

Industry 4.0 CAMI: An Elastic Cloud Zynq UltraScale FPGA Metering Architecture

K. C. Okafor[1(✉)] and A. A. Obayi[2]

[1] Department of Mechatronics Engineering, Federal University of Technology, Owerri, Nigeria
kennedy.okafor@futo.edu.ng
[2] Department of Computer Science, University of Nigeria, Nsukka, Nigeria

Abstract. In the era of Industry 4.0, commoditized services (such as electricity, healthcare, etc.), require accuracy, availability, security and Quality of service (QoS) in the Cloud space. This paper presents cloud advanced metering infrastructure (CAMI) using Zynq UltraScale+ device field programmable gate array (FPGA). The architectural layout for energy tracking and profile measurement is discussed. Unlike existing systems with digital signal processors, it uses precision-based meter reading with encryption driven demand side management (DSM) to protect end-users. An energy service application with supporting hardware prototype is designed. Cryptographic algorithms, dynamic auto-scaling and predictive QoS provisioning are introduced as features of its backend cloud virtualization Infrastructure controller (CVIC). Process integration is achieved with CVIC synthesis for energy analytics and DSM. For the use case scenario, the CAMI prototype runs on Zynq UltraScale+ device with support for end-to-end dataset captures. The system provides on-demand visualization of energy consumption patterns for the end-users. In the experimental setup, two case scenarios demonstrate how the metering system executes fast edge computing profiling. Optimal performance is achieved for latency, utilization, and throughput under CVIC overhead constraint. It was observed that resource utilization responses for heterogeneous and non heterogeneous CVIC are 71.43% and 28.57% respectively. Latency profiles gave 71.43% and 28.57% respectively. With the VM controller, FPGA CAMI offered 47.36% while without VM controller, 52.63% throughput is observed. Consequently, the results highlights how FPGA hardware acceleration can significantly improve request distribution as well as workload processing for cloud based metering systems.

Keywords: Industry 4.0 · Cloud metering AMI · FPGA device · Demand side management · Cloud analytics

1 Introduction

1.1 Background

CAMI is the next digital future for energy management (EM) systems. In the domain of Cloud driven AMI, modern smart grid combines highly intelligent digital components

© Springer Nature Switzerland AG 2020
O. Gervasi et al. (Eds.): ICCSA 2020, LNCS 12254, pp. 527–543, 2020.
https://doi.org/10.1007/978-3-030-58817-5_39

with full duplex two-way communication to achieve seamlessness on- demand response [1, 2]. The essence is to create efficient monitoring and management of energy usage data efficiently. Precision-based sample captures, smart-consumption billing, intrusion alerts, negative-tampering feature, load isolation, on-line upgrades, DSM, and analytics, among others are desired features [3]. Developing countries like Nigeria will immensely benefit from the system looking at it from cost benefit and service delivery perspectives.

Unlike the legacy smart grid AMIs, an ultra-scaled AMI provides unbiased data profile in respect of instantaneous user consumptions. This simplifies billing automation with lower computational overhead. In most smart meters (SMs), metering and communication modules are the two most essential components. Both have specific controller units relating seamlessly during its operation. Also, these meters have load enable/disable switch, input-output (I/O) control interfaces with embedded power storage subsystems, among others. Modern designs are expected to enhance energy efficiency, reduces wastages and control usage pattern by consumers. The major difference between the recent smart metering systems and the legacy metering models is that the CAMI enforces active participation of consumers in energy conservation through real time feedback communication on their exact energy consumption profiles [4].

Various efforts on EM capabilities are mainly skewed towards embedded device architectures that support sequential rather than concurrent execution. The introduction of FPGA's into CAMI offers a disruptive strategy that have recently gained attention in today's energy industry. In the years ahead, smart metering designs with FPGA devices and digital inclusive technologies such as the Internet cloud will be relied on to provide new efficiency in energy metering and billing systems. This will obviously make CAMI to fully alter the way metering is carried out. In context, this will allow customers to participate fully in demand side management schemes [4].

Till date, most metering systems does not support low latency full duplex communication for real time energy feedback. Low latency/high speed processing capabilities is yet to be fully implemented and deployed leveraging bit stream encryption and decryption. This research presents low latency based CAMI which depends on high speed Zynq UltraScale+ FPGA processor (32/64bits compatibles). The major attraction of the proposed system is the secured computational executions running on the FPGA core with little power drain while having low delay real-time clock, signal conditional converter and other RF interfaces. For end users, the advantages of CAMI include overhead reduction in data gathering, improved reading precision, efficient billing processes, and significant increase in operational efficiency through real time metering activities. This paper presents CAMI implementation for distributed energy resources (DERs) in smart grid ecosystem. The 32 bit FPGA core design strategies, security and low latency implementations are discussed. Hardware acceleration via FPGA CAMI consumer workload is equally demonstrated as the most efficient scheme for smart grid DSM.

2 Related Works

In this Section, various efforts on advanced metering systems will be presented.

Recently, the authors [4, 5] highlighted essential benefits of AMI for an energy driven society. These include: low latency energy data-stream updates; cumulative data-sets gathering from various home sources with passive backend data analytics; secured data privacy for users; bandwidth management of sensitive workloads in full duplex communication mode and trajectory data-mining and context-aware service provisioning. The work [3] presented an implementation of FPGA sigma delta architecture for the analog to digital conversion module in smart meters. The authors [6] presented a transceiver architecture that depicts resource estimations for application specific integrated circuit (ASIC) AMI. The work [7] presented a typical power measurement device whose computational units are embedded in an FPGA device. The authors [8] demonstrated a prototype implementation of smart metering utility networks using Altera's device. The authors [9] presented electric power measurement implementation in a three-phase power system by using an FPGA device architecture whose power calculation theory was verified with IEEE 1459-2010 Standard for power quality measurement methods [10]. In terms of FPGA processor architectures, various works have been carried out to address speed performance issues. For instance, the work [11] presented FPGA 32-bit extended arithmetic logic unit (ALU) architecture with Xilinx Vivado 14.4 tool and implemented on 28 nm Zynq 7000 FPGA board. The work in [12] presented a 32-bit MIPs FPGA-Based pipelined microprocessor with Very high speed hardware descriptive language (VHDL) [13]. The authors [14] presented an implementation of a 32-bit Microprocessor without Interlocked Pipeline Stage (MIPS) using rotation based algorithm. Other similar works on 32 bit FPGA designs have been studied and presented in [15]. With the FPGA device board [16], the design can be used create, implement, and test CAMI AMI design using programmable logic.

Considering FPGA based Cloud security designs, the work [17] proposed an FPGA Cloud which deals with privacy preservation and computing in the public Cloud domain. The authors [18], discussed a security pattern which supports the control-sharing and demarcation of FPGA accelerators in heterogeneous Cloud architectures. Similar work [19] focused on various advanced encryption security model at the FPGA layer without considering the latency profiles of these metering systems.

The research gaps in the existing Cloud based metering system involve ineffective DSM and the absence of concurrent hardware resource utilization due to unsatisfactory latency profiles. Also, FPGA Virtual machine provisioning for energy users has not been explored for scalability. From the literature carried out, the proposed CAMI offers a robust design with well-structured algorithm for security. Hardware acceleration using FPGA- Cloud service for optimal metering and billing has not been implemented in existing systems. In this work, CAMI is presented and it uses bit stream privacy while supporting energy traffic confidentiality with a deterministic load balancing.

3 FPGA Device Architecture

Clearly, the FPGA device architecture is perceived to have strong potential for optimal design of AMIs. This has been further buttressed in [20]. The authors identified the merits of using FPGA-ASIC namely:

- re-programmability for bug correction.
- FPGA development allows shorter time to market and reduces unnecessary costs.
- reduction in power drain.
- reduces component counts significantly.

Hence, using FPGA AMI with an isolated CPU chip often yield cost savings, miniaturized system, and reliability. Figure 1 shows the I/O ports highlighting the board layout with indications on the location and connections of various components. Quartus II IDE is used to configure the board alongside the downloaded USB blaster drivers.

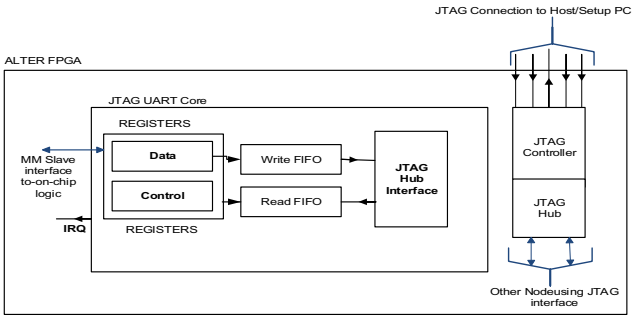


Fig. 1. Altera FPGA interface layout.

There are essential soft processor-core features in any commercial FPGAs. These include [21, 22]: gate count, clock frequency, data-path-width, pipeline- stages, register-files, instruction word, instruction cache, hardware ALUs among others. To achieve satisfactory CAMI, low latency FPGA design is absolutely needed. This can be achieved with electronic digital automation process (EDA) [23]. An earlier work in [4] shows the EDA process framework for CAMI on FPGA core processor (Zynq UltraScale+). The reason for selecting the processor in Fig. 1 (during system implementation) is as a result of cost and market availability considerations. The beneficial features of using the FPGA CAMI in the work include: workload tolerance, fair resource utilization, latency and throughput scaling supports, power efficient compared with CPUs/GPUs, output optimization. As such, the Zynq UltraScale+ Xilins MPSoC [24] was explored to achieve computational performance in the cloud domain. Section 4 introduces the CAMI context.

4 Cloud Metering Description

In this Section, a brief description of the CAMI is given. Now, Fig. 2 illustrates how the peripheral connectivity interface of the CAMI FPGA device that is mapped into a high speed 3-phase power micro-grid. The distributed energy resources (DER) earlier described served as the application context for deployment (i.e., Industry 4.0 domain). The various DERs connects to the CAMI which maps end users into the Cloud [4].

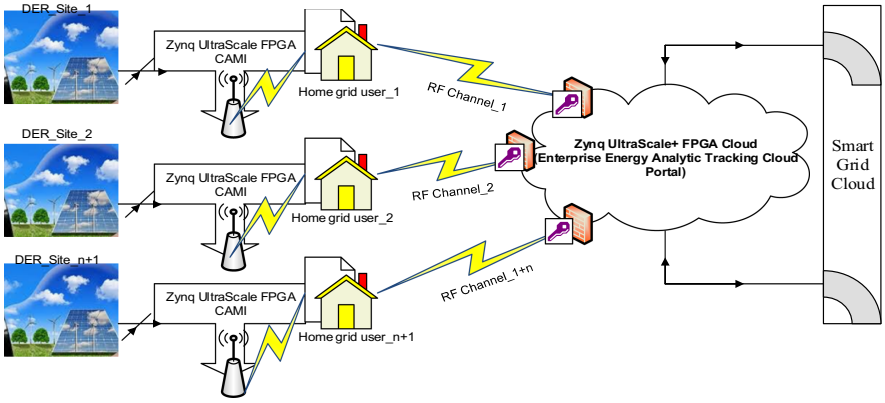


Fig. 2. Industry 4.0 framework for Smart Grid Cloud EMI.

Based on the consumption rate of end user *n+1,* the CAMI collects data locally and transmits into the cloud for analytics. For this to occur, the transmission happens as often as 1sec or as infrequently as daily, according to the usage of power. The collector retrieves the data, process in a full duplex mode for upstream processing. This is achieved in the analytics application which is represented as a Cloud service.

In this case, energy data is transmitted to the Cloud FPGA metering device for processing DSM. Such two way communications allow customers to participate in the smart grid scenario. The use of machine learning (ML) to achieve analytics was done with JAVA application program interface (API). This is based on massive online analysis (MOA). Data stream trajectory mining was achieved MOA using its JAVA ML classification for DSM audit logs. In wide-label classification, this can predict multiple output variables of DSM for every instantaneous input of the DER.

4.1 Advanced Metering Infrastructure

Let's further look at the AMI logic introduced in Fig. 2. In its operation, the metering filter (FPGA-ADC) was programmed with very high speed integrated circuit hardware descriptive language (VHDL) for Zynq UltraScale+ MPSoC device. This provided the desired System-On-a-Programmable Chip (SOPC) for CAMI. Specifically, 32-bit Cluster AMI device was built with eight Zynq UltraScale+ FPGA boards. This offers low latency profile in active state. The turning parameters used for the CAMI design is

highlighted in the data sheet [24]. The CAMI design has TCP/IP interface for data stream interactions. The CAMI on a 32 bit Zynq UltraScale+ board while using its digital signal processor (DSP) for complex signal conditioning/processing functions especially for power quality measurement features. The system has an efficient multi-core processor (SOC) that runs on low latency shared peripheral and memory interface buses. Also, the method of interconnection of resources makes the system design modular and scalable. In the implementation phase, the CAMI satisfies the requirements of low energy measurement and security computational functions. This uses specialized Arm Cortex-A53 co-processor with additional peripherals for data exchange and communication handles this task. Also, the core supports sufficient flash and internal memory. Hence, this executes the embedded real time operating system for commercially available or open source protocol stacks (Wi-Fi, Modem/3G, TCP/IP, Zigbee, Bluetooth Peripherals, Wimax), among others. For the security of the AMI for Cloud processing, it is pertinent to discuss the cryptographic transactional function needed for operational cloud automation. In this case, 30 users were considered in the algorithm.

4.2 Cryptographic Transactional Function

Considering the Zynq UltraScale+ FPGA module, its security algorithm is described for end-to-end communication. This features the FPGA block clusters, encryption keys, message block, VM_map and likely- function. These are taken into considerations as depicted in Algorithm I and II. It uses 512 key length for selected addresses. The input and output variables are activated via loop algorithms and wait states to secure all data movements. The IoT-enabled Cloud AMI security is robust using strong key length. The Zynq acquisition module gathers data from end-user meters and transmit to the Cloud portal for analytics. It uses multiple security algorithm to encrypt data (512 key length). Overall, the cryptographic algorithm supports secure lookup DSM/energy consumption prediction, grid possible failure points, peak generation and consumption forecast on daily basis. Location identification, fault tracking and assignment of grid failure correction are securely shared with an active user remotely. For an address map of 30 users, below is the functional security algorithm used in the design.

Algorithm I:. FPGA CAMI_ AES-Encrypt (FPGA block clusters)

Input: Cust_trans, Crypt_Cipher-Key, Fun () memory block
Crypt_mem_block size, bitstream map, likihood func ();
Output: FPGA_Cloud Domain, Cipher len, *N* Ran () Addresses (30);
While K_t = 512, *i*←0;**do**
Set Flag *F* = True/high (1);
Function memory block = Curent_Mem_block;
KSeed = Current Seed (time, ComputeID, addresses);

end while
Do
Repeat
K_t address (t) = model_memBlock (M_{mb}) ==.> 0;
Until *i* = 30;
Repeat
Set_Stage = Process (K_t address *i*)
Until *i* =30
While Set Stage *k* = 0
Repeat
If (*i* Ran 5 == 0) ***Then***
Encrypt_val (K_t address *i*) = key_length [L_k]
Crypt_temp = Encrypt*(K_t address *i*)
Encrypt_val[(K_t address *i*) = ObtainK block_value ($temp_i$)
L_k = L_k+1
Else
Encrypt_val[(K_t address *i*)]=Keylength [0:512]
Endif
Until *i* =30; Output ;
End
Return *Input*

Algorithm II: FPGA CAMI_ Routine Cryptographic Analytics

Input: Cust_trans, Crypt_Cipher-Key, Fun () memory block
Crypt_mem_block size, bitstream map, likihood func ();
Output: FPGA_Cloud Domain, Cipher len, *N* Ran () Addresses (30);
K_t = 512, *K, J,M* = 0; *i*←0;
While (Crypt_round = 1) **do**
Crypt_Seed = New_Seed (time, Proc_ID, Case_Address)
Encrypt_Val(K_t address (0:30)) <==Encrypt_Val
([(K_t address (0: 30)XoR R_com(Round X5/Nk)]
Repeat //redo
K_t address(i * 6 + j) <== func () mem_block + RANDt ()
% Crypt_mem_block_size;
If (*m*= 0);
Crypt_temp(K_t address(i * 6 + m)<==Crypt_temp
(K_t address(i * 6 + m)XoRtempt
Crypt_temp//Call output parameter
State [*k*] [*m*-1] = State [*k*][*m-1*]XoRtempt
Until *m; k* = 4; 3
Until round =30;

End
Return *Input*

5 System Implementation

Recall that the Cloud based FPGAs AMI has its bitstreams encrypted by AES on the client side as shown in Fig. 3. Also, the AES key is securely shared with Cloud driven FPGA core. At both the edge and Cloud, the bitstream protection on the FPGA cores facilitates advanced encryption standard (AES) key sharing for both encryption and decryption respectively. For high computational performance, the FPGA CAMIs leverages the low latency Xilinx FPGA ASIC processor for energy consumption computations. By acquiring data in real time from customer-side AMI (32 bit FPGA core) into the data-center server, this enabled the introduction of integrated OpenFlow load balancers (ISOLB) or Cloud virtual infrastructure controller (CVIC) for Cloud elasticity as shown in Fig. 3. Using ISOLB/CVIC algorithms III and IV, this handled the upward and downward scaling of compute resources considering end users.

For traffic workloads, Fig. 3 also allows on-demand power flexibility in terms of processing/computing power, storage and bandwidth. As a proof, discrete event data transfers is used to simulate the dynamic response characteristics of energy usage in the Cloud. By this approach of acquiring data, proactive management is introduced. The system sends frequent reports under stable QoS with impact on communication costs.

In terms of scalability, the FPGA elastic Cloud uses resilient engine which comes from the ISOLB/CVIC. The key benefit is on concurrent big data integration and security optimization. Overall, the cryptographic transform algorithm, predictor resource allocation and dynamic scaling are the novel features introduced to support Cloud and big data grid management. Clearly, the CVIC enables trusts on the Cloud servers thereby reducing exponential aggregation, storage, and processing. In the security activity, the joint collection points are secured thereby providing data availability always. Security history of huge datasets is computed and managed using weighted distributions for initial and posterior trends. The FPGA_cloud dynamic-scaling algorithm for is shown in Algorithm IV. AES key for the bitstream protection is securely shared between client and FPGA. The AES key is embedded in hardware bit stream and implemented in Fig. 4.

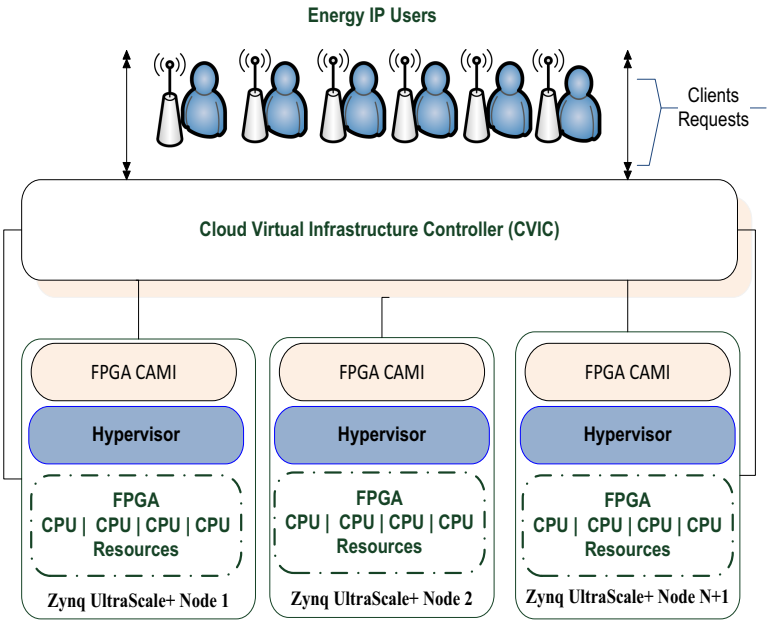


Fig. 3. Developed FPGA CAMI with Consumer loads (Energy IPs).

Algorithm III. FPGA Zynq UltraScale+ Predictor for QoS Allocation

Inputs: Bitstream_Control-CallSchedule for Zynq UltraScale *i* to *N+1*
History of CPS compute resources, QoS Provisioning and transactional workflow
$\prod \beta 1$ & $\prod \beta 2$// initialValue constant & trendPosteriorValue, respectively
pastValueForSubsytems, pastValueForSubsytems

Output: Zynq UltraScale+ _DES

Parameters: Zynq UltraScale+ _weight←Empty; // Zynq UltraScale+ weighted Moving Average
weight←0; weightedMoving←0; totalWeight←0;
fiboA←0; fiboB←1;
Zynq UltraScale+ _weight ContainerhistoryItem← null; // Resource Pattern

```
int i←0;
While i < Zynq UltraScale+ _monitorCallSchedule d do
    historyItem←HistoryList.get(HistoryList.Size()–
CPS_monitorCallSchedule – i )
    Zynq UltraScale+ _weight ←fiboA1 + fiboB2;
    Zynq UltraScale+ weightedMoving←CPSweightedMoving+ (Containerhistory
Item * weight);
    total Zynq UltraScale+ _weight ←total Zynq UltraScale+ _weight + Zynq
UltraScale+ weight;
    i ++;
end while
    Zynq UltraScale+ _DES ← Zynq UltraScale+ weightedMoving / total Zynq
    UltraScale+ Weight;
    // Calculate Dynamic Exponential Reliability
    initialValue ←(β1 * Zynq UltraScale+ _weight) + (1 – β1) * (pastInitialValue
    + trendPosteriorValue);
    trendPosteriorValue←β2 * (initialValue - pastInitialValue)+ ((1 – β2) *
     pastTrendPosteriorValue);
    Zynq UltraScale+_DES←initialValue+ trendPosteriorValue;
Return Zynq UltraScale+ _DES
```

Algorithm IV: FPGA_Cloud Dynamic-Scaling algorithm

Elasticity Container History of FPGA_Cloud Service provisioning for Zynq Ul traScale *i* to *N+1*
Zynq UltraScale+_monitorCallSchedule
QoS parameterStatus // Latency, Throughput & Service availability
Zynq UltraScale+_decision←Empty;
excessVM←Empty;

```
// Zynq UltraScale+_Monitor ( );
Redo every monitorCallPeriod Seconds
    // Analyzer
    if scalingParameterType is SLA-Aware then
        parameterStatus←Predictor (history of Response Time)
    Elseif scalingParameterType is Resource-Aware then
        parameterStatus←Predictor (history of QoS parameterStatus)
    end if
    // Zynq UltraScale+ Planner
    If parameterStatus is higher than Scale Up threshold then
        decision←Scale Up (Zynq UltraScale+)
    Elseif parameterStatus is lower than Scale Down threshold then
        decision←Scale Down (Zynq UltraScale+)
    End if
    // Zynq UltraScale+Executer
    If decision is Scale Up then
        Scale Up (new On-Demand Zynq UltraScale+VM );
    Elseif decision is Scale Down then
        excess Zynq UltraScale+VM←excessVM SelectionPolicy;
        Scale Down (excess Zynq UltraScale+VM);
    end if
end return
```

6 Evaluation

6.1 Hardware and Software Integration

In this Section, the proposed Cloud based AMI for DSM and the hardware integration is discussed. To evaluate the application program interface (API) design built with service oriented programming approach (SOPA) in MoA, real-time captures was carried out for various cases. The implementation of the CAMI was completed with its API called enterprise energy tracking analytic cloud portal (EETACP) application. This was built with C++ engine and JAVA respectively. Similarly, the hardware environment provides supports for dynamic system-level integration using C++/VHDL code while targeting Zynq SoC device architecture. The integration includes software-to-hardware translation,

device driver generation, and kernel creation. Also, its SoC platform allows for the specification of software functions for hardware translation. The FPGA Zynq SoC device explored C/C++ source code to realize the interface communications.

During the FPGA CAMI setup, the energy consumption patterns were monitored with low latency processing. Bit stream security was investigated for the workloads. In all cases, eight FPGA boards (ZCU102) were used in the configuration testbed. The individual modules contain Zynq UltraScale+ with Cortex-A53 quad-core processor (1.5 GHz, 4 GB DDR4 RAM), and has a number of programmable logic cells encapsulating the constructed 8-node cluster [24]. On the testbed, the eight nodes are connected through Ethernet via a high-end OpenFlow layer-3 switch [3]. In this regard, SDSoC 2016.4v [25] was introduced as an automation design tool from Xilinx and employed for the system code level implementation. As shown in Fig. 4, CAMI analytics application captures energy data from the CAMI for Cloud analytics. This event takes place every second based on 24-hour scale of daily energy supply. The API dynamically obtains the average power consumptions based on household loads. Case based observations was established via the metering system. In each case, AES bit-stream algorithm was invoked while showing the energy consumption for the CAMI processing. The demonstration setup revealed energy end user consumption profiles derived from the gathered load analytics.

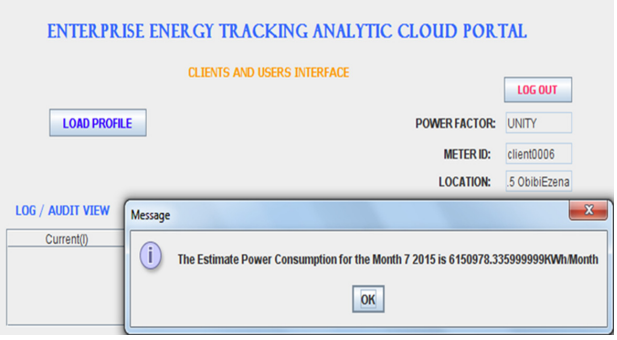


Fig. 4. Secured Cloud DSM Interface with history predictive load profiles.

6.2 Experimental Validations

In this Section, scheduling and resource allocation in the Cloud was done with Cloud Zynq UltraScale+ FPGA cores. To achieve scalability and security in CVIC, the work executed DSM (workload) in two basic instances namely: FPGA CAMI with heterogeneous virtual machine (VM) controller (proposed) and FPGA CAMI without heterogeneous VM controller (legacy). In the former, CAMI node-level scheduling with optimized dynamic scaling (Cloud resource allocation) was combined as shown in Algorithm IV. In this case, complete vertical scaling and CPU utilization cycles exploits resources in Cloud VMs, thereby reducing power drain without compromising performance. Figure 5 shows how resource utilization under heterogeneous VM controller scales relative to the legacy model for similar transactional workload. In this

case, the work varied the number of workloads per node which alters the workload intensity on the Cloud. Hence, the resource utilization responses are then observed. The CVIC actuates resources to the Cloud nodes based on CPU utilization cycles, and workload demands. About 71.43% utilization occurs due to the workload activities found in the proposed FPGA CAMI VM controller such as big data processing. Conversely, 28.57% utilization shows satisfactory performance with legacy workloads. The implication is that the resource usage of Algorithms I, II, III and IV are lowest in the Cloud without VM controllers. Though the proposed FPGA CAMI with VM offers very high resource utilization, but the impact of CVIC makes it to explore very minimal resources due to the optimal mapping on the control and management planes.

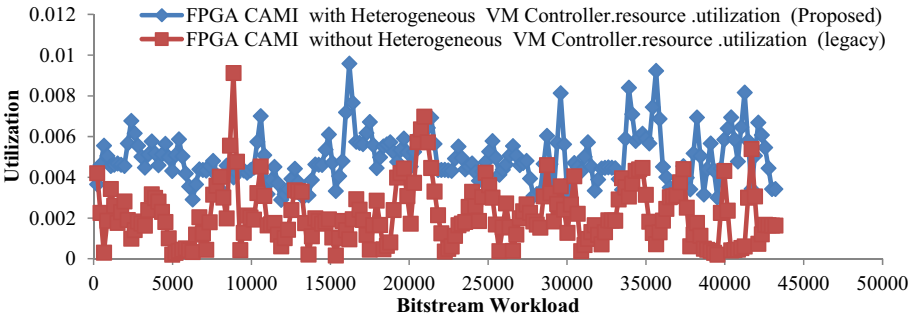


Fig. 5. Resource Utilization for FPGA CAMI transactional workload.

Figure 6 shows the latency response for FPGA CAMI transactional workload. It is the elapsed timeframe from the push-out of a service request to Cloud server for processing. To converge from Cloud failures, the latency times of both cases were observed. The plot shows that the latency profile of the proposed is higher (71.43%) when compared with the legacy FPGA CAMI scenario (28.57%) due to failure overheads. The more workload or devices that fail, the more increase the latency becomes. Also, as the workloads increases, the latency response increases as well. Hence, service requests with the proposed FPGA CAMI scenario are optimal as result of the CVIC/ISOLB.

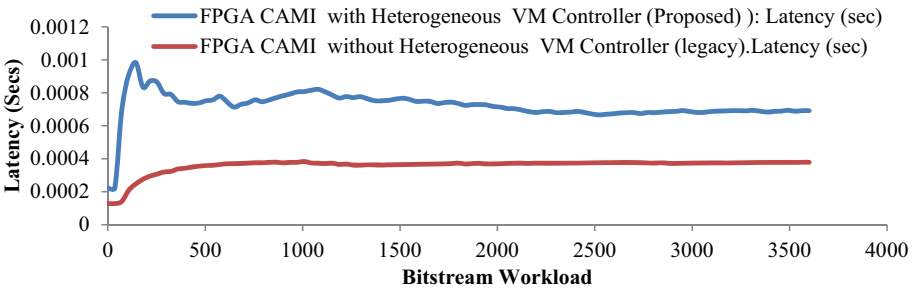


Fig. 6. Latency response for FPGA CAMI transactional workload.

Figure 7 shows the throughput response for FPGA CAMI transactional workload distribution. Clearly, he proposed algorithms suffer from overhead due to its close loop nature unlike the legacy FPGA CAMI without VM controller. From the results, the Open loop legacy algorithm is shown to provide a more reliable request distribution to the accurate Cloud servers.

However, due to dynamic auto-scaling in the proposed scheme, excess workloads and failures are reduced alongside the latency and utilization. With the VM controller, the FPGA CAMI offered 47.36% while without VM controller, 52.63% throughput was delivered. In this case, it was observed that the presence of heterogeneous VM controllers introduces overhead which deviated the latency, utilization and throughput metrics of the system. However, this is the tradeoff associated with FPGA scalability supports (and its ruggedness) necessary for high performance, security and flexibility characteristics.

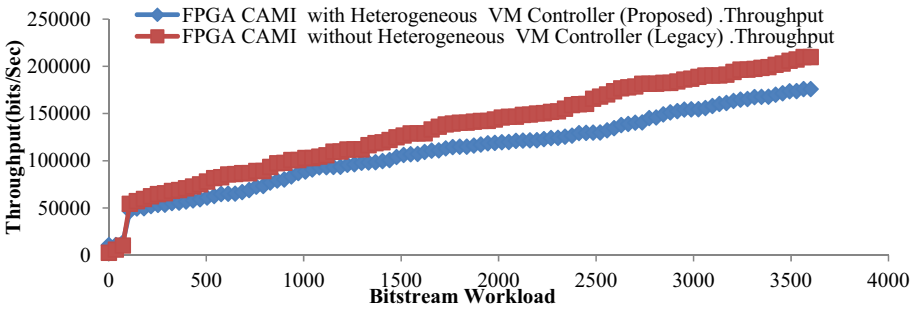


Fig. 7. Throughput response for FPGA CAMI transactional workload.

7 Challenges and Future Directions

Essentially, the author [26], highlighted that FPGAs with transition from about 10 million gates to about 50 million gates with internal logic clocks thereby creating massive opportunities for Cloud integrations. There are still issues regarding design time-market productivity, architectural scalabilities, power demands, flexibility among others. As its core density keeps evolving, the hardware-engines (logic element cores, LEC) and digital signal processors (DSP) data-paths becomes more complicated too.

Combining the LEC with soft-core and scaled embedded memories may present newer challenges for smart grid integration. First, the DSP-resource intensive constructs will need complicated arithmetic computations, massive memories, great bandwidth demands, and high-speed serial communications with dynamic reconfiguration. For mission critical services, the above specifications computationally will drain significant amount of power at peak performance states. Future direction of FPGA device architectures must focus on addressing these concerns with the least power consumption profiles. This should be done without degrading performance especially

next generation networks (e.g., 5G/6G and SD-WAN). With the different construction blocks (e.g., transceivers, memories, and multipliers, for various designs), power consumption across its implementation must be optimized.

8 Conclusion

This paper has presented a use case of Industry 4.0 elastic Cloud AMI that leverages Zynq UltraScale+ FPGA cores. The system supports bi-directional communication for transactional workloads between customer meters and Cloud. Low latency FPGA ASIC implementation is developed to satisfy the requirements of smart grid architecture. The functionalities of CAMI are highlighted while enumerating the benefits. With the built analytics application, consumer energy frauds and other issues resulting from the existing metering model are addressed through DSM. Using an on-demand FPGA CAMI, the work discussed the functional algorithms for performance optimization. In terms of security, the FPGA CAMI, used 512 AES key length for metering data and also captured in the hardware design. In the evaluation, user's requests are mapped to the CVIC/ISOLB for workload optimizations. Low latency energy usage, predictive QoS allocation and dynamic-workload balancing are investigated in selected experiments. Despite the overhead introduced by the CVIC/ISOLB, the work showed that Cloud-based FPGAs provides optimal performance benefits. The results show that leveraging FPGA hardware acceleration has the potential to increase the attention on FPGA-assisted cloud systems. The proposed scheme can be applied in big-data applications as well as other security-constrained sectors such as Block chain. Future work will focus on CAMI reliability constraints in big data streaming applications and cyber-physical systems [27].

References

1. Song, Y., Kong, P.Y., Kim, Y., Baek, S., Choi, Y.: Cellular-assisted D2D communications for advanced metering infrastructure in smart grid. IEEE Syst. J. 1–12 (2019)
2. Kaveh, D., Yuxuan, Y., Zhaoyu, W., Fankun, B.: A game-theoretic data- driven approach for pseudo-measurement generation in distribution system state estimation. IEEE Trans. Smart Grid 1 (2019)
3. Padmaprabha, V.R., Divya, D.S., Jiju, K.: SOPC based sigma delta ADC IP core for smart energy meter. In: International Conference on Circuits & Systems (ICCS), pp. 140–144. IEEE Xplore, India (2017)
4. Okafor, K.C.: Development of a model for smart green energy management using distributed cloud computing network. Ph.D. thesis, University of Nigeria, Nsukka (2017)
5. Saroa, M.K., Aron, R.: Fog computing and its role in development of smart applications. In: IEEE International Conference on Parallel & Distributed Processing with Applications, Ubiquitous Computing & Communications, Big Data & Cloud Computing, Social Computing & Networking, Sustainable Computing & Communications (ISPA/IUCC/BDCloud/SocialCom/SustainCom), pp. 1120–1127. IEEE Xplore, Melbourne, Australia (2018)

6. Jessica, A.J., Oliveira, D., Augusto, F.R.Q., Lima, E.R.D., Mertes, J.: An MR-FSK transceiver compliant to IEEE802.15.4g for smart metering utility applications: FPGA implementation and ASIC resource estimation. In: Latin-American Conference on Communications (LATINCOM), pp. 1–4. IEEE Xplore, Peru (2015)
7. Luis, D.O.A., Guilherme, A.E.M., Carlos, A.C.: Power quantities calculation by using an FPGA device, applying IEEE 1459-2010 standard. In: IEEE Simposio Brasileiro de Sistemas Eletricos (SBSE), pp. 1–6. IEEE Xplore, Niteroi, Brazil (2018)
8. Augusto, F.R.Q., et al.: FPGA implementation of an IEEE802.15.4g MR-OFDM baseband modem for Smart metering utility networks. In: IEEE 4th Global Conference on Consumer Electronics (GCCE), pp. 125–126. IEEE Xplore, Japan (2015)
9. Luis, D.O.A., Guilherme, A.M., Carlos, A.C.: FPGA-based power meter implementation for three-phase three-wire and four-wire power systems, according to IEEE 1459-2010 standard. In: IEEE Brazilian Power Electronics Conference (COBEP), pp. 1–6. IEEE Xplore, Juiz de Fora, Brazil (2017)
10. Radek, S., Josef, V., Ján, T.: Wideband partial discharge meter using FPGA. In: IEEE 9th IEEE International Symposium on Diagnostics for Electric Machines, Power Electronics and Drives (SDEMPED), pp. 396–401. IEEE Xplore, Spain (2013)
11. Nidhi, G., Anu, M., Deepika, K., Devyani, T.: A novel implementation of 32 bit extended ALU Architecture at 28 nm FPGA. In: IEEE International Conference on Emerging Trends in Communication Technologies (ETCT), pp. 1–4. IEEE Xplore, India (2016)
12. Rohit, J., Raghavendra, M.: Implementation of 32-bit RISC processors without interlocked Pipelining on Artix-7 FPGA board. In: International Conference Circuits, Controls, and Communications (CCUBE), pp. 105–108. IEEE Xplore, Bangalore (2017)
13. Okafor, K.C., Guinevere, E.C., Akinyele, O.O.: Hardware description language (HDL): an efficient approach to device independent designs for VLSI market segments. In: Proceedings of the 3rd IEEE International Conference Adaptive Science and Technology (ICAST), pp. 262–267. IEEE Xplore, Abuja (2011)
14. Safaa, S.O., Ahmed, K.A.: Design and implementation of 32-Bits MIPS processor to Perform QRD based on FPGA. In: IEEE International Conference on Engineering Technology and their Appls, (IICETA), Iraq, pp. 36–41 (2018)
15. Omran, S.S., Ahmed, A.K.: Design of 32-bits RISC processor for hardware efficient QR decomposition. In: International Conference on Advance of Sustainable Engineering & its Application (ICASEA), pp. 69–73 (2018)
16. Ankita, G., Hamid, M.: Embedded System Design Flow, using ALTERA FPGA Development Board (DE2-115 T-Pad) (2020)
17. Xu, L., Shi, W., Suh, T.: PFC: privacy preserving FPGA cloud - a case study of map reduce. In: IEEE 7th International Conference on Cloud Computing, Anchorage, AK, pp. 280–287 (2014)
18. Hategekimana, F., Mandebi, J.M., Pantho, M.J.H., Bobda, C.: Secure hardware kernels execution in CPU+FPGA heterogeneous cloud. In: International Conference on Field-Programmable Technology (FPT), pp. 182–189. IEEE, Naha (2018)
19. Kim, H.-Y.: SafeDB: spark acceleration on FPGA clouds with enclaved data processing and bitstream protection. In: IEEE 12th International Conference on Cloud Computing (CLOUD), pp. 107–114. IEEE, Milan (2019)
20. McConnel, T.E.: ESC - Xilinx All Programmable System on a Chip combines best of serial and parallel processing (2010). https://www.eetimes.com/esc-xilinx-extensible-processing-platform-combines-best-of-serial-and-parallel-processing/. Accessed 15 Nov 2020
21. Hamblen, J.O.: Using system-on-a-programmable-chip technology to design embedded systems. IJCA **13**(3) (2006)

22. Christophersen, H., Pickell, R., Neidhoefer, J., Koller, A., Kannan, S., Johnson, E.: A compact guidance, navigation, and control system for unmanned aerial vehicles. J. Aerospace Comput. Inf. Commun. ARC **3**, 187–213 (2006)
23. Jad, G.A.: EDA tools usage and tutorial authoring for basic electronic circuits education. In: IEEE 12th European Workshop on Microelectronics Education (EWME), pp. 51–54 (2018)
24. Xilinx. Zynq UltraScale+ MPSoC Data Sheet: Overview- DS891 (v1.8) (2019)
25. Kathail, V., Hwang, J., Sun, W., Chobe, Y., Shui, T., Carrillo, J.: SDSoC: a higher-level programming environment for Zynq SoC and Ultrascale+ MPSoC. In: Proceedings of the 2016 ACM/SIGDA, International Symposium, on Field-Programmable Gate Arrays, p. 4 (2016)
26. Bolsens, I.: Challenges and opportunities for FPGA platforms. In: Glesner, M., Zipf, P., Renovell, M. (eds.) FPL 2002. LNCS, vol. 2438, pp. 391–392. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-46117-5_41
27. Okafor, K.C.: Dynamic reliability modelling of cyber-physical edge computing network. Int. J. Comput. Appl. (IJCA), SI- Sustainable Comput. Intell. Syst. **40**(42), 1–10 (2019)