the master and slave sample-and-hold circuits, the timing generator, and the output buffer. In order to achieve the full bandwidth required, the master sample-and-hold circuit was designed with a 2.4 GHz bandwidth. The function of the NUS timing generator is to re-clock the NUS pattern (NIN) with the Nyquist clock (CIN). The output of the master-slave sample-and-hold is then buffered and amplified so that the signal can drive the external ADC. The output buffer bandwidth determines the settling time of the step-and-settle interface. The chip is designed for a full-scale input amplitude of 0.8 Vp-p differential and a 2.2 Vp-p differential at the output. The NUS IC is designed to perform a sample-and-hold function at a period as long as 6.1 ns and as short as 2.7 ns between consecutive samples, which is under the 400 MHz sampling limit of the ADC.

A. Circuit description

The main function of the timing generator, shown in Figure 3, is to generate timing signals for the master and slave sample-and-holds. This is accomplished by using a low-jitter clock to re-time and delay the NUS pattern input in a chain of flip-flops. Four flip-flops delays are used to delay the ADC clock to give adequate time for sampled signal to settle.

The master and slave sample-and-hold bridge circuits both use diode sampling bridges but have different power consumption based on bandwidth/spur requirements. Figure 4 shows the basic design of the sampling element. The circuit is controlled by the re-timed NUS pattern coming from the timing generator to switch the bridge on or off. The schematic shown is only for one of the two pseudo-differential circuits.

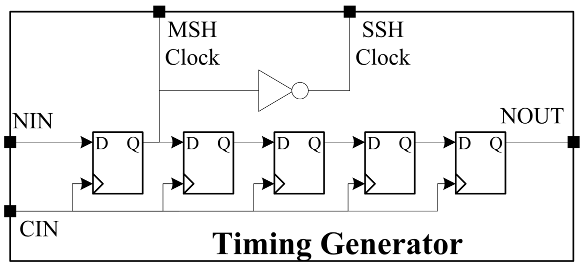


Fig. 3: Schematic of Timing Generator (TG) circuit.

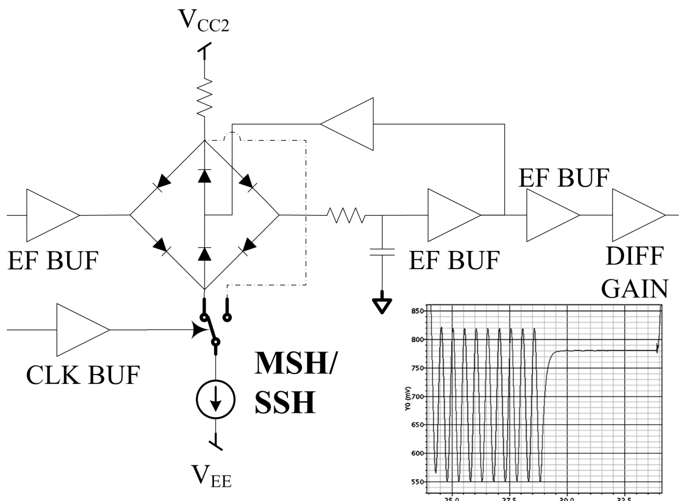


Fig. 4: Sample-and-hold circuit. The Master sample-and-hold (MSH) and Slave (SSH) are functionally identical. All signal paths and circuits are differential with the exception of the diode bridges. The diode bridges are implemented as two single-ended bridges. The graph shows the output at the transition from the tracking to the holding state (at the MSH output).

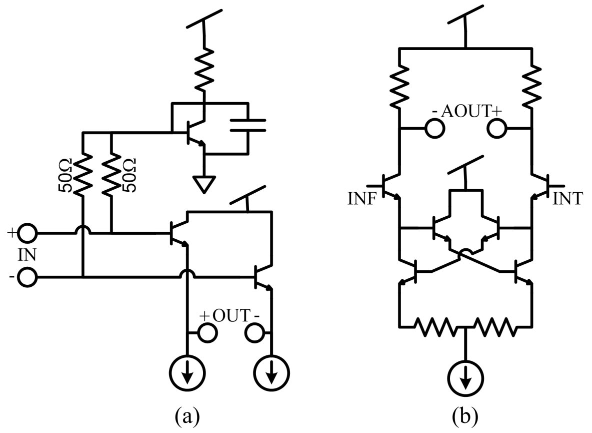


Fig. 5: NUS IC input/output interface circuits. (a) The analog input receiver is a differential 50 Ohm terminated emitter follower. This is a simplified version of the circuit called EF BUF shown in the MSH circuit in Figure 4. (b) The output buffer is a differential quartet design.

The analog input/output interface of the NUS IC was carefully designed to optimize performance. Figure 5(a) shows the NUS IC input receiver, which acts as a 50 Ohm load termination for the analog input signal and provides a low impedance drive to the subsequent sampling bridge. The analog output driver (Figure 5(b)) is a differential quartet design [10]; this was chosen because it offered the required performance while satisfying our finite power consumption goal. The driver was optimized for a large dynamic range and designed to be DC coupled to the TI ADS5474. The gain of the buffer is 2$\times$ (6dB), and it was designed to have a SFDR better than 70 dB and 500 MHz bandwidth. To increase simulation accuracy, a detailed interconnect model was used on the step-and-settle ADC interface. This model included bondwires and pallet traces for the NUS packaging, transmission line models for PCB traces, a termination network, and the equivalent loading model from the ADC datasheet.

The overall timing relationship between the NUS circuit and the external ADC is shown in the left panel of Figure 6. Also displayed in the right panel of Figure 6 is a simulation showing the NUS operation with a 1.6 GHz input sine wave and the resulting NUS samples.

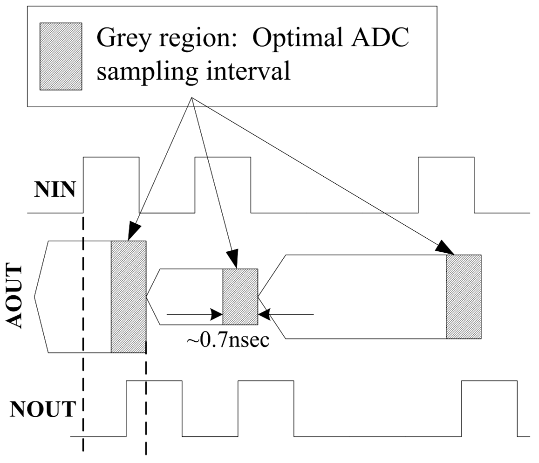
B. Circuit fabrication

The NUS IC is fabricated in Northrop Grumman Aerospace Systems' (NGAS) 0.45 μm InP HBT technology featuring f_T and $f_{\max} > 300$ GHz, 4-layer metal stack and precision TFR and MIM caps [16]. A die photograph of the $4.0 \text{ mm } \times 2.6 \text{ mm}$ NUS IC is shown in Figure 7. The NUS die is larger than it needs to be to allow the dicing of other die on the wafer. Pictured in Figure 8 is the NUS test fixture containing the NUS IC and TI ADS5474 along with various signal and power connectors. The PCB draws a total of 5.8 W: 3.2 W for the NUS IC and 2.6 W for the ADC.

C. Non-uniform sample pattern

The NUS sampling pattern is a pulse train with non-uniform spacing between pulses. It is selected to meet a list of certain criteria. First, the pattern is clocked at 4.4 GHz, and reconstruction produces the equivalent of Nyquist samples taken at this rate. Second, the pulse widths and spacings must satisfy the clocking requirements of the ADS5474. Specifically, the minimum pulse width is 6 clock cycles, the minimum spacing (Tmin) between pulses is 12 clock cycles, and the maximum spacing (Tmax) between pulses is 27 clock cycles. An example pattern illustrating these specifications is shown in Figure 9. In effect, as the sample spacings vary between 12 and 27 clock cycles, the instantaneous sampling rate of the NUS receiver varies between 163 MHz and 367 MHz, which is within the range of the 400 MHz ADC.

We designed the NUS pattern to repeat every 8192 Nyquist samples, during which time there are 440 pulses which set the sampling locations. This corresponds to an average sample rate of 236 MHz. We evaluate the quality of our pattern using a third criterion: the Fourier transform of favorable patterns will tend to have a flat, noise-like spectrum. Figure 10 compares two NUS patterns with different inter-sample spacings. The pattern shown in the top plots has strong resonances across the Nyquist band. In contrast, the pattern shown in the bottom plots, which has undergone a randomization of its sample



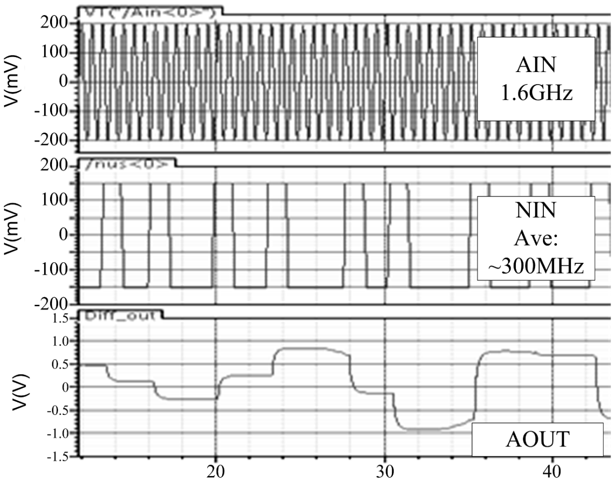


Fig. 6: NUS IC sampling timing and waveforms. Left panel: Interface timing between NUS die and ADC. Right panel: Simulated waveforms before and after being sampled by NIN. Horizontal scale is in ns.

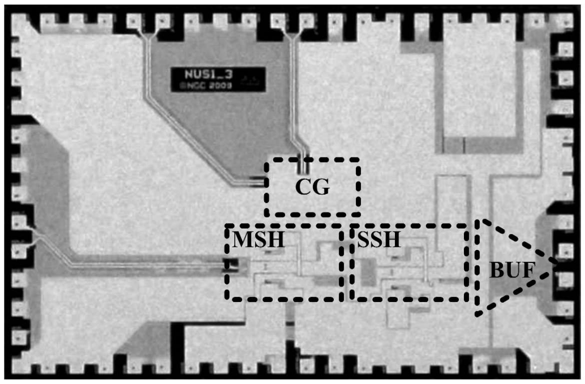


Fig. 7: NUS IC die photo. Die size is 4.0×2.6 mm.

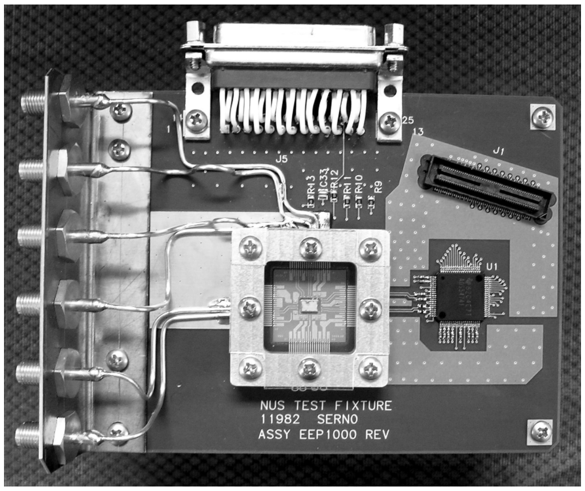


Fig. 8: NUS test fixture. The NUS IC is mounted on a custom pallet. Also shown is the 14-bit ADC as well as various test equipment connector interfaces.

locations, has a much whiter spectrum. The flat spectrum is preferred since then all signals have equal gain.

III. Data Processing

In this section, we describe our computational techniques for recovering a Nyquist-rate signal from the NUS data by filling in the missing samples. Sections III-A–III-D describe our procedures for windowing the NUS data and recovering the missing samples. Section III-E then briefly discusses additional practical concerns such as a GPU implementation to facilitate these computations.

A. Windowing

While the NUS produces an arbitrarily long sequence of samples, the recovery algorithm can only deal with a finite number of them at any given time. It is, therefore, necessary to segment the data stream, and we achieve this by windowing the signal. An effective windowing process must guard against edge effects as well as the well-known spectral spreading effect, which would destroy the very Fourier sparsity we seek to exploit. Fortunately, the concept of a *perfect reconstruction filter bank* (PRFB) [22] can be readily adapted to our purposes. A windowing procedure breaks the infinite signal into a series of (possibly overlapping) vectors by using an analysis window. After signal processing, the infinite length signal can be recovered by stitching together the finite series using the analysis window. Using windows from a PRFB ensures that the windowing process itself does not introduce any errors. An example of a PRFB is a rectangular analysis and synthesis window with no overlap, but of course this causes spectral spreading.

Figure 11 provides a filter bank representation of our processing chain. We let y denote the raw stream of NUS data, i.e., the discrete-time stream of samples coming from the non-uniformly clocked ADC. The processing begins by inserting zeros into y to produce a Nyquist-rate sample stream $\underline{y}$; the zeros are inserted at all locations where the NUS did not sample. A delay and downsampling chain then partitions

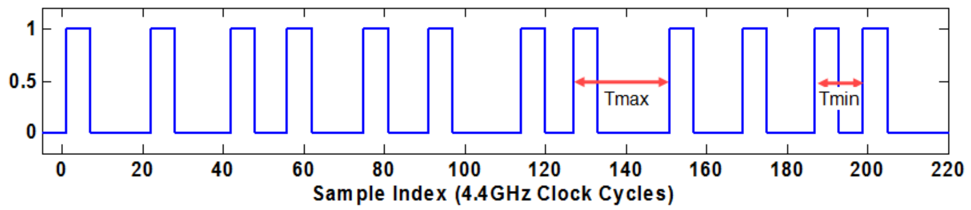


Fig. 9: NUS pattern example. High pulses have a width of 6 clock cycles. The minimum pulse spacing Tmin is 12 clock cycles, and the maximum pulse spacing Tmax is 27 clock cycles.

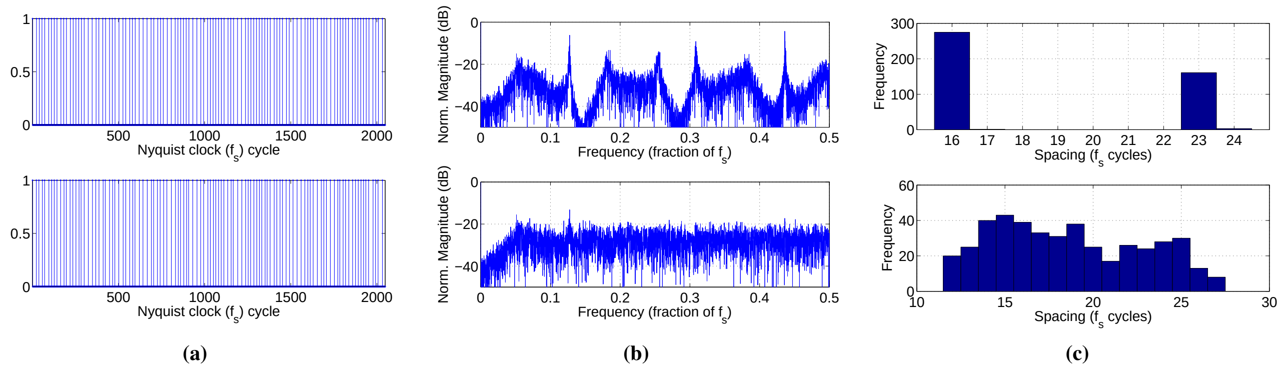


Fig. 10: Comparison of a non-optimized NUS pattern (top plots) and a properly randomized pattern (bottom plots). (a) Sample patterns in the time domain. (b) Spectral plots. (c) Histograms of inter-sample spacings.

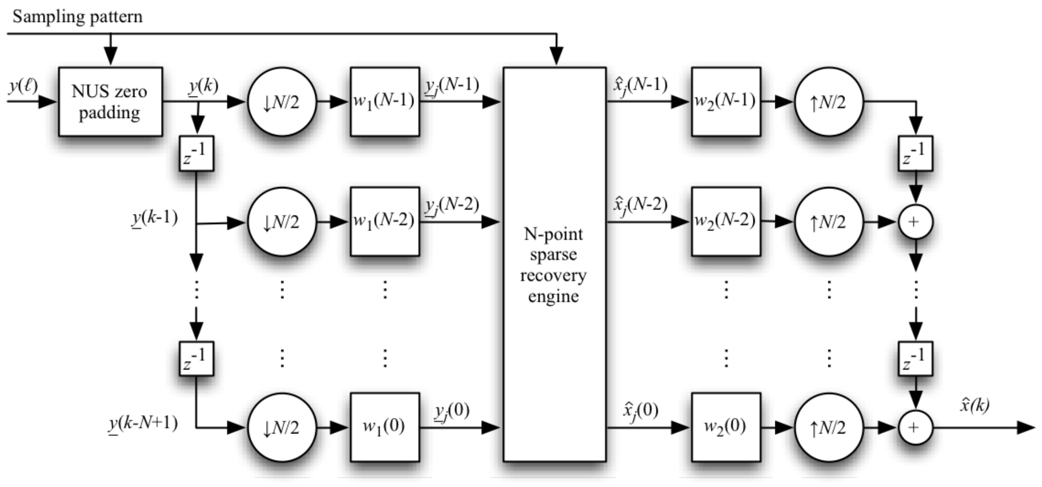


Fig. 11: A filter bank representation of the windowed recovery signal chain.

$\underline{y}$ into overlapping windows of length N and multiplies them by an *analysis window function* w_1. The result is a stream of N-point signals $\underline{y}_j \in \mathbb{R}^N$:

$$\underline{y}_j(i) = w_1(i)\underline{y}(i + jN/2), \quad i = 0, 1, \ldots, N-1.$$

Here, $\underline{y}_j(i)$ denotes sample position i in the j^{th} windowed signal. Because these windows are overlapping, each sample $\underline{y}(k)$ maps to two different entries (i, j):

$$k \to (k \bmod N/2 + N/2, \lceil 2k/N \rceil - 1), (k \bmod N/2, \lceil 2k/N \rceil).$$

The signals $\underline{y}_j$ are delivered to the sparse recovery engine, which produces a stream of estimates $\widehat{x}_j \in \mathbb{R}^N$. The upsampler delay chain stitches these estimates together using an N-point *synthesis window function* w_2 to yield the reconstructed Nyquist-rate sample stream $\widehat{x}$: for each integer j and each $i \in \{0, 1, \ldots, N-1\}$,

$$\widehat{x}(i + jN/2 + d) = w_2(i)\widehat{x}_j(i) + w_2(i + N/2)\widehat{x}_{j-1}(i + N/2).$$

Here d is the total system delay. The downsample and upsample chains introduce a combined delay of $N - 1$ Nyquist samples, so $d = N - 1$.

The perfect reconstruction criterion requires that the window functions w_1 and w_2 must satisfy

$$w_1(i)w_2(i) + w_1(i + N/2)w_2(i + N/2) = 1$$

for $i = 0, 1, 2, \ldots, N/2 - 1$. With this criterion satisfied, we can ensure that the performance of the system is limited by our precise choices of N, w_1, and w_2, and by the fidelity of our windowed sparse recovery algorithm. The design of the analysis window w_1 is critical, because it directly affects the spectrum of the estimated signals $\widehat{x}_j$. To minimize the effect of spectral spreading, we must choose an analysis window function w_1 whose spectral sidelobes are well below the system noise floor (high dynamic range), and a main lobe that is as narrow as possible (high sensitivity). We found experimentally that the square of a *Kaiser-Bessel derived* (KBD) window used in audio coding produces excellent results. Furthermore, this choice leads to a rectangular synthesis window (i.e., $w_2(i) \equiv 1$) [22], which is not only convenient but ensures that our reconstruction errors are weighted equally in time. We use $N = 65536$ and KBD parameter $\pi\alpha = 8$, for which the amplitude of the analysis window (in the frequency domain) falls below our system noise floor within 6 bins. This means that a windowed sinusoid will deliver non-trivial signal energy to no more than 11 DFT bins (of 32768 total bins). It is possible to improve upon this result by designing a window using convex optimization methods, but the KBD window is sufficient for our purposes.

While the PRFB-inspired architecture has proven useful for verifying the fidelity of our system design, our actual implementation differs in important practical respects. In particular, we choose NUS sampling patterns such that each half-window contains an identical number of NUS samples. This allows us to eliminate the zero-padding step altogether: the NUS samples can be partitioned into overlapping windows of M points each, and these windows can by multiplied by appropriately sampled versions of the analysis window function. The result is a more practical arrangement of processing steps depicted in Figure 12. The key notational difference from Figure 11 is that the zero-padded and windowed N-point signal $\underline{y}_j$ is now replaced with y_j, a windowed (but not zero-padded) M-point NUS signal.

With this architecture in place, we can treat each window separately if we choose—although we can (and do) take advantage of the spectral similarity between adjacent windows to improve performance. From this point forward, therefore, we shall focus solely on the generation of estimates $\widehat{x}_j \in \mathbb{R}^N$ of signals $x_j \in \mathbb{R}^N$, given non-uniform sample sets $y_j \in \mathbb{R}^M$ and exact knowledge of the sampling pattern. When it is clear from context that we are dealing with a single window, we will drop the j subscript altogether.

B. Frequency domain representation

The natural initial choice to compute the frequency domain representation of an N-point signal x is the discrete Fourier transform (DFT). For real signals, the DFT exhibits real symmetry, but the DC and Nyquist component are real-valued; for a more elegant treatment, we define a slight variant of the DFT that shifts the computed frequency bins by one half:

$$X_{\text{shift}}(k) = \frac{1}{\sqrt{N}} \sum_{i=0}^{N-1} e^{-\jmath 2\pi i(k+1/2)/N} x(i)$$

for $k = 0, 1, 2, \ldots, N-1$. This modified DFT remains orthonormal and preserves the sparsity behavior of compressible signals, but for real signals it exhibits a simpler symmetry: $X_{\text{shift}}(k) = \overline{X_{\text{shift}}(N-1-k)}$. Thus the frequency domain behavior is captured in $N/2$ complex values. If we preserve only the first $N/2$ frequencies and scale by $\sqrt{2}$ to preserve orthonormality, the result is what we call the *half-bin FFT* (HBFFT):

$$X(k) = \sqrt{\frac{2}{N}} \sum_{i=0}^{N-1} e^{-\jmath 2\pi i(k+1/2)/N} x(i)$$

for $k = 0, 1, 2, \ldots, N/2 - 1$. Let $\mathcal{F} : \mathbb{R}^N \to \mathbb{C}^{N/2}$ denote the real-to-complex HBFFT operation, so $X = \mathcal{F}(x)$. Because $\mathcal{F}$ is orthonormal, we have $x = \mathcal{F}^*(X)$, where $\mathcal{F}^*$ denotes the complex-to-real adjoint of $\mathcal{F}$.

It turns out that the HBFFT $\mathcal{F}$ can be computed as the composition of a single custom butterfly, a standard $N/2$-point complex DFT, and a simple reshuffling. This is because the computation of the even entries of X_{shift} can be written as follows:

$$X_{\text{shift}}(2\ell) = \frac{1}{\sqrt{N}} \sum_{i=0}^{N/2-1} e^{\frac{-\jmath\pi i}{N}} \left(x(i) - \jmath x(i+N/2)\right) e^{\frac{-\jmath 2\pi \ell i}{N/2}}$$

for $\ell = 0, 1, 2, \ldots, N/2 - 1$. Then we simply have $X(k) = \sqrt{2} \cdot X_{\text{shift}}(k)$ when k is even, and $X(k) = \sqrt{2} \cdot \overline{X_{\text{shift}}(N-1-k)}$ when k is odd. To perform the inverse operation, we recover the quantities $X_{\text{shift}}(2\ell)$ by reversing the reshuffling step, and compute an intermediate quantity $z(i)$ using a standard complex inverse FFT followed by a complex scaling:

$$z(i) = \sqrt{\frac{2}{N}} e^{\jmath\pi i/N} \sum_{\ell=0}^{N/2-1} e^{\jmath 2\pi \ell i/(N/2)} X_{\text{shift}}(2\ell)$$

for $i = 0, 1, 2, \ldots, N/2 - 1$. Then we can extract x from the real and imaginary parts of z:

$$x(i) = \begin{cases} \Re(z(i)) & i = 0, 1, 2, \ldots, N/2 - 1 \\ -\Im(z(i - N/2)) & i = N/2, N/2 + 1, \ldots, N - 1. \end{cases}$$

The tight relationship between the HBFFT and the standard complex FFT allows us to achieve high performance with standard FFT libraries.

C. Reprojection on estimated support

If the support of the signal is known—that is, if we know which frequency bins contain active signal content—and is sufficiently sparse, then we can reduce the reconstruction process to a standard least-squares problem we now introduce. Let $x \in \mathbb{R}^N$ represent the Nyquist-rate signal we wish to estimate, and let $y \in \mathbb{R}^M$ be the NUS samples. Those samples are selected from indices $\mathcal{I}_T \subseteq \{0, 1, 2, \ldots, N-1\}$, $|\mathcal{I}_T| = M$. Thus, x and y satisfy $y = E_T x$, where $E_T \in \mathbb{R}^{M \times N}$ is assembled from rows $i \in \mathcal{I}_T$ of the $N \times N$ identity matrix. Our task is to construct an estimate $\widehat{x} \in \mathbb{R}^N$ of x given these samples y. Using the real-to-complex HBFFT operator defined in Section III-B, we let $X = \mathcal{F}(x)$ and $\widehat{X} = \mathcal{F}(\widehat{x})$ denote the frequency domain representations of x and $\widehat{x}$, respectively.

Let $\mathcal{I}_F \subseteq \{0, 1, 2, \ldots, N/2-1\}$, $|\mathcal{I}_F| = P \leq M/2$, denote the support of the signal. We can write our estimate as $\widehat{X} =$

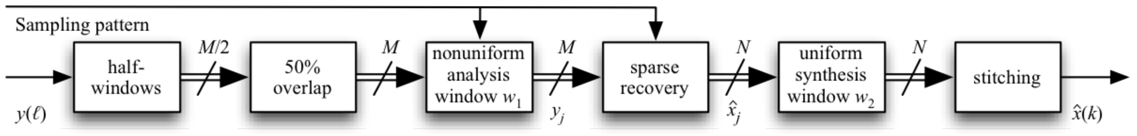


Fig. 12: A block diagram of the practical processing steps.

$E_F^* Z$, where $Z \in \mathbb{C}^P$ is a set of nonzero coefficients to be determined below, and $E_F \in \mathbb{R}^{P \times N/2}$ is a frequency sampling matrix assembled from rows $i \in \mathcal{I}_F$ of the $N/2 \times N/2$ identity matrix.

With these definitions in place, the reprojection problem can be cast as

$$\widehat{x} = \mathcal{F}^*(E_F^* Z), \quad Z \triangleq \underset{Z}{\operatorname{argmin}} \, \| E_T \, \mathcal{F}^*(E_F^* Z) - y \|_2.$$

To help explain the above notation, let us note that $E_T \, \mathcal{F}^*(E_F^* Z)$ is computed from Z by constructing a length-$\frac{N}{2}$ vector containing the P entries of Z in the positions indexed by $\mathcal{I}_F$, computing the complex-to-real inverse HBFFT of this vector, and extracting from the result the M values in the positions indexed by $\mathcal{I}_T$. The minimization on Z is a least-squares problem and can be expressed as normal equations

$$E_F \, \mathcal{F}(E_T^* E_T \, \mathcal{F}^*(E_F^* Z)) = E_F \, \mathcal{F}(E_T^* y).$$

The linear operation $Z \to E_F \, \mathcal{F}(E_T^* E_T \, \mathcal{F}^*(E_F^* Z))$ is positive definite, but it cannot be expressed as a complex matrix due to the presence of the complex-to-real operation $\mathcal{F}^*$. However, we have chosen to solve this form using conjugate gradients, which allows us to utilize the HBFFT and sampling operators directly. Furthermore, CS theory shows that the linear operator is well-conditioned, so conjugate gradients will converge rapidly.

D. Spectral occupancy estimation

The problem of recovering a signal from NUS samples can be partitioned into two subproblems: (1) first estimate the support of the unknown signal in the frequency domain (we refer to this step as *spectral occupancy estimation*), and then (2) reproject to estimate the signal given this estimated support. We have developed a customized algorithm for spectral occupancy estimation that is inspired by existing techniques in CS but adapted to the specific nuances of our problem.

The most unique aspects of our problem that we seek to exploit are as follows: (1) The nonzero HBFFT coefficients for a given window—while few in number—also tend to cluster into an even smaller number of contiguous groups. In the CS literature, this is known as a *structured sparsity model* [4]. Our algorithm is inspired by existing ones in the model-based CS literature designed to exploit block-sparse [15] and clustered-sparse [11] models. (2) Although our data are partitioned into finite windows (as described in Section III-A), the spectral occupancy is often stationary over the duration of multiple windows. In cases like this—where multiple sparse signals share the same support—the signals are said to obey a *joint sparsity model*. Like others in the distributed CS literature [5], our algorithm processes the data from multiple windows jointly in order to better identify the support.

Our spectral occupancy estimation algorithm is greedy: we first run a few iterations of a greedy selection rule that builds an estimate of the support, and we then perform a pruning procedure to remove false positives. The greedy selection procedure (step 1 below) is iterative because it is difficult to identify all of the active frequencies at once; at each iteration, only the largest frequencies can be accurately estimated, since artifacts from these large signals will hide smaller signals. As illustrated in Figure 13, however, once some blocks of active frequencies have been identified, a reprojection step removes their influence from the measurements, and weaker active frequencies can then be identified. We find the subsequent pruning (step 2 below) to be helpful because setting the thresholds in step 1 low enough to detect weak signals tends to also introduce a number of false positives. Overall, our algorithm most closely resembles the well-known OMP and CoSaMP algorithms in CS [21], [24], although we have also experimented with reweighted ℓ_1 minimization [9] and believe that, with appropriate modifications, it could be competitive as well.

To describe our algorithm, let us set the following notation. Let $x_j \in \mathbb{R}^N$ represent the unknown Nyquist-rate samples from window number j, and let $y_j \in \mathbb{R}^M$ represent the NUS samples. Those samples are selected from indices $\mathcal{I}_{T,j} \subseteq \{0, 1, 2, \ldots, N-1\}$, $|\mathcal{I}_{T_j}| = M$, and so following the notation defined in Section III-C, $y_j = E_{T,j} x_j$, where $E_{T,j} \in \mathbb{R}^{M \times N}$. From y_j we would like to estimate the positions $\mathcal{I}_F$ of the non-negligible entries of $X_j = \mathcal{F}(x_j)$. We accomplish this by considering an ensemble of windows $j \in \{0, 1, \ldots, J-1\}$ simultaneously and exploiting the assumption of stationarity (i.e., we assume that $\mathcal{I}_F$ does not change from window to window). Our algorithm consists of the following steps:

1) Preliminary support estimation
   a) Set the iteration count $\ell = 1$, and for each window j, define a residual vector $r_j = y_j$. Set the initial support estimate to be empty: $\widehat{\mathcal{I}_F} := \emptyset$. Set the maximum allowable size for the support estimate $P_{\max} \approx 0.8\frac{M}{2}$. Set $h_{\text{old}}(i) = 0$ for each $i \in \{0, 1, \ldots, N/2 - 1\}$.
   b) For each window j, compute the correlation statistics $h_j = \mathcal{F}(E_{T,j}^* r_j)$. Then, square and sum the correlation statistics over multiple windows: for each $i \in \{0, 1, \ldots, N/2 - 1\}$, compute the aggregate statistic $h_{\text{agg}}(i) = \sum_{j=0}^{J-1} |h_j(i)|^2$.
   c) Identify a set Γ of possible active frequencies. We add an index $i \in \{0, 1, \ldots, N/2 - 1\}$ to Γ if $|h_j(i)|$ is frequently among the largest entries of h_j across multiple windows j, or if $|h_{\text{agg}}(i)|$ is among the largest entries of h_{agg}. We also include all entries of the previous support estimate $\widehat{\mathcal{I}_F}$.

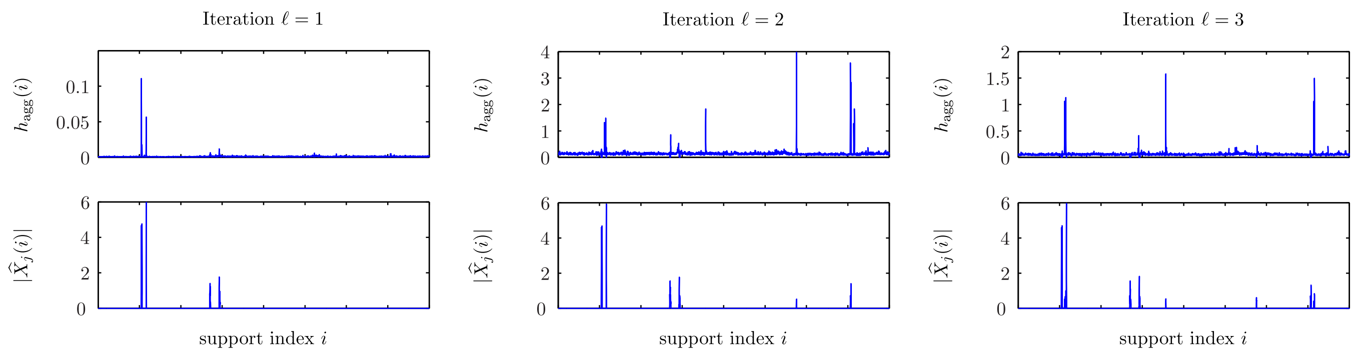


Fig. 13: Three iterations of the greedy selection procedure for preliminary support estimation. In each iteration, blocks of indices with high aggregate energy estimates $h_{\text{agg}}(i)$ are identified (top row). A reprojection step then removes their influence from the measurements so that additional active blocks can be identified in subsequent iterations. (The aggregate statistics $h_{\text{agg}}(i)$ are plotted before being reset with $h_{\text{old}}(i)$ in step 1c of our algorithm below.) Plots in the bottom row show, for one of J windows used in the estimation, the reprojected spectrum estimate $|\widehat{X}_j(i)|$ using the current support estimate.

For the purpose of computations below, we then set $h_{\text{agg}}(i) = h_{\text{old}}(i)$ for each $i \in \widehat{\mathcal{I}_F}$ and reset $\widehat{\mathcal{I}_F} = \emptyset$.

d) Pad the set Γ with some number p of indices on both sides of each index in Γ. For example, if $p = 1$ and $\Gamma = \{2, 3, 4, 6, 18, 19\}$, update Γ to $\{1, 2, 3, 4, 5, 6, 7, 17, 18, 19, 20\}$.

e) Identify contiguous blocks of indices in Γ, and compute the estimated energy of each block. For example, for the updated Γ given above, two blocks are identified, and their corresponding energies are $h_{\text{agg}}(1)+\cdots+h_{\text{agg}}(7)$ and $h_{\text{agg}}(17)+\cdots+h_{\text{agg}}(20)$.

f) Populate the new support estimate $\widehat{\mathcal{I}_F}$ with all of the indices from the highest energy blocks, such that $|\widehat{\mathcal{I}_F}|$ does not exceed $\frac{\ell}{10}P_{\max}$. This increasing threshold allows slightly larger support estimates at each iteration.

g) On each window, use a reprojection step to project the observations y_j orthogonal to the chosen support $\widehat{\mathcal{I}_F}$, and let r_j denote the resulting residual.

h) Store the aggregate energy estimates for use in future iterations, setting $h_{\text{old}}(i) = h_{\text{agg}}(i)$ for all i. Then, increment the iteration counter ℓ. Stop when $\ell = 10$ or the energy in the residual vectors r_j is sufficiently small. Otherwise, repeat steps (1b) through (1g).

2) Final pruning

a) Set the iteration count $\ell = 1$.

b) Reproject each set of samples y_j onto the estimated support $\widehat{\mathcal{I}_F}$ to obtain an estimate $\widehat{X}_j$ for the HBFFT coefficients. Square and sum these estimates: for each $i \in \{0, 1, \dots, N/2 - 1\}$, compute $\widehat{X}(i) = \sum_{j=0}^{J-1} |\widehat{X}_j(i)|^2$.

c) For each contiguous block of indices in $\widehat{\mathcal{I}_F}$, compute the largest value of $\widehat{X}$, e.g., if $\widehat{\mathcal{I}_F} = \{2, 3, 4, 6, 18, 19\}$, compute $\max\{\widehat{X}(2), \widehat{X}(3), \widehat{X}(4)\}$, $\widehat{X}(6)$, and $\max\{\widehat{X}(18), \widehat{X}(19)\}$.

d) Remove blocks from $\widehat{\mathcal{I}_F}$ whose maximum $\widehat{X}$ value does not exceed some threshold designed to eliminate false positives. Increment the iteration counter ℓ.

e) Repeat steps (2b) through (2d) for a small number of iterations.

f) Following the same procedure as in step 1d, pad the support estimate $\widehat{\mathcal{I}_F}$ with some number p of indices on each side of each estimated block. (This procedure operates only so long as $|\widehat{\mathcal{I}_F}| \leq P_{\max}$.)

After running the entire support estimation algorithm on an ensemble of J windows, one can re-run the algorithm on one or more subsequent ensembles of J windows, either for cross-validation purposes or to detect changes in the spectral occupancy.

E. Additional implementation concerns

1) Model violations: Our system is designed to support signals with up to 100 MHz of information bandwidth. There are a number of strategies that one could use to confirm that the input signal obeys this model assumption. For example, in step 1 of the support estimation algorithm described in Section III-D, the energy of the residual vectors r_j should decrease substantially as the number of iterations increases. If significant energy remains in the residual vectors after the maximum number of iterations, this means that the estimated support is not sufficient to fully capture the structure in the input signal. A second possible strategy for detecting model violations could be cross validation. For example, 95% of the NUS samples could be used for support estimation and recovery, and the remaining 5% of the NUS samples could be checked against the reconstructed estimates. A close match suggests that the information bandwidth is well captured in the estimated support.

In cases where small model violations are detected, CS theory guarantees that the reconstruction on the estimated support will be relatively accurate, although the small signal components away from this support will of course not be reconstructed. In cases where substantial model violations are detected, reconstruction on the estimated support will not be accurate. There is research into additional analog (pre-processing) and digital (post-processing) safeguards that could

Number of C2050 GPUs	1	3	6
Performance, Gflops/sec	145	426	808
Linearity	N/A	98%	93%
GPUs for real time	83	84	89

TABLE II: Performance of our best GPU cluster in the reprojection benchmark.

be added to our system to protect against such situations. If the model is violated too frequently, then the NUS is simply not the correct device for the task.

2) Computation: Because our system is designed to have a high EIBW, the calculations described in Sections III-C-III-D can be expensive. To quantify these costs, suppose we solve the reprojection problem by applying conjugate gradients (CG) to the normal equations. The cost of doing so is dominated by the FFTs, each of which requires $2.5N\log_2 N$ flops. Two FFTs are required per CG iteration, and one each for initialization and finalization. Because the windows overlap by 50%, reconstruction occurs at a rate of of $N/2$ samples per window. Therefore, the throughput required to perform real-time reprojection is

$$\begin{aligned} C &= f_r \cdot (2I+2) \cdot 2.5N \log_2 N \,/\, (N/2) \\ &= 10 f_r (I+1) \log_2 N \quad \text{flops/sec}, \end{aligned}$$

where I is the number of CG iterations and f_r is the reconstructed sample rate. Although we omit the details here, in some problems where the EIBW is less than half of the device Nyquist rate f_s we can envision using digital downconversion to reconstruct at a rate f_r equal to just twice the EIBW. When we are interested in the 800 MHz–2 GHz band, for example, it is possible to reconstruct at a rate of just $f_r = 2.4$ Gsps.[2] Taking this number as an example, with $(N, I, f_r) = (65536, 30, 2.4\,\text{Gsps})$ we estimate that reconstruction would require $C \approx 12$ Tflops/sec. No single processor achieves this type of performance; parallelism must certainly be exploited.

Is this level of performance possible? While we have not yet achieved it, we have conducted a variety of tests to demonstrate the feasibility of using graphics processing units (GPUs) to accelerate the key computations. Benchmarks provided by NVIDIA suggest that a single C2050 Tesla GPU can achieve a throughput of 175 Gflops/sec when performing complex FFTs of our required length [1]. Under optimistic assumptions of linear parallelism and no performance losses in our algorithm, we can predict that real-time performance would require at least 69 GPUs.

Using MATLAB, C++, and NVIDIA's CUDA computational libraries, we have constructed a multiple-GPU implementation of our reprojection algorithm. This code is executed on a system employing a single CPU and 6 Tesla C2050 GPUs. To minimize losses due to GPU pipelining and CPU/GPU communication, each GPU processes 2048 windows simultaneously. Our measurements of the performance of this system are summarized in Table II. Our performance is about 17% lower on a single GPU than the NVIDIA FFT benchmark, but a high degree of linearity is maintained for 6 GPUs. Extrapolating from the 6-GPU results suggests that real-time performance would require approximately 89 GPUs.

An important question is whether linearity can be preserved for such a large number of GPUs. We believe that this would be the case for for two reasons. First, the parallelism is coarse-grained: each GPU processes a separate block of data, independently of the others. Second, the communication requirements are determined primarily by the signal environment—the NUS rate, the Nyquist rate, and the respective word sizes—and not by the number of GPUs. Thus while a real-time system would certainly be expensive, we are optimistic that it could be built.

We have not yet discussed the costs of support identification. The bulk of the effort for estimating the spectral occupancy is consumed by the reprojections, and about 16 of these are performed in a typical run. Performing full support identification on every window, then, would multiply the computational burden by approximately 16. However, support identification need not be performed on every window if the signal environment is relatively stationary; the results from one window can be used in many subsequent reprojections. More work is needed to understand the tradeoff between the computational costs and the ability to track a dynamic signal environment.

IV. Testing and Validation

A. Experimental setup

In order to test our complete NUS architecture in a representative environment, we conducted a series of experiments using realistic GSM data. Signals were generated by an arbitrary waveform generator (AWG) and a vector signal generator (VSG).

For each experiment, we construct a 270.833 kbps GSM signal that has a bandwidth of $\sim$1 MHz (at -50 dBc) and is located at a center frequency of 1.595 GHz. For measurements the GSM signal power is scaled anywhere from -20 dBFS down to -80 dBFS. To the scaled GSM signal, we add various levels of "clutter" to the spectrum consisting of narrowband RF signals having random amplitudes and centered at random frequencies between 800 MHz and 2 GHz. Clutter signals with bandwidths of 20 MHz, 50 MHz, or 100 MHz are used, and two different cases were generated for each bandwidth. The clutter is used to increase the information bandwidth of the signal, even though the clutter itself is not of interest. Figure 14 shows example signals, including the GSM signal and the clutter. It is important to note that our measure of the clutter bandwidth includes spectral "tails" down to -90 dBFS (measured after windowing); as the experiments illustrate, however, it is possible to reconstruct the signal with high accuracy while omitting these tails and thus requiring less overall bandwidth.

After measurement, the NUS data are arranged into blocks of size $M = 3520$ (each corresponding to $N = 65\,536$

[2]This would require using a device Nyquist rate of 4.8 Gsps, which is part of our future specification for the system. In addition, we note that a system using downconversion would more realistically output complex-valued samples at 1.2 Gsps rather than real-valued samples at 2.4 Gsps, although the cost is the same.

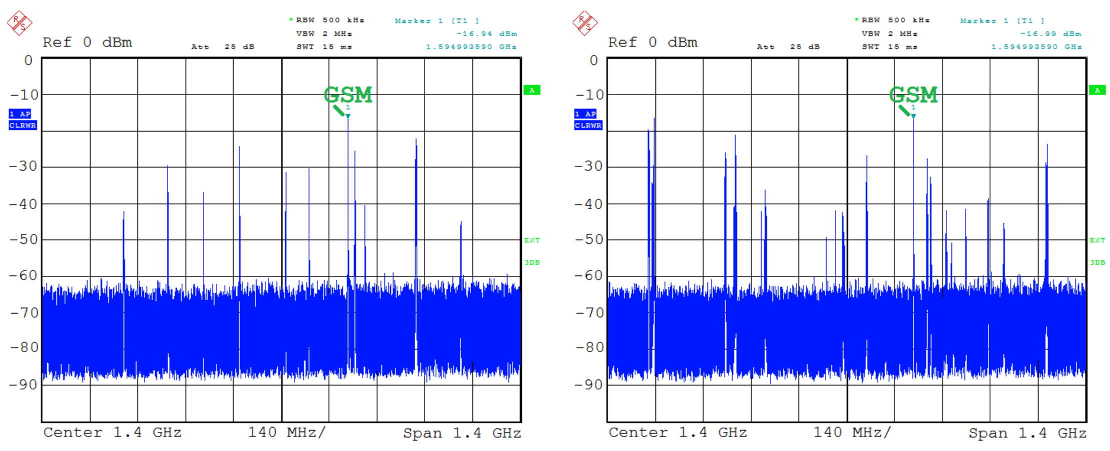


Fig. 14: Example GSM test spectra with 20 MHz clutter (left) and 100 MHz clutter (right). Spectra are plotted from 700 MHz–2.1 GHz. The GSM signal is located at 1.595 GHz and is indicated with a green marker.

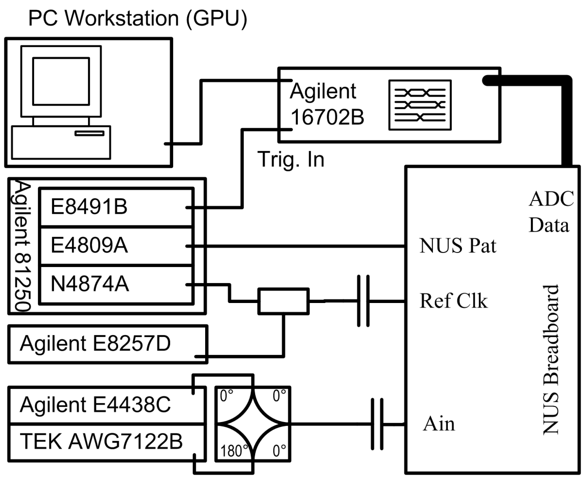


Fig. 15: NUS test setup. The GSM signal is produced by the E4438, clutter is produced by the AWG7122, and the NUS pattern is produced by the 81250. ADC data is captured via the 16702 and then downloaded and processed by the GPU.

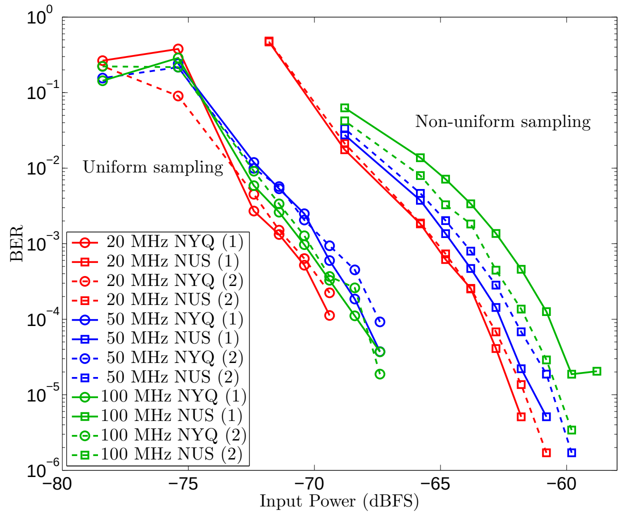


Fig. 16: BER of the decoded GSM signal as a function of input power. Circular markers indicate the performance of the uniform ADC for each of two randomly generated signals (denoted (1) or (2)) at each of three levels of clutter (20, 50 and 100 MHz). Square markers indicate the performance of the NUS on the same signals. The solid and dashed lines correspond to separate trials.

Nyquist rate samples) and run through the spectral occupancy estimation algorithm (Section III-D) using $J = 31$ overlapping windows (these span a total of 1 048 576 Nyquist-rate samples). As a means of cross validation, the support estimation procedure is repeated 10 times (each on a fresh set of $J = 31$ overlapping windows). A frequency bin is included in the final support estimate if it appears in at least 2 out of the 10 preliminary estimates.

With the estimated support, we reproject the NUS data on each window to recover an estimate of the Nyquist-rate signal samples. The windowed samples are then recombined as described in Section III-A. These estimated Nyquist-rate samples are then passed through a GSM decoder (that has a priori knowledge of the center frequency of 1.595 GHz) to measure the BER. Note that input powers in the range of -55 to -75 dBFS yield measurable BER. Above -55 dBFS the BER drops to a rate that makes collecting and processing an adequate number of samples difficult. Very low power inputs yield high BERs and make synchronization of the decoder difficult; the GSM signal will only be present in our reprojected samples if the band around 1.595 GHz is correctly identified as part of the support, and this becomes less likely when the input power is very low.

A simplified diagram of the NUS test setup used in the experiment is shown in Figure 15. Not shown are the controller connections, differential lines, filtering, and power supplies.

For the sake of comparison, we also sampled the test signals using an NGAS developed 5 Gsps 8-bit Nyquist ADC that uses the same InP technology. A description of an earlier version of this ADC chip can be found in [12]. This ADC uses folding-interpolating architecture, has a greater than 7 ENOB performance, and draws 9.6 W. Because the uniform ADC automatically produces Nyquist samples, its output can be passed directly to the GSM decoder. Testing was done using a 4.4 GHz clock and the output was sub-sampled by four because of equipment limitations.

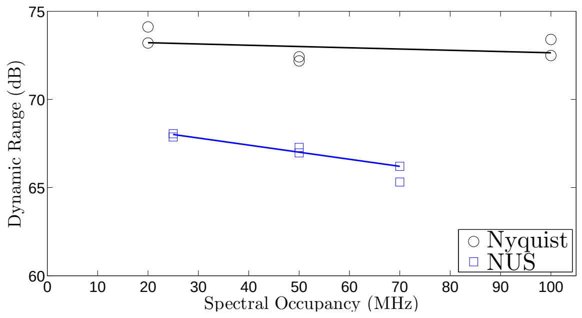


Fig. 17: Dynamic range of the uniform ADC and the NUS as a function of the information bandwidth. The information bandwidth plotted for the NUS curve indicates the total amount of bandwidth identified by the spectral occupancy estimation algorithm. For the signal with 100 MHz of clutter, it is possible to achieve high dynamic range while identifying less than 80 MHz of occupied bandwidth; this is because only the small tails of the clutter signal are omitted.

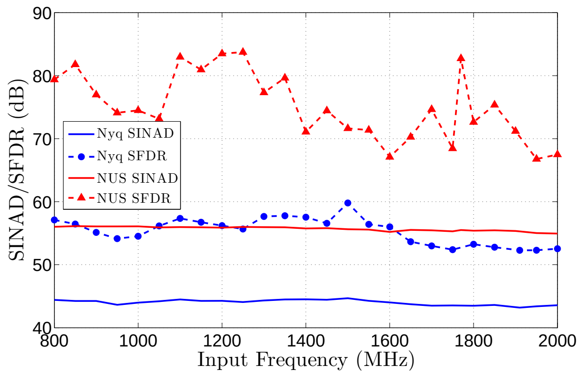


Fig. 18: Comparison of the SINAD and SFDR versus frequency of the Nyquist ADC and the NUS in uniform sampling mode.

B. Experimental results

Figure 16 plots the BER as the input power of the GSM signal is reduced. Circular markers indicate the performance of the uniform ADC for each of two randomly generated environments (distinguished by solid and dashed lines) at each of three levels of clutter (distinguished by color). Square markers indicate the performance of the NUS on the same signals. With the uniform ADC, we see relatively little variation in BER across the various signals. That is, the performance of the uniform ADC does not depend on the information bandwidth of the input signal.

With the NUS, in contrast, for signals with higher levels of information bandwidth (more clutter), we do see an increase in the BER. This is to be expected, since the difficulty of accurately estimating the spectral occupancy increases, and the accuracy of the reprojected signal will degrade slightly.

Figure 17 captures this trend more clearly by plotting the dynamic range of the two systems as a function of the information bandwidth. For this graph, dynamic range is defined as the minimum input power (dBFS) that yields a BER of 10^{-2}. For all levels of the information bandwidth, the uniform ADC has a higher dynamic range, but this is not surprising because the uniform ADC collects more samples in total (approximately $4.7\times$ more than the NUS in this case).

As a separate experiment, we operate our NUS architecture in a 300 MHz uniform sampling mode: samples are not spaced according to the PRBS but rather occur in equispaced intervals. This allows the system to be treated like a low-rate ADC and characterized without the need for reconstruction. Figure 18 compares the SINAD and SFDR (as a function of input frequency) of the Nyquist ADC and the NUS. The difference in sampling rates between the ADC and the NUS is not a problem since it does not affect the measurements, only the amount of folding that occurs. The plots show the NUS has a 10 dB advantage in both SINAD and SFDR while consuming about half the power. The SINAD advantage partially offsets the noise penalty of undersampling. Because of the wideband nature of the GSM+clutter signals, the SFDR advantage of the NUS was not obvious. Other measurements (not shown) done with only multi-carrier GSM signals present do hint at the performance advantage of the NUS.

V. Conclusions

In summary, we have presented a custom monolithic implementation of a non-uniform sampler that uses the principles of compressed sensing to entirely reconstruct the input signal. While the NUS can be similar in function to an ADC, we emphasize that it is not a drop-in replacement for an ADC. Rather, it is a powerful tool in the signal processing toolkit that is useful when the signal is supported on a small (unknown) bandwidth inside a large frequency range.

Our results show that the hardware and algorithms are performing as expected and that the custom S/H is not a bottleneck of the system. Thus, by using a faster off-the-shelf ADC, the NUS could be scaled to even higher bandwidths. The remaining limitations are the processing speed and power, and the assumption of sparsity. Our current research addresses the former issue with our custom hybrid reconstruction algorithm and a custom GPU implementation, and future work will be on improving this further to achieve real-time recovery. The assumption of sparsity is more fundamental. The underlying issue is that the signal information rate cannot exceed the sampling rate, and spectral sparsity is a convenient proxy for the information rate. For specific applications that have a tighter signal model (for example, a known form of modulation) it may be possible to devise improved recovery algorithms that have a more relaxed sparsity assumption; such ideas are being studied under the name of model-based compressed sensing, and we leave the application of these ideas to the NUS for future work.

References

[1] Tesla c2050 performance benchmarks. http://www.siliconmechanics.com/files/C2050Benchmarks.pdf.
[2] B. Adcock and A. C. Hansen. Generalized sampling and infinite dimensional compressed sensing. *Submitted*, 2011.
[3] M. B. Airola, S. R. O'Connor, M. L. Dennis, and T. R. Clark, Jr. Experimental demonstration of a photonic analog-to-digital converter architecture with pseudorandom sampling. *IEEE Photonics Tech. Lett.*, 20(24), 2008.
[4] R. G. Baraniuk, V. Cevher, M. F. Duarte, and C. Hegde. Model-based compressive sensing. *IEEE Transactions on Information Theory*, 56(4):1982–2001, 2010.
[5] D. Baron, M. F. Duarte, M. B. Wakin, S. Sarvotham, and R. G. Baraniuk. Distributed compressive sensing. *Arxiv preprint arXiv:0901.3403*, 2009.

[6] E. J. Candès and Y. Plan. A probabilistic and RIPless theory of compressed sensing. *IEEE Transactions on Information Theory*, 57(11):7235–7254, November 2011.

[7] E. J. Candès, J. Romberg, and T. Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Trans. Inform. Theory*, 52(2):489–509, 2006.

[8] E. J. Candès and M. Wakin. An introduction to compressive sampling. *IEEE Sig. Proc. Mag.*, 25(2):21–30, 2008.

[9] E. J. Candès, M. B. Wakin, and S. P. Boyd. Enhancing sparsity by reweighted ℓ_1 minimization. *J. Fourier Anal. Appl.*, 14(5–6):877–905, Dec. 2008.

[10] R. Capio. Precision differential voltage-current convertor. *Electronics Letters*, 9(6):147–148, 1973.

[11] V. Cevher, P. Indyk, C. Hegde, and R.G. Baraniuk. Recovery of clustered sparse signals from compressive measurements. In *Proc. Int. Conf. Sampling Theory and Applications (SampTA)*, 2009.

[12] B. Chan, B. Oyama, C. Monier, and A. Gutierrez-Aitken. An Ultra-Wideband 7-Bit 5-Gsps ADC Implemented in Submicron InP HBT Technology. *IEEE Journal of Solid-State Circuits*, 43(10):2187–2193, October 2008.

[13] M. A. Davenport and M. B. Wakin. Compressive sensing of analog signals using discrete prolate spheroidal sequences. *Arxiv preprint arXiv:1109.3649*, 2011.

[14] D. L. Donoho. Compressed sensing. *IEEE Trans. Inform. Theory*, 52(4):1289–1306, April 2006.

[15] Y. C. Eldar, P. Kuppinger, and H. Bölcskei. Block-sparse signals: uncertainty relations and efficient recovery. *IEEE Transactions on Signal Processing*, 58(6):3042–3054, 2010.

[16] A. Gutierrez-Aiken et al. Advanced InP HBT Technology at Northrop Grumman Aerospace Systems. In *Proc. IEEE Compound Semiconductors Integrated Circuit Symposium*, 2009.

[17] Y.-P. Lin and P. P. Vaidyanathan. Periodically nonuniform sampling of bandpass signals. *IEEE Trans. Circuits Syst. II*, 45(3):340–351, 1998.

[18] M. Mishali and Y. C. Eldar. Blind multiband signal reconstruction: Compressed sensing for analog signals. *IEEE Trans. Signal Process.*, 57(3):993–1009, 2009.

[19] M. Mishali and Y. C. Eldar. From theory to practice: Sub-Nyquist sampling of sparse wideband analog signals. *IEEE J. Sel. Top. Signal Process.*, 4(2):375–391, 2010.

[20] M. Mishali, Y. C. Eldar, O. Dounaevsky, and E. Shoshan. Xampling: Analog to digital at sub-Nyquist rates. *IET Cir. Dev. and Systems*, 5(1):8–20, Jan. 2011.

[21] D. Needell and J. A. Tropp. CoSaMP: Iterative signal recovery from incomplete and inaccurate samples. *Appl. Comput. Harmon. Anal*, 26:301–321, 2009.

[22] J. O. Smith. *Spectral Audio Signal Processing*. W3K publishing, October 2008. http://ccrma.stanford.edu/~jos/sasp/, accessed 7 December 2011.

[23] Texas Instruments (2008, Aug). 14-Bit, 400-MSPS Analog-to-Digital Converter. [Online]. Available at http://www.ti.com/lit/ds/symlink/ads5474.pdf.

[24] J. A. Tropp and A. C. Gilbert. Signal recovery from random measurements via orthogonal matching pursuit. *IEEE Tran. Info. Theory*, 53(12), 2007.

[25] J. A. Tropp and S. J. Wright. Computational methods for sparse solution of linear inverse problems. *Proc. IEEE*, 98(6):948–958, 2010.

[26] J. Yoo, S. Becker, M. Monge, M. Loh, E. Candès, and A. Emami-Neyestanak. Design and implementation of a fully integrated compressed-sensing signal acquisition system. ICASSP, *to appear*, 2012.