A 17.8-MS/s Compressed Sensing Radar Accelerator Using a Spiking Neural Network

Peter Lawrence Brown, *Student Member, IEEE*, Matthew O'Shaughnessy, *Graduate Student Member, IEEE*, Christopher Rozell, *Senior Member, IEEE*, Justin Romberg, *Fellow, IEEE*, and Michael Flynn, *Fellow, IEEE*

***Abstract*— A prototype compressed sensing radar processor boosts the accuracy of target range and velocity estimations by over 6× compared with conventional processing techniques. The prototype numerically solves basis pursuit denoising with a biologically plausible spiking neural network. A unique form of weight compression allows on-chip storage of all weights for the large fully connected network. Capable of producing over 200 000 range–velocity scene reconstructions per second, the prototype improves throughput by 8× and efficiency by 18× over the state of the art.**

***Index Terms*— Compressed sensing (CS), digital integrated circuits, Doppler radar, radar detection, radar signal processing.**

I. Introduction

In order to maximize resolution, performance-critical radar systems are forced by the Shannon–Nyquist sampling theorem to adopt such high sampling rates and correspondingly high-power transceivers that the issue of data collection, transmission, and storage can become the primary obstacle for system design [1]. State-of-the-art radar systems employ high-frequency transceivers to achieve required precision [2]. Recent work has focused on integrating radar transceivers into small form factors such as those in mobile devices or small appliances, with applications such as vital sign detection, range finding, or indoor positioning [3]–[7]. Interestingly, the design space for such devices is bottlenecked not only by the computation necessary for signal processing but also by the power required for sensing via wireless front ends, such as those used for radar [8]. Front-end power inevitably then becomes the foremost limiting factor in sensing resolution as a consequence of the sampling theorem.

Because of this fundamental limitation and the imbalance between power budgeted to sensing and signal processing, it is usually desirable to enable a lower power transceiver, even if it is at the expense of increased computation. Recent work has shown that radar images can be accurately reconstructed if they are sufficiently sparse in a time–frequency basis, according to the principles of compressed sensing (CS) [9].

This work introduces a prototype radar processor that operates on the principles of CS. In particular, it implements one approach introduced by Herman and Strohmer [10] for CS radar, which assumes a time–frequency target space quantized into discrete atoms in both time and frequency dimensions. The prototype solves the required sparse optimization with a recurrent spiking neural network and exhibits several innovations to address challenges in both enabling a large network size and adapting the network to evolve with complex-valued stimuli.

Manuscript received June 3, 2020; revised August 11, 2020; accepted September 14, 2020. This article was approved by Guest Editor Mark Oude Alink. This work was supported in part by DARPA CHIPS. *(Corresponding author: Peter Lawrence Brown.)*

Peter Lawrence Brown and Michael Flynn are with the Department of Electrical and Computer Engineering, University of Michigan, Ann Arbor, MI 48109 USA (e-mail: peterbrn@umich.edu; mpflynn@umich.edu).

Matthew O'Shaughnessy, Christopher Rozell, and Justin Romberg are with the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332 USA.

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JSSC.2020.3025864

II. Sparse Approximation With a Spiking Neural Network

Given a signal $\boldsymbol{x}$ in some N-dimensional vector space, sparse approximation seeks to represent $\boldsymbol{x}$ as a linear combination of just a few atoms from a dictionary $\Phi = \{\boldsymbol{\phi}_m\}$ (i.e., as a sparse coefficient vector $\boldsymbol{y}$ such that $\boldsymbol{x} \approx \Phi \boldsymbol{y}$). Typically, ensuring sparsity requires the quantity of components in the dictionary Φ to be much larger than the dimension of the signal $\boldsymbol{x}$, i.e., $M \gg N$. Basis pursuit denoising (BPDN) is an unconstrained optimization for sparsely approximating a signal. Formally speaking, BPDN finds the solution $\boldsymbol{y}$ of

$$\arg\min_{\boldsymbol{y}} \frac{1}{2}\|\boldsymbol{x} - \Phi \boldsymbol{y}\|_2^2 + \lambda \|\boldsymbol{y}\|_1. \tag{1}$$

By means of the parameter λ, (1) allows adjusting the approximation to favor either sparsity or reconstruction accuracy. Iterative methods based on orthogonal matching pursuit (OMP) [11] are among the most commonly used to solve BPDN. As an approximate method, OMP can fail to recover signals in certain contexts [12]. However, as BPDN solvers are both difficult to implement and particularly expensive computationally, OMP typically remains the preferred approach.

A. Competitive Evolution for Sparse Approximations

Rozell *et al.* [13] developed techniques to solve classes of optimizations including BPDN, called locally competitive

algorithms (LCAs). One such LCA can be derived by descending on the BPDN objective function

$$E(\boldsymbol{y}) = \frac{1}{2}\|\boldsymbol{x} - \Phi\boldsymbol{y}\|_2^2 + \lambda\|\boldsymbol{y}\|_1. \tag{2}$$

The gradient of this energy function is

$$\frac{\mathrm{d}E}{\mathrm{d}\boldsymbol{y}} = \frac{\mathrm{d}}{\mathrm{d}\boldsymbol{y}}\left[\frac{1}{2}\|\boldsymbol{x} - \Phi\boldsymbol{y}\|_2^2 + \lambda\|\boldsymbol{y}\|_1\right] \tag{3}$$

$$= -\Phi^{\mathsf{T}}\boldsymbol{x} + \left(\Phi^{\mathsf{T}}\Phi\right)\boldsymbol{y} + \lambda\,\mathrm{sign}(\boldsymbol{y}). \tag{4}$$

Define the nonlinear function $T_\lambda(\boldsymbol{y})$ by

$$T_\lambda(\boldsymbol{y}) = \boldsymbol{y} + \lambda\,\mathrm{sign}(\boldsymbol{y}) \tag{5}$$

which forces $\boldsymbol{y}$ to be at least λ units away from zero. We now search for the local minimum of the energy function by descending on its gradient, which then produces an update rule for temporally converging to a sparse representation $\boldsymbol{y}$

$$\frac{\mathrm{d}\boldsymbol{y}}{\mathrm{d}t} = -\frac{\mathrm{d}E}{\mathrm{d}\boldsymbol{y}} \tag{6}$$

$$= \Phi^{\mathsf{T}}\boldsymbol{x} - \left(\Phi^{\mathsf{T}}\Phi - \mathbf{I}\right)\boldsymbol{y} - \boldsymbol{y} - \lambda\,\mathrm{sign}(\boldsymbol{y}) \tag{7}$$

$$= \Phi^{\mathsf{T}}\boldsymbol{x} - T_\lambda(\boldsymbol{y}) - \left(\Phi^{\mathsf{T}}\Phi - \mathbf{I}\right)\boldsymbol{y}. \tag{8}$$

Without loss of generality, assume that $\|\boldsymbol{\phi}_m\| = 1$ for all $\boldsymbol{\phi}_m$ in the dictionary Φ (i.e., the vectors in Φ are normalized), and we can then express the update rule in (8) relative to some individual component y_m of the sparse representation vector $\boldsymbol{y}$

$$\frac{\mathrm{d}y_m}{\mathrm{d}t} = \langle\boldsymbol{\phi}_m, \boldsymbol{x}\rangle - T_\lambda(y_m) - \sum_{k\neq m}\langle\boldsymbol{\phi}_k, \boldsymbol{\phi}_m\rangle y_k. \tag{9}$$

The update rule in (9) illustrates that each component y_m in the sparse representation $\boldsymbol{y}$ evolves according to three stimuli.

1) *Primary Excitation:* Each y_m is excited proportionally to the similarity between (i.e., the magnitude of the inner product of) its associated dictionary element $\boldsymbol{\phi}_m$ and the input signal $\boldsymbol{x}$.
2) *Self-Inhibition:* Each y_m inhibits itself according to its own thresholded value $T_\lambda(y_m) = y_m + \lambda\,\mathrm{sign}(y_m)$. Here, the effect of the sparsity tradeoff parameter λ becomes clear; when λ is large, each y_m is driven to zero with greater "force," regardless of whether y_m is small.
3) *Lateral Inhibition:* Each y_m is inhibited by each of its neighbors $y_k, k \neq m$, proportionally to both the value of y_k and the similarity between their associated dictionary elements $\boldsymbol{\phi}_m$ and $\boldsymbol{\phi}_k$.

It is the components' last stimulus, lateral inhibition, which produces the competition for which LCAs are named; since each component of $\boldsymbol{y}$ inhibits all other components proportionally to its own strength, every component competes with all other components to produce nonzero values. Due to the self-inhibition, however, only a select few manage to do so in steady state.

B. Spiking LCA

Shapero *et al.* [14] converted the LCA to communicate both excitatory and inhibitory signals with discrete spike events. Known as the spiking LCA (S-LCA), the output of S-LCA is the steady-state spike rate of $\boldsymbol{y}$, rather than $\boldsymbol{y}$ itself as in LCA. In S-LCA, the update rule in (9) is modified to produce spikes when the potential y_m exceeds a firing threshold

$$y_m(t) = 0 \quad \text{and emit spike when} \quad y_m(t^-) > 1 \tag{10}$$

where t^- indicates a time immediately prior to t. The remaining dynamics for both self and lateral inhibition of y_m are then adjusted to react to spike impulses rather than the prior thresholded response $T_\lambda(y_m)$

$$\frac{\mathrm{d}y_m}{\mathrm{d}t} = \langle\boldsymbol{\phi}_m, \boldsymbol{x}\rangle - \lambda - \sum_{k\neq m}\langle\boldsymbol{\phi}_k, \boldsymbol{\phi}_m\rangle \sum_{\ell:t_{k\ell}<t}\alpha(t - t_{k\ell}) \tag{11}$$

where $\alpha(t)$ is the unit exponential decay

$$\alpha(t) = \begin{cases} 0, & t < 0 \\ e^{-t}, & t \geq 0 \end{cases} \tag{12}$$

which acts to low-pass filter the spike impulses, and $t_{k\ell}$ is the time of the ℓth spike of y_k. The resulting network has been shown to converge to the behavior of the LCA asymptotically [14]. The differences between S-LCA and LCA are summarized as follows.

1) The state variables y_m emit spike impulses and reset to zero after crossing a firing threshold.
2) The output of the network is the spike emission rate, not $\boldsymbol{y}$ directly. This also means that the output is strictly nonnegative.
3) The self-inhibition is now constant with respect to $\boldsymbol{y}$: λ instead of $T_\lambda(\boldsymbol{y})$.
4) The lateral inhibition is proportional to the low-pass filtered spike stream from other neurons, instead of the value of $\boldsymbol{y}$ itself.

It is in the final point that lies the primary advantage of S-LCA over LCA when considering an implementation strategy. In S-LCA, lateral inhibition between all pairs of y_m state variables continuously changes, requiring on the order of M^2 continually communicated values. On the other hand, S-LCA only requires the communication of lateral inhibition stimuli as a result of discrete spike events, which are likely sparse.

III. Radar Preliminaries

Consider a monostatic (colocated transmitter and receiver) pulse-Doppler radar system. The system transmits a pulse $p(t)$ that is reflected by N targets and received with a matched filter as the combination of all target reflections $r(t) = \sum_{n=1}^{N} r_n(t)$. Each target-reflected pulse $r_n(t)$ is a time-delayed and frequency-modulated version of $p(t)$ according to the range and radial velocity of the target relative to the transmitter, that is,

$$r_n(t) = e^{2\pi i f_n t}\, p(t - \tau_n) \tag{13}$$

where $\tau_n = 2d/c$ (for a target range d) and $f_n \approx -2f_0 v/c$ (for a carrier frequency f_0 and a target radial velocity $v \ll c$) are the time delay and Doppler shift of the pulse reflection, respectively. Classical radar processing correlates the combined reflections $r(t)$ with the transmitted pulse $p(t)$ via the

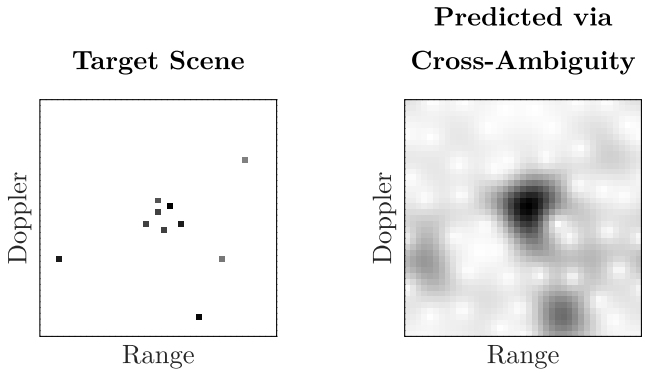


Fig. 1. Classical radar reconstruction of a target scene via cross ambiguity as in (14), which can fail to distinguish targets in close time–frequency proximity.

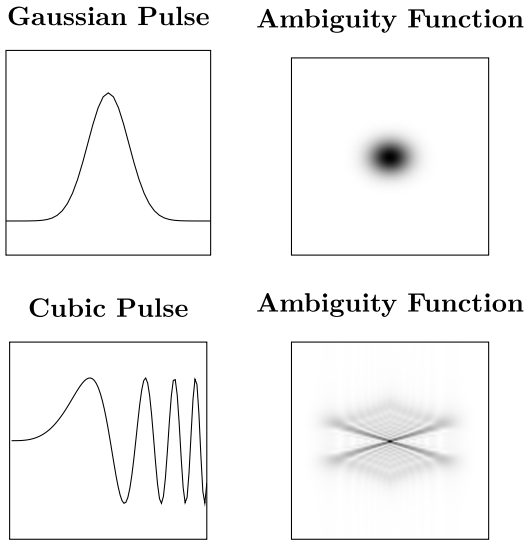


Fig. 2. Visualization of ambiguity functions for Gaussian and cubic phase pulses.

cross-ambiguity function

$$\chi_{rp}(\tau,\nu)=\int_{-\infty}^{\infty} r(t)\overline{p(t-\tau)}e^{2\pi i\nu t}\,\mathrm{d}t. \tag{14}$$

Fig. 1 shows an example of cross-ambiguity reconstruction for a simulated target scene for a Gaussian pulse. The targets clustered in the "center" of the scene are indistinguishable from each other in the reconstruction. This reason for this behavior can be understood from viewing the self-ambiguity of the transmitted pulse (i.e., the cross ambiguity of the pulse $p(t)$ with itself). The support of this ambiguity function describes the areas of uncertainty for range–Doppler estimates. Fig. 2 compares the ambiguity functions for both a Gaussian pulse (as used in the reconstruction in Fig. 1) and a cubic sweeping pulse, which is less coherent. It can be shown that this ambiguity function is volume-invariant for a pulse of given duration and bandwidth [15], that is, assuming that the pulse energy is normalized

$$\int_{-\infty}^{\infty}|p(t)|^2\,\mathrm{d}t=1 \tag{15}$$

the ambiguity function $\chi_p(\tau,\nu)$ achieves a maximum value at $\chi_p(0,0)=1$, and

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\left|\chi_p(\tau,\nu)\right|^2\,\mathrm{d}\tau\,\mathrm{d}\nu=\left|\chi_p(0,0)\right|^2 \tag{16}$$

$$=\left|\int_{-\infty}^{\infty}|p(t)|^2\,\mathrm{d}t\right|^2=1 \tag{17}$$

indicating that the volume under the ambiguity function is constant and independent of the pulse $p(t)$. Together, (15) and (17) imply that attempts to "squeeze" the region of uncertainty about the origin (by adopting a more incoherent pulse) will, by necessity, result in additional volume appearing elsewhere (as shown in the ambiguity function for the cubic pulse in Fig. 2). Thus, the only way to improve the estimation uncertainty of ambiguity reconstructions is to increase the energy of the pulse by increasing either its duration or its bandwidth, and unfortunately both of these measures necessitate an increase in transceiver power. It is desirable to instead reduce uncertainty at the cost of increased data processing, as computation is far cheaper than transmission, and this is the motivation behind CS.

IV. Compressed Sensing Radar

Donoho's 2004 landmark paper [16] on CS ignited a new field in signal processing. CS techniques promise accurate signal reconstruction with sampling rates lower than those normally demanded by the Nyquist–Shannon sampling theorem, if certain system properties (in particular, sparsity in some basis) are known. Within four years, CS techniques were proposed as an alternative estimation technique to the matched filter of traditional radar processing, such as that introduced by Herman and Strohmer [10]

A. Formal Description

The CS radar implemented in this work uses the BPDN ℓ_1 minimization problem to recover a sparse set of targets in range–Doppler space. For this approach, the basis for the sparse approximation is a Gabor frame (set of time delays and frequency shifts) for the transmitted pulse $\boldsymbol{f}$; for a basis of N time shifts and N frequency modulations, $\boldsymbol{\Phi}$ is an $N\times N^2$ dictionary of N-length vectors $\boldsymbol{\phi}_n$, indexed[1] as $\boldsymbol{\Phi}=\{\boldsymbol{\phi}_n\}_{n=0}^{N^2-1}$, where

$$\boldsymbol{T}=\begin{bmatrix}0 & & & 0 & 1\\ 1 & 0 & & & 0\\ 0 & 1 & 0 & & \\ & \ddots & \ddots & \ddots & \\ & & 0 & 1 & 0\end{bmatrix} \tag{18}$$

$$\boldsymbol{M}=\begin{bmatrix}\omega_N^0 & 0 & \cdots & 0\\ 0 & \omega_N^1 & \cdots & 0\\ \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \cdots & \omega_N^{N-1}\end{bmatrix} \tag{19}$$

$$\boldsymbol{\phi}_n=\boldsymbol{M}^{n \bmod N}\boldsymbol{T}^{\lfloor n/N\rfloor}\boldsymbol{f} \tag{20}$$

[1]For ease of notation throughout this section, all matrices and vectors are zero-indexed.

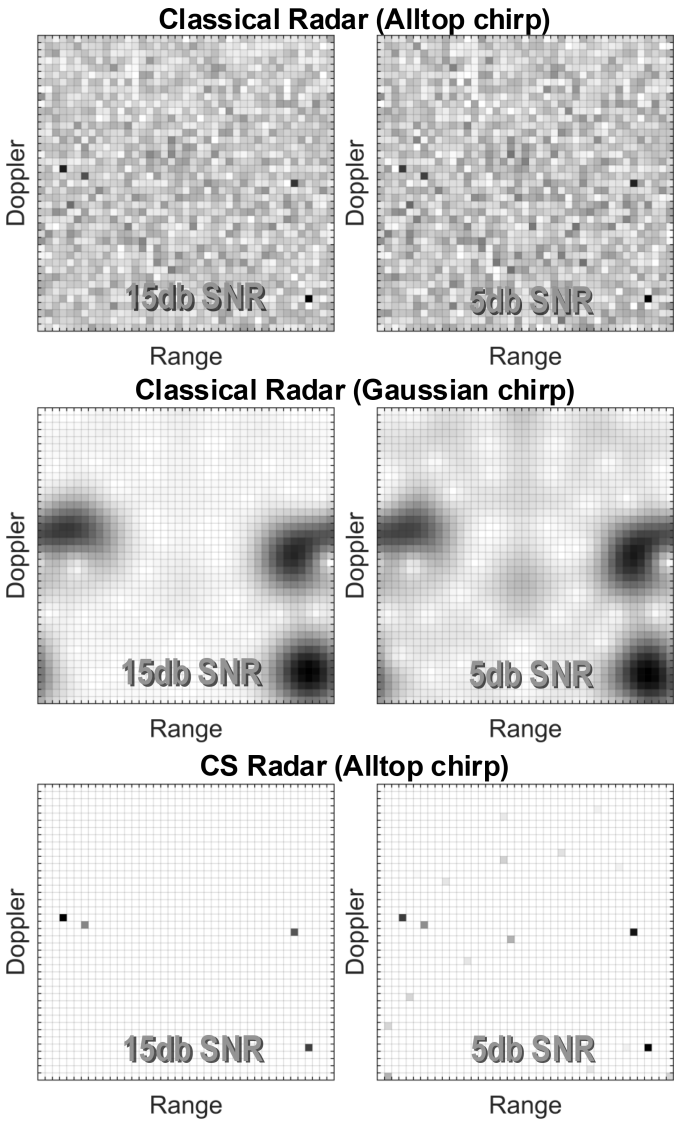


Fig. 3. Classical radar estimations either experience prohibitive interference or avoid interference but cannot identify nearby targets. CS radar's sparse estimations remain precise while avoiding interference.

and $\omega_N = e^{2\pi i/N}$, the Nth root of unity [10]. In [10], the chosen pulse is based on the Alltop sequence [17], denoted $\boldsymbol{f} = \{f_n\}_{n=0}^{N-1}$, where

$$f_n = \frac{1}{\sqrt{N}} e^{2\pi i n^3/N} \tag{21}$$

for $N \geq 5$ prime. This sparse approximation results in a natural filtering of interference that might otherwise result from a highly concentrated ambiguity function. Fig. 3 shows a comparison with traditional radar processing with both Gaussian and Alltop pulses and CS radar with the Alltop pulse. While traditional radar suffers either from imprecise estimation (Gaussian) or prohibitively high interference (Alltop), the CS radar approach is able to isolate the original target vectors without interference. Specifically, for N given as above, CS radar produces an increase in target resolution of $1/\sqrt{N}$ [10].

B. Weight Compression

Due to the required amount of data movement, efficiently realizing the inhibitory connections between neurons is a significant challenge, one that has historically limited the number of neurons in hardware LCA networks [18]. The total number of neurons required to implement S-LCA for CS radar is N^2, associated with combinations of N time shifts and N frequency modulations. Implementing the full complete set of lateral inhibitory connections would then normally require $N^2 \times N^2 = N^4$ weights, an intractable number. However, analyzing the structure of $\boldsymbol{\Phi}^*\boldsymbol{\Phi}$ (where $[\cdot]^*$ denotes the Hermitian transpose) reveals a method of weight compression. Note that $\boldsymbol{\Phi}$ can then be expressed as

$$\boldsymbol{\Phi} = \begin{bmatrix} \boldsymbol{\Phi}^{(0)} & \boldsymbol{\Phi}^{(1)} & \cdots & \boldsymbol{\Phi}^{(N-1)} \end{bmatrix} \tag{22}$$

where the $N \times N$ submatrices $\boldsymbol{\Phi}^{(n)}$ are defined as

$$\boldsymbol{\Phi}^{(n)} = \boldsymbol{D}_n \boldsymbol{W}_N \tag{23}$$

where $\boldsymbol{D}_n$ are the n time-shift matrix operators applied to the pulse sequence $\boldsymbol{f}$

$$\boldsymbol{D}_n = \begin{bmatrix} f_n & 0 & \cdots & 0 \\ 0 & f_{n+1} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & f_{n-1} \end{bmatrix} \tag{24}$$

and $\boldsymbol{W}_N$ contains all combinations of frequency shifts

$$\boldsymbol{W}_N = \left(\omega_N^{pq}\right)_{p,q=0}^{N-1} = \begin{bmatrix} \omega_N^0 & \omega_N^0 & \cdots & \omega_N^0 \\ \omega_N^0 & \omega_N^1 & \cdots & \omega_N^{N-1} \\ \vdots & \vdots & \ddots & \vdots \\ \omega_N^0 & \omega_N^{N-1} & \cdots & \omega_N^{(N-1)^2} \end{bmatrix}. \tag{25}$$

Now, when considering their construction from the Alltop sequence $\boldsymbol{f}$ defined in (21), any two submatrices $\boldsymbol{\Phi}^{(m)}$ and $\boldsymbol{\Phi}^{(n)}$ for $m \neq n$ have the property that, for any $0 \leq j, k < N$, with $\boldsymbol{\phi}_j^{(m)}$ denoting the jth column vector of the mth $N \times N$ submatrix $\boldsymbol{\Phi}^{(m)}$, the cross correlation between $\boldsymbol{\phi}_j^{(m)}$ and $\boldsymbol{\phi}_k^{(n)}$ is of constant magnitude

$$\left|\left\langle \boldsymbol{\phi}_j^{(m)}, \boldsymbol{\phi}_k^{(n)} \right\rangle\right| = \left| \sum_{\ell=0}^{N-1} \left(\omega_N^{j\ell} f_{\ell+m \bmod N}\right)^* \left(\omega_N^{k\ell} f_{\ell+n \bmod N}\right) \right| \tag{26}$$

$$= \frac{1}{N} \left| \sum_{\ell=0}^{N-1} \omega_N^{\left(k-j+3n^2-3m^2\right)\ell+3(m-n)\ell^2} \right|. \tag{27}$$

The summation is in the form of a quadratic Gauss sum and will thus always have magnitude $\sqrt{N}$

$$= \frac{1}{N}\sqrt{N} = \frac{1}{\sqrt{N}}. \tag{28}$$

The fact that the resulting magnitudes in (28) are constant is a compelling incentive for adopting the Alltop sequence from (21). It also provides a crucial behavior that eventually enables weight compression, allowing the large network size required by CS radar to be practically realizable. Now, using (22) to

develop an expression of the $N^2 \times N^2$ matrix $\mathbf{\Phi}^*\mathbf{\Phi}$, its structure can be analyzed in terms of the $N \times N$ submatrices $\mathbf{\Phi}^{(i)*}\mathbf{\Phi}^{(j)}$

$$\mathbf{\Phi}^*\mathbf{\Phi} = \begin{bmatrix} \mathbf{\Phi}^{(0)*} \\ \mathbf{\Phi}^{(1)*} \\ \vdots \\ \mathbf{\Phi}^{(N-1)*} \end{bmatrix} \begin{bmatrix} \mathbf{\Phi}^{(0)} & \mathbf{\Phi}^{(1)} & \dots & \mathbf{\Phi}^{(N-1)} \end{bmatrix} \tag{29}$$

$$= \begin{bmatrix} \mathbf{\Phi}^{(0)*}\mathbf{\Phi}^{(0)} & \dots & \mathbf{\Phi}^{(0)*}\mathbf{\Phi}^{(N-1)} \\ \mathbf{\Phi}^{(1)*}\mathbf{\Phi}^{(0)} & \dots & \mathbf{\Phi}^{(1)*}\mathbf{\Phi}^{(N-1)} \\ \vdots & & \vdots \\ \mathbf{\Phi}^{(N-1)*}\mathbf{\Phi}^{(0)} & \dots & \mathbf{\Phi}^{(N-1)*}\mathbf{\Phi}^{(N-1)} \end{bmatrix}. \tag{30}$$

Consider the structure of a single submatrix $\mathbf{\Phi}^{(i)*}\mathbf{\Phi}^{(j)}$. From (23), we have that element (p, q) of the submatrix $\mathbf{\Phi}^{(i)*}\mathbf{\Phi}^{(j)}$ will be

$$\left(\mathbf{\Phi}^{(i)*}\mathbf{\Phi}^{(j)}\right)_{pq} = \left((\boldsymbol{D}_i \boldsymbol{W}_N)^* \boldsymbol{D}_j \boldsymbol{W}_N\right)_{pq} \tag{31}$$

$$= \left(\boldsymbol{W}_N^* \boldsymbol{D}_i^* \boldsymbol{D}_j \boldsymbol{W}_N\right)_{pq}. \tag{32}$$

The product $\boldsymbol{D}_i^* \boldsymbol{D}_j$ is diagonal and, from (28), of constant magnitude $1/N$. Define e_k as $\left(\boldsymbol{D}_i^* \boldsymbol{D}_j\right)_{kk}$, the kth element of the product's diagonal

$$= \sum_{k=0}^{N-1} e_k \omega_N^{(q-p)k}. \tag{33}$$

Note that (33) depends only on the expression $(q - p) \bmod N$. This implies that the values along all wrapped forward diagonals of each $\mathbf{\Phi}^{(i)*}\mathbf{\Phi}^{(j)}$ submatrix are constant, which means that only a single row or column needs to be stored for each submatrix. This reduces the weight storage requirement for all submatrices from N^2 to N and the total weight storage from N^4 to N^3. However, additional logic is required for weight access to shift the weights back to the appropriate row or column; a variable rotator, operating on the requested weight row or column index, is enough for decompression.

C. Adapting S-LCA Dynamics to Complex Numbers

CS radar operates on digitized pulse reflections, which are transmitted and received through a front end as I/Q real/imaginary values. As S-LCA is by nature limited to solving sparse approximations with unsigned component solutions, it must therefore be adapted to allow for optimization under the complex numbers before it can be used for reconstructing range–Doppler estimations for CS radar. Two steps are necessary for this transition: adapting S-LCA to produce signed values and extending S-LCA to optimize under the complex numbers.

Prior works have typically doubled the number of neurons in a network to enable signed operation, allocating half of the network to positive evolution and half to negative [19], [20]. Alternatively, adapting S-LCA to produce signed values can be accomplished by modifying the neurons' spike emissions to be polarized. A negative spiking threshold is added to the dynamics described by (10) so that neurons emit polarized spikes. Each spike then carries an additional bit of polarity information to indicate whether it was emitted as a result of breaking the positive or negative threshold. This approach avoids the cost of doubling the network size (and correspondingly quadrupling the number of lateral weights) as in prior approaches. To our knowledge, this approach had never been put to practical use but has been hinted as a possibility [21].

After enabling signed dynamics, the next required step is to extend S-LCA to optimize under the complex numbers. This is accomplished by transforming the problem into a real-valued optimization. Each complex-valued input associates with two real-valued inputs, one each for real and imaginary components, which doubles the number of feedforward weights elements in $\mathbf{\Phi}$ from (22). The extended $\mathbf{\Phi}$, denoted $\mathbf{\Phi}_{\text{ext}}$, is defined as

$$\mathbf{\Phi}_{\text{ext}} = \begin{bmatrix} \text{real}(\mathbf{\Phi}) \\ \text{imag}(\mathbf{\Phi}) \end{bmatrix}. \tag{34}$$

Thus, the dimensions of $\mathbf{\Phi}_{\text{ext}}$ are $2N \times N^2$, doubled from those of $\mathbf{\Phi}$ at $N \times N^2$. However, despite doubling the quantity of weights stored, the total feedforward weight storage required remains unchanged because the number of components in each weight is simultaneously halved (one for real values instead of two for complex values). In addition, the lateral weight storage requirement remains unchanged, as the dimensions and type of $\mathbf{\Phi}_{\text{ext}}^{\mathsf{T}}\mathbf{\Phi}_{\text{ext}}$ remain the same at $N^2 \times N^2$ real values (that is, the conversion does not increase the number of neurons). However, it must be shown that the weight compression property presented in Section IV-B still holds when the above extended $\mathbf{\Phi}_{\text{ext}}$ is used to construct a new lateral weight matrix $\mathbf{\Phi}_{\text{ext}}^{\mathsf{T}}\mathbf{\Phi}_{\text{ext}}$. This is evident from a straightforward extension of the fact that for any complex number $z = a + bi$, we have that $\overline{z}z = a^2 + b^2$, which means that the lateral weights remain unchanged, that is,

$$\mathbf{\Phi}_{\text{ext}}^{\mathsf{T}}\mathbf{\Phi}_{\text{ext}} = \begin{bmatrix} \text{real}(\mathbf{\Phi}^{\mathsf{T}}) & \text{imag}(\mathbf{\Phi}^{\mathsf{T}}) \end{bmatrix} \begin{bmatrix} \text{real}(\mathbf{\Phi}) \\ \text{imag}(\mathbf{\Phi}) \end{bmatrix} \tag{35}$$

$$= \text{real}(\mathbf{\Phi}^{\mathsf{T}})\,\text{real}(\mathbf{\Phi}) + \text{imag}(\mathbf{\Phi}^{\mathsf{T}})\,\text{imag}(\mathbf{\Phi}) \tag{36}$$

$$= \mathbf{\Phi}^*\mathbf{\Phi} \tag{37}$$

as desired. Thus, S-LCA can emulate evolving over the complex numbers without sacrificing the beneficial properties gained from the Alltop sequence.

V. Hardware Implementation

In practice, rather than attempt exact BPDN optimization, implementations instead substitute iterative greedy methods, such as OMP [22] and greedy gradient or coordinate descent [23]. These approximations to BPDN enable realizable hardware implementations but do not always produce optimal solutions [24]. This work demonstrates the first hardware neural network BPDN solver for CS radar. Leveraging highly parallelized neuron computation and synapse weight storage optimization, the solver simultaneously improves processing throughput by more than $8\times$ and efficiency by more than $18\times$ over existing hardware efforts [18], [22], [23]. The prototype employs a time–frequency resolution of $N = 41$, resulting in a resolution improvement of $\sqrt{41} \approx 6\times$ over traditional radar.

The immediate issue concerning the implementation is the storage and distribution of the weights representing interneuron inhibitory connections. As the prototype S-LCA optimizer

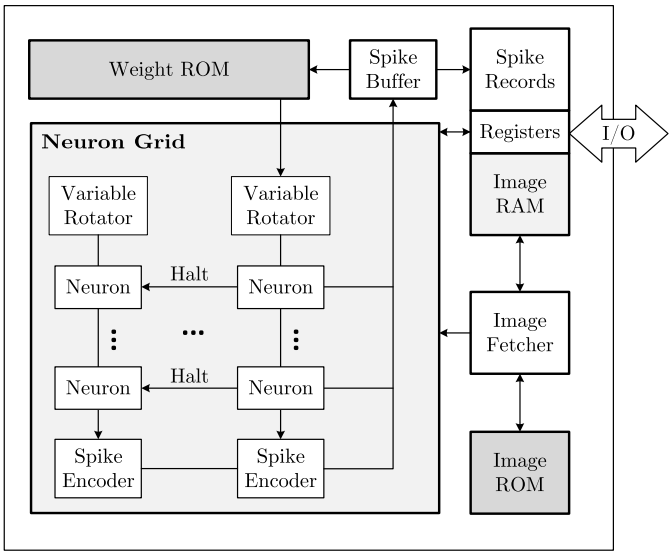


Fig. 4. Hierarchical summary of final implemented CS radar processor prototype architecture, showing the synchronization ("halting") bus, unified memory interface, weight compression (variable rotator), and hybrid RAM/ROM image selection.

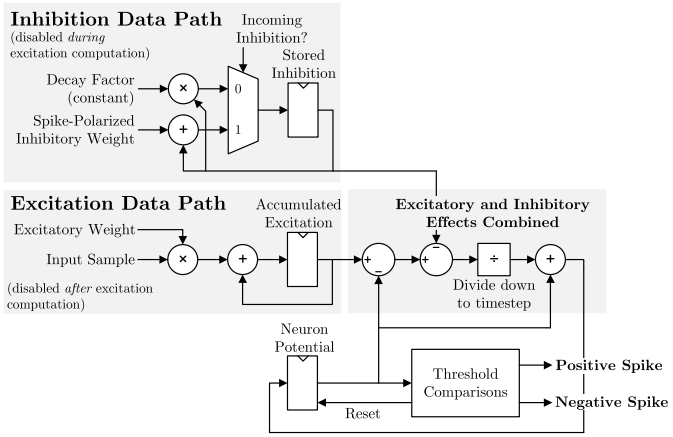


Fig. 5. Functional diagram of a dual-phase, dual-polarity LIF neuron for CS radar.

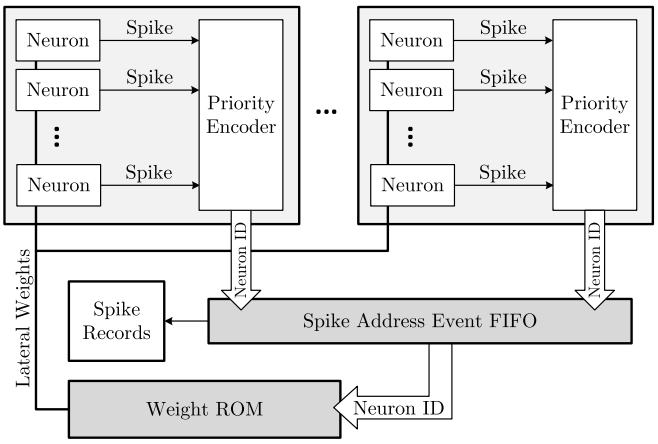


Fig. 6. Logic for encoding simultaneous spike events.

employs a grid quantization of $N = 41$, S-LCA optimization involves the simultaneous evolution of a total of $N^2 = 1681$ neurons. Normally, this fully connected layer would require the storage and distribution of $N^2 \times N^2 \approx 2.8$ million inhibitory weights, but as a result of weight compression, only $N^3 \approx 69$ thousand weights need to be stored, with minimal additional weight retrieval logic (see Section IV-B). The weights for the entire network (including $2 \times N^2 \approx 138$ thousand input layer connections) can then fit inside less than 1 Mbit of compact ROM.

A summary of the realized structure of the accelerator is presented in Fig. 4. This section discusses some of the challenges experienced and the techniques used to overcome them.

A. Dual-Phase Operation

Feedforward excitatory connections from input nodes must be distributed to neurons as the network evolves. The strength of each connection is proportional to a vector dot product of the pulse reflection and a time–frequency shifted pulse. As this product is constant for a given optimization, the network divides each optimization into two phases, as shown in Fig. 5, which illustrates the implementation of a leaky integrate-and-fire (LIF) neuron with dual-phase operation.

1) Precompute and store the excitation factor for each neuron.
2) Excite the neurons using their stored excitation values and allow the competitive network to evolve to steady state.

B. Simultaneous Spike Events

Any of the 1681 neurons can potentially spike within any iteration during network evolution, necessitating some means of efficiently detecting, encoding, and ordering spike events. To reduce bus collisions, each column of the neuron grid has its own spike bus with an attached priority encoder, as shown in Fig. 6. The priority encoder records only one spike and discards the rest. While this decision, in the case of this prototype, has minimal impact on network evolution due to the unlikelihood of more than one simultaneous spike within the same column, future work could incorporate a buffering scheme instead of prioritization for maximal synaptic integrity. Each priority encoder then reports the identifier of the spiking neuron to a common spike address event (AE) FIFO, which buffers a limited number of spike AEs. The FIFO then reports one spike AE at a time to both the spike records and the lateral weight retrieval module. The spike records maintain a history of all neurons that spiked during the evolution. After the network evolves to steady state, this information then directly indicates the time–frequency offsets in the received signal. Lateral weight retrieval involves fetching and decompressing (by means of a variable rotator, as described in Section IV-B) the lateral weights associated with the spiking neuron from ROM and distributing them to all neurons in the grid.

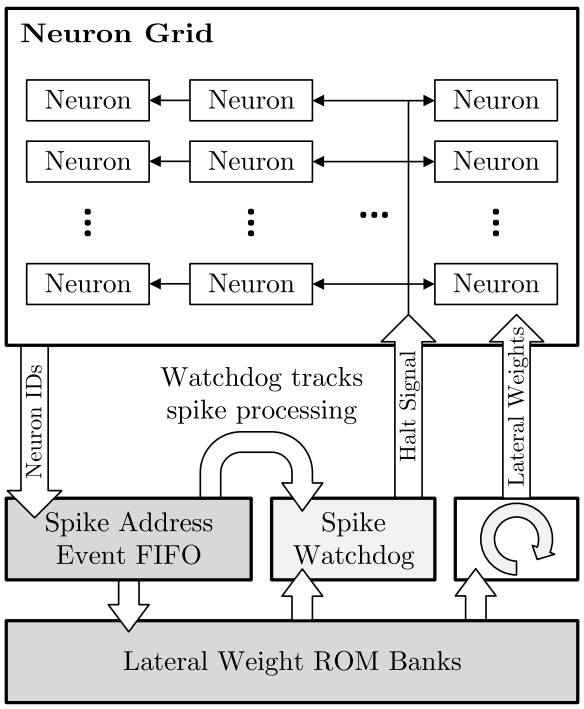


Fig. 7. Watchdog module tracks spike processing and distributes a halting signal to the neuron grid to synchronize evolution with spike propagation. This has the effect of functionally eliminating spike propagation delay.

C. Synchronization Bus

Timely and deterministic synaptic response between neurons is enabled in the fully connected layer with a synchronization bus to control network halt. This ensures that the neurons evolve as though inhibited instantly, which improves the determinism of the network evolution and functionally eliminates delays associated with weight retrieval and routing logic (see Fig. 7). The synchronization bus is governed by a watchdog module that tracks both the contents of the spike AE FIFO and the current fetching progress from lateral weight ROM banks. The network remains halted until all outstanding spike AEs have been processed and all inhibitory weights distributed to the network.

The increased determinism resulting from synchronization allows both shortening the inference duration by more than $4\times$ and minimizing the computational precision of neuron dynamics by approximately 30%. Fig. 8 shows a comparison of reconstruction error for two networks, one with a synchronization bus and one without, as they evolve to a particular solution, demonstrating that the synchronized network converges much more quickly than the unsynchronized network. Furthermore, synchronization allowed for effective evolution even with low arithmetic precision; both neuron feedforward weights and interneuron inhibitory weights require only 4-bit precision to achieve acceptable accuracy, while neuron potential and filtered inhibition are stored with 12- and 11-bit precision, respectively.

D. Weight Memory Layout

As described in Section V-A, the network operates in two phases: excitation precomputation and competitive

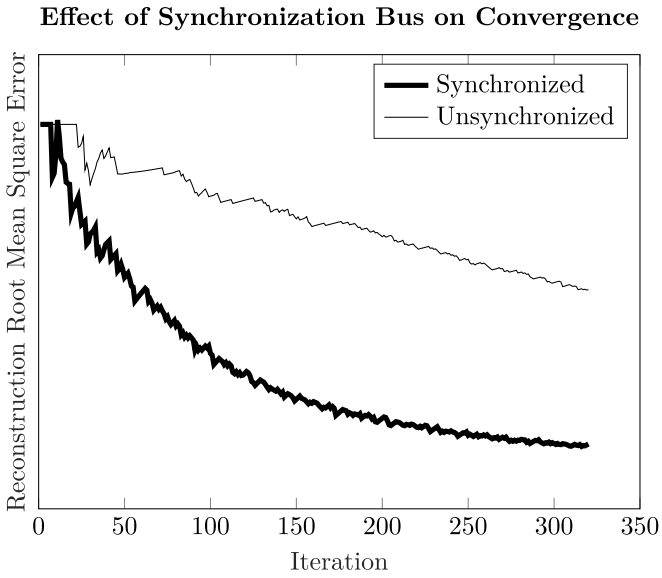


Fig. 8. Network synchronization enables much faster solution convergence.

(i.e., inhibitory) evolution to steady state. In both phases, access to weight memory is mutually exclusive. While precomputing neurons' excitatory magnitudes, the network fetches image samples and feedforward weights. During competitive evolution, neuron spike events trigger lateral weight retrieval and distribution. Because both feedforward and lateral weights are never accessed simultaneously (excitation does not require lateral weights and inhibitory evolution does not require feedforward weights), they are stored in the same ROM banks, saving periphery area.

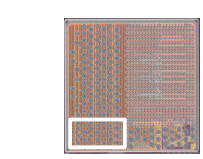
Pulse reflections are stored in a single SRAM bank. In addition, a ROM bank of predefined reflections exists for debugging purposes and can be selected at runtime. These image memory banks contain 1024 words of 82 bits each, requiring four read operations to fetch the 82 4-bit samples of a given reflection and allowing up to 256 reflections to be programmed at a time.

During the first phase, each sample of the loaded pulse reflection must be multiplied by the corresponding 4-bit feedforward weight for each of the 1681 neurons, which requires fetching a total of 6724 bits from the weight ROM. To balance between cycle time and cell area, the weight ROM banks have a word size of 160 bits, and 11 banks operate in parallel. This allows for a total of 1760 bits to be fetched each cycle so that the full 6724 bits can be fetched in four cycles. As a result, this memory access strategy fixes the duration of the first phase at 1024 cycles.

During the second phase, as the lateral weights are stored with the same precision (4 bits) as the feedforward weights, the same ROM banks are reused to store those weights for each combination of neuron and time delay. When a neuron spikes during network evolution, 1681 4-bit weights, again for a total of 6724 bits, are fetched, decompressed, and distributed with the same four-cycle access strategy.

VI. Measurement Results

Fabricated in 16-nm FinFET CMOS, the prototype die micrograph and summary are shown in Fig. 9. To evaluate



Technology	16nm FinFET
Core Area	0.6 mm^2
Memory	RAM: 84 Kbit
	ROM: 985 Kbit
Nominal Frequency	500 MHz
Power Consumption	3.6 to 278 mW
Throughput	17.8 Msamples/s

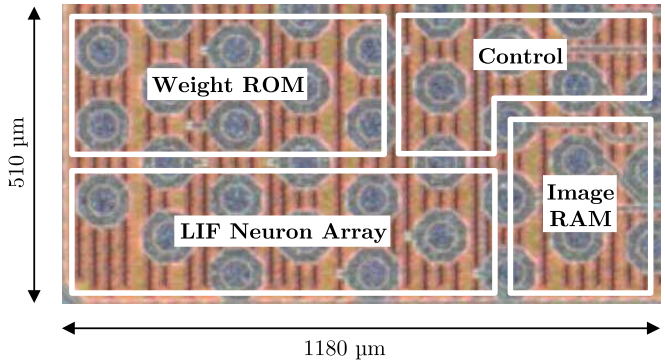


Fig. 9. Die photograph and summary table of CS radar processor prototype.

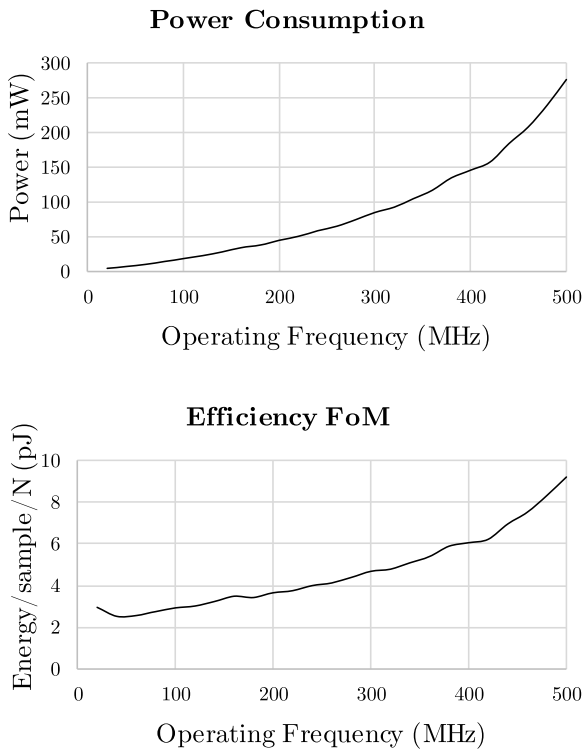


Fig. 10. CS radar processor prototype measured power consumption and computational efficiency.

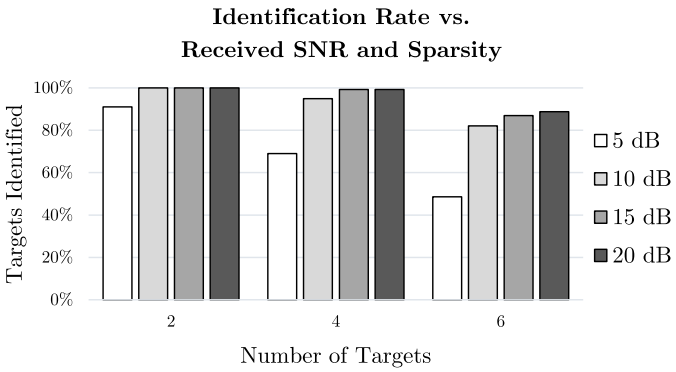


Fig. 11. CS radar processor prototype measured identification rate for varying target sparsity and pulse reflection SNR.

TABLE I
COMPARISON WITH STATE-OF-THE-ART CS PROCESSORS

Design	**ISSCC '15 [22]**	**ISSCC '18 [23]**	**This Work**
Technology	40nm	40nm	16nm FinFET
Application	Biomedical	Biomedical	Radar
Algorithm	OMP, K-OMP	SE-SP	S-LCA
Dictionary size	up to 1024	up to 2048	1681
Measurement dim.	up to 512	up to 512	82
Core Area (mm^2)	5.13	3.06[1]	0.6
On-chip memory (Kb)	1176	$6144 + 1536$	$985 + 84$
Power (mW)	8.6 to 78	21.8 to 93	3.57 to 278
Clock (MHz)	27.4	67.5	20 to 500
Throughput (MS/s)[2]	0.012 to 0.237	0.232 to 1.996	0.714 to 17.85
Efficiency FoM[3]	211[4]	43.8[4]	2.54
Efficiency FoM scaled[5]	36.4	7.50	2.54

[1]Does not include 768 KB of on-chip memory.
[2]Measured at minimum energy point, normalized by dictionary size for the optimization. Dictionary size is [22] 256 and [23] 384.
[3]For fair comparison between optimizing for different dictionary sizes, equivalent to pJ/(sample · signal dimension). Lower is better.
[4]Best FoM taken from tests at minimum energy point.
[5]Accounts for expected energy scaling at given process node (all scaled to 16nm). [25]

the prototype processor, a test computer communicated read and write requests to a field-programmable gate array (FPGA), which provided to the prototype both signaling for the memory interface (all registers and RAM are mapped to memory addresses) and the core clock signal.

Simulated reflections received by a monostatic radar system in an environment with moving targets are digitized under varying noise conditions and provided to the protoype as I/Q values, as if produced by an analog front end and analog-to-digital converter. The prototype then processes the reflection and produces estimates of distance and radial velocities for all identified targets.

Fig. 10 shows a summary of measured computational efficiency. The top graph shows the total power consumption over a standard range of operating frequencies; the prototype consumes 3.57 mW at 20 MHz and 278 mW at 500 MHz. The bottom graph identifies the computational efficiency for the same range of operating frequencies, where efficiency is measured via a figure of merit defined as the energy consumed to process a single input sample, normalized by the "signal dimension" for the processor. Within the context of sparse approximation problems, the signal dimension is the number of elements in the dictionary of atoms. This normalizing factor

is added to more accurately compare against other sparse approximation accelerators, as most algorithms for solving the required optimization (greedy or otherwise) increase at least linearly in runtime with the signal dimension. For this prototype, the minimum energy point occurs at an operating frequency of 40 MHz.

To evaluate target identification performance, the prototype is given a digital representation of a received radar pulse with a given SNR. As shown in Fig. 11, when the targets are sufficiently sparse, the processor locates more than 99% of all targets even at low-to-moderate SNR conditions.

Table I shows a comparison with prior state-of-the-art approaches in CS hardware.

VII. Conclusion

This work demonstrates the first CS radar processor with a non-greedy sparse optimizer, enabling radar applications to leverage the greater accuracy of CS to reduce front-end power requirements and making ultra-low-power smart-home and mobile radar sensors realizable. The optimizer combines S-LCA, a biologically inspired spiking neural network model, with a novel weight compression scheme in an architectural approach that enables both high efficiency and low-latency processing. The prototype operates at up to 500 MHz and 218k estimates per second, with a peak efficiency of 2.4 pJ per sample per dictionary atom, demonstrating a sample throughput improvement of more than 8× and efficiency of more than 18× over prior work [22], [23].

References

[1] J. H. G. Ender, "On compressive sensing applied to radar," *Signal Process.*, vol. 90, no. 5, pp. 1402–1414, May 2010.

[2] D. Guermandi *et al.*, "A 79-GHz 2 × 2 MIMO PMCW radar SoC in 28-nm CMOS," *IEEE J. Solid-State Circuits*, vol. 52, no. 10, pp. 2613–2626, Jul. 2017.

[3] C. Li, X. Yu, C.-M. Lee, D. Li, L. Ran, and J. Lin, "High-sensitivity software-configurable 5.8-GHz radar sensor receiver chip in 0.13-μm CMOS for noncontact vital sign detection," *IEEE Trans. Microw. Theory Techn.*, vol. 58, no. 5, pp. 1410–1419, May 2010.

[4] D. Zito, D. Pepe, M. Mincica, and F. Zito, "A 90nm CMOS SoC UWB pulse radar for respiratory rate monitoring," in *IEEE ISSCC Dig. Tech. Papers*, Feb. 2011, pp. 40–41.

[5] T.-Y.-J. Kao, A. Y.-K. Chen, Y. Yan, T.-M. Shen, and J. Lin, "A flip-chip-packaged and fully integrated 60 GHz CMOS micro-radar sensor for heartbeat and mechanical vibration detections," in *Proc. IEEE Radio Freq. Integr. Circuits Symp.*, Jun. 2012, pp. 443–446.

[6] M. K. Kang and T. W. Kim, "CMOS IR-UWB receiver for ±9.7-mm range finding in a multipath environment," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 59, no. 9, pp. 538–542, Sep. 2012.

[7] H. G. Han, B. G. Yu, and T. W. Kim, "19.6 a 1.9 mm-precision 20GS/S real-time sampling receiver using time-extension method for indoor localization," in *IEEE ISSCC Dig. Tech. Papers*, Feb. 2015, pp. 1–3.

[8] B. Martinez, M. Monton, I. Vilajosana, and J. D. Prades, "The power of models: Modeling power consumption for IoT devices," *IEEE Sensors J.*, vol. 15, no. 10, pp. 5777–5789, Oct. 2015.

[9] L. C. Potter, E. Ertin, J. T. Parker, and M. Cetin, "Sparsity and compressed sensing in radar imaging," *Proc. IEEE*, vol. 98, no. 6, pp. 1006–1020, Jun. 2010.

[10] M. A. Herman and T. Strohmer, "High-resolution radar via compressed sensing," *IEEE Trans. Signal Process.*, vol. 57, no. 6, pp. 2275–2284, Jun. 2009.

[11] Y. C. Pati, R. Rezaiifar, and P. S. Krishnaprasad, "Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition," in *Proc. 27th Asilomar Conf. Signals, Syst. Comput.*, Nov. 1993, pp. 40–44.

[12] J. A. Tropp and A. C. Gilbert, "Signal recovery from random measurements via orthogonal matching pursuit," *IEEE Trans. Inf. Theory*, vol. 53, no. 12, pp. 4655–4666, Dec. 2007.

[13] C. J. Rozell, D. H. Johnson, R. G. Baraniuk, and B. A. Olshausen, "Sparse coding via thresholding and local competition in neural circuits," *Neural Comput.*, vol. 20, no. 10, pp. 2526–2563, Oct. 2008.

[14] S. Shapero, M. Zhu, J. Hasler, and C. Rozell, "Optimal sparse approximation with integrate and fire neurons," *Int. J. Neural Syst.*, vol. 24, no. 5, Aug. 2014, Art. no. 1440001.

[15] N. Levanon and E. Mozeson, *Radar Signals*. Hoboken, NJ, USA: Wiley, 2004.

[16] D. L. Donoho, "Compressed sensing," *IEEE Trans. Inf. Theory*, vol. 52, no. 4, pp. 1289–1306, Apr. 2006.

[17] W. Alltop, "Complex sequences with low periodic correlations (Corresp.)," *IEEE Trans. Inf. Theory*, vol. 26, no. 3, pp. 350–354, May 1980.

[18] J. K. Kim, P. Knag, T. Chen, and Z. Zhang, "A 640M pixel/s 3.65 mW sparse event-driven neuromorphic object recognition processor with on-chip learning," in *Proc. Symp. VLSI Circuits (VLSI Circuits)*, Jun. 2015. pp. C50–C51.

[19] S. Shapero, A. S. Charles, C. J. Rozell, and P. Hasler, "Low power sparse approximation on reconfigurable analog hardware," *IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 2, no. 3, pp. 530–541, Sep. 2012.

[20] S. Shapero, C. Rozell, A. Balavoine, and P. Hasler, "A scalable implementation of sparse approximation on a field programmable analog array," in *Proc. IEEE Biomed. Circuits Syst. Conf. (BioCAS)*, Nov. 2011, pp. 141–144.

[21] A. S. Charles, P. Garrigues, and C. J. Rozell, "Analog sparse approximation with applications to compressed sensing," 2011, *arXiv:1111.4118*. [Online]. Available: https://arxiv.org/abs/1111.4118

[22] F. Ren and D. Markovic, "18.5 a configurable 12-to-237KS/s 12.8 mW sparse-approximation engine for mobile ExG data aggregation," in *IEEE ISSCC Dig. Tech. Papers*, Feb. 2015, pp. 1–3.

[23] T.-S. Chen, H.-C. Kuo, and A.-Y. Wu, "A 232-to-1996KS/s robust compressive-sensing reconstruction engine for real-time physiological signals monitoring," in *IEEE ISSCC Dig. Tech. Papers*, Feb. 2018, pp. 226–228.

[24] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM Rev.*, vol. 43, no. 1, pp. 129–159, Jan. 2001.

[25] A. Stillmaker and B. Baas, "Scaling equations for the accurate prediction of CMOS device performance from 180 nm to 7 nm," *Integration*, vol. 58, pp. 74–81, Jun. 2017.

Peter Lawrence Brown (Student Member, IEEE) received the B.S.E.E. degree in electrical engineering from the University of Idaho, Moscow, ID, USA, in 2015, and the M.S. and Ph.D. degrees in electrical and computer engineering from the University of Michigan, Ann Arbor, MI, USA, in 2016 and 2020, respectively.

Since 2020, he has been a Design Engineer in emerging memory with Micron Technology, Inc., Boise, ID, USA.

Matthew O'Shaughnessy (Graduate Student Member, IEEE) received the B.S. degree in electrical engineering and the M.S. degree in mathematics from the Georgia Institute of Technology, Atlanta, GA, USA, in 2016 and 2019, respectively, where he is currently pursuing the Ph.D. degree.

His research interests include causal inference, low-dimensional structure, and the intersection of technology and public policy.

Christopher Rozell (Senior Member, IEEE) received the B.S.E. degree in computer engineering and the B.F.A. degree in performing arts technology (music technology) from the University of Michigan, Ann Arbor, MI, USA, in 2000, and the M.S. and Ph.D. degrees in electrical engineering from Rice University, Houston, TX, USA, in 2002 and 2007, respectively. He attended the Graduate School at Rice University.

He was a Texas Instruments Distinguished Graduate Fellow with Rice University. Following graduate school, he was a Post-Doctoral Research Fellow with Redwood Center for Theoretical Neuroscience, University of California at Berkeley, Berkeley, CA, USA. In July 2008, he joined Georgia Tech, Atlanta, GA, USA, as a Faculty Member, where he is currently a Professor with the School of Electrical and Computer Engineering.

Dr. Rozell received the Scholar Award in Studying Complex Systems from the James S. McDonnell Foundation 21st Century Science Initiative and the National Science Foundation CAREER Award in 2014. He also serves as an Associate Editor for the IEEE OPEN JOURNAL OF SIGNAL PROCESSING.

Justin Romberg (Fellow, IEEE) received the B.S.E.E., M.S., and Ph.D. degrees from Rice University, Houston, TX, USA, in 1997, 1999, and 2004, respectively.

From Fall 2003 until Fall 2006, he was a Post-Doctoral Scholar in applied and computational mathematics with the California Institute of Technology, Pasadena, CA, USA. He is currently the Schlumberger Professor with the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA, USA, where he has been on the faculty since 2006. His research interests lie at the intersection of statistical signal processing, machine learning, optimization, and applied probability.

Michael Flynn (Fellow, IEEE) received the Ph.D. degree from Carnegie Mellon University, Pittsburgh, PA, USA, in 1995.

From 1995 to 1997, he was a Member of Technical Staff with Texas Instruments, Dallas, TX, USA. From 1997 to 2001, he was with Parthus Technologies, Cork, Ireland. He joined the University of Michigan, Ann Arbor, MI, USA, in 2001, where he is currently a Professor. His technical interests are in RF circuits, data conversion, serial transceivers, and biomedical systems.

Dr. Flynn is a 2008 Guggenheim Fellow. He received the 2016 University of Michigan Faculty Achievement Award, the 2011 Education Excellence Award and the 2010 College of Engineering Ted Kennedy Family Team Excellence Award from the College from Engineering, University of Michigan, the 2005–2006 Outstanding Achievement Award from the Department of Electrical Engineering and Computer Science, University of Michigan, and the NSF Early Career Award in 2004. He is also the Chair of the Data Conversion Committee of the International Solid-State Circuits Conference. He formerly served on the Technical Program Committee of ESSCIRC, IEEE Asian Solid-State Circuits Conference (A-SSCC), and the Symposium on VLSI Circuits. He is a former Distinguished Lecturer of the IEEE Solid-State Circuits Society. He has served as an Associate Editor for the IEEE JSSC and the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS. He was the Editor-in-Chief of the IEEE JOURNAL OF SOLID-STATE CIRCUITS from 2013 to 2016.