Pre-print. Accepted for publication, April 2019.
Digital Object Identifier

Experimental Quantification of Hardware Requirements for FPGA-based Reconfigurable PMUs

PROTTAY M. ADHIKARI[1], (Student MEMBER, IEEE), HOSSEIN HOOSHYAR[2](MEMBER, IEEE) AND LUIGI VANFRETTI.[3] (Senior MEMBER, IEEE)
[1]Department of Electrical Computer and Systems Engineering, Rensselaer Polytechnic Institute (e-mail: mondap2@rpi.edu)
[2]Electric Power Research Institute (e-mail: hhooshyar@epri.com)
[3]Department of Electrical Computer and Systems Engineering, Rensselaer Polytechnic Institute (email: luigi.vanfretti@gmail.com)
Corresponding author: Prottay M. Adhikari (e-mail: mondap2@rpi.edu).

ABSTRACT Phasor Measurement Units (PMUs) are becoming intrinsic components of modern power systems. The synchrophasor estimation algorithms in PMUs pose stringent computational demands, which makes the application of Field Programmable Gate Arrays (FPGA) highly attractive. Previous works reported the implementation of PMU algorithms on specific FPGA-targets using a particular PMU design. This paper explores the implementation of different PMU designs on multiple FPGA targets using Xilinx and NI software and hardware infrastructures and toolsets. In this process, a metric has been formulated to predict FPGA-target hardware requirements. The metric allows to predict if an FPGA-target meets the needs to deploy a given PMU design resulting in significant engineering design time savings. Since, compilation/synthesis on FPGAs is a time consuming job, this metric can reduce the implementation time for FPGA based PMUs drastically and can help in determining if additional functionalities can be added.

INDEX TERMS Phasor Measurement Unit , Field Programmable Gate Array .

ABBREVIATIONS

FPGA: Field Programmable Gate Array
PMU: Phasor Measurement Unit
LUT: Look Up Table
BRAM: Block- Random Access Memory
TVE: Total Vector Error
FE: Frequency Error
VI: Virtual Instruments
cRIO: Compact Reconfigurable Input Output
TVE: Total Vector Error
FE: Frequency Error
RFE: Rate of change of Frequency (ROCOF) Error

I. INTRODUCTION

A. MOTIVATION

Field Programmable Gate Arrays (FPGA) have become a very appealing hardware platform for algorithm implementation and embedded controls in various electrical engineering applications. Authors in [16] have explored that FPGAs provide a better throughput when compared to high-performance DSPs. It is also important to note that, FPGAs can typically provide a sampling frequency upto 100 Mhz. i.e. for Xilinx Spartan 6 family of FPGAs the primary clock frequency is 100 MHz. For more advanced FPGAs (e.g. Xilinx Ultrascale Virtex 7 family) the maximum frequency can be as high as 500 MHz [26]. In the current architecture though, the voltage and current sensing is executed via NI C-series modules, which have a maximum voltage and current sampling rate of 50000 samples/second [27]. Because, the computing hardware is capable of functioning at much higher frequency, it is possible to generate data seamlessly in the current setup, without the need of any further optimization at the hardware level. In power system engineering, Phasor Measurement Units (PMU) are used for real-time synchornized measurement of various power system quantities. Given the computational demands of PMU algorithms and their functionalities, the use of FPGAs for PMU implementation has become of recent interest.

Currently, to the knowledge of the authors, there are only a few available implementations for FPGA-based PMUs in the literature. The notable implementations are the Reason

MU320 from General Electric[22], the National Instruments' Advanced PMU Development System which is also used in [16], and another NI based design reported in [2] which is commercialized by Zaphiro Technologies. This paper is limited to the National Instruments (NI) based implementations only. The default National Instruments implementation uses the real-time compact-RIO 9068 with a Xilinx Artix-7 FPGA for their PMU. However, the possibility of implementing the same PMU design across other FPGAs has not been reported, and no guideline for multiple-target hardware implementations is available. This paper investigates the possibility of using the same PMU design on other real-time targets with FPGAs from different families. The same is performed for the PMU in [2].

In this work, a new metric has been formulated to determine the requirements to deploy the two different PMU designs in terms of hardware consumption. These metrics can help to quantify hardware specifications for these PMU-designs, and reduce the design time and effort drastically.

B. RELATED WORKS

The major advantages of using FPGAs for complex system implementation are listed and described in [1]. With the configurable hardware architecture, FPGAs are extremely efficient for implementing high-speed, data-intensive applications. With the inherent re-programming features of the hardware, they can be programmed for any particular application at a much lower level. Hence, as a platform FPGAs provide much better real-time performance. Available research where FPGAs were used to design PMU functions are discussed in [2]-[6]. The implementation in [2] was carried out by National Instruments technologies similar to [3],[4],[6],[14],[15],[16] where some applications and observations for single-platform FPGA-based PMUs are described. Because [2] uses a similar infrastructure as described in this paper, the same PMU design is used herein while characterizing various FPGA platforms. However, none of these previous works, discusses the challenges related to single-design-multiple-platform implementations.

[7], [9] and [25] present an extension of the design in [2], in terms of analysis, estimation and decision making in a power system. In [8], the challenges of protocol implementation for FPGA based PMUs using open source software are reported. [12] and [13], on the other hand, presented the application of FPGAs to implement real time control hardware. A recent report on using the similar infrastructure for PMU implementation has been published by the processor industry-leaders in [16].

Some standard methodologies for bench-marking FPGA cores are described extensively in [28]. In [29], FPGA cores are characterized in relevance to DSP algorithms. In [30], Xilinx Virtex-1 to Virtex-5 families are reviewed in terms of their applicability in critical operations.

C. CONTRIBUTIONS OF THIS PAPER

The main contributions of this paper are:

- To describe methods to deploy PMU designs (implemented using National Instruments' infrastructure) on multiple FPGA targets. This single-design-multiple-platform infrastructure allows easier implementations of real-time PMUs on FPGAs.
- To derive a metric that helps to predict if a PMU design can be implemented and executed on a certain FPGA target. It needs to be noted that this metric is specific to existing PMU implementations only, and not expected to work for any generic FPGA-based digital system.
- The proposed metric is then validated using different FPGA targets for different PMU designs. Both physical hardware and virtual FPGA emulation were used for this validation process.
- Finally, comments are made regarding the cost-effectiveness FPGA based PMU implementations in light of the experimental results along with some of the recent developments in this field by the industry.

D. STRUCTURE OF THE PAPER

In the next section the overall infrastructure of the FPGA based PMU implementation is discussed. The detailed description of the PMU software, hardware, and overall workflow for hardware implementation are described. Case Studies and Experiments are described in Section III which is divided into four subsections, describing test specimens, test-procedures using Physical FPGA hardware, test-procedures with virtual FPGA emulations and lastly, methods for characterization of error. While describing experimental results in Section IV, a metric for projecting the hardware requirements for FPGA based PMUs is proposed. That proposed metric is verified using both physical FPGA targets and virtual FPGA emulation techniques. To test the PMU functions implemented in the physical FPGAs, standard error metrics (Total Vector Error (TVE) and Frequency Error (FE)) were computed. Finally, cost vs capacity trade-off for the selection of FPGAs is explored.

II. PMU SOFTWARE, HARDWARE, AND DESIGN FLOW

This section describes the PMU designs (software), hardware and the design flow methodology used in this study. The generic PMU-implementation provided by National Instruments as a part of their Advanced PMU Development System and the implementation in [2] are used. The PMU designs used are only supported by proprietary Compact Reconfigurable I/O (cRIO) devices. It is important to note that all the experiments and observations presented in this paper were based on M-class PMU designs.

A. SOFTWARE IMPLEMENTATION OF THE PMU

The PMU design was implemented in the LabVIEW Virtual Instrumentation (VI) environment. The organization of each design block is shown in the block diagram in Figure 1. Software components are divided into the host PC and the client compact RIO device. The compact RIO houses a RT-microprocessor and an FPGA. The FPGA target is used to

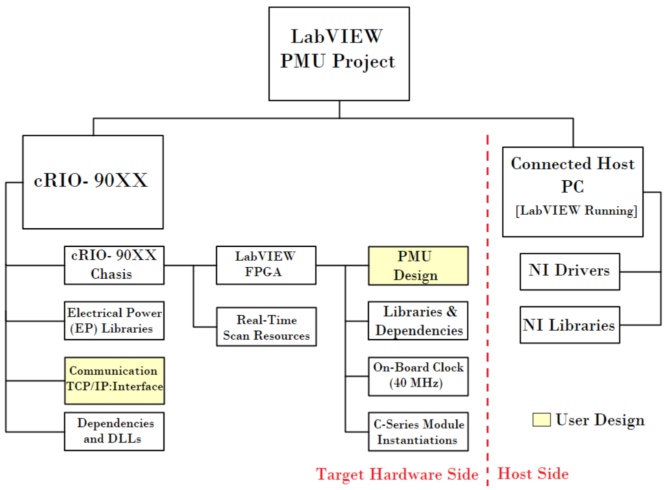


FIGURE 1: Software and Hardware resources of the PMU implementation

implement the PMU design, whereas the rest of the cRIO runs the TCP/IP communication interfaces and performs other auxiliary functions. The overall execution on the cRIO depends on LabVIEW libraries (Electrical Power Library) and DLLs. The cRIO chasis contains the physical FPGA device. The FPGA device incorporates the PMU design, along with FPGA-specific library files from National Instruments, and all the instantiations of the C-series modules for analog to digital signal acquisition.

The host PC communicates with the cRIO (running the PMU and TCP communication), via NI drivers, and uses NI library functions for this communication. The PMU design described so far is supported by the latest LabVIEW release of NI [26], and will be referred as NI PMU from here on. The PMU design from [2] will be referred as Beta PMU.

B. REVIEW OF THE EXISTING PMU-HARDWARE

NI provides a PMU implementation for the cRIO-9068 device (Figures 2 and 3). A cRIO-9068 houses a Xilinx Artix-7 FPGA and an ARM Cortex A9 processor as its real-time processing unit. The 3-phase voltage and current inputs were taken using NI-9225 and NI-9227 C-series modules that are capable of obtaining 50 K samples/second, with 24-bit resolution. The master time is specified by the NI-9467 GPS synchronizer module. When used together with an FPGA, this module synchronizes the internal 40 MHz FPGA time keeper clock within (+/-)100 ns to the GPS 1 PPS received from GPS satellites. All the outputs from these modules are used by the PMU algorithm running on the FPGA hardware.

The FPGA-based PMU communicates with the real-time unit running on the cRIO, which handles initialization, real-time communications, and broadcasting of the PMU output. The PMU output follows the IEEE C37.118-2005 protocol, and is broadcasted through an ethernet port using TCP/IP.

TABLE 1: Hardware Platforms, FPGAs and Processors

Compact-RIO Device	FPGA Family/Device	Real-Time Processor
cRIO 9068	Artix 7 /ZYNQ 7020	ARM Cortex A9
cRIO 9082	Spartan 6/LX 150T	Intel Core i7-660UE
cRIO 9081	Spartan 6/LX 75T	Intel Celeron U3405
cRIO 9074	Spartan 3/ XC3S200	PowerPC

Similar hardware configurations were used in 4 different compact RIO platforms, with different specifications as listed in Table 1. For all of them, the detailed synthesis results provided by the Xilinx Vivado tools were studied and analyzed. One of the PMU hardware platforms used, is shown in Fig 3.

C. HARDWARE DESIGN FLOW METHODOLOGY

In this subsection, the process for designing FPGA based hardware is described. The design flow methodology adopted in this work is described in the flow-chart shown in Figure 4. Note that the flow contains software infrastructure from both National Instruments (LabVIEW libraries) and Xilinx (Vivado synthesis tools). It can be observed that the PMU design is carried out in the LabVIEW Virtual Instrumentation environment from NI. The design is then converted into intermediate HDL (Hardware Description Language) files by NI libraries. Those files are converted into a bit-stream by Xilinx Vivado synthesis tools. This is a lengthy process, and it utilizes the libraries provided by Xilinx. This part of the design-flow consists of four steps: analysis, synthesis, mapping, and place & route. The Xilinx framework is designed to generate a report (pre-synthesis) after analysis, a report after synthesis (post-synthesis) and a last report for the place & route process (post place & route report).

These reports were analyzed to develop the design metrics for two different PMU designs. Note that, for some families of FPGAs, the pre-synthesis report is not generated. Because, this part of the work is time-consuming and computationally intense, National Instruments' online Cloud Compile Service (a high-performance-computing cluster) was used.

The same flow was used for all FPGAs studied and the reports were analyzed to develop metrics that can determine if a PMU design can be put on a certain FPGA. Because the pre-synthesis report takes the least amount of time, the developed metric analyzes the pre-synthesis report for one specific FPGA. Once the metric derived allowed the prediction of the hardware resource requirements, further experiments were conducted using the FPGA emulation for verification.

1) Observations realted to the NI Cloud Compile Service

As mentioned before, the National Instruments high performance computing cluster service (Cloud Compile Service) was used for compilation and synthesis of the designs. It is to be noted that, theoretically this entire procedure can be executed offline. However, the compilation time is usually

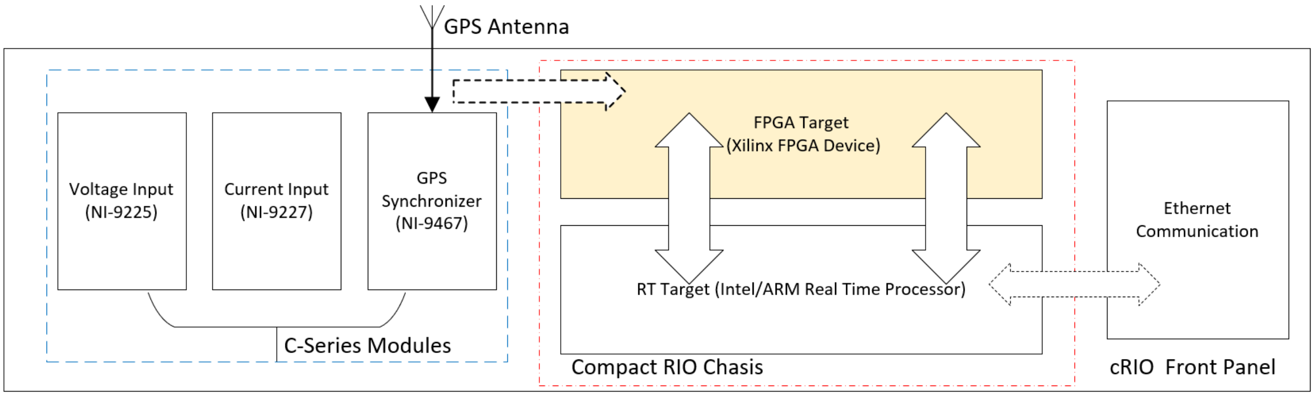


FIGURE 2: Block Diagram for the Hardware Architecture of the PMU Assembly

6-7 times more in a standard work-station (Intel i7, Gen 3 processor with 16 GigaBytes of RAM) when compared to the Cloud Compile Service. Because, the procedure is computationally demanding, sometimes the resources of standalone PCs are not enough to complete the compilation. Hence, it was recommended by NI to always use their compile service for this part of the job. Usually, compilation time on such servers depends on the number of jobs it is handling at a given time. However, in this particular infrastructure, the compilation time was always similar, but the queuing time/ waiting time used to vary depending on the number of compilation jobs it was handling at the given time.

III. CASE-STUDIES AND EXPERIMENTS

The experiments performed were classified in two categories, which are described in subsections B and C below. The test specimens are described in subsection A. The design flow in Fig. 4, was carried out for several targets using the redesigned PMU software and reports were regenerated.

A. TEST SPECIMENS/DESIGNS

Three test-specimens were used to characterize hardware requirements. Two of them were the PMU implementations (NI PMU and Beta PMU described in Section II.(a). Additionally, a simple 64-point FFT design was used across all the platforms for additional validation of the proposed hypothesis. A short description of these specimens is given below.

- Dummy Design: A 64-point FFT algorithm that was expected to be synthesized successfully in all available platforms. It is to be noted that, there is no direct relation between this design and the actual PMUs implemented in the cRIO devices. The results for this design is reported only to infer the fact that even though smaller designs can be synthesized on lower cost hardware with less resources, larger real-life applications such as a PMU requires stronger hardware for implementation. As it will be shown in Section 4, the two PMU designs can not be implemented on some of the targets due to insufficient hardware resources. For those targets it was necessary to use a simpler design (FFT Algorithm) to compare reports generated in all the different platforms.

FIGURE 3: Physical PMU hardware in a Compact-RIO 9068

- NI PMU : The latest PMU design provided by NI, which incorporates all the latest LabVIEW libraries. This design is implemented and released by NI as a part of their Advanced PMU Development System. Their implementation used a ZYNQ-7020 Artix-7 FPGA along with a NI-9068 cRIO and was modified for synthesis in other platforms.
- Beta PMU: This particular implementation [2] was developed for the NI-9076 with a Spartan 6 LX45 FPGA. Although, it has similar functionalities, the hardware requirements are different from the NI PMU design.

It is to be noted that, even though the two PMUs share the same functionalities, the phasor estimation algorithm used was different. The NI PMU utilizes a recursive DFT algorithm for frequency estimation which is well documented by NI in [32]. The Beta PMU, however, utilizes a recently published algorithm - iterative Interpolated DFT (i-IPDFT) for estimating the frequency [2]. As the i-IPDFT algorithm is iterative, and the DFT is recursive, the FPGA hardware requirements are expected to be sufficiently different.

B. IN VIVO EXPERIMENTS ON PHYSICAL FPGA DEVICES

The test specimens described in Section 3.1, were used to synthesize the functions for the physical hardware. The hardware platforms, which were used for this purpose are listed in the Table I.

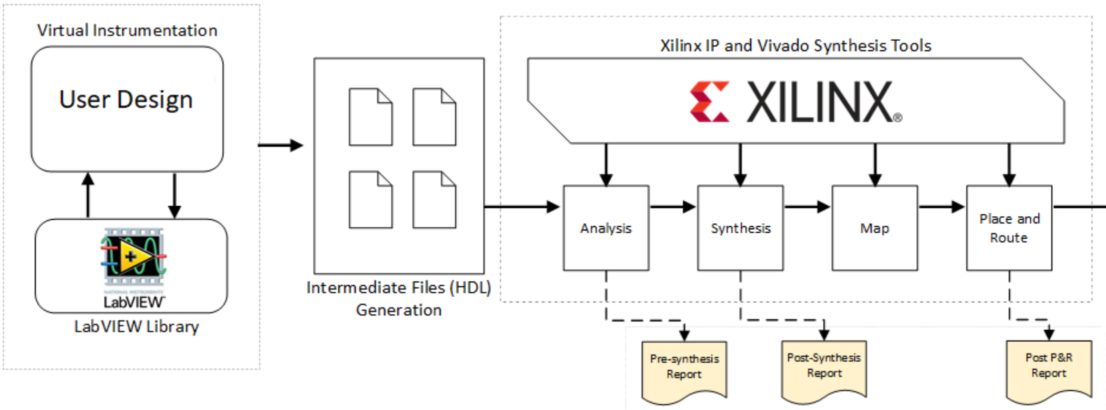


FIGURE 4: The FPGA based Hardware Generation Flow and Report Generation

cRIO displays the detailed results for all the synthesis-jobs carried out. In column 6, the number of LUTs used are displayed, and column 5 displayed the number of available LUT-s in that specific FPGA core. In the case of the cRIO 9082, the FPGA core is a Spartan 6 LX150T which contains 92152 LUTs, each having 6 inputs. Out of those 92152 LUTs, 28826 were used as shown in column 6. Column 10 shows the number of available Block RAMs in that FPGA core. This number is 268 for the Spartan 6 LX150T FPGA. Out of these BRAMs, only 23 were used. As shown in Table 3, the Beta PMU design stresses the Block-RAM (BRAM) demand and the NI PMU stresses the LUT (Look Up Table) demand of the FPGAs. Experiments based on the Dummy (FFT) design were performed to establish that the proposed metric in Section 4 can be extrapolated for different algorithms other than PMUs.

C. IN SILICO EXPERIMENTS USING FPGA EMULATION

National Instruments provides tools to virtually implement a design on an FPGA, without the need to access the physical FPGA hardware. This process is known as FPGA emulation. In this process, Xilinx Intellectual Property(IP) is used to emulate the actual behavior of an FPGA. FPGA emulation was used to predict whether or not the PMU designs can be executed on the other cRIO devices with different FPGA targets and results were compared against the proposed metric. As seen from Table 6(second row), it predicted and verified that for the Spartan 3 XC3S2000 FPGA, the available LUTs are not sufficient to host the PMU design. Based on the proposed metric, it was successfully predicted whether the PMU would fit in for other FPGA cores as well.

D. CHARACTERIZATION

As seen from the flow chart in Figure 4, Xilinx provides important information in different stages of the overall process. The first set of reports is generated after the analysis stage, and requires relatively lower time to generate because it does not involve time consuming stages, i.e. synthesis, place & route and mapping. However, this report only gives a rough estimate of the actual device utilization. Regardless, the entire procedure tends to take hours. It is observed in Table 3 that for a Spartan 6 LX-150T the time for place & route increases by 13.5 fold, and for a Spartan 6 LX-75T it increases 15.6 fold as compared to generating pre-synthesis results. However, pre-synthesis reports are not certain and they do not guarantee that the identified requirements at this stage will remain the same when proceeding with the subsequent stages. The results in Table 2 show that it is extremely inefficient to run a complete synthesis job resulting in unsuccessful execution due to hardware constraints. Therefore the metric proposed in this work uses the pre-synthesis report to predict whether or not a certain design can be implemented and executed on a certain FPGA target as shown next.

TABLE 2: Time Comparison for Report-Generation for various stages

	Pre-Synthesis	Synthesis	Post P & R
Spartan 6 LX 150T	9 Min	22 Min	122 Min
Spartan 6 LX 75T	12 Min	30 Min	187 Min

IV. EXPERIMENTAL RESULTS

A. RESULTS ON PHYSICAL FPGA HARDWARE

1) Hypothesis and Metric Derivation

As mentioned before, three designs were implemented on four physical hardware platforms. The final hardware consumption by these designs across the 4 different FPGA devices are reported in Table 3.

It is to be noted that, the cRIO 9068 is the only device, for which the NI PMU implementation was originally developed and the cRIO 9076 in the case of the Beta PMU. It was observed from the synthesis reports, that the two determining factors that decide whether a design can be implemented on a certain FPGA target are the number of available Look-Up-Tables (LUT-s) and number of available Block-RAM units. The functional blocks such as adders, multipliers, multiplex-

TABLE 3: Synthesis Statistics for Hardware Compilation for Different Designs

1	2	3	4	5	6	7	8	9	10	11	12
Design	Hardware	FPGA Family	FPGA Device	Available LUT-s	LUT-s Used (L)	LUT Size (Inputs) (I)	Available LUT × Input	Consumed LUT × Input	Available BRAM	Used BRAM	Synthesis Outcome
NI PMU	cRIO 9074	Spartan 3	XC3S2000	40960	42892	4	163840	171568	40	26	**Failed**
	cRIO 9076	Spartan 6	LX45T	27288	28823	6	163728	172938	116	26	**Failed**
	cRIO 9081	Spartan 6	LX75T	46648	28824	6	279888	172944	172	23	Passed
	cRIO 9082	Spartan 6	LX150T	92152	28826	6	552912	172956	268	23	Passed
	cRIO 9068	Artix 7	XC7Z020	53200	34909	5/6[a]	319200 [b]	174545	140	21	Passed
Beta PMU	cRIO 9074	Spartan 3	XC3S2000	40960	6539	4	163840	26156	40	48	**Failed**
	cRIO 9076	Spartan 6	LX45T	27288	4324	6	163728	25944	116	48	Passed
	cRIO 9081	Spartan 6	LX75T	46648	4326	6	279888	25956	172	48	Passed
	cRIO 9082	Spartan 6	LX150T	92152	4344	6	552912	26064	268	48	Passed
	cRIO 9068	Artix 7	XC7Z020	53200	17226	5/6	319200	86130	140	33	Passed
64 Point FFT	cRIO 9074	Spartan 3	XC3S2000	40960	4561	4	163840	18244	40	33	Passed
	cRIO 9076	Spartan 6	LX45T	27288	3188	6	163728	19128	116	26	Passed
	cRIO 9081	Spartan 6	LX75T	46648	3193	6	279888	19158	172	29	Passed
	cRIO 9082	Spartan 6	LX150	92152	3215	6	552912	19290	268	29	Passed
	cRIO 9068	Artix 7	XC7Z020	53200	13245	5/6	319200	66225	140	28	Passed

[a]LUTs are of configurable sizes, while most of them were configured as 5 (some as 6) input LUTs
[b]Estimated Maximum Count assuming each of the LUTs were utilized to their fullest extent of 6 inputs

TABLE 4: FPGA Emulation

Device	Physical FPGA target: Compilation Result	FPGA Emulation: Compilation Result	Physical FPGA target: Compilation Time	FPGA Emulation: Compilation Time	Final Result	Comments
Spartan 6 (LX 150T)	LUT = 23851 BRAM = 23	LUT = 23851 BRAM = 23	90 Minutes	4-5 Hours	Physical FPGA target- Success FPGA Emulation- Success	Matched
Spartan 3 2M (XC3S2000)	LUT = 42892 BRAM = 26	LUT = 42892 BRAM = 26	120 Minutes	5-6 Hours	Physical FPGA target- Failed FPGA Emulation- Failed	Matched

TABLE 5: Cross Validation of the Derived Metric with HDL (Verilog) Designs

Design Under Test	Spartan 3 XC3S2000: LUTs Consumed	Spartan 3 XC3S2000: LUT Size	LUT Consumed x LUT Size	Spartan 6 LX75T: LUTs Consumed	Spartan 6 LX75T: LUT Size	LUT Consumed x LUT Size
Square root Finder	27	4	108	19	6	114
64 bit ALU	118	4	472	79	6	474

ers, are all implemented using LUTs and BRAMs through the synthesis process. Hence, the number of LUTs and number of BRAMs should be the most important metric to quantify the hardware consumption on FPGAs for PMU implementation.

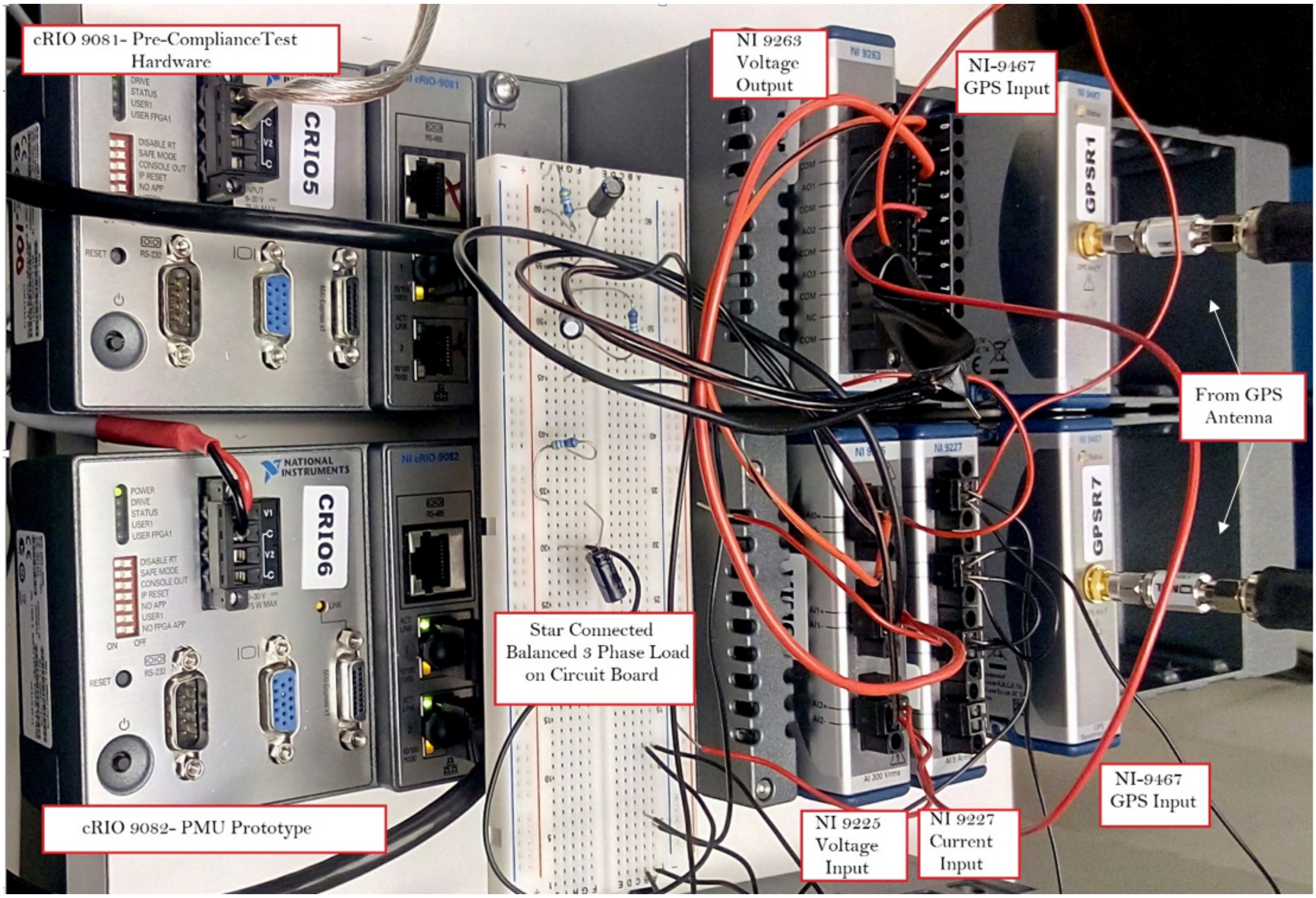
However, the most interesting observation across all the different platforms was that the same PMU design consumed different amount of hardware in terms of LUTs in different FPGA targets, which contradicts the previous assumption to take number of available LUTs as the metric to quantify the hardware consumption.

To address this issue, together with cRIO, detailed specifications of various FPGA families were studied revealing that the size of a single LUT is different among those families. For example, the Spartan 3 family of FPGAs have LUTs of width 4, where as Spartan 6 FPGAs have LUTs of width 6. Another interesting observation was that the product of the LUT size and the number of consumed LUTs for synthesis, were similar across different FPGA-targets for the same design. Consequently, the metric in equation (1) (LUTs × LUT-size) was observed to be the determining factor on whether a design would fit into a certain FPGA. For the NI PMU design, noting the results in column 9 of Table 3, the base value of 175000 was selected as the lower bound to develop the design metric. This choice was made based on the fact that the maximum observed value of $(N_{LUT} \times N_{LUT-Size})$ was 174545 (for Artix 7) across all the tested/emulated hardware (discarding Artix 7, the maximum bound would be 172956). This value was determined by computing the nearest round number above the highest reported hardware consumption (of Artix 7), while determining the exact value of N_{Max}. However, it is not recommended to use this proposed metricw Artix FPGAs .

$$(N_{LUT} \times N_{LUT-Size})_{New-Target} > N_{Max}{}^{1} \quad (1)$$

However, as discussed, Artix 7 has configurable LUT blocks, and hence its synthesis process is more complicated. For such FPGA cores, where pre-synthesis reports are not generated, synthesis reports are considered for our proposed decision making process. Even using synthesis reports, one can save

[1]Where N_{LUT} is the number of available LUT-s and $N_{LUT-Size}$ is the number of inputs in each LUT. N_{Max}=175000



PMU Testing Hardware

PMU Under Test

FIGURE 5: Physical Hardware of the cRIO based PMU and cRIO based PMU testing Hardware

significant amount of time savings by avoiding the time consuming P & R (Place and Route) steps, and post P & R report generation during the hardware selection process. For the Beta PMU, the Block-RAM was the determining factor as it consumes very little space in terms of LUT-s.

It is to be noted that the architecture of Xilinx Artix family is significantly different from the other families. The LUTs of the Artix 7 are re-configurable in nature and can be of size 5 or 6 according to [21]. In fact, a careful evaluation of the reports from Fig. 4 reveals that the synthesis tools actually configures quite a few of those LUTs with smaller (of size 1-4) size. Hence, the proposed metric is perfectly consistent for every target, except when using Artix 7 family of FPGAs. Note that, the Artix 7 implementation is also reported and documented by National instruments in their PMU Development system in [21]. In order to maximize the savings in terms of computational effort and time-consumption, the earliest estimations of device utilization were used to construct the upper bound N_{Max} for the proposed metric. For Artix family, it was provided during the synthesis for all other tested FPGAs

To better incorporate scenarios where the post-synthesis resource allocation is more than the pre-synthesis resource allocation, a pessimistic adjustment to the margin is be proposed. [31] specifies that the accuracy for resource allocation for pre-synthesis analysis by Xilinx has a $\pm 15\%$ error. Keeping that in mind, the proposed metric can be scaled by a factor of 1.15. As a result, equation (1) is modified to

$$\frac{1}{1.15} \times (N_{LUT} \times N_{LUT-Size})_{New-Target} > N_{Max}{}^{2} \quad (2)$$

However, this will lead to a conservative estimate of the ability of the target being analyzed, and reduction in cost/performance analysis; see Section IV.C.

2) Cross-verification of the Proposed Metric with HDL Designs

To investigate the validity of the proposed metric without the need to use National Instruments software tools, two simple digital designs written in Synthesizable Verilog were considered:

- Square-root finder: This was a successive approximation circuit that calculates the square root of an 8 bit number.

[2]Where N_{LUT} is the number of available LUT-s and $N_{LUT-Size}$ is the number of inputs in each LUT. N_{Max}=175000

- 64 bit Arithmetic Logic Unit: This was a 64 bit ALU, with basic arithmetic and logical operations with two 64 bit inputs and one 128 bit output.

Columns 4 and 7 of Table 5, show that the hardware consumption based on the proposed metric is valid for a single digital design across different FPGA cores. Thus, it can be concluded that the proposed metric is legitimate, since it holds for completely different code-generation work-flows with or without the usage of National Instruments' libraries. It needs to be noted, that Verilog codes are at a much lower level of abstraction than LabVIEW VIs, which results in overall much lower hardware consumption.

3) Errors in Different Stages for Physical PMU Implementation

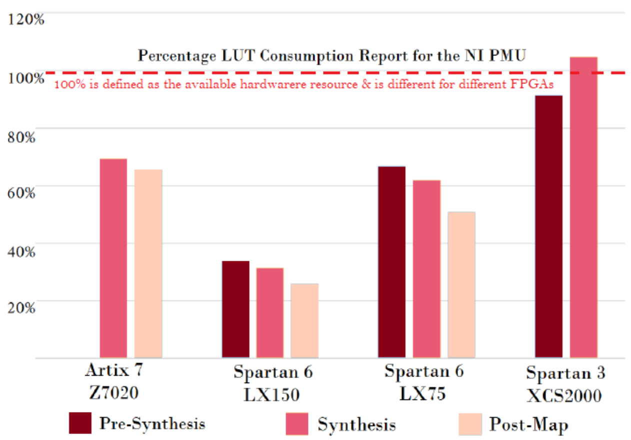


FIGURE 6: Comparison of Pre-Synthesis, Synthesis and Post-Map/PnR results

This section analyzes the accuracy of the intermediate reports depicted in Figure 4. More specifically it is of interest to determine the accuracy of the pre-synthesis report as it takes substantially less time, it will be used by the design metric (1). The time taken by the NI-cloud compile service to complete different stages of synthesis and generate the report is given in Table 2. The NI PMU design was used for the observations.

From Table 2, it is seen that the use of the proposed metric can provide drastic time savings. This is relevant because one can take decisions based on pre-synthesis results, instead of running the entire compilation and synthesis job until completion.

In Figure 6, the results of pre-synthesis and synthesis stage are summarized with respect to the actual post place & route results. This figure is key to understand the need for the proposed metric. The four groups displayed in this bar graph represents 4 physical FPGA cores tested in the experiments. However, it can be seen that for the Artix 7 family, there is no pre-synthesis column, as the compilation procedure for Artix family is entirely different. For Spartan-3 device, the post-map results are not reported because the synthesis job failed during synthesis due to lack of hardware resources.

The bar chart of Figure 6 shows the percentage LUT consumption, the available number of LUTs in each FPGA device is taken in percent. For the second bar chart, the final compilation results (post-map for successful compilations) are taken as the base value. In general, the LUT-consumption varies slightly between pre-synthesis, synthesis and post-map, because of the additional optimizations carried out by the Xilinx tools in each stage. It was observed that, during synthesis, the design is flattened out (resulting into a larger design), optimized and packed, resulting in a reduction of the hardware consumption of the flattened design. In the case of a failed synthesis, the packing and optimization steps are skipped and the reported post-synthesis resource allocation is based on the larger flattenned out design, which is larger than the pre-synthesis resource estimate. This can be observed for the case of Spartan 3 XCS2000 device. (Column 4, in Fig 6)

4) Verifying the functionalities of the PMU implementation

It is obviously necessary to verify that the designed PMU functions still operate as desired in a cRIO after compilation. Hence, the PMU implementation along with the TCP/IP communication protocol reporting the synhrophasor was tested. For this verification, a three phase voltage source along with a balanced star connected load, was used. NI-9263 C series module was used to implement a configurable 3 phase supply, which is fed to the NI-9225 module that is already a part of the hardware implementation of the NI PMU as mentioned in Section 2. The inputs of the current sensing NI-9227 module are connected to the balanced star load designed on a simple circuit board. In this way, both voltage and current phasors are expected to be balanced. It is to be noted that the frequency for all the phasors should be fixed at 50 Hz when in steady state.

The most important metric to characterize the PMU measurements is the Total Vector Error (TVE). For a given complex variable, $X(n)$ the TVE is given by the following expression:

$$TVE = \sqrt{\frac{(X_{Re_M}(n) - X_{Re}(n))^2 + (X_{Im_M}(n) - X_{Im}(n))^2}{((X_{Re}(n))^2 + (X_{Im}(n))^2)}} \quad (3)$$

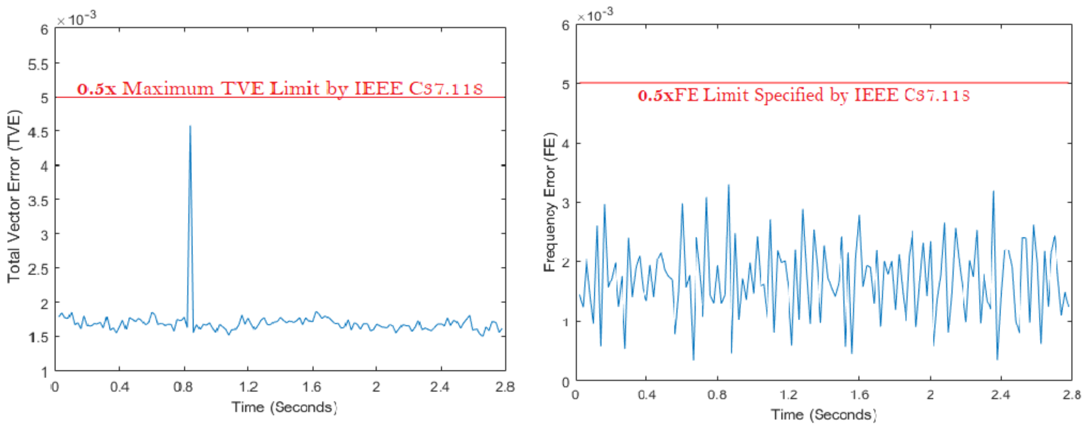
where the variables X_{Re_M} and X_{Im_M} denote the real and imaginary part of that complex variable as measured by the PMU.

It can be seen from Figure 7 (a) that in steady state, the computed TVE for the implemented PMU was less than the specified limit (in [20]). In fact, the TVE was always observed to be within 0.5 times the maximum permissible (by [20]) limit.

TABLE 6: Assessment of the Proposed Metric Based on FPGA Emulation Results For Unknown Hardware

Compact RIO	FPGA Family	FPGA Device	LUT Count(A)	LUT Size(B)	A × B	BRAM Count	Compilation Prediction	FPGA Emulation Results	Matched	Percentage Consumption
cRIO-9030	Kintex 7	XC7K70T	41000	6	244000	24	Success	Success	Yes	69 %
cRIO-9072	Spartan 3	XC3S1000	17280	4	69120	24	Failure	Failure	Yes	200 % +
cRIO-9113	Virtex 5	LX-50	28800	6	172800	48	Uncertain[a]	-	-	98.9 %
cRIO-9032	Kintex 7	XC7K160T	101400	6	608400	48	Success	Success	Yes	27.1 %
cRIO-9036	Kintex 7	XC7K70T	41000	6	244000	24	Success	Success	Yes	69.1 %
cRIO-9035	Kintex 7	XC7K70T	41000	6	244000	24	Success	Success	Yes	69.1 %

[a]This was a marginal case with the design consuming 99 % of the available resources



(a) TVE of the PMU at Steady State Operation

(b) FE of the PMU at Steady State Operation

FIGURE 7: Total Vector Error and Frequency Error exhibited by the PMU through TCP/IP network during steady state operation

The implemented PMU was able to compute the signals' frequency and broadcast that information to a workstation connected to the network. An important index to analyse the quality of these measurements is the Frequency Error (FE), which is given by the following expression

$$FE = |f_{Measured} - f_{Actual}| \tag{4}$$

For the test-case, Frequency Error (FE) is computed and plotted based on the instantaneous frequency measurements and shown in Figure 7 (b). It can be seen that FE is in the range of 10^{-2} order, which is well within the permissible limits set in [20].

The results presented in Fig. 7 clearly shows that the implemented FPGA based PMU works well under steady state. To test it further, the standard compliance tests were performed on it, using the same cRIO hardware.

In table 8, the Frequency Errors, ROCOF Errors, and Total Vector Errors are reported for steady state tests and all the dynamic compliance tests suggested by [24]. The dynamic compliance tests as suggested in the literature are (a) Step Increase in Magnitude and Phase, (b) Sinusoidal Modulation of amplitude and phase (known as Bandwidth Test), and (c)

TABLE 7: Specifications for the Pre-compliance Tests Performed to validate the PMU

Test Name	Specifications
Steady State Test	Balanced 3 phase, 10 V, 50 Hz
Step Change Test	10^o Change in phase, 0.1 pu change in magnitude, both across all the phases
Modulation Test	Amplitude and Phase angle modulation with modulation index= 0.1
Frequency Ramp Test	1 Hz/Sec

TABLE 8: Validation of PMU functionality for NI-PMU using standard compliance tests

Experiment Type	Steady State Compliance Tests				Dynamic Compliance Tests										
					Step Change Test				Bandwidth Test			Frequency Ramp Test			
Test	TVE	FE	RFE	Phase Angle Error	TVE	FE	RFE	Delay	TVE	FE	RFE	TVE	FE	RFE	Delay
Observation	0.001	0.001Hz	4.5 Hz/s	0.002	0.08	0.009	0.08	0.0025	0.01	0.05 Hz	12 Hz/s	0.0075	0.0045 Hz	0.45 Hz/s	0.002
IEEE Spec	0.01	0.005 Hz	6 Hz/s	0.01	0.199	0.12	0.134	0.005	0.03	0.3 Hz	30 Hz/s	0.01	0.005 Hz	0.05 Hz/s	0.005
Comment	Passed				Passed										

Ramp Increase of Frequency. The specifications of these tests summarized in table 7.

B. FPGA EMULATION RESULTS

With the actual compilation results from four physical FPGA targets, the metric described previously was developed. It was quite successful in predicting if the FPGA targets were capable of supporting different designs. For further verification, the metric needs to be tested using a few other FPGA cores. However, it is costly to test the same design on every other physical hardware.That is why FPGA emulation was used to further test the metric. Before beginning this verification, it is necessary to determine the accuracy of the FPGA emulation tool.

Table 4 clearly exhibits that FPGA emulation tools were sufficiently accurate in mimicking the performance of a real FPGA target. In fact, the detailed synthesis results for FPGA emulation matched perfectly with the synthesis results of the physical FPGA targets, when the synthesis was run for the same device.

Using the proposed metric it was now possible to predict the prospective synthesis results for other FPGA targets, for which physical hardware were not available. It can be seen from Table 6, that in each of the test cases the proposed metric was successful in determining whether the PMU design can be implemented on the test-FPGA target.

In Table 6, the last two rows suggest that, the compilation job, when applied to the same FPGA hardware across different compact RIO hardwares, produces exactly same results. In this particular case results, from cRIO 9035 and cRIO 9036 (both of them contain the same Kintex 70T FPGA) are reported.

C. FPGA SELECTION

In this subsection, Xilinx FPGA devices are compared in terms of their cost and projected resource consumption based on the proposed metric. This set of results is an example of how this metric can be used in practice for FPGA selection. The graph in Fig. 9 shows how the cost and projected resource consumption for PMU implementation varies across various FPGA devices of the same family. This particular test case shows results from all the devices of Spartan 6 family.

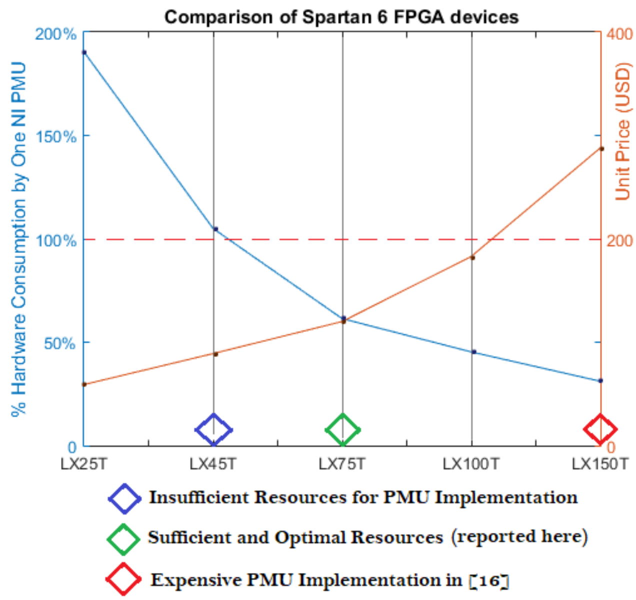


FIGURE 8: Xilinx Spartan 6 FPGA family IC's Variation of cost and hardware resources

It is to be noted, that the price of only the FPGA IC is reported in this paper. Similar FPGA ICs are also available, embedded in a development board (manufactured by both Xilinx and other third party vendors) with multiple connectivities at significantly higher prices, which are not reported in this paper.

V. CONCLUSION

In this paper, the methods for deploying the same PMU design on different FPGA targets were discussed. That makes the single-design-multiple-target infrastructure easier to implement. In that process, a metric was developed to predict the FPGA-hardware requirements for different PMU implementations.

It is clear that, the metric proposed in the current work gives accurate prediction of the FPGA hardware requirement with satisfactorily lower effort. The experimental results show that the usage of the metric can drastically reduce the implementation time for FPGA-based PMU-s, specially in low to medium capacity FPGA targets. In the case of the cRIO 9036, it was predicted using the metric that the PMU can be implemented in the hardware and it would approximately consume 175000/244000 = 71% of the hardware. Where as, the experimental results in Table 6 shows that the PMU was successfully implemented in the hardware, and it utilized 69.1% of the hardware. This implies that, if a new function requiring the FPGA and consuming up to 29% of the resources was added, the target would be able to support the function. Finally, it can be concluded that the methodology proposed, provides a systematic way to chose FPGA targets for PMU implementations considering the tradeoff between cost and hardware resources.

REFERENCES

[1] R. Tessier, K. Pocek and A. DeHon, “Reconfigurable Computing Architectures,” in Proceedings of the IEEE, vol. 103, no. 3, pp. 332-354, March 2015.

[2] P. Romano, M. Paolone, T. Chau, B. Jeppesen and E. Ahmed, “A high-performance, low-cost PMU prototype for distribution networks based on FPGA,” 2017 IEEE Manchester PowerTech, Manchester, 2017, pp. 1-6.

[3] A. Agarwal, N. Verma, H. Tiwari, J. Singh and V. Maheshwari” Design and development of phasor measurement unit on FPGA,” 2016 IEEE Region 10 Humanitarian Technology Conference (R10-HTC), Agra, 2016, pp. 1-6.

[4] S. Ramadhan, F. I. Hariadi and A. S. Achmad, “Development FPGA-based Phasor Measurement Unit (PMU) for smartgrid applications,” 2016 International Symposium on Electronics and Smart Devices (ISESD), Bandung, 2016, pp. 21-25.

[5] A. L. Akhila, R. Micky, A. Mohemmed, E. P. Cheriyan and S. Ashok,“Implementation of an algorithm for phasor estimation and system parameters for an FPGA based PMU”, 2017 IEEE International Conference on Signal Processing, Informatics, Communication and Energy Systems (SPICES), Kollam, 2017, pp. 1-6.

[6] S. Karn, A. Malkhandi and T. Ghose, “Laboratory prototype of a phasor measurement unit using FPGA based controller,” 2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT), Chennai, 2016, pp. 2029-2034.

[7] P. Romano, M. Paolone, J. Arnold and R. Piacentini, “An interpolated-DFT synchrophasor estimation algorithm and its implementation in an FPGA-based PMU prototype,,” 2013 Power and Energy Society Annual Meeting, Vancouver. pp 1-6.

[8] A. Mohemmed, R. Mickey, A. L. Akhila and S. Ashok, “Open source implementation of IEEE C37.118-2011 protocol and serial communication interface for an FPGA based phasor measurement unit,,” 2017 2nd IEEE International Conference on Recent Trends in Electronics, Information and Communication Technology (RTEICT), Bangalore, 2017, pp. 308-313.

[9] P. Romano and M. Paolone, "Enhanced Interpolated-DFT for Synchrophasor Estimation in FPGAs: Theory, Implementation, and Validation of a PMU Prototype," in IEEE Transactions on Instrumentation and Measurement, vol. 63, no. 12, pp. 2824-2836, Dec. 2014.

[10] E. Rebello, L. Vanfretti and M. Shoaib Almas, "PMU-based real-time damping control system software and hardware architecture synthesis and evaluation," 2015 IEEE Power and Energy Society General Meeting, Denver, CO, 2015, pp. 1-5.

[11] J. Lu, P. Siqueira, V. Vijayendra, H. Chandrikakutty, and R. Tessier, Real-Time Differential Signal Phase Estimation for Space-based Systems Using FPGAs, in IEEE Transactions on Aerospace and Electronic Systems, vol. 49, no. 2, April 2013, pp. 1192-1209.

[12] E. Monmasson and M. N. Cirstea, "FPGA Design Methodology for Industrial Control SystemsâĂŤA Review," in IEEE Transactions on Industrial Electronics, vol. 54, no. 4, pp. 1824-1842, Aug. 2007.

[13] Scott Hauck, AndrÃľ DeHon ”Reconfigurable Computing: The Theory and Practice of FPGA-Based Computation ”. Morgan Kauffman Publishers, Elsevier.

[14] M. Pignati et al., "Real-time state estimation of the EPFL-campus medium-voltage grid by using PMUs," 2015 IEEE Power and Energy Society Innovative Smart Grid Technologies Conference (ISGT), Washington, DC, 2015, pp. 1-5.

[15] A. Katsimichas, P. Brogan and D. Laverty, "XML data interface and PMU testing applications for synchrophasor estimation with a reconfigurable controller," 2014 49th International Universities Power Engineering Conference (UPEC), Cluj-Napoca, 2014, pp. 1-5.

[16] https://www.intel.com/content/dam/www/public/us/en/documents/solution-briefs/energy-smart-grid-solution-blueprint.pdf

[17] V. Pallares-Lopez, A. M. MuÃśoz, A. Gil-de-Castro and I. Santiago-Chiquero, "FPGA-based embedded system architecture for power quality measurements," 2012 IEEE 15th International Conference on Harmonics and Quality of Power, Hong Kong, 2012, pp. 507-511.

[18] A. Moreno-Munoz, V. Pallares-Lopez, J. J. Gonzalez de la Rosa, R. Real-Calvo, M. Gonzalez-Redondo and I. M. Moreno-Garcia, "Embedding Synchronized Measurement Technology for Smart Grid Development," in IEEE Transactions on Industrial Informatics, vol. 9, no. 1, pp. 52-61, Feb. 2013.

[19] P. Romano, M. Paolone, T. Chau, B. Jeppesen and E. Ahmed, "A high-performance, low-cost PMU prototype for distribution networks based on FPGA," 2017 IEEE Manchester PowerTech, Manchester, 2017, pp. 1-6.

[20] Q. Guo and Rui Gan, "An Arbitrary-Resampling-based synchrophasor measurement algorithm in compliance with IEEE Std C37.118.1a-2014: Design, implementation, and validation," 2016 IEEE/PES Transmission and Distribution Conference and Exposition (T&D), Dallas, TX, 2016, pp. 1-5.

[21] IEEE Standard for Synchrophasor Measurements for Power Systems-IEEE Std C37.118-2011, IEEE Power & Energy Society

[22] https://www.xilinx.com/products/silicon-devices/fpga/spartan-6.html

[23] https://www.gegridsolutions.com/multilin/catalog/mu320.htm

[24] Technique for Pre-Compliance Testing of Phasor Measurement Units International Journal of Electrical Power and Energy Systems Brogan, P.V.; Laverty, D.M.; Zhao, X.; Hastings, J.; Morrow, D.J.; Vanfretti, L.; "Technique for Pre-Compliance Testing of Phasor Measurement Units", IJEPES, vol. 99, pp. 323-330, July 2018

[25] G. Frigo, A. Derviskadic and M. Paolone, "Reduced Leakage Synchrophasor Estimation: Hilbert Transform Plus Interpolated DFT," in IEEE Transactions on Instrumentation and Measurement.

[26] https://www.xilinx.com/support/documentation/selection-guides/ultrascale-fpga-product-selection-guide.pdf

[27] http://www.ni.com/en-us/support/model.ni-9225.html

[28] Khatri, Abdul. (2016). Selecting right FPGA for the right application: a technical survey for Xilinx FPGAs. Quaid-e-Awam University Research Journal of Engineering, Science Technology. 15. 46-49.

[29] R. Selow, H. S. Lopes and C. R. E. Lima, "A comparison of FPGA and FPAA technologies for a signal processing application," 2009 International Conference on Field Programmable Logic and Applications, Prague, 2009, pp. 230-235.

[30] H. Quinn, K. Morgan, P. Graham, J. Krone and M. Caffrey, "A review of Xilinx FPGA architectural reliability concerns from Virtex to Virtex-5," 2007 9th European Conference on Radiation and Its Effects on Components and Systems, Deauville, 2007, pp. 1-8.

[31] https://www.xilinx.com/support/documentation/sw-manuals/xilinx11/PlanAheadUserGuide.pdf

[32] http://sine.ni.com/cs/app/doc/p/id/cs-13305

[33] S. Mondal, C. Murthy, D. S. Roy and D. K. Mohanta, "Simulation of Phasor Measurement Unit (PMU) using labview," 2014 14th International Conference on Environment and Electrical Engineering, Krakow, 2014, pp. 164-168.

[34] http://cigre-usnc.tamu.edu/wp-content/uploads/2015/06/Virtual-Instrumentation-Based-PMU-Calibrator-for-IEEE-C37.118.1-2011-Compliance-Testing.pdf

[35] http://sine.ni.com/cs/app/doc/p/id/cs-13305

[36] http://zone.ni.com/reference/en-XX/help/373375G-01/lvept/ep-syncphasor-pal/

PROTTAY M. ADHIKARI received his Masters' in Electrical Engineering from Indian Institute of Technology, Bombay (IIT Bombay) in 2016, and his Bachelors' in Electrical Engineering from Indian Institute of Engineering Science and Technology, Shibpur (IIEST Shibpur) in 2013. He is currently a doctoral student at Renssalaer Polytechnic Institute (RPI), Troy, NY, USA. From 2016 to 2017 he worked in the Semiconductor Industry, at ARM.

His research interest is in Embedded Systems, Smart Grid, Hardware-Software Co-design and Co-Simulation.

HOSSEIN HOOSHYAR R (S 00, M 14) received the Ph.D. degree in electrical engineering from North Carolina State University, Raleigh, NC, USA, in 2012. He was a Researcher and a PostDoctoral Associate with the KTH Royal Institute of Technology, Sweden, from 2013 to 2017. He was a Research Associate at the Rensselaer Polytechnic Institute, Troy, NY, USA, since 2017. His research interests include digital simulation of power systems, integration of renewable energy resources, and applications of phasor measurement unit data for smart grids.

LUIGI VANFRETTI (SM 14) received the M.Sc. and Ph.D. degrees in electric power engineering from the Rensselaer Polytechnic Institute, Troy, NY, USA, in 2007 and 2009, respectively.

He was with the KTH Royal Institute of Technology, Stockholm, Sweden, as an Assistant from 2010 to 2013 and an Associate Professor (Tenured) and a Docent from 2013 to 2017, where he led the SmarTS Lab and research group. He was a consultant with Statnett SF, the Norwegian electric power transmission system operator, from 2011 to 2012, where he was a Special Advisor in research and development from 2013 to 2016. He joined the Rensselaer Polytechnic Institute in 2017, where he is currently a Tenured Associate Professor. His research interests are in the areas of synchrophasor technology applications and cyber-physical power system modeling and simulation.