Hardware-Efficient and Short Sensing-Time Multicoset-Sampling Based Wideband Spectrum Sensor for Cognitive Radio Network

Rahul Sharma, *Graduate Student Member, IEEE*, Rahul Shrestha, *Senior Member, IEEE*, and Satinder K. Sharma, *Senior Member, IEEE*

***Abstract*— This work proposes implementation friendly algorithm for multicoset sampling based wideband spectrum sensing that alleviates computational space and enables parallel execution, incurring lower latency. Based on this proposed algorithm, we provide a new hardware-efficient VLSI architecture of wideband spectrum sensor (WSSR), which offers short sensing time while sensing the wideband spectrum. Additionally, this paper presents a comprehensive discussion of all the submodule micro-architectures of the proposed WSSR. Subsequently, extensive performance analyses performed in the AWGN channel environment have demonstrated that our WSSR delivers adequate detection probability of 0.9 at -5 dB of SNR. Furthermore, the proposed WSSR design also uses a Zynq UltraScale+ FPGA board with a 14.16μs sensing time and a 2.63 GHz maximum sensing bandwidth. Comparison of our hardware implementation results has shown that the proposed WSSR achieves 38.5% higher sensing bandwidth and 90% shorter sensing time, in comparison to the state-of-the-art work. Eventually, this paper concludes by showing the ASIC synthesis and post-layout simulation results of the proposed WSSR in 90 nm-CMOS technology, which senses 5.4× wider bandwidth than the state-of-the-art implementation.**

***Index Terms*— Cognitive radio, wideband spectrum sensing, very large scale integration (VLSI), application specific integrated circuit (ASIC), digital VLSI architecture design, and field programmable gate array (FPGA).**

I. Introduction

CONTEMPORARY research works on machine intelligence possess substantial capability to revolutionize the way electronics gadgets communicate with each other by sensing their surrounding and performing the appropriate tasks. Inline with this notion, Mitola and Maguire [1] coined the idea of cognitive radio that inculcates intelligence in the transceivers (or radios) to sense the spectrum band for its availability and allows them to opportunistically communicate via unused band (i.e. spectrum holes). It plays significant role in the present-day wireless communication systems where higher data-rate or bandwidth demand has surged due to the proliferation of connected wireless gadgets, most notably the smart-phones which consume ≈2.5 quintillion bytes of data per day [2]. For sustainable usage of spectrum band, which is limitedly available in nature, cognitive radio technology enables these gadgets to access the spectrum more efficiently by using the spectrum holes for communication. Thus, cognitive radio technology has showcased profound potential of mitigating the prospective spectrum-scarcity problem. Two of the major challenges in the deployment of this technology are sensing the wide range of spectrum band (i.e. wideband) and dynamically configuring communication parameters of the radio for transmission as well as reception of information. Our work deals with the former challenge, as the wideband (>100 MHz) spectrum sensing is a desirable feature for the deployment of cognitive radio technology in the current wireless communication systems that permits simultaneous sensing of multiple channels for better detection performance. In addition, seamless handoff of cognitive radio among different spectrum bands is an imperative factor for the real-world deployment that alleviates the data interruption which occurs when the licensed user of sensed spectrum band frequently returns for accessing it. To circumvent this, wideband spectrum sensing (i.e. carried out by dividing the wideband spectrum into multiple subbands) enables to backup extra subbands, apart from the one which is already in use. Thus, cognitive radio can seamlessly handoff to one of such reserved subbands when the currently used band needs to be released for the licensed user, to avoid data interruptions.

Unlike the narrowband spectrum sensing [3], wideband spectrum sensing (WSS) aims to determine extra spectral possibilities across a wider frequency range and boost the opportunistic aggregate throughput of cognitive radio network. However, the traditional WSS technique based on standard analog-to-digital converters (ADCs) incurs an unaffordable high sampling rate as well as implementation complexity. The WSS can be carried out by using two sampling methods: Nyquist and sub-Nyquist sampling methods. Later one is also referred as compressive sampling [4] that is acquiring massive popularity in various research and industrial applications, as it mitigates the drawback of high sampling rates or high implementation complexity of Nyquist sampling method. In sub-Nyquist WSS, wideband signals are obtained at the sampling rate lower than Nyquist rate. Such sub-Nyquist sampling can be achieved by accounting for the sparse occupancy of radio frequency spectrum that has inspired theoretical research on WSS techniques for the past half decade [5], [6]. In this

Manuscript received 14 July 2022; revised 10 October 2022 and 31 October 2022; accepted 16 November 2022. This work was supported by the Indian Institute of Technology Mandi. This article was recommended by Associate Editor M. Martina. *(Corresponding author: Rahul Shrestha.)*

The authors are with the School of Computing and Electrical Engineering, Indian Institute of Technology Mandi, Himachal Pradesh 175075, India (e-mail: s18018@students.iitmandi.ac.in; rahul_shrestha@iitmandi.ac.in; satinder@iitmandi.ac.in).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCSI.2022.3223356.

Digital Object Identifier 10.1109/TCSI.2022.3223356

context, research findings in the literature have presented various techniques like analog-to-information conversion (AIC) [11], [12], modulated wideband conversion (MWC) [11], [13], and multi-coset sampling [14], [15]. These reported works are promising candidates for developing commercial WSS systems, as they can achieve sub-Nyquist sampling rate which permits the use of low power and slower ADCs in their implementations. It has been reported that design flaws or model incompatibilities may readily impair the performance of the AIC-based model [12]. Furthermore, Mishali and Eldar proposed the MWC-based model in [16] which transformed AIC-based model to avoid the model mismatches. Major difference between MWC and AIC based models is that the former has more than one sampling channel with a low pass filter, replacing the accumulator in each channel. Unlike, the multi-coset sampling is another way to perform sub-Nyquist sampling with more than two channels. It is analogous to picking some samples from a uniform grid, by using the sampling rate (say f_s) which is higher than the Nyquist rate. Such uniform grid is segregated into blocks of L samples that are adjacent to each other. In every block, p samples are retained (where $p < L$) and rest are skipped. Hence, multi-coset sampling is carried out by using p sampling channels with a sampling rate of f_s/L and different time offsets for each sampling channel. Therefore, this technique of multi-coset sampling is better than AIC and MWC based techniques, as the sampling rate in each channel of multi-coset sampling is $L\times$ slower than the Nyquist rate.

However, the reported works on multi-coset sampling based WSS for cognitive radio networks are limited in literature. Note that the multi-coset sampling can be performed in both frequency and time domains. In frequency domain, the sparse fast Fourier transform (sFFT) algorithm has been used to improve both accuracy and execution time of the spectrum sensor [17], [18]. These improvements allow the sFFT algorithms to estimate the Fourier coefficients in shorter time than the standard FFT algorithms for frequency sparse signals. In the reported works from [19], [20], sub-Nyquist rates have been achieved, however, the system is not robust due to its higher noise sensitivity. Later, Agarwal et al. [21] achieved better noise robustness by using random sampling that approached the Nyquist sampling rates, requiring a complex analog frontend module. Besides, a significant number of cooperative spectrum sensing (CSS) algorithms are reported in the literature [7], [8], [9], [10]. These algorithms provide extremely reliable detection-performance at the cost of higher computational complexity. Thus, random sampling techniques that approach the Nyquist sampling rates are used to conceive cooperative spectrum sensor (CSR) hardware designs which are suitable for the implementation of cooperative topology-based cognitive radio networks in real-world scenarios. However, these designs require more energy and large space. In recent literature, several CSR implementations have been reported [32], [33], [34] for eigenvalue-based CSS algorithms. They deliver excellent detection performance under the presumption that noise and signal intensity are distributed equally among secondary users in the CSS network. Further, reported work from [22] addressed all the aforementioned concerns and designed the spectrum sensor for cooperative WSS [23]. In comparison to [22], the state-of-the-art implementation from [24] improved the sensing time and achieved higher throughput. However, to the best of authors' knowledge, hardware implementation of time domain based multi-coset sampling for WSS has not been reported till date in the literature. In order to enhance the reliability of WSS, our work realizes the multi-coset sampling based WSS (in time domain) using the blind eigenvalue-based detection (EVD). Hence, it does not require the knowledge of signal as well as noise power and hence it is highly immune to noise, delivering reliable detection performance. Unlike in time domain, the hardware implementation of blind EVD in frequency domain is extremely complex. Therefore, this work attempts to report algorithmic and architectural contributions for the time domain based multi-coset sampling WSS. The highlights of our contributions are as follows.

- Hardware-friendly VLSI algorithm for the multi-coset sampling based wideband spectrum sensing has been proposed in this paper. It has been fundamentally formulated by optimizing the interpolation and the model selection processes.
- Corresponding to the proposed wideband spectrum sensing algorithm (WSSA), this paper presents a new VLSI architecture of WSSR that delivers short sensing time (low latency) and high sensing bandwidth.
- Performance analysis of our WSSA and hardware implementation of WSSR in FPGA as well as ASIC platforms are reported in this work. Eventually, the implementation results of proposed WSSR are compared with the relevant works from the literature.

II. Preliminaries and Proposed VLSI Algorithm

1) Preliminaries: A system level overview for digitally sensing the spectrum holes in wideband spectrum is shown in Fig. 1. It primarily comprises of WSSR, multicoset sampler (MCS), low noise amplifier (LNA), band pass filter (BPF), and antenna. They are the key modules of cognitive-radio receiver used by the secondary uaser for stand-alone WSS. At first, antenna transforms the captured electromagnetic waves into a high-frequency analog signal $x_R(t)$ which is fed to analog BPF that filters the signals from sensed spectrum of bandwidth B_{max}. Such filtered signals are further amplified by LNA to generate $x(t)$ signal, as shown in Fig. 1. Here, $x(t)$ represents wideband sparse signal that is bandlimited to $[0, B_{max}]$. For WSS [14], received analog signal $x(t)$ is segregated into L subband/narrowband channels in which each of them has a bandwidth of B and hence the maximum scanned bandwidth is $B_{max} = L \times B$, as shown in Fig. 1. Furthermore, if $X(f)$ denotes the Fourier transform of $x(t)$ then $X(f)\in B(F)$ such that $B(F)$ is the set of finite-energy continuous complex-value signals which is bandlimited to $F \subset [0, B_{max}]$ where $F = \bigcup_{i=1}^{N} \left[f_i^l, f_i^u\right]$. Here, N denotes the number of active subband channels, out of L subband channels (i.e. $N\in L$), those are being utilized by the primary users (PUs). Based on the prior information of B_{max}, B and L, the received wideband

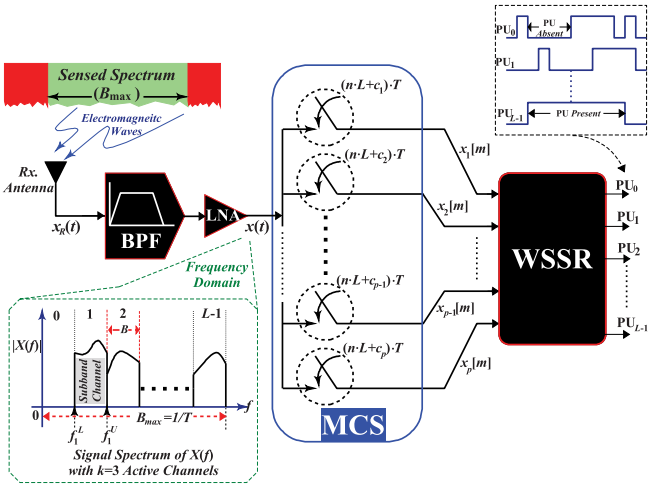

Fig. 1. Schematic illustration for system level overview of wideband spectrum sensing.

signal $x(t)$ is sampled using MCS that generates p digital samples as

$$x_i[m] = x\left[(n{\cdot}L + c_i)/B_{max}\right], \quad m{\in}\mathbb{Z}, \tag{1}$$

$\forall\ i = \{1, 2, \ldots p\}$ where c_i is a co-primes of p distinct numbers from the set $\$ = \{0, 1, \ldots L-1\}$ [15]. Here, average sampling rate of MCS is $f_{avg} = \alpha \times B_{max}$ such that $\alpha = p/L$ is the sub-Nyquist factor. According to Landau's lower bound [25], α is lower limited to the maximum channel occupancy, that is $\alpha{\geq}N_{max}/L$ where $N_{max}{\geq}N$ is maximum possible number of occupied/active channels. In order to perform blind channel detection, the number of cosets $p{\geq}2N$ must be set [16]. Following that, WSSR receives the parallel p digital samples: $x_1[m], x_2[m] \cdots x_p[m]$ where each sample is a complex quantity that contains in-phase and quadrature components. Eventually, these samples are processed by WSSR to produce decision outputs $PU_0, PU_1, PU_2, \cdots PU_{L-1}$, indicating the presence or absence of PUs in the subband channels of sensed spectrum band, as shown in Fig. 1.

2) Proposed VLSI Algorithm: We present the proposed hardware-friendly WSSA in Algorithm 1 for the micro-architecture design of the WSSR in Fig. 1. Here, a complex correlation matrix $\hat{R}_{p\times p}$ of order p is generated for a particular configuration of MCS outputs $x_i[m]\ \forall\ i = \{1, 2\cdots p\}$ which are the inputs to WSSR, as shown in Fig. 1. To begin with, each $x_i\,[m]$ data sequence is up-sampled by a factor of L, resulting in $x_{u_i}\,[n] = x_i(n/L)$ if $n = m \times L$, else $x_{u_i}\,[n] = 0\ \forall\ m{\in}\mathbb{Z}$. Thereafter, it is convoluted for the computation of $x_{h_i}\,[n] = x_{u_i}\,[n] * h\,[n]$ where $h[n]$ is an interpolation filter with the frequency response $H(f) = 1$ if $f{\in}[0, B]$, otherwise $H(f) = 0$. Specifically for a particular case, Fig. 2 (a) shows conventional solution for the convolution problem between $h[n]$ and $x_{u_i}[n]$. It shows that the zero rows are computed for $L-1$ zeros, contained within the $x_i[m]$ data sequence. Thus, such convolution requires very high computational space. To circumvent this issue, a hardware-efficient method is proposed in Algorithm 2. It proposes to skip the zero-padding data during convolution such that $x_i[m]$

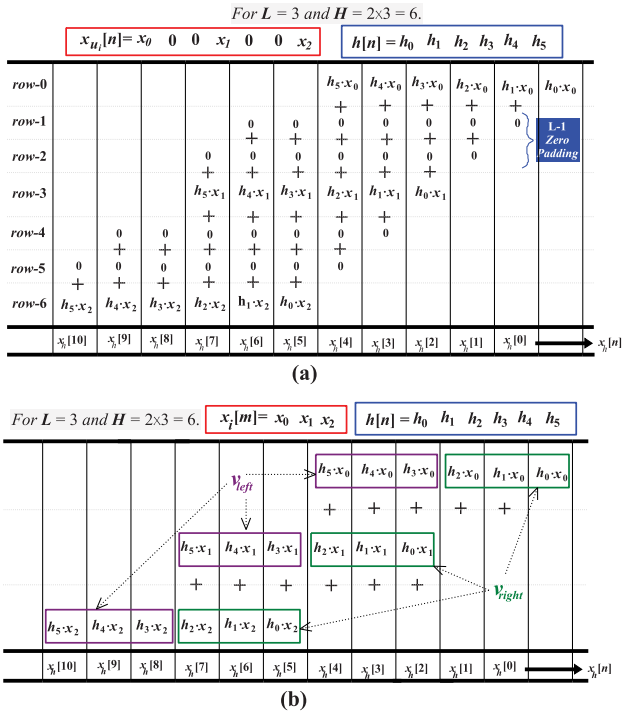

Fig. 2. Schematic representations of (a) conventional and (b) proposed methods of computing convolutions between two samples.

data samples are delivered one-by-one, as shown in Fig. 2 (b). Subsequently, the proposed Algorithm 2 simultaneously computes two vectors: v_{left} and v_{right} for each $x_i[m]$ data sample. In v_{right} vector, one $x_i[m]$ (i.e. $m = 0$) data sample is multiplied with 0 to $L-1$ coefficients of the $h[n]$ filter. For v_{left} vector, the same data sample is multiplied with $L{+}1$ to H coefficients of $h[n]$ filter, as shown in Fig. 2 (b) for the values of $L = 3$ and $H = 6$. This process is applied to the data sequence of $x_i[m]$, as presented in lines 5−16 in Algorithm 2. In order to arrange v_{right} and v_{left} vectors in right order, $zeros(1, L)$ vector has been appended at the beginning as well as end of both these vectors and consecutively, they are added, as illustrated in Fig. 2 (b). Eventually, a dl delay is applied at the output of $x_h[n]$ samples to achieve correctly convoluted $x_h[n]$ data set, as presented in lines 17−20 of Algorithm 2. Thereafter, $x_{h_i}[n]$ filtered data is lagged with a constant delay of c_i to generate the samples $x_{c_i}[n] = x_{h_i}[n - c_i]$, referring line 11 in Algorithm 1. Consequently, frequency domain of the received signal $x_{c_i}[n]$ is represented in the matrix form as $y(f) = A{\cdot}X(f) + n(f)\ \forall\ f{\in}[0, B]$ where $A_{p\times L}$ is a modulation matrix with $(i, j)^{th}$ element given by [16]

$$A_{(i,j)} = B \times \exp\left(\frac{j2\pi\, c_i(j-1)}{L}\right) \tag{2}$$

and $n(f)$ is frequency-domain representation of the Gaussian complex noise with $\mathcal{N}(0, \sigma^2 I)$ distribution.

Thereafter, the $\hat{R}_{p\times p} = \frac{1}{M}\sum_{n=1}^{M} x_{c_i}[n]{\cdot}x^*_{c_i}[n]$ matrix is decomposed into eigenvalues $\Lambda = \{\lambda_1, \lambda_2 \ldots \lambda_p\}$ and corresponding eigenvectors $W = \{w_{\lambda_1}, w_{\lambda_2} \ldots w_{\lambda_p}\}$ where M denotes number of samples in each sequence. Thus, the

Algorithm 1 Proposed Implementation-Friendly Multicoset-Based WSSA

1: **Input-1:** $x_i[m]\ \forall\ 0 \le i \le p$, ▷ *p-number of multicoset data sequences from* (1).
2: **Input-2:** (a) Value of L subband channels, (b) pre-computed constant c_i values, (c) $A \in \mathbb{C}^{p\times L}$ matrix, (d) coefficients $h_i\ \forall\ 0 \le i \le H = 2\cdot L$ of H^{th}-ordered FIR filter ($h[n]$), and (6) ψ threshold value.
3: **Output:** PU detection values $PU_i\ \forall\ i = \{0, 1, 2\cdots L-1\}$ for L different subband channels.
4: Set $k = 0$.
5: **Start Parallel Execution**
6: $\{x_{h_1}\}$ = HFCONV$(x_1[m], L, h[n], x_h)$, ▷ *Calling proposed hardware-friendly convolution function from* Algorithm 2.
7: $\{x_{h_2}\}$ = HFCONV$(x_2[m], L, h[n], x_h)$,
8: : :
9: $\{x_{h_p}\}$ = HFCONV$(x_p[m], L, h[n], x_h)$.
10: **End Parallel Execution**
11: $x_{c_i}[n] = x_{h_i}[n - c_i]$.
12: Construct $\hat{R}_{p\times p}$ matrix.
13: Compute $[W, \Lambda]$= EVD$(\hat{R}_{p\times p})$. ▷ *Using Parallel Jacobi Method from* [27].
14: Compute $\hat{\Lambda} = Sort(\Lambda)$.
15: Compute $\hat{N} = \text{HFMDL}\left(\hat{\Lambda}, M, \hat{N}\right)$.
16: Compute $\hat{E}_n$=$W(:, \hat{N}+1 : p)$.
17: **Repeat:**
18: $P_{MU}(k) = \dfrac{1}{\parallel a_k \cdot \hat{E}_n \parallel^2}$,
19: $k = k + 1$,
20: **Until** $k = L - 1$. Return $P_{MU}(k)\ \forall\ k = \{0, 1\cdots L-1\}$.
21: **Start Parallel Execution**
22: PU_0= DETECT$(PU, P_{MU}(0), \psi)$,
23: PU_1= DETECT$(PU, P_{MU}(1), \psi)$,
24: : :
25: PU_{L-1}= DETECT$(PU, P_{MU}(L-1), \psi)$.
26: **End Parallel Execution**
27: **function** DETECT$(PU,\ P_{MU}, \psi)$
28: **if** $P_{MU} > \psi$ **then**
29: $PU = 1$; ▷ PU Present.
30: **else**
31: $PU = 0$. ▷ PU Absent.
32: **End of DETECT function.**

Algorithm 2 Hardware-Friendly Interpolation Algorithm for Multicoset-Based WSSA

1: **Function** HFCONV $(x, L, h_i[n], x_h)$
2: **Initialize**: $\vec{x} = x;\ , \vec{h} = h_i[n]$;
3: $\vec{v}_{left} = 0_E;\ , \vec{v}_{right} = 0_E;\ , \vec{x}_h = 0_E;\ , \vec{x}_{ci} = 0_E$; ▷ *Set the empty vectors.*
4: dl=$(H+1)$/2;
5: **for** $i \leftarrow 0$ to $length(\vec{x}) - 1$ **do**
6: **for** $j \leftarrow 0$ to $L - 1$ **do**
7: **Start Parallel Execution**
8: **Right elements**:
9: $m_{right} = \vec{x}(1, i) \cdot \vec{h}(1, j + L)$,
10: $v_{right} = \left[v_{right}\ \ m_{right}\right]$;
11: **Left elements**:
12: $m_{left} = \vec{x}(1, i) \cdot \vec{h}(1, j)$,
13: $v_{left} = \left[v_{left}\ \ m_{left}\right]$;
14: **End Parallel Execution**
15: **end**
16: **end**
17: $v_{right} \longleftarrow \left[zeros(1, L), v_{right}\right]$;
18: $v_{left} \longleftarrow \left[v_{left}, zeros(1, L)\right]$;
19: $x_h = v_{left} + v_{right}$;
20: $x_h \longleftarrow x_h(dl + 1 : end - dl + 1)$;
21: **End of HFCONV function.**

ordered eigenvalues $\lambda_1 \ge \lambda_2 \ge \cdots \ge \lambda_p$ are fed into information-theoretic criteria like the minimum description length (MDL) [26] for the process of model selection, as illustrated in lines 12−15 from Algorithm 1. As a result, the number of active channels for $0 \le r \le N_{max}$ can be estimated using the MDL criteria as

$$\hat{N} = \arg\min_r -M(p-r)\log\frac{g(r)}{a(r)} - \frac{1}{2}r(2p-r)\log M \quad (3)$$

where $g(r) = \prod_{i=r+1}^{p} \lambda_i^{1/p-r}$ and $a(r) = \dfrac{1}{(p-r)}\sum_{i=r+1}^{p}\lambda_i$ are geometric and arithmetic means, respectively, of $p - r$ eigenvalues of $\hat{R}_{p\times p}$ matrix. From (7), it can be observed that the MDL criteria is extremely complex and requires significant amount of hardware resources for its implementation. To mitigate this issue, we propose to subject such MDL criteria to logarithmic quotient, power, and product properties. Thus, the proposed hardware-friendly MDL-criteria is given by

$$\hat{N} = \arg\min_r -M\left[L_\Lambda\right] + M(p-r)\left[\log_{10}(S_\Lambda)\right] + C_r \quad (4)$$

where $L_\Lambda = \sum_{i=r+1}^{p}\log_{10}(\Lambda_i)$ and $S_\Lambda = \sum_{i=r+1}^{p}\Lambda_i$ are the cumulative sums of logarithmic eigenvalues and eigenvalues, respectively. On the other side, $C_r = \frac{1}{2}\cdot r\cdot(2\cdot p - r)\cdot\log_{10}(M) + M\cdot(p-r)\cdot\log_{10}\left(\frac{1}{p-r}\right)$ is a constant value associated with a particular r value. Based on (4), a hardware efficient algorithm has been presented in Algorithm 3. Here, both L_Λ and S_Λ can be determined concurrently, as presented in lines 8−14 of Algorithm 3. Following that, L_Λ is multiplied by $-M$ and the logarithmic S_Λ is multiplied by B_r, then both these products are added. Thereafter, a C_r constant is added in line 15 of Algorithm 3. Finally, index $\hat{N}$ of the mdl_r minimum absolute-value is determined for the next operation. Subsequently, eigenvalues of noise are defined as the $\left(p - \hat{N}\right)$ lowest eigenvalues after determining the number of active channels (i.e. magnitude of N). Here, the noise eigenvectors are associated with $\hat{E}_n$ as $p\cdot(p - \hat{N})$-ordered matrix, referring line 16 in Algorithm 1. To reconstruct the positions of active channels, a MUSIC-Like algorithm has been utilized to compute

$$P_{MU}(k) = \frac{1}{\parallel a_k \cdot \hat{E}_n \parallel^2}\ \ \forall\ \ 0 \le k \le L-1 \quad (5)$$

where $||\cdot||$ represents the 2-norm, a_k is a column of A matrix, provided by $a_k = \left[e^{j2\pi k c_1/L}, e^{j2\pi k c_2/L}, \ldots, e^{j2\pi k c_p/L}\right]^T$. It generates respective L different values for all the L subband channels i.e. $P_{MU}(k)\ \forall\ k = \{0, 2, \cdots L-1\}$. Eventually, all these $P_{MU}(k)$ values are compared with the pre-computed threshold ψ value. Hence, if $P_{MU}(k)$ is greater than ψ then the output is logical high which specifically indicates that k^{th} subband channel is active and all other $L - 1$ channels are vacant, as presented in lines 17−31 of Algorithm 1.

III. Proposed Hardware Architectures

A. Overall WSSR Architecture

Referring Fig. 1, consider an analog signal $x(t)$ (at the output of LNA) with the bandwidth of $[0, B_{max}]$ that is sampled by MCS for the fixed values of $p = 8$, $L = 22$, and $c_i = \{2, 3, 7, 10, 12, 14, 19, 21\}$ which are co-prime numbers. These data samples from $x_1[m]$ to $x_8[m]$ are fed as inputs to the proposed WSSR architecture that is presented in Fig. 3. Here, each $x_i[m]$ sample has been 32-bit quantized where

Algorithm 3 Proposed MDL VLSI Algorithm for Multicoset-Based WSSA

1: **Function** HFMDL $\left(\Lambda, M, \hat{N}\right)$
2: $p = lenght(\Lambda)$;
3: **Initialize**: $S_\Lambda = 0_E, L_\Lambda = 0_E, mdl_r = 0_E$; ▷ *Set empty vectors.*
4: **for** $r \leftarrow 1$ to p **do**
5: $B_r = M \cdot (p - r)$;
6: $C_r = \frac{1}{2} r \cdot (2p - r) \cdot \log_{10}(M) + M(p - r) \log_{10}\left(\frac{1}{p-r}\right)$;
7: **for** $i \leftarrow r + 1$ to p **do**
8: **Start Parallel Execution**
9: $\hat{L_\Lambda} = \log_{10}(ev(i))$,
10: $L_\Lambda = L_\Lambda + \hat{L_\Lambda}$;
11: $S_\Lambda = S_\Lambda + ev(i)$;
12: **End Parallel Execution**
13: **end**
14: $mdl_r = -M \cdot L_\Lambda + B_r \cdot \log_{10}(S_\Lambda) + C_r$;
15: $mdl_r \longleftarrow \left[mdl_r + m\hat{d}l_r\right]$;
16: **end**
17: [value, $\hat{N}$] $= min\{abs(mdl_r)\}$;
18: **End of HFMDL function.**

16 most-significant-bit (MSB) represent the real part of $x_i[m]$, and remaining 16 least-significant-bit (LSB) quantifies its imaginary part. Furthermore, the proposed WSSR-architecture operates at two distinct clock frequencies of $CLK1$ and $CLK2$ clock signals, as shown in Fig. 3. The $CLK1$ signal synchronizes WSSR with MCS such that its frequency is given by $f_{clk1} = \alpha \times B_{max}$. Unlike, clock frequency f_{clk2} of $CLK2$ signal is the maximum operating frequency of the proposed WSSR architecture that is derived based on its critical path delay. To start with the spectrum sensing process, input signal samples $x_i[m]\ \forall\ m = \{0, 1, 2 \cdots N_x\}$ are first transferred to the storing data (SD) block where N_x number of data samples are serially stored in the registers of SD. After successfully storing N_x number of $x_i[m]$ data samples, SD block generates the finish signal (denoted as $FnSgn$) which enables the entire WSSR architecture, via 2:1 multiplexer, to operate with $CLK2$ master-clock signal of clock frequency f_{clk2}, as shown in Fig. 3. Subsequently, the convolution sample (CS) block carries out the operations of lines 6−16 of Algorithm 2. Following the completion of convolution for single $x_i[m]$ data sample, which is stored in last register of SD block, the CS block feeds back a control signal (denoted as $NxtSgn$) that indicates the SD block to release subsequent $x_i[m]$ data samples for convolution. Such procedure repeats until all the stored data in SD block are convoluted and this marks the end of process, as illustrated in line 5 of Algorithm 2. Consecutively, the outputs of CS block are passed to covariance matrix (CM) block that generates the $\hat{R}_{8\times8}$ matrix. For the computations of eigenvalues (Λ) and eigenvectors (W), this complex Hermitian matrix is transformed into real symmetric matrix by CM block and subsequently transfers its elements to EVD block, as shown in Fig. 3. This block generates both Λ and W that are fed to MDL and MUSIC blocks, respectively. Here, MDL block determines $\hat{N}$ number of active channels using the eigenvalues. Subsequently, MUSIC block immediately begins after the MDL block supplies the $\hat{N}$ value. Eventually, MUSIC block generates information on L-numbers of the P_{MU} values, which are compared with ψ (referring lines 17−32 in Algorithm 1) to determine the spectrum occupancy by PU(s), as shown in Fig. 3. Note that the following subsections present comprehensive explanations of the proposed micro-architectures of SD, CS, MDL, and MUSIC blocks. It is to be noted that CM and EVD blocks used in our WSSR architecture are reported in [27].

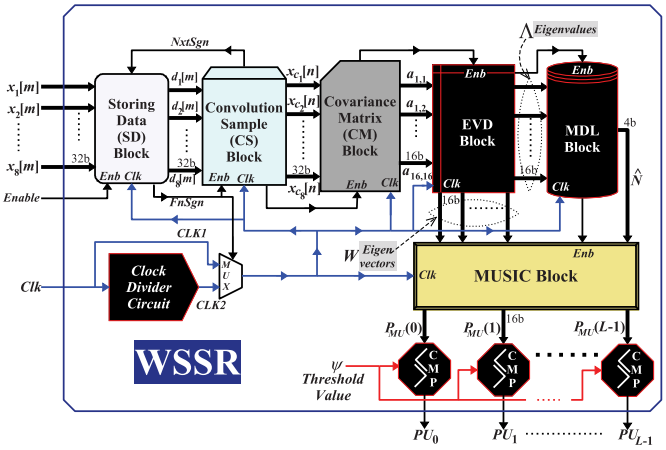

Fig. 3. Proposed overall architecture of WSSR for cognitive radio network.

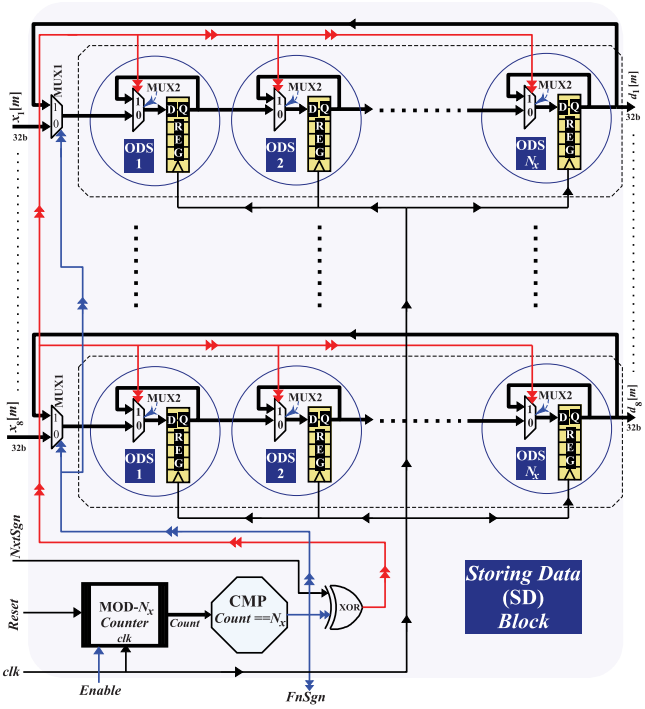

Fig. 4. Proposed micro-architecture of storing data (SD) block.

B. Hardware Architecture of SD Block

Detailed micro-architecture of the proposed SD block of WSSR is presented in Fig. 4. It stores the data sequences in parallel from $x_1[m]$ to $x_8[m]\ \forall\ m = \{1, 2, 3 \cdots N_x\}$. Initially, these data samples are fed to 2:1 multiplexer (denoted as MUX1) at the zero input data-line, as shown in Fig. 4. It shows that MUX1 output is connected to the one data store (ODS) block. Consecutively, N_x number of such ODS blocks are

serially connected along the first row of SD block to store $x_1[m]$ data sequence in every clock cycle. In addition, same rows in SD block are placed in-parallel for storing $x_2[m]$ to $x_8[m]$ data sequences, as shown in Fig. 4. Eventually, outputs of the last ODS blocks of these rows (i.e. $d_1[m]$ to $d_8[m]$) are passed to subsequent CS block in WSSR as well as these outputs are also simultaneously fed back to the one input data-line of respective MUX1, as shown in Fig. 4. In addition, SD block also contains a MOD-N_x counter which is initiated and initialized (to zero value) with enable and reset signals, respectively. Once the count of this counter attains the value of N_x, it generates logic-level high at *FnSgn* output signal that is routed to the select line of MUX1 and it terminates the data insertion from the MCS, as shown in Fig. 4. In the ODS micro-architecture, input is connected to a 2:1 multiplexer (MUX2) at zero input data-line and its output is passed to 32-bit register (REG). Its output is subsequently routed to the next ODS block and fed back to MUX2 at one input data-line. Finally, select line for MUX2 has been generated by XORing *FnSgn* and *NxtSgn* (fed back from CS block) signals.

C. Hardware Architecture of CS Block

Referring to our WSSR architecture from Fig. 3 and corresponding to lines 5−10 of proposed Algorithm 1, CS block is fed with $d_1[m]$ to $d_8[m]$ samples from SD block. The proposed architecture of CS block is presented in Fig. 5 where its input data samples are concurrently passed to one input data-lines of 2:1 multiplexers that are represented as MUX-11 to MUX-18. Select line of these multiplexers is the *FnSgn* control signal from SD block, as shown in Fig. 3. At the logic high level of *FnSgn* signal, outputs of MUX-11 to MUX-18 multiplexers are transferred to the HFCONV blocks, as presented in Fig. 5. In these HFCONV blocks, $d_1[m]$ to $d_8[m]$ samples are filtered via low pass filter with a cutoff frequency of $f_c = B_{max}/L$. In order to generate a real-valued (non-ideal) low-pass filter $h_r[n]$ with a length of $H = 2{\cdot}L$, we have used in-built high-level MATLAB-command $h_r[n]$=*fircsl1*$(H, 1/L, 0.02, 0.008)$ with a normalized cutoff frequency of $f_c = 1/L$, a pass-band ripple of 0.02, and a stop-band ripple of 0.008. This $h_r[n]$ filter has been frequency shifted to obtain the complex filter $h[n] = h_r[n] \times \exp(j{\cdot}\pi{\cdot}n/L)$ [16]. Thereafter, the coefficients of $h[n]$ filter are 32-bit fixed-point quantized such that 16-MSB and 16-LSB represent real and imaginary components, respectively. Additionally, filter coefficients $h[0], h[1]\cdots h[L-1]$ are steered by L:1 MUX-h$_{right}$ multiplexer, and the remaining coefficients $h[L], h[L+1]\cdots h[H]$ are routed via MUX-h$_{left}$ multiplexer, as shown in Fig. 5. It also shows that the outputs of MUX-h$_{left}$ and MUX-h$_{right}$ are further fed as inputs h_l and h_r, respectively, to the HFCONV block. Select line for both MUX-h$_{left}$ and MUX-h$_{right}$ multiplexers has been generated by Counter-1 module that stores the counter value in register REG1 via 2:1 multiplexer (i.e. MUX3) and an adder, as illustrated in Fig. 5. Note that the select line of MUX3 multiplexer has been generated by XORing *FnSgn* and *NxtSgn* signals. Here, *NxtSgn* signal is the output from REG2 register which is assigned with logic-high value, via 2:1 multiplexer MUX4, when the content of REG1 register from Counter-1 module attains $L-1$ value, as illustrated in Fig. 5.

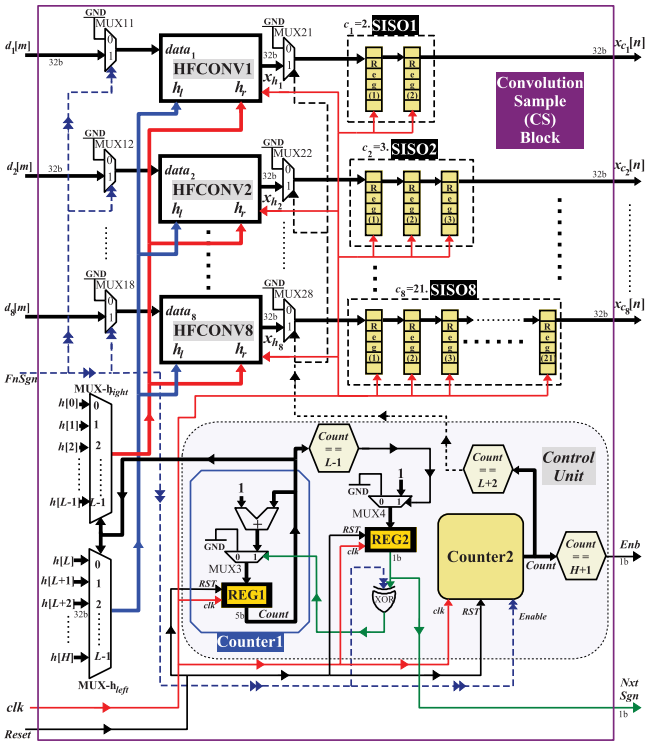

Fig. 5. Proposed hardware architecture of convolution sample (CS) block for the WSSR.

Furthermore, the outputs from HFCONV1 to HFCONV8 blocks (i.e. $x_{h1}[n]$ to $x_{h8}[n]$) are routed via 2:1 multiplexers (MUX-21 to MUX-28), along their one input data-lines. The outputs of these multiplexers are fed in-parallel to eight serial-in-serial-out blocks (SISO1−SISO8), as presented in Fig. 5. Referring line 11 of Algorithm 1, these SISO blocks introduce the delays using SISO1 to SISO8 blocks that contain $r_n = \{2, 3, 7, 9, 11, 13, 19, 21\}$ number of serially connected 32-bit registers, respectively, as shown in Fig. 5. Additionally, a Counter-2 module of CS block is enabled when *FnSgn* signal becomes logic high. Once this Counter-2 module attains $L{+}2$ value, it generates logic-high level for the select line of MUX-21 to MUX-28 multiplexers, as shown in Fig. 5. Consecutively, the outputs of SISO blocks that are denoted as $x_{c1}[n]$ to $x_{c8}[n]$ have been tapped as outputs from CS block and further fed to CM block of our WSSR. In addition, a high logic-level of *Enb* signal from CS block is delivered to the CM block when the Counter-2 module attains a value of $H{+}1$ that enables to skip $dl{=}(H{+}1)/2$ number of $x_{ci}[n]$ samples. Thereafter, the process of determining $\hat{R}_{8\times8}$ matrix for next $M = 1024$ data samples starts in CM block, as described in line 20 of Algorithm 2.

1) Micro-Architecture of HFCONV: This subsection specifically discusses the proposed micro-architecture of HFCONV1 block (which is the same architecture for other HFCONV blocks), as shown in Fig. 6. It processes $data_1$, h_l and h_r which are the outputs from MUX-11, MUX-h$_{left}$ and MUX-h$_{right}$, respectively, to perform the convolution from the proposed Algorithm 2. To begin with, $data_1$ and h_r are passed to the right complex multiplier (denoted as m_{right})

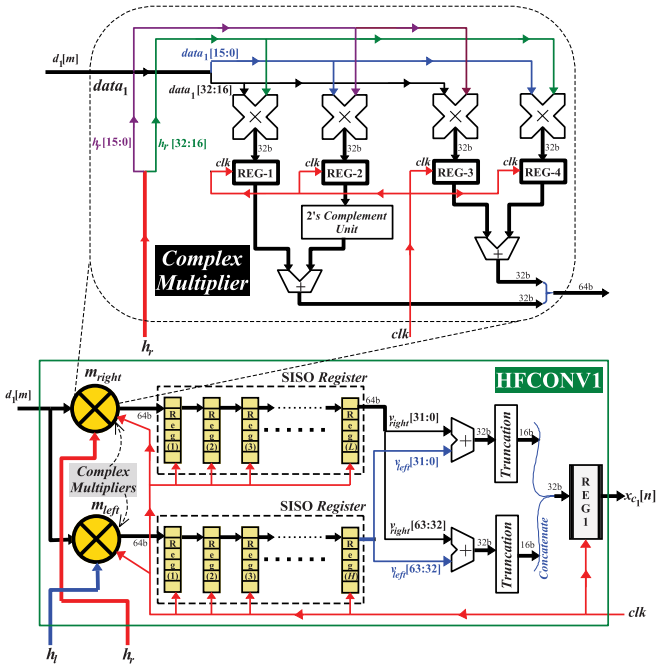

Fig. 6. Detail micro-architecture of HFCONV1 used in the proposed CS block.

whose product is fed to the right SISO block. It contains L number of serially connected 64-bit registers, as shown in Fig. 6. Concurrently, $data_1$ and h_l are also multiplied by the left complex multiplier (i.e. m_{left}). Its output is delivered to the left SISO block that contains $H = 2{\cdot}L$ number of 64-bit registers, as shown in lines 7−18 of Algorithm 2. Furthermore, outputs of these two SISO blocks: υ_{right} and υ_{left} are supplied to complex adder for summing the real components of $\upsilon_{right}[63{:}32]$ and $\upsilon_{left}[63{:}32]$ using Adder1. Concurrently, the imaginary parts $\upsilon_{right}[31{:}0]$ and $\upsilon_{left}[31{:}0]$ are added with the aid of Adder2 (referring line 19 of Algorithm 2). Eventually, both the outputs of Adder1 and Adder2 are 16-bit truncated, concatenated, and stored as 32-bit value in REG1 register, as shown in Fig. 6.

D. Design of MDL-Block Architecture

The proposed architecture of MDL block has been designed based on the proposed Algorithm 3, as shown in Fig. 7. It processes 16-bit input eigenvalues $\Lambda = \{\Lambda_1, \Lambda_2 \cdots \Lambda_{16}\}$ which are generated by EVD block in our WSSR. These eigenvalues are first processed by merge sorting (MS) block that sorts $\Lambda_1, \Lambda_2 \cdots \Lambda_{16}$ eigenvalues in decreasing order [28]. In fact, EVD block generates double eigenvalues of $\hat{R}_{8\times8}$ matrix. Here, MS block generates following outputs: $\hat{\Lambda} = \left\{\hat{\Lambda}_1 = \hat{\Lambda}_2 \geq \hat{\Lambda}_3 = \hat{\Lambda}_4 \cdots \hat{\Lambda}_{15} = \hat{\Lambda}_{16}\right\}$. Thus, the duplicate eigenvalues (i.e. eigenvalues from even positions) are omitted and only $\hat{\Lambda} = \left\{\hat{\Lambda}_3, \hat{\Lambda}_5, \hat{\Lambda}_7, \cdots \hat{\Lambda}_{15}\right\}$ eigenvalues are concurrently passed to the logarithm (LGM) and cumulative-sum (CMS) blocks, as shown in Fig. 7. Here, both LGM and CMS blocks simultaneously generates $L_\Lambda = \left\{\hat{L}_{\Lambda1}, \hat{L}_{\Lambda2} \ldots \hat{L}_{\Lambda7}\right\}$ and $S_\Lambda = \left\{\hat{S}_{\Lambda1}, \hat{S}_{\Lambda2} \ldots \hat{S}_{\Lambda7}\right\}$ values, respectively, as

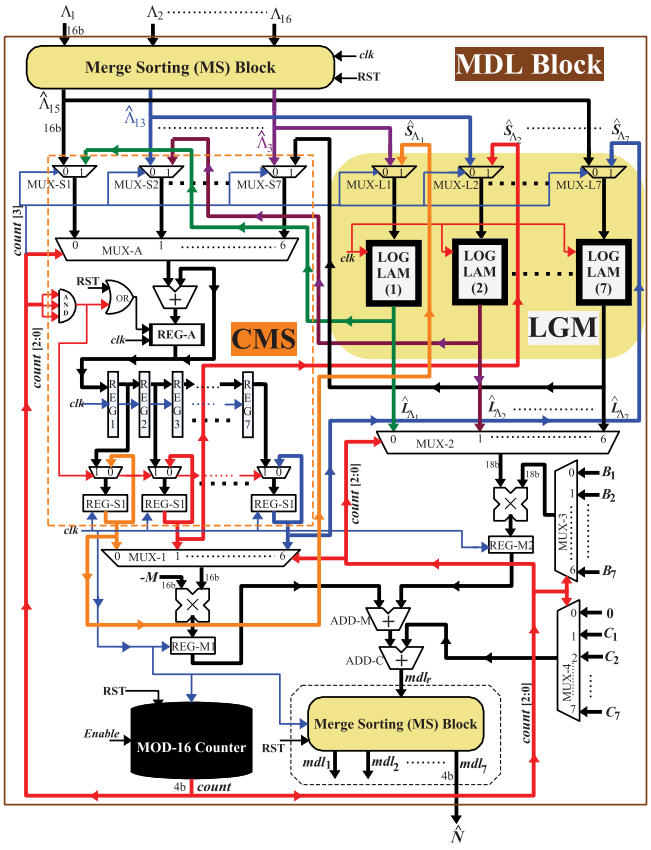

Fig. 7. Proposed micro-architecture of MDL block for our WSSR.

illustrated in lines 8-13 of Algorithm 3. Further, the inputs $S_{\Lambda_1} - S_{\Lambda_7}$ and $\hat{\Lambda}_3 - \hat{\Lambda}_{15}$ are fed to LOGLAM blocks via respective 2:1 multiplexers (MUX-L1 to MUX-L7) along one and zero input data-lines, respectively. Following that, LGM block uses the CORDIC algorithm to compute logarithms of the outputs from MUX-L1 to MUX-L7 using parallel LOGLAM-1 to LOGLAM-7 blocks, as presented in Fig. 7. The MOD-16 counter in MDL block generates a four-bit *count* signal such that the select line of MUX-L1 to MUX-L7 multiplexers is the MSB of this *count* signal (i.e. *count*[4]), as demonstrated by Fig. 7. Further, these multiplexer outputs are routed to parallel LOGLAM blocks which generate $L_{\Lambda_1} - L_{\Lambda_7}$ values after sixteen clock cycles. In the CMS block, inputs $\hat{\Lambda}$ and L_Λ are steered through 2:1 multiplexers (MUX-S1 to MUX-S7) along zero and one input data-lines, respectively. Subsequently, these multiplexer outputs are routed to a 7:1 multiplexer (MUX-A) and its select line has been tapped from three LSB of the *count* signal (i.e. *count*[2:0]). Using an adder and a 19-bit register (REG-A), MUX-A output is accumulated in the CMS block. Consecutively, REG-A data is transferred to serially connected registers REG-A1 to REG-A7, as shown in Fig. 7. It shows that that data from these serial registers are permanently stored in REG-S1 to REG-S7 registers via 2:1 multiplexers.

Furthermore, S_Λ and L_Λ outputs from CMS and LGM blocks are fed to two 7:1 multiplexers: MUX-1 and MUX-2, respectively. Select line of these multiplexers is three LSB of

count signal. Based on line 5 of Algorithm 3, a 7:1 multiplexer (MUX-3) has been used to steer each of the constant values $B_r \ \forall \ r = \{1, 2, 3, \cdots 7\}$ in every clock cycle. Thereafter, the MUX-1 output is multiplied with a constant value of $-M$ and stored in the REG-M1 register. Similarly, MUX-2 and MUX-3 outputs are multiplied and stored in REG-M2 register. Following that, the contents of REG-M1 and REG-M2 registers are added using ADD-M adder and its output is further added with C_r constant value (routed via 7:1 multiplexer MUX-4) using the adder ADD-C, in order to realize lines 6, 14 and 15 of Algorithm 3, as shown in Fig. 7. Finally, ADD-C adder output (i.e. mdl_r) is passed to the MS block that calculates absolute value of mdl_r and index value $\hat{N}$, as illustrated in line 17 of Algorithm 3. Here, $\hat{N} = min\{mdl_1, mdl_2 \cdots mdl_7\}$ and since $mdl_7 < mdl_6 < mdl_5 \cdots mdl_2 < mdl_1$, mdl_7 has been tapped as $\hat{N}$ output from the MS block, as shown in Fig. 7. This index value is passed on to the following MUSIC block in our WSSR for further processing, as illustrated in Fig. 3.

E. Hardware-Architecture for MUSIC Block

As described earlier in Fig. 3, the sorted eigenvectors $W = \left\{\hat{w}_{\lambda_1}, \hat{w}_{\lambda_3}, \hat{w}_{\lambda_5} \ldots \hat{w}_{\lambda_{15}}\right\}$ and $\hat{N}$ index from EVD and MDL blocks, respectively, are fed to MUSIC block. Its proposed architecture is shown in Fig. 8 where elements of one eigenvector $\hat{w}_{\lambda_1}$=$\{v_{11}, v_{21} \cdots v_{81}\}$ are routed via an 8:1 MUX-E1 multiplexer. Similarly, the remaining elements of eigenvectors (i.e. $\hat{w_{\lambda_3}} - \hat{w}_{\lambda_{15}}$) are simultaneously steered through MUX-E2 to MUX-E8 8:1-multiplexers. Our MUSIC block also contains two counter modules: Counter1 and Counter2 that generate *count1* and *count2* signals, respectively, as presented in Fig. 8. Here, Counter1 begins with the aid of an active-high *Enb* signal, which is received from the MDL block. Once *count1* reaches a value of seven, *count2* value is consecutively incremented by one. This *count1* signal is used as a select line for MUX-E1 to MUX-E8 multiplexers whose outputs are connected to a single 8:1 MUX-U multiplexer where the select line is *count2* signal. Thereafter, the output from MUX-U is transferred to a 2:1 MUX-V multiplexer, as illustrated in Fig. 8. It shows that the select line (*sel-v*) of MUX-V multiplexer is generated by comparing $\hat{N}$ index with *count2* value such that *sel-v* is set to logic-high when *count2* $\geq \hat{N}$, otherwise *sel-v* is reset to logic low. In the proposed MUSIC block, MU0 to MU21 sub-blocks are used for realizing equation (5), as illustrated in lines 17−20 of Algorithm 1. The inputs of each MU sub-block are the output of MUX-V multiplexer and *count2* signal. Following that, the $A_{8\times22}$ matrix has been priory computed and its elements are column-wise routed to MU0 to MU8 sub-blocks via MUX-A1 to MUX-A22 8:1 multiplexers in parallel, as illustrated in Fig. 8. It also shows that the select line of these multiplexers is *count1* signal. Subsequently, the output lines of these MUX-A1 to MUX-A22 multiplexers are connected to corresponding MU0 to MU21 sub-blocks. Here, $P_{MU}(0)$ to $P_{MU}(21)$ signals (each of 16 bit) from MU0 to MU21 sub-blocks, respectively, are the final outputs of MUSIC block. Eventually, these signals are compared with ψ threshold value to determine the active $L-1$ channels in our WSSR, as discussed in Fig. 3.

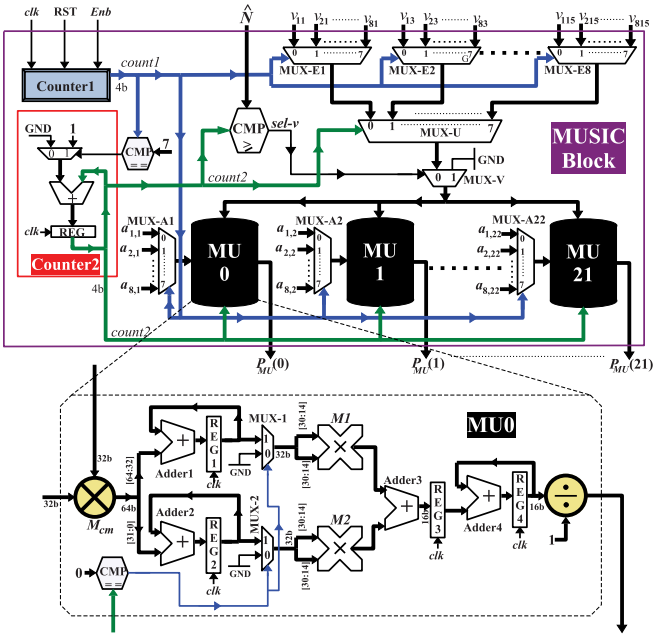

Fig. 8. Proposed hardware design of MUSIC block and micro-architecture of MU sub-block.

On the other hand, a detailed micro-architecture of MU0 has been presented in Fig. 8 (this design is the same for the rest of the MU1 to MU21 sub-blocks). To begin with, outputs from MUX-V and MUX-A1 multiplexers (which are inputs to the MU0 sub-block) are first multiplied by M_{cm} complex multiplier, whose architecture is the same as the m_{right} or m_{left} multiplier used in CS block from Fig. 6. The 64-bit output from M_{cm} is segregated into two data lines: its 32 MSB data is fed to Adder1 and the remaining 32 LSB is processed by Adder2. Furthermore, outputs from Adder1 and Adder2 are buffered into 32-bit REG1 and REG2 registers, respectively. Following that, REG1 data is delivered to the 2:1 MUX-1 multiplexer and data in REG2 is steered through the 2:1 MUX-2 multiplexer. Both these multiplexer outputs are supplied to $M1$ and $M2$ multipliers when *count2* value resets to zero, as shown in Fig. 8. Subsequently, the products from $M1$ and $M2$ are added (using Adder3) and stored in a 16-bit REG3 register. With the aid of Adder4, the data from REG3 is accumulated and stored in the REG4 register. Eventually, REG4 data has been processed by the divider to generate a 16-bit value of $P_{MU}(0)$, as illustrated in line 18 from Algorithm 1.

F. Sensing-Time Computation of WSSR

This subsection presents timing analysis to estimates the sensing time of proposed WSSR architecture. As discussed earlier in section III-A, our WSSR operates with two distinct clock frequencies: f_{clk1} and f_{clk2}. These frequencies and the latency Π (in terms of clock cycles) of WSSR are used for computing its sensing time τ_{sen}. Furthermore, Π has been segregated into six parts: Π_1, Π_2, $\cdots$ Π_6. Here, Π_1 and remaining $\Pi_2 - \Pi_6$ portions of Π correspond to f_{clk1} and f_{clk2} clock frequencies, respectively, while calculating the τ_{sen}

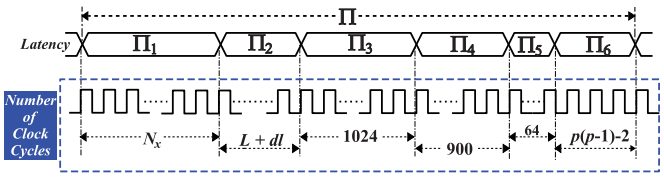

Fig. 9. Timing diagram for estimating the latency of proposed WSSR architecture.

value (in second) of proposed WSSR as

$$\tau_{sen} = \Pi_1/f_{clk1} + \Sigma_{i=2}^{6}\Pi_i/f_{clk2}. \quad (6)$$

Each of these parts ($\Pi_1 - \Pi_6$) represents the total number of clock cycles required to perform certain task by our WSSR. Segregation of clock cycles for different parts of Π has been schematically presented by the timing diagram in Fig. 9. Subsequently, explanations of the tasks performed during these time durations are explained below.

- Π_1 : Stores N_x number of $x_i[m]$ data samples in SD block, consuming N_x clock cycles.
- Π_2 : Waits $L+dl$ clock cycles while transmitting the correct $x_{ci}[m]$ data samples into the CM block.
- Π_3 : Calculates $\hat{R}_{8\times8}$ matrix for 1024 samples of $x_{ci}[m]$, requiring 1024 clock cycles.
- Π_4 : EVD block computes Λ and W values of $\hat{R}_{8\times8}$ matrix in 900 clock cycles.
- Π_5 : MDL block calculates the value of $\hat{N}$ index in 64 clock cycles.
- Π_6 : Finally, $L-1$ values of test statistics are computed using MUSIC block and compared with ψ value to detect the presence/absence of PUs, consuming $p(p-1)+2$ clock cycles.

Therefore, the expression for computing sensing time (in second) of the proposed WSSR architecture is

$$\tau_{sen} = (N_x/f_{clk1}) + \{(L + dl + p(p-1) + 1990)/f_{clk2}\}. \quad (7)$$

IV. Performance Analyses, Hardware Validation and Comparisons

A. Performance Analyses

The proposed WSSR must deliver higher probability of detection (P_d) and lower probability of false alarm (P_{fa}) for reliable performance. Hence, this section presents comprehensive performance analysis of our WSSR by plotting the P_d values obtained at various signal-to-noise ratios (SNRs) of the channel environment. We have carried out such performance analyses with the aid of extensive Monte-Carlo simulations in an additive-white Gaussian noise (AWGN) channel environment with the SNR values ranging between -10 to 10 dB. In order to evaluate the performance of proposed WSSR, $x(t)$ wideband analog-signal at the input of MCS (referring Fig. 1) is mathematically represented as

$$x(t) = \sum_{i=1}^{N} \alpha_i \cdot \text{sinc}\{B_i(t - t_i)\} \cdot \exp(j \cdot 2\pi \cdot f_i \cdot t) + n(t) \quad (8)$$

where $\text{sinc}(x) = \sin(x)/(x)$, and N denotes the number of bands. Here, i^{th} band of $x(t)$ has an energy of α_i, a precise support bandwidth of B_i, a time offset of t_i that is relative to the carrier frequency f_i, and this $x(t)$ is corrupted by the AWGN $n(t) = \mathcal{N}(0, 1)$. Further, performance of the proposed WSSR is regulated by p, L, N_x, and H coefficients (of the $h[n]$ low-pass filter) at various channel SNRs. In this work, the P_d value that quantifies probability of correct detection of an active channel has been computed as

$$P_d = \sum_{i=1}^{N} P_r(k_i \in \hat{k} \mid k_i \in k). \quad (9)$$

In this expression, k denotes the indices of N active-channels set and $\hat{k}$ is the estimated active-channels set. Note that P_d value is proportional to channel SNRs and P/L compression ratio (CR) which is a factor used for the calculation of reduction in multicoset sampling frequency. Consequently, we generate a wideband $x(t)$ signal in the range of 0−220 MHz with a single frequency band, such that $B_i = 10$ MHz $\forall$ $i = \{1, 2, 3 \cdots N\}$, with the varying SNR values between −10 to 10 dB and CR values of 0.1, 0.2, and 0.3 for computing the plots of P_d, as shown in Fig. 10 (a). It clearly shows that the P_d value increases with the CR value. Hence, it can be understood that the proposed WSSR delivers better detection performance at lower SNR by using larger CR value with substantial N_x samples and H coefficient values. However, performance analysis shown in Fig. 10 (a) has been carried out for $N_x = 46$ samples and $H = 383$. On the other side, by fixing the CR value of 0.36, $p = 8$, and $L = 22$, we have presented P_d versus SNR plots for various N_x samples, ranging between 20 to 50 samples, as shown in Fig. 10 (b). Here, the proposed WSSR delivers adequate detection performance with $N_x = 46$ samples and beyond this value, the performance improvement is marginal. Furthermore, serially connected ODS micro-architectures in SD block of our WSSR (referring Fig. 4) increases with the rise in N_x value and consequently elongates the sensing time. Thereby, $N_x = 46$ samples is adequate value for reliable performance and hardware-efficient architecture of the proposed WSSR. Similarly, its performance analysis has been carried out for various H coefficients of $h[n]$ values, using the $p = 8$, $L = 22$, CR=0.36, and $N_x = 50$, as illustrated in Fig. 10 (c). Here, once the $H \geq 40$, the proposed WSSR maintains consistent performance even at lower SNR. However, this will increase the sensing time due to increase in the sizes of CS block as well as steering logics in our WSSR architecture.

B. Hardware Implementation and Functional Validation

The proposed WSSR architecture has been coded using Verilog hardware-description-language (HDL) and synthesized in the environment of Vivado design-suite 2017. Its final netlist has been hardware implemented on the Zynq UltraScale+ FGPA board. With the aid of MATLAB simulation platform, the FPGA prototype of our WSSR has been functionally validated. At -5 dB of channel SNR, a wideband signal $x(t)$ has been generated in MATLAB environment based on (8) model using the additional specifications like

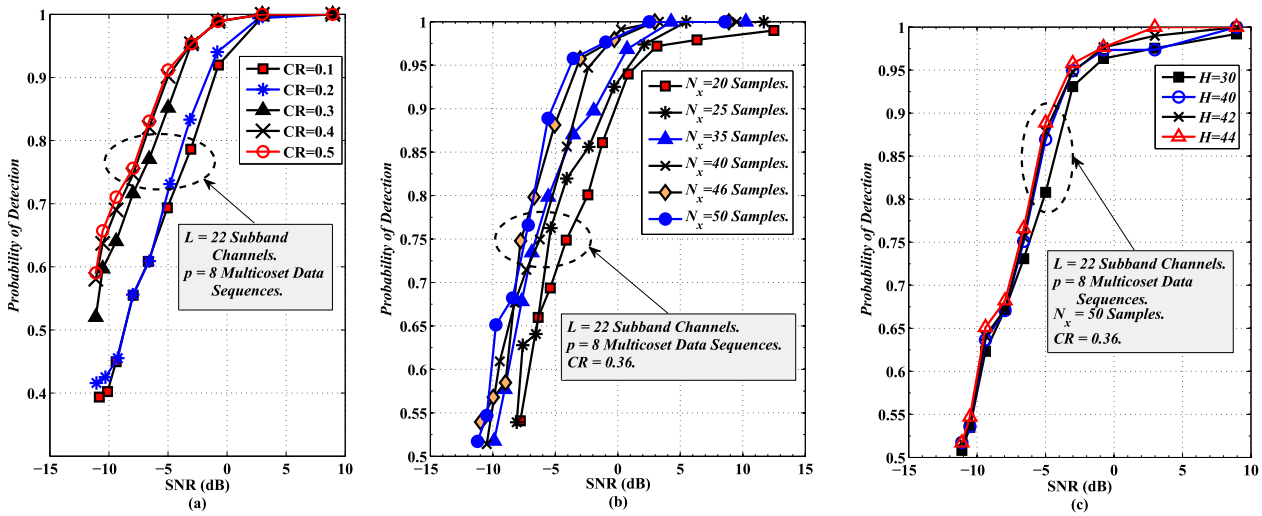

Fig. 10. Comprehensive performance analyses of the proposed WSSR, showing the P_d-versus-SNR plots with the varying values of (a) compression ratio, (b) number of data samples, and (c) filter coefficients.

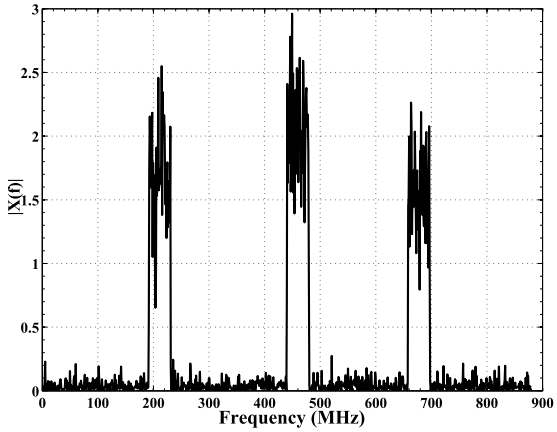

Fig. 11. Fast Fourier transform of the wideband signal $x(t)$ for validating the FPGA prototype of our WSSR with $N = 3$, f_i=[181.77, 395.4, 586.98] MHz, $B_i = 37.87$ MHz, t_i=[6, 13, 17], and $B_{max} = 833.3$ MHz.

TABLE I
COMPARISON OF MULTICOSET-BASED PROPOSED AND REPORTED WSSR IMPLEMENTATIONS IN STRATIX-IV FPGA BOARD

Design Metrics	This work	[24]	[22]
Device series	Startix IV	Startix IV	Startix IV
Sensing Bandwidth (GHz)	2.44	1.5	1.5
Dedicated Logic Registers	51542	5488	30193
Combinational ALUTs	94270	8328	23168
Total Block Memory (Bit)	43872	1572864	2242378
DSP48Es Cores	608	144	88
Clock Frequency (MHz)	170	126.691	205.85
Latency (Clock Cyles)	2162	15360	36864
Sensing Time (μs)	12.48	130	175

$N = 3$ bands, f_i=[181.77, 395.4, 586.98] MHz, $B_i = 37.87$ MHz, t_i=[6, 13, 17], and $B_{max} = 833.3$ MHz, as shown in Fig. 11. Furthermore, this wideband signal has been sampled using the MATLAB model of MCS (referring Fig. 1) with $c_i = \{2, 3, 7, 10, 12, 14, 19, 21\}$, $p = 8$, $L = 22$ and $T = 1/B_{max}$. Thereafter, each of the $x_i[m]$ output samples from MCS are quantized into 32 bit where 16 MSB represent the real component of $x_i[m]$ sample and the remaining 16 LSB quantify its imaginary part. Subsequently, these bit quantized values are stored in a text file (say *data.txt*). As discussed earlier in section III-A, the proposed WSSR architecture essentially operates at two clock frequencies: f_{clk1} and f_{clk2}. In order to synchronize SD block of WSSR with MCS, the longest path delay (∂_{SD}) of SD block plays a vital role and it includes the combinational delays (i.e. ∂_{MUX1} and ∂_{MUX2}) of two multiplexers: MUX1 and MUX2, referring Fig. 4. Hence, such longest path delay of SD block is expressed as

$$\partial_{SD} = t_{cq-REG} + \partial_{MUX1} + \partial_{MUX2} + t_{su-REG} \qquad (10)$$

where t_{cq-REG} and t_{su-REG} represent clock-to-Q delay and setup time of flip-flops in the registers, respectively. Based on the static timing analysis of WSSR FPGA implementation, the SD block operates at a maximum clock frequency of f_{max-SD}=$1/\partial_{SD}$=946 MHz, while processing 32-bit output samples from MCS. On the other side, an overall longest-path delay of our WSSR lies in the EVD block, which is presented in [27], that incurs a maximum clock frequency (f_{clk2}) of 215 MHz. In order to evaluate the functionality of our WSSR FPGA-prototype, we have designed a testbench (in Verilog HDL) that enables to feed $x_i[m]$ samples from *data.txt* file to SD block of WSSR at the clock rate of $f_{clk1} = 300$ MHz ($\because f_{clk1} = \alpha \times B_{max} = 0.36 \times 833.3 = 300$ MHz). Consequently, SD block stores $N_x = 50$ samples that takes $\tau_1 = N_x/f_{clk1} = 0.166$ μs. Hereafter, our WSSR operates at $f_{clk2} = 150$ MHz frequency of *CLK2* clock signal that is produced using a clock divider circuit (i.e. frequency division-by-two module), as shown in Fig. 3. Hence, WSSR design requires $\tau_2 = 2112/f_{clk2} = 14.08$ μs to deliver the final output. Eventually, an overall sensing time of the proposed WSSR architecture is $\tau_{sen} = \tau_1 + \tau_2 = 14.16$ μs. Its post placed-and-routed simulation results from Zynq UltraScale+ FPGA board has been presented in Fig. 12. This output

TABLE II
COMPARISON OF PROPOSED ARCHITECTURE WITH PRIOR REPORTED IMPLEMENTATIONS

Design Metrics	This work♣	[32]✠	[33]♣	[34]♣	[35]✠	[3]♣	[29]✠	[30]♣	[31]♣	[36]✠	[37]✠	[38]✠
Algorithm	**EVSR**‡	EVSR	EVSR	EVSR	EVSR	EVSR	FFT-MWED	FFT-DPED	FFT-EVSR	FFT-ED⊡	ED	ED
Topology	**SSSR**	CSR	CSR	CSR	CSR	SSSR	SSSR	SSSR	SSSR	SSSR	SSSR	SSSR
Sampling Method	**Sub-Nyquist**	Nyquist	Nyquist	Nyquist	Nyquist	Nyquist	Nyquist	Nyquist	Nyquist	Nyquist	Nyquist	Nyquist
Sensing Bandwidth (MHz)	**2170**	44.4⊙	43.85⊙	50.91⊙	44.4	400	200	250	40	0.36-0.72	132	40
Channel SNR (dB)	**-5**	-10	-10	-10	-10	0	-5	0	-10	27.9-25.7	>0	-NA-
Number of Frequency Sub-Bands	**4**	1	1	1	1	1	1024	32	32	1	1	1
CMOS Technology Node (nm)	**90**	130	90	90	130	90	65	180	180	130	65	130
Area (mm^2)	**6.7**	0.35	2.41	2.47	0.564	0.42	1.65	2.416	3.4	1.33	2.53	0.165
Scaled Area (mm^2)	**6.7**	0.73^{δ}	2.41	2.47	1.176^{η}	0.42	3.16^{α}	0.604^{β}	0.85^{γ}	1.33	1.31^{ν}	0.344^{φ}
Maximum Clock Frequency (MHz)	**160**	88.8	87.71	101.83	88.8	404	-NA-	-NA-	-NA-	-NA-	-NA-	-NA-
Sensing Time (μs)	**13.50**	43	60.4	133	236	53	$<$50000	–	610	133	420	1000

- ‡: Eigenvalue-based spectrum sensor; ♯: FFT-based multi-tap windowing energy-detector; §:FFT-based discrete wavelet-packet transform energy-detector. ⊡: FFT-based energy-detector.
- ⊙: Signal sensing bandwidth of CSR that is situated in the digital baseband part of spectrum sensing receiver.
- ✠: Measured results from ASIC chip; ♣: Synthesized and post-layout simulation results.
- α: $1.65\times x^2$, ν:$2.53\times x^2$ where $x=(90/65)^2$; β: $2.416\times y^2$ where $y=(90/180)^2$; γ: $3.4\times y^2$; δ: $0.35\times z^2$, η: $0.564\times z^2$, φ: $0.165\times z^2$ where $z=(90/130)^2$.

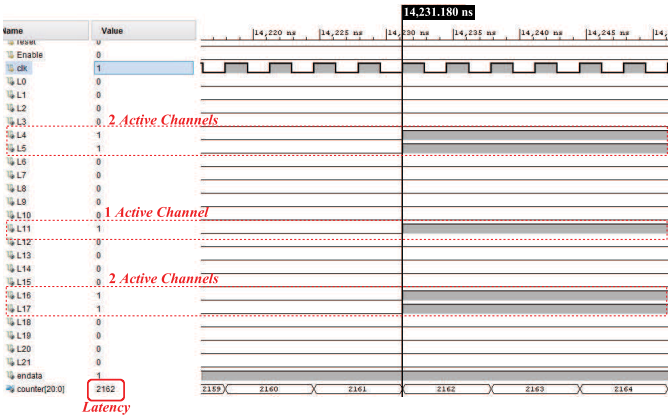

Fig. 12. Snapshot of the output waveform obtained from the post placed-and-routed FPGA implementation of the proposed WSSR architecture.

waveform clearly shows that the active channel set $PU_{MU}(\hat{k})$ $\forall\, \hat{k} = \{4, 5, 11, 16, 17\}$ (denoted as L4, L5, L11, L16 and L17 in Fig. 13) are asserted to high value after $\Pi = 2162$ clock cycles. Further, we have compared all MATLAB simulated and FPGA measured values of $P_{MU}(k)$ $\forall\, k = \{0, 1, 2, \cdots L-1\}$, as illustrated in Fig. 13.

As discussed earlier in section III-A, the sampling rate of MCS and the clock frequency of SD block (that is used for storing N_x samples) must be synchronized. Therefore, the sampling rate of MCS must be equal to $f_{clk1} = f_{max-SD} = 1/\partial_{SD} = \alpha \cdot B_{max}$. As a result, the maximum sensing bandwidth B_{max} of the proposed WSSR is given by

$$B_{max} = f_{clk1}/\alpha. \tag{11}$$

Hence, our WSSR architecture has a maximum sensing bandwidth of $B_{max} = \left(946 \times 10^6\right)/0.36 = 2.63$ GHz, when implemented on the Zynq UltraScale+ FPGA-board.

C. Comparisons and Discussion

In this subsection, the proposed WSSR architecture has been compared with the similar multicoset-based reported implementations from the literature [22], [24]. These reported architectures are cooperative wideband spectrum sensors that employ sparse fast Fourier transform (sFFT) for computing the FFT of a subset of time-domain input data [22], [24]. Unlike, the proposed multicoset-based WSSR is a stand-alone wideband spectrum-sensor which is implemented in time domain. For fair comparison, proposed WSSR architecture has been implemented on a Stratix-IV FPGA board (i.e. used by [22], [24]) with the same specifications that are earlier presented in section IV-B. In this FPGA platform, the SD block is capable of operating at a maximum operating frequency of $f_{max-SD} = 880$ MHz. Therefore, based on (11), the achievable B_{max} of our WSSR is 2.44 GHz when implemented on Stratix-IV FPGA board that is 38.5% higher than the reported implementations from [22], [24], as shown in Table I. It also presents that an overall maximum clock frequency (i.e. f_{clk2}) of our design is 170 MHz and the total latency remains same as $\Pi = 2162$ clock cycles which is 85.92% and 94.1% lesser than [24] and [22], respectively. Furthermore, the sensing time of proposed WSSR is computed as $\tau_{sen} = (50/850 + 2112/170) \times 10^{-6} = 12.48$ μs. Therefore, Table I shows that the sensing time of our WSSR is 90% and 92% faster than [24] and [22], respectively. However, the numbers of dedicated logic registers and combinational ALUTs in the proposed design is more than the reported implementations [22], [24]. At the same time, the usage of total block memory in our design is considerably lower than these reported works, as listed in Table I. Thus, the proposed WSSR architecture is capable of monitoring a wideband spectrum of ≈2.4 GHz at a higher rate (due to low sensing time) with adequate detection performance and moderate consumption of hardware resources.

Unlike the proposed WSSR that uses sub-Nyquist sampling, there are other reported implementations of multicoset-based WSSRs that use Nyquist sampling methods like FFT-based multi-tap windowing energy detector (FFT-MWED) [29], FFT-based discrete-wavelet packet-transform energy detector (FFT-DPED) [30], FFT-based eigenvalue-based spectrum sensor (FFT-EVSR) [31], EVSR [3], and CSRs (as discussed earlier in section I) [32], [33], [34], [35]. On the other hand, [37], [38] are an analog CMOS-RF based ED spectrum sensors. All the aforementioned implementations sample the received $x(t)$ signal at the Nyquist rate. Furthermore, comparison of the proposed WSSR architecture with these contributions has been presented in Table II. According to (11), our WSSR

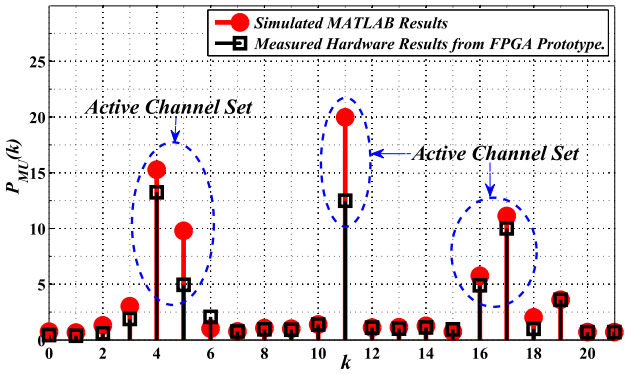

Fig. 13. Comparison plots of simulated and measured results of the proposed WSSR output.

has $B_{max} = 781.2/0.36 = 2.17$ GHz at -5 dB of SNR, when synthesized and post-layout simulated in 90 nm-CMOS technology node. It also delivers a sensing time of $\tau_{sen} = (50/781.2 \times 10^6) + (2112/160 \times 10^6) = 13.2$ μs. Thus, the proposed WSSR supports 10.85×, 8.68×, 54.25× and 5.4× wider bandwidth than the prior implementations from [3], [29], [30], [31], respectively. In addition, the proposed WSSR implementation delivers 42× to 49× greater bandwidth compared to reported CSR implementations [32], [33], [34], [35]. Furthermore, our design delivers a shortest sensing time of 13.5 μs among all the implementations, as listed in Table II.

V. Conclusion

This work contributed towards algorithmic, architectural, and hardware implementation aspects of multicoset sampling based wideband spectrum sensing in time domain for the cognitive radio network. Various contemporary (like 5G-NR) and next-generation (like 6G) wireless communication technologies need to access wide spectrum band and WSSR plays significant role to enhance the spectrum utilization. Therefore, VLSI algorithms and hardware architectures of such WSSR were proposed in this paper that delivered faster sensing time and wider sensing bandwidth, compared to existing implementations in the literature. Hence, our work possesses profound potential to be adopted for the deployment of next-generation wireless technologies to achieve higher spectral efficiency.

References

[1] J. Mitola and G. Q. Maguire, Jr., "Cognitive radio: Making software radios more personal," *IEEE Pers. Commun.*, vol. 6, no. 4, pp. 13–18, Apr. 1999.

[2] B. Marr, "How much data do we create every day? The mindblowing stats everyone should read," *Forbes*, May 2018. [Online]. Available: https://www.forbes.com/sites/bernardmarr/2018/05/21/how-much-data-do-we-create-every-day-the-mind-blowing-stats-everyone-should-read/?sh=30b2c21860ba

[3] R. B. Chaurasiya and R. Shrestha, "Hardware-efficient and fast sensing-time maximum-minimum-eigenvalue-based spectrum sensor for cognitive radio network," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 66, no. 11, pp. 4448–446, Nov. 2019.

[4] D. L. Donoho, "Compressed sensing," *IEEE Trans. Inf. Theory*, vol. 52, no. 4, pp. 1289–1306, Apr. 2016.

[5] L. De Vito, "A review of wideband spectrum sensing methods for cognitive radios," in *Proc. IEEE Int. Instrum. Meas. Technol. Conf.*, May 2012, pp. 2257–2262.

[6] E. Axell, G. Leus, E. G. Larsson, and H. V. Poor, "Spectrum sensing for cognitive radio: State-of-the-art and recent advances," *IEEE Signal Process. Mag.*, vol. 29, no. 3, pp. 101–116, May 2012.

[7] D. A. Guimarães, "Gini index inspired robust detector for spectrum sensing over Ricean channels," *Electron. Lett.*, vol. 55, no. 12, pp. 713–714, Jun. 2019.

[8] D. A. Guimaraes, "Robust test statistic for cooperative spectrum sensing based on the Gerschgorin circle theorem," *IEEE Access*, vol. 6, pp. 2445–2456, 2018, doi: 10.1109/ACCESS.2017.2783443.

[9] R. Zhang, T. J. Lim, Y.-C. Liang, and Y. Zeng, "Multi-antenna based spectrum sensing for cognitive radios: A GLRT approach," *IEEE Trans. Commun.*, vol. 58, no. 1, pp. 84–88, Jan. 2010.

[10] B. Nadler, F. Penna, and R. Garello, "Performance of eigenvalue-based signal detectors with known and unknown noise level," in *Proc. IEEE Int. Conf. Commun.*, Jun. 2011, pp. 1–5.

[11] H. Sun, A. Nallanathan, C.-X. Wang, and Y. Chen, "Wideband spectrum sensing for cognitive radio networks: A survey," *IEEE Wireless Commun.*, vol. 20, no. 2, pp. 74–81, Apr. 2013.

[12] J. N. Laska, S. Kirolos, M. F. Duarte, T. S. Ragheb, R. G. Baraniuk, and Y. Massoud, "Theory and implementation of an analog-to-information converter using random demodulation," in *Proc. IEEE Int. Symp. Circuits Syst.*, May 2007, pp. 1959–1962.

[13] C.-P. Yen, Y. Tsai, and X. D. Wang, "Wideband spectrum sensing based on sub-Nyquist sampling," *IEEE Trans. Signal Process.*, vol. 61, no. 12, pp. 3028–3040, Jun. 2013.

[14] M. Rashidi, K. Haghighi, A. Owrang, and M. Viberg, "A wideband spectrum sensing method for cognitive radio using sub-Nyquist sampling," in *Proc. Digit. Signal Process. Signal Process. Educ. Meeting (DSP/SPE)*, Jan. 2011, pp. 30–35.

[15] R. Venkataramani and Y. Bresler, "Perfect reconstruction formulas and bounds on aliasing error in sub-Nyquist nonuniform sampling of multiband signals," *IEEE Trans. Inf. Theory*, vol. 46, no. 6, pp. 2173–2183, Sep. 2000.

[16] M. Mishali and Y. C. Eldar, "Blind multiband signal reconstruction: Compressed sensing for analog signals," *IEEE Trans. Signal Process.*, vol. 57, no. 3, pp. 993–1009, Mar. 2009.

[17] H. Hassanieh, P. Indyk, D. Katabi, and E. Price, "Simple and practical algorithm for sparse Fourier transform," in *Proc. 23rd Annu. ACM-SIAM Symp. Discrete Algorithms*, Jan. 2012, pp. 1183–1194.

[18] H. Hassanieh, P. Indyk, D. Katabi, and E. Price, "Nearly optimal sparse Fourier transform," in *Proc. 44th Symp. Theory Comput. (STOC)*, 2012, pp. 563–578.

[19] H. Hassanieh, L. Shi, O. Abari, E. Hamed, and D. Katabi, "GHz-wide sensing and decoding using the sparse Fourier transform," in *Proc. IEEE Conf. Comput. Commun.*, Apr. 2014, pp. 2256–2264.

[20] O. Abari et al., "0.75-million-point Fourier-transform chip for frequency-sparse signals," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2014, pp. 458–459.

[21] A. Agarwal, H. Hassanieh, O. Abari, E. Hamed, D. Katabi, and Arvind, "High-throughput implementation of a million-point sparse Fourier transform," in *Proc. 24th Int. Conf. Field Program. Log. Appl. (FPL)*, Sep. 2014, pp. 1–6.

[22] A. Lopez-Parrado and J. Velasco-Medina, "Cooperative wideband spectrum sensing based on sub-Nyquist sparse fast Fourier transform," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 63, no. 1, pp. 39–43, Jan. 2016.

[23] I. F. Akyildiz, B. F. Lo, and R. Balakrishnan, "Cooperative spectrum sensing in cognitive radio networks: A survey," *Phys. Commun.*, vol. 4, no. 1, pp. 40–62, Mar. 2011.

[24] M. Khayyeri and K. Mohammadi, "Design and implementation of a high-performance and high-speed architecture for wideband spectrum sensing in cognitive radio networks," *Circuits, Syst., Signal Process.*, vol. 39, no. 4, pp. 2151–2177, Apr. 2020.

[25] H. J. Landau, "Necessary density conditions for sampling and interpolation of certain entire functions," *Acta Math.*, vol. 117, no. 1, pp. 37–52, 1967.

[26] M. Wax and T. Kailath, "Detection of signals by information theoretic criteria," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. ASSP-33, no. 2, pp. 387–392, Apr. 1985.

[27] R. Sharma, R. Shrestha, and S. K. Sharma, "Low-latency and reconfigurable VLSI-architectures for computing eigenvalues and eigenvectors using CORDIC-based parallel Jacobi method," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 30, no. 8, pp. 1020–1033, Aug. 2022, doi: 10.1109/TVLSI.2022.3170526.

[28] B.-C. Huang and M. A. Langston, "Practical in-place merging," *Commun. ACM*, vol. 31, no. 3, pp. 348–352, Mar. 1988.

[29] T. H. Yu, C. H. Yang, D. Cabric, and D. Markovic, "A 7.4-mW 200-MS/s wideband spectrum sensing digital baseband processor for cognitive radios," *IEEE J. Solid-State Circuits*, vol. 47, no. 9, pp. 2235–2245, Sep. 2012.

[30] C.-K. Yang, C.-H. Hsieh, and Y.-H. Huang, "An energy-saving spectrum sensing processor based on partial discrete wavelet packet transform," in *Proc. VLSI Design, Autom. Test*, 2012, pp. 1–4.
[31] S. M. Safavi and M. Shabany, "A VLSI architecture for multiple antenna eigenvalue-based spectrum sensing," in *Proc. 19th IEEE Int. Conf. Electron., Circuits, Syst. (ICECS)*, Dec. 2012, pp. 153–156.
[32] R. B. Chaurasiya and R. Shrestha, "Design and ASIC-implementation of hardware-efficient cooperative spectrum-sensor for data fusion-based cognitive radio network," *IEEE Trans. Consum. Electron.*, vol. 68, no. 3, pp. 221–235, Aug. 2022, doi: 10.1109/TCE.2022.3167471.
[33] R. B. Chaurasiya and R. Shrestha, "A new hardware-efficient spectrum-sensor VLSI architecture for data-fusion-based cooperative cognitive-radio network," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 29, no. 4, pp. 760–773, Apr. 2021, doi: 10.1109/TVLSI.2021.3055344.
[34] R. B. Chaurasiya and R. Shrestha, "Area-efficient and scalable data-fusion based cooperative spectrum sensor for cognitive radio," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 68, no. 4, pp. 1198–1202, Apr. 2021, doi: 10.1109/TCSII.2020.3029874.
[35] R. B. Chaurasiya and R. Shrestha, "Hardware-efficient ASIC implementation of eigenvalue based spectrum sensor reconfigurable-architecture for cooperative cognitive-radio network," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2021, pp. 1–5.
[36] K. Banović and A. C. Carusone, "A sub-mW integrating mixer SAR spectrum sensor for portable cognitive radio applications," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 65, no. 3, pp. 1110–1119, Mar. 2018, doi: 10.1109/TCSI.2017.2772208.
[37] N.-S. Kim and J. M. Rabaey, "A dual-resolution wavelet-based energy detection spectrum sensing for UWB-based cognitive radios," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 65, no. 7, pp. 2279–2292, Jul. 2018, doi: 10.1109/TCSI.2017.2781542.
[38] V. Khatri and G. Banerjee, "A 0.25–3.25-GHz wideband CMOS-RF spectrum sensor for narrowband energy detection," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 24, no. 9, pp. 2887–2898, Sep. 2016, doi: 10.1109/TVLSI.2016.2530305.

Rahul Sharma (Graduate Student Member, IEEE) received the Bachelor of Technology degree in electronics and communication engineering from the Rajasthan Technical University, Kota, India, in 2017. He is currently pursuing the Master of Science (M.S.) degree with the School of Computing and Electrical Engineering, Indian Institute of Technology Mandi. His research interest focuses on developing efficient VLSI architectures and implementation-friendly algorithms for the spectrum sensing process of cognitive radio technology.

Rahul Shrestha (Senior Member, IEEE) received the Bachelor of Engineering degree in telecommunication engineering from the B.M.S. College of Engineering, Bangalore, India, in 2008, and the Ph.D. degree in electronics and electrical engineering from the Indian Institute of Technology Guwahati, in 2014. From 2014 to 2016, he was worked as an Assistant Professor at the Center for VLSI and Embedded System Technologies, International Institute of Information Technology Hyderabad, India. Currently, he holds the position of an associate professor with the School of Computing and Electrical Engineering, Indian Institute of Technology Mandi. In 2018, he served as Visiting Assistant Professor at the Blekinge Institute of Technology, Karlskrona, Sweden, under the European Erasmus+ International Credit Mobility Programme. His primary research interest is designing efficient algorithms as well as digital VLSI architectures and its transformation into ASIC-chip or FPGA-prototype for the real-world applications of signal processing, wireless communication, deep neural networks, forward-error-correction channel decoders, cognitive radio, and cooperative spectrum sensing.

Satinder K. Sharma (Senior Member, IEEE) received the Master of Science degree in physics (electronic science) from Himachal Pradesh University, Shimla, India, in 2002, and the Ph.D. degree from the Department of Electronic Science, Kurukshetra University, Kurukshetra, India, in 2007. From 2007 to 2010, he was a Post-Doctoral Fellow at the DST Unit on Nanoscience and Nanotechnology, Department of CHE, Indian Institute of Technology Kanpur, India. From 2010 to 2012, he worked as a Faculty Member at the Electronics and Microelectronics Division, Indian Institute of Information Technology, Allahabad, India. Since 2012, he has been working as a Faculty Member with the School of Computing and Electrical Engineering, Indian Institute of Technology Mandi, India. In 2015, he worked as a Visiting Faculty, Institute of Semiconductor Electronics, Stuttgart University, Germany. He has published more than 92 publications in the international peer-reviewed journals, and also presented several invited talks and research papers at more than 65 international and national conferences plus eight submitted patents. His current research interests include microelectronics circuits and systems, CMOS memories and 2D-FETs fabrication and characterization, MEMS/NEMS, sensors, and next generation lithography.