A Vision of Fog Systems with Integrating FPGAs and BLE Mesh Network

Xiaokun Yang[1], Lei Wu[1], Xianlong He[2], Archit Gajjar[1], and Yi Feng[3]
[1] University of Houston Clear Lake, Houston, TX77059, USA
[2] Bluefnt Co., Ltd, Hong Kong, China
[3]Algoma University, Marie, ON P6A 2G4, Canada
Email: yangxia@uhcl.edu

Abstract —This paper presents a novel integrated field-programmable gate arrays (FPGA) and Bluetooth Low Energy (BLE) mesh system in the field of fog computing. Combining the merits of 1) programmability of FPGAs, and 2) wide-area communication and low-energy consumption of BLE mesh technology, this platform will create a diverse range of applications such as low-latency computation at the network edge, as well as showing a demo for Internet-of-Things (IoT) connections to cover an entire building. By integrating FPGAs, more important, the fog node can offer a great potential for flexible acceleration of many workloads and improve power efficiency with more hardware parallelism than CPUs and GPUs. These advantages are going to move FPGAs into the mainstream of fog computing for the foreseeable future. The preliminary results of a BLE mesh network is demonstrated on http://sceweb.sce.uhcl.edu/xiaokun/#-> project. Generally we provide three test cases: to control the mesh network by local device, local server, and cloud server. It can be observed that the feedback control from the local device achieves the lowest latency, and the response delay from the cloud server is the highest. Therefore, it is necessary to pre-process more than 90% raw data from sensor network at the proximity of the network edge/fog, particularly with the number of IoT devices in tens of billions levels

Index Terms—Bluetooth Low Energy (BLE) mesh, Fog computing, Internet-of-Things (IoT).

I. INTRODUCTION

The growth in Internet-of-Things (IoT) is explosive, impressive, and is likely to overwhelm traditional storage systems and cloud servers. It is a big challenge for today's data centers to process such a large volume of data available from sensors, controllers, monitors and other devices. Hence a new computing paradigm, fog computing, is originally coined by Cisco as an enabling technology allowing computation to be performed at the proximity of data sources [1]. Generally, as shown in Fig. 1 the fundamental idea of fog computing is that instead of uploading all the raw data into the cloud servers/centers, more than 90% of sensing data will be pre-processed or filtered at the network edge/fog. It has several benefits compared to conventional cloud-based computing paradigm, such as low-latency response, wise usage of bandwidth, as well as data security and privacy [2].

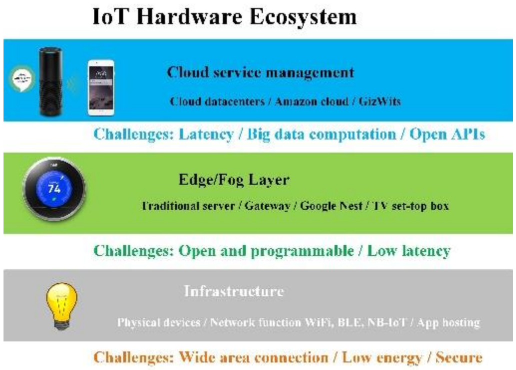


Fig. 1. Cloud-Fog-IoT system

Under this context, this paper focuses on presenting a novel fog system with reconfigurable field-programmable gate array (FPGA) accelerators and the Bluetooth Low Energy (BLE) network using the latest CSRmesh technology [3], [4]. The specific contributions are as follows:

- FPGA in general provide more hardware parallelism than is possible with software-based applications, so tasks executed in hardware will be more efficient on an FPGA than in software on a GPU or CPU. Motivated by this consideration, in our proposed architecture FPGAs will be integrated with the CSRmesh servers as computation accelerators.
- Second, instead of sending sensing data to cloud servers directly, our proposed edge system allows computation to be performed at the proximity of data sources. It has several benefits compared to conventional cloud-based computing paradigm, such as high-speed response, low throughput required for cloud computing, as well as data security and privacy due to analyzing data inside the local network.
- To test the system on a number of devices, the BLE mesh solution is adopted to deploy an IoT network and address the cost-energy-range limitations. We believe that the combination of mesh and BLE technology will be one of the keys to many wide-area and low-bandwidth network applications, capable of providing a low-cost and low-energy way (10% of Wi-Fi and Zigbee) to network together an almost unlimited number (65535 per group) and unlimited range (×relays) of Bluetooth enabled devices [4]. The mesh technology has not been released in the

---

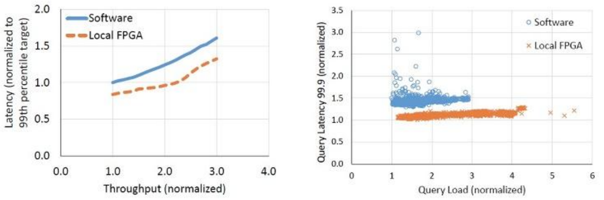
Manuscript received May 18, 2018; revised February 20, 2019.
doi:10.12720/jcm.14.3.210-215

Bluetooth standard yet, hence we manufacture and customize some BLE development boards by using BLE modules provided by the CSR vendor.

The organization of this paper is as follows: section 2 briefly reviews the motivation and related works of our proposed system, and section 3 illustrates the challenges of the future fog computing. In section 4, the proposed work is introduced, and in section 5, the future work is discussed. Finally, section 6 concludes this paper.

II. PRIOR WORKS AND MOTIVATIONS

The basic idea of this work comes from the optimized Microsoft Bing architecture published in 2016 [5], [6]. Basically Bing reaps the benefits in two ways: 1) by embedding a 6×8 torus of high-end Stratix V FPGAs into a half-rack of 48 machines, the throughput is increased by ×2.25, which means that fewer than half as many servers would be needed to sustain the target throughput at the required latency (as shown in Fig. 2 (a) [5]); 2) because the FPGA is able to process requests while keeping latencies low, it is able to absorb more than twice the offered load while executing queries at a latency that never exceeds the software data center at any load (as an example shown in Fig. 2 (b) [5]).



(a) 99% Latency vs. Throughput of ranking service queries running on a signal server, with and without FPGAs enabled

(b) Query 99% Latency vs. Offered Load, , with and without FPGAs enabled

Fig. 2. Performance of microsoft bing system with and without integrating FPGAs [5]

Given that FPGAs show promise for accelerating many computational tasks, we propose a novel system by offloading some complex algorithms, such as deep learning [7], high performance computing [8]-[10], data encryption and decryption [11], [12], and energy minimization [13], [14], from CSRmesh server to an FPGA coprocessor.

In addition, our previous works [15] and [16] demonstrated a smart system where it is possible to utilize intelligent analytics to filter sensor data, provide new insights, and improve decision making. In this paper we focus on presenting a vision with integration of FPGA and BLE mesh technology, as well as showing a prototype of a BLE mesh system with the latest CSR1020 modules. Comparing with the prior works in [15], [16] employing CSR1010, the CSR1020 chip optimizes the resource cost and is able to provide 15 GPIO (12 GPIO for CSR1010) which is more extendable by combing with FPGAs.

III. CHALLENGES OF FOG SYSTEMS

In this work we 1) focus on discussing two challenges of fog computing and bringing forward one of the potential solutions; and 2) demonstrate our idea with a prototype of a low-energy CSRmesh network.

A. *Wide-area, Low-cost, and Low-energy Communication of the IoE Layer*

Because of the expected boom in IoT with smart agriculture, industrial IoT, smart city, and such wide-area applications, there is a strong interest in developing wide-range communication solutions within an IoT/fog system. Second, in addition to the desired property of low energy consumption, the hardware must be inexpensive and ease of use.

To address the cost-energy-range limitations, the CSRmesh solution will be adopted in our proposed system as a case study. Since billions of existing devices have been compatible with Bluetooth technology for many years, we believe that the combination of mesh network and Bluetooth Low Energy (BLE) technology would be regarded as a key to many large-area and low-bandwidth network applications, capable of providing a low-cost and low-energy way to network together an almost unlimited number and unlimited range of Bluetooth enabled devices [4].

B. *Programmability and Power Efficiency of the Fog Node*

Homogeneity is highly desirable for fog nodes to reduce management issues and to provide a consistent platform that applications can rely on. In other words, fog servers need continued improvements in performance and efficiency, but cannot obtain those optimizations from general-purpose systems and non-programmable hardware.

Hence, in our proposed system the reconfigurable architecture—FPGA, will be integrated in the fog layer to improve the flexibility of the fog system. More important, FPGA in general provide more hardware parallelism than is possible with software-based applications, so tasks executed in hardware will be more efficient on an FPGA than in software on a MCU or CPU.

IV. PROPOSED ARCHITECTURE

In this section, a vision of fog systems containing a cloud layer, a fog layer, and an IoT layer is introduced, and furthermore, a prototype of the physical/network layer is demonstrated

A. *A Vision of the Framwork*

As a case study, Fig. 3 reports the main blocks that compose a fog system. First of all, an FPGA board is expected to provide the needed interfacing-plus-computing support by nearby fog nodes. The main block is an Arbiter to direct different types of data to the optimal place for analysis. Different channels will be considered as different priorities, such as:

- Math acceleration: the most time-sensitive data is analyzed on the FPGA at the edge of the network;
- Data filter channel: data that can wait seconds or minutes for action is passed along to the fog servers for analysis and action;
- Data compression: data that is less time sensitive is compressed and sent to the cloud for historical analysis, bid data analytics, and long-term storage.

In addition, to test this system on a number of devices, we deploy a CSR mesh network that can detect and analyze environment data such as humidity and temperature, and then make a decision to feedback control the actuators or send valuable information to fog/cloud servers. Three interfaces are offered to communicate with the network: cloud server control, local server feedback, and accessing by local devices. In the future, we will focus on reducing the computation latency by offloading workload to FPGAs. By sensing and automating our surrounding environment, the application will create many practical improvements in our lives, increasing our convenience, health, and safety, while at the same time improving energy efficiency and speed.

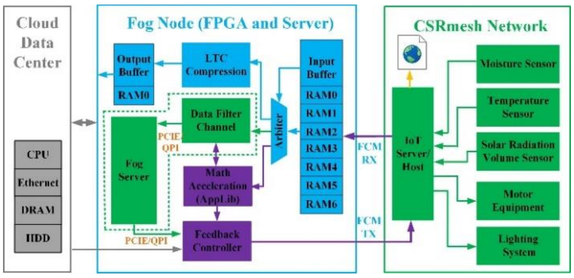


Fig. 3. Proposed fog system

B. Self-fabricated CSRmesh Boards

As per a full vision shown in Fig. 3, in this paper we concentrate on demonstrating that an IoT mesh network with sensors and actuators is actually feasible and practical. While the BLE mesh standard has not been published yet and currently only a few vendors have it available in the market, we select CSR (one of the earlier vendors) Bluetooth models to manufacture the demo boards [4], which is shown in Fig. 4.

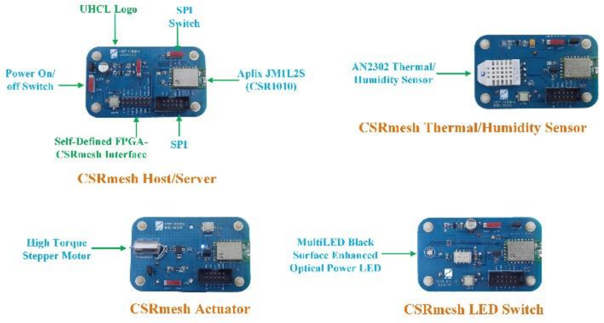


Fig. 4. Self-fabricated boards using CSR1020 modules

More specifically, we employ APlix CSR1020 modules for fabricating four development boards [17], and then establish a BLE mesh network. In addition to the basic operations such as sensing and actuating devices, more important, this prototype also shows a BLE star-mesh integration topology to extend the connectivity range to more than 100 feet [15], [16].

As customized devices, we create our own specification for the host/server and three devices. The control packets for LED, sensor, motor are illustrated in Fig. 5 (a)-(d), and the data packet received from the sensors is depicted in Fig. 5 (e).

The LED can be configured as 256 colors by sending a data packet with data type as hexadecimal "F1", and can be switched ON/OFF by transmitting hexadecimal "F2" through the CSR mesh host/server. Similarly, the motor is controlled by data type of "F4", and the humidity and temperature data will be collected with sending commands of packet type "F3".

Header	Size	Type	On/Off	Red	Green	Blue	Checker	Stop
2	1	1	1	1	1	1	1	2
FA, F5	07	F1	00~01	00~FF	00~FF	00~FF	XX	0D, 0A

(a) LED Control Packet (TX)

Header	Size	Type	On/Off	Checker	Stop
2	1	1	1	1	2
FA, F5	04	F2	00~01	XX	0D, 0A

(b) LED On/Off Packet (TX)

Header	Size	Type	Sensing data	Checker	Stop
2	1	1	1	1	2
FA, F5	04	F3	00~01	XX	0D, 0A

(c) Sensor control Packet (TX)

Header	Size	Type	Direction	Strength	Checker	Stop
2	1	1	1	1	1	2
FA, F5	05	F4	00~01	00~05	XX	0D, 0A

(d) Motor control Packet (TX)

Header	Size	Type	Humidity	Temperature	Checker	Stop
2	1	1	2	2	1	2
FA, F5	07	F3	0000 ~FFFF	0000 ~ FFFF	XX	0D, 0A

(e) Thermal/humidity data (RX)

Fig. 5. Transmitting and receiving data packets

The fourth field in Fig. 5 (e) represents the humidity and the fifth field indicates the temperature. Each of the field spends two bytes of the whole packet. To convert the hexadecimal data into humidity, we create an equation as

$$Humidity\ (h3\text{-}h2\text{-}h1\text{-}h0)$$
$$=(h3\times4096+h2\times256+h1\times16+h0)/10\times100\% \tag{1}$$

where h3-h2-h1-h0 represents the four hexadecimal data, from the most significant four bits to the least significant four bits. For example, if the received humidity data is hexadecimal "0292", the humidity will be $(2\times256+9\times16+2)/10\times100\%=65.8\%$.

Similarly, let t3-t2-t1-t0 denote the sixteen bits of the hexadecimal temperature, which is shown in the fifth field in Fig. 5(e). Then the temperature can be obtained by using the following equation:

$$Temp\ (t3\text{-}t2\text{-}t1\text{-}t0)=(t2\times256+t1\times16+t0)/10\ Celsius \tag{2}$$

In equation (2), we define the most significant four bits as the sign bit of the temperature, “0001” meaning negative and “0000” meaning positive. As an example, if the receiving data is hexadecimal “010D”, the temperature can be computed as $(1\times256+0\times16+13)/10=26.9$ *Celsius.*

Furthermore, we define four data packets for testing or checking the UART connection and CSR mesh network. As shown in Fig. 6, (a)-(b) depict the detecting commands. Packet type “AA” denotes the UART connection checking and the packet type “AB” indicates the mesh network checking.

Fig. 6 (c)-(d) show the received packets in regard to the connection and network status. In Fig. 6 (c), if the receiving data in the fourth field is hexadecimal “55”, the UART is normally connected. Otherwise the UART is disconnected. Likewise, when the receiving data in the fourth field in Fig. 6 (d) is hexadecimal “01”, the mesh network is indicated as normal status. Otherwise the network is disconnected.

Header	Size	Type	Check	Checker	Stop
2	1	1	1	1	2
FA, F5	04	AA	01	AF	0D, 0A

(a) UART connection checking (TX)

Header	Size	Type	Device No.	Checker	Stop
2	1	1	1	1	2
FA, F5	04	AB	XX	XX	0D, 0A

(b) Mesh network checking (TX)

Header	Size	Type	Data	Checker	Stop
2	1	1	1	1	2
FA, F5	04	AA	55	03	0D, 0A

(c) UART checking data (RX)

Header	Size	Type	Device No.	Status	Checker	Stop
2	1	1	1	1	1	2
FA, F5	05	AB	XX	01	XX	0D, 0A

(d) Mesh network checking data (RX)

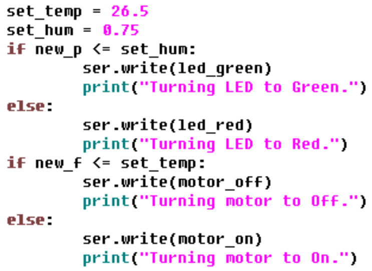
Fig. 6. Prototype

The specification is created for developing the physical layer of an IoT ecosystem shown in Fig. 3. The main contributions are, 1) we provide a simple and inexpensive way to connect together a large number of devices to establish a mesh network. The communication area can be dramatically extended over intermediate relays. 2) In order to further extend the IoT host/server as a high-performance edge/fog computing node, we develop an external I/O connector with self-defined commands on the host board. By connecting the other heterogeneous resources with the host board, the computing workload can be offloaded to such as FPGAs, GPUs, and other MCU components to improve the system speed and create many future applications.

V. A PROTOTYPE OF THE CSRMESH NETWORK

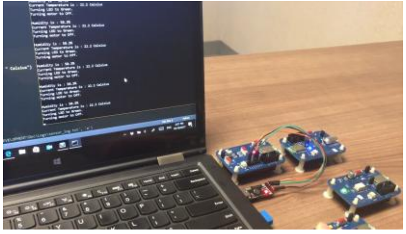
In order to test the response latency, three different control channels are used to configure the mesh network. The benchmark is that the raw data available from humidity and temperature sensors in the mesh network is collected by the IoT host/server. After analyzing the sensor data, the host engine is able to automatically make a decision to control the irrigation and lighting systems, which are simulated as a motor and LED.

More specifically, the first demonstration is to access the mesh network by local devices for filtering the raw data and feedback control the network. As an example shown in Fig. 7, the threshold of temperature and humidity is set as 26.5 celeries and 75%, respectively. The LED will be turned to green color when the sensing humidity is lower than 75%. Otherwise the LED sustains in red color. The motor is controlled by the received sensing temperature. If the temperature is greater than 26.5 celeries, the motor/fan will be turned on, otherwise off.



```
set_temp = 26.5
set_hum = 0.75
if new_p <= set_hum:
        ser.write(led_green)
        print("Turning LED to Green.")
else:
        ser.write(led_red)
        print("Turning LED to Red.")
if new_f <= set_temp:
        ser.write(motor_off)
        print("Turning motor to Off.")
else:
        ser.write(motor_on)
        print("Turning motor to On.")
```

(a) Design code



(b) Demo of feedback control

Fig. 7. A feedback control system

The second test is to control the mesh network by Gizwits APP through accessing GoKit servers. As shown in Fig. 8, by employing the GoKit board we are able to connect with the GoKit server, and then feedback control the three LED and the actuator ON/OFF.

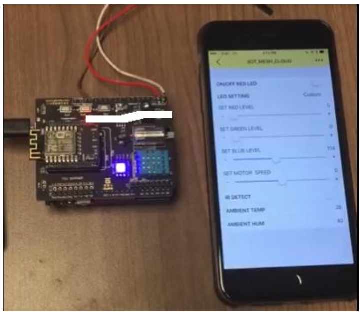
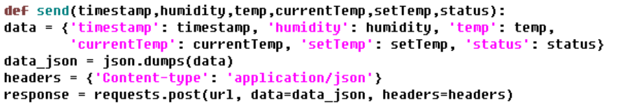


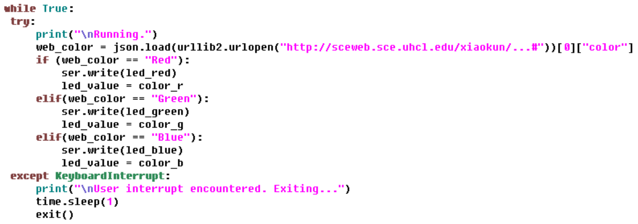
Fig. 8 Demo of network control by GoKit server

The final test is to configure the mesh network through the locals servers provided by our research lab. As shown in Fig. 10 (a), the sensing data can be displayed on the demonstration webpage in real time. Additionally three buttons are created to control the LED as three typical colors: red, green, and blue, which is shown in Fig.10 (b). The demonstration is depicted in Fig. 10(c).



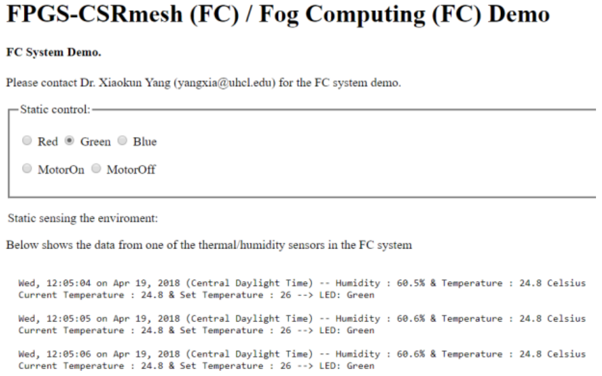
```
def send(timestamp,humidity,temp,currentTemp,setTemp,status):
data = {'timestamp': timestamp, 'humidity': humidity, 'temp': temp,
        'currentTemp': currentTemp, 'setTemp': setTemp, 'status': status}
data_json = json.dumps(data)
headers = {'Content-type': 'application/json'}
response = requests.post(url, data=data_json, headers=headers)
```

(a) Display sensing data on webpage



```
while True:
 try:
     print("\nRunning.")
     web_color = json.load(urllib2.urlopen("http://sceweb.sce.uhcl.edu/xiaokun/...#"))[0]["color"]
     if (web_color == "Red"):
         ser.write(led_red)
         led_value = color_r
     elif(web_color == "Green"):
         ser.write(led_green)
         led_value = color_g
     elif(web_color == "Blue"):
         ser.write(led_blue)
         led_value = color_b
 except KeyboardInterrupt:
     print("\nUser interrupt encountered. Exiting...")
     time.sleep(1)
     exit()
```

(b) LED control from webpage



FPGS-CSRmesh (FC) / Fog Computing (FC) Demo

FC System Demo.

Please contact Dr. Xiaokun Yang (yangxia@uhcl.edu) for the FC system demo.

Static control:

○ Red ● Green ○ Blue

○ MotorOn ○ MotorOff

Static sensing the enviroment:

Below shows the data from one of the thermal/humidity sensors in the FC system

```
Wed, 12:05:04 on Apr 19, 2018 (Central Daylight Time) -- Humidity : 60.5% & Temperature : 24.8 Celsius
Current Temperature : 24.8 & Set Temperature : 26 --> LED: Green

Wed, 12:05:05 on Apr 19, 2018 (Central Daylight Time) -- Humidity : 60.6% & Temperature : 24.8 Celsius
Current Temperature : 24.8 & Set Temperature : 26 --> LED: Green

Wed, 12:05:06 on Apr 19, 2018 (Central Daylight Time) -- Humidity : 60.6% & Temperature : 24.8 Celsius
Current Temperature : 24.8 & Set Temperature : 26 --> LED: Green
```

(c) Demo of network control

Fig. 9. Demo of network control by local servers

Finally, the comparison of the response latency is summarized in Table I. It can be observed that the local host achieves the lowest latency since the computation occurs at the network edge. And the feedback control from the GoKit server obtains the longest delay due to the access with cloud servers. All the prototypes are available on http://sceweb.sce.uhcl.edu/xiaokun/#-> project either by videos or by demonstrations.

TABLE I: RESPONSE LATENCY OF THREE CHANNELS

Test Cases	Response Latency
GoKit server	High
Local server	Medium
Local host	Low

VI. FUTURE WORK

The integration of traditional edge/cloud servers with FPGAs is one of the solutions to design a new generation of products to address emerging workloads in IoT. As a case study, thus, in this paper we provide a specific FPGA-CSRmesh vision in the field of edge/fog computing and hope this prototype will gain attention from the community and inspire more research in this direction. To the best of our knowledge, no technical contributions have been published on the actual FPGA integrations of the BLE mesh network.

The topic is novel and timely with the emerging IoT technology. We believe that in the coming future many wide-area wireless communication protocols (maybe the extension of Bluetooth 5.0 [18]) and FPGA/GPU based fog/edge computing platform, will be standardized and supported by industry and academy. This prototype is thus likely to create a big impact to the design of such systems by enabling simple and effortless interactions across the vast range of connected devices.

VII. CONCLUSION

This paper presents an innovative FPGA-CSRmesh platform with three main contributions: 1) an FPGA integration is able to offload and accelerate tasks to improve power efficiency and reduce cost; 2) CSR mesh network provides a simple and inexpensive way to network together an almost unlimited number of BLE enabled devices; 3) since no gateway or router is needed in the mesh network, it not only saves considerable cost and ease the deployment process but also protects sensitive IoT data by analyzing it inside the local network. IoT is becoming a reality when a large amount of sensors and equipment can securely and intelligently connect and interoperate, and can be efficiently and securely processed by fog/cloud nodes. Our proposed system is able to automatically control hundreds of different devices in a large mesh network, right from the programmable and high-speed fog node without having to have a ubiquitous Wi-Fi connection typically required for other IoT platforms. This especially helpful in factory, rural or agricultural environments for applications such as industrial IoT and smart farming. This platform is thus likely to make some changes to the IoT ecosystems in a profound way when the connected devices is in tens of billions levels.

REFERENCES

[1] C. Public. (2015). Cisco Global Cloud Index: Forecast and Methodology, 2015-2020. (2015). [Online]. Available: http://www.cisco.com/c/dam/en/us/solutions/collateral/service-provider/global-cloud-index-gci/white-paper-c11-738085.pdf

[2] W. Shi, "Edge computing: Vision and challenges," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 637–646, Oct. 2016.

[3] Build a Mesh Network with Bluetooth Low Energy (BLE). (2016). [Online]. Available: https://www.qualcomm.com/products/features/csrmesh

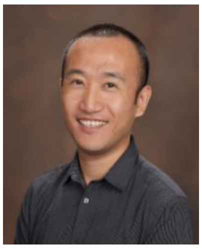
[4] Qualcomm CSR Mesh, Build a Mesh Network with Bluetooth Low Energy (BLE). [Online]. Available: https://www.qualcomm.com/products/features/csrmesh,

[5] A. M. Caulfield, E. S. Chung, A. Putnam, etc., "A cloud-scale acceleration architecture," in *Proc. 49th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, Oct. 2016, pp. 506-511.

[6] A. Putnam, A. M. Caulfield, E. S. Chugn, etc., "A reconfigurable fabric for accelerating large-scale datacenter services," *IEEE MICRO*, vol. 35, no. 3, pp. 10-22, May 2015.

[7] H. He, L. Wu, X. Yang, *et al.*, "Dual long short-term memory networks for sub-character representation

learning," in *Proc. 15th Intl. Conference on Information Technology-New Generations (ITNG)*, Jan. 2018, pp. 1-6.
[8] X. Yang, N. Wu, and J. Andrian, "A novel bus transfer mode: Block transfer and a performance evaluation methodology," *Elsevier, Integration, the VLSI Journal*, vol. 52, no. C, pp. 23-33, Jan. 2016.
[9] X. Yang and J. Andrian, "A high performance on-chip bus (MSBUS) design and verification," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst. (TVLSI)*, vol. 23, no. 7, pp. 1350-1354, July 2015.
[10] X. Yang and J. Andrian, "A low-cost and high-performance embedded system architecture and an evaluation methodology," *IEEE Compt. Society Annual Symposium on VLSI (ISVLSI)*, pp. 240-243, Sept. 2014.
[11] X. Yang, W. Wen, and M. Fan, "Improving AES core performance via an advanced IBUS protocol," *ACM Journal on Emerging Technologies in Computing (ACM JETC)*, vol. 14, no. 1, pp. 61-63, Jan. 2018.
[12] X. Yang and W. Wen, "Design of A Pre-Scheduled Data Bus (DBUS) for Advanced Encryption Standard (AES) Encrypted System-on-Chips (SoCs)," in *Proc. 22nd Asia and South Pacific Design Automation Conference*, Chiba, Japan, Feb. 2017, pp. 1-6.
[13] M. Fan, Q. Han, and X. Yang, "Energy minimization for on-line real-time scheduling with reliability awareness," *Elsevier Journal of Systems and Software*, vol. 127, pp. 168–176, May 2017.
[14] X. Yang, N. Wu, and J. Andrian, "Comparative power analysis of an adaptive bus encoding method on the MBUS structure," *Journal of VLSI Design*, vol. 2017, pp. 1-7, May 2017.
[15] A. Gajjar, Y. Zhang, and X. Yang, "Demo abstract: A smart building system integrated with an edge computing algorithm and IoT mesh networks," in *Proc. Second ACM/IEEE Symposium on Edge Computing*, no. 35, Oct. 2017, pp. 1-2.
[16] X. Yang and X. He, "Demo abstract: Establishing a BLE mesh network with fabricated CSRmesh devices," in *Proc. Second ACM/IEEE Symposium on Edge Computing* (SEC2017), Oct. 2017.
[17] JM1L2S Hardware Specification. (Aug. 2015). [Online]. Available: http://business.aplix.co.jp/en/product/jm1l2s/
[18] Bluetooth Specification Version 5.0. (Dec. 2016). [Online]. Available: https://www.bluetooth.com/specification/bluetooth-core-specification /bluetooth5.



Prof. Xiaokun Yang received his Ph.D. from the Department of Electrical and Computer Engineering (ECE), Florida International University (FIU), USA in Spring 2016, his dual M.S. from the Department of ECE at FIU and the Department of Software Engineering at Beihang University, China in 2007. He is currently an Assistant Professor at the College of Science and Engineering, University of Houston, Clear Lake.

From 2007 to 2012, he has also worked as a Senior ASIC Design/Layout Engineer at Advanced Micro Devices (AMD), China Electronic Corporation (CEC), and PowerLayer MicroSystems (PLM). His chip tape-out experiences include AMD CPUs/APUs - IPs (Kabini, Kaveri, Bonaire, Kryptos, and Samara), 802.11 a/b/g/n MIMO mixed-signal SoC - BBP (CEC TL3 and TL5), and PLM high-definition TV (HDTV) SoCs – CPU/DMA/DDRC/Interface (PLM3K and PLM5K).

Prof. Lei Wu received his Ph.D. from the Department of Computer Science, University of Montreal, Canada in 2005, and his M.S. in Computer Science from Huazhong University of Science and Technology, China, in 1999.

He is currently an Associate Professor at University of Houston-Clear Lake. His research interests focus on Artificial Intelligence, Robotics, Serious Game, IoT and the applications in health domain.

Mr. Xianlong He is the vice president of Bluefont Ltd., Hong Kong, China. His research interests include cyber-physical system design, sensor network, and IoT technology.

Mr. Archit Gajjar received his B.Tech. from Dhirubhai Ambani Institute of Information and Communication Technology (DA-IICT) in 2016, and currently he is a final year M.S. candidate at University of Houston, Clear Lake (UHCL).

His research interests include ASIC/FPGA design, Image/Video processing, and approximate computing on system-on-chip.

Prof. Yi Feng received the M.E. degree in Computer Engineering from the Chinese Academy of Sciences, China, in 1997, and the Ph.D. degree from the Universite de Montreal, Canada, in 2002. She is currently an associate professor at Algoma University, Sault Ste. Marie, ON. Her research interests include neural networks, formal verification and signal processing.