Design and Experimental Validation of an FPGA-based PMU Simultaneously Compliant with P and M Performance Classes

Asja Derviškadić, Mario Paolone
Distributed Electrical Systems Laboratory
École Polytechnique Fédérale de Lausanne (EPFL)
CH-1015 Lausanne, Switzerland
{asja.derviskadic, mario.paolone}@epfl.ch

***Abstract*—Low-inertia grids are characterized by high shares of harmonic and inter-harmonic distortion, produced by the inverters that interface non-conventional generation assets to the electrical grid. These interfering tones largely compromise synchrophasor estimation and may jeopardize protection and control strategies based on Phasor Measurement Units (PMU). Indeed, the IEEE Std. C37.118 does not require resiliency against inter-harmonic tones for protection PMUs. In view of increasing synchrophasor technology reliability, the paper presents the design and the experimental validation of a PMU that is simultaneously compliant with both protection (P) and measurement (M) class of PMU performance defined in the IEEE Std. C37.118. The synchrophasor estimator is based on the interpolated DFT and iteratively estimates and compensates the effects of the spectral interference produced by a generic interfering tone and the negative image of the fundamental tone. The proposed hardware architecture is based on a Field Programmable Gate Array (FPGA).**

***Index Terms*—Field Programmable Gate Array (FPGA), IEEE Std. C37.118, Interpolated Discrete Fourier transform (IpDFT), Phasor Measurement Unit (PMU), Synchrophasor Estimation.**

I. Introduction

The IEEE Std. C37.118 [1] and the recent IEC/IEEE Std. 60255-118 [2] have defined two performance classes, i.e., P-class and M-class, to which Phasor Measurement Units (PMUs) must comply with. P-class is meant for applications requiring fast response time, such as power system protection. M-class is meant for high accuracy measurements requiring resiliency against interfering spectral components. Recently, the idea of a single PMU satisfying P and M-class requirements at the same time has become increasingly popular for electrical utilities interested to use PMUs to simultaneously supply monitoring and protection functionalities of situational awareness systems [3], [4].

In this respect, this manuscript presents the design, implementation and experimental validation of a PMU that simultaneously fulfills the requirements of P and M performance classes. That is to say, the developed PMU complies with protection applications in terms of responsiveness and with measurement applications in terms of accuracy, without the need of switching between the two classes. Therefore, the reliability of subsequent mission-critical actions is preserved despite high shares of harmonic and inter-harmonic components. Also, from a cost perspective, the same PMU device can be used to supply simultaneously monitoring and protection operations.

The PMU integrates the iterative-Interpolated Discrete Fourier Transform (i-IpDFT) to estimate the synchrophasors, that has been analytically formulated in [5]. The i-IpDFT iteratively estimates and compensates the effects of the spectral interference produced by both a generic interfering tone, harmonic or inter-harmonic, and the negative image of the fundamental tone. The proposed approach is simultaneously P- and M-class compliant, without the need of switching between the two classes.

The PMU hardware is based on the National Instruments CompactRIO system embedding a Xilinx Kintex-7 Programmable SoC that integrates an ARM-based hard processor system with a Field Programmable Gate Array (FPGA) [6]. The synchronization to the Coordinated Universal Time (UTC) reference can be achieved using three different time dissemination technologies, i.e., the Global Positioning System (GPS), the Precision Time Protocol (PTP) and the White Rabbit protocol, that provide an overall time uncertainty of 100 ns, 1 μs and 1 ns, respectively [7].

The performance of the developed PMU is assessed for all the test conditions defined in [1] by means of the PMU calibrator in [8]. The study is meant to experimentally validate the theoretical findings of [5] in terms of synchrophasor estimation accuracy and measurement reporting latency.

The paper is structured as follows. Section II presents the synchrophasor estimation algorithm. Section III describes the FPGA design and the associated computational complexity. Section IV provides the results of the experimental validation. Section V concludes the paper with final remarks.

This work was supported in part by the Swiss Centre for Competence in Energy Research on the Future Swiss Electrical Infrastructure (SCCER) through Swiss Innovation Agency (Innosuisse - SCCER Program) and in part by Qatar Environment and Energy Research Institute (QEERI).

II. The Iterative Interpolated DFT for Synchrophasor Estimation

This Section aims at describing the iterative Interpolated DFT (i-IpDFT), a technique proposed in [5] to estimate the synchrophasors even in presence of an inter-harmonic tone. We simultaneously achieve compliance with both PMU classes of performance by:

- Adopting a 3-points IpDFT technique based on the Hanning window function (Section II-A) [9];
- Iteratively compensating the spectral interference produced by the main tone negative image (Section II-B) [10];
- Iteratively compensating the spectral interference produced by harmonic and inter-harmonic tones (Section II-C) [5];
- Using a computationally-efficient method to compute the DFT spectrum (Section II-D) [11].

A. The Interpolated DFT (IpDFT)

The IpDFT is a method for estimating the parameters of a sinusoidal waveform, i.e., its amplitude, frequency and phase. The technique enables the mitigation of the detrimental effects of spectral sampling by interpolating the highest amplitude DFT bins of the signal spectrum [9], [12]. Also, the technique reduces the effects of spectral leakage thanks to the adoption of bell-shaped windowing functions [13]. Finally, 3-points are adopted because multipoint interpolation schemes lead to more accurate results, as they inherently reduce long-term spectral leakage effects [14], [15].

In this respect, let us assume a static and discrete single-tone signal model $x(n)$:

$$x(n) = A_0 \cos\left(2\pi f_0 n T_s + \varphi_0\right), \quad n \in [0, N-1] \tag{1}$$

being N the number of samples, T_s the sampling time interval, and $\{f_0, A_0, \varphi_0\}$ the signal frequency, amplitude and initial phase, respectively.

The DFT $X(k)$ of the input signal is defined as:

$$X(k) \triangleq \sum_{m=0}^{N-1} x(m) W_N^{km}, \quad k, m \in [0, N-1] \tag{2}$$

being $T = N \cdot T_s$ the observation interval length and W_N the so-called twiddle factor:

$$W_N \triangleq e^{-j2\pi/N} \rightarrow W_N^{km} = e^{-j2\pi km/N} \tag{3}$$

The DFT can be performed with any computation method. Recursive DFT computation methods that update the DFT spectrum on a sample basis are normally more efficient when calculating a small amount of DFT bins (see Section II-D). The signal is then windowed in the frequency domain, leveraging on the convolution theorem that states that the point-wise multiplication of two signals in the time domain equals the convolution between their DFT spectra in the frequency domain. The Hanning window function is adopted, as it provides a good trade-off between the main-lobe width and side-lobe decay levels [9]. The DFT $X(k)$ of the windowed input signal is updated as:

$$X(k) = -0.25X(k-1) + 0.5X(k) - 0.25X(k+1) \tag{4}$$

where the k-th bin of the windowed DFT spectrum is computed as a function of the three neighboring non-windowed DFT bins $\{k-1, k, k+1\}$.

In case of incoherent sampling ($f_0/\Delta f \notin \mathbb{N}$), the peak value of the Discrete Time Fourier Transform (DTFT) of the fundamental tone is located between two consecutive DFT bins and the signal frequency can be expressed as:

$$f_0 = (k_m + \delta)\Delta f \tag{5}$$

being -0.5 $\leq \delta <$ 0.5 a fractional correction term, k_m the index of the highest bin and $\Delta f = 1/T$ the frequency resolution. The IpDFT problem lies in finding the correction term δ (and, consequently, the fundamental tone's parameters $\{f_0, A_0, \varphi_0\}$) that better approximates the exact location of the main spectrum tone.

In case of using the Hanning window function, the fractional term δ can be computed in a closed form by interpolating the 3 highest DFT bins [14]:

$$\delta = 2\varepsilon \cdot \frac{|X(k_m+\varepsilon)| - |X(k_m-\varepsilon)|}{|X(k_m-\varepsilon)| + 2\,|X(k_m)| + |X(k_m+\varepsilon)|} \tag{6}$$

where $\varepsilon = \pm 1$ if $|X(k_m+1)| \gtrless |X(k_m-1)|$. The fundamental tone's amplitude and phase are computed as:

$$A_0 = |X(k_m)| \left|\frac{\pi\delta}{\sin(\pi\delta)}\right| \left|\delta^2 - 1\right|, \quad \varphi_0 = \angle X(k_m) - \pi\delta \tag{7}$$

IpDFT-based synchrophasor estimation algorithms typically adopt sampling rates in the order of tens of kHz and observation windows containing few periods of the signal at the power system nominal frequency. These settings enable the fulfillment of the Nyquist–Shannon sampling theorem and of the assumption behind the IpDFT of processing samples characterized by time-invariant parameters. Conversely, they introduce various well-known signal processing issues, such as poor frequency resolution and spectral leakage. In particular, the major issue is the fact that the spectral energy is concentrated around the DC component and, therefore, the positive and the negative image of the main tone of the signal under analysis are close to each other leading to a high amount of spectral leakage in case of incoherent sampling. In turns, the DFT bins adopted for the interpolation contain a contribution coming from the negative image. This effect, also known as *spectral interference*, has demonstrated to considerably corrupt the IpDFT estimations when applied to synchrophasor estimation [16].

B. The Enhanced-IpDFT (e-IpDFT)

To reduce the effects of spectral interference, [10] presents a technique, called enhanced-IpDFT (e-IpDFT), that mitigates the effect of the spectral leakage produced by the negative image of the tone under analysis.

Algorithm 1 The e-IpDFT algorithm [10].

1: $X(k) = \texttt{DFT}(x(n))$
2: $X(k) = -0.25X(k-1) + 0.5X(k) - 0.25X(k+1)$
3: $\{\widehat{f}_0^0, \widehat{A}_0^0, \widehat{\varphi}_0^0\} = \texttt{IpDFT}(X(k))$
4: **for** $p = 1 \rightarrow P$
5: $\quad \widehat{X}_0^{p-}(k) = \texttt{wf}(-\widehat{f}_0^{p-1}, \widehat{A}_0^{p-1}, -\widehat{\varphi}_0^{p-1})$
6: $\quad \widehat{X}_0^{p+}(k) = X(k) - \widehat{X}_0^{p-}(k)$
7: $\quad \{\widehat{f}_0^p, \widehat{A}_0^p, \widehat{\varphi}_0^p\} = \texttt{IpDFT}(\widehat{X}_0^{p+}(k))$
8: **end for**

As described in Algorithm 1, the method starts with the computation of the DFT spectrum of the input signal and of its windowing in the frequency domain (line 1-2 in Alg. 1). Then, a preliminary estimation of the main tone parameters is obtained by applying the IpDFT to $X(k)$ (line 3). These values are used to estimate the main tone's negative image $\widehat{X}_0^-(k)$ (line 5), whose analytic expression is known for the Hanning windowing function:

$$\begin{aligned} X(k) &= X_0^+(k) + X_0^-(k) \\ &= A_0 e^{+j\varphi_0} W(k - f_0/\Delta_f) + A_0 e^{-j\varphi_0} W(k + f_0/\Delta_f) \end{aligned} \tag{8}$$

being $W(k)$ the DFT of the Hanning function:

$$W(k) = -0.25 \cdot D_N(k-1) + 0.5 \cdot D_N(k) - 0.25 \cdot D_N(k+1) \tag{9}$$

where D_N is the so-called Dirichlet kernel:

$$D_N(k) = e^{-j\pi k(N-1)/N} \frac{\sin(\pi k)}{\sin(\pi k/N)}, \quad k \in [0, N-1] \tag{10}$$

The negative image is then subtracted from $X(k)$, to return an estimation of the main tone's positive image (line 6). Finally, the IpDFT is applied to the resulting spectrum (line 7). The compensation of the spectral interference produced by the negative image of the fundamental tone, can be improved by iterating the procedure a predefined number of times P. In the following, the function `e-IpDFT` refers lines 3-8 of Algorithm 1:

$$\{\widehat{f}_0, \widehat{A}_0, \widehat{\varphi}_0\}|_P = \texttt{e-IpDFT}[X(k)] \tag{11}$$

C. The Iterative-IpDFT (i-IpDFT)

The e-IpDFT does not account for the spectral interference produced by tones other than the fundamental one. In this regard, the i-IpDFT has been specifically designed to cope with the presence of interfering tones that are relatively close to the main one [5]. The proposed i-IpDFT algorithm iteratively estimates and compensates the effects of spectral leakage generated by an interfering tone and by the negative image of the main tone, such that the IpDFT is applied to a DFT spectrum that is only composed by the positive image of the main tone $X_0^+(k)$. The pseudo-code of the proposed i-IpDFT is reported in Algorithm 2.

Algorithm 2 The i-IpDFT algorithm [5].

1: $X(k) = \texttt{DFT}(x(n))$
2: $X(k) = -0.25X(k-1) + 0.5X(k) - 0.25X(k+1)$
3: $\{\widehat{f}_0^0, \widehat{A}_0^0, \widehat{\varphi}_0^0\}|_P = \texttt{e-IpDFT}[X(k)]$
4: $\widehat{X}_0^0(k) = \texttt{wf}(\widehat{f}_0^0, \widehat{A}_0^0, \widehat{\varphi}_0^0) + \texttt{wf}(-\widehat{f}_0^0, \widehat{A}_0^0, -\widehat{\varphi}_0^0)$
5: **if** $\sum |X(k) - \widehat{X}_0^0(k)|^2 > \lambda \cdot \sum |X(k)|^2$
6: $\quad$ **for** $q = 1 \rightarrow Q$
7: $\quad\quad \{\widehat{f}_i^q, \widehat{A}_i^q, \widehat{\varphi}_i^q\}|_P = \texttt{e-IpDFT}[X(k) - \widehat{X}_0^{q-1}(k)]$
8: $\quad\quad \widehat{X}_i^q(k) = \texttt{wf}(\widehat{f}_i^q, \widehat{A}_i^q, \widehat{\varphi}_i^q) + \texttt{wf}(-\widehat{f}_i^q, \widehat{A}_i^q, -\widehat{\varphi}_i^q)$
9: $\quad\quad \{\widehat{f}_0^q, \widehat{A}_0^q, \widehat{\varphi}_0^q\}|_P = \texttt{e-IpDFT}[X(k) - \widehat{X}_i^q(k)]$
10: $\quad\quad \widehat{X}_0^q(k) = \texttt{wf}(\widehat{f}_0^q, \widehat{A}_0^q, \widehat{\varphi}_0^q) + \texttt{wf}(-\widehat{f}_0^q, \widehat{A}_0^q, -\widehat{\varphi}_0^q)$
11: $\quad$ **end for**
12: **else**
13: $\quad$ **break**
14: **end if**

Let us consider a steady-state discrete signal composed of two tones, a fundamental and an interfering tone (not necessary harmonic, i.e., $f_i/f_0 \in \mathbb{R}$), both unknown:

$$x(n) = A_0 \cos(2\pi f_0 n T_s + \varphi_0) + A_i \cos(2\pi f_i n T_s + \varphi_i) \tag{12}$$

The first steps (lines 1-3 in Alg. 2) of the i-IpDFT algorithm correspond exactly to the e-IpDFT technique in Algorithm 1. Based on this preliminary estimate, the fundamental component contribution is computed (line 4), and then it is subtracted from the original spectrum in order to verify whether the resulting DFT bins contain spurious narrow-band components. An energy thresholding process is applied, and if the energy content E_n is larger than a predefined threshold λ (line 5), the iterative compensation routine is activated. The normalized spectral energy E_n is defined as the ratio between the spectral energy of the interfering tone detected during the first iteration of the i-IpDFT $X(k) - \widehat{X}_0^0(k)$, and the spectral energy of the original spectrum $X(k)$ (line 5).

If the threshold is not exceeded, the estimates $\{\widehat{f}_0^0, \widehat{A}_0^0, \widehat{\varphi}_0^0\}$ obtained via the first e-IpDFT (i.e., in line 3) are validated as the final outcomes. Otherwise, a peak search identifies the plausible bin associated to the interfering component and a new e-IpDFT is performed to estimate the interfering tone parameters $\{\widehat{f}_i^q, \widehat{A}_i^q, \widehat{\varphi}_i^q\}$ (line 7). Once identified the spurious component spectrum (line 8), it is subtracted from the original DFT bins obtaining the spectrum $X(k) - \widehat{X}_i^q(k)$ that does not contain the interfering tone (line 9). Finally, the e-IpDFT is applied to such spectrum leading to an enhanced estimation of the main tone parameters $\{\widehat{f}_0^q, \widehat{A}_0^q, \widehat{\varphi}_0^q\}$ (line 9). The whole procedure can be iterated a predefined number of times Q, leading to more and more accurate estimates as Q increases.

It is worth to point out that within the iterative procedure, the e-IpDFT is computed twice: once on the interfering tone (line 7) and once of the fundamental tone (line 9). At the first iteration q = 1, the IpDFT in line 3 of Algorithm 1 roughly estimates the parameters of the tone under analysis. Then the e-IpDFT refines this estimation by removing spectral

interference and these results are stored in a memory. At the next iterations $q > 1$ the e-IpDFT is computed again on the interfering and on the fundamental tone. However, at this stage the IpDFT estimates (line 3 of Alg. 1) are less accurate then the e-IpDFT estimates produced during the previous iteration (those stored in a memory). Therefore, for $q > 1$ every time the e-IpDFT is called, line 3 of Algorithm 1 is skipped, and the estimates of the previous iteration stored in the memory are directly used in the spectral interference compensation routine in line 5 of Algorithm 1.

Finally, the Rate-Of-Change-Of-Frequency (ROCOF) is computed by means of a classic backward first-order approximation of a first-order derivative:

$$ROCOF(n) = \left| \widehat{f}_0(n) - \widehat{f}_0(n-1) \right| \cdot F_r \qquad (13)$$

where $\widehat{f}_0(n)$ and $\widehat{f}_0(n-1)$ represent the fundamental frequency estimations at two successive reporting times. It is worth pointing out that this first-order approximation is known to perform badly under a low signal-to-noise ratio, therefore the use of ROCOF refinement techniques is advised [10].

D. The Modulated Sliding DFT (msDFT)

The modulated sliding DFT (msDFT) is a computationally efficient recursive algorithm that computes the DFT spectrum of a signal on a sample-by-sample basis [17]. The method is based on the DFT modulation property, that states the k-th bin of a DFT spectrum can be shifted to index k = 0 (i.e., the DC component) by multiplying the time domain input signal $x(n)$ by a modulated twiddle factor.

More specifically, every time a new sample $x(n)$ is received, the k-th DFT bin at time n $X_n(k)$ could be updated based on the N last received samples in the time interval $[n-N+1, n]$:

$$X_n(k) = \sum_{m=0}^{N-1} x(n-N+1+m) \cdot W_N^{km} \qquad (14)$$

Using the msDFT technique, instead, the same spectrum is computed as a shifted version of the DFT bin in k = 0 [17]:

$$X_n(k) = W_N^{-k(m+1)} \cdot X_n(0) \qquad (15)$$

According to the msDFT technique, the bin in position k = 0 is computed as a function of the previous iteration DFT bin in k = 0 and of the difference between the first $x(n-N)$ and the last $x(n)$ received samples in the sliding window [17]:

$$X_n(0) = X_{n-1}(0) + W_N^{km}(x(n) - x(n-N)) \qquad (16)$$

For a more extensive formulation of the msDFT technique the reader may refer to [11], that also includes implementation details within an embedded FPGA-based PMU device.

E. i-IpDFT Parameter Setting

Table I, resumes the final i-IpDFT settings that enable us to fulfill jointly the P and M-class requirements. It is worth pointing out that the parameters resumed in Table I are kept constant in the proposed design. In particular, the i-IpDFT considers a sampling rate of 50 kHz and a window containing 3 periods of a signal at the nominal power system frequency (corresponding to 60 ms at 50 Hz). As a convention, the time-stamp of the synchrophasor is referred to the center of the applied window. The synchrophasors are estimated at overlapping observation windows, that are shifted by the given PMU reporting rate of 50 frames per second (fps). The 3-points IpDFT is used to estimate the fractional correction term δ. Having set the window length to 60 ms, 8 windowed DFT bins have to be calculated in order to account for the frequency contributions in the range [10, 100] Hz. In this configuration, the first bin refers to the DC component, the second last bin refers to the second harmonic, and the last bin is needed for the three-points interpolation around 100 Hz. This means that the msDFT in (15) has to produce 2 additional bins. However, in view of the symmetry of the DFT spectrum with respect to the DC component, bin k = -1 is not computed but it is set as the complex conjugate of bin k = +1. Finally, 9 non-windowed msDFT bins are computed.

TABLE I: i-IpDFT parameters.

Parameter	Variable	Value
Nominal system frequency	f_n	50 Hz
Window length	T	60 ms ($3/f_n$)
Sampling rate	F_s	50 kHz
PMU reporting rate	F_r	50 fps
IpDFT interpolation points		3
msDFT bins		9
DFT bins	K	8
Iterations comp. of the image	P	2 (fund), 1 (interf)
Iterations overall procedure	Q	28
Threshold routine activation	λ	3.3E-3

TABLE II: Maximum i-IpDFT frequency error in mHz in estimating the fundamental tone synchrophasor.

	Sign Freq (Q = 0)	**OOBI** (Q = 28) Inter P = 1	Inter P = 2
Fund P = 1	0.13	10.8	3.7
Fund P = 2	0.11	10.9	3.8

The sensitivity analysis in [5] has shown that it is necessary to iterate the compensation routines P = 2 and Q = 28 times in order to accurately estimate the synchrophasors. In the current implementation we perform the compensation of the negative image of the inter-harmonic tone twice. As regards the fundamental tone, a new sensitivity study reported in Table II has shown that it is sufficient to compensate for its negative image once and still fulfill all the IEEE Std. C37.118 [1] requirements. For this analysis the signal frequency (*Sign Freq*) and the Out-Of-Band Interference (*OOBI*) tests of [1] are performed with signals characterized by a signal-to-noise-ratio (SNR) of 80 dB. The table reports the maximum obtained frequency errors. It is shown that increasing the number of iterations P for the fundamental tone does not lead to a significant accuracy benefit. Instead, having P = 2 for the compensation of the negative image of the interfering tone is necessary to achieve good synchrophasor estimates. In conclusion, P is set to 1 when estimating the fundamental tone, and to 2 when estimating the interfering one. These are

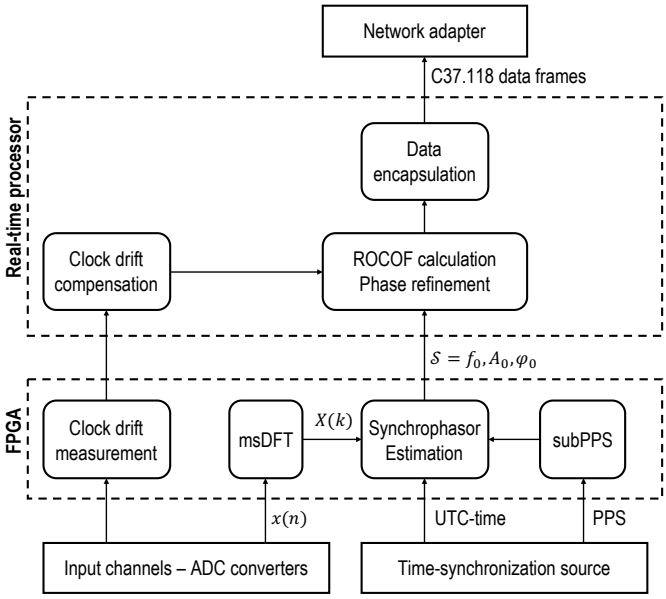

Fig. 1: High level design of the proposed PMU architecture.

the minimum P and Q values that enable us to comply with P and M performance class as a compromise between algorithm accuracy and computation complexity.

In [5] we present a sensitivity analysis that demonstrates that the threshold λ is a discriminatory parameter that enables us to discern the presence of an interfering tone. In order to determine its value, the variability of the normalized spectral energy E_n for tall the operating conditions indicated in [1] should be assessed. Then, a value that enables discerning between the presence of an interfering tone (when the routine should be activated) and the presence of a dynamic (when the routine should not be activated) can be identified. In [5] λ is quantified in 3.3E-3, in order to activate the routine only in presence of interfering components higher than 10% of the fundamental tone.

III. Embedded Hardware Implementation

The i-IpDFT has been fully integrated into an hardware platform that embeds all the key features of a PMU, i.e., (i) synchrophasor estimation, (ii) time synchronization, (iii) data acquisition, and (iv) data streaming. The hardware of the developed PMU is based on the National Instruments compactRIO-9039 controller, integrating (i) a 1.91 GHz quad-core Intel Atom E3845 real-time processor characterized by 16 GB solid state drive nonvolatile data storage and 2 GB DDR3L memory, and (ii) a reconfigurable Xilinx Kintex-7 325T FPGA with an on-board clock frequency of 40 MHz, 407600 flip-flops, 203800 look-up tables (LUTs), 16020 kbits of block RAM and 840 DSP slices [6].

A high level overview of the proposed design is shown in Fig. 1. As it can be seen, the three most significant processes, i.e., signal acquisition, PMU time synchronization and synchrophasor estimation, run at the FPGA level. This choice guarantees that all operations are governed by the FPGA on-board clock that provides hardware-timed speed and reliability, which are essential features for PMUs. Also, thanks to their intrinsic parallelism, FPGAs are particularly efficient when solving recursive problems. A parallel process to synchrophasor estimation measures the eventual drift of the ADC sampling clock. This value, together with the estimated synchrophasors is transferred to the real-time processor through dedicated Direct Memory Access (DMA) channels. This is a common practice to timely share information among the two parts of the cRIO. At the real-time processor level the synchrophasor phase and frequency are corrected for the ADC clock drift and the ROCOF is computed as the finite difference between two consecutive frequency estimates. The data are then encapsulated according to the protocol specified in the IEEE Std. C37.118.2 [18] and streamed out through one of the two available Ethernet adapters to the phasor data concentrator. The whole software, including real-time processor and FPGA, is programmed with NI LabVIEW programming language.

Overall, the workload is higher than the one of the e-IpDFT procedure presented in [10] because of the computational complexity of the proposed iterative routine to compensate interfering tones. Nevertheless, the possibility to share FPGA resources when implementing iterative algorithms, such as the i-IpDFT, enables a considerable reduction of the overall FPGA occupation and the possibility to fit the proposed i-IpDFT in an embedded device. Besides, the cost associated to a larger FPGA platform is marginal with respect to the overall cost of a PMU embedded device, comprising time-synchronization, ADC conversion and CPU processing. The problem that has to be carefully addressed is the time needed to complete the synchrophasor estimation. In this context, the PMU measurement reporting latency is defined as the time delay between the instant a specific event occurs in the PMU input signal and the instant the same event is reported by the PMU. According to the IEEE Std. C37.118 [1], this value is a function of the PMU performance class and reporting rate F_r. In particular, for P-class PMUs the allowed latency is of 2 reporting periods, whereas for M-class ones this value is relaxed to 5 periods. Having fixed F_r to 50 fps, these values correspond to 40 and 100 ms respectively. In order to demonstrate that the proposed PMU is P- and M-class compliant, the most stringent requirement of 40 ms is considered. In the proposed design, an observation interval of 60 ms is adopted and the time-stamp of the synchrophasors is referred to the center of the applied window. Therefore, the windowing of the signal already accounts for 30 ms, leaving only 10 ms of time budget to the processing unit.

The synchronization to the UTC reference can be achieved using three different time dissemination technologies, i.e., the GPS, the PTP and the White Rabbit protocol. Thorough details in terms of operating principle, hardware implementation and performance assessment are provided in [7].

The sampling of the voltage and current waveforms is realized by means of two parallel 24-bits delta–sigma converters, module NI 9225 and 9227 respectively, characterized by a sampling rate F_s of 50 kHz and an input range of 300

TABLE III: FPGA compilation results for computing each step of the i-IpDFT algorithm. The latency and the computational complexity refer to the computation of 1 channel. The column "line" refers to the lines in Algorithm 2.

Function	Config.	line	Flip-Flops	LUTs	DSP	Latency
msDFT	1 bin		10'868	10'162	24	0.7 μs
	9 bin	1				3.1 μs
Hanning	9 bin	2	451	177	0	0.1 μ
IpDFT	3-point		1'356	2'376	15	2.3 μs
e-IpDFT	P = 1	3, 9	7'885	13'316	127	8.1 μs
	P = 2	7				13.7 μs
λ		5	487	254	0	9.9 μs
Spectrum		4, 8, 10	2'508	2'440	60	3.3 μs
i-IpDFT	Q = 28	3 - 14	27'373	41'370	343	0.96 ms
	Q = 0					54.1 μs

TABLE IV: Final FPGA allocation and latency for a 6-channels PMU.

FPGA allocation				Latency [ms]		
Flip-Flops	LUTs	RAM	DSP		1 ch	6 ch
77'246	97'409	107	503	Q = 28	1.14	3.40
19.0%	47.8%	24.0%	59.9%	Q = 0	0.07	0.18

V_{RMS} for the voltage and 5 A_{RMS} for the current [19], [20]. They are equipped with built-in anti-aliasing filters with a pass-band of 0.453 F_s and a stop-band starting at 0.547 F_s (the lobe of the anti-aliasing filter extends below the Nyquist frequency). In particular, the design accounts for 3 voltage and 3 current channels, enabling the sensing of a 3-phase system. The sampling process of the waveforms is free-running and the UTC-time synchronization is achieved *a posteriori*. At the moment of actual deployment into the electrical grid, the A/D converters should be interfaced with the high voltage signals through dedicated instrument transformers.

A. FPGA Design and Resources Allocation

This Section describes the implementation of Algorithm 2 into the target FPGA. Table III presents the FPGA compilation results for each step of the estimation process, considering a single input channel, and Table IV presents the overall FPGA allocation for a six-channels PMU. The results are presented in terms of used number of flip-flops, LUTs, blocks of RAM (one block of RAM can store up to 32 kbits of data) and DSP blocks (one DSP block contains a 18 x 25 multiplier). Also, the execution time of each function is reported. The results are obtained using a standard Xilinx Vivado compiler.

The FPGA receives a continuous stream of samples $x(n)$ from the ADC input channels that are processed over a sliding window of N samples. At every time step n, the DFT is computed using the msDFT technique described in Section II-D. The last N samples are stored in a First-In-First-Out (FIFO) circular memory, and used to compute the difference $x(n) - x(n-N)$ at each time step n. At the same time, a counter increments the value of the index $m \in [0, N-1]$. The 9 msDFT bins are computed for each channel as in (15) and their values are stored in circular memories in the FPGA to be used at the next loop iteration. As indicated in Table III, the time needed to compute a single msDFT bin is 0.7 μs. This action is then serialized 9 times to compute the whole spectrum, by sharing the portion of the FPGA dedicated to the msDFT computation although sacrificing latency. Nevertheless the latency increase is neglibile and limited to 3.1 μs to compute 9 msDFT bins. It is worth pointing out that the msDFT is computed every time a new sample n is received, i.e., at the sampling rate F_s = 50 kHz. Therefore, the time budget to compute the msDFT is of 20 μs, well above the msDFT computation time.

By looking at the definition of the twiddle factor in (3), it is clear that W_N^{km} is characterized by a circular behavior. In particular, for k = 1, W_N^m represents a unitary complex vector whose phase ranges from 0 to $2\pi(N-1)/N$ as m cycles through the range $[0, N-1]$. For k = 2, W_N^{2m} rotates twice as fast and at each m has the same value as W_N^{1m} when m = 2. And so on for every value of $k \in [0, N-1]$. Therefore, in order to cover all the possible twiddle factor values needed to compute the msDFT in (15) for all k bins, it is sufficient to compute W_N^{km} for k = 1 and then select the correct element of W_N^m as k increases. In this view, in the proposed FPGA design the instantaneous twiddle factor values W_N^{km} with k = 1 and $m \in [0, N-1]$ are preallocated in a dedicated RAM circular memory in terms of real and imaginary components. Each time the msDFT is computed, a dedicated counter selects the most appropriate element of the memory.

While the msDFT runs at the ADC sampling frequency, the i-IpDFT runs less frequently, i.e., at the PMU reporting rate. The two processes communicate by means of a shared RAM memory in the FPGA, where msDFT bins are stored according to the rising edge of the signal that triggers the synchrophasor estimation process (see Section III-B). The calculated DFT samples $X(k)$ are then transferred to the process that estimates the synchrophasors. The signal windowing of the acquired msDFT bins is performed in the frequency domain, according to the Hanning windowing weights in (4).

By taking a close look at Algorithm 2, it is possible to notice that the code is composed of a recursive two-stage routine that first estimates the signal parameters via e-IpDFT (line 3, 7, 9) and then, based on these estimates, reconstructs the signal DFT spectrum (line 4, 8, 10). This feature has been exploited in order to efficiently allocate FPGA resources, as shown in Fig. 2. In particular, the code is composed of two consecutive stages that are repeated recursively a predefined number of times R. Specifically, *Stage A* refers to the e-IpDFT that, at each iteration r, processes the input samples $X_r(k)$ and returns the estimated parameters $\{f_r, A_r, \varphi_r\}$. *Stage B* reconstructs spectrum relative to the estimated parameters and subtracts it from the original DFT bins $X(k)$. The result is a new spectrum $X_{r+1}(k)$ that will be processed at the next iteration. In particular, at the first iteration r = 0, *Stage A* processes the original DFT bins $X(k)$. At the next iterations it processes the outcome of *Stage B*. The iterations relative to an even value for r refer to the fundamental tone. In particular, according

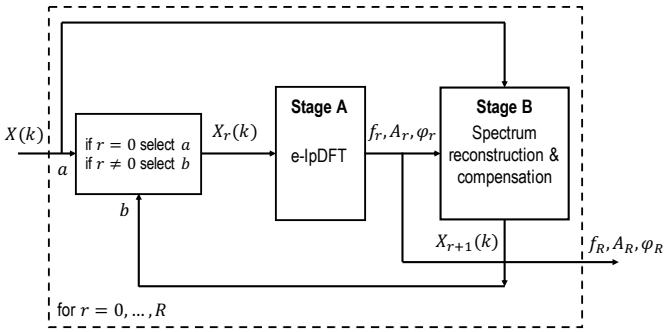

Fig. 2: Block diagram of the FPGA implementation.

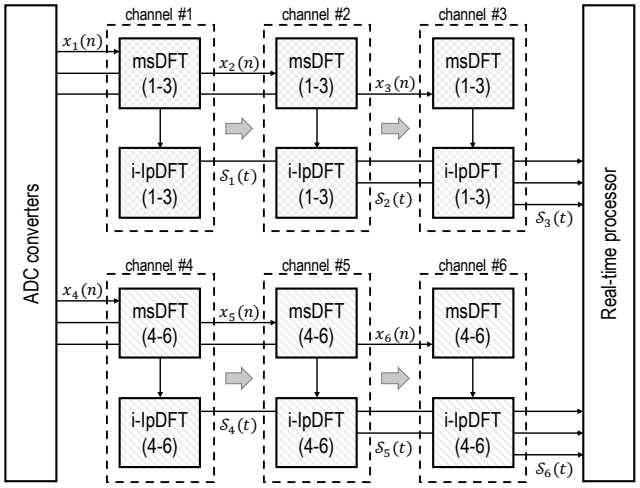

Fig. 3: FPGA allocation. The estimated synchrophasor parameters of each channel i are $\mathcal{S}_i = \{f_0, A_0, \varphi_0\}_i$

to Algorithm 2, *Stage A* can be identified with lines 3 and 9 and *Stage B* with lines 4 and 10. When r is odd, instead, the interfering tone is being estimated and *Stage A* refers to line 7, whereas *Stage B* refers to line 8. The procedure is repeated a predefined number of times R. In order to account for Q = 28 iterations of the routine to compensate the interfering tone, the loop accounts for R = 2Q +1, corresponding to a total of 57 iterations.

The FPGA resources are allocated as shown in Fig. 3. For each PMU channel i, two operations must be computed sequentially in order to estimate the synchrophasors associated to the input signal $x_i(n)$: (i) msDFT and (ii) i-IpDFT. These two actions correspond to the FPGA resources required to execute a single PMU channel. In the proposed design, the selected FPGA can embed twice this amount of calculations. This enables the parallel execution of two PMU channels at the same time. In order to achieve a total amount of 6 PMU channels, these calculations are executed in series three times. As indicated in Fig. 3, channels 1-3 use the top FPGA resources (chessboard pattern) and are executed in series one after the other. In parallel, channels 4-6 use the bottom FPGA resources (striped pattern) and are executed in series. In turns, channels 1-4 are executed in parallel, then channels 2-5 and then 3-6. The choice of executing some operations in sequence, although enlarging the overall time delay, keeps the same amount of allocated FPGA resources independently of the processed number of channels.

The synchrophasors are finally forwarded to the real-time processing units through dedicated DMA channels.

As shown in Table IV, such a two-stage structure enables the efficient allocation of the FPGA resources. The overall FPGA occupation reaches 59% and the final time needed to estimate the synchrophasor of a single input channel is of roughly 1.14 ms. Such a value enables the estimation of the 6 channels in series/parallel leading to a total latency 3.4 ms, compliant with the IEEE Std. C37.118 [1] and IEC/IEEE Std. 60255-118 [2] regulations for P-class PMUs. It is worth to point out that such a long latency is needed only when an interfering tone is detected. Indeed, in case the compensating iterative routine is not activated, the time to estimate the synchrophasors is limited to 0.07 ms per channel and 0.18 ms for 6 channels.

B. The Free-Running Sampling Process

As shown in Fig. 1, at the FPGA level a sub PPS square waveform (hereafter called subPPS) is derived from the UTC-PPS signal and is locked to it. The subPPS is characterized by a frequency corresponding to the PMU reporting rate F_r that in the proposed design is 50 fps. The signal acquisition, the synchrophasor estimation, and the synchrophasor time-stamping are triggered by the rising edge of such subPPS. However, there is no guarantee that the sampling process is locked to such subPPS signal: there must be an *a posteriori* time refinement, that has been proposed in [10].

Specifically, two delays have to be compensated. The first resulting from the fact that the sampling frequency might drift from its nominal value, due to oscillator degradation or environment conditions variation. We measure this drift over observation windows of few seconds (corresponding to M samples). In case of uniform sampling, this window would account for the ideal amount of time MT_s. In reality, instead, the difference between the time instant when the last sample is acquired t_{M-1} and the time instant when the first sample is acquired t_0, might differ from the ideal delay. We define the clock drift f_D as the normalized difference between these two delays, and we compensate for it by updating the frequency resolution Δf in (5):

$$\Delta f_c = \Delta f(1-f_D) = \Delta f \left(1 - \frac{(t_{M-1} - t_0) - MT_s}{MT_s}\right) \quad (17)$$

The second delay is due to the possible offset between the two clocks. Indeed, in ideal operating conditions, i.e., if the sampling process was locked to the subPPS, the time delay between the rising edge of the subPPS t_{subPPS} and the time instant when the first sample of the related window is acquired t_0 would be exactly zero. In real operating conditions, there could be a delay that would result in bad initial phase estimations. This time delay is measured at every subPPS and compensated by updating the estimated phase as follows:

$$\hat{\varphi}_{0,c} = \hat{\varphi}_0 + 2\pi \hat{f}_c(t_0 - t_{subPPS}) \quad (18)$$

IV. Performance Assessment

The performance of the developed PMU is assessed by means of the dedicated PMU calibrator described in [8], that enables us to validate the conformity of the PMU under test with respect to the IEEE Std. C37.118 [1] and the IEC/IEEE Std. 60255-118 [2] in terms of accuracy and measurement reporting latency. In particular, we use the test-bed illustrated in Fig. 4 and for each test we consider an overall test duration of 5 s. The calibrator consists of a forward path, used to generate the reference waveforms, and a return path, introduced to re-acquire the waveforms and estimate their reference parameters through a non-linear least-squares fitting algorithm. The two actions are suitably synchronized to a highly stable timing source. The uncertainty contributions due to generation, acquisition, and synchronization are experimentally characterized in [8], demonstrating that the TVE associated to the reference synchrophasors is in the order of $0.00x\%$ in static and $0.0x\%$ in dynamic conditions.

As illustrated in Fig. 4, the forward path of the calibrator generates a set of static reference waveforms characterized by a sampling rate of 500 kHz and peak amplitude of 10 V. These signals are amplified by a CMS-356 OMICRON precision voltage and current amplifier, characterized by an amplification gain of 30, and simultaneously acquired by the PMU under test [21]. The precision amplifier introduces an unavoidable frequency delay that must be carefully characterized off-line and then compensated on-line in terms of magnitude and phase response [8]. It is worth pointing out that, as highlighted in Fig. 4, the master clock of the PMU and the one of the PMU calibrator use separate GPS receivers, thus guaranteeing the non correlation among their absolute times.

The PMU calibrator integrates a Phasor Data Concentrator (PDC) for PMU latency assessment. The PMU under test encapsulates the measured values into C37.118 data frames, and sends this information to the PDC through a common data switch. The calibrator gathers the estimates provided by the PMUs under test, and aligns them based on their time-stamp. In this way, it is possible to compare reference and measured values associated to the same time-stamp and accordingly determine the uncertainty associated to the devices under test.

The results are expressed in terms of Total Vector Error (TVE), Frequency Error (FE) and ROCOF Error (RFE) and are summarized in Table V. The table further includes the limits imposed by the IEC/IEEE Std. 60255-118 [2] demonstrating that the developed PMU fulfills P and M-class requirements simultaneously. These results further validate the theoretical findings of [5].

It is worth to point out that one of the main differences between P and M-class requirements is represented by the Out-Of-Band Interference (OOBI) test. The latter, has been specifically designed for the M-class, to test the PMU capability to reject superposed inter-harmonics close to the main tone. These are defined as signals characterized by an amplitude that is 10% of the main tone and a frequency f_i in the bands [10, f_n-F_r/2] and [f_n+F_r/2, 2f_n], being f_n the nominal power

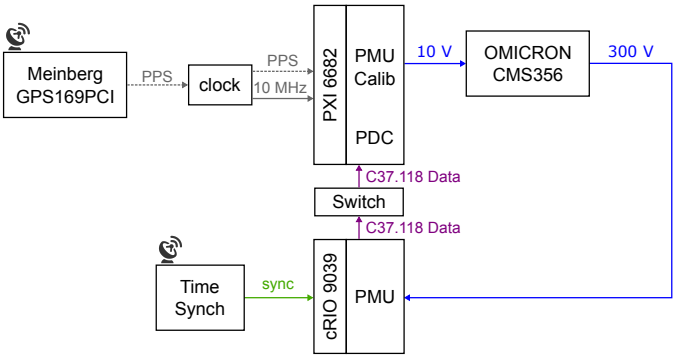

Fig. 4: Measurement setup for the performance assessment. The PMU calibrator generates user-defined test waveforms, that are amplified up to 300 V by the CMS 356 OMICRON amplifier, and then supplied to PMU.

TABLE V: Maximum TVE, FE and RFE for the i-IpDFT and maximum P- and M-class limits allowed by [2].

	TVE [%]			FE [mHz]			RFE [Hz/s]		
	P	**M**	**i-IpDFT**	**P**	**M**	**i-IpDFT**	**P**	**M**	**i-IpDFT**
Sign Freq	1	1	1.7E-3	5	5	0.05	0.4	0.1	6.7E-4
Harm Dist	1	1	2.2E-3	5	25	0.10	0.4	-	5.2E-4
OOBI	-	1.3	1.2E-2	-	10	1.14	-	-	6.7E-2
Ampl Mod	3	3	6.0E-1	60	300	0.70	2.3	14	1.6E-2
Freq Ramp	1	1	3.8E-2	10	10	1.1	0.40	0.2	1.2E-2

system frequency, F_r the PMU reporting rate and $f_i/f_n \notin \mathbb{N}$. The other results in Table V refer to the signal frequency test (*Sign Freq*), the harmonic distortion test (*Harm Dist*), the amplitude modulation test (*Ampl Mod*) and the frequency ramp test (*Freq Ramp*).

The PDC in the calibrator can be used to evaluate the PMU measurement reporting latency and response time, as the PDC time-tags the data frames arrival time. The IEEE Std. C37.118 [1] requires to quantify response time and reporting latency with an accuracy not lower than 2 ms and 100 μs, respectively. As soon as the waveform generation begins, a counter estimates the clock ticks elapsed until each packet arrives. In this way, the reporting latency can be characterized with the resolution of the internal PXI time-base, i.e., 100 ns. It is worth mentioning that the communication network latency is negligible in the proposed PMU calibrator, being the packets streamed over a 0.5 meter Ethernet cable through a passive data switch. Therefore the difference between the data frame arrival time and the data frame time stamp represents the PMU measurement reporting latency. It is also worth pointing out that this design enables taking into account the actual PMU latency, including both the operations at the FPGA level (channel delays, ADC delays, synchrophasor estimation) as well as the operations within the RT processor (clock drift compensation, ROCOF and phase calculation, C37.118 data frames encapsulation and streaming, output of the data frames bits from the Ethernet port).

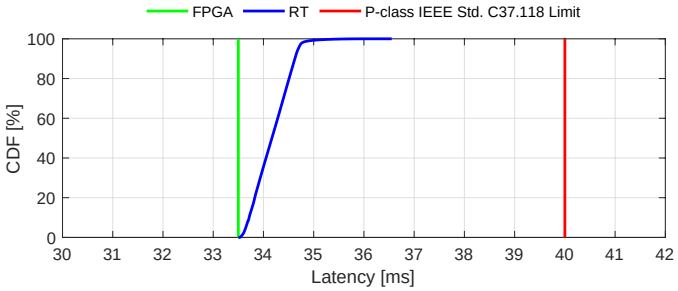

Fig. 5: CDF of the PMU measurement reporting latency.

Figure 5 presents the cumulative distribution function (CDF) of the PMU measurement reporting latency as assessed by the PMU calibrator during a 24 hours test. It is worth pointing out that for this assessment the routine for compensating the interfering tone was activated, in order to quantify the worst case latency. In particular, the latency introduced by the FPGA is deterministic and quantified in 3.4 ms (same as Table III). Then the synchrophasors are transferred to the RT controller, that is characterized by a less-performing level of determinism, with the tails of the CDF that reach up to 36.5 ms. Nevertheless, the totality of the PMU data frames are received by the PMU calibrator within the P-class requirement of 40 ms.

V. Conclusion

The paper presented the design, implementation and experimental validation of a PMU that is simultaneously compliant with both P- and M-class, as defined in the IEEE Std. C37.118. The PMU estimates the synchrophasors using the i-IpDFT, specifically designed to deal with inter-harmonic tones that fall within the PMU pass-band and, therefore, degrade the performance of most synchrophasor estimators. In particular, the interfering components are detected, estimated, and suitably compensated from the original spectrum through a dedicated iterative routine. A window length of 3-cycles at the nominal power system frequency is adopted (60 ms at 50 Hz) in order to successfully handle the high share of spectral leakage, but still distinguish between adjacent tones.

The performance assessment of the proposed PMU has demonstrated that the device fulfills all the requirements defined in the IEEE Std. C37.118, for both P- and M-class at the same time. In particular, the PMU fulfills the OOBI test constraints, with a maximum TVE of 0.1% (being 1.3% the maximum limit allowed by [1]) and the maximum FE is 1.1 mHz (being 10 mHz the maximum limit [1]). As regards the computational complexity, the measurement reporting latency of the developed PMU is of 36.5 ms when an interfering tone is detected (being 40 ms the maximum limit [1]).

References

[1] "IEEE Standard for Synchrophasor Measurements for Power Systems," *IEEE Std C37.118.1-2011 (Revision of IEEE Std C37.118-2005)*, pp. 1–61, Dec 2011.

[2] "IEEE/IEC International Standard - Measuring relays and protection equipment - Part 118-1: Synchrophasor for power systems - Measurements," *IEC/IEEE 60255-118-1:2018*, pp. 1–78, Dec. 2018.

[3] A. J. Roscoe, "Exploring the relative performance of frequency-tracking and fixed-filter phasor measurement unit algorithms under C37.118 test procedures, the effects of interharmonics, and initial attempts at merging P-class response with M-class filtering," *IEEE Transactions on Instrumentation and Measurement*, vol. 62, no. 8, pp. 2140–2153, Aug 2013.

[4] P. Castello, J. Liu, C. Muscas, P. A. Pegoraro, F. Ponci, and A. Monti, "A fast and accurate PMU algorithm for P+M class measurement of synchrophasor and frequency," *IEEE Transactions on Instrumentation and Measurement*, vol. 63, no. 12, pp. 2837–2845, Dec 2014.

[5] A. Derviškadić, P. Romano, and M. Paolone, "Iterative-interpolated DFT for synchrophasor estimation: A single algorithm for P- and M-class compliant PMUs," *IEEE Transactions on Instrumentation and Measurement*, vol. 67, no. 3, pp. 547–558, Mar. 2018.

[6] "National Instruments CompactRIO 9039 Controller," [Online]. Available: http://www.ni.com/en-us/support/model.crio-9039.html, accessed: 2020-04-01.

[7] A. Derviškadić, R. Razzaghi, Q. Walger, and M. Paolone, "The white rabbit time synchronization protocol for synchrophasor networks," *IEEE Transactions on Smart Grid*, vol. 11, no. 1, pp. 726–738, 2020.

[8] G. Frigo, A. Derviškadić, D. Colangelo, J.-P. Braun, and M. Paolone, "Characterization of uncertainty contributions in a high-accuracy PMU validation system," *Measurement*, 2019.

[9] T. Grandke, "Interpolation algorithms for Discrete Fourier Transforms of weighted signals," *IEEE Transactions on Instrumentation and Measurement*, vol. 32, no. 2, pp. 350–355, 1983.

[10] P. Romano and M. Paolone, "Enhanced interpolated-DFT for synchrophasor estimation in FPGAs: Theory, implementation, and validation of a PMU prototype," *IEEE Transactions on Instrumentation and Measurement*, vol. 63, no. 12, pp. 2824–2836, Dec 2014.

[11] ——, "An enhanced interpolated-modulated sliding DFT for high reporting rate PMUs," in *2014 IEEE International Workshop on Applied Measurements for Power Systems Proceedings (AMPS)*, Sept. 2014, pp. 1–6.

[12] V. K. Jain, W. L. Collins, and D. C. Davis, "High-accuracy analog measurements via interpolated FFT," *IEEE Transactions on Instrumentation and Measurement*, vol. 28, no. 2, pp. 113–122, 1979.

[13] F. J. Harris, "On the use of windows for harmonic analysis with the Discrete Fourier Transform," *Proceedings of the IEEE*, vol. 66, no. 1, pp. 51–83, 1978.

[14] D. Agrez, "Weighted multipoint interpolated DFT to improve amplitude estimation of multifrequency signal," *IEEE Transactions on Instrumentation and Measurement*, vol. 51, no. 2, pp. 287–292, Apr 2002.

[15] D. Belega, D. Petri, and D. Dallet, "Frequency estimation of a sinusoidal signal via a three-point interpolated DFT method with high image component interference rejection capability," *Digital Signal Processing*, vol. 24, pp. 162–169, 2014.

[16] D. Belega and D. Petri, "Accuracy analysis of the multicycle synchrophasor estimator provided by the interpolated DFT algorithm," *IEEE Transactions on Instrumentation and Measurement*, vol. 62, no. 5, pp. 942–953, May 2013.

[17] K. Duda, "Accurate, guaranteed stable, sliding discrete Fourier transform [DSP tips & tricks]," *IEEE Signal Processing Magazine*, vol. 27, no. 6, pp. 124–127, 2010.

[18] "IEEE standard for synchrophasor data transfer for power systems," *IEEE Std C37.118.2-2011 (Revision of IEEE Std C37.118-2005)*, pp. 1–53, Dec 2011.

[19] "National Instruments NI 9225 C Series Voltage Input Module," [Online]. Available: http://www.ni.com/en-us/support/model.ni-9225.html, accessed: 2020-04-01.

[20] "National Instruments NI 9227 C Series Current Input Module," [Online]. Available: http://www.ni.com/en-us/support/model.ni-9227.html, accessed: 2020-04-01.

[21] "CMS 356 Voltage and current amplifier," accessed: 2020-04-01. [Online]. Available: https://www.omicronenergy.com/en/products/cms-356/